ViQG: Web Tool for Automatic Question Generation from Code for Viva Preparation

Hunny Gaur
Computer Science and Engineering
IGDTUW
Delhi, India
gaurhunny20@gmail.com

Devendra K Tayal
Computer Science and Engineering
IGDTUW
Delhi, India
devendrakumartayal@gmail.com

Amita Jain
Computer Science and Engineering
NSUT
Delhi, India
amita_jain_17@yahoo.com

2023 26th Conference of the Oriental COCOSDA International Committee for the Co-ordination and Standardisation of Speech Databases and Assessment Techniques (O-COCOSDA) | 979-8-3503-4402-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/O-COCOSDA60357.2023.10482988

***Abstract*—Natural language generation, a sub-field of natural language processing has extensive applications in the domain of education. One of the applications, widely explored by researchers is generating questions automatically from the text. Ample amount of research work is available with respect to automatic generation of questions for framing the question paper. However, framing the question paper is a teacher centric application. Automatic question generation can also be used for enhancing the student learning. Majority of the computer science papers include practical component and students need to appear for practical exam which encompass viva. In this paper, we have proposed a method to generate the questions automatically from the programming code for three different languages, C++, Java and Python. The proposed method is implemented as part of a web tool *ViQG (Viva Question Generator)*. The web tool can aid students to prepare for the viva by generating the probable questions related to a specific piece of code. This can reduce the time and effort required to explore the questions related to certain code, on different platforms.**

***Index Terms*—Natural Language Processing, Automatic Question Generation, Programming Questions, Web Tool**

I. Introduction

Assessment of student learning is an important task in education system. This assessment is not only limited to subjective examinations. It comprises of other components also, such as oral examinations, generally called viva. With the advancement in education policies, it is required to assess the students by performing continuous evaluation. This continuous evaluation includes conducting test periodically. However, for the papers that include practical component, the continuous evaluation is required during practical sessions also. Numerous amounts of research work are available with respect to providing solutions to ease the preparation required for evaluation. One of the research problems comprehensively addressed by researcher is automatic question generation from text.

Literature has shown promising results in automatic question generation from the text. Several types of question generation, such as, MCQ (Multiple Choice Question), True/False, Fill-in-the blank and factual (Wh-question) has been addressed in the literature. Authors in [23] have presented a detailed study about generating questions automatically using machine learning approaches. Out of the work available in literature, majority of the work focuses on generating MCQ from text. Authors in [19] have discussed in detail about generic flow of generating MCQs along with the techniques being used in each phase of the flow. A wide variety of approaches has been used for automatic generation of MCQs, such as, Natural Language Processing (NLP) techniques, namely, lemmatization, POS (Part of Speech) tagging [1][2][20], dependency tree [4][5]. Apart from NLP, authors have also used abstract rules to extract knowledge from text [3], defined pattern for selecting text from Wikipedia [6], performed semantic labeling on sentences to generate questions [7][17] [34], used preference learning based on fireflies along with the concept of hyponyms and hypernyms from lexical database [12].

Authors in [13] used the English Chinese parallel corpus for automatic MCQ generation that select sentences from corpus on the basis of certain criteria such as length of the sentence, choice of words and syntactic structure of the sentence. In [28] and [37], authors concentrated on generating MCQs using language models such as abstractive LSTM (Long Short Term Memory) and BERT (Bidirectional Encoder Representations from Transformers) XL, respectively. Crowdsourcing [14], genetic algorithms [8] and graph neural networks [39] have also been used by researchers for generating questions automatically from the text. Some researchers [18] [27] have also focused on generating questions for a specific domain only. Apart from MCQs, researchers have also worked on automatic generation of True/False questions [32], fill-in-the- blank questions [16] and programming exercise questions [29]. The research problem of short answer questions has also been addressed by many researchers by using lexical functional grammar approach [10], ontologies [11], structure analysis [30], k-nearest neighbor machine learning approach [35] and hierarchical skill-based schema technique [38]. Authors in [36] have presented the method for generating all types of questions from text and authors in [33] have presented method for generating questions from images and chart input data that can help in enhancing the knowledge of the user.

