

# Toward Crowdsourced Knowledge Graph Construction: Interleaving Collection and Verification of Triples

Helun Bu<sup>1</sup> <sup>a</sup> and Kazuhiro Kuwabara<sup>2</sup> <sup>b</sup>

<sup>1</sup>Graduate School of Information Science and Engineering, Ritsumeikan University,  
1-1-1 Noji Higashi, Kusatsu, Shiga 525-8577, Japan

<sup>2</sup>College of Information Science and Engineering, Ritsumeikan University,  
1-1-1 Noji Higashi, Kusatsu, Shiga 525-8577, Japan

**Keywords:** Knowledge Graph Construction, Crowdsourcing, Knowledge Collection, Knowledge Verification.

**Abstract:** This paper presents a method for building a knowledge graph using crowdsourcing. The collection and verification of pieces of knowledge are essential components of building a high-quality knowledge graph. We introduce fill-in-the-blank-type of quizzes to collect knowledge as triples and true-or-false-type quizzes to verify the collected triples. We also present score functions to evaluate and select a quiz for efficient knowledge graph construction based on the workers' past inputs. The collection and verification processes are dynamically interleaved using weights in the score function. Simulation results show that the proposed approach can collect and verify distributed knowledge among casual workers.

## 1 INTRODUCTION

Information on the Web has become more diverse, with more online content containing machine-readable metadata. With the development of machine learning technologies, knowledge can be extracted from the data available on the Internet. Knowledge graphs is a promising technique for storing and communicating real-world knowledge with nodes representing entities and edges representing relationships between entities (Hogan et al., 2021). Knowledge graph have been utilized in various applications, such as query answering (Yang et al., 2014).


Building a high-quality knowledge graph requires knowledge collection and verification, which often involves human intervention. Crowdsourcing is a promising method for building a knowledge graph from the knowledge of many casual users (Cao et al., 2021).


When we apply crowdsourcing to knowledge graph construction, the amount and quality of knowledge are major issues. For the former (the amount of knowledge), it is important to collect the pieces of knowledge each user has and merge them into a larger knowledge graph. However, as pieces of knowledge collected from a user may not be correct, we need to

verify them to attain a high-quality knowledge graph. In this sense, there are two different tasks for knowledge graph construction: collection and verification. It is necessary to balance these properly. For example, we may prioritize collection tasks and conduct verification tasks only after a threshold amount of knowledge pieces are obtained. Alternatively, we may verify a piece of knowledge when it is obtained. With crowdsourcing, the tasks assigned to users need to be carefully selected.

Here, we examine a case in which a task is represented as a game-like quiz. For example, a *fill-in-the-blank* quiz is used for knowledge collection (Bu and Kuwabara, 2021a), and a *true-or-false* quiz is used for knowledge verification (Bu and Kuwabara, 2021b). In such a crowd sourced knowledge graph construction process, task assignment to users—in other words, what kind of quiz should be presented to each user—is important to efficiently extract knowledge from a large number of users who potentially have different partial knowledge.

In this paper, we present a method of dynamically interleaving the knowledge collection and verification processes by introducing a score function for each task (quiz). We conduct simulation experiments to examine how the different score functions affect the efficiency of the overall knowledge construction process.

<sup>a</sup>  <https://orcid.org/0000-0001-9537-0757>

<sup>b</sup>  <https://orcid.org/0000-0003-3493-1076>

The remainder of the paper is organized as follows. Section 2 describes the related work. Section 3 presents our proposed approach to knowledge graph construction. Section 4 examines the characteristics of the proposed approach using the experimental results. Section 5 concludes the paper and discusses future work.

## 2 RELATED WORK

One of the main advantages of structuring human knowledge in large scale graphs is the flexibility of the schema. In particular, the inference of knowledge graphs can be represented by labels with descriptions of the relationships between entities in standard representation formats such as RDF, RDFS and OWL (Tiddi and Schlobach, 2021). In this context, several approaches to dynamically creating knowledge bases using RDF data have been proposed. For example, there is a system for users to dynamically incorporate web services that describe facts about an entity or topic in a paradigm called “Active Knowledge” into a dynamic RDF knowledge base (Preda et al., 2010). RDF models have also been created in the field of clinical pharmacogenetics to use semantic knowledge bases to manage and solve quality-related problems with complex and large amounts of data used in drug selection and dosing (Samwald et al., 2013).

