Information Technology and Management
https://doi.org/10.1007/s10799-021-00355-3

Multicriteria decision making taxonomy of code recommendation system challenges: a fuzzy-AHP analysis

Muhammad Azeem Akbar[1] · **Arif Ali Khan**[2,3] · **Zhiqiu Huang**[4]

Accepted: 29 December 2021

Abstract
The recommendation systems plays an important role in today's life as it assist in reliable selection of common utilities. The code recommendation system is being used by the code databases (GitHub, source frog etc.) aiming to recommend the more appropriate code to the users. There are several factors that could negatively impact the performance of code recommendation systems (CRS). This study aims to empirically explore the challenges that could have critical impact on the performance of the CRS. Using systematic literature review and questionnaire survey approaches, 19 challenges were identified. Secondly, the investigated challenges were further prioritized using fuzzy-AHP analysis. The identification of challenges, their categorization and the fuzzy-AHP analysis provides the prioritization-based taxonomy of explored challenges. The study findings will assist the real-world industry experts and to academic researchers to improve and develop the new techniques for the improvement of CRS.

Keywords Code recommendation system · Empirical investigations · Fuzzy-AHP

1 Introduction

In the modern world, the recommendation system has become an indispensable part, especially fore-commerce, medical, and social media systems. The recommendation systems provide suggestions and recommendation based on the user interests. Recommendation system usually use data sources to develop the system components and train for making appropriate decisions. In digital business world the recommender system could predict whether a particular user would prefer an item or not based on the user's profile. Recommender systems are beneficial to both service providers and users [20]. They minimize transaction expenses of finding and selecting items in an online shopping environment [20].

In this paper we have considered the following formal definitions of recommendation system: Ricci et al. [44], defines that "A recommender system or a recommendation system is a subclass of information filtering system that seeks to predict the rating, suitability or preference a user would give to an item." Hossain et al.[20] stated that "A Recommender System refers to a system that is capable of predicting the future preference of a set of items for a user and recommends the top items." Moreover, Sridevi et al. [50] underlined that "A recommendation engine filters the data using different algorithms and recommends the most relevant items to users."

The current era is considered the revolution period in the field of artificial intelligence (AI). "To train the AI models, the selection of efficient source code is a critical phase [7]. Most of the practitioners use the source code which is available on public sources, e.g., GitHub, source frog, etc. and the recommender system plays an important role in the selection of appropriated source code [7]. A code recommender system refers to a system that uses the code sources (e.g., Github, source frog) and recommends the most suitable source code to the developers and researchers. While selection the code, the affectability, and reliability of the code

✉ Arif Ali Khan
arif.khan@oulu.fi

[1] Software Engineering Department, Lappeenranta-Lahti University of Technology, 53851 Lappeenranta, Finland

[2] Faculty of Information Technology, University of Jyvaskyla, Jyvaskyla, Finland

[3] Present Address: M3S Empirical Software Engineering Research Unit, University of Oulu, 90014 Oulu, Finland

[4] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211100, China

Published online: 14 February 2022

are very important [34]. There are critical concerns related to code recommender system i.e. code analysis concerning the code quality of and code implementation capability [34]. Mark et al. [18] underlined the significance of the code analysis before its preprocessing for feature extraction and implementation. Moreover, Gregorio et al. [45] highlighted the importance of analyzing the complexity, code size, and the available resources for code implementations. Yamashita and Moonen [55] emphasized the compatibility of the development environment capacity and for the selected source code. They further mention that to get successful results; the code should be executed in a compatible development environment."

The importance of CRS in recent era motivated to conduct a comprehensive empirical study to explore the key challenges which might hinder the performance of CRS. The objective of the study consists of two main fold: (1) to explore the CRS challenges from the literature and verify them with industry practices; (2) to prioritize the investigated challenges concerning to their significance for CRS using fuzzy-AHP. The results and analysis of this study will provide the prioritization-based taxonomy of the investigated challenges. We believe that the deep analysis of CRS challenges will assists the industry experts and researcher to entertain the most priority challenges and develop the new techniques for the improvement in code recommendation systems. Following are the research questions develop to achieve the key objectives of this study:

[RQ1] What challenges of code recommendation system are discussed in the existing literature?
[RQ2] What is the real-world significance of the code recommendation system challenges?
[RQ3] What would be the prioritization-based taxonomy of investigated challenging?

2 Related work

The recommendation is helpful to "decide if there is no prior experience or knowledge about a particular matter. This is the technological revolutionized era, and the peoples believe in the auto recommendation system [38]. Due to the higher acceptance level, the business industry motivated to automate its businesses with a strong and reliable recommendation system [24]. We found several studies conducted to improve the performance of recommendation systems, e.g., [16, 39]."

Like the other areas of life, "the recommender system also has significant importance in the selection of source code for the training of artificial intelligence systems. Code recommender system refers to a system that uses the source code sources (e.g., Github, source frog) and recommends the most suitable code to the developers and researchers [41]. Currently, the source codes related to the machine learning filed received much attention from the software industry and academic researcher's community. With the intersection of research areas, i.e., "software engineering," "programming languages," "machine learning" and "natural language processing," the various communities have been composed into the areas of "big code" or "code naturalness" with numerous significant outcomes [44]. Mostly, in the field of machine learning, the researcher needs large corpora of code to train the artificial intelligence model and to learn the probabilistically causes concerning coding practices at a large scale. The primary aim is to train and implement the trained model as a useful tool in the required area. However, besides the importance of source code, there is little research has been conducted to address the problem of code recommendation systems. Mens and Lozano [40] highlighted that the selection of reliable source code from the available sources is an important activity to get the fruitful results. Janjic et al. [22] also emphasized the importance of a reliable code recommendation system."

Considering the state of the art literature and to the best of our knowledge, little empirical research has been conducted to address and highlight the concerns of CRS systems. Though, we tried to fill this gap by conducting a comprehensive empirical study aiming to identify the key challenges that could hinder the performance of CRS. The systematic literature review (SLR) approach has been adopted to explore the existing literature studies and investigate the factors that could be critical challenges for CRS. The survey questionnaire method has been further used to evaluate the SLR results and encapsulate the perceptions of the field experts. The finally summarise list of the challenging factors is used to develop the prioritization taxonomy using the fuzzy AHP technique. Fuzzy AHP is widely used approach for multi criteria problems and has been used in different software engineering research projects [37, 43, 48, 49, 54]. For example, Khan and Shameem [29] used the fuzzy-AHP analysis to rank the success factors of software process improvement paradigm. Similarly, Shameem et al. [47] taxonomies the factors that could influence the agile processes in geographically distributed environment. Moreover, Akbar et al. [3] prioritize the DevOps challenging factors using fuzzy AHP. Based on the above discussion, we could justify the application of fuzzy AHP method for this research study.

3 Research methodology

To address the study objectives, three different steps were adopted. In first step, the systematic literature review was conducted to explore the challenges of CRS, reported by the

researchers. In second step, the finding of literature review were verified by conducting the questionnaire survey study with industry experts. In third step, the fuzzy-AHP was used to prioritize the identified list of challenges considering their criticality for CRS systems. All the adopted research methodology steps are presented in Fig. 1 and described in below section.

3.1 Systematic literature review (SLR)

The SLR is the most significant approach of identifying and interpreting the available research evidence, in formal manner, based on the developed research questions and protocols. In this study, we have performed the SLR study using the guidelines of [31]. The SLR steps are discussed in in below sections.

3.1.1 Review process planning

3.1.1.1 Research questions To identify the challenges related to CRS the following research question was developed:

[RQ1] What challenges of code recommendation system are reported in the literature?

3.1.1.2 Database selection The selection of appropriated data sources are critical for the collection of most potential literature relevant to the study proposed research the questions [12]. We have considered the following seven databases for data extraction considering recommendations provided by Chen et al. [12, 25, 42]:"IEEE Xplore (http://ieeexplore.ieee.org)", "ACM Digital Library (http://dl.acm.org)", "Springer Link (link.springer.com)", "Wiley Inter Science (www.wiley.com)", "Science Direct (http://www.sciencedirect.com)", "Google Scholar (scholar.google.com)", "IET-digital libraries (www.theiet.org)".

3.1.1.3 Search strings A search string is a combination of text, symbols, keywords and their alternatives used to extract the data from digital repositories [11, 15, 19, 23, 30, 42]. The Boolean "OR" and "AND" the selected keywords and their alternative were concatenated:

("barriers" OR "obstacles" OR "hurdles" OR "difficulties" OR "impediments" OR "hindrance" OR "challenges" OR "limitations") AND ("code recommendation systems" OR "code recommender systems" OR "code filtering systems").

3.1.1.4 Inclusion and exclusion criteria The inclusion criteria are the characteristics that must be included in study, while exclusion criteria are the characteristics to disqualify certain material from inclusion in the study. The same approach has been used in other studies of software engineering domain e.g. [42, 57] and [35].

