Mathematics and Computer Science
2022; 7(4): 68-74
http://www.sciencepublishinggroup.com/j/mcs
doi: 10.11648/j.mcs.20220704.12
ISSN: 2575-6036 (Print); ISSN: 2575-6028 (Online)

PolyAQG: An Automated Question Generation Framework Based on a Combination of Four Analytical Approaches

Tee Hean Tan[1, *], Zaharin Yusoff[2]

[1]Centre for American Education, Sunway University, Sunway City, Malaysia

[2]Institute for Research, Development and Innovation, International Medical University, Kuala Lumpur, Malaysia

Email address:

teeheant@sunway.edu.my (Tee Hean Tan), zarinby@gmail.com (Zaharin Yusoff)

[*]Corresponding author

To cite this article:

Tee Hean Tan, Zaharin Yusoff. PolyAQG: An Automated Question Generation Framework based on a Combination of Four Analytical Approaches. *Mathematics and Computer Science*. Vol. 7, No. 4, 2022, pp. 68-74. doi: 10.11648/j.mcs.20220704.12

Received: July 26, 2022; **Accepted**: August 11, 2022; **Published**: August 17, 2022

Abstract: Automated Question Generation (AQG) that automatically generates questions from text effectively reduces the time and effort of educators in preparing and setting assessments questions. Since the 1970s, AQG researchers have proposed and applied a vast range of approaches to generate questions. With that, the PolyAQG Framework is proposed in this paper by combining four analytical AQG approaches into a framework to increase the number, the variety, and the quality of questions. The novelties offered by this PolyAQG Framework are highlighted in this paper. A prototype is developed under JavaFX platform and integrated with the Stanford NLP parser. From the 300 test sentences, the framework has successfully generated ten-times more questions with more than 85% of the generated questions were rated accurate and relevant. This test result revealed that the framework did successfully meet the three research objectives formulated for this study, as follows: First, a framework is developed by combining four analytical AQG techniques. Second, two phases are used to generate new sentences from the input and earlier phases are re-used to develop a new set of questions. Lastly, the generation of ontology-based questions phase does not merely generate distractors (typical in other ontology-based AQGs), but instead generates new sentences with ontology knowledge and uses them to create a new set of questions.

Keywords: Automated Question Generation, Syntax-Based, Semantic-Based, Ontology-Based

1. Introduction

Automated Question Generation (AQG) was initiated in the 1970s by John H. Wolfe [1]. Notably, he compared a set of selected sentences with predefined question patterns and automatically generated questions from the matched sentences. Since then, many researchers in this area have proposed a wide range of approaches. The 113 papers reviewed in the study of T. H. Tan et al. [2] within this research area can be categorized into two main groups: analytical and empirical approaches. The analytical approach employs the linguistic structures of the input sentence to generate questions. On the contrary, the empirical approach generates questions by observing and comparing the phenomena of the input sentences against a certain theory or some defined hypotheses.

The four analytical approaches are as follows: (1) template-based AQG, (2) syntax-based AQG, (3) semantic-based AQG, and (4) ontology-based AQG. The template-based AQG compares surface/morphological information of the input sentence against predefined question templates. Upon identifying a match, the AQG generates questions by filling up the empty placeholders of the matched templates with words from input sentence [3]. Next, syntax-based AQG leverages the syntactic features of the input sentence, whereby the AQG applies certain transformation rules to convert input sentence into question sentence [4]. On another level, semantic-based AQG comprehends the meaning of sentence lexical items to generate questions. By using knowledge from outside the sentence, ontology-based AQG substitutes words in the input sentence with concepts from the selected domain ontology knowledge-base to construct questions.

The 113 papers reviewed in the study of T. H. Tan et al [2] within this research area shown that only 6% applied more

than a single approach in their AQG. This revealed that 94% of the others employed the single approach and displayed limited choice of question varieties, question types, and number of generated questions. The use of more questions generation approaches, thus, should increase question varieties, question types, and number of generated questions. The four analytical approaches listed above are closely related to one another, mainly because they all generate questions from the structure of input sentence. Many researchers use words from input sentence to create questions, which result in limited question varieties. Therefore, this study conjectured that combining the four analytical approaches into a single framework can increase question varieties, question types, and number of generated questions.

