THE INTERNATIONAL JOURNAL OF SCIENCE & TECHNOLEDGE

Automatic Question Generator

Ravina Dhami
Business Analyst, Tata Consultancy Services
Past -Student, Department of Information Technology, Dwarkadas J. Sanghvi College of Engineering, Mumbai, India
Dr. Abhijit R. Joshi
Professor, Department of Information Technology, Dwarkadas J. Sanghvi College of Engineering, Mumbai, India
Vinaya Sawant
Professor, Department of Information Technology, Dwarkadas J. Sanghvi College of Engineering, Mumbai, India
Neha Mendjoge
Professor, Department of Information Technology, Dwarkadas J. Sanghvi College of Engineering, Mumbai, India

Abstract:
The task of generating questions for a test is tedious and time consuming process for teachers. The project, works toward automating that process. The addition of Automatic Generator to assessment applications decreases the time spent on constructing examination papers [2]. It is designed to create a system for question generation (QG) that takes as input an article of text (e.g., a web page, encyclopedia or textbook material), and generates a list of factual questions of what ,who type.
These questions are then presented to the students in form of a test paper. A score card is also generated after the student takes the test based on their performance.
This efficient question generation software makes use of the linguistic areas in NLP (Natural Language Processing) is useful and has potential for educational applications. Also, work in this area contributes to the literature on artificial intelligence in education and educational applications of NLP.

Keywords: *Natural Language Processing (NLP), Artificial Intelligence (AI), User Interface Design*

1. Problem Definition

Generating questions can be a time consuming and effortful process. The project, works toward automating that process. In particular, focus is on the problem of automatically generating 'WH' (which, what, where, when, who) questions from individual texts [3].
This project aims to generate one line questions for which the source of answer is the information in the text provided by the user (specifically information that is focused at the sentence or paragraph level rather than spread across the whole document.)

2. Introduction

The Automatic Question Generator is a software which generates a list of questions from the paragraph inputted by the teacher. The teacher can review the questions and the confirmed questions are added to the test paper. The answer given by the student is compared with the actual answer generated by the QG software and a score card is displayed to the user at the end after he finishes answering all the questions.
Artificial intelligence (AI) is the intelligence of machines and the branch of computer science that aims to create it. This software in itself is an intelligent application which can perform the tasks of generating questions automatically without any human interference. Thus, it comes under the umbrella of Artificial intelligence.
Natural language processing (NLP) is a field of computer science, artificial intelligence, and linguistics concerned with the interactions between computers and human (natural) languages. The project enables the software to derive meaning from human or natural language input. This meaning is then used to derive questions for the user.

2.1. Benefits of the System

1. The Auto question generator hastens the process of generating questions. Therefore question papers can be generated easily and very quickly by the teacher. The teacher only has to input the text document from which she wants to generate the desired questions.
2. It facilitates questions to be made from outside textbook material. The text document selected by the teacher need not be restricted to material from the text book. The teacher can input any text document and expect questions to be made from it [1].

3. Student can take answer questions and get a better understanding of the passage. The students can check their own understanding of a particular topic by taking the quiz. The correct answers for each question are also shown to the student. They can compare their answers with the correct answers and evaluate themselves better.
4. Facilitates score card generation. At the end of the test a score card is generated. This score card can be given to the parents to see the development of their child.

3. Architecture Diagram

The Automated Question Generator system is constituted of independent modules which perform separate tasks, and together they form a working system. The system provides users with the ability to automate the process of question generation from any piece of text.