Inclusion criteria:

(1) Studies published in conference proceedings, workshop, journal, and book chapters.
(2) Selected literature should be in English.
(3) Articles whose findings directly related with the objective of this study.
(4) Most recent article will be considered if two or more studies are of similar nature or from same research project.

Exclusion criteria:

(1) Articles out of the CRS scope.
(2) Studies that did not provide the detail discussion of CRS.
(3) Studies that have not focused on CRS challenging factors.

3.1.1.5 Study quality assessment (QA) To exemplify the degree of conformity of primary selected studies QA is performed. The checklist questions and Likert scale used for

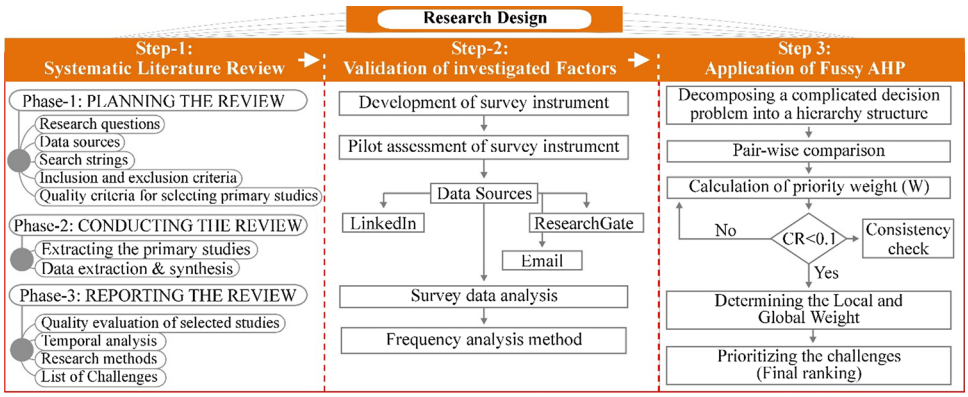

Fig. 1 Proposed research design

Table 1 Selected studies quality assessment criteria

S. no	Checklist questions	Likert scale
QA1	Does the selected study explore CRS challenging factors?	"Yes = 1, partial = 0.5, no = 0"
QA2	Does the selected study results are relevant to the research questions?	"Yes = 1, partial = 0.5, no = 0"
QA3	Does the selected study conducted empirical investigation to conclude their research findings?	"Yes = 1, partial = 0.5, no = 0"
QA4	Is any CRS standard or framework has been discussed in the selected primary study?	"Yes = 1, partial = 0.5, no = 0"
QA5	Does the CRS challenging factors are explicitly discussed?	"Yes = 1, partial = 0.5, no = 0"

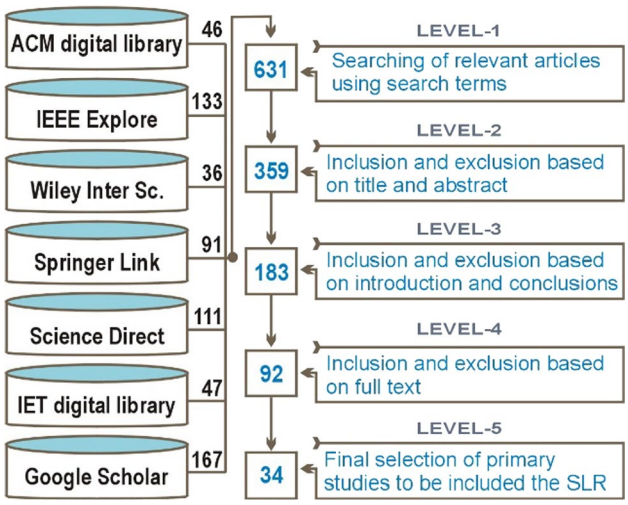

Fig. 2 Selection of primary studies for data extraction

QA are given in Table 1. The objective of QA is to check the suitability and appropriability of the selected literature concerning to address the research questions of this paper.

3.1.2 Conducting the review

Study selection and data extraction process

Total 631 studies were extracted from the selected repositories (Sect. 3.1.1.2) using the search strings discussed in (Sect. 3.1.1.3) and the given inclusion/exclusion criteria (Sect. 3.1.1.4). We further use tollgate approach [1] to further refine the selected studies and identify the most relevant primary studies. The tollgate approach consists of five phases. Figure 2 highlight the steps and flow of the tollgate approach used for this study. We finally shortlist total 34 primary studies after performing the five phase process of the tollgate approach. Each selected study is tagged as "ST" to differentiate it with other references. All the selected studies are given in Appendix A.

The data were extracted from the finally selected 34 primary studies that were synthesized by all the authors of the study. The first and third author thoroughly reviewed the selected studies and extracted the most relevant material (barriers, titles, and publication year). The review process of the extracted data has been conducted by the second author. He thoroughly reviewed the findings and identify if there are any inconsistences and incompleteness. The finally identified themes, concepts and statements were carefully reviewed and classified into 19 compact statement of challenges.

Moreover, the research biasness for the data extraction process has been removed by performing the inter-rater reliability analysis using the "non-parametric Kendall's coefficient of concordance" (W) test [1]. Five external experts were requested to participate in the inter-rater reliability analysis. The two experts were from King Saud University Saudi, one from NetSole Pakistan and two from ST Tech Sweden. The experts were requested to randomly select ten studies from the selected 34 primary studies. They were asked to perform the step-by-step data extraction process to measure the research biasness between the authors and the invited experts. Finally, the "non-parametric Kendall's coefficient of concordance" (W) value was calculated to compare the findings of the authors and the external experts. The value of W ranges from 0 to 1, where 0 show no agreement between the findings and 1 is the strong positive agreement [33]. In this study, the given results ("$W = 0.88$, $p = 0.003$") indicate that there is positive agreement between the data extraction process of the authors and the external experts. It justified that the extracted SLR findings are reliable, and we could use it for further analysis and discussions.

3.1.3 Reporting the review

Selected primary studies quality assessment

Quality assessment (QA) was performed to measure the quality of the selected literature and their association to the research questions. The QA criteria have been developed (Sect. 3.1.1.5) to evaluate the quality of each study. The QA results obtained based on the given criteria are provided in Appendix A. It is noted that 84% of the selected score $\geq 80\%$, which justify that the data extracted from the selected primary studies are suitable and appropriate to answer the research questions. The threshold value for the QA score is 50% which is adopted from different other SLR studies published in the domain of software engineering [2, 21].

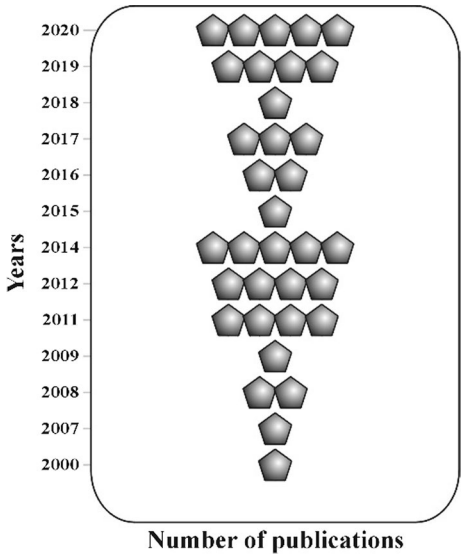

Fig. 3 Publication years of selected studies

Temporal analysis

The temporal distribution of primary studies is given in Fig. 3. The publication years of the primary studies range from 2000 to 2020, which indicated that how the research consistently working on code recommendation systems area. According to the Fig. 3, the frequency of per year publication is higher in 2020 which is 5 paper, this indicated that the code recommendation system is currently an active research area.

3.2 Empirical study

3.2.1 Development of survey instrument

The questionnaire survey was developed to collect opinion of experts related to the challenges of coding schemes in software development. The designed question consists of bibliographic detail i.e. organization type, designation, gender, work experience etc. We added open-ended section in the questionnaire survey to elicit any additional challenges from the experts, which are not mentioned in questionnaire. To mark the importance of each challenge Likert scale was also provided in the questionnaire. The five scale Likert scale is "strongly agree, agree, neutral, disagree, and strongly disagree". It is significant to consider neutral option that will give neutral space to those survey participants who are not sure about the criticality of a specific challenging factor [17].

3.2.2 Pilot assessment of survey instrument

The pilot assessment in an effective approach to improve the quality of the survey instrument. Lewis-Beck et al. [36] stated the importance of pilot assessment as "a clear and complete survey instrument is significant to collect appropriate responses". The survey instrument (questionnaire) assessment has been conducted based on the expert's opinion. Four qualitative software engineering experts from Western University Canada, City University of Hong Kong and Wuhan University, China were invited to assess the appropriate-ability survey instrument. The experts reviewed the questionnaire and provide suggestions and recommendations based on their experience and domain knowledge. The experts suggested to improve the presentation of questionnaire. They recommended to present the survey variables in tabular form rather than plain questions which are more confusing. Moreover, they point out multiple grammatical and spelling mistakes. We finally incorporated all the suggestions, and the updated survey questionnaire is provided in Appendix B.

