An Ontology-Based Model for Task Recommendation in Crowdsourced Software Engineering Environment

Umair Ayaz Kamangar, Isma Farah Siddqui*, Qasim Ali Arain, Zainab Umair Kamangar

Department of Software Engineering, Mehran University of Engineering and Technology, Jamshoro, Pakistan

Abstract: Assigning tasks to appropriate people in crowd-based software engineering has always been an arduous task for a manager because the volume of the crowd to which a task is assigned is increasing day by day, and the process of selecting which task should be allocated to which person is becoming difficult. In order to make task recommendations more effective and less time-consuming, much research has been done. However, the problem is that such research does not contain the required implementation strategy and current recommendation techniques in crowdsourcing use workers' history and mining methods, which cannot relate workers with tasks and other workers. This research aims to incorporate semantic capabilities in task allocation in a crowdsourced software engineering environment so that machines could guide crowdsourcing managers in choosing appropriate crowd workers for a task to be solved. In order to solve this problem and allow the machine to prescribe which resource is more suitable for a task semantically, an ontology-based recommendation technique is proposed. Ontologies relate information in a way that the machine can understand its semantics. These ontologies, along with data, create a knowledge base upon which semantic web works. Two ontologies, one of the crowds to which the task must be assigned and one of the tasks under consideration, have been developed to incorporate semantic capabilities in the crowdsourced task allocation process. These ontologies were designed keeping in view various factors obtained through analyzing different researches done to improve the task allocation in crowdsourcing. Then these ontologies were developed using RDF and OWL languages, and, in the end, these ontologies were tested using SPARQL. The performance and accuracy of the results returned by queries were also measured to know the efficiency of this approach in task allocation in a crowdsourcing environment. This approach can save time and provide more efficiency in a manager responsible for assigning tasks to different persons in a crowd-based software engineering environment.

Keywords: crowdsourcing, software engineering, semantic web, ontology, open call.

眾包軟件工程環境下基於本體的任務推薦模型

摘要：在基於人群的軟件工程中，將任務分配給合適的人對於管理者來說一直是一項艱鉅的任務，因為分配任務的人群數量日益增加，選擇哪個任務應該分配給哪個的過程人變得困難。為了使任務推薦更有效、更省時，已經進行了大量研究。然而，問題在於此類研究不包含眾包中所需的實施策略和當前推薦技術，使用工人的歷史和挖掘方法，無法將工人與任務和其他工人聯繫起來。本研究旨在將語義能力納入眾包軟件工程環境中的任務分配，以便機器可以指導眾包管理人員為要解決的任務選擇合適的眾包工作者。為了解決這個問題並允許機器從語義上規定哪種資源更適合任務，提出了一種基於本體的推薦技術。本體以機器可以理解其語義的方式關聯信息。這些本體與數據一起創建了語義網絡工作的知識庫。已經開發了兩種本體，一種是任務必須分配到的人群，另一種是正在考慮的任務，已被開發以在眾包任務分配過程中結合語義功能。這些本體的設計考慮到了通過分析為改善眾包任務分配所做的不同研究而獲得的各種因素。然後使用資源描述框架和網絡本體語言語言開發這些本體，最後使用簡單協議和資源描述框架查詢語言測試這些本體。還測量了查詢返回結果的性能和

Received: July 24, 2021 / Revised: September 21, 2021 / Accepted: October 20, 2021 / Published: November 30, 2021

About the authors: Umair Ayaz Kamangar, Isma Farah Siddiqui, Qasim Ali Arain, Zainab Umair Kamangar, Department of Software Engineering, Mehran University of Engineering and Technology, Jamshoro, Pakistan

Corresponding author Isma Farah Siddiqui, isma.farah@faculty.muet.edu.pk

準確性，以了解這種方法在眾包環境中任務分配的效率。這種方法可以在基於人群的軟件工程環境中為負責將任務分配給不同人員的經理節省時間並提供更高的效率。

关键词：眾包、軟件工程、語義網、本體、公開徵集。

1. Introduction

The Semantic Web is defined as an extension of the world wide web with meaningful information understandable by both machines and people to work together cooperatively [1]. It is based upon three technologies, which are ontologies, URIs, and RDF [2].

In ontology, knowledge is represented by a set of notions within a field and the associations between those notions. [3]. Crowdsourcing is an evolving technique based upon a disseminated model for solving a problem; Howe and Robinson also formulated "crowdsource." In crowdsourcing, the companies outsource the crowd workers on the network through an open-call format to get their job done [4]. Crowd-based software engineering is a process of assigning tasks related to the design, development, and testing of software to a crowd through an open-call format. In this process, many designers, developers, and testers are acquired online to solve a task.

Selecting a suitable task from many tasks is problematic for crowded people responsible for solving a task [5]. It is also very exhausting for people working on a platform to look into many solved tasks and determine appropriately solved [6]. According to the research [7], if the task is not properly assigned to an appropriate person, it will considerably reduce software quality. Proper assignment of tasks to an appropriate worker is a grave issue [8], solving which would provide an advantage for every stakeholder in the process of crowdsourced software engineering, whether it is task performer or platform manager.

Much research has been done to address the mentioned issue by assigning tasks only to those good at performing those tasks [9-11]. This approach would save a lot of time and money in the crowdsourced software engineering environment. These approaches focused upon different aspects, i.e., considering the prize money, complexity, and deadline of a task and personality of a person, based upon which task is assigned. The personality of a person solving a task is a vital component that directly influences the output of the task solution [12]. Many researchers associated with crowdsourcing software development or the creation of software in-house have emphasized the importance of personality for the effective development of the task [13, 14]. It has also been established that a task can be well accomplished when it completes its budgetary requirements and meets its deadline [15].

