Multiple-Choice Item Generator for Narrative and Declarative Texts

Deo Jonathan J. Guillermo, Leonil N. Llona, Madeline L. Tolentino and Richard B. Domingo, and Ria A. Sagum

***Abstract*—Multiple Choice Item Generator is a research focused on generating grammatically and semantically correct questions and generating plausible distractors. The system's framework has three main modules: pre-processing, question generation and distractor selection. Pre-processing module implements Hobbs Algorithm for anaphora resolution and direct to indirect speech conversion for handling quoted statements. In generating questions, question overgeneration and ranking framework of Heilman and Smith is used. The distractor selection module uses collocation extraction and Wordnet-based methods such as Lin's semantic similarity measure, hypernyms, hyponyms and maximal bipartite graph matching .**

***Index Terms*—Hobbs algorithm, question overgeneration, multiple-choice item generator.**

I. Introduction

Question Generation (QG) is the task of automatically creating series of questions from a specific input text [1]. Researchers have confirmed that humans are not very skilled in asking good questions [2]. QG can be a tool for both learning and assessment. With respect to education, QG mechanism might improve and diversify the questions posed to student by tutoring systems leading to more natural and effective students-tutor interactions [3].

Multiple-choice item is a selection-type item which presents students with a set of possible options from they are to choose the correct or the best answer [4]. They are most widely used for measuring knowledge, comprehension, and application outcomes. A couple of factors showed advantage of multiple choice questions, thus eliciting these types of question as a standard assessing tool.

This paper introduces a Multiple-Choice Item Generator (MCIGen) for Narrative and Declarative texts.

The research implemented the algorithm of Hobbs in Pronoun Anaphora Resolution; Heilman and Smith's Question Generation via Over generation and Ranking; Similarity-Based Distract or Selection using Collocation Patterns and Lin's Measure. The system creates multiple-choice items from narrative or declarative texts.

Manuscript received November 28, 2013; revised February 23, 2014. This work was supported in part by the Polytechnic University of the Philippines. Multiple Choice Item Generator for Narrative and Declarative Texts.

The authors are with the College of Computer and Information Sciences, Polytechnic University of the Philippines (e-mail: riasagum31@yahoo.com, deo_jonathan22@yahoo.com, leonilllona@yahoo.com, tolentinomadeline@ymail.com, richardglen_domingo@outlook.com).

II. System Architecture

A. Pre-Processing Module

Pre-Processing module has three main processes. These are parsing of input file, direct to indirect sentence conversion and anaphora resolution. This module mainly prepares the input for the next module to process. The outputs of this module are sentences, parse trees, tokens, POS tags and NER tags. Noun phrases and noun frequencies are prepared for Anaphora Resolution and Distractor Selection.

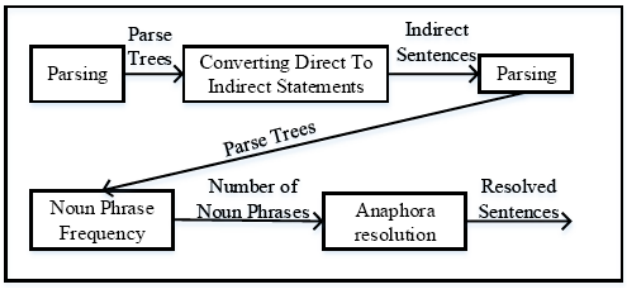

Fig. 1. Pre-processing module.

Fig. 1 shows the composition of the pre-processing module. The input is read from a text file (.txt). MCIGen is solely dependent on input. Sentence splitting is included in this parser. MCIGen used Stanford's Parser and NER tagger. The output of parsing process is a set of Sentence object. Every instance of a Sentence class has an instance of its parse tree and a token list. The input text is expected to be in narrative/declarative form, which may include phrases enclosed in quotation marks - *direct phrases.*Direct to indirect statement conversion starts by extracting the *direct phrase* from the quotation marks and then separated from the unquoted phrase. *Direct phrase* refers to the phrase enclosed in quotation marks. After separating the two phrases, *the direct phrase* and the unquoted phrase are combined using a connector (that). The pronouns "I" and "Me" inside the quotation phrase are referred. "I" and "me" referral comes from the subject of the unquoted phrase.

After conversion, the sentences are re-parsed, and tokens are tagged again, for the changes made in conversion. Noun phrases from the newly created parse trees will then be counted for NP Frequency. Reparsing of sentences is relatively faster than the initial parsing. Unconverted sentences will not undergo the process of parsing anymore. A phrase whose parent from the parse tree is a Noun Phrase (NP) is counted for anaphora resolution and distractor selection.