The primary objective of this study is to increase the number of generated questions, improve the variety of questions, and provide a higher quality set of questions.

After combining the four analytical approaches [2], the outcomes of the generated questions were tested to meet the above research objectives. The syntactic-based question generation approach applies words from the original input sentence, while that in this present study transformed the input sentence syntactically into a new sentence, thus generating a new set of questions from the new sentence. Next, the ontology-based question generation approach mostly focuses on applying ontology knowledge to generate distractors. Turning to this present study, the ontology-based questions phase was embedded to substitute relevant words/concepts in the sentence with words/concepts from ontology-knowledge to generate new sentences, which later developed a new set of questions. Therefore, the uniqueness of the proposed method in this study differs from others.

The scope of this study is to generate questions from a single input sentence to develop well-formed and meaningful question sentences. However, it is noteworthy to highlight that text analysis at discourse (or paragraph) level will be performed in future work.

The rest of this paper is organized as follows: section 2 discusses the AQG approaches from prior studies. section 3 provides detailed explanation for each phase incorporated into the PolyAQG Framework. The development plan is presented in section 4 and the study is concluded in section 5.

2. Literature Review

Since the individual AQG approaches were already reviewed in the study of T. H. Tan et al. [2], this present study focused prior research work that combined more than one approach in their AQG endeavours.

Yao, Baoma, and Zhang [5] introduced the Minimal Recursion Semantics (MrsQG), which reflects the combination of semantic- and ontology-based approaches to generate questions. A transformation process was embedded to convert a single-line input sentence into the predicate logic representation and then determine the relevant WH questions (what, why, where, etc.) in order to generate semantic-based short-answer questions. In addition, hypernym relations were included by matching words on the input sentence from WordNet. Upon identifying a match, the sentence was converted based on the matching terms by the system into ontology-based questions.

Next, Becker, Basu, and Vanderwende [6] applied semantic and syntactic approaches in their AQG to generate gap-fill quiz questions. Their AQG was infused with SumBasic algorithm to retrieve sentences containing the most frequently occurring words, as well as a parser and a semantic role labeller to tag each word in the sentences. After that, the AQG generated the gap-fill questions by leaving a gap on every verb predicate, noun phrase, and adjectival phrase in the selected sentences.

Huang and He [7] applied semantic and syntactic approaches in their AQG to generate WH short-answer questions for comprehension assessment. A Lexical Functional Grammar (LFG) framework [8] was built by using WordNet to define the correlation between words and semantic-role labelling in order to capture the predicate-argument relations at clause level. The sentence meaning, on the other hand, was represented by latent semantic space. The proposed framework successfully generated questions based on syntactic and semantic information.

Most researchers used ontology-based knowledge to generate distractors for multiple-choice questions (MCQs). They used different relationships in the ontology to substitute words in the selected sentence with ontology knowledge. For instance, A. Papasalouros et al. [9] used class, property, and terminology relations; while M. Tosic and M. Cubric [10] and E. Holohan et al. [11] applied class-subclass and class-instance relationships to generate MCQ distractors. Meanwhile, M. Al-Yahya [12] employed the ontology-based approach to generate three questions formats: gap-fill, true/false, and MCQs, which concentrated on the three ontology relationships – individual, class, and property. Gap-fill questions were generated by leaving either the subject or the object from the statements. Original words (subject or entity) in true questions were replaced with ontology knowledge to generate false questions. Next, MCQ distractors were produced by retrieving relevant terms from ontology knowledge.

As such, one may conclude that no AQG thus far has supported the following scenarios:

1) Combination of the four analytical approaches into a single framework.
2) Transformation of the input sentence into a new format of sentences to generate a new set of questions.
3) Application of ontology knowledge to replace words in the input sentence to form sentences and generate a new set of questions from the new sentences.

These gaps are reflected in the novelties of the framework developed in this present study.