In the current scenario, when a crowdsourced manager wants to choose from diverse resources to solve a particular task, the machine cannot navigate semantically through the knowledge base to identify appropriate resources. The knowledge base may include information about URI software engineers and semantically guide crowdsourced managers about the appropriate resource (person). This problem can be solved by creating domain ontologies and relating with one another based upon knowledge present at different sources. These ontologies shall be used by machines to recommend outsourcing tasks to a relevant resource using its associated knowledge based upon the profile of a person and traits of the task under consideration.

In this research, information has been gathered on factors that affect task allocation in a crowdsourcing software engineering environment, created ontologies, and relationships between things present in these ontologies. Hence, whenever a task to be solved comes under consideration, the machine will automatically search for appropriate resources for that task over the web. This result will make the process of resource recommendation and task allocation less time-consuming and efficient.

The contributions of the approach in this research are:

1. Two ontologies are developed in this research which are the crowd and task ontologies.
2. Relationships are created between traits of crowd workers and tasks.
3. Data is collected from different sources and combined with ontologies to form a knowledge base.
4. The knowledge base is tested by using SPARQL.
5. Precision and recall of results obtained are calculated to know the usefulness of this resource recommendation technique in the task allocation process of crowdsourcing.

The other sections of this research paper are structured as follows. Section 2 discusses the related works. In section 3, the model for ontology-based task recommendation is presented. Section 3 has four sub-sections: design of ontologies is presented in section 3.1, the algorithm for ontology-based task recommendation is given in section 3.2, implementation of ontologies is discussed in section 3.3, and section 3.4 discusses the testing of these ontologies. Section 4 discusses the results with their accuracy and performance in a crowd-based environment with different crowd workers. Finally, Section 5 presents the conclusion.

2. Related Works

Task allocation is a very challenging process in a crowdsourcing software engineering environment. If the tasks are allocated solely upon the preference of the person responsible for assigning tasks, it could waste much time and money, according to S. Zhao et al. [16]. Both the complexity of a task and the profile of a task performer affect the performance in crowd-based software development, as stated by [17, 18]. The issue of connection of task performers with their capabilities and knowledge and benefit of a crowdsourced developer had been discussed [8].

In order to address the issue of correspondence between task and developer and to assign a task to an appropriate crowd worker, much research has been done. Model for allocation of tasks in Crowdsourcing software development had also been offered [19], based on which domain ontologies for allocating tasks can be designed appropriately. Recommendation of tasks in a crowd-based environment has been made using personal preference [20]. It has been identified that personality types of crowdsourcing developers are related to deadlines and prize money of a task to be developed. Personality characteristics are associated with the selection of tasks in a crowdsourcing environment. It was also established that some personalities are more inclined to choose prize money than the time limit, while others consider the time limit when selecting a task [21]. The association between different techniques in crowd-based testing and personality characteristics has also been identified [22]. It was established that software testers having extrovert personality type are more appropriate at performing black box testing than testers having introvert personality, while testers having introvert personality type are more appropriate for performing white box testing and also it was established that testers having judging dimension in their personalities are more appropriate for black-box testing [22]. In these researches, the relationship between personality types of software testers and testing levels has been established, which shows that testers with introverted personalities perform more efficiently at the unit testing level. In contrast, testers with extrovert personalities perform more efficiently at integration and system testing levels. It was also stated that testers with sensing dimensions perform better at the Integration testing level, while testers having judging dimensions in their personality perform better at the System testing level [23, 24].

Ontologies have been used to develop different recommendation systems and algorithms. Obeid, C. et al. [25] proposed a recommendation system in higher education based on ontology, which recognizes different traits of students and recommends the major or university that student. Different recommender systems are based upon ontologies [26]. An ontology-based System recommending appropriate COTS components was proposed by Yanes, N. et al. [27]. Williams I. et al. [28] proposed a collaborative recommender system based on an ontology that will help software engineers elicit security requirements; it works by recommending Common Attack Patterns Enumeration Classification attack patterns and Common Weakness Enumeration based on use case descriptions. Framework for the personalized course recommendation based on ontology had also been proposed [29]. Neethukrishnan K. et al. [30] proposed a recommendation of research papers related to computer science based on ontology, which recommends ten most relevant papers to the user. It is also significant to point out that a Knowledge base was used to put sense in machines [31, 32].

Some techniques presented in other researches for recommendation in the task allocation process are content-based, i.e., they search through history to recommend a resource for a task. In contrast, others are scenario-based, but they do not consider the relationship between task and crowd worker. In the technique presented in this research, traits of worker and task are related through ontologies, and recommendations are based upon these ontologies.

3. Ontology-Based Model for Task Recommendation

The proposed approach will work the same as the open-call format in crowd-based software engineering. However, it includes using ontologies by which a machine will semantically guide a manager about appropriate resources for the task to be solved. Once the manager knows about appropriate resources, he will assign tasks and get improvised results in less time. In order to know which person is suitable for which task, different traits of a person and different factors related to a task are considered. Firstly, in a methodology to implement this approach, information has been collected regarding the task allocation and developers through different sources. Then these ontologies (Task ontology and Crowd ontology) have been designed (Fig. 1). After that, these ontologies are implemented using Ontology editors, which are Fluent editor and Protégé. Then these ontologies have been tested through SPARQL. Lastly, the performance of results obtained from this approach is evaluated.