3.2.3 Data sources

The data sampling and data collection are the key phases of questionnaire survey studies. Snowball technique was adopted to develop the data sample from the targeted population [33]. Snowballing is more appropriate and effective way to approach the large targeted population [6, 46]. The targeted population were approached using the professional social media networks including ResearchGate, LinkedIn and WeChat. Moreover, the authors used personal industrial contacts to distribute the questionnaire with the practitioners and further requested to share with others [27, 28, 42, 48]. The data were collected from 87 survey participants during the time period February 2020 to April 2020.The survey responses were manually analysed by the first and third authors and excluded the ten incomplete responses. The final 77 responses were used for further data analysis process. The details bibliographic data of the survey respondents are provided in Appendix-C.

3.2.4 Survey data analysis

The collected responses were analyzed using frequency analysis method as it is consider the most appropriate technique for ordinal and nominal data type [28, 32]. Different other researchers used the same data analysis approach for similar nature of studies [33].

3.3 Fuzzy set theory and AHP

A fuzzy-AHP was used to rank the list of investigated challenges with respect to CRS. This section provides detail discussion regarding the key perceptions of fuzzy set theory and AHP approach.

3.3.1 Fuzzy set

Zadeh et al. [56] introduce fuzzy set theory to deal with vagueness and uncertainties of problems in real world. It also manages to control ambiguities while making group decisions. The main contribution of fuzzy set theory to represent the vague data [52]. A membership function in fuzzy setis characterized to map the objects between '0' and '1'. The detail and definition about fuzzy set theory is given below.

Definition A triangular fuzzy number (TFN) F is denoted by a set (f^l,f^m,f^u), as shown in Fig. 4. The given Eq. (1) defines the membership function $\mu_F(x)$ of F.

$$\mu_F(x) = \begin{cases} \frac{x-f^l}{f^m-f^l}, & f^l \le x \le f^m \\ \frac{f^u-x}{f^u-f^m}, & f^m \le x \le f^u \\ 0, & Otherwise \end{cases} \quad (1)$$

where, f^l,f^m and f^u presents the crisp numbers that shows the lowest, most significant and high possible values respectively.

The algebraic operations for the two triangular fuzzy numbers (TFNs) i.e. $\check{T}_1$, $\check{T}_2$ are given in Table 2.

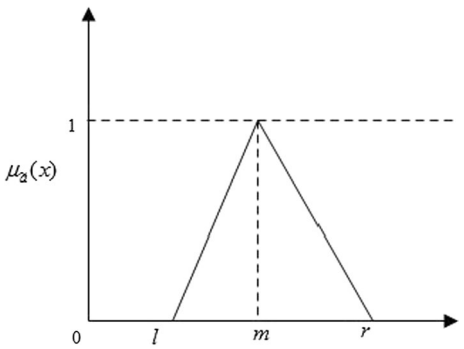

Fig. 4 Triangular fuzzy number

3.3.2 Fuzzy AHP

AHP is commonly used technique for decision making in "multi-criteria decision-making" (MCDM) problems. The AHP takes the pair-wise comparison of all alternatives with respect to selected criteria. It provides decision support tool to handle multi-criteria decision-making problems. AHP is the most widely used technique for quantitative and qualitative MCDM problems. Following are the key phases of AHP process:

Phase1: "Decompose the complex decision problem into the hierarchical structure" (Fig. 5)
Phase2: "Calculate priority vector at each level of hierarchy with the help of pair-wise comparison."
Phase3: "Compute the consistency ratio of the pairwise comparison."
Phase4: "Calculate the final priority weight for the factors and the sub-factors" (Fig. 5).

However, common AHP approach has some limitations that could be covered by combining both AHP and fuzzy set theory i.e., "crisp environment, and absence of uncertainty, judgmental scale is unbalanced, selection of judgment is subjective". Therefore, fuzzy analytical hieratical process FAHP is the most popular approach to deal with fuzziness and uncertainties in MCDM problems [5]. The FAHP deals with multiple decision-makers to capture data by handling the linguistic terms. These linguistic variables were transformed into numerical form by using TFNs scale. This approach has been considered in other engineering fields to measure vagueness of fuzzy environment [9]. In this study the fuzzy AHP approach proposed by Chang [48] has been applied which is more appropriate and consistent.

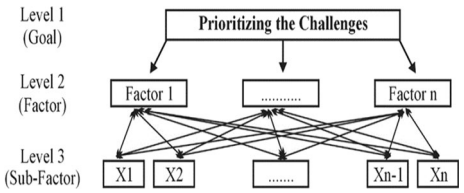

Fig. 5 Fuzzy AHP decision hierarchy

Table 2 Triangular fuzzy numbers

Operation law	Expression
Addition ($F_1 \oplus F_2$)	$(f^l_1, f^m_1, f^u_1)_{\oplus}(f^l_2, f^m_2, f^u_2) = (f^l_1 + f^l_2, f^m_1 + f^m_2, f^u_1 + f^u_2)$
Subtraction ($F_1 \oplus F_2$)	$(f^l_1, f^m_1, f^u_1)_{\oplus}(f^l_2, f^m_2, f^u_2) = (f^l_1 - f^l_2, f^m_1 — f^m_2, f^u_1 - f^u_2)$
Multiplication ($F_1 \oplus F_2$)	$(f^l_1, f^m_1, f^u_1)_{\oplus}(f^l_2, f^m_2, f^u_2) = (f^l_1 * f^l_2, f^m_1 * f^m_2, f^u_1 * f^u_2)$
Division ($F_1 \oplus F_2$)	$(f^l_1, f^m_1, f^u_1)_{\oplus}(f^l_2, f^m_2, f^u_2) = (f^l_1 / f^l_2, f^m_1 / f^m_2, f^u_1 / f^u_2)$
Inverse ($F_1 \oplus F_2$)	$(f^l_1, f^m_1, f^u_1)^{-1} = (1/f^l_1, 1/f^m_1, 1/f^u_1)$
For any real number k (kF_1)	$k(f^l_1, f^m_1, f^u_1) = k f^l_1, k f^m_1, k f^u_1$

In a prioritization problem, let $X = \{x_1, x_2, \ldots, x_n\}$ indicated the challenges of main categories as an object set and $U = \{u_1, u_2, \ldots, u_n\}$ indicated the each challenge of every category as a goal set. According to [10] approach, every object is measured, and the level of analysis of each goal (gi) is executed. Hence, for object, there are (m) extent analysis values that can be determined using Eqs. (2) and (3):

$$F_{gi}^1, F_{gi}^2, \ldots, F_{gi}^m, \tag{2}$$

$$i = 1, 2, \ldots, n \tag{3}$$

where, all $F^j_{gi,}$ (j = 1, 2, …, m) are indicated the TRNs.

To perform the Chang's extent analysis approach [48] the used steps are presented below:

Step 1: The Eq. 4 is used to define the ith object of a fuzzy synthetic extent as:

$$S_i = \sum_{j=1}^{m} F_{gi}^j \otimes \left[\sum_{i=1}^{n}\sum_{j=1}^{m} F_{gi}^j\right]^{-1} \tag{4}$$

To achieve the expression $\sum_{j=1}^{m} F_{gi}^j$, "execute the fuzzy addition operation of m extent analysis such as:"

$$\sum_{j=1}^{m} F_{gi}^j = \left(\sum_{j=1}^{m} f_{gi}^l, \sum_{j=1}^{m} f_{gi}^m, \sum_{j=1}^{m} f_{gi}^u\right) \tag{5}$$

and to achieve the expression $\left[\sum_{i=1}^{n}\sum_{j=1}^{m} F_{gi}^j\right]^{-1}$, "the fuzzy addition operation is executed on" $F_{gi}^j (j = 1, 2, \ldots..m)$ value, as follow:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} F_{gi}^j = \left(\sum_{i=1}^{n} f_i^l, \sum_{i=1}^{n} f_i^m, \sum_{i=1}^{n} f_i^u\right) \tag{6}$$

and finally, calculate the inverse of the vector with the help of Eq. (7):

$$\left[\sum_{i=1}^{n}\sum_{j=1}^{m} F_{gi}^j\right]^{-1} = \left(\frac{1}{\sum_{i=1}^{n} f_i^u}, \frac{1}{\sum_{i=1}^{n} f_i^m}, \frac{1}{\sum_{i=1}^{n} f_i^l}\right) \tag{7}$$

Step 2: "As F_a and F_b are two triangular fuzzy number then the degree of possibility of" $F_a = (f^l_{a,}\ f^m_{a,}\ f^u_a) \geq F_b = (f^l_{b,}\ f^m_{b,}\ f^u_b)$ is defined as follows. The Eq. 8 is also given below:

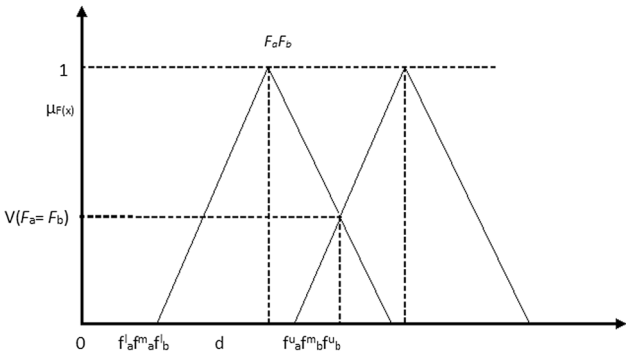

Fig. 6 Triangular Fuzzy number

$$V(F_a \geq F_b) = \sup[\min(\mu_{Fa}(x), (\mu_{Fb}(x))] \tag{8}$$

$$V(F_a \geq F_b) = hgt(F_a \cap F_b) = \mu_{F_a}(d) = \begin{Bmatrix} 1 & if\ f_a^m \geq f_b^m \\ \frac{f_a^u - f_b^l}{(f_a^u - f_a^m) + (f_b^m - f_b^l)} & f_b^l \leq f_a^u \\ 0 & Otherwise \end{Bmatrix} \tag{9}$$

In this context, the value of d indicates ordinate of the highest intersection point between D, μ_{Fa} and μ_{Fb} (Fig. 6). The values of V_1 ($F_a \geq F_b$) and V_2 ($F_a \geq F_b$) are required for determining the value of P_1 and P_2.