One effective approach to building a knowledge base is crowdsourcing, where casual users collaborate without the need for experts. For example, a knowledge base of urban emergencies was built from social media data using a crowdsourcing framework that considered performance and effectiveness (Xu et al., 2016). There has also been a crowdsourcing approach proposed using mobile applications to collect human subjective knowledge to support human decision making (Hosio et al., 2016). However, quality of knowledge can be a significant issue when dealing with subjective human knowledge. Crowdsourcing has been utilized to check the validity of fake news and alternative facts (Sethi, 2017).

In addition, to solve the motivation problem of crowdsourcing, it is important to provide workers with appropriate task choices that take into account the worker’s performance. One study suggested creating a list of tasks using the worker’s past task preferences and performance and presenting this list to the worker at the task selection stage (Yuen et al., 2011).

In this study, we propose a process that oversees tasks to collect pieces of knowledge to build a knowledge graph and tasks to verify this knowledge. In the proposed process, user performance is predicted

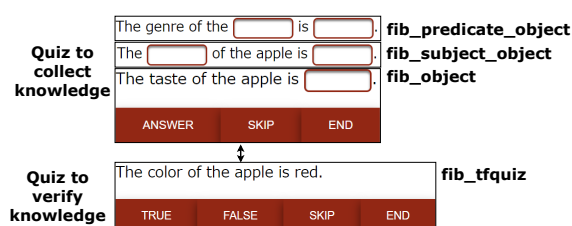


Figure 1: Image of the proposed quiz.

based on their knowledge areas and past input history.

## 3 PROPOSED APPROACH

We regard a knowledge graph as composed of RDF triples: subject, predicate and object. The knowledge collection process corresponds to collecting triples, and the knowledge verification process corresponds to verifying the collected triples. In the triple collection process, when knowledge is distributed among many users, a prediction algorithm is introduced to select a task (quiz) to efficiently collect knowledge from a specific user.

The user is presented with different types of quizzes in the form of *fill-in-the-blank* tasks. To increase the reliability of the collected knowledge content, the collected triples are first stored in the *temporary* knowledge base. A *true-or-false* quiz is introduced to verify the collected triples, and only verified triples are moved into the *formal* knowledge base.

The knowledge collection and verification processes are essentially independent. The verification process may start after all potential triples are accumulated in the temporary knowledge base. However, by interleaving knowledge collection and verification, efficiency can be improved.

To achieve this, it is necessary not only to carefully select a quiz for a user so that users with different partial knowledge can provide a piece of knowledge but also to decide when the system is in knowledge collection mode and when it is in knowledge verification mode. In this sense, we aim to clarify a way to interleave the collection and verification processes when selecting a task (quiz) for each user.

### 3.1 Game Design

The proposed system introduces three types of quizzes for knowledge collection, and one type of quiz for knowledge verification. As shown in Figure 1, the quizzes for collecting knowledge are variants of *fill-in-the-blank* (*fib*) quizzes. These quizzes are generally presented to the user with a question text

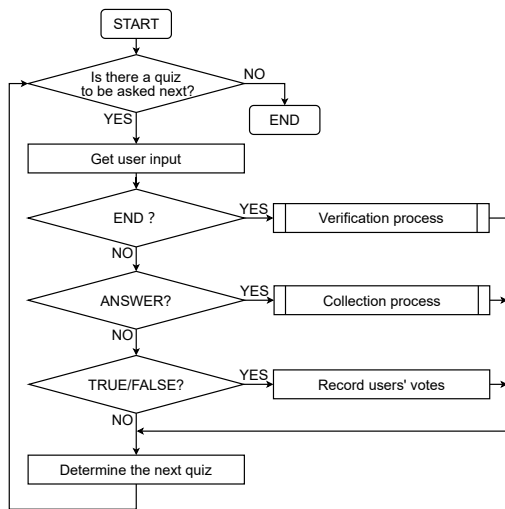


Figure 2: Proposed system main processing flow.

containing a triple with one or two items missing. By answering the quiz questions, parts of the knowledge are collected by the system as triples.