The list of sentences' parse tree is traversed to search for the deepest NP nodes. By 'deepest', it aims to search NP nodes not containing any more NPs. Resolution is done for every pronoun that is encountered in the sentences. This is

done in three steps; first, it finds possible candidate antecedent for every pronoun using Hobbs' Algorithm Accordingly [5]; then, it filters invalid antecedent of a pronoun - this reduces the set of possible antecedents to select from; third, it scores the listed candidates based on features given by Mitkov [6] and some added features. And lastly, the top scored candidate then replaces the pronoun. This process goes on for every pronoun that is encountered.

B. Question Generation Module

The question generation module covers: sentence selection by scoring, generating possible questions (what, who, when, and where) from the sentences, and question ranking to produce good questions. Sentence scoring is based from summer.

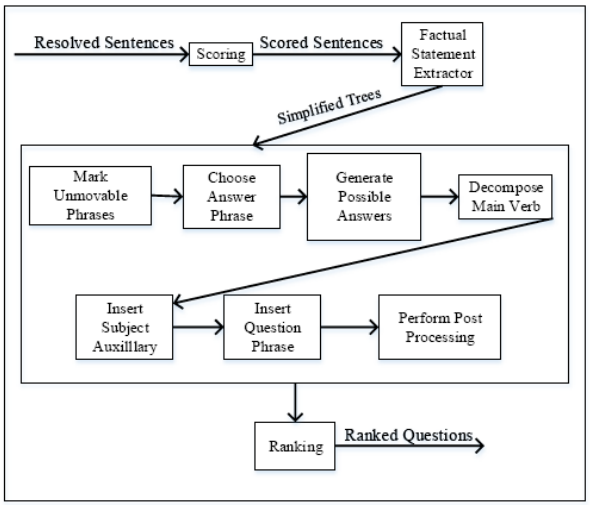

Fig. 2. question generation module.

Fig. 2 shows the question generation module of the system. From the list of pre-processed sentences, the main task of scoring is to choose and filter sentences that are good for generating questions. Sentences are scored by using sentence-level features introduced from a summarization. The scores given by each features are computed to get the total score of the question. Sentences are scored through its sentence-level and word-level features. Sentence-level features are: the position of the sentence and, the presence of verb and referring pronoun in the sentence. Word length, noun-word frequency, familiarity in the input text, named-entity tag, and POS tag are word-level features. The score contribution of each feature score to the total score was based on their importance and relevance in generating questions. Named-entity signifies that the word is not a common noun. Words that are tagged as a named entity in this feature will be scored. Part of speech tag is also essential to know the importance of the word. Not all words with tag are considered in this feature. The tags are considered in scoring are nouns, verbs and adjectives.

The scores obtained from the sentence-level features were summed up and added to the score of the sentence. In word-level features, the score of the word is obtained from the product of the scores of the features. The scores per word from the sentence are summed up to obtain the score of the sentence. Scored sentences were processed and less scored sentences were removed. The sentences having scores below the 60% of the highest score are removed. The sentence scorer takes a list of sentences and returns reduced list of sentences.

The list of scored sentences will be processed again to simplify complex sentences into simpler sentences. To do this, an API by Heilman and Smith called Factual Sentence Extraction was used. The API also uses supersense tagger in tagging each word in the sentence. The tagger was used to determine what WH-phrase is applicable for the answer phrase. The sentences from the API will be the ones used for Question Over generation.

For each sentence, there are several possible questions that can be generated. The first step on the process is to mark phrases in the sentence that cannot be answer phrases. These phrases are marked as unmovable (-UNMOVABLE) due to the fact that they cannot be used as islands of movement. This was implemented using Tregex expressions and Tsurgeon operations by the Stanford Parser which is used to modify the parse tree of the sentence. A series of Tregex expressions are used to find unmovable phrases on the parse tree. Finding and marking these unmovable phrases will give way in finding all possible answer phrases in the sentence, this process is called Question Transducer. It will iterate over all the answer phrases in the sentence, and attempt to generate a question for each of them. The process involves generating question phrase, main verb decomposition, subject-auxiliary inversion and question phrase insertion. A duplicate of the sentence's parse tree will be made for the process to avoid modification of the original parse tree.

Verb phrase that must be decomposed is identified using Tregex. For example, Peter drank the juice after decomposition of the verb became Peter did drank the juice in the question. Next step in transformation is subject-auxiliary inversion that take the task of transferring the decomposed verb in the beginning of the sentence for example in the above sentence, after the subject auxiliary inversion took place, the sentence became did Peter drank the juice. After inverting the subject and verb, question phrase is inserted then the answer phrase is deleted the example above became "What did Peter drank?"