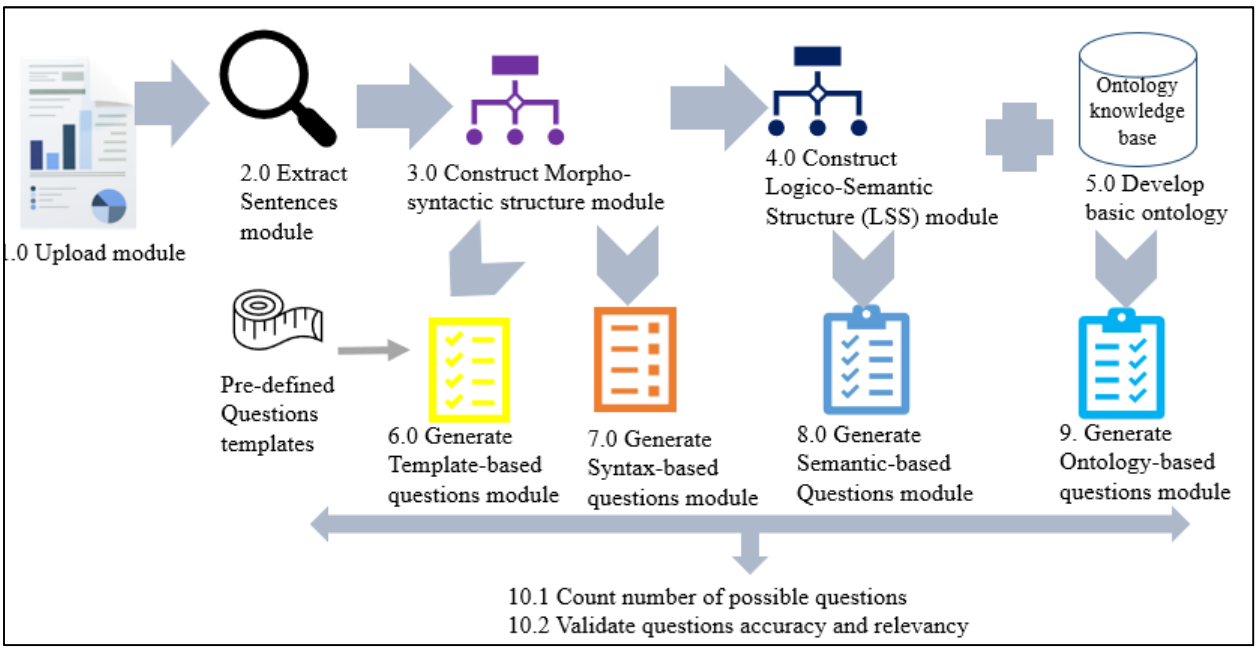

Figure 1. *The PolyAQG Framework.*

3. The PolyAQG Framework

Figure 1 illustrates the PolyAQG Framework that consists of 10 phases. The proposed framework is semi-automated and has several manual phases that involve a domain expert or the user being 1.0, 4.0, 5.0, 10.2, and predefined sentence templates. Meanwhile, the remaining phases refer to automated processes. This section is presented in the following manner:

1) Upload and extract sentences for questions generation (phases 1.0 and 2.0);
2) Generation of template-based questions (phases 3.0, 6.0, and 7.0);
3) Generation of semantic-based questions (phases 4.0 and 8.0);
4) Generation of ontology-based questions (phases 5.0 and 9.0);
5) Count and validate questions (phases 10.1 and 10.2).

3.1. Upload and Extract Sentences

This section involves the upload module (phase 1.0) and the extract sentences module (phase 2.0).

The upload module (phase 1.0) demands the user or domain expert to upload a relevant source document to generate questions. Once the system has successfully read the uploaded document, it moves on to the extract sentences module (phase 2.0) to read and extract complete sentences from the document. The system recognizes a complete sentence as a sentence that ends with a full-stop (.), question-mark (?), exclamation mark (!), single quotation mark (‘) or double quotation mark (“). Next, the system presents the extracted sentences to the user, in which the user must choose one or more of these sentences to continue the process. The process ends after the user selects the sentences.

