International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 09 Issue: 04 | Apr 2022 www.irjet.net p-ISSN: 2395-0072

Vikalp - Automatic multiple choice questions generator

Amit Singh[1], Parth Kalbag[2], Divya Kukreja[3], Ritu Kalyani[4]

[1]*Assistant Professor, amit.singh@ves.ac.in, Dept. of AI and Data, Vivekanand Education Society's institute of Technology, Mumbai, Maharashtra, India*

[2-4]*Student, 2018.parth.kalbag@ves.ac.in, 2018.divya.kukreja@ves.ac.in, 2018.ritu.kalyani@ves.ac.in, Dept. of Information Technology, Vivekanand Education Society's institute of Technology, Mumbai, Maharashtra, India*

Abstract - *In any education system, examinations are conducted to judge the caliber of the students. To conduct the examination, teachers need to generate the questions manually which is a very time-consuming process. To reduce time and effort, a system through which multiple choice questions can be automatically generated for user-given text is proposed in this paper. Fill In the Blanks, True or False and Match the Following are the types of multiple-choice questions covered.*

Key Words: **Automatic Multiple choice questions, Natural Language Processing, Mcq's, Distractors, Conceptnet, Wordnet, Sense2Vec.**

1. INTRODUCTION

In this modern world where technology is always evolving and has increased its reach to various sectors of the educational industry, various schools and colleges have adopted the e-learning platforms in which they assess students through various exams like GATE, CAT, College Exams, etc. it has be-come quite a cumbersome task to generate multiple-choice questions for the whole syllabus. MCQ-style questions are a basic instructional tool that may be used for several reasons. These types of questions can affect student learning in addition to serving as an assessment tool. In certain examinations, the only way of ensuring proper assessment of students is by generating good quality MCQs so that they can explore much more than usual and are well-suited to the current e-learning measures.

The paper focuses on generating various types of MCQs like Fill in the Blanks, True or False, and Match the following. Multiple Choice questions are a form of evaluation in which respondents are inquired to choose the most appropriate reply out of the choices from a list. The MCQs are formed from basically two entities: a question and various possible options including the correct answer. The question is formed by determining which type of question can be formed. For e.g., "The number of days in a week are " For the following sentence as the sentence contains a numerical value so the type of question generated can be a numerical one with the correct answer replaced by a " " and suppose considering another example "Still, numbers for server use of Windows (that are comparable to competitors) show one third market share, similar to that for end user use.". If this is the respective question generated, then the type of MCQ the system should generate is either a True or False or a Fill in the blank type of MCQ.

The major challenge in generating any MCQ is the requirement for great distractors, i.e., distractors ought to show up as a conceivable reply to the question indeed to an understudy with great information on the space. At the same time, it should not be a substitute reply or synonym. Besides, a well written MCQ should contain sufficient data to answer the question. For example: "The color of the blood is ." and the corresponding distractors are: Red, Maroon, Dark Red, Crimson Red. As in this example all the distractors have very similar meaning. Hence, the distractors should actually make more sense as in this example: "The color of the blood is ." and the corresponding distractors are: Red, Blue, Green, White. Here, all the distractors have similar context but are clearly distinguishable from each other and are making a good impact.

2. LITERATURE SURVEY

This section discusses research works which proposed the idea for Automatic MCQ generation. Cloze Questions contain questions where questions contain one or more blanks and multiple choices listed to pick an answer is discussed in [1]. It was actually a goal-oriented system, that is, a specific field based on cricket world cup data. Cloze system is divided into three modules: sentence selection, keyword selection and distractor selection. So, the end output gives an English article on cricket World cup and the system generates Cloze questions.

Real time multiple-choice question generation for language testing and English grammar is discussed in [2]. For the application, NLP technique and basic machine learning semi supervised algorithm such as Naive Bayes Classifier and KNearest Neighbors algorithm were used. A real-time system generates only one type i.e Fill in the Blanks type of questions on English grammar and vocabulary from online news articles which takes an HTML file as input and turns it into the quiz session.

The next research paper that was discussed and examined was based on generating MCQ questions using string similarity measure in [3].The research paper mainly focused on keyword selection and generating distractors based on the 1 semantic labels and named entities in text and string similarity measures respectively. So, for selection of sentences and keywords, it is based on the semantic labels and named entities that exist in the sentence. And for the distractor generator concept of similarity measure between the sentences is used. In this proposal, three algorithms of the character based type and five algorithms of the term based type to measure the similarity between two sentences were used.