Instead of generating specific types of question, researchers

have also focused on generating the complete question paper [9]. To ensure the quality of the question paper, few researchers have also considered that the question paper should be generated as per Bloom's taxonomy [15] [22]. Few researchers have also worked in the way of refining the quality of questions being generated automatically. Authors in [21] proposed a question similarity mechanism to rule out the irrelevant questions from the pool of questions generated from the text. In [24], researchers proposed a multitask learning framework that can introduce commonsense knowledge into the process of question generation. Authors in [25] and [26] concentrated on enhancing the quality of question generation by using deep linguistic representations and fine tuning pre-trained language models, respectively. In [31], authors proposed a generic module to pre-process the text being used for automatic question generation.

Although, literature has widely addressed the research problem of question generation, however, it has been addressed as the teacher centric application. Automatic question generation can also be applied to enhance the student learning. With continuous evaluation, students are also required to prepare for the oral examination that takes place during practical sessions. In this paper, we have addressed the research problem of automatic question generation to strengthen learning of the students. We have proposed a method *QGCode (Question Generation form Code)* that accepts the programming code as the input. Process the input to extract the keywords and automatically generate questions related to the keywords present in the input code. The generated questions can be helpful for students to prepare for the oral examination during practical sessions. The proposed method has been implemented as the web tool *ViQG (Viva Question Generator)*. ViQG accepts programming code in three different languages, namely, C++, Java and Python. User is required to select any one language and provide the input code. As output, the proposed method produces the prospective list of questions related to code specified.

The rest of the paper is organized as follows: Section 2 discusses about the proposed method, followed by Section 3 that presents illustration of the proposed method with the help of an example. Section 4 presents the experimental setup and result, followed by the last section conclusion and future work.

II. Proposed Method

The literature comprises of numerous approaches for automatic generation of several types of questions from the text. These generated questions can be helpful for teachers to frame the question paper. However, the proposed approach is more student centric. The proposed method QGCode (Question Generation from Code), intend to help students to prepare for viva. Fig 1 presents the workflow diagram of the proposed method.

QGCode: Question Generation from Code

The proposed method accepts the programming code as input from user. The programming code input can be in any language, namely, C++, Java and Python. The input is processed to remove the punctuations from the code and a list of keywords is prepared. These input keywords are compared with the keywords present in the selected language data set. For all the keywords fetched from the input code and matched with the keywords presented in the selected language data set, the proposed method accesses all the question from data set, mapped with the specified keyword. As an output, the accessed questions are presented as a numbered list to the user.

Algorithm 1: QGCode: Question Generation from Code

Input: PL: Programming Language
CD: Code Written in PL
Output: List of Questions generated from Code

```
Define a list of punctuations PUNC
Initialized New String newString
for each word w in CD do
    if w not in PUNC then
        newString = newString + w
    end
end
Convert newString into list L_key (keywords of input code)
for each keyword K_F in dataset do
    for all keywords K_L in L_key do
        if K_F == K_L then
            Access all the questions mapped with K_F
        end
    end
end
Display all the Viva Questions.
```

III. Method Illustration with Example

This section describes the proposed method with the help of an example. Consider the following Python code as the input:

Input:

```
def addition():
    x = int(input('Enter the first
    integer number'))
    y = int(input('Enter the second
    interger number'))
    print('Addition of the entered
    numbers is:', x+y)
```

