Automatic Generation of Fill-in-the-blank Question with Corpus-based Distractors for E-Assessment to Enhance Learning

Bidyut Das
Department of Information Technology,
Haldia Institute of Technology,
Haldia, West Bengal, India,
Email iD: bidyut2002in@gmail.com, ORCID iD: 0000-0002-8588-1913

Mukta Majumder
Department of Computer Science and Application,
University of North Bengal,
Darjeeling, West bengal, India,
Email iD: mukta_jgec_it_4@yahoo.co.in, ORCID iD: 0000-0003-2608-5762

Santanu Phadikar
Department of Computer Science and Engineering,
Maulana Abul Kalam Azad University of Technology,
Kolkata, West Bengal, India
Email iD: sphadikar@yahoo.com, ORCID iD: 0000-0002-7620-5518

Sk. Arif Ahmed
Department of Physics and Technology
UiT The Arctic University of Norway, Norway
Hansine Hansens veg 18, 9019 Troms, Norway
Email iD: skarifahmed@gmail.com, ORCID iD: 0000-0003-0706-2565

August 2, 2019

Abstract

Knowledge acquisition is the prime objective of a learner from an educational system and evaluating the learner's knowledge is the eventual goal of an examination process. This paper introduces a system which is able to produce fill-in-the-blank questions to test the knowledge of a learner that he or she has accumulated after reading a course material. The question generation task is subdivided into three modules: sentence selection, answer-key identification and question formation along with distractors generation. The sentence is selected using a coarse-grain part-of-speech tagset. The answer-key is extracted by identifying topic-word in the sentence and question is formed by omitting this topic-word from the sentence. This paper also highlights an efficient corpus-based distractors generation technique to produce multiple-choice fill-in-the-blank test items.

Keywords: Automatic question generation; Fill-in-the-blank question; Coarse-grain tagset; Topic-word; Corpus-based distractors

1 Introduction

Question generation has become an emerging field of research in Educational Technology and Natural Language Processing [1–4]. While knowledge acquisition is the primary motto of a learner from an educational system, testing the learner knowledge is the ultimate target of assessment or evaluation process [5–7]. It requires questions to judge the content knowledge of the learner [8–10]. Questions are mainly two categories: subjective question and objective question [11]. With the advantage of quick and real-time evaluation, objective type test items are receiving major importance from intelligence tutorial system (ITS) and active learning classroom framework [12, 13].

Objective type test item requires to choose the correct answer from a set of alternatives or to provide a phrase or word to complete a statement. Fill-in-the-blank, true-false and multiple-choice questions (MCQ) are popularly used objective test items to test the learner's knowledge from lecture notes or learning materials [14]. Generation of the handcrafted assessment item is extremely time-consuming and laborious. However, the system for automatic question generation can leverage the advantage of ITS and active learning framework. To make the assessment process easier and less laborious; questions with alternatives are the best choice to test the knowledge of the learner [15]. Fill-in-the-blank is one of the popularly used assessment tool in REAP (REAder-specific Practice) tutoring system [16].

Fill-in-the-blank test item is alternately known as cloze question(CQ) where a sentence is given with one or more gaps in it with four alternative answers to fill those gaps [17, 18]; unlike the WH questions where someone has to generate test items with *When*, *Where*, *Who*, *Which*, etc. [19]. For a decade automatic CQ generation has taken a lot of attention from the researchers [20].

The proposed work has concentrated to generate fill-in-the-blank questions with alternative answer set to test the learning gap of a learner. The majority of the sentences of a text are not suitable for generating good quality questions. Therefore, the informative sentence selection task catches our attention to generate questions. A Coarse-grain part-of-speech tagset has been incorporated here to select the informative sentences. The answer-key is identified by selecting topic-word in the sentence. The selected topic-word is omitted to generate the question or stem from an informative sentence. Finally, a corpus-based distractors selection technique is illustrated to generate multiple-choice fill-in-the-blank test items. A graphical representation of the proposed fill-in-the-blank questions generation system is depicted in Figure 1.

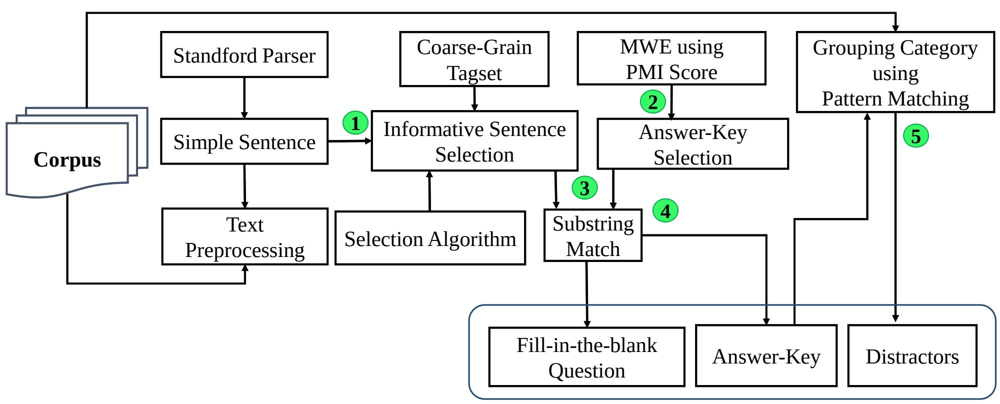

Figure 1: A system for generating fill-in-the-blank questions

2 Related Work

The system generates fill-in-the-blank questions with multiple-choice answers which include three steps: sentence selection, identifying answer-key and question formation along with distractors generation. We have discussed some of the previous works related to our tasks as follows.