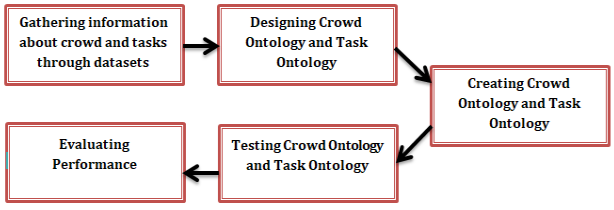
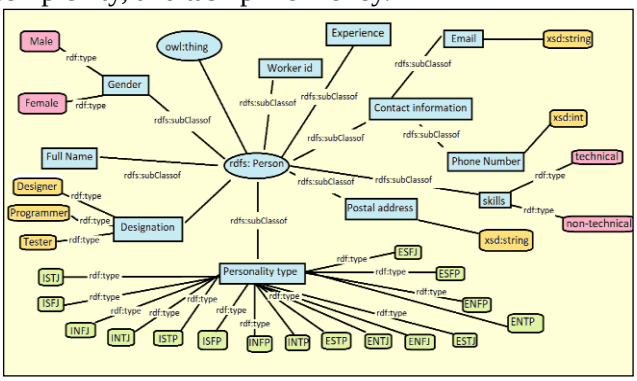


Fig. 1 Ontology-based task recommendation model

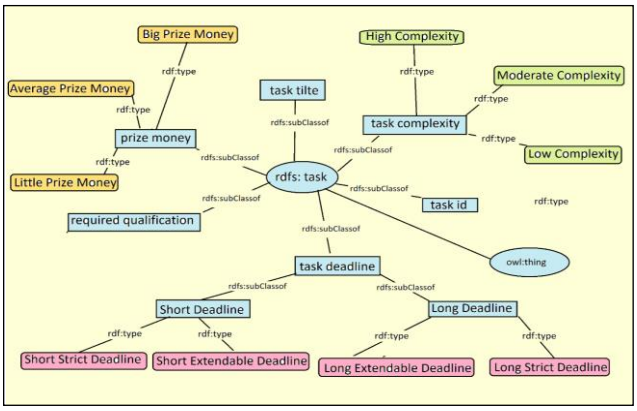
3.1. Design of Ontologies

Ontologies have been designed after thorough consideration of different factors which play a vital role in improving the process of task allocation in a crowdsourcing software engineering environment. Classes, Sub-classes, and Instances have been included, and relations among them are established to understand better the process of task assignment in a crowd-based software engineering environment. These classes and their relationships are designed by analyzing and generalizing information obtained from various software houses and present in other research. Two ontologies have been designed, i.e., (i) Crowd ontology and (ii) Task ontology. These ontologies will be combined and related together to generate results that will help managers assign tasks to designers, developers, and testers more efficiently. First of all, crowd ontology has been designed, as shown in Fig. 2 (a), which focuses on the person responsible for performing these tasks. This ontology contains the personality type, designation of a person such as designer, developer, or tester, education of a person, experience and contact information, and others.

Then task ontology has been designed as shown in Fig. 2 (b), which will have information about tasks under consideration such as task deadline, task complexity, and task prize money.



(a)



(b)

Fig. 2 Design of crowd ontology (a) and task ontology (b)

Based upon the information on which personality type is more appropriate for which type of task and what skills are required for a particular task and considering other factors, the relationships between these ontologies have been developed and implemented.

3.2. Algorithm of Ontology-Based Task Recommendation Model

The algorithm presented below shows the operation of the approach presented in this research, which includes the structure of the implementation of the crowd ontology and task ontology, the formation of knowledge-base after the insertion of data in these ontological structures, and querying through this knowledge-base:

```
begin
  creating project;
          project <--- create project("Task recommendation using
ontology");
          output <--- ("project created");
    creating ontologies;
          ontology <--- create ontology(Project-id,"crowd
ontology");
        ontology <--- create ontology(project-id,"task ontology");
  creating classes and subclasses for ontologies;
        class <--- (crowd ontology,"person");
          subclass <-- (super class "person", "Full name");
      subclass <-- (super class "person", "designation");
          subclass <-- (super class "person", "experience");
          subclass <-- (super class "person", "personality type");
          subclass <-- (super class "person", "education");
          subclass <-- (super class "person", "skills");
          class <--- (task ontology,"task");
          subclass <-- (super class "task", "complexity");
          subclass <-- (super class "task", "deadline");
          subclass <-- (super class "task", "prizemoney");
    creating relationship between classes of person and tasks;
  inserting crowd in crowd ontology;
     for (int i=0; i<= length of dataset; i++){
       crowd ontology<--- person(i)
          person(i).name<--- insert name of ith person;
          person(i).designation<--- insert designation("designer",
"developer", "tester");
          person(i).experience<---- insert experience of person;
          person(i).personality_type<--- insert one of 16
personality types;
          person(i).qualification<--- insert qualification;
          person(i).contact<----- insert contact information;
          person(i).email<----- insert email id;}
  inserting task to be solved;
   for each task to be solved
      insert task.complexity<----("Simple","Moderate","Complex");
          insert task.deadline<--- ("short deadline","long
deadline");
         insert task.prize money<----("big prize money","little prize
money");
  Knowledgebase is completed through ontologies along with its
relations and data;
    Now there is SPARQL endpoint which can used to query
knowledgebase created;
        recommendation<---SPARQL_Endpoint(query);
  Ask the query to get results required for allocation of task to
suitable resource;
        write SPARQL query;
          result<--- recommend resource
  Display the results.
     for recommendation in results. recommendation do
      output<---(information of person suitable for the task required
in query)
```

3.3. Implementation of Ontologies

OWL classes and sub-classes have been implemented in RDF language using Fluent Editor. Crowd ontology and Task ontology have been

implemented and related with each other forming Task Allocation Ontology. This ontology takes various traits of a person into account, i.e., designation, experience, personality type, and others, and suggests which person is suitable for which task. It also considers which task is how complex, which task has higher prize money, or which tasks have a strict deadline or not. Finally, this ontology helps in recommending a task to the appropriate crowd by suggesting suitable people as per that task type. After the materialization of relationships, the outcome of the structure of ontologies is presented in Fig. 3.