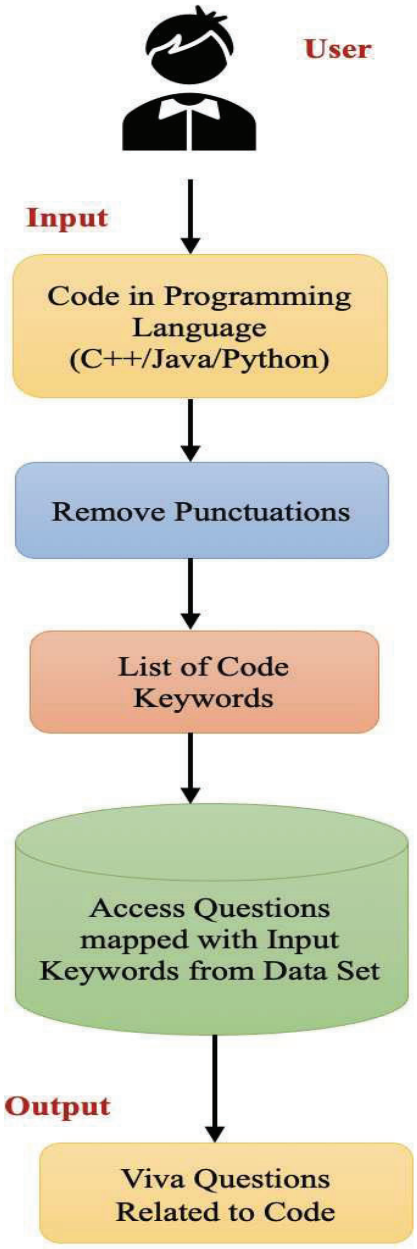

Fig. 1. Workflow Diagram of Proposed Method.

After processing the code, the punctuations will be removed and the list of input keywords will be:

[*def, addition, x, int, input, Enter, the, first,....., y*]

After matching the input keywords with the keywords present in the data set, the potential list of questions generated as output are:

Output: List of questions mapped with the keywords

1) What is the significance of input() function?
2) Why do we use print() function?
3) Can a python function return multiple values?
4) Does python function returns any value if return statement is not given?

IV. Experiment Set Up and Result

This section discusses about the data set, technologies used and the result of the experiment. The data set created for automatically generating the questions from code, for the viva preparation, has been discussed in subsection A. To make it easy to use by students, a web tool ViQG (Viva Question Generation) has been developed. The web tool accepts the programming language as well as the code written in selected language as input. The web tool processes the input and produce the potential questions related to code as the output. The technologies used for developing the web tool has been discussed in the subsection B. The detailed result of the experiment has been discussed in subsection C.

A. DataSet

To test the efficiency of the proposed method, a dataset for three programming languages, namely, C++, Java and Python was created. The data set comprises of keywords and viva questions mapped with these keywords, for each language. The dataset contains 75, 82 and 97 viva questions of C++, Java and Python programming language, respectively. Figure 2 presents the sample of viva questions for C++, Java and Python programming language.

B. Technologies Used

To develop the web tool ViQG, an open-source web framework Django was used along with Python as the backend language. The web pages were designed using web technologies HTML (Hyper Text Markup Language) and CSS (Cascading Style Sheet).

C. Result Analysis

This section presents result analysis of the questions generated automatically from the programming code using web tool *ViQG*. Due to space constraint, we have presented the result generated by Java programming language only. The first page of the web tool *ViQG*, called the home page, provides three buttons to the user. Each button corresponds to one programming language. Figure 3 presents the screenshot of the home page of the web tool.

When user selects a programming language by clicking on the button, here, its Java, user is directed to the next web page. This web page ask user to enter the code in the selected programming language. Figure 4 presents the screenshot of the input page of the programming language Java. Now user can enter the code in the given textbox and click on the *Get the Questions* button to get the generated questions. Figure 5 and 6 represents the screenshots of the pages with Java input code and viva questions generated from the code, respectively.

	Keyword	Question
0	while	What is the difference between while and for l...
1	while	How while loop is different from do-while loop?
2	while	How to create an infinite loop?
3	for	What is the difference between while and for l...
4	for	Is it possible to initialize multiple variable...
5	for	Why we use nested loops?