Hill and Simha [21] proposed an approach of generating fill-in-the-blank test items with multiple-choice answers for testing a reader's comprehension abilities and contextual awareness. They used the Google ngram corpus and an application of word co-occurrence likelihoods to pick words with strong contextual links to their surrounding text. The authors also generated distractors that make sense in an isolated narrow context of the passage. Kumar et al. [22] proposed RevUP system for generating gap-fill questions. To pick valuable sentences from texts, they proposed a sentence ranking approach with the help of topic distributions, taken from topic models. To select keywords from each picked sentence, they collected human annotations using the Amazon Mechanical Turk (AMT). This data was applied for training a classifier to predict the significance of gaps and obtaining accuracy of 81.0%. Finally, they proposed an innovative way to select distractors those were semantically related to the gap-phrase and had contextually fit in the gap-filling question. Kumar et al. [23] also proposed a fill-the-blank question generator for self-assessment of students. It provided tools for teachers to generate and edit questions from their lecture notes. The accuracy of 94% for question sentences, 87% gaps and 60% distractors were considered to be relevant. Rus et al. [24] proposed a method to automatically generate gap-fill questions by exploiting recorded data from massive online education environments such as DeepTutor. Knoop and Wilske [25] presented a smartphone application for learners of English that instantaneously generated gap-filling exercises from a source text, with options (key and distractors). Sakaguchi et al. [17] proposed discriminative approaches to generate semantic distractors for fill-in-the-blank quiz using a large corpus. The methods had been satisfying both validity and reliability of generating distractors. To avoid many answers in a single quiz, the distractors were exclusive against the answer key. Kurtasov [9] described a method for generating cloze questions from the Russian text. It consisted of three stages: sentence splitting, sentence filtering and question generation. The system was able to process texts with the morpho-syntactic features of the language and recognized a sentence's subject. Narendra et al. [20] described an automated system to generate a set of significant cloze questions using an English article. They utilized a summarizer (MEAD) for identifying valuable sentences for generating CQs. They also presented evaluation guidelines to evaluate CQG systems for Cricket World Cup 2011 data. Mostow and Jang [26] described a system DQGen (Diagnostic Question Generator) which used NLP techniques to generate diagnostic cloze questions for checking student's comprehension knowledge. DQGen was developed to reduce disruption in the reading process and diagnose a number of comprehension failures. Becker et al. [27] proposed an approach for generating quizzes from online text. It had two parts: selection of sentences and identifying gaps in the resulting sentences to generate questions. They applied a summarization technique to identify key sentences from a passage and trained a classifier to select gaps in the sentence for asking questions. Agarwal and Mannem [28] proposed a method to generate gap-filling questions from a biology textbook using heuristically weighted features. They employed a set of features like *'sentences length'*, *'is it the first sentence'*, *'does it contain abbreviations or superlatives'*, *'position of the sentence in a document'*, *'number of nouns and pronouns'*, *'contains the token which occurred in the title'* etc. They didn't utilize any external knowledge and just rely on information given in the document for generating distractors. But they did not mention how to integrate these features and what would be the optimum value of these. Smith et al. [29] proposed a system, TEDDCLOG (Testing English with Data Driven CLOze Generation), to automatically generate test items from a test corpus. TEDDCLOG had taken the correct answer as input and obtained distractors from a distributional thesaurus. Hoshino and Nakagawa [30] described a semi-automated system to generate cloze questions from news articles. The cloze questions were formed by omitting words from a passage and the readers were asked to fill the omitted words. Two different types of distractors, grammar distractors and vocabulary distractors were produced by the system. Their evaluation revealed that 80% of the generated questions were appropriate. We have observed that some of the existing systems used summarizer for question sentence identification [20, 27]. Though obtaining reasonable accuracy in sentence selection using a summarizer is highly questionable.

3 Proposed Method

The proposed fill-in-the-blank question generation method is subdivided into the following steps: sentence selection, answer-key identification and question formation (stem creation) along with distractors generation.

3.1 Sentence selection and answer-key identification

Sentence Selection: To generate fill-in-the-blank questions, sentence selection is the task of selecting informative sentences from the corpus that carry proper information for testing the knowledge of learners. An algorithm is proposed here for selecting informative sentences using coarse-grain part-of-speech tagset. A coarse-grain tagset is a set of subcategories of original part-of-speech tags derived from Penn Treebank tagset [31]. The coarse-grain tagset is derived by mapping NNP and $NNPS$ into $NNP^\wedge$ (Proper Noun); similarly VB, VBD, VBG, VBN, VBP and VBZ are grouped into $VB^\wedge$ (Verb) etc. Table 1 shows the proposed coarse-grain tagset. The tags other than mentioned in Table 1 are same with the original Penn Treebank tagset.