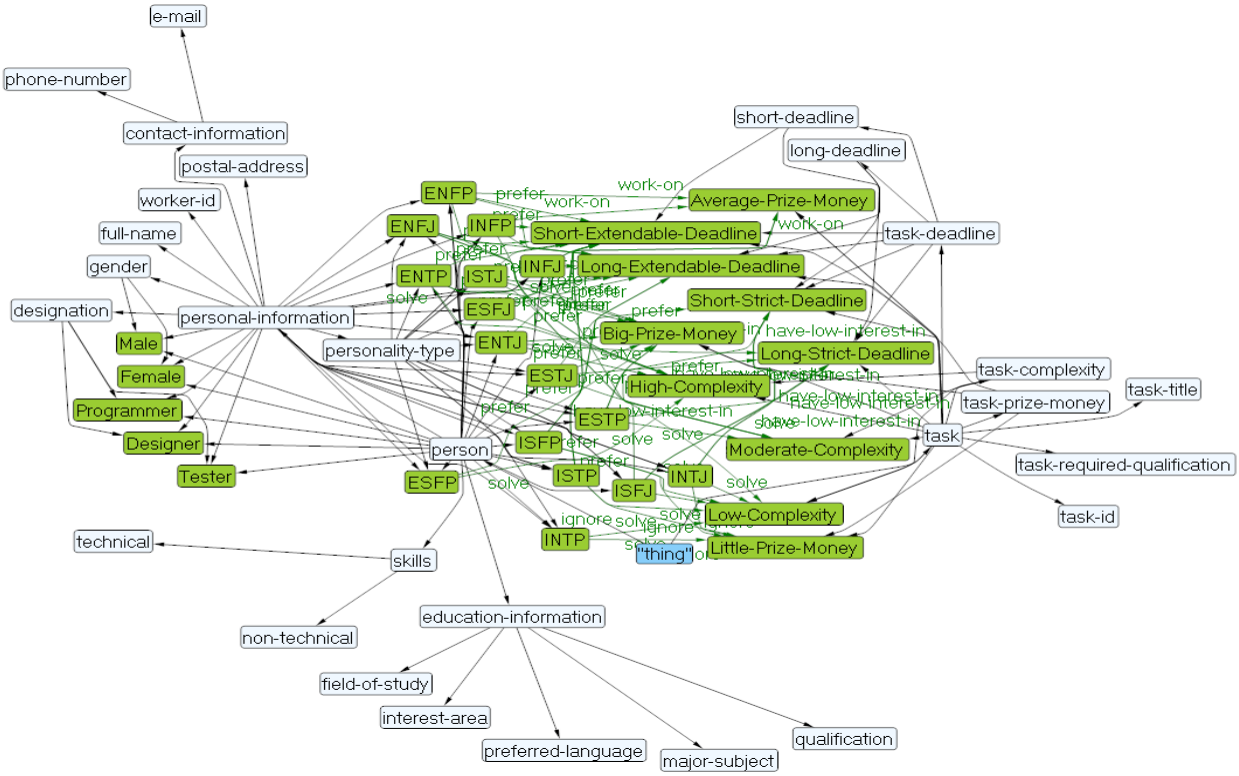


Fig. 3 Structure of ontologies after materialization of relationships

3.4. Testing of Ontologies

SPARQL is the query language used to test the relationship between the things present in both ontologies, i.e., Crowd ontology and Task ontology. Datasets obtained from different software houses and online sources have been inserted in these ontologies and tested with the help of SPARQL to test the relationship created between classes of these ontologies and to make sure that results returned can be used by the crowdsourced manager for appropriate task allocation. Firstly, queries regarding the designation of people present in Crowd ontology (i.e., designer, developer, or tester) have been asked, which returned us the results of designations of all the people present in knowledge-base. Then queries related to personality types, i.e., which personality type is suited for which type of task, have been given to ontologies through SPARQL, which returned us the results showing us that some personality types preferred extended task deadline, some ignore tasks with high complexity while some prefer big prize money. Testing of ontologies has shown us that the relationship between Crowd ontology and Task ontology is established to recommend a suitable resource for the task to be solved.

4. Results and Discussion

After carefully inserting datasets in these ontologies from some registered software houses, six queries were executed to test the performance and accuracy of recommendations obtained. In Query 1, the ontological server was asked to recommend suitable developers for a complex task. In Query 2, it was asked to recommend the developers who prefer simple tasks. In Query 3, the ontological server was used to recommend the testers who prefer tasks with extended deadlines. In Query 4, the ontological server was asked to recommend the testers suitable for tasks with a short deadline. In Query 5, the ontological server was asked to recommend the personality types of every person, whether designer, developer, or tester, who prefers prize money over the complexity that means they are not bothered by complex tasks as far as prize money is enormous. Finally, in Query 6, the ontological server was asked to recommend the personality types of every person, whether designer, developer, or tester, who looks upon complexity rather than prize money that means persons who prefer simple task and does not bother about money.