Ranking is task to rank questions since over generation generated many question. Not all questions generated are acceptable. Some question is erroneous due some reasons. The ranking of questions are based from Heilman and Smith statistical ranking of question generation.

The questions are ranked considering the features given by Heilman and Smith. This includes length features, WH-words, N-gram language models, negation, grammatical features, transformations, vagueness, and histogram. This feature set was developed by an analysis of questions generated. Questions are ranked by the predictions of a logistic regression model of question acceptability.

C. Distractor Selection

The Distractor Selection Module process involves two major processes: to propose candidate distractors and to rank the candidates according to their semantic similarity scores. To determine the word similarity score, Wordnet Similarity is used to perform Lin's method of measuring semantic relatedness [7]. In determining similarity score between two phrases, a maximal bipartite graph is constructed first. The graph is formed using the weight of every pair of tokens from two phrases. The weight is defined by the similarity score

between the pair of tokens using Lin's measure. After forming a bipartite graph, the Hungarian Algorithm is used to determine the maximal weight of the graph which would determine the similarity score between the two phrases [8]. The scores produced will be used in both filtering and ranking of the candidate distractors produced through Collocation Extraction and WordNet.

Collocation extraction is the process of finding distractors from the input text. The questions generated by MCIgen are: factual WH (what, where and when) questions only, if so, then the answer phrases must be noun phrases (NP's). In this case, the process only searches for NP's and proposes them as candidate distractors. The collected NP's will proceed to a filtering process in which several constraints are considered such as: their similarity scores, their length in comparison with the correct answer, and their NER tag if it agrees with the type of the question (e.g What, When or Where). After the filtering process, the system will check if the number of distractors is sufficient for output. If the number of distractors is not enough, then the tool WordNet is consulted to give additional distractors. If the size of the candidates exceeds the limit, then relatedness scoring will be performed to rank the candidates, choose the top three on rank, and then send them for post-processing and output.

The tool Wordnet is used to propose distractors by utilizing Hyponyms and Hypernyms of a given concept. In proposing candidates, the system asks Wordnet to generate the hyponyms of each hypernym in a given concept. It gives hypernyms of single words only, not of multi-word phrases. To solve this matter, if a multi-word phrase is detected, the system finds a way to mark the most significant word or key in the phrase. The marked word will be the one sent to Wordnet for hypernym generation. The produced set of hyponyms will then proceed to a filtering process by performing different constraints to ensure valid candidates as distractors. Distractors that are too close or too distant semantically to the correct answer are not of good quality; thus, the filter removes hypernyms that satisfy these constraints by having allowable maximum and minimum semantic scores. The filter also removes hypernyms that are semantically equal to the answer.

The generated set after the filter will be added as distractors if its size meets the required number, or else if the size is greater, only the top distractor(s) via their similarity score will be added. Once a question has its choices ready, it will immediately be sent for post-processing and output.

III. Implementation

MCIGen was evaluated in terms of: correctness of the questions generated which, encompasses the syntactic and semantic correctness of the questions, difficulty of the questions, accuracy which involves the precision, recall and f-scores and lastly the plausibility of the distractors was computed using item analysis.

To measure the correctness of the questions, 3 English teachers from Pinagbuhatan High School (PHS) and 2 English teachers from St. Joseph Catholic School (SJCS) were asked to rate the questions generated by MCIGen on the rate of 1 to 4 (4 being the highest) and an overall ranking of 8 to each questions. The required respondents supposed to be, are 5 English teachers and a number of students from a school. However, retrieving the questionnaires from SJCS has been very hard and time consuming so the researchers decided to perform another evaluation on the teachers of PHS. PHS provided 2 teachers and SJCS provided 5 teachers to answer the evaluation. The data used for item analysis also came from examinations performed on Grade 5 students of SJCS.

IV. Results

The computation resulted to an average semantic correctness of 2.75 out of 4 and an average syntactic correctness is 2.94 out 4. To sum it all up, the total correctness of the generated questions is 5.69 out of 8.

TABLE I: Question Difficulty

Question Set	**Average Difficulty**
1	**2.04**
2	**2.13**
3	**2.77**
4	**2.43**
5	**2.80**
Overall Average Difficulty = 2.27	

Table I shows the overall average difficulty of the questions generated. The question difficulty was evaluated using 5 Question Sets, and averaged all the difficulty of all the questions generated which makes the overall average difficulty of 2.27 out 4. This means that most of the questions generated are questions with easy difficulty.