Table 1: List of part-of-speech tags used for sentence selection

Penn-Treebank tagset	**Coarse-Grain tagset**	**Description**
DT, PDT, WDT	^DT	Determiner
JJ, JJR, JJS	JJ^	Adjective
NN, NNS	NN^	Noun
NNP, NNPS	NNP^	Proper Noun
PRP, PRP$	PRP^	Pronoun
WP, WP$	WP^	Wh-Pronoun
RB, RBR, RBS, WRB	^RB^	Adverb
VB, VBD, VBG, VBN, VBP VBZ	VB^	Verb

Das et al. [5] studied the dependency structures of the input corpus, generated by Stanford Parser [32] to separate the simple sentences from other sentences. The number of *'nsubj'* or *'nsubjpass'* is counted from the dependencies. A simple sentence has only one *'nsubj'* or *'nsubjpass'*. The *'nsubj'* and *'nsubjpass'* are categorized as subject according to *'Stanford Typed Dependency Manual'* [33].

The sentence length is the easiest technique for selecting worthy sentences. Heuristically, we have checked the simple sentences with the word length 8 to 25 are more suitable in our corpus for generating questions. For selecting our informative sentences, we have only considered those simple sentences which contain 8 to 25 words and have at least two disjoint NNP^ tags without any ^RB^ tag. Now, the simple sentences are further fine-tuned based on the coarse-grain tag set by applying the Algorithm 1.

In the algorithm *"S_i starts with $(NNP^\wedge)^*$ followed by $(VB^\wedge)^*$"* means a sentence S_i begins with one or more $NNP^\wedge$ tags and next one or more $VB^\wedge$ tags would appear consecutively; *"S_i has $^\wedge DT$ followed by $(NNP^\wedge)^*$"* means one or more $NNP^\wedge$ tags would appear sequentially after $^\wedge DT$. Similarly, *"S_i has $^\wedge DT$ followed by $(NN^\wedge)^*$, IN and $(NNP^\wedge)^*$"* means after $^\wedge DT$ tag one or more $NNP^\wedge$ tags, an IN tag and finally one or more $NNP^\wedge$ tags occur sequentially. Note that, any sentence containing $^\wedge DT$ tag is selected based on similar approach.

Algorithm 1 Sentence Selection

Require: Simple sentences $S = (S_1, S_2, S_3......S_N)$ of Corpus C
Ensure: A set of informative sentences D
  for $i = 1$ to N **do**
    if (S_i starts with $(NNP^\wedge)^*$ followed by $(VB^\wedge)^*$) **then**
      $S_i \in D$
    else if (S_i has $^\wedge DT$ followed by $(NNP^\wedge)^*\|$
  S_i has $^\wedge DT$ followed by $(NN^\wedge)^*$ and $(NNP^\wedge)^*\|$
  S_i has $^\wedge DT$ followed by $(NN^\wedge)^*$, IN and $(NNP^\wedge)^*\|$
  S_i has $^\wedge DT$ followed by $(JJ^\wedge)^*$ and $(NNP^\wedge)^*\|$
  S_i has $^\wedge DT$ followed by $(JJ^\wedge)^*$, $(NN^\wedge)^*$ and $(NNP^\wedge)^*\|$
  S_i has $^\wedge DT$ followed by $(JJ^\wedge)^*$, $(NN^\wedge)^*$, IN and $(NNP^\wedge)^*$) **then**
      $S_i \in D$
    end if
  end for

Answer-key identification (AKI): In answer-key identification, a word or a group of words is selected as the correct answer from an informative sentence. The fill-in-the-blank question has one correct answer-key and three to four wrong options which are called distractors. The answer-key is omitted with a blank for generating a question sentence or stem. Each informative sentence consists of topic-word which is either single-word (unigram) or multiword (ngram). It has been observed that the multiword key is more suitable than the single-word key to become an appropriate answer-key. So, we have first decided to find the multiword answer-key from an informative sentence. If there is no multiword key available in the sentence, then we have considered the unigram key for generating question. The number of words in the multi-word key is restricted up to three in our experiment to avoid the long answer-key. The answer-key identification task is subdivided into three stages: (a) sentence preprocessing, (b) multi-word extraction and (c) question formation with answer-key.

(A) Sentence preprocessing: The sentence is preprocessed in such a way that all the punctuation marks are removed from the sentence. We have split a sentence and create a new line when a punctuation mark occurred within it. Next, we have eliminated the stopwords which are not important to the domain. We have again introduced a new line where a stopword is present in the sentence. After this, using coarse-grain part-of-speech tagger, we have filtered the words that have tags like $NNP^{\wedge}$ and CD (proper noun and number) without changing the line sequence of the words. This helps us to determine the frequency of unigram, bigram and trigram in the corpus which are useful in our experiment.