For example, the system checks whether all object items are present for any subject and predicate pair in the temporary knowledge base. If an object item does not exist, a *fib\_object* quiz is used to request an object item that corresponds to the pair of subject and predicate. Alternatively, to add a new word as a subject item, a *fib\_subject\_object* quiz is selected, and the user is asked to answer the pair of subject and object. Finally, a *fib\_predicate\_object* quiz asks the user to answer a pair of a predicate and an object to add a new predicate item.

When the game starts, choices of ANSWER, SKIP and END are displayed along with a blank field(s). When the user enters the answer to the quiz and send it to the server by selecting ANSWER, the system stores the user’s response and presents the next quiz. If the user does not know the answer to the quiz, they can select SKIP to jump to the next quiz. If the user selects END, the game session ends. In this process, the collected triples are first placed in the temporary knowledge base.

The *fib\_tfquiz* is for verifying triples. For this type of quiz, the user is presented with a quiz containing a triple, and the user has to choose between two options: YES or NO.

### 3.2 System Flow

The main flow of the proposed approach is shown in Figure 2. After the user starts the game, a quiz containing either knowledge collection or verification is presented to the user.

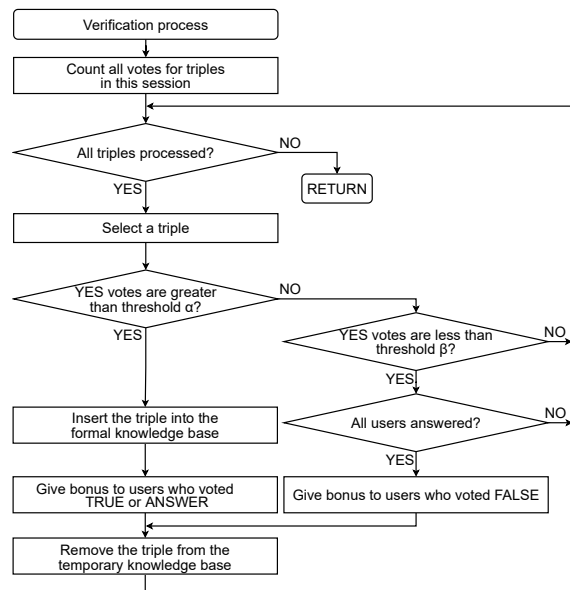


Figure 3: Verification process flow.

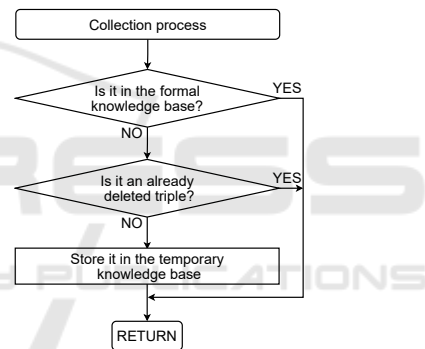


Figure 4: Collection process flow.

If the user selects END, the game session terminates. When the game session is finished, the verification process starts. The verification process takes the form of a majority vote to check whether the triples that the user has answered in this session have received a sufficient number of votes, according to a threshold  $\alpha$  (Figure 3). If the triple has accumulated more than or equal to  $\alpha$  votes, it is treated as *true*, moved into the formal knowledge base and then removed from the temporary knowledge base.

Conversely, after all users have voted, if the triple still has fewer than or equal to  $\beta$  votes, the triple is treated as *false* and removed from the temporary knowledge base.

When the user answers a knowledge collection quiz (by selecting ANSWER with the answers), the knowledge collection process stores the triple if it is not already in the formal knowledge base and has not been previously deleted from the temporary knowl-

edge base (Figure 4).

For the knowledge verification quiz, we record the number of votes for triples that the user has verified with TRUE or FALSE.

