IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

Constructing Knowledge Graphs for Online Collaborative Programming

YUANYI ZHEN[1], LANQIN ZHENG[1], and PENGHE CHEN[2]

[1] Faculty of Education, School of Educational Technology, Beijing Normal University, Beijing 100875, China
[2]Advanced Innovation Center for Future Education, Beijing Normal University, Beijing 100875, China

Corresponding author: Lanqin Zheng (e-mail: bnuzhenglq@bnu.edu.cn).

This work was supported by the National Natural Science Foundation of China (61907003) and the International Joint Research Project of Huiyan International College, Faculty of Education, Beijing Normal University (ICER202101).

ABSTRACT This study aimed to automatically construct knowledge graphs for online collaborative programming. We proposed several models and developed a system to construct knowledge graphs based on online discussion texts and the target knowledge graph for the C programming language. Our system included two main modules, namely, entity recognition and relation extraction. We proposed an innovative approach for recognizing knowledge entities, which included sequence tagging, text classification, and keyword matching. The extraction of relationships among knowledge entities was performed through queries of the target knowledge graph. The six kinds of knowledge graphs could be automatically generated through our method, including the activated and unactivated knowledge graphs of each student, each group, and each class. The accuracy of entity recognition reached 87.27%. The accuracies of relation extraction for students, groups, and the class achieved 89.7%, 90.4%, and 90.2%, respectively. This study is very promising and significant for both teachers and practitioners to provide interventions and personalized learning services based on the constructed knowledge graphs.

INDEX TERMS Knowledge graph, entity recognition, relation extraction, deep neural network model, online collaborative learning, collaborative programming.

I. INTRODUCTION

Online collaborative learning has been widely used in the field of education, especially during Corona Virus Disease 2019 (COVID-19). Online collaborative learning is also known as telecollaboration, which is defined as an educational endeavor that engages learners in different locations through networks and resources to learn together [1]. The benefits of online collaborative learning have been well documented. For example, online collaborative learning creates learning communities and provides multiple interactive opportunities with others anywhere and anytime [2]. Online collaborative learning contributes to improving problem-solving abilities, enhancing communication skills, increasing learning outcomes, and promoting intellectual curiosity [3].

Furthermore, learners often generate a large volume of discussion texts during online collaborative learning [4]. It has been pointed out that online discussion texts should be analyzed in real-time to detect the learners' perceptions, problems or difficulties [5]. However, previous studies analyzed online discussion texts manually through content analysis methods [6], lag sequential analysis [7], or epistemic network analysis [8]. It is very difficult to capture learners' knowledge gains, their knowledge building progress, and difficulties directly through manual analysis methods. Moreover, students often have difficulties in terms of establishing shared understanding [9] and engaging in learning [10] during online collaborative learning. Therefore, it is necessary to provide real-time analytics results to demonstrate the latest knowledge-building progress. Only in this way can teachers and instructors provide real-time and personalized support for learners to improve their learning performance.

To automatically analyze online discussion texts and to understand the learners' knowledge-building progress, natural language processing and knowledge graphs techniques have potential for facilitating the automatic generation of knowledge graphs. Knowledge graphs can demonstrate knowledge and their relationships [11] to visually represent the learners' knowledge-building progress and problems. Nevertheless, there is still a lack of studies on the automatic generation of knowledge graphs in the online

collaborative programming field. To close this research gap, this study aimed to automatically construct knowledge graphs based on online discussion texts during online collaborative programming. There are six types of knowledge graphs that can be constructed automatically in this study, including the activated knowledge graph of each student, the unactivated knowledge graph of each student, the activated knowledge graph of each group, the unactivated knowledge graph of each group, the activated knowledge graph of the whole class, and the unactivated knowledge graph of the whole class.

The main contributions of this paper are as follows:

- To the best of our knowledge, this is the first work that automatically constructed knowledge graphs based on online discussion transcripts during online collaborative programming.
- We proposed an innovative method of constructing knowledge graphs, including entity recognition through sequence tagging, text classification, and keyword matching as well as relation extraction through a query from the target knowledge graph.
- Six kinds of knowledge graphs could be automatically constructed through our method, including the activated and unactivated knowledge graphs of each student, each group, and each class. The constructed knowledge graphs provide important references for personalized learning support.
- We demonstrate an experiment by constructing a knowledge graph for C programming. The accuracies of entity recognition and relation extraction are very high and satisfactory.

The rest of this paper is organized as follows: Section 2 introduces the related work. Section 3 illustrated the proposed system and method. Section 4 demonstrated the experiment, results, and an exemplary case. Section 5 discussed the implications and limitations. Section 6 concludes the paper.

II. RELATED WORKS

A. ONLINE DISCUSSION TEXT ANALYSIS OF ONLINE COLLABORATIVE LEARNING

In the field of online collaborative learning, researchers sought to adopt qualitative analysis methods, quantitative analysis methods, and mixed analysis methods to analyze online discussion texts. Commonly used qualitative analysis methods in this field include conversation analysis, ethnographic analysis, and case studies [12]. The quantitative analysis method aims to create and test predictions about the relationships between collaborative learning processes and learning outcomes, such as the content analysis method and the social network analysis method [13]. Furthermore, the mixed-analysis method extracts the strengths of both the qualitative and quantitative analysis methods to provide a comprehensive analysis of the online collaborative learning process and outcomes.

Some studies attempted to analyze online discussion texts through machine learning methods. For example, Liu et al. [14] adopted the naive Bayes algorithm to automatically classify online discussion texts to analyze the teachers' reflective thinking. Xie et al. [15] used a logistic regression model and the adaptive boosting model to classify online discussion texts to identify opinion leaders. Wu et al. [16] employed a supervised machine learning algorithm to automatically analyze online discussion texts in large-scale online forums to predict academic performance.

However, Hadi et al. [17] addressed the concept that traditional machine learning depended heavily on human-designed features, making it difficult to capture semantic representations [18]. Furthermore, very few studies have adopted deep learning technology to automatically classify online discussion texts. Compared with traditional machine learning techniques, the use of deep learning techniques for text classification can eliminate the process of manual feature extraction and capture the contextual semantic relationships [19]. Therefore, this study adopted deep neural network models to automatically analyze online discussion texts in an online collaborative programming setting.

B. ONLINE COLLABORATIVE PROGRAMMING

Collaborative programming is an effective pedagogical approach, in which a group of learners complete the programming code and tasks together [20]. The purpose of online collaborative programming is to improve programming skills through online writing and revising programming code with peers [21]. It has been found that there are many advantages of online collaborative programming, including improving computational thinking competence [22], programming skills [23], and problem-solving abilities [24].

However, learners often have many difficulties during online collaborative programming. For example, students become less engaged in programming when they have difficulties in writing code [10]. Learners cannot achieve shared understanding and knowledge convergence during collaborative programming [9]. Moreover, group members cannot capture the latest progress during online collaborative programming. There is a lack of continuous inspection during programming, which leads to poor-quality programming skills [25].

In addition, learners generate a large amount of data during online collaborative programming. If there is a lack of real-time analysis, teachers and instructors cannot provide personalized guidance or intervention when students have difficulties in programming [21]. Therefore, there is an urgent need to automatically analyze online discussion texts during online collaborative programming. This study aimed to automatically construct knowledge

graphs for online collaborative programming to provide the latest progress in collaborative knowledge building.