Step 3: Determining the complete "degree of possibility of a convex fuzzy number and the other convex fuzzy" numbers F_i ($i = 1, 2, \ldots, k$) can be defined as follow:

$$V(F \geq F_1, F_2, F_3 \ldots F_k) = \min V(F \geq F_i) \tag{10}$$

Assuming that,

$$d'(F_i) = \min V(F_i \geq F_k) \tag{11}$$

for $k = 1,2,\ldots,n; k \neq i$.

The Eq. 12 is used to determine the weight vector.

$$W' = (d'(F_1), d'(F_2), d'(F_3), \ldots..d'(F_n)) \tag{12}$$

where, F_i ($i = 1,2,\ldots,n$) are definite variables.

Step 4: Equation 13 shows the normalised values of weight vector in Eq. 12. The normalised non-fuzzy valueis considered the priority weight for each challenging factor.

$$W = (d(F_1), d(F_2), d(F_3), \ldots..d(F_n)) \tag{13}$$

Table 3 Random consistency index (RI) with respect to matrix size

Matrix size	1	2	3	4	5	6	7	8	9	10
RI	0	0	0.58	0.9	1.12	1.24	1.32	1.41	1.45	1.49

where W is the priority weight of a specific factor.

Step 5: Consistency check: It is mandatory that the fuzzy AHP pairwise comparison matrixes should be consistent [51]. Therefore, it is significant to determine the consistency ratio (CR) of each matrix. The given matrices are diffuzified using the graded mean technique. For example, the Eq. 14 is used to difuzify the triangular fuzzy matrix P = (l, m, u):

$$P_{crisp} = \frac{(4m + l + u)}{6} \tag{14}$$

The final consistency ratio could be determine using Eqs. 15 and 16:

$$CI = \frac{\lambda_{\max} - n}{n - 1} \tag{15}$$

$$CR = \frac{CI}{RI} \tag{16}$$

where, λ_{max}: "the largest eigenvalue of the comparison matrix", n: "the number of items being compared in the matrix and", RI: "the random index and its value can be opted from Table 3". CI: "the consistency index, which could be calculated using Eq. 15".

If the calculated of CR is less than 0.1, it renders that the matrix is consistent, else the matrix is not consistent and there is need the decision makers gain conduct pairwise judgements.

4 The results and analysis

The result and discussion of this study are provided in this section.

4.1 Investigations of SLR study

By carefully reviewing the collected studies, we have extracted a total of 19 challenges (Table 4). The ultimate aim of this study is to prioritize the investigated challenges with respect to their significant for code recommendation system. Though, the identified list of challenges were further mapped into three core categories (i.e., "Human resources", "Process" and "Technology") (Table 4). The identified challenges were categorized to develop the hierarchy structure which is needed for fuzzy-AHP analysis. To do this, the coding scheme of ground theory technique [14] was used in categorization process.

4.2 Findings of empirical study

We identified total 19 challenges during the SLR study that were further informally classified into three different categories. The survey study was conducted to know the opinions and perceptions of experts regarding the SLR findings and formally validate the categorical classification of the challenges.

The collected survey responses were classified in positive, negative and neutral categories (Table 5), where positive category results show the percentage of survey participants who were positively agree with the SLR findings. Similarly, the negative category results illustrate the percentage of those respondents who were disagree with SLR results and the challenges classification. The natural category gives neutral option to those participants who were not sure about the impact of investigated challenge on code recommendation system.

The summarized results of empirical study are presented in Table 5 that shows majority of the survey respondents are agree as the investigated challenges could have negative impact on code recommendation systems. It is noticed that the results of challenges mention in positive category are greater than 60%. We also noted that the respondents are strongly agree with the categorization of the investigated challenges (Table 5). Moreover, open ended questions were also added in the survey questionnaire in order to note additional novel challenges of code recommendation systems. However, the survey participants have not reported any new challenge, therefore we finalize the list of the identified 19 challenges for fuzzy AHP based prioritization. Different other studies also use the same data analysis approach [29].

4.3 Fuzzy-AHP analysis

This section consists of fuzzy AHP results and analysis. All the steps of fuzzy-AHP were carefully performed to determine the weight of challenges within the category and for among all the categories. The fuzzy-AHP was performed using "MATLAB R2016b programming environment" which has been executed personal computer with the specification of "Intel Corei3 3.5-GHz processor and 8-GB memory". All the steps of fuzzy-AHP are performed and their implications are presented in this section.

Table 4 List of explored challenges

Categorise	IDs	Challenges	ID (N = 34)
Human resources	CH1	"Lack of software engineering knowledge"	ST3, ST5, ST17, ST20, ST28, ST31
	CH2	"Lack of monitoring and management"	ST5, ST8, ST9, ST11, ST14, ST17, ST21, ST23, ST26, ST30, ST31
	CH3	"Possible solutions have to be compared."	ST7, ST12, ST17, ST24, ST27
	CH4	"Lack of multiple facts adoption"	ST2, ST7, ST13, ST26, ST29, ST34
Process	CH5	"All dependence needs to track down."	ST1, ST5, ST11, ST14, ST20, ST22
	CH6	"Lack of data quality analysis."	ST4, ST7, ST10, ST13, ST15, ST19, ST20, ST24, ST27, ST28, ST30, ST34
	CH7	"Need for clone removal."	ST4, ST9, ST14, ST18, ST22, ST32,
	CH8	"Ad-hoc approach"	ST5, ST15, ST16, ST21, ST25, ST32
	CH9	"Search issue due to big size"	ST5, ST10, ST18, ST23, ST25, ST29
	CH10	"Lack of proper documentation"	ST4, ST5, ST9, ST10, ST13, ST16, ST18, ST19, ST20, ST27, ST28
	CH11	"Perform similar tasks"	ST6, ST10, ST15, ST23, ST30
Technology	CH12	"Lack of data implementation analysis."	ST2, ST5, ST10, ST13, ST16, ST17, ST20,, ST25, ST26, ST27, ST30, ST33
	CH13	"Architecturally-relevant code anomalies are not detected soon enough."	ST3, ST7, ST8, ST14, ST16, ST22, ST24, ST26, ST30, ST33
	CH14	"Rapidly occurrence of changes."	ST6, ST10, ST12, ST14, ST16, ST19, ST21, ST22, ST25, ST30
	CH15	"Change traceability issues"	ST1, ST4, ST12, ST26, ST33, ST34
	CH16	"Poor functions design with respect to reusability"	ST5, ST10, ST16, ST22, ST30
	CH17	"Exception handling is not a primary focus,"	ST4, ST17, ST18, ST20, ST24, ST31
	CH18	"Missing of resalable components"	ST1, ST3, ST8, ST11, ST12, ST14, ST17, ST19, ST24, ST26, ST29, ST31
	CH19	"Lack of proper assistance by existing tools"	ST2, ST5, ST9, ST14, ST19, ST24, ST27, ST31

Step-1: (Decomposing a problem into hierarchy structure)

In this step, we develop a hierarchy structure of complication decision making problem [4, 48]. The hierarchy structure was classified at three levels as presented in Figure as presented in Fig. 5. The key problem is presented at level-1, the categories and their respective challenges are presented at level-2 and 3. The proposed hierarchy structure is shown in Fig. 7.

Step-2: Pairwise comparison

The objective of fuzzy AHP analysis is to rank the identified list of challenges considering their importance for CRS. To do this, the pairwise comparison was conducted to prioritize the invested challenges and their respective categories. The pairwise comparison was conducted based on the expert's opinions. To collect the responses of experts, we have conducted fuzzy-AHP survey. We developed a questionnaire survey and approach to the participant of first survey study and out of 77 participants only 28 were agreed to participate in fuzzy-AHP survey. The developed questionnaire is given in appendix-C. Therefore, during data collection process, we have collected a total of 28 complete response from the experts. The collected responses were manually check to find the inconsistencies and uncompleted entries. We found that all the 28 response were complete, and we considered them to develop the pairwise comparison matrixes. The data size of 28 studies might not strong enough to generalize the findings of fuzzy AHP. Though, considering the existing studies [13, 48, 53], the collected 28 response are justified for the generalization of results.