3.2. Generation of Template- and Syntax-Based Questions

This section involves the following three phases: construction of morpho-syntactic structure module (phase 3.0), generation of template-based questions module (phase 6.0), and generation of syntax-based questions module (phase 7.0).

As depicted in our previous paper [2], the construction of morpho-syntactic structure module (phase 3.0) is composed of the following two parts: morphological and syntactic analyses. The morphological analysis consists of a series of natural language processing (NLP) processes, such as tokenization, lemmatization, and proper noun tagging. This results in a lemma list with the respective part-of-speech (POS) tags for each selected sentence. Meanwhile, the syntactic analysis results in a morpho-syntactic tree structure on each selected sentence that contains additional information of syntagmatic classes for phrases (S, NP, VP, etc.) and syntactic relations (SUBJ, OBJ, COMP, etc.).

In this study, one of the available NLP parsers, the Stanford parser, was applied in this phase (phase 3.0) mainly because it is compatible with the programming language Java deployed to develop the system. Phase 3.0 ends with a collection of lemma lists, POS lists, and morpho-syntactic structures on each selected sentence.

In the generation of template-based questions phase (phase 6.0), sentence templates are defined based on the common sentence structures and not on the question format as used in other template-based AQGs [13, 14]. Hence, the syntactic information of the input sentence is matched against the predefined sentence templates [15]. Upon identifying a match, the phase continues to generate questions. Figure 2 illustrates several examples of the sentence templates.

(i) <NN1> <has/have/is/are/am> <PP/DT/JJ>^ <NN2>
Template (i) matches the sentence with ONLY ONE verb, either has/have/is/are/am.
Example of a sentence:

POS List:	NN	VB	DT	NN
input sentence:	Keyboard	is	a	hardware

(ii) <NN1> <VB> <PP/DT/CC>^ <NN2> <PP/DT/CC>^ <NN3>^
Template (ii) is to match the sentence with ONLY ONE verb, which is NOT has/have/is/are/am
Example of sentence is:

POS List:	NN	NN	VB	PP	DT	NN	NN
input sentence:	Java	program	begins	with	a	class	definition.

Figure 2. *Examples of sentence templates.*

Phase 6.0 generates the following three formats of questions: true-false questions, gap-fill (fill in blanks), and WH short-answer questions.

The verb-fronting technique is used to generate true-false question. The system can identify the verb and its tense from the POS list of the input sentence. If the verb is in singular (plural) present tense, the question sentence will start with the "Does" ("Do") word; otherwise, the question will start with a "Did" word. Figure 3 presents an example of generating true-false question via verb-fronting.

POS list:	NN NN VBZ PP DT NN NN
Input sentence:	Java program begins with a class definition.
Question:	Does the Java program begin with a class definition?

Figure 3. *An example of generating true-false question through verb-fronting.*

The gap-fill (fill-in-the-blanks) questions are generated by replacing the noun or/and verb words with empty spaces sequentially and incrementally. The total number of gap-fill questions can be calculated as follows:

$$\sum_{i=1}^{N} Nc_i$$

where N = number of nouns + number of verbs.

Figure 4 presents an example of generating gap-fill questions by leaving out the noun and verbs sequentially and incrementally.

POS List:	NN	NN	VB	PP	DT	NN	NN
input sentence:	Java	program	begins	with	a	class	definition

From the input sentence, there is one verb (begins) and two nouns ('Java program' and 'class definition')
Number of gap-fill questions = $\sum_{i=1}^{N} 3c_i = 7$
The generated gap-fill questions are:

i. _____ begins with a class definition.
ii. Java program _____ with a class definition.
iii. Java program begins with a _____.
iv. _____ _____with a class definition.
v. _____ begins with a _____.
vi. Java program _____ with a _____.
vii. ____ _____ with a _____

Figure 4. *An example to generate true-false question via verb-fronting.*

This phase applies the proper-noun list and the POS-list to generate WH short-answer questions. A proper noun refers to a specific name for a person, place or thing. For instance, if the proper noun is a person (place), the system will generate a WH-Who (WH-Where) question; otherwise, the system should generate a WH-What question. Figure 5 shows an example of WH short-answer questions generated from the input.