(B) Multiword extraction (MWE): The pointwise mutual information(PMI) association technique is used for identifying the set of multiword for obtaining candidate answer-keys. The higher the association score of a multiword has more potential to become an answer-key. Let us consider, m be the number of useful unigrams present in the corpus, then

$$F_1 = \{F(t_1), F(t_2), ..., F(t_m)\} \tag{1}$$

Where $F(t_i)$ be the frequency of unigram t_i $(1 \leq i \leq m)$

$$F_2 = \{F(t_1, t_2), F(t_2, t_3), ..., F(t_{m-1}, t_m)\} \tag{2}$$

Where $F(t_i, t_j)$ be the frequency of bigram (t_i, t_j) $(1 \leq i \leq m-1$ and $2 \leq j \leq m)$

$$F_3 = \{F(t_1, t_2, t_3), F(t_2, t_3, t_4), ..., F(t_{m-2}, t_{m-1}, t_m)\} \tag{3}$$

Where $F(t_i, t_j, t_k)$ be the frequency of trigram (t_i, t_j, t_k) $(1 \leq i \leq m-2$, $2 \leq j \leq m-1$ and $3 \leq k \leq m)$. Note that, after splitting the sentences, when t_i, t_j and t_k are not present in the same line, the frequency of $F(t_i, t_j)$ or $F(t_i, t_j, t_k)$ is not considered in the set F_2 and F_3 respectively. The PMI association scores of multiwords (bigrams and trigrams) are calculated in set F_4.

$$F_4 = \{PMI(t_i, t_j), PMI(t_i, t_j, t_k)\} \tag{4}$$

The PMI is described for bigram as follows

$$PMI(t_i, t_j) = log_2[(P(t_i, t_j)/\{P(t_i).P(t_j)\}] \tag{5}$$

Where $P(t_i, t_j)$ is the joint probability of two words t_i and t_j coming sequentially in a text and $P(t_i)$ and $P(t_j)$ are the probabilities of t_i and t_j appearing individually in the text, respectively. $P(t_i, t_j) = P(t_i)P(t_j)$ signifies the two words are independent of each other and $PMI(t_i, t_j) = 0$ indicates that these two words are not good candidates for answer-key. A high PMI score indicates the bigram as an answer-key. Similarly, PMI for the three words t_i, t_j and t_k is given by

$$PMI(t_i, t_j, t_k) = log_2[(P(t_i, t_j, t_k)/P(t_i).P(t_j).P(t_k)] \tag{6}$$

Where $P(t_i, t_j, t_k)$ is the probability of three words t_i, t_j and t_k coming sequentially in a text and $P(t_i)$, $P(t_j)$ and $P(t_k)$ are the probabilities of t_i, t_j and t_k appearing individually in the text, respectively. $P(t_i, t_j, t_k) = P(t_i)P(t_j)P(t_k)$ signifies these three words are independent of each other and $PMI(t_i, t_j, t_k) = 0$ indicates that these three words are not good candidates for generating an answer-key. A high PMI score indicates the trigram is a good option to be an answer-key.

(C) Question formation with answer-key: In the next stage, a keyword matching approach is applied to extract the answer-key. A multiword in the informative sentence, which has maximum number of words (trigram or bigram) and highest PMI score in set F_4, is omitted to generate a question or stem. When an informative sentence does not have any multiword, an unigram is chosen, whose frequency is highest in the set F_1. The following two sentences have been considered for the detail explanation:

'Acharya Vinoba Bhave received serious brickbats in 1975 for supporting the state of emergency imposed by the then prime minister Indira Gandhi'

'Ramkumar obliged and sent for Gadadhar to join him at Dakshineshwar to assist him in the daily rituals'.

The two multiwords, *'Acharya Vinoba Bhave'* and *'Indira Gandhi'*, are matched with the substring of the first sentence. The association score of *'Acharya Vinoba Bhave'* is greater than *'Indira Gandhi'* because it is observed that after co-reference resolution *'Gandhi'* appears individually many times in the corpus with *'Indira Gandhi'*, *'Rajiv Gandhi'* and *'Mahatma Gandhi'*. Therefore, we have omitted *'Acharya Vinoba Bhave'* from the above sentence to generate the fill-in-the-blank question and selected *'Acharya Vinoba Bhave'* as the answer-key.

Question: ____ received serious brickbats in 1975 for supporting the state of emergency imposed by the then prime minister Indira Gandhi.

Answer: Acharya Vinoba Bhave

In the second sentence, there is no match for the multiword key. So, we have extracted a match for the unigram key. The three unigrams, *'Ramkumar'*, *'Gadadhar'* and *'Dakshineshwar'*, are found in the sentence. The frequency of *'Gadadhar'* is higher than the other two words in F_1; so, we have omitted the word to generate a question and *'Gadadhar'* is taken as the answer-key.

Question: Ramkumar obliged and sent for ____ to join him at Dakshineshwar to assist him in the daily rituals.

Answer: Gadadhar

3.2 Corpus-based distractors selection

Distractors, which are the wrong answers set among the alternatives in a multiple-choice test item, make the question an interesting and popular one. The distractors are similar enough to answer-key and their purpose is to confuse the learner to give the correct answer [34,35]. Normally, WordNet, domain ontologies or knowledge base is used to find similar or related words for generating distractors. Here, we have used a pattern search approach to generate domain-specific distractors, which is achieving good accuracy. The proposed corpus-based distractors selection task is composed of two sub-tasks, described as follows.