C. KNOWLEDGE GRAPHS

The phrase "knowledge graph" was initially used in 1972 [26] and the modern incarnation of the phrase originates from the Google Knowledge Graph in 2012 [27, 28]. The knowledge graph is defined as a graph that represents real-world entities and their relationships [26]. Entities can be real-world objects and abstract concepts, and their relationships represent the relationships among the entities [29]. The main tasks of constructing knowledge graphs include entity recognition and relation extraction [29].

Entity recognition or named entity recognition (NER) aims to identify the mentions of named entities in a text [30], including people, organizations, locations, and other types [31]. Typically, there are three approaches to conduct entity recognition. The first approach is to adopt supervised methods to manually label all entities in a training data set [28, 32]. The second approach is the bootstrapping-based approach that utilizes seed examples of the entity to recognize entities [28, 33]. The third approach is the distant supervision, which adopts known entities in a knowledge graph as seed examples to recognize entities [28, 34]. To effectively recognize entities, different models have been developed and applied in previous studies. There are two types of mainstream models. One is the conditional random field (CRF) [35] and the other is the neural networks model [29, 36]. The CRF model has been widely used in terminology recognition [37] and Chinese entity recognition [29, 38]. In addition, the neural network model has been used to recognize entities, including the gated recurrent unit [39], long short-term memory (LSTM) [40], bidirectional LSTMs (BiLSTM) with a sequential conditional random layer above it [41], and transition-based parsing with states represented by stack LSTMs [42].

Relation extraction is used to extract unknown relational facts and add them into knowledge graphs [29]. Currently, most studies have adopted learning-based frameworks [43] to extract relations. The learning-based frameworks include supervised methods [44], bootstrapping [45], and weak supervision [46]. The neural relation extraction methods include convolutional neural networks (CNNs), recurrent neural networks (RNNs), attention mechanism, graph convolutional networks (GCNs), adversarial training, reinforcement learning, and others [29]. In addition, some researchers have adopted knowledge base completion [47] and refinement techniques [48] to detect missing relations. Furthermore, it has been found that a knowledge graph embedding-based link prediction method is a promising method, which can address knowledge graph incompleteness through predicting missing facts among entities [49].

Knowledge graphs have become a popular research direction and many topics emerged, including knowledge graph representation learning, knowledge acquisition and completion, temporal knowledge graph, and knowledge-aware applications [50]. In addition, knowledge graphs have been widely used in different fields, such as question answering [51], search [52], recommender systems [53], and machine reading comprehension [54]. A domain-specific knowledge graph is represented in terms of semantically interrelated entities and relations in a specific domain and domain-specific knowledge graph can be adopted to alleviate the problem of selecting resources to benefit teachers and learners [55]. Recently, knowledge graphs have been adopted in the field of education. For example, Chen et al. [36] proposed a KnowEdu system to automatically construct a knowledge graph for mathematics in middle school. Liu et al. [56] proposed three approaches to constructing directed concept graphs based on online courses from different providers, including a classification approach, a learning to rank approach, and a nearest-neighbor search approach. In addition, Chaplot and Koedinger [57] created a prerequisite structure graph based on educational material as well as student activity data, and they found that an unsupervised approach outperformed the supervised methods. Furthermore, Abu-Salih et al. [58] proposed a credibility-based domain-specific knowledge graph embedding framework to construct a domain-specific knowledge graph based on an extended politics domain augmented ontology.

However, there is a lack of studies on constructing knowledge graphs based on online discussion transcripts in online collaborative programming settings. This study aimed to close the research gap to construct knowledge graphs for online collaborative programming. The following sections will illustrate the proposed models and system.

III. SYSTEM

The purpose of the system is to generate the activated and unactivated knowledge graphs of each student, each group and the whole class (all groups) based on discussion transcripts during online collaborative programming. The construction of these knowledge graphs depends on the target knowledge graph. Before online collaborative programming, teachers drew a target knowledge graph to represent the expected knowledge gains. The target knowledge graph consists of all knowledge entities and their relationships. The types of knowledge entities and relationships were adapted from Yang [59]. There are five types of knowledge entities, namely, concepts, principles, processes, examples, and formats. There are three types of relationships, namely, "is a kind of", "has characteristics of", and "include". The target knowledge graph is stored in the Neo4j graph database.

A. SYSTEM OVERVIEW

Figure 1 shows the system architecture of constructing knowledge graphs for online collaborative programming. There are two modules in this system: the entity recognition module and the relation extraction module. The entity recognition module takes charge of recognizing the

knowledge entity from online discussion texts during online programming. Different entity recognition techniques, such as sequence tagging, text classification, and keyword matching are adopted to recognize the knowledge entity. The relation extraction module takes charge of extracting the relationships among the knowledge entities. Specifically, the activated and unactivated relationships can also be recognized through our system. After recognizing the knowledge entity and relations, knowledge graphs for C programming can be generated automatically to provide services for teachers, students, groups, and the whole class. In total, there are six kinds of knowledge graphs that can be automatically generated through the system, including the activated and unactivated knowledge graphs for each student, each group, and the whole class. In the following sections, we will describe the two modules in detail.

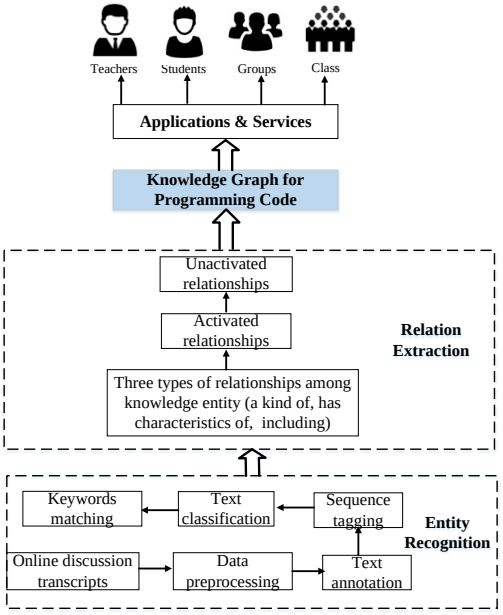


FIGURE 1.The system architecture of constructing knowledge graphs for online collaborative programming.

B. ENTITY RECOGNITION MODULE

In this study, we proposed an innovative method to recognize knowledge entities accurately. This method includes three steps, namely sequence tagging, text classification, and keyword matching. Figure 2 shows the entity recognition process. In the present study, there are two types of target knowledge entities: one is the directly matched knowledge entity, and the other is the semantically matched knowledge entity. Sequence tagging aims to recognize semantically matched knowledge entities. Text classification aims to recognize directly matched knowledge entities and irrelevant knowledge. Keyword matching aims to recognize directly matched knowledge entities. The following sections will illustrate each step in depth.

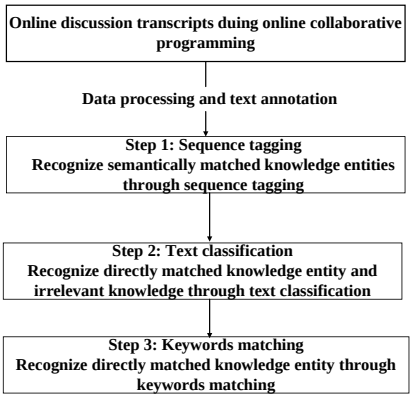


FIGURE 2. The entity recognition process.