Fig 2(a). C++ Data Set Sample

	Keyword	Question
0	import	What is the purpose of using import statement ...
1	import	Is it possible to import a package/class twice?
2	import	What is a static import?
3	import	Which of the following syntax is correct:\n(a)...
4	class	Why is it mandatory to create an object to acc...
5	class	Can a class be declared as static?

Fig 2(b). Java Data Set Sample

	Keyword	Question
0	def	What are the two types of functions in Python?
1	def	Explain recursion and why it is useful?
2	def	Is it possible for a function to read a variab...
3	def	Is it possible for a function to write to a va...
4	def	Do python functions have return values?
5	def	Can a Python function return multiple values?

Fig 2(c). Python Data Set Sample

Fig. 2. Samples of C++, Java and Python DataSet.

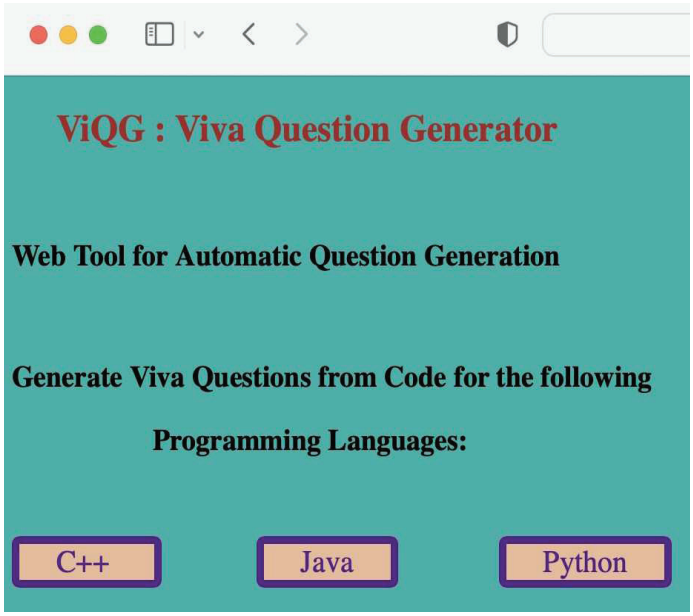

Fig. 3. ViQG Home Page.

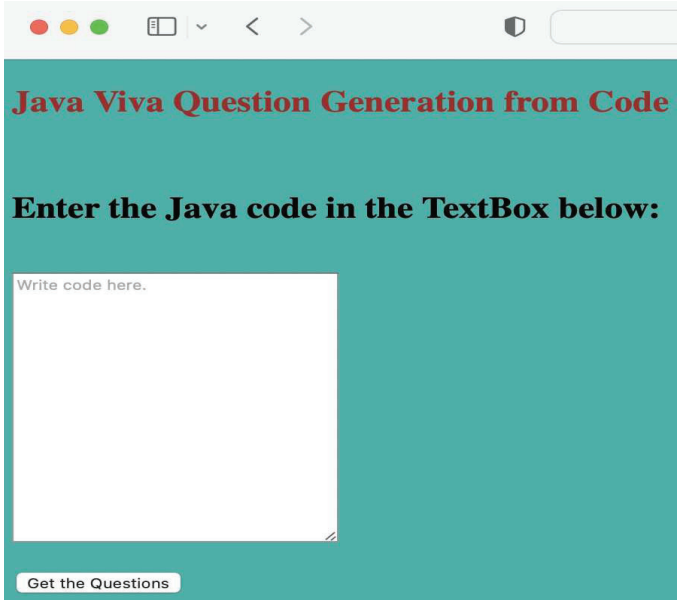

Fig. 4. Java Input Page Screenshot.