TABLE II: Evaluation Results for Precision, Recall and F-Score

Question Set	Correct	Spurious	Missed	Precision	Recall	F-Score
1	4	32	10	0.11	0.29	0.16
2	9	66	10	0.09	0.47	0.16
3	4	18	4	0.18	0.50	0.27
4	11	37	5	0.23	0.69	0.34
5	2	17	11	0.11	0.15	0.13
All Questions	30	192	40	0.14	0.43	0.21

Table II shows that the overall precision of the system is 0.14 out of 1, recall is 0.43 out of 1 and f-score is 0.21 out of 1.

Lastly, item analysis was used to determine the total number of plausible distractors. In item analysis, the researchers divided a class of Grade 5 students into two groups (upper 50% and lower 50%). The rule for distractor plausibility or attractiveness is that at least 3% of students from each group must be attracted [9]. The item analysis showed results that there are a total of 27 plausible distractors produced by the system, and the rest are considered non-plausible and should be revised. There are a total of 69 non-plausible distractors, 45 of those distractors have attracted students from lower group but completely ignored by the upper group and 18 are completely ignored by both groups.

V. Conclusion

MCIGen's performance is 2.94 out of 4 or 73.50% in generating questions that are grammatically correct; and 2.75 out of 4 or 68.75% in generating semantically correct questions. The assessed difficulty of the generated questions had an average of 2.27 out of 4. The values obtained from the assessed difficulty conclude that most of the system-generated questions have an easy difficulty. The system's accuracy in terms of precision is 0.14. The reason behind this low score is that the manually generated questions are very few compared to the system's overgenerated questions. The recall is 0.43 as some of the manually generated questions contains questions that cannot be generated by the system (e.g. how and why). This resulted to a large number of "missed" questions and it became a huge factor in the low recall. The combined low precision and recall resulted to an f-score of 0.21. In addition, the evaluation for the accuracy is not that suitable for the study because the manual generation by experts generated less questions compared with the system; these factors made the precision, recall, and f-score obtained low values. It can be concluded that these results are not yet conclusive until a proper implementation of the study is performed. The plausibility of the generated distractors is 28.13%. The reason behind this is that most of the generated distractors are only effective for students coming from the lower group and was completely ignored by the upper group.

VI. Recommendation

Recommendations are based from the result of this research's evaluation and implementation. Points listed below is intended to improve the output of the system, question generation and distractor selection, and increase evaluation and implementation score. To address recommendations clearly, they are elicited per module.

Pre-processing is the sole basis for the output of the system. In order for a natural language to be processed correctly by the computer, the language itself should be prepared and turned into a computer process-able language. Preprocessing does the said process. There are however a number of encountered problems in this module.

1) Improvised algorithm for direct to indirect sentence conversion failed to follow rules mentioned for conversion (Present Simple to Past Simple, Past Simple to Past perfect, etc.)
2) Direct phrases with period, exclamation point and question mark inside the quotation marks were not handled properly causing premature sentence splitting.
3) Noun Phrase Frequency counting is flawed when a supposedly two same noun phrase are counted as two different noun phrase (a shiny red apple, the red apple)
4) Anaphora resolution introduces few more challenges:
    a) Pleonastic pronoun it refers to an invalid antecedent.
    b) Hobbs algorithm fails to candidate a valid antecedent
    c) Antecedent selection feature scoring needs reconsidering and thorough study
    d) Three more features should be considered for antecedent selection feature, Semantic consistency and Semantic and Syntactic parallelism.

For question generation module, the following processes needs improvement.

1) Sentence selection - It is better to select sentence that that has more noun phrases and select sentences that conveys information about the topic of the input text.
2) In feature-based selection of sentence - more feature can be added to improve the selection process.
3) Supersense tagging - use a tagger that the answer phrase based on its function in the sentence (e. g. Cat is living in a jet plane. Cat must be 'Who' in question not 'What').
4) For the generation of question, addition of question phrases to generate such as why, how and more accurate question phrases for a specific answer phrases.
5) In ranking of the generated questions, it is better to have other methods to use in scoring or to add more features to improve filtering questions.

For generating distractors, the main challenge for future researchers is to improve its plausibility.

Acknowledgment

The researchers cannot express enough gratitude to our beloved Chairperson, Michael DelaFuente and Professor Jude MikhaelLuzuriaga Cruz for their readiness to answer any questions inquired at any point in time, even at the critical moments. The completion of this project could not have been accomplished without St. Joseph Catholic School and Pinagbuhatan High School, whose institution showed welcome and interest in the field of learning, especially Mr. Manuel S. Are for his time and kindness. To all our families especially to the Guillermo family, who gave us home to work on, gave us abundant amounts of food, love and support. It was a great comfort and relief to know that they were willing to provide all things we ever needed. Our heartfelt thanks. And of course, to the undeniable presence of God, for the unending grace we receive every day that made all things possible and right.