### 3.3 Score Function

To facilitate selecting the next quiz to present, we define the score for each quiz type as follows. Let  $S_i$  be a subject of triples in the knowledge base,  $P_j$  be a predicate, and  $O_l$  be an object. Let  $Q_{i,j}$  represent a knowledge collection (*fib\_object*) quiz about  $S_i$  and  $P_j$ , and let  $Q_j$  represent a *fib\_subject\_object* quiz regarding predicate  $P_j$ . Further, let  $Q_i$  represent a *fib\_predicate\_object* quiz regarding subject  $S_i$ , and  $Q_{i,j,l}$  represent a knowledge verification quiz about a triple of  $(S_i, P_j, O_l)$ . In addition, let  $b_i$  be the number of missing subject items for a given subject item  $S_i$ , and  $b_j$  be the number of missing object items for a given predicate item  $P_j$ . The answer history of the knowledge collection quiz for user  $u_k$  is represented by  $c_{i,j,k}$  and is updated as follows:

$$c_{i,j,k} \leftarrow \begin{cases} c_{i,j,k} + 1 & \text{when } u_k \text{ responds with ANSWER for the quiz } Q_i, Q_j, \text{ or } Q_{i,j} \\ c_{i,j,k} - 1 & \text{when } u_k \text{ responds with SKIP for the quiz } Q_i, Q_j, \text{ or } Q_{i,j} \end{cases}$$

That is,  $c_{i,j,k}$  represents how many  $S_i$  and  $P_j$  pairs the user  $u_k$  may know based on  $u_k$ 's answer history to the knowledge collection quizzes.

For the knowledge verification quiz, the answer histories of both the collection and verification quizzes are considered. The score of the verification quiz  $Q_{i,j,l,k}$  for a triple of subject  $S_i$ , predicate  $P_j$  and object  $O_l$  and user  $u_k$  is calculated as the sum of all previous responses by user  $u_k$  as follows:

$$t_{i,j,l,k} \leftarrow \begin{cases} t_{i,j,l,k} + 1 & \text{when } u_k \text{ responds with ANSWER for } Q_{i,j}, Q_i, \text{ or } Q_j \text{ or when } u_k \text{ responds with TRUE or FALSE for } Q_{i,j,l} \\ t_{i,j,l,k} & \text{when } u_k \text{ responds with SKIP for the quiz but already responded ANSWER for the corresponding item in } Q_i, Q_j, Q_{i,j} \text{ or } Q_{i,j,l} \\ t_{i,j,l,k} - 1 & \text{when } u_k \text{ responds with SKIP for the quiz and there is no ANSWER for the corresponding item in } Q_i, Q_j, Q_{i,j} \text{ and } Q_{i,j,l} \end{cases}$$

For each *type* of quiz, score functions  $SC_{type}$  are defined as follows:

$$\begin{aligned} SC_{obj}(Q_{i,j}, u_k) &= \sum_{s \in S} c_{s,j,k} + \sum_{p \in P} c_{i,p,k} + b_i \\ SC_{sub\_obj}(Q_j, u_k) &= \sum_{s \in S} c_{s,j,k} + \max_j b_j - b_j \\ SC_{pred\_obj}(Q_i, u_k) &= \sum_{p \in P} c_{i,p,k} + \max_i b_i - b_i \\ SC_{ifquiz}(Q_{i,j,l}, u_k) &= \left( \sum_{p \in P, o \in O} t_{i,p,o,k} + \sum_{s \in S, o \in O} t_{s,j,o,k} \right. \\ &\quad \left. + \sum_{s \in S, p \in P} t_{s,p,l,k} \right) * w \end{aligned}$$

Here,  $w$  specifies how much priority the verification quiz should be given compared to the collection quiz. Each time we select the next quiz to present to the user, the quiz with the highest score is chosen. If there are multiple quizzes with the same highest score, one of them is selected randomly.