POS list:	NN NN VBZ PP DT NN NN
Input sentence:	Java program begins with a class definition
Question 1: What begins with a class definition?	
Question 2: Java program begins a what?	

Figure 5. *An example of the WH short-answer questions.*

The sentence templates in phase 6.0 serve as the base to generate questions in phases 7.0 and 9.0. When generation of syntax- and ontology-based questions modules (phases 7.0 & 9.0, respectively) have developed new sentences from the input sentence, the new sentences are then moved to phase 6.0 in order to match the new sentences against the predefined sentence templates. This process should generate a new set of questions, thus increasing the number and the varieties of the generated questions.

The generation of syntax-based question module (phase

7.0) does not produce questions directly. Instead, it retrieves additional information on syntagmatic classes and syntactic relations to reorder several elements, thus transforming the input sentence into new sentences. Next, this phase sends the new sentences to generate template-based questions (phase 6) in order to yield a new set of questions. Two transformations are performed: one is to convert the input sentence into its opposite voice (from active-voice sentence to passive-voice sentence or vice versa), and the second is to reorder the objects and complements in the input sentence. An active-voice sentence reflects the subject being in action, whereas a passive-voice sentence denotes that the subject is subjected to an action by someone or something. In this case, the infused algorithm swaps the subject (object) in the active-voice sentence to become the object (subject) in the passive-voice sentence, and vice versa. Figure 6 illustrates an example of such transformations. The resulting sentences are sent to the construction of morpho-syntactic structure module (phase 3.0) to generate the syntactic lists and to match the list against the sentence templates in template-based question module (phase 6.0) to develop a new set of questions.

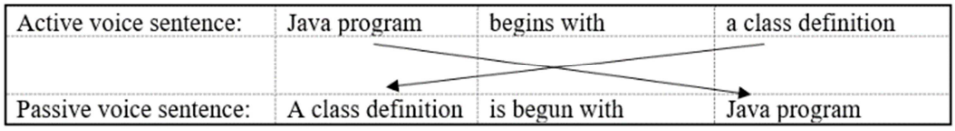

Figure 6. *An example of transforming an active-voice sentence into a passive-voice new sentence.*

3.3. Generate Semantic-Based Questions

This section involves 2 phases, namely: construction of Logico-Semantic Structure (LSS) module (phase 4.0) and generation of semantic-based questions (phase 8.0).

Apart from the entities treated earlier, it is impossible to generate WH-Where and WH-When short-answer questions from the syntactic lists generated in the previous phase. Instead, semantic information is required to develop deep questions [7].

The LSS module (phase 4) uses semantic parser to construct LSS. The structure shows the relationships (LSS) between the lexical items (especially nouns & verbs) in the sentence and labels of each lexical item with its semantic feature.

This module can use semantic features to generate new WH short-answer questions, such as WH-where (from location (LOC)), WH-whom (from human (HUM)), WH-when (from TIME), etc. Additional information on LSS relations enables further questions, such as for From-Where (SOURCE), Where-To (DESTINATION), Why (CAUSE), etc.

Figure 7 displays an instance of a logical-semantic structure based on the sentence: John drove Mary to school every weekday.