Distractors list generation using pattern search approach: From the corpus, we have searched for a few closely related ngrams ($NNP^\wedge$ and CD and here, $n \geq 1$) to create a list for generating distractors. To identify the distractors categories, we have processed the corpus and separate the closely related ngrams into different groups. The web pages from where the input corpus is taken (biographies of leaders and social reformers), contain a set of information in a structured format [36]; such as *'date-of-birth'*,*'place-of-birth'*, *'father's name'*, *'mother's name'*, *'spouse name'*, *'children name'* etc. Additionally, most of the pages contain a set of links for connecting the pages from one to another (top-right corner). Those fields are extracted to get the names of *'leaders'* and *'social reformers'*. Next, we have collected a list of related ngrams using a search pattern for the same category in the corpus. For a *'Father's name'*, we have run a search pattern *'Parents: <ngram/s>(Father)'*. Similarly, for *'Children's name'*, we have searched a pattern *'Children: <ngram/s>'*. If we have found multiple ngrams separated with a comma(,) or 'and' or semicolon (;) in a search query, then the last ngram is taken and put into the list of distractors' category.

For example, *'Place of Birth: Ratnagiri, Maharashtra'*; here, we have considered the last ngram of the search query *'Maharashtra'* as an entry of the distractors list of *'Place of Birth'*. Similarly, *'Children: Ramabai Vaidya, Parvatibai Kelkar, Vishwanath Balwant Tilak, Rambhau Balwant Tilak, Shridhar Balwant Tilak and Ramabai Sane'*; here, the last ngram is *'Ramabai Sane'* in the list of children. We have taken one child name in the possible distractors list to avoid the ambiguity of multiple entries of child's name for same leaders or social reformers. Similarly, we run the search query for other categories. From the search result, we have extracted a set of similar entities. The similar entity is defined as a distractors list of different categories.

Selection of distractors: From the distractors list prepared earlier, we have picked three entities as the distractors of a question based on similarity of word lengths close or equal to answer-key. Unigram ($n = 1$) answer-key has unigram, and ngram ($2 \leq n \leq 3$) answer-key has ngram or (n-1) gram distractors. For example, answer-key *'Bal Gangadhar Tilak'* has three words (n=3). Therefore, distractors list for it contains three words (n) or two words (n-1), like 'Indira Gandhi', 'Jawaharlal Nehru' and 'Lala Lajpat Rai'; otherwise, single-word answer-key *'Yashodabai'* has single word distractors like *'Radhabai'*, *'Laxmibai'* and *'Chimnabai'*.

4 Results and Discussion

Since the fill-in-the-blank question contains multiple components, different approaches we have adopted for assessing the quality of the individual components. No standard dataset has been found in the literature to measure the correctness of individual components. Most of the systems in the literature have been assessed by human evaluators [37]. So, we have created a test data set through which the system is evaluated with the help of human evaluators. Five evaluators are employed to check the correctness of the system generated results. The accuracy is estimated using the confusion matrix [38] in Table 2. Precision can be seen as a measure of exactness or quality, whereas recall is a measure of completeness or quantity. In simple terms, high precision means that a system returned substantially more relevant results than irrelevant ones, while high recall means that a system returned most of the relevant results. Precision is more important than recall in the area of question generation where the exactness of generating question is more important than the completeness. Therefore, the accuracy is measured here in terms of precision and high precision indicates the efficiency of the system.

Table 2: The confusion matrix, precision, recall and F-score

		Predicted	
		YES	NO
Actual	YES	TP	FP
	NO	FN	TN

Accuracy	$\frac{TP+TN}{TP+FP+FN+TN}$
Precision	$\frac{TP}{TP+FP}$
Recall	$\frac{TP}{TP+FN}$
F-score	$2 \times \frac{Precision \times Recall}{Precision+Recall}$

The system has been tested using the web documents. The test corpus is created by extracting the web pages of fourteen *Indian leaders* [39] and eleven *Indian social reformer's* [40]. The test corpus has twenty-five documents that consist of 1893 sentences is shown in Table 3.

Table 3: The dataset used in our experiments

Categories	Web pages	Number of sentences
Leaders	Bal Gangadhar Tilak	81
	Bhagat Singh	78
	Chandra Shekhar Azad	75
	Gopal Krishna Gokhale	76
	Indira Gandhi	82
	Jawaharlal Nehru	77
	Lal Bahadur Shastri	63
	Lala Lajpat Rai	56
	Maulana Abul Kalam Azad	68
	Netaji Subhash Chandra Bose	74
	Rajendra Prasad	86
	Rajiv Gandhi	75
	Sardar Vallabhbhai Patel	78
	Sarojini Naidu	79
Social Reformers	Acharya Vinoba Bhave	81
	Baba Amte	76
	Dr. B. R. Ambedkar	75
	Ishwar Chandra Vidyasagar	77
	Jyotiba Phule	73
	Mother Teresa	85
	Raja Ram Mohan Roy	74
	Ramakrishna Paramhansa	77
	Shahu Chhatrapati	69
	Swami Dayanand Saraswati	76
	Swami Vivekananda	82

Table 4 illustrates the results of sentence selection. From 1893 sentences in the input corpus, the system has identified 1236 simple sentences. After preprocessing, we get 344 sentences. Out of 344, 136 sentences are selected as informative sentences for generating fill-in-the-blank questions. Therefore, the system required a large corpus as input for generating questions. But the correctness of the system signifies that it generates good quality questions that are useful in the automated assessment item generation.