1) SEQUENCE TAGGING

The first step of sequence tagging is to recognize semantically matched knowledge entities. The model is the Bidirectional Encoder Representations from the Transformers (BERT)-LSTM-CRF model. The BERT is designed to pretrain deep bidirectional representations from unlabeled texts and the pre-trained model can be fine-tuned with one additional output layer [60]. We integrated BERT, LSTM, and CRF to achieve the best performance. Figure 3 shows the architecture of the BERT-LSTM-CRF model. First, the sentence is represented as a sequence of vectors through the BERT embedding layer. The Bert model is mainly based on a transformer structure with an attention mechanism. The probability related to each word in the sentence is calculated by formula (1) [61]:

$$Attention(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (1)$$

where Q denotes query, K denotes keys, V denotes values, and d_k denotes the keys of the dimension. In this study, BERT-Base in Chinese was selected as the pretrained model. We adopted the BERT sentence vector as the text representation and the dimension of a single sentence vector was set as 768 dimensions. The sentence length can be adjusted by setting a sliding window.

Second, the embedding layer is given as an input from the LSTM layer. LSTM computes a representation of the sequence at each time point t. The LSTM memory cell is implemented based on [62]. Given the input sequence X= $\{x_1, x_2, \ldots, x_t\}$, at each time point t, there are input gate i_t, forget gate f_t and output gate o_t. The memory unit c_t controls the input and the forgetting of data through the different gate control units. The calculation formulas are as follows:

$$i_t = \sigma(W_i \cdot [h_{(t-1)}, x_t] + b_i) \quad (2)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_c) \quad (3)$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \quad (4)$$

$$f_t = \sigma(W_f \cdot [h_{(t-1)}, x_t] + b_f) \quad (5)$$

$$o_t = \sigma(W_o \cdot [h_{(t-1)}, x_t] + b_o) \quad (6)$$

$$h_t = o_t \cdot \tanh(C_t) \quad (7)$$

where σ is the sigmoid function; W_i, W_C, W_f, W_o are weight matrices; and b_i, b_c, b_f, b_o are bias units. In terms of our model, dropout is set as 0.5 in the LSTM layer. The model optimization function is Adam and the loss function is crf.loss_function. By adjusting the sliding window, the sentence length ranges from 20 to 500.

Finally, we adopted CRF [35] to calculate the conditional probability distribution of the labeling sequence according to the given prediction sequence, and then we used the Veterbi algorithm [63] to predict the maximum possible sequence as the labeling result.

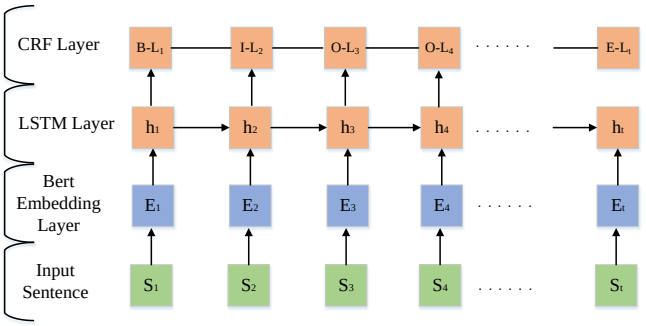


FIGURE 3. The architecture of the BERT-LSTM-CRF model.

2) TEXT CLASSIFICATION

The second step of entity recognition is to classify the directly matched knowledge entity and the irrelevant knowledge texts. Figure 4 shows the flow chart of the text classification. We tested and compared the performance of four types of classifiers, including BERT, BERT + random forest (RF), BERT + LSTM, and BERT + BiLSTM. Then, the best model was adopted to predict the text classification results.

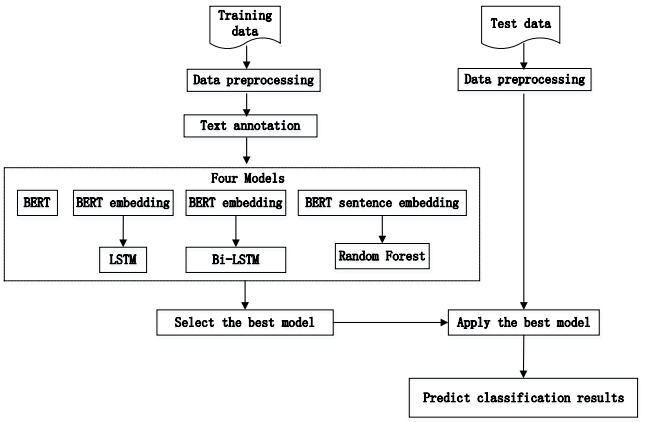


FIGURE 4. The flow chart of text classification.

3) KEYWORD MATCHING

The third step of entity recognition is to conduct keyword matching to extract the matched knowledge entity directly. First, a keyword library was built based on the target knowledge and online discussion transcripts. Second, the directly matched knowledge entity was extracted automatically through string matching. Finally, the accuracy of automatic keyword extraction was evaluated through the precision, recall rate, and accuracy rate.

C. RELATION EXTRACTION

After recognizing knowledge entities, the next step is to extract relations. As shown in Figure 5, there are three types of relations that need to be extracted, including the relations between knowledge entities, the activated relations, and unactivated relations. The relations between knowledge entities are extracted through query from the target knowledge graph. The activated relations were established among knowledge entities, each student, each group, and the class. The unactivated relations were established through comparing activated relationships with the target knowledge graph. The following section will illustrate the relation extraction through an example.

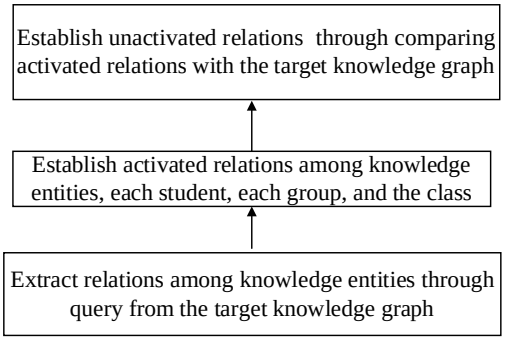


FIGURE 5. The flow chart of relation extraction.

1) EXTRACTION OF RELATIONS AMONG KNOWLEDGE ENTITIES

Figure 6 shows an example of relation extraction. The relations between knowledge entities can be regarded as the relation A between knowledge entity 1 and knowledge entity 2, as shown in Figure 6. The relation extraction between the two entities was conducted through a query from the target knowledge graph. When two knowledge entities have been recognized, the relation between them could be mapped automatically from the target knowledge graph.

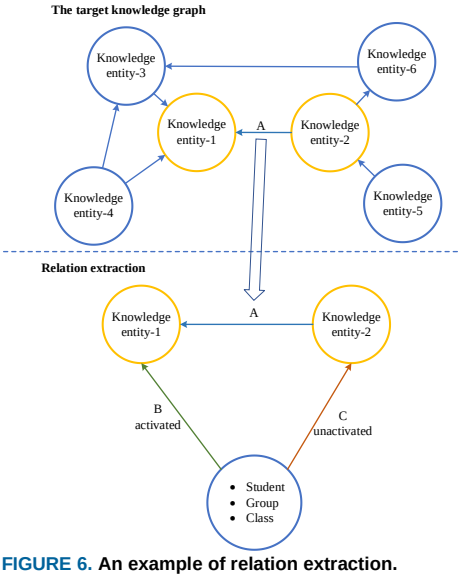


FIGURE 6. An example of relation extraction.