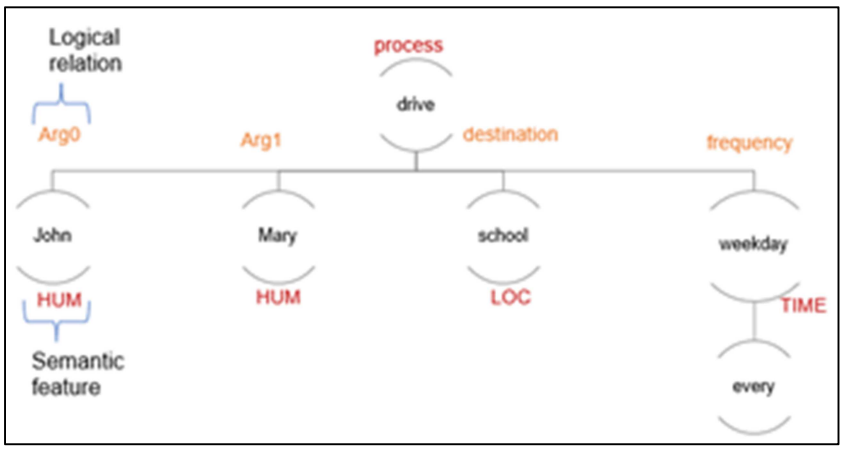

Figure 7. *An example of a logico-semantic structure.*

Referring to the LSS, the following new WH-short answer questions are generated:

1) Who drives Mary to school every weekday?
2) John drives who to school every weekday?
3) Where does John drive Mary?
4) When does John drive Mary to school?

3.4. Generate Ontology-Based Questions

This last section involves 2 phases, namely: development of basic ontology module (phase 5.0) and generation of ontology-based questions module (phase 9.0).

Ontology-based AQG has been widely applied by researchers to generate distractors [12, 16, 17]. Turning to this present study, ontology-based knowledge is applied to replace words or concepts from the input sentence to yield new sentences. Later, a new set of questions is generated from the new sentences by using the earlier phases, with some of the new sentences remaining TRUE and some turning to FALSE, though still related and do make sense.

The development of basic ontology module (phase 5.0)

demands the domain expert to set up an ontology knowledge base on the selected domain prior to system operation. The domain refers to Java programming language. The ontology knowledge serves as a source of external knowledge (vis-à-vis input sentence) to generate more questions. The ontology represents the conceptual description of specific contents, terms, and relationships in a given knowledge domain [18].

Out of the numerous relationships that exist in ontologies, three primary relations are applied to set up the ontology knowledge base in phase 5.0. These relationships are vertical, lateral, and others. Vertical relations look for terms that possess either inheritance (is-a) or aggregation (part-of) relation. Lateral relations include either synonymy or antonym. In light of other relations, 'belongs to' or 'owner of' relations are considered.

Figure 8 presents instances of relations used to set up the ontology knowledge base for the Java programming language domain.

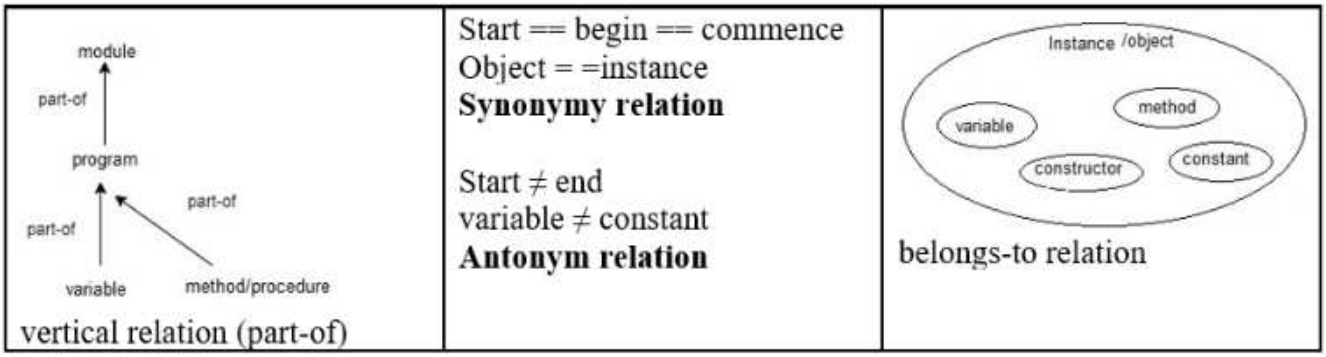

Figure 8. *Instances of ontology knowledge base with different relations.*

This module matches words from input sentence against knowledge base. Upon identifying a match, the word is replaced with the matched knowledge-based word to create new sentences. This step is executed in phase 9.0 (ontology-based questions module).

With these new sentences, phases 3.0, 6.0, and 7.0, as well as phases 4.0 and 8.0, are re-called to generate the corresponding new set of questions.