## 4 EXPERIMENTS

Since it is difficult to experiment under different conditions with many human users, we conducted simulation experiments, in which BOT programs corresponding to virtual users were run on the simulation system. We assume  $N$  virtual users and  $M$  distinct triples (excluding duplicates), with triples distributed among virtual users. We also assume that there may be more than one object for each pair of subject and predicate in the triples held by the virtual user. The server in the simulation system is assumed to initially have only one subject and predicate, and no corresponding object.

The BOT program that implements a virtual user has triples assigned to the corresponding virtual user. It receives a quiz sent from the server, looks for triples related to the quizzes, and sends them to the server in the form of entering "answers" or selecting "true/false" to the "questions" in the quizzes. The rules for operating a BOT program for a virtual user are as follows:

1. For a knowledge collection quiz, if a virtual user has knowledge about the triple associated with the quiz, they will always reply with ANSWER.
2. For knowledge verification quizzes, if a virtual user has the same triples as the triple in the quiz, they will always vote TRUE. If the virtual user's triples differ from the triples in the quiz, the virtual user will vote FALSE.



3. For all quiz types, if the virtual user does not have the relevant knowledge, they will always reply with SKIP and ask for the next quiz.
4. If the virtual user sends all of their triples to the server, they will not send ANSWER for the knowledge collection quiz, even if they have knowledge of the relevant triples, but will instead send SKIP.
5. In one game session, the virtual user has to answer  $Q$  quizzes given by the server. At the end of one game session, the virtual user sends END. Here, we set  $Q$  to 5.

The rules for the server side to send quizzes to BOTs (virtual users) are as follows:

1. If there are no quizzes to be presented to the user, the simulation ends.
2. If triples have been collected from all users into the temporary knowledge base, and all triples in the temporary knowledge base have been verified, and the maximum score of the candidates for the next quiz is less than or equal to zero, the simulation ends.
3. The same quiz will not be presented to the same user twice if the user has answered with SKIP.
4. A triple collected by a knowledge collection quiz is treated the same as one vote of TRUE in the knowledge verification quiz. Thus, the collected triple will not be submitted as the knowledge verification quiz.
5. The server initially has one subject item and one predicate item among the  $M$  pre-populated triples, but the server does not have the object item for this subject and predicate.
6. The only triples stored in the temporary knowledge base are the triples collected from the knowledge collection quiz. Triples collected from other sources are not considered. Therefore, the triples in the quiz to be verified will always correspond to the ones that were answered with ANSWER in the collection quiz.

To investigate the characteristics of the proposed approach, we conducted two experiments, as described in the following subsections.

#### 4.1 Collection and Verification

The purpose of this experiment was to demonstrate how interleaving the collection and verification of knowledge is possible by changing the value of  $w$  in the proposed approach while obtaining triples that are distributed among many virtual users.

Table 1: The triples user ( $u_1$ ) has.

	<i>genre</i>	<i>p2</i>	<i>p3</i>	<i>p4</i>
<i>s1</i>	<i>g1</i>	<i>o2</i>	<i>o3</i>	<i>o4</i>
<i>s2</i>	<i>g1</i>	<i>o5</i>	<i>o6</i>	<i>o7</i>
<i>s3</i>	<i>g1</i>	<i>o8</i>	<i>o9</i>	<i>o10</i>

##### 4.1.1 Simulation Parameters

In this experiment, we prepared 60 triples ( $M = 60$ ) and 100 virtual users ( $N = 100$ ). In terms of triples, we set the number of subject items to 15, and the number of predicate items to 4, comprising 60 triples. The virtual users were divided into 5 groups, each of which had 20 virtual users. Virtual users in the same group were assumed to have the same triples. Each virtual user was assumed to have 12 triples with 3 subject items and 4 predicate items. Table 1 shows the triples of an example virtual user. We regard the predicate *genre* contains *genre* or a knowledge area of the triple. The triples a virtual user has have the same object value for predicate *genre*, meaning that the virtual user has knowledge about a certain genre (knowledge area).

The threshold  $\alpha$  was set to 16, and the threshold  $\beta$  was set to 4. That is, a triple needs to obtain 16 or more votes from the virtual users (in this particular experiment, from the same group) to be considered *true* and moved to the formal knowledge base. Further, if only 4 or fewer votes are obtained, the corresponding triple is treated as *false* and removed from the temporary knowledge base.