To transform the survey (expert's judgements) into TFN numbers, the geometric mean was calculated. The following formula of geometric mean was considered:

$$\text{Geometric mean} = \text{n}\sqrt{\text{v1x v2} \times \text{v3} \ldots \ldots \ldots \text{.vn}}$$

where t = "Weight of each response", n = "Number of responses".

We have used the Linguistic variable corresponding to the fuzzy triangular values as given in Table 6. The triangular fuzzy matrix developed by [8] was used in the development of pairwise comparison matrixes.

Step-3: Consistency evaluation

The steps adopted to determine the consistency check are presented in this section. To interpret these steps, we have used the matrix of core categories of the challenges (Table 7). We have defuzzified to crisp number of pairwise comparisons of main categories of the challenges using

Table 5 Empirical assessment of identified challenges

S. no	Challenge factors	No. of empirical investigation (n = 77)							
		Positive			Negative			Neutral	
		SA	A	%	SD	D	%	N	%
C1	*Human resources*	28	44	94	1	2	4	2	3
CH1	"Lack of software engineering knowledge"	26	36	81	5	7	16	3	4
CH2	"Lack of monitoring and management"	21	31	68	6	8	18	11	14
CH3	"Possible solutions have to be compared"	17	37	70	4	9	17	10	13
CH4	"Lack of multiple facts adoption"	19	39	75	5	6	14	8	10
C2	*Process*	32	38	91	2	4	9	0	-
CH5	"All dependence needs to track down"	21	35	73	4	7	14	10	13
CH6	"Lack of data quality analysis"	20	29	64	5	9	16	16	21
CH7	"Need for clone removal"	23	36	77	4	7	14	7	9
CH8	"Ad-hoc approach"	24	31	71	4	7	14	11	14
CH9	"Search issue due to big size"	21	40	79	3	7	13	6	8
CH10	"Lack of proper documentation"	18	34	68	5	12	22	8	10
CH11	"Perform similar tasks"	25	29	70	6	9	19	8	10
C3	*Technology*	31	40	92	1	4	6	1	1
CH12	"Lack of data implementation analysis"	28	38	86	2	5	9	3	5
CH13	"Architecturally-relevant code anomalies are not detected soon enough"	22	35	74	5	7	16	8	10
CH14	"Rapidly occurrence of changes."	25	36	79	4	6	13	6	8
CH15	"Change traceability issues"	21	41	81	3	7	13	5	6
CH16	"Poor functions design with respect to reusability"	24	29	68	6	8	16	13	16
CH17	"Exception handling is not a primary focus"	21	28	64	7	9	21	12	16
CH18	"Missing of resalable components"	17	32	64	5	7	16	16	21
CH19	"Lack of proper assistance by existing tools"	21	39	78	3	9	16	5	6

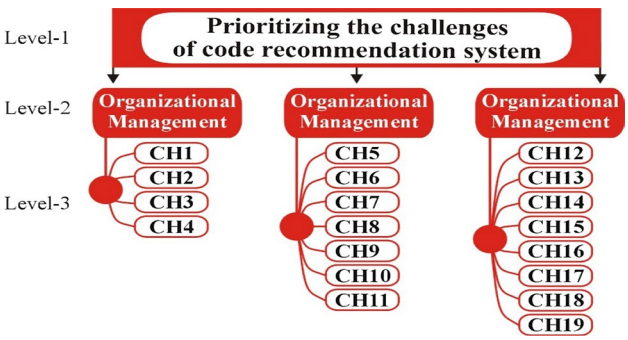

Fig. 7 Hierarchy structure of the investigated challenges

Eq. 14, and get the corresponding Fuzzy Crisp Matrix (FCM) as presented in Table 7:

Step-4: Identified challenges and their categories local priority weight

i. A numerical example

The priority vector for each category is given in Table 7. The priority weight for each challenging factor and the category was determined using Eq. 3.

Firstly, "the synthetic extent values of three categories" (human resources, process and technology) were calculated, and finally applied the Eq. 4 to calculate the priority weight of

Table 6 "Conversion scale of triangular fuzzy numbers" [8]

"Linguistic scale"	"Triangular fuzzy scale"	"Triangular fuzzy reciprocal scale"
"Just equal (JE)"	"(1, 1, 1)"	"(1, 1, 1)"
"Equally important (EI)"	"(0.5, 1, 1.5)"	"(0.6, 1, 2)"
"Weakly important (WI)"	"(1, 1.5, 2)"	"(0.5, 0.6, 1)"
"Strongly more important (SMI)"	"(1.5, 2, 2.5)"	"(0.4, 0.5, 0.6)"
"Very strongly more important (VSMI)"	"(2, 0.5, 3)"	"(0.3, 0.4, 0.5)"
"Absolutely more important (AMI)"	"(2.5, 3, 3.5)"	"(0.2, 0.3, 0.4)"

Table 7 "Pairwise comparison of challenges categories

	Human resource	Process	Technology
Human resources	(1,1,1)	(1.5, 2.5, 3)	(1, 1.5, 2)
Process	(0.3, 0.4, 0.6)	(1,1,1)	(0.4, 0.5, 0.6)
Technology	(0.5, 0.6, 1)	(1.5, 2, 2.5)	(1,1,1)

Table 8 Results of V values for criteria

	"HR"	"P"	"T"	"d (Priority Weight)"
"V (HR ≥ ….)"	–	1	1	1
"V (P ≥ ….)"	0.030016	–	0.26504	0.030016
"V (T ≥ ….)"	0.69837	1	–	0.69837

the given categories. Following are the calculations to determine the priority weight of the categories.

$$\sum_{i}^{n}\sum_{j}^{m} F_{gi}^{j} = (1,1,1) + (1.5,2,2.5) + (1,1.5,2)... + (0.5,0.6,1) + (1,1,1) = (14.1, 18.2, 22.8)$$

$$\left[\sum_{i}^{n}\sum_{j}^{m} F_{gi}^{j}\right]^{-1} = \left(\frac{1}{22.8}, \frac{1}{18.2}, \frac{1}{14.1}\right) = (0.04386,\ 0.054945,\ 0.070922)$$

$$\sum_{j=1}^{m} F_{g1}^{j} = (1,1,1) + (1.5,2.5,3) + (1,1.5,2) = (5,7,8.5)$$

$$\sum_{j=1}^{m} F_{g2}^{j} = (0.3,0.4,0.6) + (1,1,1) + (0.4,0.5,0.6) = (2.2,2.5,3.2)$$

$$\sum_{j=1}^{m} F_{g3}^{j} = (0.5,0.6,1) + (1.5,2,2.5) + (1,1,1) = (4,5.1,6.5)$$

Equation 4 is used to calculate the synthesis values of "Human resources (HR)", "Process (P)", and "Technology (T)" categories:

$$HR = \sum_{j}^{m} F_{g1}^{j} \otimes \left[\sum_{i}^{n}\sum_{j}^{m} F_{gi}^{j}\right]^{-1}$$

$$= (5,7,8.5) \otimes (0.04386,\ 0.054945) = (0.219298,\ 0.384615)$$

$$P = (2.2,2.5,3.2) \otimes (0.04386,\ 0.054945) = (0.096491,\ 0.137363)$$

$$T = (4,5.1,6.5) \otimes (0.04386,\ 0.054945) = (0.175439,\ 0.280220)$$

Equation 6 is used to calculate the degree of possibility and the minimum degree of possibility is determined using Eq. 8, that specifically presents the priority weight of the categories.

Hence, we have calculated the weight vector as: $W' = (1, 0.030016, 0.69837)$ (Table 8). By normalizing the values the significance of attributes were determined as W = ("0.4789, 0.01435, 0.3337"). The determined results shows that human resources category is declared as the most important category to the investigated challenges as it gain the highest priority weights compared with other challenges categories.

Table 9 Fuzzy Crisp Matrix (FCM) for challenges categories

	HR	P	T
HR	1.7	2.3	1.4
P	0.9	1.0	0.7
T	0.8	2.0	1.0
Column sum	3.4	5.2	3.1

Table 10 Normalized matrix of challenges main categories

	HR	P	T	"Priority vector weight (W)"
HR	0.5	0.4423	0.4516	"0.37938"
P	0.2647	0.1923	0.2258	"0.14945"
T	0.2353	0.3846	0.3225	"0.27593"

ii. Consistency check

In fuzzy-AHP, every pairwise comparison matrix should be consistent. In order to present the consistency check procedure, an example of consistency check is presented using the core categories of challenges (Table 9). For consistency check, the Eq. 14 and the determined FCM table was used.