As mentioned earlier, new sentences generated in phase 9.0 may retain their TRUE value or turn to FALSE; the latter may be applied as distractors:

Vertical: A is-a B, C part-of D

Replace: B → A (TRUE); A → B (FALSE)

Replace: C → D (FALSE); D → C (FALSE)

Lateral: A synonym-of B, C antonym-of D

Replace: B → A (TRUE); A → B (TRUE)

Replace: C → D (FALSE); D → C (FALSE)

Others: A belongs-to B, C owner-of D

Replace: B → A (FALSE); A → B (FALSE)

Replace: C → D (FALSE); D → C (FALSE)

3.5. Count and Review Questions

All the generated questions are saved in the central repository and automatically counted in the Count number of possible questions module (phase 10.1). The final phase, validation of questions accuracy and relevancy module (phase 10.2), is performed manually by the domain experts. These experts validate the accuracy of the questions in terms of grammar usage, ambiguity of words, question structure/format, question relevancy in terms of the level of knowledge required (Bloom's taxonomy), and coverage of questions. Next, the domain user deletes all invalid and irrelevant questions from the central repository. Validated questions are applied to generate the assessment paper or other related assessment processes.

4. Development and Testing

A prototype system is developed to test the PolyAQG Framework using the JavaFX platform. The Stanford NLP parser is deployed to perform all the required syntactic transformations. The prototype system uses the MySQL database to store all the extracted sentences, the generated questions, and the selected questions. The Java programming language domain ontology is set up for testing purpose. The algorithm is developed to construct the semantic structure as no suitable ready-made parser is readily and freely available.

As a result, the domain experts recommended 300 sentences to generate questions. Table 1 tabulates the outcomes.

Table 1. *Test results.*

Input sentences 300	Template-based questions	Syntax-based questions	Semantic-based questions	Ontology-based Questions
Number of questions	3000	3000	502	5500
Accurate questions	2760 (92%)	2760 (92%)	438 (87%)	4730 (86%)
Relevant questions	2100 (76%)	2120 (76%)	400 (91%)	4300 (90%)

From the 300 input sentences, the prototype system had successfully generated 10 times more questions; which resulted in 3000 questions from the template- and syntax-based questions, 5500 questions from the ontology-based questions, but only 438 from the semantic-based questions (not all input sentences had extra semantic information). Next, 92% of the generated template- and syntax-based questions were rated as accurate questions because these questions used most of the words from the input sentences. Nevertheless, only 87% and 86% of semantic- and ontology-based questions, respectively, were rated as accurate and required human manual corrections to address grammatical errors, sentence structures, etc. Next, only 76% of template- and syntax-based questions were rated as relevant questions due to the generation of numerous similar questions, especially for gap-fill questions. In addition, 91% and 90% of semantic- and ontology-based questions, respectively, were rated as (more) relevant questions, as the questions did not use all words from input sentences. This made the new questions to differ from the input sentence, thus increasing the difficulty level of the questions and posing a more challenging task to the test takers.

5. Conclusion

This paper describes the mechanism of PolyAQG Framework, which combined the four common analytical AQG approaches to increase the number and the variety of questions, as well as to provide a higher quality set of questions. Besides that, we have two of the phases to generate new sentences from the input and re-used the earlier phases to develop a new set of questions. The results from the prototype system revealed that the proposed framework has successfully met the three objectives formulated for this study, in line with the study novelties.

References

[1] J. H. Wolfe, “An Aid to Independent Study through Automatic Generation (Autoquest),” SIGCSE ’76 Proc. ACM SIGCSE-SIGCUE Tech. Symp. Comput. Sci. Educ., p. Pages 104–112, 1975, [Online]. Available: https://doi.org/10.1145/800107.803459.

[2] T. H. Tan, P. L. Teh, and Y. Zaharin, “PolyAQG Framework: Auto-generating assessment questions,” in 2021 IEEE Computing Conference (ICOCO 2021), 2021, p. 5.