Table 5 displays the identification result of answer-key. A comparative study of PMI with RAKE [41] is shown in Figure 2 for identifying answer-key from the

Table 4: The accuracy of informative sentence identification from the corpus

<table>
<tr><th>Corpus</th><th>Sentences</th><th>Simple sentences</th><th>Sentences after preprocessing</th><th>Informative sentences</th><th>Correct Informative sentences</th><th>Precision(%)</th><th>F-score (%)</th></tr>
<tr><td>Leaders</td><td>1048</td><td>694</td><td>202</td><td>74</td><td rowspan="3">Evaluator 1: 130
Evaluator 2: 134
Evaluator 3: 133
Evaluator 4: 132
Evaluator 5: 131</td><td rowspan="3">97.06</td><td rowspan="3">58.47</td></tr>
<tr><td>Social Reformers</td><td>845</td><td>542</td><td>142</td><td>62</td></tr>
<tr><td>Total</td><td>1893</td><td>1236</td><td>344</td><td>136</td></tr>
</table>

informative sentences. The system generates various types of questions based on the grouping of distractors. We have grouped the distractors by *'name'*, *'father's name*, *'mother's name'*, *'date-of-birth'* and *'date-of-death'* using pattern search technique. Table 6 presents the accuracy of distractors generation.

Finally, we have conducted a pilot test and taken the average feedback of five students to present the overall accuracy of our proposed system that is shown in Figure 3. Table 7 shows five sample questions with distractors that are generated by the proposed system.

Table 5: The accuracy of answer-key identification from the identified informative sentences using PMI

<table>
<tr><th>Corpus</th><th>Answer-Key Identified</th><th>Correct answer-key</th><th>Precision (%)</th><th>F-score (%)</th></tr>
<tr><td>Leaders</td><td>74</td><td>73</td><td rowspan="3">96.32</td><td rowspan="3">52.45</td></tr>
<tr><td>Social Reformers</td><td>62</td><td>59</td></tr>
<tr><td>Total</td><td>136</td><td>131</td></tr>
</table>

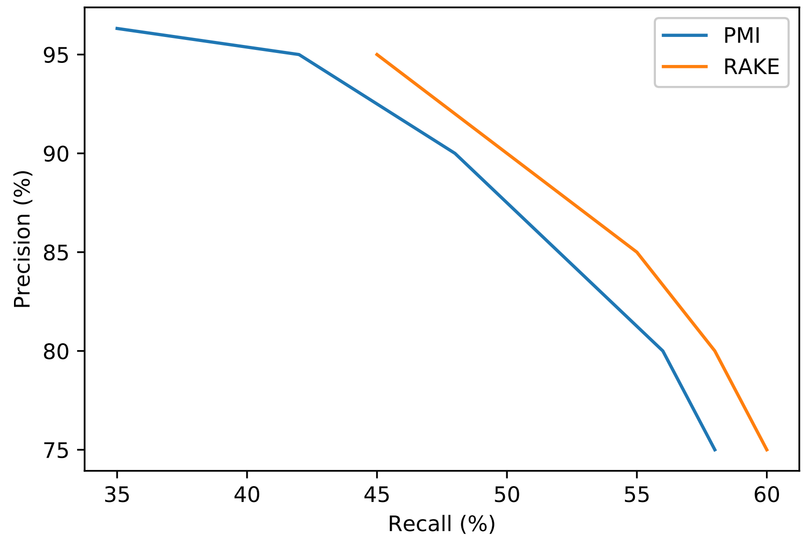

Figure 2: Answer-Key Identification: A comparative study of PMI with RAKE

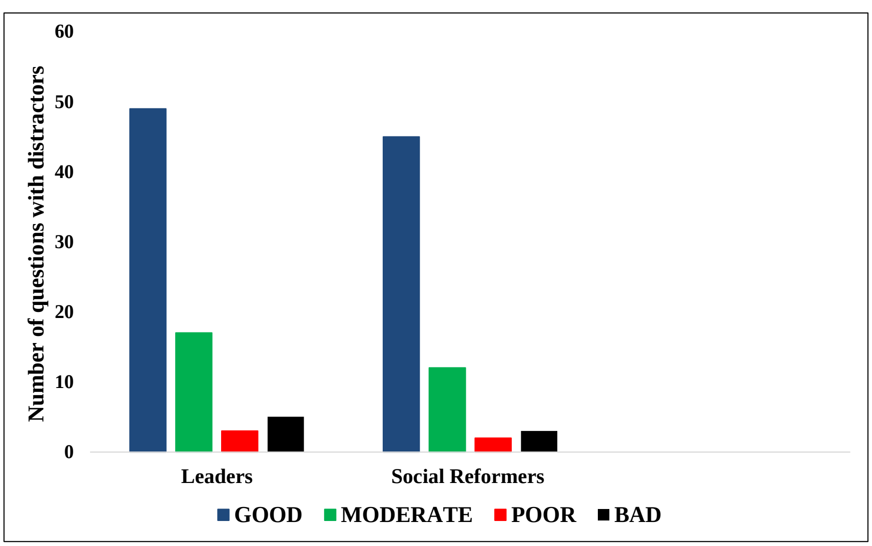

Figure 3: A pilot test: The average feedback of students

Table 6: The corpus-based distractors selection accuracy of our proposed system

Leaders				
Distractors list (grouping by)	**Generated questions with distractors**	**Relevant question with distractors**	**Precision (%)**	**F-score (%)**
Name	65	62	93.24	62.54
Father Name	4	3		
Mother Name	2	2		
Children Name	3	2		
Total	74	69		
Social Reformers				
Distractors list (grouping by)	**Generated questions with distractors**	**Relevant question with distractors**	**Precision (%)**	**F-score (%)**
Name	56	53	95.16	67.25
Father Name	3	3		
Mother Name	2	2		
Children Name	1	1		
Total	62	59		