The objectives of this experiment is to verify that  $M = 60$  triples can be acquired in the temporary knowledge base by running the quiz game with  $N = 100$  virtual users, and that  $M = 60$  exact triples can be verified and moved into the formal knowledge base. We also varied the weight ( $w = 1, 10, 100$ ) that controls how much the verification quiz is prioritized to examine the effect of the weight on the knowledge collection and verification processes.

##### 4.1.2 Simulation Results and Discussions

Figure 5 shows how many triples were collected and verified for different weight values. The horizontal axis represents the number of game sessions, and the vertical axis represents the number of triples. The solid line represents the changes in the number of verified triples in the *formal* knowledge base, and the dotted line represents the changes in the number of collected triples in the *temporary* knowledge base.

As can be seen in the graph, for all values of  $w$ , all triples were collected from the virtual user and verified. However, there were differences in the speed of

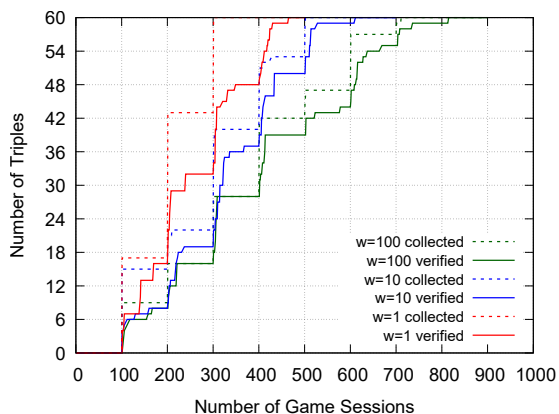


Figure 5: Collected and verified triples.

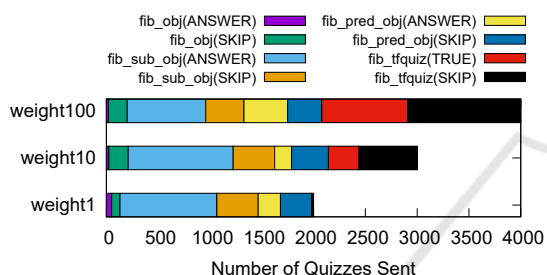


Figure 6: Breakdown of users’ answers.

collection and verification. Since both the number of collected triples and the number of verified triples increased as the game session proceeds, collection and verification were performed in parallel. More specifically, as the number of collected triples increases, the score for the verification quiz increases, resulting in a higher priority for verification.

The breakdown of the users’ answers to each type of quiz is shown in Figure 6. From the figure, the weight  $w$  can control the ratio between the collection quizzes and the verification quizzes; smaller  $w$  values result in fewer verification quizzes. The reason the number of verification quizzes in the case of  $w = 1$  is less than that of  $w = 100$  can be explained as follows: With a lower  $w$  value, the number of verification quizzes presented to the user is reduced and the collection quizzes effectively verify the triples by receiving the same triples as the responses to the collection quizzes.

In addition, for the  $w = 100$  case, the number of verification quizzes answered with SKIP is larger than the cases with a lower  $w$  value. As SKIP answers to quizzes are a cause of inefficiency, decreasing the number of SKIP responses is a future challenge.

Table 2: The triples user ( $u_1$ ) has (2nd experiment).

	$p1$	$p2$	$p3$	$p4$	$p5$
$s1$	$o_{10}$	$o_{10}$	$o_{10}$	$o_{10}$	$o_{10}$
	$o_8$	$o_8$	$o_8$	$o_8$	
	$o_6$	$o_6$	$o_6$		
	$o_4$	$o_4$			
	$o_2$				

## 4.2 Thresholds in Verification

In the second experiment, we checked if the threshold of verifying triples reflects the number of users that have the target triple. We assume that triples are distributed among virtual users, and the triples that many users have are treated as *true*, and put into the formal knowledge base. The triples that only a few users have are treated as *false* and removed from the temporary knowledge base. Other triples remain in the temporary knowledge base. Here, the threshold is used to determine the truth of a collected triple based on the number of votes.