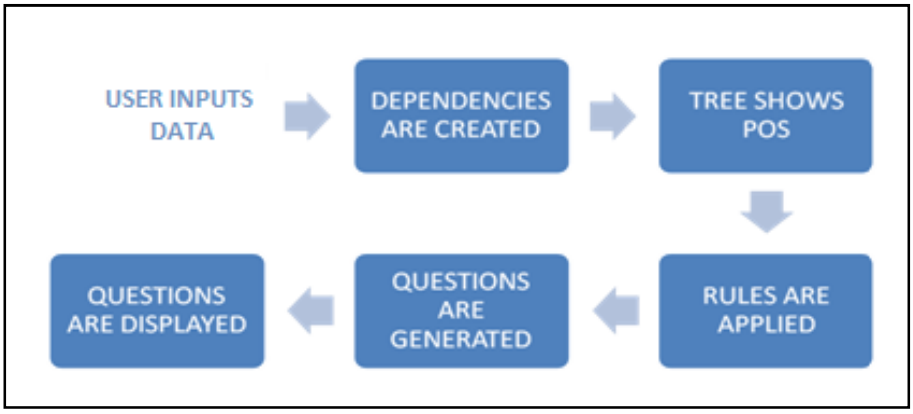

Figure 1: Architecture

3.1. Description of the Architecture

1. User enters data: The user enters text in the text box.
2. The Stanford Parser creates dependencies between words: A set of dependencies between the words of a sentence are generated by the parser.
3. A tree showing part of speech is shown ('POS'): The parser identifies the word class for each word in the sentence and displays the related grammatical label (e.g. 'vp' – verb phrase, 'np' – noun phrase). The 'POS' tagging is displayed in form of a tree.
4. The rules are applied: A set of rules which are written to manipulate the tree structure are applied on the tree generated for each sentence.
5. Questions generated: The system creates questions using the rules.
6. Questions displayed: The questions generated by the system is shown to the user.

3.2. Rules

The core logic of the system is contained in the set of rules for converting the parsed sentence tree structure into question form, by substituting the matching sub trees with a question word. Example of the rules are:

1. Rule 1:
   Noun + Verb + Predicate => Who/What + Verb + Predicate?
   e.g. The mouse ate the cheese => Who/What ate the cheese?
2. Rule 2:
   Noun + Verb + Preposition => Where + Did + Noun + Verb?
   e.g. The boy went outside => Where did the boy go?
3. Rule 3:
   Noun + Verb + Adjective => Who + Verb + Adjective?
   e.g. The girl is smart => Who is smart?

Such rules combine together to form a very powerful system for generating questions from input sentences. We augmented this core NLP module with a user interface for taking paragraphs as input and outputting all possible questions from sentences of that paragraph.

4. Implementation Details

For designing the system, we will follow a modular approach to make the task easier to code and handle the task effectively.