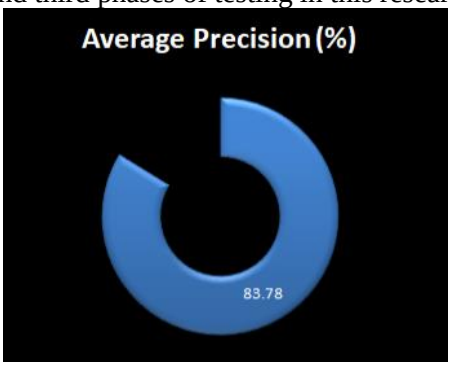
In the first phase of testing, the dataset of 300 persons was incorporated into a knowledge base consisting of Crowd ontology and Task ontology. Then same queries were executed using SPARQL. Table 1

shows the total number of the result returned by these queries, along with their precision and recall.

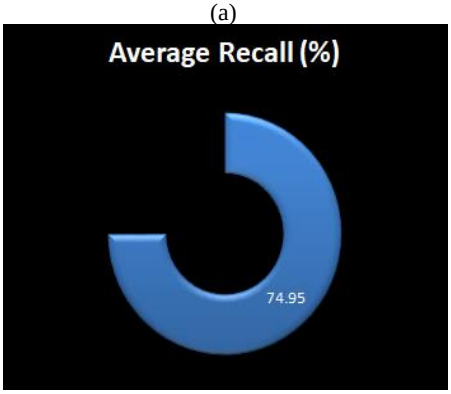
Table 1 Results returned by SPARQL queries when dataset consists of 300 crowd workers

	No. of results	True Positive	Actual Positive	Precision (%)	Recall (%)
Query 1	60	55	65	91.66	84.61
Query 2	75	70	85	93.33	82.35
Query 3	32	25	40	78.12	62.5
Query 4	46	38	50	82.66	76
Query 5	78	60	85	76.92	70.58
Query 6	35	28	38	80	73.68

The average precision obtained in the first phase of testing is 83.78 %, which is reasonably satisfactory and the Average Recall for all executed queries is 74.95% (Fig. 4). However, the dataset size in the first phase of testing is 300, which is not large, so this ontology-based resource recommendation must be tested on larger datasets of crowd workers, which is done in the second and third phases of testing in this research.



(a)



(b)

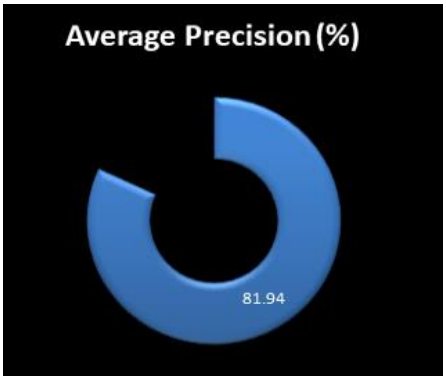
Fig. 4 All queries on a dataset of 300 crowd workers: (a) precision, (b) recall

In the second testing phase, the dataset of 1000 persons was incorporated into the knowledge base and tested using the same queries. Table 2 shows the precision and recall of the results returned on the dataset of 1000 persons.

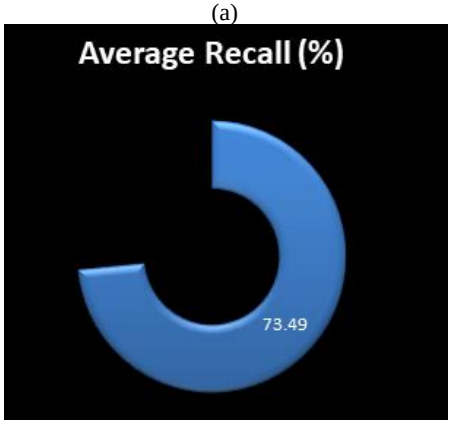
Table 2 Results returned by SPARQL queries when dataset consists of 1000 crowd workers

	No. of results	True Positive	Actual Positive	Precision (%)	Recall (%)
Query 1	235	210	255	89.36	82.35
Query 2	450	410	506	91.11	81.02
Query 3	320	245	403	76.56	60.79
Query 4	222	179	239	80.63	74.89
Query 5	342	255	369	74.56	69.1
Query 6	380	302	415	79.47	72.77

After increasing size considerably to 1000, average precision and an average recall of results obtained are slightly decreased to 81.94 and 73.49%, respectively (Fig. 5).



(a)



(b)

Fig. 5 Average of all queries on a dataset of 1000 crowd workers: (a) precision, (b) recall

In the third testing phase, the dataset of 2000 persons was incorporated into the knowledge base. Table 3 shows the results returned by the same queries on the dataset of 2000 persons with their precision and recall.

Table 3 Results returned by SPARQL queries when dataset consists of 2000 crowd workers

	No. of results	True Positive	Actual Positive	Precision (%)	Recall (%)
Query 1	458	405	495	82.42	81.81
Query 2	966	879	1087	90.99	80.86
Query 3	699	529	878	75.67	60.25
Query 4	478	378	516	79.07	73.25
Query 5	720	531	770	73.75	68.96
Query 6	815	638	889	78.28	71.76

After increasing the size to 2000, average precision and an average recall of results obtained slightly decreased to 81.03 and 72.81%, respectively (Fig. 6). It shows that an increase in the dataset size in this ontology-based task recommendation in a crowdsourcing environment does not affect its performance much.