V. Conclusion and Future Work

In this paper, we have presented a web tool ViQG, that accepts the programming code as input and automatically generates the potential questions from entered code. The proposed method QGCode generates the list of potential viva questions on the basis of keyword identified in the input code. The list of potential questions can be helpful for students to prepare for viva. By entering the programming code as input to the web tool, students can get the idea about the questions related to the topics presented in the code.

The proposed method can be further enhanced by including other programming languages. Also, the method can be optimized to map the viva questions with punctuation symbols to include the questions on the topics such as C++ array, Python list, set, tuple and dictionary. These concepts are identified in the programming code using punctuation symbols, instead of any specific keyword.

References

[1] C. Chen, H. Liou, and J.S.Chang, "FAST-an automatic generation system for grammar tests," Proceedins of the COLING/ACL, pp. 1–4, July 2006.

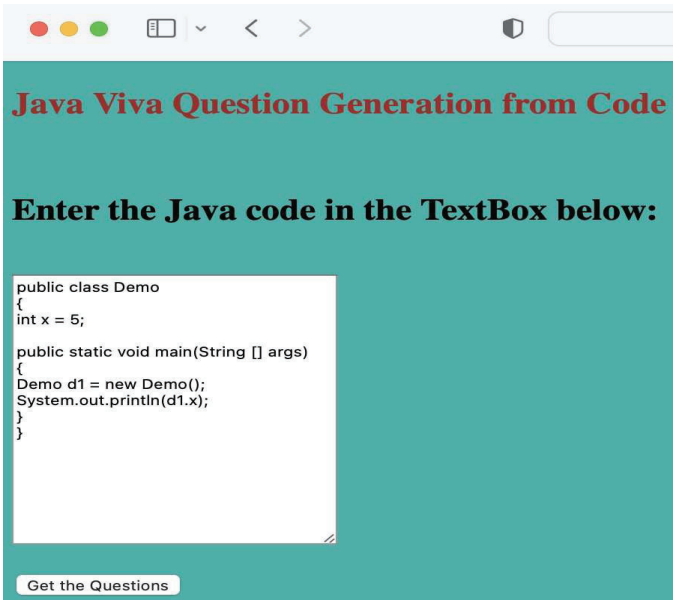

Fig. 5. Programming Code Entered by User.

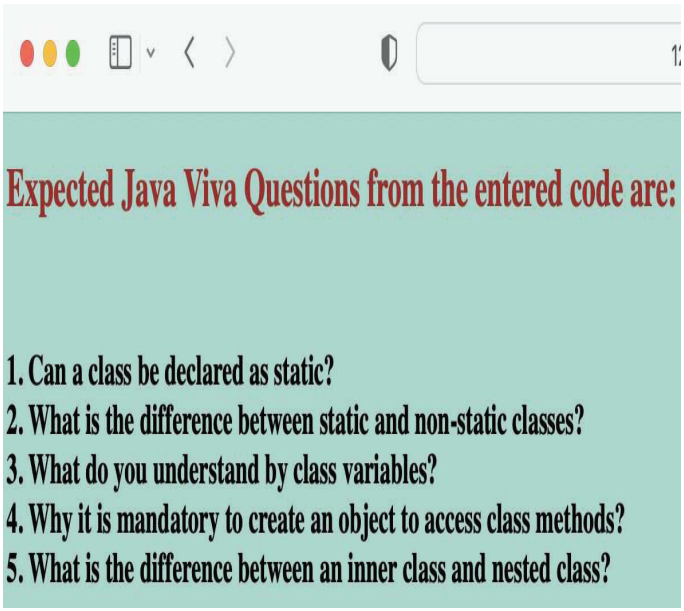

Fig. 6. List of Java Viva Questions.

[2] N. Afzal, and V.Pekar, "Unsupervised relation extraction for automatic generation of multiple choice questions," International Conference RANLP,Bulgaria, pp.1–5, 2009.