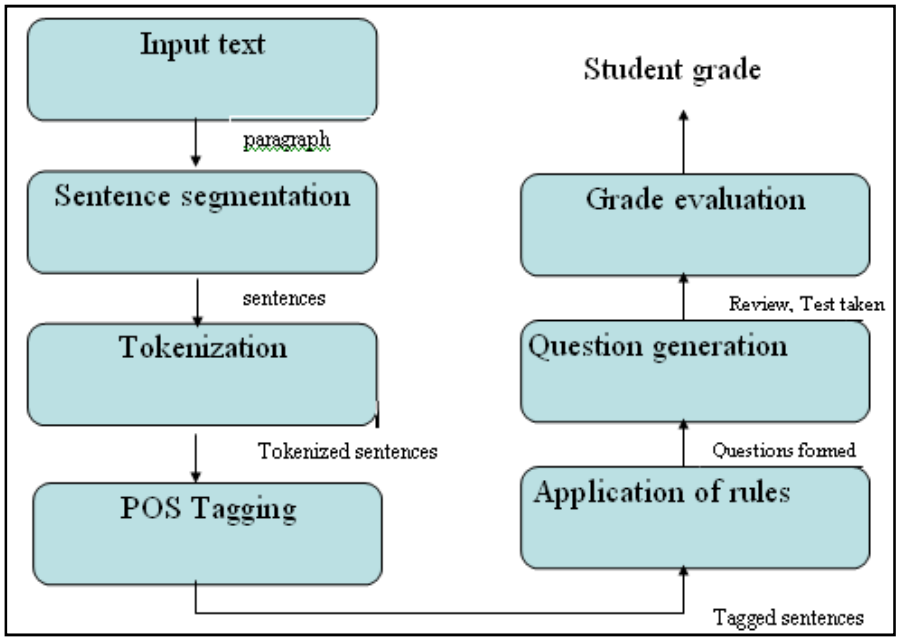

Figure 2: Implementation Diagram

4.1. Module Description

- Module 1: Input acceptance
  - The user provides the text from which the questions are intended to be generated.
  - The input is in the English language.
  - Input could be a line or even multiple paragraphs
- Module 2: Segmentation
  - The raw text is initially taken in by the system which passes the text to its segmenter module.
  - The sentence segmenter then splits these raw texts into separate individual sentences
- Module 3: Tokenization
  - These split segments are then passed to the systems tokenizer module.
  - The tokenizer divides the segments into individual words.
  - E.g. A sentence "the movie was awesome." when given to the tokenizer becomes [the, movie, was, awesome].
  - Each word here is known as a token.
- Module 4: Part of speech tagging
  - These tokens are then passed to the 'Part Of Speech' tagger.
  - Here each token is tagged with an appropriate part of speech.
  - This is one of the most important modules of the system and its accuracy is vital for the next module and also the entire system.
  - E.g. [the PN, movie NN, was AR, awesome AJ].
- Module 5: Application of rules
  - The pre-defined rules of grammar which have been fed into the code are applied to the POS tagged sentences in order to form questions.
  - The rules check for the certain common patterns(like SUBJ-VERB-OBJ etc) found to be commonly occurring in sentences
  - Depending on the type of rule applicable corresponding questions of the WH-type are formed.
- Module 6: Question generation
  - The questions which are formed by the application of the predefined rules may not be precisely accurate (on certain occasions, they could turn out to be grammatically incorrect or sometimes they could seem ambiguous.)
  - Also, not all of the questions formed could be worthy of being put into a particular test (like: if the sentence reads:" She went there." In such a case," Where did she go?" would not be a question worth putting in the test.)Hence, the formed questions are reviewed by the paper setter for its worth.
- Module 7: Grade evaluation
  - After the test questions are reviewed by the professor, the entire test paper is provided to the desired students along with a unique system generated password and id.
  - The test taker attempts the test. The answers of which are submitted and the system checks the submitted answers with those which are the expected (correct) answers and evaluates each test taker individually-by +2 for every correct answer and -2 for each incorrect one.
  - The final grade is then assigned based on the overall performance of the test taker.

5. Algorithm

1. Input paragraph
2. Break into sentence using “.”
3. Break into word using “ ”
4. Load the parser
5. Find ‘Part Of Speech’ for each word
6. If (“VP ” preceded/followed by “NP”)
7. Then -> make question ("Who/What " + (subtree) + "?")
8. Answer= “NP”

For e.g.;

5.1. Sentence 1:
The mouse was killed by the cat.
Tree structure:
(ROOT
(S
(NP (DT The)
(NN mouse)
(VP (VBD was)
(VP (VBN killed)
(PP (IN by)
(NP (DT the)
(NN cat)

NP (The) NN (mouse) is followed by the VP (VBD was) (VBN killed) (IN by) NP (DT the) (NN cat)
Hence,
Question: Who/What + VBD + VBN + IN + DT + NN +?
Who/What was killed by the cat?

Answer= NP=mouse

5.2. Sentence 2:
Once upon a time there lived a lion in a forest
Tree structure:
(ROOT
(SBAR (RB Once) (IN upon)
(S
(NP (DT a) (NN time))
(ADVP (RB there))
(VP (VBD lived)
(NP (DT a) (NN lion))
(0PP (IN in)
(NP (DT a) (NN forest.)))))))

SBAR (RB Once) (IN upon) (DTa) (NN time) ADVP (RBthere) **VPis followed by the**(VBDlived)**NP** (DTa)(NN lion) 0PP (IN in) NP (DT a) (NN forest)

Hence,
Question: Who/What + VBD + IN + DT + NN +?
Who lived in aforest?

Answer= NP=lion

6. Interface Design

6.1. *Textual user input:* The user is provided with a text area in order to place the text from which the questions are to be generated.

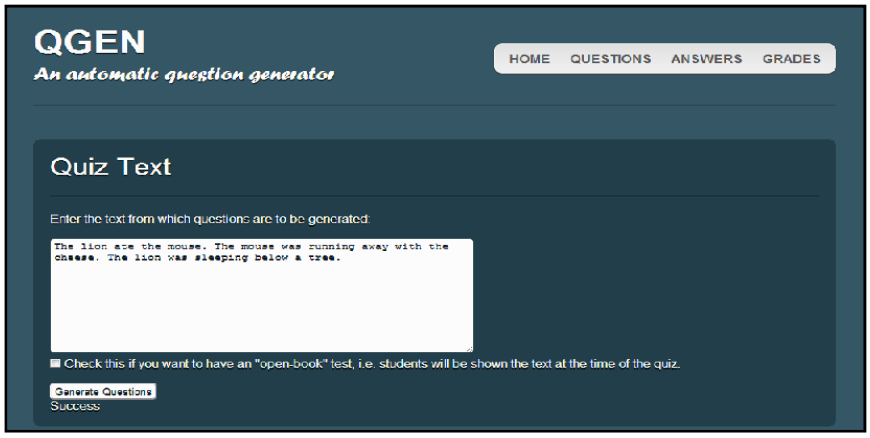

Figure 3: Input Text

6.2. *Questions review:* The generated questions are displayed to the user. The user is also the functionality to review and confirm the questions which he wishes to use.