References

[1] P. Paul, "Question generation shared task and evaluation challenge – status report," in *Proc. 9th International Conference on Computational Semantics*, pp. 318-320, 2011.
[2] A. Graesser and V. Rus. The Question Generation Shared Task and Evaluation Challenge. [Online] Available: http://qg.cs.memphis.edu/TheQuestionGenerationSharedTaskAndEvaluationChallenge.pdf
[3] M. Heilman and N. A. Smith. Question Generation Via Over Generating Transformations and Ranking. [Online] Available: http://www.lti.cs.cmu.edu/research/reports/2009/cmulti09013.pdf
[4] W. Kuechler and M. Simkin. (2005). Multiple-Choice Tests and Student Understanding: What is the Connection?*Decision Sciences Journal of Innovative Education* [Online]. 3(1) pp. 73-97. Available: http://www.coba.unr.edu/faculty/kuechler/cv/DSJIE.3.1.05.pdf
[5] J. Hobbs. Resolving Pronoun References. Lingua. [Online] Available: http://www.isi.edu/~hobbs/ResolvingPronounReferences.pdf
[6] R. Mitkov, "Factors in anaphora resolution: they are not the only things that matter. a case study on two different approaches," in *Proc. 5th International Workshop on Natural Language Generation*, Dawson, Pennsylvania, 1990, pp. 14-21.
[7] R. Mitkov, L. Ha, A. Varga, and L. Rello, "Semantic similarity of distractors in multiple-choice tests: extrinsic evaluation," in *Proc. EACL 2009 Workshop on GEMS: GEometrical Models of Natural Language Semantics*, Athens, Greece, 2009, pp. 49–56.

[8] S. Bhagwani, S. Sataphaty, and H. Karnick, "Sranjans: semantic textual similarity using maximal weighted bipartite graph matching," in *Proc. 1st Joint Conference on Lexical and Computational Semantics*, Montreal, Canada, 2012, pp. 579–585.
[9] D. Gutierrez, *Assessment of Learning Outcomes (Cognitive Domain)*, Malabon City, Philippines: Kerusso Publishing House, 2007, pp. 92-94.

Ria A. Sagum was born in Laguna, Philippines on August 31, 1969. She received her bachelor of computer data processing management from the Polytechnic University of the Philippines and Professional Education at the Eulogio Amang Rodriguez Institute of Science and Technology. She received her master's degree in computer science from the De La Salle University in 2012.

She is currently teaching at the Department of Computer Science, College of Computer and Information Sciences, Polytechnic University of the Philippines in Sta. Mesa, Manila and a lecturer at the Information and Computer Studies, Faculty of Engineering, University of Santo Tomas in Manila.

Ms. Sagum has been a presenter at different conferences, including the 2012 International Conference on e- Commerce, e-Administration, e-Society, e-Education, and e-Technology and National Natural Language Processing Research Symposium. She is a member of different professional associations including ACMCSTA and an active member of the Computing Society of the Philippines, Natural Language Processing Special Interest Group.

Madeline L. Tolentino was born in Mandaluyong City, Manila Philippines on March 6, 1993. She is taking Bachelor of Science in Computer Science, at Polytechnic University of the Philippines. A high-school graduate with honors. She is very interested in database and project management and eager to learn more in computer programming. She was a web programmer in Management in Information System Department at the Department of Justice during her On-The-Job training.

Deo Jonathan J. Guillermo was born in Manila Philippines on July 22, 1994. He is an undergraduate student form the Polytechnic University of the Philippines, currently taking bachelor of science in computer science. Has knowledge in programming, algorithm design, database and project management. Eager to learn more from the field of Computing, he intends to be an academic one day and be involved in different researches for computer science like natural language processing.

Leonil N. Llona was born in the Province of Albay, Philippines on November 26, 1993. He is a student in Polytechnic University of the Philippines and he received him bachelor's degree in computer sciences. He had been worked on different project in web, software engineering and compiler design and artificial intelligence. His research interest is in artificial intelligence and natural language processing.

Richard Glen B. Domingo was born on June 15 1993 in Quezon City, Metro Manila. He graduated elementary as top 4 of his batch and received his high school diploma as top 6 in the same school Saint Joseph Catholic School in Project 3, Quezon City. He is currently taking a computer science degree in Polytechnic University of the Philippines in Sta. Mesa, Manila. He took his on-the-job training as a .NET developer. With his background on android development, software engineering and artificial intelligence, he strives as a leader of a promising start-up software development group, Optika Studios.