Similarly, another system is proposed which generates the automatic MCQ based questions using the text extracted from the web in [4]. So mainly web scraping is done, and it summarizes the text using the technique of fireflies preference learning. So, the main part is sentences and distractors. They transformed the sentence into stems and for distractor generation they used the similarity metrics such as hyponyms and hypernyms. Also, the system is used to generate the analogy questions to test the verbal ability of students.

3. PROPOSED METHODOLOGY

The system takes the input text from the user and generates MCQs. The MCQs that will be generated are in the mixed form of Fill in the Blanks, True or False and Match the following.

The system would analyze the text by summarizing the text using the "BERT - State of Art" language model algorithm. Besides, all the watchwords from the summarized content are extracted using POS tagging. At that point the Sentence Mapping is done by extracting the sentences for each keyword. Within the Sentence Mapping, all the sentences from which MCQs are to be shaped are extracted. Since the system isn't fairly kept to one sort of MCQ, thus classification is an important step of the system. All the sentences which were extracted within the Sentence Mapping phase are classified here as which sort of MCQ should be shaped from which sentence. And after that based on the sort of sentence classified, comparing Distractors are created. At last, different techniques are used so that the user would get syntactically correct MCQs. The steps carried out by the system are shown in Fig -1.

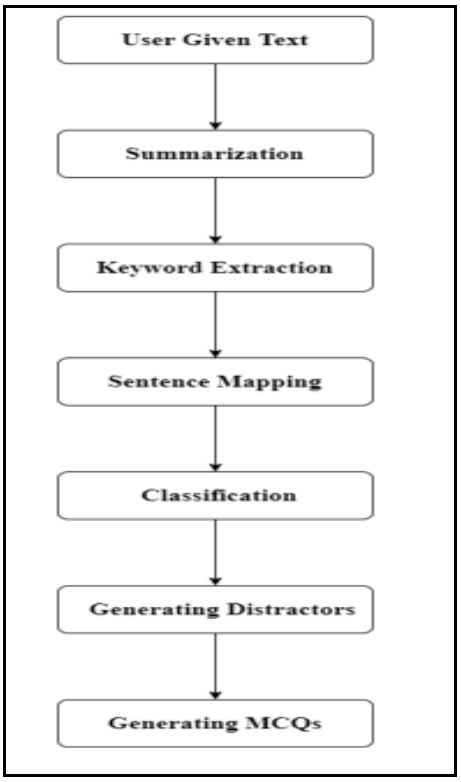

Fig -1: Proposed System

A. Summarization

For the text summarization the system is using a predefined BERT (Bidirectional transformer) Extractive Summarizer model. It is a state of art language model in which minimum length and maximum length is provided to the model to generate summarized text.

B. Keyword Extraction

After finding the relevant information from the generated summarized text the next step is to find all the important keywords which can be treated as an answer to MCQ. Hence, POS tagging is used so that all the relevant nouns, proper nouns, adjectives, and verbs are extracted as keywords. Then they are sorted in descending order according to the preference of the best ones.

C. Keyword Sentence Mapping

Keyword sentence mapping is done by tokenizing the sentences. Here, all the sentences of the corresponding keywords are extracted and tokenized. These are the sentences from which corresponding Multiple choice questions will be generated.

D. Classification

If the sentence contains any numeric value, then those sentences are classified into numeric MCQs category as such type of MCQs would be more accurate, else a random number

is generated between 0 to 1 and if the number is between 0 to 0.75 then Fill in the Blanks MCQ would be generated and if the number is between 0.76 to 1 then True or False type of MCQ would be generated.

E. Generating Distractors

A good distractor is one that gives similar meaning as the answer but is not the answer itself. The goodness and toughness of an MCQ question depend on how close the 2 distractors are to the key. The closer the distractors are to the key, the more difficult the question is. This step is independent for each type of MCQ. Based on the category of the MCQ that is to be formed from the sentence, corresponding types of distractors are generated. Before the generation of distractors, 'word sense' is first applied for the answer and based on that sense the distractors are generated.

4. ALGORITHMS

This section discusses the implementation part of our proposed system. First we show the outcome of coep-package developed to standardise the question generation process. Next, we show the integration of the proposed system with a website through APIs along with the Performance Analysis and Navigation part.