2) EXTRACTION OF ACTIVATED AND UNACTIVATED RELATIONSHIPS

As shown in Figure 6, the activated and unactivated relationships can be regarded as the relation B and the relation C, respectively. We established activated relationships among each knowledge entity and each student, group, and class. This can be represented by a triple. The triples include (student name, activated, knowledge entity), (group number, activated, knowledge entity), and (class, activated, knowledge entity). Through comparing the active relationships with the relationships of the target knowledge graph, all unactivated relationships can be identified clearly. This can also be represented by a triple. The triples are (student name, unactivated, knowledge entity), (group number, unactivated, knowledge entity) and (class, unactivated, knowledge entity). Thus, activated and unactivated relationships can be extracted and established.

D. CONSTRUCT AND STORE A KNOWLEDGE GRAPH

The purpose of this step is to construct knowledge graphs through integrating several optimal models. We integrated the entity recognition model, the text classification model, and the keyword matching model to recognize and predict knowledge entities in online discussion texts during online collaborative programming. After completing entity recognition and relationship extraction, the activated and unactivated knowledge graphs of each student, each group, and the class can be stored in the Neo4j graph database. The knowledge graph of the class consists of the knowledge graphs of all groups.

IV. EXPERIMENT AND RESULTS

A. DATA SET

This study adopted the VSCode software as an online collaborative programming platform for data collection. Learners used the LiveShare chat room in VSCode for online discussion. The VSCode software includes two areas: one is the programming area and the other is the discussion area, as shown in Figure 7. Online discussion transcripts were automatically recorded through the VSCode software.

This experiment initially recruited 103 college students who attended the C language course. After the pretest, 13 students with a score of less than 60 were excluded. Finally, there were 90 college students with 59 men and 31 women who participated in this experiment. The 90 participants were divided into 30 groups of three students each. The collaborative programming task was to develop a program about the three-person version of the snake game through C language. The game applied four rules. The first rule is that different snakes are represented by different colors. The second rule is that the length adds one if a snake eats one piece of food. The third rule is that the game will fail when the player encounters other game players, walls, or their own body. The fourth rule is that the player with the highest score wins within 3 minutes of the countdown. All of groups completed the same online collaborative programming task.

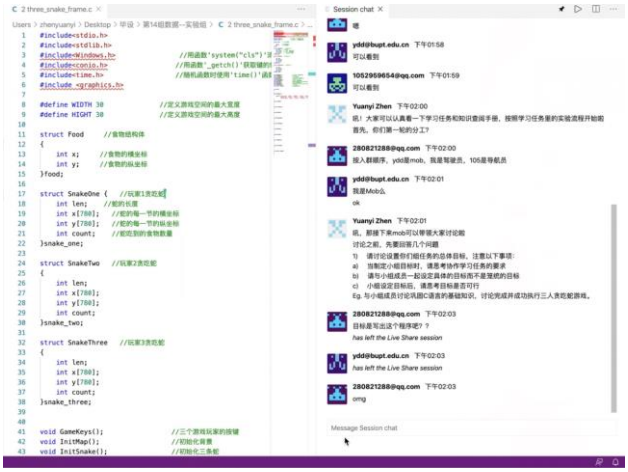


FIGURE 7. Screen shot of the online collaborative programming platform

B. EXPERIMENTAL SETUP

1) THE TARGET KNOWLEDGE GRAPH

In this experiment, there are 92 knowledge entities in the target knowledge graph, including 38 conceptual entities, 14 principle entities, 5 process entities, 24 example entities, and 11 format entities. In addition, there were 45 relations about "a kind of", 14 relations about "has characteristics of", and 35 relations about "including" in the target knowledge graph. In total, there were 94 triples in the target knowledge graph.

Furthermore, this study set up the class graph. The node types in the class knowledge graph included the class, group, and student. There are 121 nodes, including 1 class node, 30 group nodes, and 90 student nodes. The relationship types in the class knowledge graph are, namely, belong to (group node, belong to, class) and (student node, belong to, group). In total, there are 120 relationships among 121 nodes. Both the target knowledge graph and class graph were stored in the Neo4j graph database for retrieval later.

2) ENTITY RECOGNITION

a) Sequence tagging

In this experiment, there were 14193 online discussion transcripts generated by 30 groups, which comprised the overall data sets. According to the labeling rules, two research assistants who know C language well manually labeled all of the discussion transcripts and resolved any discrepancies. The order of the original online discussion transcripts was kept to obtain contextual semantics. We selected a single discussion transcript as a word and multiple discussion transcripts as sentences to form 12 classification labels, among which there were 11 semantically matched knowledge entities as well as other directly matched knowledge entities and irrelevant knowledge texts. Table 1 shows all of the labels and the data distribution.

TABLE 1
THE 12 CLASSIFICATION LABELS AND THE DATA DISTRIBUTION

Labels	Explanations	Data distributions
1	GameKeys function	405
2	InitMap function	442

3	InitSnake function	527
4	SetFood function	329
5	MoveSnake function	898
6	EatFood function	214
7	SetTime function	246
8	JudgeEnd function	243
9	Win function	187
10	StartGame function	69
11	Main function	282
12	Directly matched knowledge entity and irrelevant knowledge texts	10351

b) Text classification

After recognizing 11 semantically matched knowledge entities, the next step is to classify directly matched knowledge entities and irrelevant knowledge texts. In total, there were 10351 discussion transcripts for these two types of texts. Two research assistants classified 10351 discussion transcripts, among which 0 represented irrelevant knowledge texts, and 1 representing directly matched knowledge entities. The data set is shown in Table 2. Before the model training, the overall data set was divided into the training set, the validation set, and the test set according to the ratio of 7:2:1. That is, the overall data set contains 8493 irrelevant knowledge texts and 1858 directly matched knowledge entities.

TABLE 2
THE DATA DISTRIBUTION OF THE TEXT CLASSIFICATION

	Train	Validation	Test	Total
0	5945	1698	850	8493
1	1300	371	187	1858
Total	7245	2069	1037	10351

c) Keyword matching

In this study, the keyword library includes 92 entities and 58 of them include 148 synonyms. Table 3 shows some examples of the entity and its synonyms. Each comma in the entity synonyms separates a synonymous entity. The thesaurus size is 240 words in total.

TABLE 3
SOME EXAMPLES OF THE KEYWORD LIBRARY

Entity	Entity synonyms
Debugger	bug, debug, program error, debug code, debugging
TurnSnake()	turn, moving direction
Function call	call, parameter passing, transfer, transfer value
And	&
Loop statement	loop
Selection statement	select
For statement	for
Do while statement	do while
While statement	while
Break statement	break
Continue statement	continue
If else statement	if else
If statement	if
Else statement	else
Switch case statement	switch case
Switch statement	switch
Case statement	case
Integer	int, symbol
Character	char, character
String constant	str, string

C. RESULTS

1) ENTITY RECOGNITION

a) Sequence tagging

Table 4 shows the accuracy of the different lengths. It was found that the accuracy of our model achieved the highest (85.8%) when the sentence length was set to 500. Therefore, the sentence length was fixed at 500 to achieve the best performance.