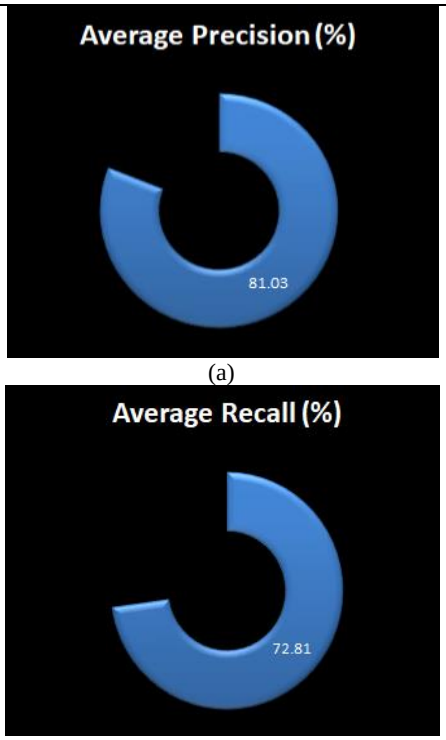


Fig. 6 Average of all queries on a dataset of 2000 crowd workers: (a) precision, (b) recall

Results obtained after testing the developed knowledge base show that this method can be usefully applied to help the crowdsourced manager recommend suitable workers for the appropriate task. This method can help software companies by providing them with semantic capabilities in task allocation in crowdsourcing. In contrast, other methods do not help because they do not relate the traits of worker and task in such a manner through ontologies. The limitations of this method are that if data used with these ontologies are not collected from a reliable source, then the accuracy of results is affected considerably. Another limitation of this method is that these ontologies should be regularly updated as the new research on the relationship between worker and task should be done in the future.

5. Conclusion

This research aimed to incorporate semantic capabilities in the process of task allocation in a crowdsourced software engineering environment so that machines could guide crowdsourcing managers in choosing appropriate crowd workers for a task to be solved. This aim is achieved by developing two domain ontologies, which are crowd ontology and Task ontology. First of all, the design of these ontologies was created. The crowd ontology consists of the designation, experience, personality type, education, and skills of a crowd worker. In contrast, task ontology consists of complexity, deadline, and prize money related to the task. Then these ontologies were implemented together, and relationships were created between things present in both ontologies so that machines could infer by looking through these relationships and recommend which crowd worker is suitable for which type of task. Finally, these ontologies were tested on three datasets with a different number of crowd workers using SPARQL queries. The average precision and recall of the results obtained after testing on the dataset of 300 crowd workers is 83.78% and 74.95%, respectively; the average precision and recall of the results obtained after testing on the dataset of 1000 crowd workers is 81.94% and 73.49% respectively, while the average precision and recall of the results obtained after testing on the dataset of 2000 crowd workers is 81.03% and 72.81% respectively. From the results acquired, it is concluded that precision and recall of results returned decreases slightly with the increase in the number of crowd workers. However, the overall average precision and average recall on all datasets are satisfactory, which would help the crowdsourcing manager in recommending mostly those crowd workers who are suitable for the task under consideration. This approach would make the whole process of crowdsourced software engineering less time-consuming and more efficient.

References

[1] BERNERS-LEE T., HENDLER J., and LASSILA O. The Semantic Web. *Scientific American*, 2001, 284: 34-43.

[2] ANTONIOU G., GROTH P., VAN HARMELEN F. and HOEKSTRA R. *A Semantic Web Primer*. 3rd ed. The MIT Press, Cambridge, Massachusetts. 2012.

[3] MAN D. Ontologies in Computer Science. *Didactica Mathematica*, 2013, 31(1): 43-46.

[4] HOWE J. The rise of crowdsourcing. *Wired Magazine*, 2006, 14(6): 1-4.

[5] MAO K., CAPRA L., HARMAN M., and JIA Y. A survey of the use of crowdsourcing in software engineering. Journal *of Systems and Software*, 2017, 126: 57-84.

[6] BU Q., SIMPERL E., CHAPMAN A., and MADDALENA E. Quality assessment in crowdsourced classification tasks. *International Journal in Crowd Science*, 2019, 3(3): 222-248.

[7] GONCALVES J., FELDMAN M., SUBINGQIAN H., and KOSTAKOS V. Task Routing and Assignment in Crowdsourcing based on Cognitive Abilities *Proceedings of 26th International World Wide Web Conference*, 2017. DOI:10.1145/3041021.3055128

[8] DURWARD D., BLOHM I., and LEIMEISTER J.M. The Nature of Crowd Work and its Effects on Individuals' Work Perception. *Journal of Management Information Systems*, 2020, 37(1): 66-95.

[9] XING Y., WANG L., and LI Z. Multi-Attribute crowdsourcing task assignment with stability and satisfactory. *IEEE Access*, 2019, 7: 133351-133361.

[10] TRAN L., TO H., FAN L., and SHAHABI C. A real-time framework for task assignment in hyperlocal spatial crowdsourcing. *ACM Transactions on Intelligent Systems and Technology*, 2018, 9(3): 1-26.

[11] ALIREZA S., SHAFIGHEH H., and MASOUD R.A. Personality classification based on profiles of social