A. Fill in the Blanks

For the text summarization the system is using a predefined BERT (Bidirectional transformer) Extractive Summarizer model. It is a state of art language model in which minimum length and maximum length is provided to the model to generate summarized text.

Algorithm 1 Generating Fill in the Blanks questions

Require: summarized_text

```
questions_set ⇐ {}
pos ⇐ GetPosFromSentences(summarized_text) ▷ GetPostFromSentences() help mark the word corresponding to part of speech based on definition or context
sentences ⇐ TokenizeSentences(summarized_text) ▷ TokenizeSentences() is used to split the paragraph into various sentences
keywordSentenceMapping ⇐ KeywordSentenceMapping(pos, sentences)
for sentence in keywordSentenceMapping do
    for keyword in keywordSentenceMapping do
        word_sense ⇐ GetWordSense(keyword)
        if word_senseisProperNounorNoun then
            Distractors ⇐ Set(Conceptnet)
        else
            Distractors ⇐ Set(wordnet)
        end if
        for sentence in keywordSentenceMapping do
            Replace the keyword with "___" in question
            Add the question in question_set
```

For the Fill in the blank type of MCQs, the distractors are generated by 2 ways. They are as follows:

1) Wordnet: For all the dictionary keywords wordnet works well. It is used to determine the similarity between words. The Wordnet algorithm measures the distance among words and synsets in WordNet's graph structure, such as by counting the number of edges among synsets. If the words are closer, the synsets are similar. It works in 2 stages : i) Generating the Synsets. ii) Generating distractors for all the Synsets.

2) Conceptnet: Conceptnet works well with Nouns and Proper Nouns. Hence for the noun and proper noun type of keywords Conceptnet is used over Wordnet. It is a knowledge graph that connects words and phrases of natural language with labelled edges. Its knowledge is collected from many sources that include expert-created resources, crowdsourcing, and games with a purpose.

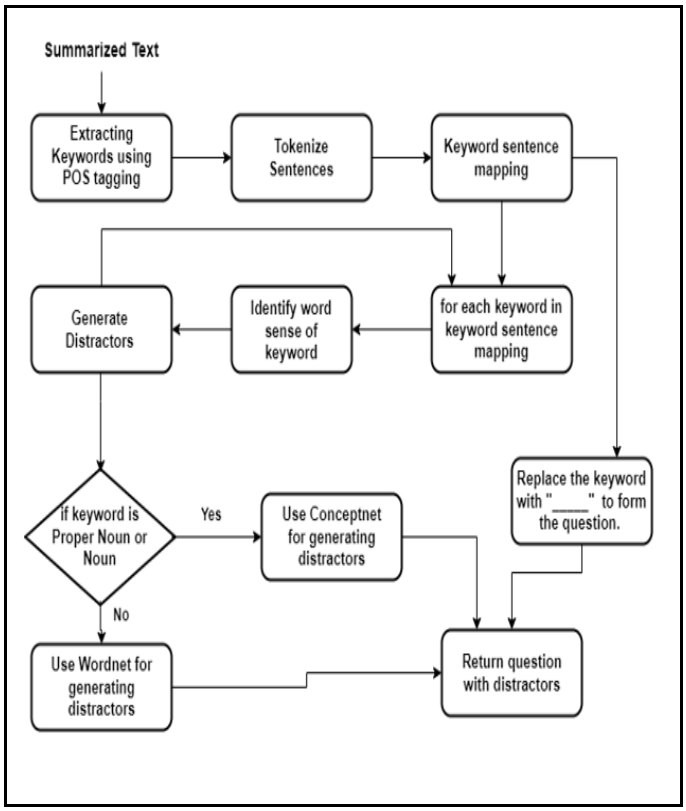

Fig -2: Fill in the blanks

B. Numeric Fill in the Blanks

For generating appropriate distractors for a numeric answer, adding any integer between +5 and -5. Any random number will be selected between this range and it will be added to answer and corresponding three distractors would be generated. Then shuffling these numbers using the FisherYates Shuffle algorithm so that the answer won't be just a particular choice rather it would be a random choice.