Table 7: The questions with distractors generated by the automated system

Question Type	**Question**	**Answer**	**Answer with Distractors**
Name	___ joined the Indian National Congress in 1890.	Bal Gangadhar Tilak	Indira Gandhi Bal Gangadhar Tilak Jawaharlal Nehru Lala Lajpat Rai
Name	For higher studies ___ went to a Sanskrit Pathashala in Varanasi.	Chandrasekhar Azad	Jawaharlal Nehru Rajiv Gandhi Chandrasekhar Azad Sarojini Naidu
Date of Death	Dr. Prasad passed away after suffering from brief illness for around six months on February 28 ___.	1963	1958 1945 1991 1963
Father Name	Sri Ramakrishna Paramhansa was born as Gadadhar Chattopadhyay on February 18, 1836, to ___ and Chandramani Devi.	Khudiram Chattopadhyay	Khudiram Chattopadhyay Devidas Amte Hakurdas Bandyopadhyay Govindrao Phule
Mother Name	Swami Dayanand Saraswati was born on February 12 1824 in Tankara Gujarat as Mool Shankar to Karshanji Lalji Tiwari and ___.	Yashodabai	Radhabai Laxmibai Chimnabai Yashodabai

5 Conclusion

We have proposed here an automated system that generates multiple-choice fill-in-the-blank question with one correct answer and three distracters. In the proposed technique the identification of informative sentence is performed by using coarse-grain part-of-speech tag set. Topic-word or domain-specific word is selected to create the answer-key and the question is generated by omitting this topic-word from the selected sentence. Next, an efficient corpus-based distractors selection approach is presented to create the wrong answers that distract the learner to give the correct answer. The overall accuracy of our system highlights that it is helpful to generate suitable questions for the learner's assessment purpose.

References

[1] Beg Azam, Beg Ajmal. Using open technologies for automatically creating question-and-answer sets for engineering MOOCs *Computer Applications in Engineering Education.* 2018;26:617–625.

[2] Pabitha P, Mohana M, Suganthi S, Sivanandhini B. Automatic question generation system in *International Conference on Recent Trends in Information Technology (ICRTIT)*:1–5Institute of Electrical and Electronics Engineers (IEEE) 2014.

[3] Bednarik Laszlo, Kovacs Laszlo. Implementation and assessment of the automatic question generation module in *3rd International IEEE Conference on Cognitive Infocommunications (CogInfoCom)*:687–690Institute of Electrical and Electronics Engineers 2012.

[4] Liu Ming, Calvo Rafael A, Rus Vasile. Automatic question generation for literature review writing support in *International Conference on Intelligent Tutoring Systems*:45–54Springer 2010.

[5] Das Bidyut, Majumder Mukta. Factual open cloze question generation for assessment of learners knowledge *International Journal of Educational Technology in Higher Education.* 2017;14:1-12.

[6] Brown J C, Frishkoff G A, Eskenazi M. Automatic Question Generation for Vocabulary Assessment in *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language*

Processing:819-826Association for Computational Linguistics, Canada 2005.

[7] Antonis Konstantinos, Daradoumis Thanasis, Papadakis Spyros, Simos Christos. Evaluation of the effectiveness of a web-based learning design for adult computer science courses *IEEE Transactions on Education.* 2011;54:374–380.

[8] Yildirim Mehmet. A genetic algorithm for generating test from a question bank *Computer Applications in Engineering Education.* 2010;18:298–305.

[9] Kurtasov A.. A System for Generating Cloze Test Items from Russian-Language Text in *Proceedings of the Student Research Workshop associated with RANLP*:107–112 2013.

[10] Lee John, Seneff Stephanie. Automatic generation of cloze items for prepositions in *Proceedings of Interspeech*:2173–2176International Speech Communication Association (ISCA), Antwerp, Belgium 2007.

[11] Liu Zhe, Jansen Bernard J. Subjective versus objective questions: Perception of question subjectivity in social Q&A in *International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction*:131–140Springer 2015.

[12] Shaban Abdul-Majeed Selman. A comparison between Objective and subjective tests *Journal of College of Languages.* 2014;30:44-52.

[13] Graesser Arthur C, Chipman Patrick, Haynes Brian C, Olney Andrew. AutoTutor: An intelligent tutoring system with mixed-initiative dialogue *IEEE Transactions on Education.* 2005;48:612–618.

[14] Divate Manisha, Salgaonkar Ambuja. Automatic question generation approaches and evaluation techniques. *Current Science (00113891).* 2017;113.

[15] Ramesh S, Sidhu S Manjit, Watugala GK. Exploring the potential of multiple choice questions in computer-based assessment of student learning *Malaysian Online Journal of Instructional Technology.* 2005;2:1–15.