networks' users and the five-factor model of personality. *Human-Centric Computing and Information Science*, 2018, 8(1): 24-38.
[12] SHI X., EVANS R., PAN W., and SHAN W. Understanding the effects of personality traits on solver engagement in crowdsourcing communities: a moderated mediation investigation. *Information Technology & People*, 2021.
[13] GILAL A.R., TUNIO M.Z., WAQAS A., and ALMOMANI M.A. Task Assignment and Personality: Crowdsourcing Software Development. *Human Factors in Global Software Engineering*, 2019, 1: 1-19.
[14] PATHAN N., ALI Q., IFTIKHAR S., BATOOL G., and MEMON I. Personality Type Recommendation System using Crowdsourcing. *Proceedings of International Conference on computing, Mathematics and Engineering Technologies*, 2019.
[15] YADAV A., CHANDRA J., and SAIRAM A.S. A Budget and Deadline Aware Task Assignment Scheme for Crowdsourcing Environment. *IEEE Transactions on Emerging Topics in Computing*, 2021.
[16] ZHAO Y., ZHENG K., YIN H., LIU G., FANG J., and ZHOU X. Preference-aware Task Assignment in Spatial Crowdsourcing: from Individuals to Groups. *IEEE Transactions on Knowledge and Data Engineering*, 2020.
[17] TA H.H. Assessing the Impacts of Crowdsourcing in Logistics and Supply Chain Operations. *Theses and Dissertations*, 2018.
[18] HETTIACHCHI D., VAN BERKEL N., HOSIO S., KOSTAKOS V., and GONCALVES J. Effect of Cognitive Abilities on Crowdsourcing Task Performance. *Proceedings of the 17th International Federation for Information Processing International Conference, Part I. Lecture Notes in Computer Science book series*, 11746: 442-464.
[19] TUNIO M.Z., LUO H., WANG C., ZHAO F., GILAL A.R., and SHAO W. Task Assignment Model for Crowdsourcing Software Development. *Journal of Information Processing Systems*, 2018, 14(3): 621-630.
[20] KANG Q., and TAY W.P. Task Recommendation in Crowdsourcing Based on Learning Preferences and Reliabilities. *IEEE Transactions on Services Computing*, 2018.
[21] TUNIO M., LUO H., and WANG C. Impact of Personality on Task Selection in Crowdsourcing Software Development: A Sorting Approach. *IEEE Access*, 2017, 5: 18287-18294.
[22] KAMANGAR Z., KAMANGAR U., ALI Q., FARAH I., NIZAMANI S., and ALI T.H. To Enhance Effectiveness of Crowdsource Software Testing by applying Personality Types. *Proceedings of the 8th International Conference on Software and Information Engineering*, 2019: 15-19.
[23] KAMANGAR Z.U., SIDDIQUI I.F., ALI Q., KAMANGAR U.A., and KHOWAJA S.S. To Improve Software Testing in Crowd-based Outsourcing Environment by Relating Levels of Testing with Personality Characteristics. *Proceedings of the 12th International Conference on Internet*, 2020.
[24] ZAINAB U. KAMANGAR, SIDDIQUI I.F., ARAIN Q.A., KAMANGAR U.A., and QURESHI N.M.F. Personality Character-based Enhanced Software Testing Levels for Crowd Outsourcing Environment. *KSII Transactions on Internet and Information Systems*, 2021, 15(8): 2974-2992.
[25] CHARBEL OBEID, INAYA LAHOUD, HICHAM KHOURY, and CHAMPIN P.-A. Ontology-based Recommender System in Higher Education. *Companion Proceedings of the Web Conference 2018*, 2018: 1031-1034.
[26] TARUS J., NIU Z., and TARUS G.M. Knowledge-based recommendation: a review of ontology-based recommender systems for e-learning. *Artificial Intelligence Review*, 2017, 50: 21-48.
[27] YANES N., SASSI S., and GHEZALA H. Ontology-based recommender system for COTS components. *Journal of Systems and Software*, 2017, 132: 283-297.
[28] WILLIAMS I. Ontology Based Collaborative Recommender System for Security Requirements Elicitation. *Proceedings of IEEE 26th International Requirements Engineering Conference*, 2018.
[29] IBRAHIM M., YANG Y., NDZI D., YANG G., and ALMALIKI M. Ontology-Based Personalized Course Recommendation Framework. *IEEE Access*, 2018, 7: 5180-5199.
[30] NEETHUKRISHNAN K.V., and SWARAJ K.P. Ontology based research paper recommendation using personal ontology similarity method. *The 2nd International Conference on Electrical, Computer and Communication Technologies*, 2017: 1-4.
[31] SIDDIQUI I.F., LEE S.U.-J., and ABBAS A. A Novel Knowledge-Based Battery Drain Reducer for Smart Meters. *Intelligent Automation and Soft Computing*, 2020, 26(1): 107-119.
[32] SIDDIQUI I.F., QURESHI N.F., SHAIKH M.A., CHOWDHRY B.S., ABBAS A., BASHIR A.K., and LEE S.U.-J. Stuck-at Fault Analytics of IoT Devices Using Knowledge-based Data Processing Strategy in Smart Grid. *Wireless Personal Communications*, 2019, 106(4): 1969-1983.

参考文:

[1] BERNERS-LEE T.、HENDLER J. 和 LASILA O. 語義網。科學美國人，2001，284：34-43。
[2] ANTONIOU G.、GROTH P.、VAN HARMELEN F. 和 HOEKSTRA R. 語義網絡入門。第 3 版。麻省理工學院出版社，馬薩諸塞州劍橋市。2012 年。
[3] MAN D. 計算機科學中的本體論。數學教義，2013，31(1): 43-46。
[4] HOWE J. 眾包的興起。連線雜誌，2006 年，14(6)：1-4。
[5] MAO K.、CAPRA L.、HARMAN M. 和 JIA Y. 軟件工程中眾包使用的調查。系統與軟件雜誌，2017 年，126：57-84。
[6] BU Q. 、SIMPERL E. 、CHAPMAN A. 和 MADDALENA E. 眾包分類任務 中的質量評估。國際人群科學雜誌，2019 年，3(3)：222-248。
[7] GONCALVES J., FELDMAN M., SUBINGQIAN H., 和 KOSTAKOS V. 基於認知能力的眾包任務路由和分配，2017 年第 26 屆國際萬維網會議論文集。DOI: 10.1145/304102128055

[8] DURWARD D.、BLOHM I. 和 LEIMEISTER J.M. 人群工作的性質及其對個人工作認知的影響。管理信息系統雜誌, 2020, 37(1): 66-95。

[9] XING Y., WANG L., 和 LI Z. 穩定良好的多屬性眾包任務分配. 電氣和電子工程師學會. 訪問，2019，7：133351-133361。

[10] TRAN L.、TO H.、FAN L. 和 SHAHABI C. 超局部空間眾包中任務分配的實時框架。計算機協會智能係統與技術彙刊，2018 年，9(3)：1-26。

[11] ALIREZA S.、SHAFIGHEH H. 和 MASOUD R.A. 基於社交網絡用戶畫像和人格五因素模型的人格分類。以人為本的計算與信息科學, 2018, 8(1): 24-38.