Algorithm 2 Generating Numeric Fill in the Blanks questions

Require: summarized_text
$questions\_set \Leftarrow \{\}$
$sentences \Leftarrow TokenizeSentences(summarized\_text)$
▷ TokenizeSentences() is used to split the paragraph into various sentences
for *sentence* in *sentences* **do**
  if *sentence* contains a number **then**
    $question \Leftarrow sentence$
    $answer \Leftarrow$ number
    $choice1 \Leftarrow answer+$ random integer between -5 to 5
    $choice2 \Leftarrow answer+$ random integer between -5 to 5
    $choice3 \Leftarrow answer+$ random integer between -5 to 5
    $choice4 \Leftarrow answer$
    Shuffle choice1, choice2, choice3. choice4 using Fisher-Yates Shuffle algorithm Add $question, answer, choice1, choice2, choice3, choice4$ to the $questions\_set$
  end if

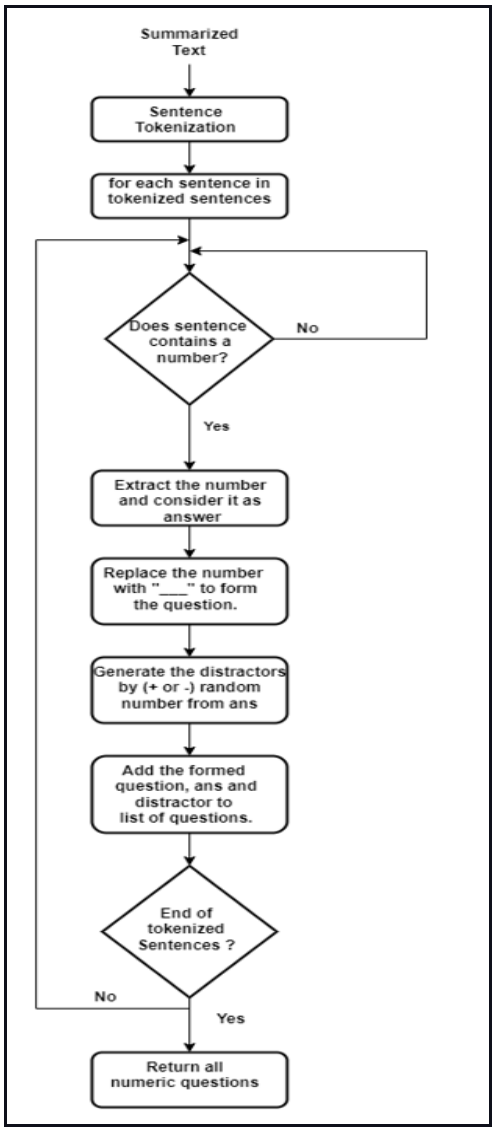

Fig -3: Numeric Fill in the blanks

C. True or False questions

By using this algorithm true or false questions are generated so only options for the question would be True or False so no algorithm for generating the distractors is required.

Algorithm 3 Generating True or False questions

Require: summarized_text
$questions\_set \Leftarrow \{\}$
Initialize the *paraphraser_model*
$noun \Leftarrow GetNounsFromSentences(summarized\_text)$
▷ GetNounsFromSentences() help mark the word corresponding to part of speech based on definition or context
$sentences \Leftarrow TokenizeSentences(summarized\_text)$
▷ TokenizeSentences() is used to split the paragraph into various sentences
$keywordSentenceMapping \Leftarrow KeywordSentenceMapping(pos, sentences)$
for *sentence* in *keywordSentenceMapping* **do**
  Select the answer by randomly choosing between True and False
  if *answer* is True **then**
    $question \Leftarrow sentence$
    $answer \Leftarrow True$
    $questions\_set.add(question, answer)$
  end if
  if *answer* is False **then**
    if *sentence* contains a number **then**
      Replace the number *by adding* 1 *to it*
      $question \Leftarrow sentence$
      $answer \Leftarrow False$
    end if
    if *sentence* contains a auxiliary verb **then**
      Replace the word by it's opposite meaning
      $question \Leftarrow paraphraser\_model(sentence)$
      $answer \Leftarrow False$
    end if
    if *sentence* contains words in [is, with, without, by, be] **then**
      Replace the word by it's opposite meaning
      $question \Leftarrow paraphraser\_model(sentence)$
      $answer \Leftarrow False$
    else
      Replace the word by it's opposite meaning
      $question \Leftarrow paraphraser\_model(sentence)$
      $answer \Leftarrow False$
    end if
    $questions\_set.add(question, answer)$
  end if