We further determine the column sum of each FCM matrix aiming to calculate largest Eigenvector (λ_{max}) value. Each value of FCM matrix is divided by its respective column sum to develop the normalised matrix (Table 10). Similarly, the priority weight of the categories is determined by taking average of their respective rows (Table 10).

Table 11 Pairwise comparison of 'human resource' category

C1				
	CH1	CH2	CH3	CH4
CH1	(1,1,1)	(0.3, 0.4, 0.5)	(0.4, 0.5, 0.6)	(1.5, 2, 2.5)
CH2	(2, 2.5, 3)	(1,1,1)	(2, 2.5, 3)	(0.5, 1, 1.5)
CH3	(1.5, 2, 2.5)	(0.3, 0.4, 0.5)	(1,1,1)	(2, 2.5, 3)
CH4	(0.4, 0.5, 0.6)	(0.6, 1, 2)	(0.3, 0.4, 0.5)	(1,1,1)

I_{max} = 4.50, CI = 0.12, CR = 0.10

$$\lambda_{\max} = \Sigma\left(\left[\Sigma Cj\right] \times \{W\}\right) \tag{17}$$

where, ΣCj = column sum [C] (Table 7), W = priority weight (Table 10), therefore λ_{max} = 3.6*0.37938 + 5.5*0.14945 + 3.2*0.27593 + = 3.0707.

The total number of elements are (n = 4), therefore the value of random index (CI) is 0.9 (Table 3). The final values of consistency index (CI) and consistency ration (CR) are respectively calculated using Eqs. 15 and 16

$$CI = \frac{\lambda_{\max} - n}{n-1} = \frac{3.0707 - 3}{3-1} = 0.035553 \tag{15}$$

Table 14 Pairwise comparison of in between the categories

	Human resources	Process	Technology
Human resources	(1,1,1)	(2, 2.5, 3)	(1, 1.5, 2)
Process	(0.3, 0.4, 0.5)	(1,1,1)	(0.3, 0.4, 0.5)
Technology	(0.5, 0.6, 1)	(2, 2.5, 3)	(1,1,1)

$$CR = \frac{CI}{RI} = \frac{0.035553}{0.58} = 0.061 \tag{16}$$

The value of CR is 0.061 which is < 0.10 and it is presented that the pairwise matrix of challenges categories is consistent. Using the same steps, the consistency for all the pairwise comparison matrixes are determined as given in Tables 11, 12, 13, 14 and 15, respectively.

Phase 5: Calculating the weight of challenges

The weights of the identified challenges were calculated aiming to determine their ranking with concerning to their significant for code recommendation systems. Hence, to calculate the rankings of the challenges, we have determined the local and global weights as presented in Table 15.

The local weights were calculated to determine the local ranking of each challenge. The local ranking renders

Table 12 Pairwise comparison of 'process' category

C2							
	CH5	CH6	CH7	CH8	CH9	CH10	CH11
CH5	(1,1,1)	(0.4, 0.5, 0.6)	(1.5, 2, 2.5)	(0.3, 0.4, 0.5)	(0.4, 0.5, 0.6)	(1, 1.5, 2)	(1, 1.5, 2)
CH6	(1.5, 2, 2.5)	(1,1,1)	(2, 2.5, 3)	(0.5, 1, 1.5)	(1, 1.5, 2)	(1, 1.5, 2)	(1, 1.5, 2)
CH7	(0.4, 0.5, 0.6)	(0.3, 0.4, 0.5)	(1,1,1)	(2, 2.5, 3)	(2.5, 3, 3.5)	(0.4, 0.5, 0.6)	(0.5, 0.6, 1)
CH8	(2, 2.5, 3)	(0.6, 1, 2)	(0.3, 0.4, 0.5)	(1,1,1)	(0.5, 0.6, 1)	(1, 1.5, 2)	(2, 2.5, 3)
CH9	(1.5, 2, 2.5)	(0.5, 0.6, 1)	(0.2, 0.3, 0.4)	(1, 1.5, 2)	(1,1,1)	(2, 2.5, 3)	(0.4, 0.5, 0.6)
CH10	(0.5, 0.6, 1)	(0.5, 0.6, 1)	(0.5, 0.6, 1)	(0.5, 0.6, 1)	(0.5, 0.6, 1)	(1,1,1)	(1, 1.5, 2)
CH11	(1, 1.5, 2)	(0.3, 0.4, 0.5)	(0.5, 0.6, 1)	(0.6, 1, 2)	(2.5, 3, 3.5)	(1.5, 2, 2.5)	(1,1,1)

I_{max} = 6.85, CI = 0.21, CR = 0.19

Table 13 Pairwise comparison of 'technology' category

C3								
	CH12	CH13	CH14	CH15	CH16	CH17	CH18	CH19
CH12	(1,1,1)	(1, 1.5, 2)	(1.5, 3, 2.5)	(0.6, 1.5, 2)	(2.5, 2, 3.5)	(0.3, 1.5, 0.5)	(0.5, 1.5, 1)	(1, 2.5, 2)
CH13	(0.5, 1, 1.5)	(1,1,1)	(0.2, 0.3, 0.4)	(1,1,1)	(0.6, 1, 2)	(2, 2.5, 3)	(0.5, 1, 1.5	(2, 2.5, 3)
CH14	(0.2, 0.3, 0.4)	(1, 1.5, 2)	(1,1,1)	(0.5, 1, 1.5)	(0.5, 0.6, 1)	(0.5, 0.6, 1)	(1, 1.5, 2)	(0.5, 0.6, 1)
CH15	(0.5, 0.6, 1)	(0.5, 0.6, 1)	(0.5, 0.6, 1)	(1,1,1)	(0.4, 0.5, 0.6)	(1, 1.5, 2)	(2, 2.5, 3)	(1, 1.5, 2)
CH16	(0.4, 0.5, 0.6)	(1.5, 2, 2.5)	(1, 1.5, 2)	(2.5, 3, 3.5)	(1,1,1)	(1, 1.5, 2)	(0.2, 0.3, 0.4)	(0.4, 0.5, 0.6)
CH17	(0.5, 0.6, 1)	(0.5, 0.6, 1)	(1, 1.5, 2)	(0.3, 0.4, 0.5)	(1.5, 2, 2.5)	(1,1,1)	(0.4, 0.5, 0.6)	(2, 2.5, 3)
CH18	(1, 2.5, 2)	(0.3, 0.4, 1.5)	(0.5, 0.5, 1)	(0.6, 1.5, 2)	(2.5, 2, 3.5)	(1.5, 1.5, 2.5)	(1,1,1)	(0.4, 1.5, 0.6)
CH19	(2, 1.5, 3)	(0.5, 1.5, 1)	(1, 2.5, 2)	(0.3, 2, 0.5)	(0.5, 1.5, 1	(0.3, 2, 0.5)	(1.5, 3, 2.5)	(1,1,1)

I_{max} = 10.4, CI = 0.17, CR = 0.11

Table 15 Local and global weights

Categories	"Categories weight (CW)"	"Challenges"	"Local weights (LW)"	"Local ranking (LR)"	"Global weights (GW)"	"Global ranking (GR)"
Human resources	0.37938	CH1	0.241	2	0.759	11
		CH2	0.180	3	1.518	5
		CH3	0.420	1	0.379	16
		CH4	0.160	4	1.897	3
Process	0.14945	CH5	0.342	2	0.299	17
		CH6	0.104	6	0.897	9
		CH7	0.378	1	0.149	19
		CH8	0.176	4	0.598	13
		CH9	0.104	6	0.897	9
		CH10	0.170	5	0.747	12
		CH11	0.320	3	0.448	15
Technology	0.27593	CH12	0.063	8	2.207	1
		CH13	0.191	5	1.380	6
		CH14	0.161	6	1.656	4
		CH15	0.487	1	0.276	18
		CH16	0.120	7	1.932	2
		CH17	0.210	4	1.104	8
		CH18	0.450	2	0.552	14
		CH19	0.340	3	0.828	10

the importance of a challenge within their category. The local ranking helps the practitioners to consider the most important challenges within a particular category. For example, CH3 ("Possible solutions have to be compared", LW = 0.420) is ranked as the highest priority challenge in human resource category. This indicated that to successfully address the human resources domain in code recommendation systems, the practitioners should consider CH3 on top priority. Moreover, the results shows that CH1 ("Lack of software engineering knowledge", LW = 0.241) and CH2 ("Lack of monitoring and management", LW = 0.180) are ranked as 2nd and 3rd most important challenges of code recommendation in human resource category. The local ranking assists to fix the challenges concerning to their significance in a particular category.

We also determine the global ranking, by calculating the global weights of each challenge. The global weight was calculated by multiplying the local weight of each challenges with the weights of their respective categories. For example, the global weight (GW) of CH1 is GW = 0.241 × 0.37938 = 0.759. Though, based on the determined global with of CH1 it is ranked as the 11 most important challenge for the efficiency of code recommendation system. The results presented in Table 15, shows that CH12 (Lack of data implementation analysis, GW = 2.207) is ranked as the highest priority challenge for effective and efficient execution of code recommendation system. We further observed that CH16 ("Poor functions design with respect to reusability", GW1.932) and CH4 ("Lack of multiple facts adoption", GW = 1.897) are ranked as the top priority challenges for recommendation systems. The global weights assists to address the most priority challenges for the efficient execution of code recommendation systems.