[12] SHI X.、EVANS R.、PAN W. 和 SHAN W. 了解個性特徵對眾包社區中求解器參與度的影響：經過調節的調解調查。信息技術與人，2021 年。

[13] GILAL A.R. 、 TUNIO M.Z. 、 WAQAS A. 和 ALMOMANI M.A. 任務分配和個性：眾包軟件開發。全球軟件工程中的人為因素，2019，1:1-19。

[14] PATHAN N.、ALI Q.、IFTIKHAR S.、BATOOL G. 和 MEMON I. 使用眾包的個性類型推薦系統。 2019 年計算、數學和工程技術國際會議論文集。

[15] YADAV A.、CHANDRA J. 和 SAIRAM A.S. 眾包環境的預算和截止日期感知任務分配方案。 電氣和電子工程師學會. 計算新興主題交易, 2021。

[16] 趙 Y., ZHENG K., YIN H., LIU G., FANG J., 和 ZHOU X. 空間眾包中的偏好感知任務分配：從個人到團體。 電氣和電子工程師學會. 知識與數據工程彙刊，2020 年。

[17] TA H.H. 評估眾包對物流和供應鏈運營的影響。論文和學位論文，2018 年。

[18] HETTIACHI D.、VAN BERKEL N.、HOSIO S.、KOSTAKOS V. 和 GONCALVES J. 認知能力對眾包任務績效的影響。第 17 屆國際信息處理聯合會國際會議論文集，第一部分。計算機科學叢書講義，11746：442-464。

[19] TUNIO M.Z.、LUO H.、WANG C.、ZHAO F.、GILAL A.R. 和 SHAO W. 眾包軟件開發的任務分配模型。信息處理系統雜誌, 2018, 14(3): 621-630。

[20] KANG Q., 和 TAY W.P. 基於學習偏好和可靠性的眾包任務推薦。 電氣和電子工程師學會. 服務計算彙刊，2018 年。

[21] TUNIO M.、LUO H. 和 WANG C. 個性對眾包軟件開發中任務選擇的影響：一種排序方法。電氣和電子工程師學會. 訪問，2017，5：18287-18294。

[22] KAMANGAR Z.、KAMANGAR U.、ALI Q.、FARAH I.、NIZAMANI S. 和 ALI T.H. 通過應用個性類型來提高眾包軟件測試的有效性。第八屆軟件與信息工程國際會議論文集, 2019: 15-19.

[23] KAMANGAR Z.U.、SIDDIQUI I.F.、ALI Q.、KAMANGAR U.A. 和 KHOWAJA S.S. 通過將測試水平與個性特徵相關聯來改進基於人群的外包環境中的軟件測試。2020 年第 12 屆互聯網國際會議論文集。

[24] ZAINAB U. KAMANGAR、SIDDIQUI I.F.、ARAIN Q.A.、KAMANGAR U.A. 和 QURESHI N.M.F. 面向人群外包環境的基於個性特徵的增強軟件測試級別。KSII 互聯網和信息系統交易，2021，15(8)：2974-2992。

[25] CHARBEL OBEID、INAYA LAHOUD、HICHAM KHOURY 和 CHAMPIN P.-A. 高等教育中基於本體的推薦系統。2018、2018 年網絡會議的配套論文集：1031-1034。

[26] TARUS J.、NIU Z. 和 TARUS G.M. 基於知識的推薦：用於電子學習的基於本體的推薦系統綜述。人工智能評論，2017，50：21-48。

[27] YANES N.、SASSI S. 和 GHEZALA H. 用於商業現貨 組件的基於本體的推薦系統。系統與軟件雜誌，2017 年，132：283-297。

[28] WILLIAMS I. 基於本體的安全需求獲取協作推薦系統。2018 年 電氣和電子工程師學會. 第 26 屆國際需求工程會議論文集。

[29] IBRAHIM M.、YANG Y.、NDZI D.、YANG G. 和 ALMALIKI M. 基於本體的個性化課程推薦框架。 電氣和電子工程師學會. 訪問，2018 年，7：5180-5199。

[30] NEETHUKRISHNAN K.V. 和 SWARAJ K.P. 使用個人本體相似度方法的基於本體的研究論文推薦。第二屆電氣、計算機和通信技術國際會議，2017：1-4。

[31] SIDDIQUI I.F.、LEE S.U.-J. 和 ABBAS A. 一種基於知識的新型智能電錶電池排水減速器。智能自動化與軟計算, 2020, 26(1): 107-119.

[32] SIDDIQUI I.F.、QURESHI N.F.、SHAIKH M.A.、CHOWDHRY B.S.、ABBAS A.、BASHIR A.K. 和 LEE S.U.-J. 在智能電網中使用基於知識的數據處理策略對物聯網設備進行故障分析。無線個人通信，2019，106(4): 1969-1983。