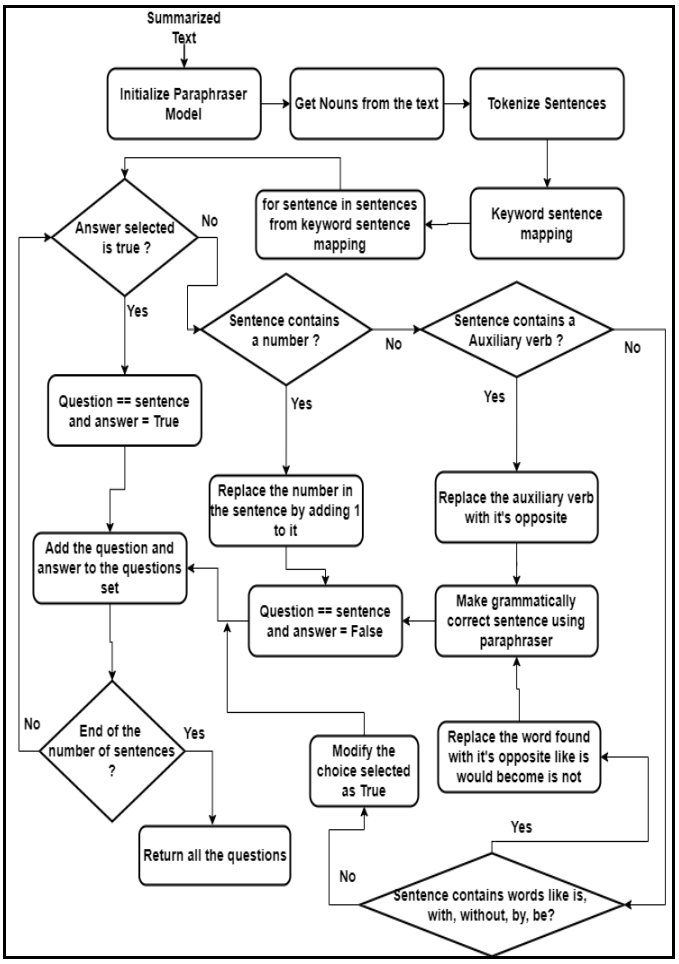

Fig -4: True or False questions

Algorithm 4 Generating Match the Following questions

Require: summarized_text
Initialize matchupset, optionset, keys, filteredkeys
pos ⇐ *GetPosFromText*(*summarized_text*) ▷
GetPosFromText() help mark the word corresponding to part of speech based on definition or context
keywords ⇐ *KeywordExtractor*(*pos*, *summarized_text*)
for *keyword* in *keywords* **do**
  if *keyword is Proper Noun or Noun* **then**
    Add *keyword* to *filtered_keys*
  end if
end for
Mostsimilarkeyword ⇐ *set*(*GenerateMostsimilarkeywordusingSense2Vec*)
for *word* in *Mostsimilarkeyword* **do**
  Add *word* to *keys*
end for
*column*1 ⇐ *filteredkeys*
*column*2 ⇐ *keys*
Add *column*1 and *column*2 into *matchupset*
answer ⇐ *True*
otheroptions ⇐ *column*2
if *answer is True* **then**
  correctoption ⇐ *True*
  Add *correctoption* in *optionset*
  Shuffle the *otheroptions* using Fisher-Yates Shuffle algorithm
end if
if *otheroptions not equals answer* **then**
  Add otheroptions to optionset
end if
Shuffle *optionset* using Fisher-Yates Shuffle algorithm =0

D. Match the Following .

'Sense2Vec' is applied to generate the most sense options for the filtered keywords as column 2 to match the following pattern, ensuring to find the most identical word for the filtered keywords. For Example : " most sense word for forest is woods likewise for paws is back feet." Framing the correct option first, then the other distractors. To begin with, the generated choices are a,b,c,d and used the loop to find the appropriate option, i.e., the correct value for the filtered keyword, and assigned the random choice, ensuring that other keywords are assigned to other choices, and repeated this process for the other filtered keywords. For example : "woods gets the choice d as a random choice then the back feet will be assigned to another choice but not d as it is already assigned". After generating the correct option, other distractors are generated. Finally, the Fisher-Yates Shuffle algorithm is used to shuffle these distractors so that the output is not a definite choice but rather a random choice.