TABLE 4
THE ACCURACY OF DIFFERENT LENGTHS

Sentence length	The accuracy
20	0.745
50	0.764
100	0.772
120	0.798
150	0.767
200	0.788
250	0.778
300	0.786
350	0.805
400	0.808
450	0.833
500	0.858

b) Text classification

Table 5 shows the results of the text classification. It was found that the BERT+RF achieved the best performance. The overall accuracy achieved 0.91.

TABLE 5
THE RESULTS OF TEXT CLASSIFICATION

Models	Indicators	Irrelevant knowledge text	Directly matched knowledge entity
BERT	Precision	0.95	0.32
	Recall	0.61	0.84
	Accuracy	0.65	
BERT+RF	Precision	0.91	0.89
	Recall	0.98	0.58
	Accuracy	0.91	
BERT+LSTM	Precision	0.82	0.21
	Recall	0.94	0.06
	Accuracy	0.79	
BERT+BiLSTM	Precision	0.82	0.17
	Recall	0.93	0.06
	Accuracy	0.77	

c) Keyword matching

The evaluate indicators of keyword matching included precision, recall and F1 value. It was found that the keyword recognition accuracy was 0.87, the recall rate was 0.98, the F1 value was 0.92, and the overall keyword matching model

accuracy rate was 85%. In summary, the overall accuracy of the proposed model achieved 87.27%.

2） RELATION EXTRACTION

Relation extraction aims to extract three types of relations. One is to recognize the relations among different knowledge entities. Regarding this type of relations, the main task is to retrieve the specific type of relations from the target knowledge graph. As a result, all the relations between recognized knowledge entities are extracted through correct ones. The second and third type of relations include the activated and unactivated relations among knowledge entities, students, groups, and the class. The performance of this type of relation extraction is closely related to the accuracy of entity recognition. If an entity is correctly recognized, the relation will also be correctly extracted, and vice versa. Therefore, the accuracy of relations extraction is actually identical to the overall accuracy of entity recognition, which is 89.7% for students, 90.4% for group, and 90.2% for class, respectively.

3）KNOWLEDGE GRAPH DEMONSTRATION

After construction of the knowledge graphs, the activated and unactivated knowledge graphs of each student, each group, and the class can be observed. The following sections will be illustrated one by one.

a) The knowledge graph of each student

- The activated knowledge graph of each student

In this study, we adopted the cypher query to retrieve the knowledge entity and relationships from the Neo4j database. The cypher query sentence is:

MATCH (:student { name: '***' })-[:`activated`]->(knowledge)

RETURN (:student { name: '***' })-[:`activated`]->(knowledge)

Take one student from group 1 as an example. Figure 8 shows the knowledge graph that this student has activated. As shown in Figure 6, there were 34 knowledge entities and 21 relationships. More specifically, there were 6 example entities, 22 concept entities, 1 format entity, 4 principle entities, and 1 process entity. In terms of activated relationships, there were 13 relationships about "is a kind of", 3 relationships about "has characteristics of", and 5 relationships about “including”.

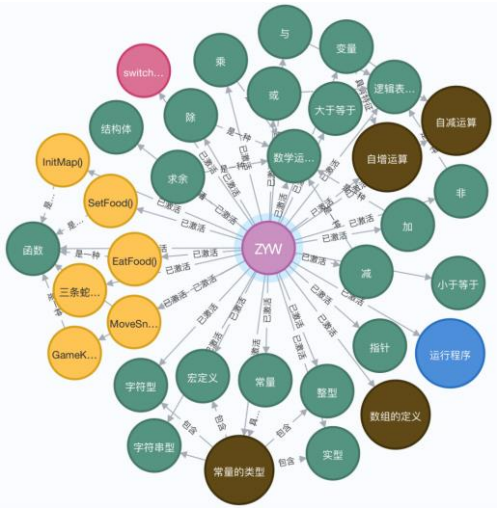


FIGURE 8. The activated knowledge graph of a student.

- The unactivated knowledge graph of each student

In this study, we adopted the cypher query to retrieve the knowledge entity and relationships from the Neo4j database. The cypher query sentence is:

MATCH (:student { name: '***' })-[:`unactivated`]->(knowledge)

RETURN (:student { name: '***' })-[:`unactivated`]->(knowledge)

Take the same student as an example. Figure 9 shows the knowledge graph that this student has not activated. As shown in Figure 7, there were 58 knowledge entities and 34 relationships. More specifically, there are 18 example entities, 16 concept entities, 10 format entities, 10 principle entities, and 4 process entities. In terms of activated relationships, there are 17 relationships about "is a kind of", 3 relationships about "has characteristics of", and 14 relationships about “including”.

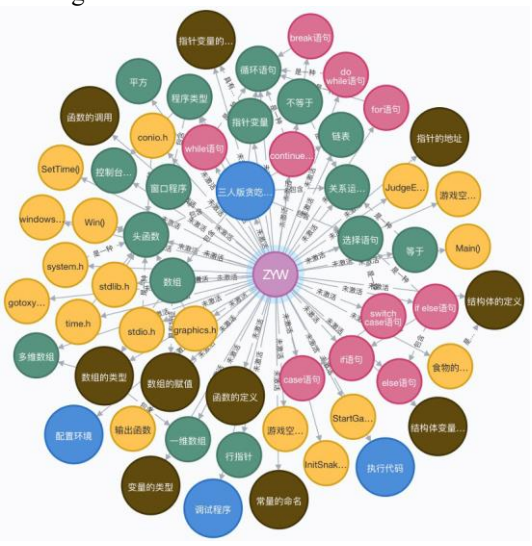


FIGURE 9. The unactivated knowledge graph of a student.

b) The knowledge graph of each group

- The activated knowledge graph of a group

Similarly, the activated knowledge graph of any group can be automatically generated. Figure 10 shows the activated knowledge graph of group 1. As shown in Figure 6, there were 61 knowledge entities and 38 relationships. More specifically, there were 9 example entities, 32 concept entities, 9 format entities, 9 principle entities, and 2 process entities. In terms of activated relationships, there are 22 relationships about "is a kind of", 9 relationships about "has characteristics of", and 7 relationships about “including”.

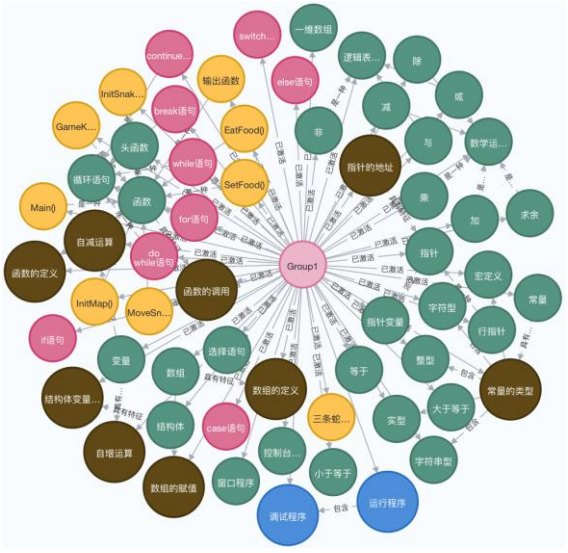


FIGURE 10. The activated knowledge graph of a group.