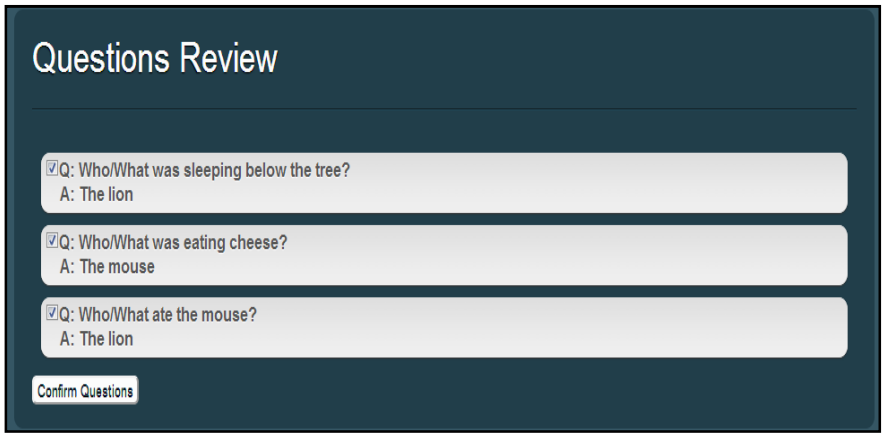

Figure 4: Generated Questions

7. Experiments

We tested the efficiency and usefulness of our working system by deploying it to elementary school teachers who used the QGen system to generate simple in-class quizzes and assignments for their students. The overall response was very positive and the teachers were happy with reduction in their workload, while at the same time maintaining the quality of their assignments.

Generated Quiz: The quiz displayed details of such as Quiz Name, teacher's name, Test documents and questions. The quiz also showed the correct answer for each question with the option to only show questions if needed.

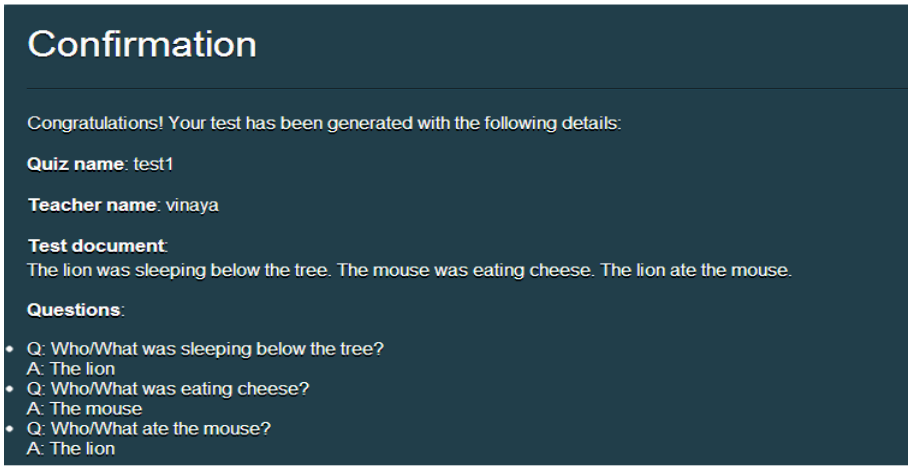

Figure 5: Generated Quiz

8. Conclusion

The system gives teachers the freedom to generate questions from any text book or outside textbook material. This system also gives an option for students to give a test which will be evaluated and also giving a score card at the end. Thus the system we have created is a simple, quick and efficient manner to generate questions and take a test for improving the understanding of a subject.

9. Acknowledgement

We would like to thank respected Principal, Dr. Hari Vasudevan, D. J. Sanghvi College of Engineering for giving us facilities and guidance. We would also like to thank Shri Vile Parle Kelavani Mandal for encouraging us in such co- curricular activities.

10. References

i. Li-Chun Sung, Yi-Chien Lin, Meng Chang Chen.The Design of Automatic Quiz Generation for Ubiquitous English ELearning.
ii. Iztiar Albade, Maddalen Lopez de Lacalle, Montse Maritxalar, Edurne Martinez, Larraitz Uria (2006). ArikIturri:An Automatic Question Generator Based on Corpora and NLP Techniques.
iii. Vasile Rus, Zhiqiang Cai, Arthur C. Graesser (2007). Experiments on Generating Questions About Facts.