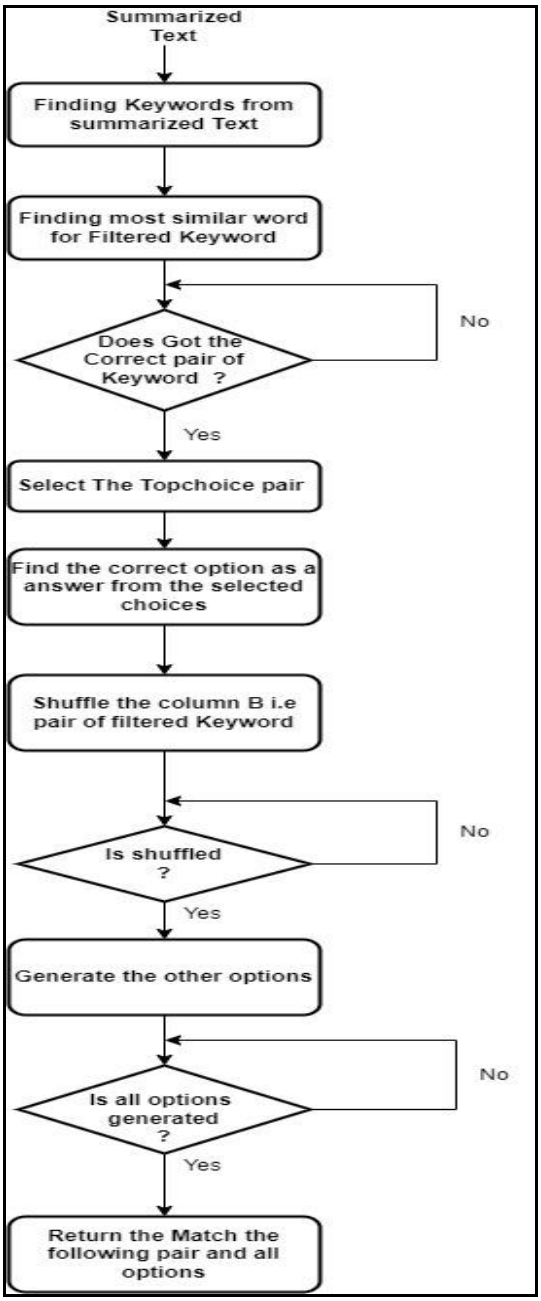

Fig - 5: Match the Following

5. RESULTS

The algorithms are implemented in Python using Flask as a framework and tested on a number of different text files which are taken from a variety of different websites which includes blogs, Wikipedia articles, articles based on some topics and so on. The experimental results show how effective the system is, for extracting the MCQs from the text. Following are some of the resultant MCQs that have been obtained when providing the following text file.

Microsoft Windows, commonly referred to as Windows, is a group of several proprietary graphical operating system families, all of which are developed and marketed by Microsoft. Each family caters to a certain sector of the computing industry. Active Microsoft Windows families include Windows NT and Windows IoT; these may encompass subfamilies, (e.g. Windows Server or Windows Embedded Compact) (Windows CE). Defunct Microsoft Windows families include Windows 9x, Windows Mobile and Windows Phone. Microsoft introduced an operating environment named Windows on November 20, 1985, as a graphical operating system shell for MS-DOS in response to the growing interest in graphical user interfaces (GUIs). Microsoft Windows came to dominate the world's personal computer (PC) market with over 90% market share, overtaking Mac OS, which had been introduced in 1984. Apple came to see Windows as an unfair encroachment on their innovation in GUI development as implemented on products such as the Lisa and Macintosh (eventually settled in court in Microsoft's favor in 1993). On PCs, Windows is strunsill the most popular operating system in all countries. However, in 2014, Microsoft admitted losing the majority of the overall operating system market to Android, because of the massive growth in sales of Android smartphones. In 2014, the number of Windows devices sold was less than 25% that of Android devices sold. This comparison, however, may not be fully relevant, as the two operating systems traditionally target different platforms. Still, numbers for server use of Windows (that are comparable to competitors) show one third market share, similar to that for end user use. As of May 2021, the most recent version of Windows for PCs, tablets and embedded devices is Windows 10, version 21H1. The most recent version for server computers is Windows Server 2022, version 21H2. A specialized version of Windows also runs on the Xbox One and Xbox Series X/S video game consoles

Fig - 6: Input file

Output :

[
{
"question": "The most recent version for server computers is Windows Server ____ version 21H2",
"answer": "2022",
"id": 1,
"choice3": "2022",
"choice4": "2021",
"choice1": "2024",
"choice2": "2023"
},
{
"answer": false,