[3] E. Sneiders, “Automated question answering using question templates that cover the conceptual model of the database,” Lect. Notes Comput. Sci. (including Subsea. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 2553, pp. 235–239, 2002, DOI: 10.1007/3-540-36271-1_24.

[4] G. Kurdi, J. Leo, B. Parsia, U. Sattler, and S. Al-Emari, “A Systematic Review of Automatic Question Generation for Educational Purposes,” Int. J. Artif. Intell. Educ., vol. 30, no. 1, pp. 121–204, 2020, DOI: 10.1007/s40593-019-00186-y.

[5] X. Yao, G. Bouma, and Y. Zhang, “Semantics-based Question Generation and Implementation,” Dialogue & Discourse, vol. 3, no. 2, pp. 11–42, 2012, DOI: 10.5087/dad.2012.202.

[6] L. Becker, S. Basu, and L. Vanderwende, “Mind the gap: Learning to choose gaps for question generation,” NAACL HLT 2012 - 2012 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. Proc. Conf., no. June 2012, pp. 742–751, 2012.

[7] Y. Huang and L. He, "Automatic generation of short answer questions for reading comprehension assessment," Nat. Lang. Eng., vol. 22, no. 03, pp. 457–489, 2016, DOI: 10.1017/s1351324915000455.

[8] R. M. Kaplan and J. Bresnan, “Lexical-Functional Grammar: A Formal System for Grammatical Representation,” in The Mental Representation of Grammatical Relations, J. Bresnan, Ed. MIT Press.

[9] A. Papasalouros, K. Kanaris, and K. Kotis, “Automatic Generation Of Multiple Choice Questions from Domain Ontologies,” IADIS Int. Conf. e-Learning 2008, Amsterdam, Netherlands, 2008.

[10] M. Tosic and M. Cubric, “SeMCQ – Protégé Plugin for Automatic Ontology-Driven Multiple Choice Question Tests Generation,” Proc. 11th Int. Protege Conf., pp. 6–7, 2009, [Online]. Available: http://hdl.handle.net/2299/6710.

[11] E. Holohan, M. Melia, D. Mcmullen, and C. Pahl, “Adaptive E-Learning Content Generation based on Semantic Web Technology,” Int. Work. Appl. Semant. Web Technol. E-Learning (SW-EL 2005) 12th Int. Conf. Artif. Intell. Educ. AIED 2005, pp. 1–8, 2005.

[12] M. Al-Yahya, “OntoQue: A question generation engine for educational assessment based on domain ontologies,” Proc. 2011 11th IEEE Int. Conf. Adv. Learn. Technol. ICALT 2011, pp. 393–395, 2011, DOI: 10.1109/ICALT.2011.124.

[13] A. E. Awad and M. Y. Dahab, "Automatic Generation of Question Bank Based on Predefined Templates," Dahab Int. J. Innov. Adv. Comput. Sci. IJIACS ISSN, vol. 3, no. 1, pp. 2347–8616, 2014.

[14] A. Fabbri, P. Ng, Z. Wang, R. Nallapati, and B. Xiang, “Template-Based Question Generation from Retrieved Sentences for Improved Unsupervised Question Answering,” pp. 4508–4513, 2020, DOI: 10.18653/v1/2020.acl-main.413.

[15] H. Hussein, M. Elmogy, and S. Guirguis, "Automatic English question generation system based on template-driven scheme," Int. J. Comput. Sci. Issues, vol. 11, no. 6, p. 45, 2014.

[16] V. E., T. Alsubait, and P. S. Kumar, “Modeling of Item-Difficulty for Ontology-based MCQs,” 2016, [Online]. Available: http://arxiv.org/abs/1607.00869.

[17] K. Stasaski and M. A. Hearst, “Multiple Choice Question Generation Utilizing An Ontology,” no. 2011, pp. 303–312, 2018, DOI: 10.18653/v1/w17-5034.

[18] S. Ou, C. Orasan, D. Mekhaldi, and L. Hasler, “Automatic question pattern generation for ontology-based question answering,” Proc. 21th Int. Florida Artif. Intell. Res. Soc. Conf. FLAIRS-21, pp. 183–188, 2008.