- The unactivated knowledge graph of a group

The unactivated knowledge graph of any group can be automatically generated. Figure 11 shows the unactivated knowledge graph of group 1. As shown in Figure 9, there were 31 knowledge entities and 6 relationships. More specifically, there were 15 example entities, 6 concept entities, 2 format entities, 5 principle entities, and 3 process entities. In terms of activated relationships, there were 2 relationships about "is a kind of" and 4 relationships about “including”.

FIGURE 11. The unactivated knowledge graph of a group.

c) The knowledge graph of the class

- The activated knowledge graph of the class

Similarly, the activated knowledge graph of the class (all groups) can be automatically generated. Figure 12 shows the activated knowledge graph of the class. As shown in Figure 9, there were 84 knowledge entities and 84 relationships. More specifically, there were 21 example entities, 36 concept entities, 11 format entities, 13 principle entities, and 3 process entities. In terms of activated relationships, there were 41 relationships about “is a kind of”, 13 relationships about “has characteristics of”, and 30 relationships about “including”.

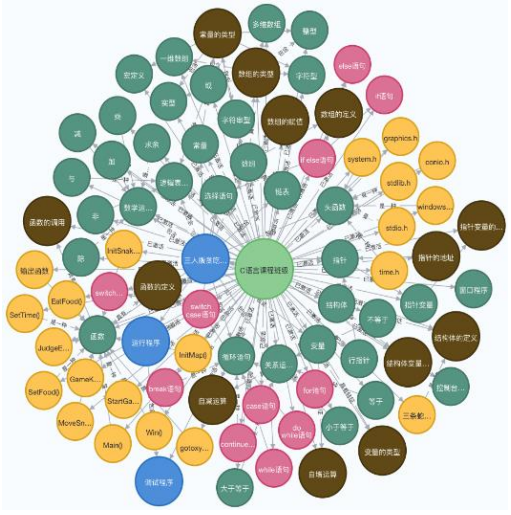


FIGURE 12. The activated knowledge graph of the class.

d) The knowledge graph of the class

- The unactivated knowledge graph of the class

Similarly, the unactivated knowledge graph of the class can be automatically generated. Figure 13 shows the unactivated knowledge graph. As shown in Figure 11, there were 8 unactivated knowledge entities. More specifically, there were 3 example entities, 2 concept entities, 1 principle entity, and 2 process entities.

FIGURE 13. **The unactivated knowledge graph of the class.**

4) EXAMPLE OF ILLUSTRATION

Take the student ZYW of the first group as an example. The process of knowledge graph construction is shown in Figure 14.

The discussion transcripts output by ZYW during the online collaborative programming were the input of the entire model. For example, ZYW said three sentences: "Hello everyone, I'm ZYW, I hope to complete the online collaborative programming tasks with you". "Logical expressions refers to the operators as ||, and &&, and not". "The forward direction of each section of the snake body is along the direction of the previous section that is connected to it". The three sentences were input to conduct sequence tagging. The third sentence was recognized as the entity 'MoveSnake()'. The first sentence and the second sentence were considered as the input to conduct text classification. The results indicated that the first sentence was classified into irrelevant knowledge and the second sentence is classified into directly matched knowledge entity. The second sentence was conducted keyword matching and four entities were recognized, including 'logical expressions', 'and', 'or', 'not'.

The next step is to extract relations from the target knowledge graph. There are three triples, including (and, is a kind of, logical expressions), (or, is a kind of, logical expressions), (not, is a kind of, logical expressions). In addition, activated relations were built between ZYW and the recognized entities, while unactivated relations were built between ZYW and other entities. Thus, the activated knowledge graph and inactive knowledge graph of ZYW are generated. The processes of generating activated and unactivated knowledge graphs of groups and the class are the same as that of the student ZYW.

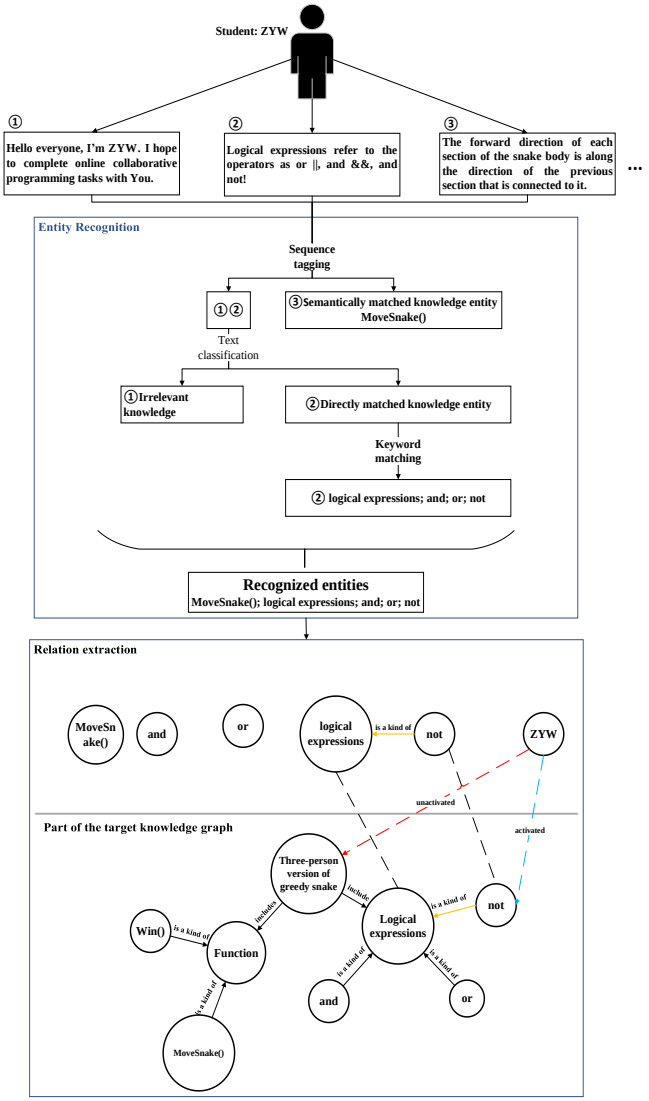


FIGURE 14. **An example of knowledge graph construction**

V. DISCUSSION

This study developed and examined a system that automatically constructed knowledge graphs for online collaborative programming. To the best of our knowledge, this is the first work that constructed knowledge graphs for online collaborative programming. This study has several implications for teachers, researchers, and developers.

First, the automatic generated knowledge graph is very helpful for each student in terms of obtaining a better understanding of the collaborative knowledge building progress. When students find an unactivated knowledge graph during online collaborative learning, they can interact and discuss the unactivated knowledge with group members.

Second, the automatically generated knowledge graph is very useful for each group. For example, all group members can quickly understand the latest progress in knowledge building through the activated and unactivated knowledge graphs. The constructed knowledge graphs also serve as a knowledge awareness tool for each group.

Third, the automatically generated knowledge graph can provide a picture of the whole class to teachers. Teachers can provide real-time intervention and personalized support for each student and each group based on the activated and unactivated knowledge graphs.

However, this study had several limitations. First, this study only focused on constructing knowledge graphs for a C language course. In a future study, we will explore how to automatically construct knowledge graphs for other learning domains. Second, the extraction of relationships was performed through queries of the target knowledge graph, which cannot detect relationships different from the target knowledge graph. In a future study, we will investigate other deep neural network models to extract relationships from online discussion texts.