"question": "Microsoft Windows, commonly referred to as Windows, is a group of multiple proprietary graphics operating system families, all of which are not developed and sold by Microsoft",

"id": 2

},

{

"question": "A specialized version of Windows also runs on the Xbox One and Xbox Series XS video _____ consoles",

"answer": "Game",

"id": 3,

"choice3": "Positivity",

"choice4": "Critical Mass",

"choice1": "Game",

"choice2": "Acting"

},

{

"question": "1) server → Just The Number 2) number → Whole Game 3) game → Desktop Computer 4) computer → Same Server ",

"answer": "1 → d 2 → a 3 → b 4 → c",

"id": 4,

"choice3": "1 → b 2 → c 3 → d 4 → a",

"choice4": "1 → d 2 → a 3 → b 4 → c",

"choice1": "1 → a 2 → d 3 → b 4 → c",

"choice2": "1 → d 2 → a 3 → c 4 → b"

},

{

"answer": true,

"question": "Still numbers for server use of Windows that are comparable to competitors show one third market share similar to that for end user use",

"id": 5

}

]

So, the system is evaluated manually where evaluators check the system on various text files to ensure the syntactic and semantic correctness of the questions and also the quality of distractors. It is observed that the number of MCQs that are formed from any text are variable. This is directly proportional to the summarization of the text. The exactness of MQCs generator is found to be really good (nearly all questions are of great level) since distractors are chosen by applying various techniques. Hence, the framework effectively creates the programmed different types of MCQs.

6. CONCLUSION

The system is tested by giving various inputs on different domains in the form of text and the system is working well by generating good quality MCQs and generates the syntactic and semantic correct questions along with the good quality of distractors. The problem of manually creating the MCQs is solved, and the system is helpful for teachers for generating the MCQ type questions.

Following are the future scopes of the system:

1) To build the same system for other languages such as Hindi, Urdu, South Indian Languages, etc.

2) Various different types of multiple-choice questions like answers containing images, a multiple choice question containing question and multiple correct answers could be added.

REFERENCES

[1] Annamaneni Narendra, Manish Agarwal and Rakshit shah (2013) "Automatic Cloze-Questions Generation". In Proceedings of Recent Advances in Natural Language Processing, pp. 511–515.

[2] Ayako Hoshino, Hiroshi Nakagawa (2005). "A real-time multiple-choice question generation for language testing a preliminary study". In Proceedings of the 2nd Workshop on Building Educational Applications Using NLP, pp. 17–20.

[3] [3] Ibrahim Eldesoky Fattoh, 2014. "Automatic Multiple Choice Question Generation System for Semantic Attributes Using String Similarity Measures". In Computer Engineering and Intelligent Systems www.iiste.org ISSN 2222-1719 (Paper) ISSN 2222-2863 (Online) Vol.5, No.8, pp. 66- 73.

[4] [4] Santhanavijayan, A., Balasundaram, S.R., Hari Narayanan, S., Vinod Kumar, S., and Vignesh Prasad, V., 2017. "Automatic generation of multiple choice questions for e-assessment". In Int. J. Signal and Imaging Systems Engineering, Vol. 10, Nos. 1/2, pp. 54-62.

[5] [5] Ming Liu, Rafael Calvo, A., and Vasile Rus, 2012. "G-Asks: An Intelligent Automatic Question Generation System for Academic Writing Support". In Dialogue and Discourse 3(2), pp. 101–124.

[6] [6] Shivank Pandey, Rajeswari, K.C., 2013. "Automatic Question Generation Using Software Agents for Technical Institutions". In International Journal of Advanced Computer Research (ISSN (print): 2249-7277

ISSN (online): 2277-7970) Volume-3 Number-4 Issue-13, pp. 307-311

[7] [7] Manish Agarwal, Rakshit Shah and Prashanth Mannem, 2011. "Automatic Question Generation using Discourse Cues". In Proceedings of the Sixth Workshop on Innovative Use of NLP for Building Educational Applications, pp. 1–9.

[8] [8] Arjun Singh Bhatia, Manas Kirti, and Sujan Kumar Saha,2013. "Automatic Generation of Multiple Choice Questions Using Wikipedia". In P. Maji et al. (Eds.): PReMI 2013, LNCS 8251, pp. 733–738.