[3] H.Ogawa, H. K.bayashi, N.Matsuda, T.Hirashima, and H.Taki, "Knowledge externalization based on differences of solutions for automatic generation of multiple choice question," Proceedins of the 19th International Conference on computers in education, Thailand, 2011.

[4] N. Afzal, R. Mitkov, and A.Farzindar, "Unsupervised relation extraction using dependency trees for automatic generation of multiple choice questions," LNAI 6657, Springer Verlag Berlin Heidelberg,pp. 32–43, 2011.

[5] N. Afzal, and R. Mitkov, "Automatic generation of multiple choice questions using dependency based semantic relations," Soft Computing, 18,pp. 1269–1281, 2013.

[6] A. S.Bhatia, M.Kirti, and S. K.Saha, "Automatic generation of multiple choice questions using Wikipedia," In Pattern Recognition and Machine Learning: 5th International Conference PReMI,pp. 733–738, 2013.

[7] I. E.Fattoh, A.E.Aboutabl, and M. H.Haggag, "Semantic attributes model for automatic generation of multiple choice questions," International Journal of Computer Applications,pp. 18–24, 2014.

[8] K. Zhang, and L. Zhu, "Applications of improved genetic algorithm in automatic test paper generation," In Chinese Automation Congress,IEEE,pp. 495–499, 2015.

[9] K.H.Pinjani, R.Y.Raut, and S. E.Yedey, "Developing an intelligent agent for automatic question paper setting," International Journal of Electronics Communication and Soft Computing Science and Engineering,pp. 231, 2015.

[10] Y. Huang, and L. He, "Automatic geneartion of short answer questions for reading comprehension assessment," Natural Language Engineering,pp. 457–489, 2016.

[11] N.S.Stancheva, A. S.Doycheva, I.Popchev, and S.Stoyanov, "Automatic generation of test questions by software agents using ontologies," 8th International Conference on Intelligent Systems, IEEE, pp. 741–746, 2016.

[12] A.Santhanavijayan, S. R.Balasundaram, S.H.Narayanan, S.V.Kumar,and V.V.Prasad, "Automatic generation of multiple choice questions for e-assessment," International Journal of Signal and Imaging Systems Engineering, pp. 54–62, 2017.

[13] L.Wang, S. Li,H.Wang, and S.Yu, "Corpus based automatic generation of multiple choice questions for college english tests," Advances in Social Science, Education and Humanities Research, vol 146,pp. 156–162, 2017.

[14] J.Welbl, N. F.Liu, and M.Gardner, "Crowdsourcing multiple choice science questions," arXiv preprint arXiv: 1707.06209, 2017.

[15] S.Dhainje, R. Chatur,K.Borse, and V.Bhamare, "An automatic question paper generation: using Bloom's taxonomy," International Research Journal of Engineering and Technology, vol 05, 2018.

[16] S.Pannu, A.Krishna, S.Kumari, R.Patra,and S.K.Saha, "Automatic generation of fill in the blank questions from history books for school level evaluation," Progress in Computing, Analytics and Networking: Proceedings of ICCAN, pp. 461–469, 2018.

[17] P.Alvarez, and S.Baldassarri, "Semantics and service technologies for the automatic generation of online MCQ tests," Global Engineering Education Conference, IEEE, pp. 421–426, 2018.

[18] R.P.Azevedo, M.J.V.Pereira, and P.R. Henriques,"DSL based automatic generation of Q and A systems," New Knowledge in Information Systems and Technologies, Springer, pp. 460–471, 2019.

[19] D.Rao, and S.K.Saha, "Automatic multiple choice question generation from text: A survey," IEEE Transactions on Learning Technologies, vol 13, pp. 14–25, 2020.

[20] C.A.Nwafor, and I.E.Onyenwe, "An automated multiple choice question generation using Natural Language Processing techniques," arXiv preprint arXiv: 2103.14757, 2021.

[21] S.G.Aithal, A.B.Rao, and S.Singh, "Automatic question answer pairs generation and question similarity mechanism in question answering system," Applied Intelligence, pp. 1–14, 2021.