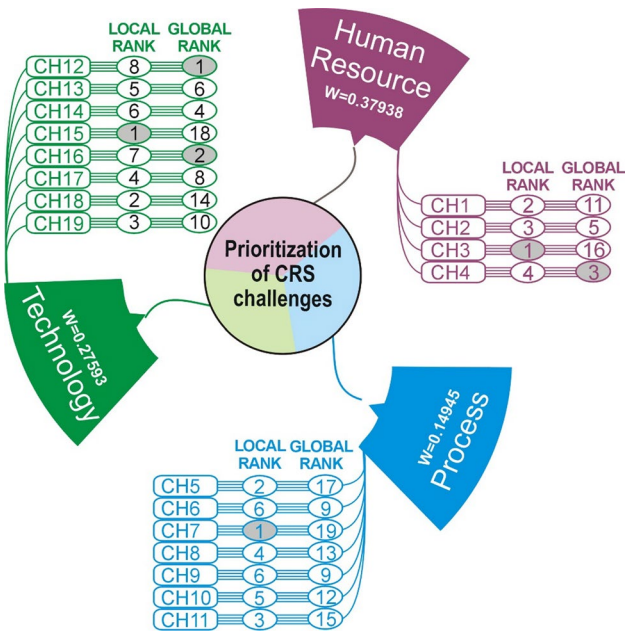

Fig. 8 Prioritization based taxonomy

Phase 6: Taxonomy of challenges

We further developed the taxonomy identified challenges by considering their categorization and their prioritization (Fig. 8). The prioritization-based taxonomy of the investigated challenges is develop using the local and global rankings of each challenge and their core categories. The developed taxonomy of the challenges shows that impact of a particular challenge within the category (local rankings) and for overall code recommendation systems (global ranking). The results shows that CH1 (Lack of software engineering knowledge) is ranked as 2nd in human resource category and standout 11th most significant challenge in overall ranking. This shows how impact of a particular vary with respect to local and global ranking. Similarly, it is observed that CH3 ("Possible solutions have to be compared) is ranked as 1st most priority challenging factor in human resource category and it ranked as 16th according to the global ranking. This variation between the priorities of challenging factors assists the practitioners to consider the most important factor with considering their receptive category and by considering the overall impact of a challenge on code recombination system. To conclude, the developed prioritization-based taxonomy assist the researchers and practitioners to consider the most significant challenges with respect to their influence on code recommendation systems.

5 Study implications and limitations

Implications: this study enlisted the challenges reported in the literature that could hinder the efficiency and effectiveness of code recommendation system were explored. The investigated challenges indicated the key areas that need to be addressed for the efficiency of code recommendation systems. The empirical investigations shows that the identified challenges are important and need special considering for the effectives of code recommendation systems. Moreover, the explored challenges were classified and the fuzzy-AHP was applied to prioritize the explored challenges concerning to their significance for code recommendation systems. The categorization and fuzzy-AHP analysis provide the prioritization-based taxonomy of the investigated challenges. The developed taxonomy provides the body of knowledge to academic researcher's community to develop the new techniques for the effective code recommendation systems.

The results of this study also have practical implications as the prioritization-based taxonomy of the identified challenges educate the real-world practitioners to consider the most critical challenging factors on priority basis. Furthermore, the identified challenging factors gives the direction to the practitioners to focus on the most critical areas and develop the new strategies for the development of an efficient recommendation system.

Limitations: Besides, there is a chance of researcher's baseness in data extraction process and the researchers might continually extract wrong data. To address this threat the inter-rater reliability text was conducted, and the results show that there is no baseness in the extracted data. The execution of search string on the selected database might lead towards the incomplete data collection. Therefore, based on the exiting studies [26, 28, 42], this omission is not systematic. Similarly, the small sample size of response for fuzzy-AHP analysis is a critical threat towards the findings of this study. As the fuzzy-AHP is a subjective study, though the results based on small data size can generalizable.

6 Summary of research findings

The aim of this research study is to identify the factors that could be potential challenges for the code recommendation system. This objective has been achieved by conducting the SLR and empirical study to explore the key challenges of the code recommendation system. Using SLR approach, total of 19 challenging factors were identified that could be critical for the code recommendation system. Additionally, the survey questionnaire study was conducted with the code recommendation systems experts to know their perceptions regarding the identified challenges. The survey results indicate that the identified challenges are significant for code recommendation system. Finally, the reported challenges and their categories were prioritize using the fuzzy AHP approach. The given prioritization provides a roadmap to the researchers and practitioners who work on code recommendation system projects. It could be used to tackle the key challenges that could hinder the code recommendation system projects. The summary of the findings against research question are briefly discussed in Table 16.

7 Conclusion and future direction

The importance of code recommendation system inspired us to identify the challenges that could hinder its effectiveness and efficiency. Using the systematic literature review, 19 challenges were investigated that are reported in state-of-the-art literature and were "mapped into three core categories of" system process improvements. The investigated challenges and their categorization were further verified conducting the questionnaire survey study. The "results of questionnaire survey study" shows that the identified challenging factors of code recommendation system from the literature are practical oriented. Finally, we performed the fuzzy-AHP analysis to prioritize the investigated challenges and their categories. The results of fuzzy-AHP analysis shows that

Table 16 Summary of study findings

Research questions	Findings
[RQ1] What challenges of code recommendation system are discussed in the existing literature?	"Lack of software engineering knowledge" "Lack of monitoring and management" "Possible solutions have to be compared" "Lack of multiple facts adoption" "All dependence needs to track down" "Lack of data quality analysis" "Need for clone removal" "Ad-hoc approach" "Search issue due to big size" "Lack of proper documentation" "Perform similar tasks" "Lack of data implementation analysis" "Architecturally-relevant code anomalies are not detected soon enough" "Rapidly occurrence of changes" "Change traceability issues" "Poor functions design with respect to reusability" "Exception handling is not a primary focus" "Missing of resalable components" "Lack of proper assistance by existing tools"
[RQ2] What is the real world significance of the code recommendation system challenges?	The results of empirical study shows that the identified challenges could negatively impact the code recommendation systems
[RQ3] How to develop the taxonomy of the investigated challenging factors based on their prioritization?	Using the fuzzy-AHP, the identified challenges are prioritized, and the results shows that "Lack of data implementation analysis", "Poor functions design with respect to reusability", "Lack of multiple facts adoption", "Rapidly occurrence of changes" and "Lack of monitoring and management" are declared as the highest priority challenges for code recommendation system The prioritization-based taxonomy of the challenging factors was developed using the local and global ranking of each challenge. The given taxonomy will provide detail knowledge to both researchers and practitioners to consider the most important challenges with respect to their significance of code recommendation systems

"Lack of data implementation analysis", "Poor functions design with respect to reusability", "Lack of multiple facts adoption", "Rapidly occurrence of changes" and "Lack of monitoring and management" are declared as the highest priority challenges for code recommendation systems. The identified list of challenges, their categorization into core three categories and their priority rankings provides the robust taxonomy that render that impact of a particular challenge within their respective category and for overall code recommendation systems. We are confident that the results and analysis of this study will be contributed towards the improvement and development of new techniques for the effective and efficient code recommendation systems.

The next phase of this research project will be based on the multivocal study in which we will identify the challenging factors discussed in grey literature and formally published literature studies. Moreover, the success factors will be identified that could positively influence the code recommendation system.

Appendix

Appendix-A: Primary studies list (https://tinyurl.com/yyhz8unf).

Appendix-B: Survey questionnaire (https://tinyurl.com/yy4mtrfe).

Appendix-D: Sample of pairwise comparison questionnaire (https://tinyurl.com/yy6tyjea).

Funding Open Access funding provided by University of Oulu including Oulu University Hospital.