VI. CONCLUSION

This study proposed an innovative approach and developed a system to automatically construct knowledge graphs for online collaborative programming. We adopted sequence tagging, text classification, and keyword matching to recognize knowledge entities. The extraction of relationships among knowledge entities was performed through queries of the target knowledge graph. In total, six types of knowledge graphs could be automatically generated, including the activated and unactivated knowledge graphs of each student, each group, and the class. The accuracy of entity recognition reached 87.27%. The accuracies of relation extraction for students, groups, and the class achieved 89.7%, 90.4%, and 90.2%, respectively. This study is very significant and promising for detecting the knowledge-building progress and for providing personalized services for learners.

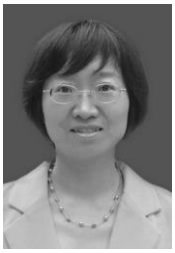
REFERENCES

[1] J. Harris, "First steps in telecollaboration," *Learn. Lead. Technol*., vol. 27, no. 3, pp. 54−57, 1999.
[2] P. Redmond and J. V. Lock, "A flexible framework for online collaborative learning," *Inter. Hig. Educ*., vol. 9, no. 4, pp. 267−276, 2006.
[3] T. C. Reeves, J. Herrington, and R. Oliver, "A development research agenda for online collaborative learning," *Educ. Technol. Res. Develop.*, vol. 52, no. 4, pp. 53−65, 2004.
[4] O. Noroozi, I. Alikhani, S. Järvelä, P. A. Kirschner, I. Juuso, and T. Seppänen, "Multimodal data to design visual learning analytics for understanding regulation of learning," *Comput Hum Behav.*, vol. 100, pp. 298–304, 2019.
[5] L. Zheng, L. Zhong, and J. Niu, "Effects of personalised feedback approach on knowledge building, emotions, co-regulated behavioural patterns and cognitive load in online collaborative learning," *Assess. Eval. Hig. Educ*., pp. 1−17, 2021.
[6] M. Sobocinski, J. Malmberg, and S. Järvelä, "Exploring temporal sequences of regulatory phases and associated interactions in low- and high-challenge collaborative learning sessions," *Metac. Learn.*, vol. 12, no. 2, pp. 275–294, 2017.
[7] N. Zhang, Q. Liu, X. Zheng, L. Luo, and Y. Cheng, "Analysis of Social Interaction and Behavior Patterns in the Process of Online to Offline Lesson Study: A Case Study of Chemistry Teaching Design based on Augmented Reality," *A. P. J. Educ*., pp. 1−22, 2021.
[8] D. W. Shaffer, W. Collier, and A. R. Ruis, "A tutorial on epistemic network analysis: Analyzing the structure of connections in cognitive, social, and interaction data," *J. Learn. Anal.*, vol. 3, no. 3, pp. 9–45, 2016.
[9] B. Wu, Y. Hu, A. R. Ruis, and M. Wang, "Analysing computational thinking in collaborative programming: A quantitative ethnography approach," *J. Comput. Ass. Learn*., vol. 35, no. 3, pp. 421–434, 2019.
[10] S. Mladenović, D. Krpan, and M. Mladenović, "Using games to help novices embrace programming: From elementary to higher education," *Int. J. Eng. Educ*., vol. 32, no.1, pp. 521–531, 2016.
[11] H. Paulheim, "Knowledge graph refinement: A survey of approaches and evaluation methods," *Semantic web.*, vol. 8, no. 3, pp. 489−508, 2017.
[12] C.E. Hmelo-Silver, "Analyzing collaborative knowledge construction: Multiple methods for integrated understanding," *Comput. Educ*., vol. 41, no. 4, pp. 397−420, 2003.
[13] U. L. R. I. K. E. Cress and F.W. Hesse, "Quantitative methods for studying small groups," in *Int. han. Collabo. learn.*, 2013, pp. 93−111.
[14] Q. Liu, S. Zhang, Q. Wang, and W. Chen, "Mining Online Discussion Data for Understanding Teachers' Reflective Thinking," *IEEE Trans. Learn. Technol*., vol.11, no. 2, pp. 243−254, 2018.
[15] K. Xie, G. Di Tosto, L. Lu, and Y. S. Cho, "Detecting leadership in peer-moderated online collaborative learning through text mining and social network analysis," *Int. Hig Educ.*, vol. 38, pp. 9−17, 2018.
[16] J. Y. Wu, Y. C. Hsiao, and M. W. Nian, "Using supervised machine learning on large-scale online forums to classify course-related Facebook messages in predicting learning achievement within the personal learning environment," *Interact. Learn. Environ*., vol. 28, no. 1, pp. 65−80, 2020.
[17] W. Hadi, Q. A. Al-Radaideh, and S. Alhawari, "Integrating associative rule-based classification with Naïve Bayes for text classification," *Appl. Soft. Comput.*, vol. 69, pp. 344–356, 2018.
[18] G. Shan, S. Xu, L. Yang, S. Jia, and Y. Xiang, "Learn#: A Novel incremental learning method for text classification," *Exp. Syst. Appl.*, vol. 147, pp. 1−11, 2020.
[19] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, "Deep learning," *Cambridge: MIT press.*, vol. 1, no. 2, 2016.
[20] J. T. Nosek, "The case for collaborative programming," *Commun. ACM*., vol. 41, no. 3, pp. 105–108, 1998.