### 4.2.1 Simulation Parameters

We assumed that the triples were collected by a knowledge collection quiz such as a *fib\_object* type quiz. In this experiment, the number of triples,  $M$ , was set to 25. The number of subject items was set to 1, and the number of predicate items was set to 5. We prepared five different values of object for each subject and predicate pair. The total number of distinct triples used in the experiment was 25.

The number of virtual users,  $N$ , was set to 20. For each 5 distinct object values ( $o_{10}$ ,  $o_8$ ,  $o_6$ ,  $o_4$ , and  $o_2$ ), 100%, 80%, 60%, 40%, and 20% of users were respectively assumed to have the same triple that contains a certain object value. For example, all 20 virtual users have a triple of ( $s1$ ,  $p1$ ,  $o_{10}$ ), 16 users have a triple of ( $s1$ ,  $p1$ ,  $o_8$ ), and so on. Table 2 shows the triples a virtual user has. We distributed triples to virtual users so that each user had the same number of triples.

The value of threshold  $\alpha$  was set to 16. As the number of virtual users was  $N = 20$ , triples that 80% or more of the virtual users have would be considered *true*. The value of threshold  $\beta$  was set to 4, meaning that triples that 20% or less of the virtual users have would be considered *false*. The value of  $w$ , which determines the priority of verification quizzes, was set to 1, 10, and 100, as in the first experiment.

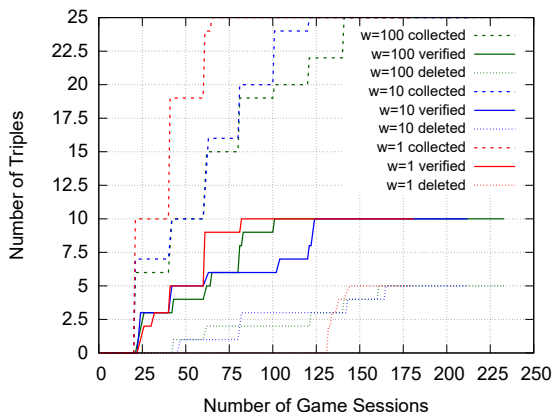


Figure 7: Collected and verified triples (2nd experiment).

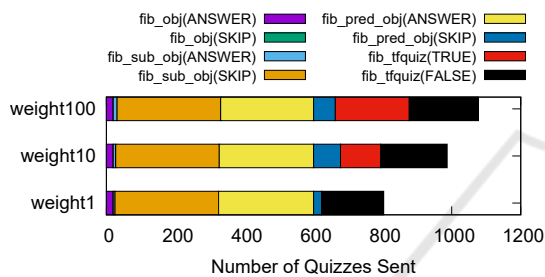


Figure 8: Breakdown of users' answers (2nd experiment).

#### 4.2.2 Simulation Results and Discussion

As shown in Figure 7 and 8, the results of the experiment with 20 virtual users show that out of 25 triples, all 10 triples that 80% and 100% of the users have in common were moved into the formal knowledge base. By contrast, all 5 triples that only 20% of users had were considered false and removed from the temporary knowledge base, while the other triples remained in the temporary knowledge base. Note that users' responses of FALSE for *true-or-false* quizzes were observed for all the cases (Figure 8). In this way, only the triples possessed by more users than the threshold  $\alpha$  were verified, and triples possessed by fewer users than the threshold  $\beta$  were eventually removed from the temporary knowledge base.

We also found that the system effectively interleaved the collection and verification of knowledge even when multiple object items existed for a pair of subject and predicate.

## 5 CONCLUSION

In this paper, we presented an approach to perform interleaved collection and verification of triples to build a knowledge graph using crowdsourcing. In this ap-

proach, quizzes were introduced to collect and verify triples that constitute a knowledge graph: *fill-in-the-blank* quizzes for knowledge collection and a *true-or-false* quiz for knowledge verification.

Score functions based on the user's history were adopted to improve knowledge graph building efficiency. To interleave the collection and verification processes, we also introduced a weight in score function calculations. The simulation results show how weight can influence the performance of the collection and verification of knowledge. In addition, the verification threshold works reasonably when a majority rule is adopted.