References

1. Afzal W, Torkar R, Feldt R (2008) A systematic mapping study on non-functional search-based software testing. In: SEKE, vol 8, pp 488–493
2. Akbar MA, Sang J, Khan AA, Mahmood S, Qadri SF, Hu H et al (2019) Success factors influencing requirements change management process in global software development. J Comput Lang 51:112–130
3. Akbar MA, Shameem M, Khan AA, Nadeem M, Alsanad A, Gumaei A (2020) A fuzzy analytical hierarchy process to prioritize the success factors of requirement change management in global software development. J Softw Evol Process 33:e2292
4. Albayrak E, Erensal YC (2004) Using analytic hierarchy process (AHP) to improve human performance: an application of multiple criteria decision making problem. J Intell Manuf 15(4):491–503
5. Ayhan MB (2013) A fuzzy AHP approach for supplier selection problem: a case study in a Gear motor company. arXiv:1311.2886
6. Baltar F, Brunet I (2012) Social research 2.0: virtual snowball sampling method using Facebook. Internet research
7. Binkley D (2007) Source code analysis: a road map. In: 2007 Future of software engineering, 2007. IEEE Computer Society, pp 104–119
8. Bozbura FT, Beskese A, Kahraman C (2007) Prioritization of human capital measurement indicators using fuzzy AHP. Expert Syst Appl 32(4):1100–1112
9. Chamodrakas I, Batis D, Martakos D (2010) Supplier selection in electronic marketplaces using satisficing and fuzzy AHP. Expert Syst Appl 37(1):490–498
10. Chang D-Y (1996) Applications of the extent analysis method on fuzzy AHP. Eur J Oper Res 95(3):649–655
11. Chang YB, Gurbaxani V (2012) Information technology outsourcing, knowledge transfer, and firm productivity: an empirical analysis. MIS Q 36(4):1043–1065
12. Chen L, Ali Babar M, Zhang H (2010) Towards an evidence-based understanding of electronic data sources. In: 14th International conference on evaluation and assessment in software engineering (EASE), pp 1–4
13. Cheng EW, Li H (2002) Construction partnering process and associated critical success factors: quantitative investigation. J Manag Eng 18(4):194–202
14. Corbin JM, Strauss A (1990) Grounded theory research: procedures, canons, and evaluative criteria. Qual Sociol 13(1):3–21
15. Espino-Rodríguez TF, Padrón-Robaina V (2006) A review of outsourcing from the resource-based view of the firm. Int J Manag Rev 8(1):49–70
16. Fanning MC, Nagappan N, Ball TJ, Sandys S (2014) Prioritizing quality improvements to source code. Google Patents
17. Finstad K (2010) Response interpolation and scale sensitivity: evidence against 5-point scales. J Usability Stud 5(3):104–110
18. Harman M (2010) Why source code analysis and manipulation will always be important. In: 2010 10th IEEE working conference on source code analysis and manipulation. IEEE, pp 7–19
19. Heininger R (2012) IT service management in a cloud environment: a literature review. In: Proceedings of 9th workshop on information systems and services sciences, pp 1–12
20. Hossain MS, Tanim AS, Nawal N, Akter S (2019) An innovative tour recommendation system using graph algorithms. J Inf Syst Eng Bus Intell 5(1):32–39
21. Inayat I, Salim SS, Marczak S, Daneva M, Shamshirband S (2015) A systematic literature review on agile requirements engineering practices and challenges. Comput Hum Behav 51:915–929
22. Janjic W, Hummel O, Atkinson C (2014) Reuse-oriented code recommendation systems. In: Recommendation systems in software engineering. Springer, pp 359–386
23. Kedia BL, Mukherjee D (2009) Understanding offshoring: a research framework based on disintegration, location and externalization advantages. J World Bus 44(3):250–261
24. Kellens A, Mens K (2011) Mendel: source code recommendation based on a genetic metaphor. In: ASE'11 Proceedings of the IEEE/ACM international conference on Automated software engineering
25. Khan AA, Keung J (2016) Systematic review of success factors and barriers for software process improvement in global software development. IET Softw 10(5):125–135
26. Khan AA, Keung J, Hussain S, Niazi M, Kieffer S (2018) Systematic literature study for dimensional classification of success factors affecting process improvement in global software development: client–vendor perspective. IET Softw 12(4):333–344
27. Khan AA, Keung J, Hussain S, Niazi M, Tamimy MMI (2017) Understanding software process improvement in global software development: a theoretical framework of human factors. ACM SIGAPP Appl Comput Rev 17(2):5–15
28. Khan AA, Keung J, Niazi M, Hussain S, Ahmad A (2017) Systematic literature review and empirical investigation of barriers to process improvement in global software development: client–vendor perspective. Inf Softw Technol 87:180–205
29. Khan AA, Shameem M (2020) Multicriteria decision-making taxonomy for DevOps challenging factors using analytical hierarchy process. J Softw Evol Process 32(6):1–26
30. Khan SU, Niazi M, Ahmad R (2011) Factors influencing clients in the selection of offshore software outsourcing vendors: an exploratory study using a systematic literature review. J Syst Softw 84(4):686–699
31. Kitchenham B, Charters SM (2007) Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE Technical Report EBSE-2007-01, Keele Universityand Durham University
32. Kitchenham B, Pfleeger SL (2002) Principles of survey research: part 5: populations and samples. ACM SIGSOFT Softw Eng Notes 27(5):17–20
33. Kitchenham B, Pfleeger SL (2003) Principles of survey research part 6: data analysis. ACM SIGSOFT Softw Eng Notes 28(2):24–27
34. Klint P, Van Der Storm T, Vinju J (2009) Rascal: a domain specific language for source code analysis and manipulation. In: 2009 Ninth IEEE international working conference on source code analysis and manipulation. IEEE, pp 168–177
35. Kosar T, Bohra S, Mernik M (2018) A systematic mapping study driven by the margin of error. J Syst Softw 144:439–449
36. Lewis-Beck M, Bryman AE, Liao TF (2003) The Sage encyclopedia of social science research methods. Sage Publicatons, London
37. Li W (2009) Application of AHP analysis in risk management of engineering projects. J Beijing Univ Chem Technol 1:46–48
38. Lozano A, Kellens A, Mens K (2011) Mendel: source code recommendation based on a genetic metaphor. In: Proceedings of the 2011 26th IEEE/ACM international conference on automated software engineering. IEEE Computer Society, pp 384–387
39. Meijer HJM, Gates III, WH, Ozzie RE, Flake GW, Bergstraesser TF, Blinn AN et al (2010) Recommendation system that identifies a valuable user action by mining data supplied by a plurality of users to find a correlation that suggests one or more actions for notification. Google Patents

40. Mens K, Lozano A (2014). Source code-based recommendation systems. In: Recommendation systems in software engineering. Springer, pp 93–130
41. Mkaouer MW, Kessentini M, Bechikh S, Deb K, Ó Cinnéide M (2014) Recommendation system for software refactoring using innovization and interactive dynamic optimization. In: Proceedings of the 29th ACM/IEEE international conference on automated software engineering. ACM, pp 331–336
42. Niazi M, Mahmood S, Alshayeb M, Qureshi AM, Faisal K, Cerpa N (2016) Toward successful project management in global software development. Int J Project Manag 34(8):1553–1567
43. Palcic I, Lalic B (2009) Analytical hierarchy process as a tool for selecting and evaluating projects. Int J Simul Modell IJSIMM 8(1):16–26
44. Ricci F, Rokach L, Shapira B (2011) Introduction to recommender systems handbook. In: Recommender systems handbook. Springer, pp 1–35
45. Robles G, Gonzalez-Barahona JM, Merelo JJ (2006) Beyond source code: the importance of other artifacts in software development (a case study). J Syst Softw 79(9):1233–1248
46. Sadler GR, Lee HC, Lim RSH, Fullerton J (2010) Recruitment of hard-to-reach population subgroups via adaptations of the snowball sampling strategy. Nurs Health Sci 12(3):369–374
47. Shameem M, Khan AA, Hasan MG, Akbar MA (2020) Analytic hierarchy process based prioritisation and taxonomy of success factors for scaling agile methods in global software development. IET software
48. Shameem M, Kumar RR, Kumar C, Chandra B, Khan AA (2018) Prioritizing challenges of agile process in distributed software development environment using analytic hierarchy process. J Softw Evol Process 30(11):e1979
49. Sloane E, Liberatore M, Nydick R, Luo W, Chung Q (2002) Clinical engineering technology assessment decision support: a case study using the analytic hierarchy process (AHP). In: Proceedings of the second joint 24th annual conference and the annual fall meeting of the biomedical engineering society engineering in medicine and biology, vol 3. IEEE, pp 1950–1951
50. Sridevi M, Rao RR, Rao MV (2016) A survey on recommender system. Int J Comput Sci Inf Secur 14(5):265
51. Stelzer D, Mellis W (1998) Success factors of organizational change in software process improvement. Softw Process Impro Pract 4(4):227–250
52. Viadiu FM, Fa MC, Saizarbitoria IH (2006) ISO 9000 and ISO 14000 standards: an international diffusion model. Int J Oper Prod Manag 26:141–165
53. Wong JK, Li H (2008) Application of the analytic hierarchy process (AHP) in multi-criteria analysis of the selection of intelligent building systems. Build Environ 43(1):108–125
54. Yaghoobi T (2018) Prioritizing key success factors of software projects using fuzzy AHP. J Softw Evol Process 30(1):e1891
55. Yamashita A, Moonen L (2012) Do code smells reflect important maintainability aspects? In: 2012 28th IEEE international conference on software maintenance (ICSM). IEEE, pp 306–315
56. Zadeh LA, Klir GJ, Yuan B (1996) Fuzzy sets, fuzzy logic, and fuzzy systems: selected papers, vol 6. World Scientific, Singapore
57. Zhang H, Babar MA, Tell P (2011) Identifying relevant studies in software engineering. Inf Softw Technol 53(6):625–637