[16] Heilman Michael, Collins-Thompson Kevyn, Callan Jamie, Eskenazi Maxine. Classroom success of an Intelligent Tutoring System for lexical practice and reading comprehension in *Ninth International Conference on Spoken Language Processing*:829–832 2006.

[17] Sakaguchi Keisuke, Arase Yuki, Komachi Mamoru. Discriminative Approach to Fill-in-the-Blank Quiz Generation for Language Learners in *Association for Computational Linguistics(2)*:238–242 2013.

[18] Marrese-Taylor Edison, Nakajima Ai, Matsuo Yutaka, Yuichi Ono. Learning to Automatically Generate Fill-In-The-Blank Quizzes *arXiv preprint arXiv:1806.04524.* 2018.

[19] Majumder M, Saha S K. Automatic selection of informative sentences: The sentences that can generate multiple choice questions *Knowledge Management and E-Learning: An International Journal).* 2014;6(4):377-391.

[20] Narendra A, Agarwal M, Shah R. Automatic Cloze-Questions Generation in *Proceedings of Recent Advances in Natural Language Processing*:511-515Hissar, Bulgaria 2013.

[21] Hill Jennifer, Simha Rahul. Automatic Generation of Context-Based Fill-in-the-Blank Exercises Using Co-occurrence Likelihoods and Google n-grams in *Proceedings of the Workshop on Innovative Use of NLP for Building Educational Applications*:23–30 2016.

[22] Kumar Girish, Banchs Rafael, D'Haro Luis Fernando. Revup: Automatic gap-fill question generation from educational texts in *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*:154–161 2015.

[23] Kumar Girish, Banchs Rafael E, D'Haro Luis Fernando. Automatic fill-the-blank question generator for student self-assessment in *Frontiers in Education Conference (FIE)*:1–3Institute of Electrical and Electronics Engineers(IEEE) 2015.

[24] Rus Vasile, Stefanescu Dan, Niraula Nobal, Graesser Arthur C. Deeptutor: Towards macro-and micro-adaptive conversational intelligent tutoring at scale in *Proceedings of the first ACM conference on Learning@ scale conference*:209–210Association for Computing Machinery 2014.

[25] Knoop Susanne, Wilske Sabrina. WordGap-Automatic generation of gap-filling vocabulary exercises for mobile learning in *Proceedings of the second workshop on NLP for computer-assisted language learning, NEALT Proceedings Series 17*no. 086:39–47Linköping University Electronic Press 2013.

[26] Mostow Jack, Jang Hyeju. Generating diagnostic multiple choice comprehension cloze questions in *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*:136–146Association for Computational Linguistics 2012.

[27] Becker Lee, Basu Sumit, Vanderwende Lucy. Mind the gap: learning to choose gaps for question generation in *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*:742–751Association for Computational Linguistics 2012.

[28] Agarwal M, Mannem P. Automatic gapfill question generation from text books in *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*:56–64Association for Computational Linguistics, Stroudsburg, PA, USA 2011.

[29] Smith Simon, Avinesh PVS, Kilgarriff Adam. Gap-fill tests for language learners: Corpus-driven item generation in *Proceedings of ICON: 8th International Conference on Natural Language Processing*:1–6 2010.

[30] Hoshino A, Nakagawa H. Assisting cloze test making with a web application in *Proceedings Of Society for Information Technology and Teacher Education International Conference*:2807-2814San Antonio, Texas 2007.

[31] Santorini B.. Part-of-speech tagging guidelines for the Penn Treebank Project (3rd Revision) tech. rep.Technical Reports (CIS)http://www-rohan.sdsu.edu/ corpora/corpus_course/tagguid1.pdf 1990.

[32] Lex-Parser 2019. https://nlp.stanford.edu/software/lex-parser.shtml.

[33] De Marneffe Marie Catherine, Manning Christopher D. Stanford typed dependencies manual tech. rep.Technical report, Stanford University-https://nlp.stanford.edu/software/dependencies_manual.pdf 2008.

[34] Karamanis N, Ha R. Generating multiple-choice test items from medical text: A pilot study in *Proceedings of the Fourth International Natural Language Generation Conference*:111-113Association for Computational Linguistics 2006.

[35] Susanti Yuni, Tokunaga Takenobu, Nishikawa Hitoshi, Obari Hiroyuki. Automatic distractor generation for multiple-choice English vocabulary questions *Research and Practice in Technology Enhanced Learning.* 2018;13:15.

[36] Cultural India 2019. https://www.culturalindia.net/.

[37] Patra Rakesh, Saha Sujan Kumar. A hybrid approach for automatic generation of named entity distractors for multiple choice questions *Education and Information Technologies.* 2018:1–21.

[38] Ting Kai Ming. *Confusion Matrix*. Boston, MA: Springer US 2017.

[39] Cultural India 2019. https://www.culturalindia.net/leaders/index.html.

[40] Cultural India 2019. https://www.culturalindia.net/reformers/index.html.

[41] Rose Stuart, Engel Dave, Cramer Nick, Cowley Wendy. Automatic keyword extraction from individual documents *Text mining: applications and theory.* 2010:1–20.