Since currently only triples collected in the knowledge collecting quizzes are considered for the verification task, we plan to incorporate triples collected from a variety of other sources into the target of the verification task in future work. Furthermore, it would be beneficial to expand the gamification elements to motivate users. We plan to examine how different ways of giving rewards to users affect the collection and verification of triples. For example, we are considering offering multiple tasks with different rewards to the user and letting the user choose one of them for task execution.

## ACKNOWLEDGEMENTS

This work was partially supported by JSPS KAKENHI Grant Number 18K11451.

## REFERENCES

Bu, H. and Kuwabara, K. (2021a). Task selection based on worker performance prediction in gamified crowdsourcing. In Jezic, G., Chen-Burger, J., Kusek, M., Sperka, R., Howlett, R. J., and Jain, L. C., editors, *Agents and Multi-Agent Systems: Technologies and Applications 2021*, pages 65–75, Singapore. Springer Singapore.

Bu, H. and Kuwabara, K. (2021b). Validating knowledge contents with blockchain-assisted gamified crowdsourcing. *Vietnam Journal of Computer Science*, pages 1–21.

Cao, M., Zhang, J., Xu, S., and Ying, Z. (2021). Knowledge graphs meet crowdsourcing: A brief survey. In Qi, L., Khosravi, M. R., Xu, X., Zhang, Y., and Menon, V. G., editors, *Cloud Computing*, pages 3–17, Cham. Springer International Publishing.

Hogan, A., Blomqvist, E., Cochez, M., D'amato, C., Melo, G. D., Gutierrez, C., Kirrane, S., Gayo, J. E. L., Navigli, R., Neumaier, S., Ngomo, A.-C. N., Polleres, A., Rashid, S. M., Rula, A., Schmelzeisen, L., Sequeda,

- J., Staab, S., and Zimmermann, A. (2021). Knowledge graphs. *ACM Comput. Surv.*, 54(4).
- Hosio, S., Goncalves, J., van Berkel, N., and Klakegg, S. (2016). Crowdsourcing situated & subjective knowledge for decision support. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, UbiComp '16, pages 1478–1483, New York, NY, USA. Association for Computing Machinery.
- Preda, N., Kasneci, G., Suchanek, F. M., Neumann, T., Yuan, W., and Weikum, G. (2010). Active knowledge: Dynamically enriching RDF knowledge bases by web services. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, SIGMOD '10, pages 399–410, New York, NY, USA. Association for Computing Machinery.
- Samwald, M., Freimuth, R., Luciano, J. S., Lin, S., Powers, R. L., Marshall, M. S., Adlassnig, K.-P., Dumontier, M., and Boyce, R. D. (2013). An RDF/OWL knowledge base for query answering and decision support in clinical pharmacogenetics. *Studies in health technology and informatics*, 192:539–542.
- Sethi, R. J. (2017). Crowdsourcing the verification of fake news and alternative facts. In *Proceedings of the 28th ACM Conference on Hypertext and Social Media*, HT '17, pages 315–316, New York, NY, USA. Association for Computing Machinery.
- Tiddi, I. and Schlobach, S. (2021). Knowledge graphs as tools for explainable machine learning: a survey. *Artificial Intelligence*, 103627.
- Xu, Z., Zhang, H., Hu, C., Mei, L., Xuan, J., Choo, K.-K. R., Sugumaran, V., and Zhu, Y. (2016). Building knowledge base of urban emergency events based on crowdsourcing of social media. *Concurrency and Computation: Practice and Experience*, 28(15):4038–4052.
- Yang, M., Ding, B., Chaudhuri, S., and Chakrabarti, K. (2014). Finding patterns in a knowledge base using keywords to compose table answers. *Proc. VLDB Endow.*, 7(14):1809–1820.
- Yuen, M.-C., King, I., and Leung, K.-S. (2011). Task matching in crowdsourcing. In *2011 International Conference on Internet of Things and 4th International Conference on Cyber, Physical and Social Computing*, pages 409–412.