[21] O. H. Lu, J. C. Huang, A. Y. Huang, and S. J. Yang, "Applying learning analytics for improving students engagement and learning outcomes in an MOOCs enabled collaborative programming course," *Interact. Learn. Environ.*, vol. 25, no. 2, pp. 220–234, 2017.
[22] J. Denner, L. Werner, S. Campe, and E. Ortiz, "Pair programming: Under what conditions is it advantageous for middle school students?" *J. Res. Technol. Educ.*, vol. 46, no. 3, pp. 277–296, 2014.
[23] X. M. Wang, and G. J. Hwang, "A problem posing-based practicing strategy for facilitating students' computer programming skills in the team-based learning mode," *Educ. Technol. Res. Develop.*, vol. 65, no. 6, pp. 1655–1671, 2017.
[24] L. Beck and A. Chizhik, "Cooperative learning instructional methods for CS1: Design, implementation, and evaluation," *ACM Trans. Comput. Educ.*, vol.13, no. 3, pp. 10−21, 2013.
[25] Y. Lu, X. Mao, T. Wang, G. Yin, and Z. Li, "Improving students' programming quality with the continuous inspection process: a social coding perspective," *Frontiers of Computer Science*, vol.14, no.5, pp.1–18, 2020.
[26] E. W. Schneider, "Course Modularization Applied: The Interface System and Its Implications For Sequence Control and Data Analysis," presented at *In ADIS*, Chicago, Illinois, April 1972.
[27] A. Singhal, Introducing the Knowledge Graph: things, not strings. Google Blog. [Online]. Available: https://www.blog.google/products/search/introducing-knowledge-graph-things-not/.2012.
[28] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, and A. Zimmermann. (2020). "Knowledge graphs." [Online]. Available: https://arxiv.org/pdf/2003.02320.pdf
[29] S. Ji, S. Pan, E. Cambria, P. Marttinen, and S. Y. Philip, "A Survey on Knowledge Graphs: Representation, Acquisition, and Applications," *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1−21, 2021. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9416312.
[30] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingv. Invest.*, vol. 30, no. 1, pp. 3–26, 2007.
[31] L. Xiao and S. W. Daniel, "Fine-Grained Entity Recognition," in Proc. *AAAI*, Toronto, Ontario, Canada, 2012, pp. 94−100.
[32] L. Guillaume, B. Miguel, S. Sandeep, K. Kazuya, and D. Chris, "Neural Architectures for Named Entity Recognition," in *Proc. NAACL-HLT*, San Diego California, USA, 2016, pp.260–270.
[33] G. Sonal and D. M. Christopher, "Improved Pattern Learning for Bootstrapped Entity Extraction," in *Proc. CoNLL*, Baltimore, Maryland, USA, 2014, pp. 98–108.
[34] Y. Dani, G. Daniel, and L. Nevena, "Embedding Methods for Fine Grained Entity Type Classification," in *Proc. ACL*, Beijing, China, 2015, pp. 291–296.
[35] J. Lafferty, A.McCallum, F.C.N.Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. ICML*, Williams College, Williamstown, MA, USA, 2001, pp. 282–289.
[36] P. Chen, Y. Lu, V. M. Zheng, X. Chen, and B. Yang, "KnowEdu: a system to construct knowledge graph for education," *IEEE Access*, vol. 6, pp. 31553−31563, 2018.
[37] R. Leaman, C. H. Wei, and Z. Lu, "tmChem:A high performance approach for chemical named entity recognition and normalization," *J. Chem.*, vol. 7, no. 1, pp. 1−10, 2015.
[38] A. L.-F. Han, D. F.Wong, and L. S. Chao, "Chinese named entity recognition with conditional random fields in the light of chinese characteristics," in *Lang. Process. Intell. Inf. Syst*. Springer, 2013, pp. 57−68.
[39] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. (2014). "Empirical evaluation of gated recurrent neural networks on sequence modeling." [Online]. Available: https://arxiv.org/abs/1412.3555
[40] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780,1997.
[41] Z. Huang, W. Xu, and K. Yu. (2015). "Bidirectional LSTM-CRF models for sequence tagging." [Online]. Available: https://arxiv.org/abs/1508.01991
[42] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer (2016). "Neural architectures for named entity recognition." [Online]. Available: https://arxiv.org/abs/1603.01360
[43] R. Stephen, K. Douwe and N. Maximilian, "Hearst Patterns Revisited: Automatic Hypernym Detection from Large Text Corpora," in *Proc. ACL*, Melbourne, Australia, 2018, pp. 15−20.
[44] L. Guillaume, B. Miguel, S. Sandeep, K. Kazuya, and D. Chris, "Neural Architectures for Named Entity Recognition," in *Proc. NAACL-HLT*, San Diego California, USA, 2016, pp.260–270.
[45] N. Ndapandula, T. Tomasz, and W. Gerhard, "Fine-grained Semantic Typing of Emerging Entities," in *Proc. ACL*, Sofia, Bulgaria, 2013, pp. 4−9.
[46] X. Wei, H. Raphael, Z. Le, and G. Ralph, "Filling Knowledge Base Gaps for Distant Supervision of Relation Extraction," in *Proc. ACL*, Sofia, Bulgaria, 2013, pp. 665–670.
[47] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 926−934.
[48] H. Paulheim, "Knowledge graph refinement: A survey of approaches and evaluation methods," *Semantic Web.*, vol. 8, no. 3, pp. 489−508, 2017.
[49] A. Rossi, D. Barbosa, D. Firmani, A. Matinata, and P. Merialdo, "Knowledge Graph Embedding for Link Prediction: A Comparative Analysis", ACM T. Knowl. Discov. D., Vol. 15, no. 14, pp. 1−49, 2021.
[50] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, "A survey on knowledge graphs: representation, acquisition, and applications", arXiv., preprint arXiv. 2002.00388.

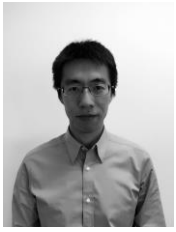
[51] X. Huang, J. Y. Zhang, D. C. Li, and P. Li, "Knowledge graph embedding based question answering," in *Proc. WSDM*, Melbourne, Australia, 2019, pp. 105–113.
[52] X. Chen, P. Russell, and C. Jamie, "Explicit semantic ranking for academic search via knowledge graph embedding," in *Proc. WWW*, Perth, Australia, 2017, pp. 1271–1279.
[53] W. Hong, Z. Miao, X. Xing, L. Wen, and G. Min, "Knowledge graph convolutional networks for recommender systems," in *Proc. WWW*, San Francisco, America, 2019, pp. 3307–3313.
[54] Y. Bi and M. Tom, "Leveraging knowledge bases in LSTMs for improving machine reading," in *Proc. ACL*, Vancouver, Canada, 2017, pp.1436–1446.
[55] B. Abu-Salih, "Domain-specific knowledge graphs: A survey", J. Netw. Comput. Appl., Vol. 185, pp. 103076, 2021.
[56] H. Liu, W. Ma, Y. Yang, and J. Carbonell, "Learning concept graphs from online educational data," *J. Artif. Intell. Res.*, vol. 55, pp. 1059−1090, 2016.
[57] D. Chaplot and K. R. Koedinger, "Data-driven automated induction of prerequisite structure graphs," in *Proc. Edu. Data Mining (EDM)*, 2016, pp. 318−323.
[58] B. Abu-Salih, M. Al-Tawil, I. Aljarah, H. Faris, P. Wongthongtham, K. Y. Chen, and A. Beheshti, "Relational learning analysis of social politics using knowledge graph embedding", Data Min. Knowl. Disc., Vol. 35, pp. 1497-1536, 2021.
[59] K. C. Yang, "*The guidelines for Instructional design based on learning activities*," China: Electronic Industry, 2016, pp.55−57.
[60] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova. (2018). "BERT: Pre-training of deep bidirectional transformers for language understanding." [Online]. Available: https://arxiv.org/abs/1810.04805
[61] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, and I. Polosukhin. (2017). "Attention is all you need." [Online]. Available: https://arxiv.org/abs/1706.03762
[62] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735−1780, 1997.
[63] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inf. Theory*, vol. TIT-13, no. 2, pp. 260−269, Apr. 1967.

Yuanyi Zhen was born in Changzhi, Shanxi, China in 1996, female, master, graduated from Ocean University of China in July 2018 with a B.S. degree and Beijing Normal University in July 2021 with a M.S. degree in Educational Technology, with research interests in computer supported collaborative learning, knowledge graph construction, and learning analytics.



Lanqin Zheng received the M.S. and Ph.D. degrees in educational technology from Beijing Normal University in 2004 and 2012, respectively.

She currently works as an associate professor at the School of Educational Technology, Faculty of Education in Beijing Normal University. She has published almost 100 refereed academic papers. She is an Associate Editor of Australasian Journal of Educational Technology (AJET). Her research interests include computer-supported collaborative learning, knowledge graph construction, and learning analytics.



Penghe Chen received the B.S. and Ph.D. degrees in Computer Science from the National University of Singapore (NUS).

He currently serves as the principle researcher at the Advanced Innovation Center for Future Education, Beijing Normal University (BNU), China. Before that, he had been working at the Advanced Digital Sciences Center (ADSC), Singapore. His research interests include knowledge graph construction, data mining and learning analytics.