[22] S.Joshi, P.Shah, and S.Shah, "Automatic question paper generation according to Bloom's taxonomy by generating questions from text using Natural Language Processing," International Journal of Innovative Science and Research Technology, vol 6, 2021.

[23] K.B.Dhomse, and I.Ranjan, "Automatic question paper generation using ML: A review," Turkish Journal of Computer and Mathematics Education, vol 12, pp. 239–245, 2021.

[24] X.Jia, H.Wnag, D.Yin, and Y.Wu, "Enhancing question generation with commonsense knowledge," China National Conference on Chinese Computational Linguistics, Springer, pp. 145–160, 2021.

[25] W.Yuan, T.He, and X.Dai, "Improving neural question generation using deep linguistic representation," Proceedings of the Web Conference, pp. 3489–3500, 2021.

[26] M.Srivastava, and N.Goodman, "Question generation for adaptive education," arXiv preprint arXiv: 2106.04262, 2021.

[27] A.D.Lelkes, V.Q.Tran, and C.Yu, "Quiz style question generation for news stories," Proceedings of the Web Conference, pp. 2501–2511, 2021.

[28] M.Pranav, G.Deepak, and A.Santhanavijayan, "Automated multiple choice question creation using synonymization and factual confirmation, " Advances in Data Computing Communication and Security: Proceedings of 13CS2021, pp. 273–282, 2022.

[29] S.Sarsa, and P.Denny, "Automatic generation of programming exercises and code explanations using large language models, " Proceedings of

the ACM conference on International Computing Education Research, vol 1, pp. 27–43, 2022.
[30] M.Blstak, and V.Rozinajova, “Automatic question generation based on sentence structure analysis and machine learning approach, ” Natural Language Engineering, vol 28, pp. 487–517, 2022.
[31] N.Ch, and D.R.Ch, “Development of a generic pre-processing module for automatic question generation, ” International Journal for Advanced Research in Science and Technology, vol 12, pp. 108–115, 2022.
[32] R.Kasakowskij, T.Kasakowskij, and N.Seidel, “Generation of multiple true false questions, ” Fachtagung Bildungstechnologien (DELFI), 2022.
[33] D.K.Tayal, A. Jain, N.Shrivastava, A.Jain, and H.Gaur, “Knowledge enhancement using question generation for images and chart data input, ” 4th International Conference on Artificial Intelligence and Speech Technology, IEEE, pp. 1–6, 2022.
[34] A.P.Kumar, A. Nayak, M.Shenoy, Chaitanya, and K.Ghosh, “A novel framework for the generation of multiple choice questions stems using semantic and machine learning techniques, ” International Journal of Artificial Intelligence in Education, pp. 1–44, 2023.
[35] L.S.Riza, Y. Firdaus, R.A.Sukamto, Wahyudin, and K.A.Samah, “Automatic generation of short answer questions in reading comprehension using NLP and KNN, ” Multimedia Tools and Applications, pp. 1–28, 2023.
[36] A.Virani, R. yadav, P.Sonawane, and S.Jawale, “Automatic question answer generation using T5 and NLP,” International Conference on Sustainable Computing and Smart Systems, pp. 1667–1673, 2023.
[37] S.A.Lakshmi, R. Saturi, A.Bharti, M.Avvari, and B.Bhavana, “Multiple choice question generation using BERT XL NET, ” No. 10299, Easy-Chair, 2023.
[38] X.Wang, B. Liu, S.Tang, and L.Wu, “SkillQG: Learning to generate question for reading comprehension assessment, ” arXiv preprint arXiv: 2305.04737, 2023.
[39] Y.Chen, L. Wu, and M.J.Zaki, “Toward subgraph guided knowledge graph question generation with graph neural networks, ” IEEE Transactions on Neural Networks and Learning Systems, 2023.