ISSN No:-2456-2165

Automatic Question Paper Generation, according to Bloom’s Taxonomy, by generating questions from text using Natural Language Processing

Shivali Joshi[*], Parin Shah[*], Sahil Shah[*]
[*]Bachelor of Technology, Department of Information Technology, K.J. Somaiya College of Engineering, Mumbai, India.

Abstract:- The ongoing research on "Natural Language Processing and its applications in the educational domain”, has witnessed various approaches for question generation from paragraphs. Despite the existence of numerous techniques for the automatic generation of questions, only a few have been implemented in real classroom settings. This research paper reviews existing methods and presents an AQGS (Automatic Question Generation System) that uses Natural Language Processing Libraries like NLTK and Spacy to suggest questions from a passage provided as an input. The Question Paper is generated by randomly selecting questions for a specific level of Bloom’s Taxonomy. We conclude by determining the efficacy of the AQGS using performance measures like accuracy, precision, and recall.

Keywords:- *Question Generation, Bloom’s Taxonomy, Natural Language Processing (NLP), Natural Language Toolkit (NLTK), Spacy, POS Tagging, Named Entity Recognizer (NER).*

I. INTRODUCTION

Researchers belonging to various disciplines have started working on AQGS for educational purposes. Examinations, being one of the crucial parts of education, are conducted to test the caliber of the examinees. Examiners are majorly dependent on themselves for making test papers. Setting a question paper with the least number of repeated questions is the most time-consuming task. Going through pages to find new questions that are appropriate for the exam is cumbersome. Additionally, the capabilities of a candidate can be evaluated only with a question paper that consists of the right proportion of theoretical and application-based questions.

The proposed system aims to find a solution to both of the above-mentioned problems. The feature of question extraction from paragraphs provides professors with ample new questions within few seconds saving a considerable amount of time. AQGS facilitates the examiner with a huge repository of questions in order to avoid question repetition in the examination. Secondly, a question paper set with respect to Bloom’s Taxonomy ensures goal-based learning and can efficiently evaluate the knowledge of a student. For a specific level of Bloom’s Taxonomy, the random selection of questions from the large question bank eliminates any possibility of human bias and thus, making every test paper unpredictable. Thus, the system proves to be beneficial for the online school examinations, especially during times of the pandemic, for the creation of new questions whose answers are not directly available on the internet; thus, reducing student malpractices. These questions generated can be used by teachers to set test papers. Students can leverage it for self-evaluation to understand their grasp on a particular topic. This automation reduces costs, labor, and rules out the occurrence of human errors, arming the user with a fast and easy-to-use question-generating tool at their fingertips.

Generally, the three major components of Question Generation are input pre-processing, sentence selection, and question formation. The input text is filtered by removing unnecessary words and punctuations that do not contribute to the meaning of the sentence. The sentences or phrases from which questions can be formed are segregated from the remaining text. These are mapped to the type of question (what, where, when, etc.) that can be formulated with the selected sentence, followed by the final step of framing a grammatically sound question.

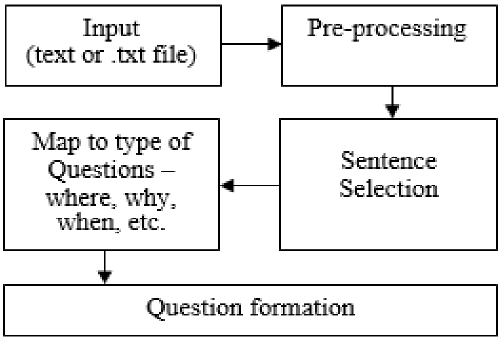

Fig. 1. Generic Approach of Question Generation

II. LITERATURE SURVEY

Automatic Multiple Choice Question Generation from Text: A Survey [1], reviews 86 articles and summarizes existing methodologies for generating MCQs from a text. After analyzing these methods, the paper presents a detailed flowchart of a question generating system that consists of six phases- pre-processing, sentence selection, key selection, question formation, distractor generation, and post-processing. A comprehensive discussion of various techniques for implementation of each phase along with recent trends and challenges for MCQ generation is presented in the paper.

Automatic Cloze question generation or CQG [2], an article in English is provided as an input from which the system generates a list of cloze questions- a sentence comprising of one or multiple blanks. Sentence Selection, Keyword selection (potential blank selection), and Distractor selection (selecting alternate answers for the blank) are major components of CQG. To begin with, potential sentences are selected followed by keyword selection on the basis of NER and finally, domain-specific distractors are generated based on the knowledge base provided to the model. Manual evaluation of the system is done for each sentence, keyword and distractor selection.

Automatic Question Generation using Discourse Cues [3], the system can be viewed as content selection and question formation. The emphasis is on recognition of discourse markers and discerning the important discourse relations like casual, temporal, contrast, result, etc. After identification of relevant text for framing a question, (seven) discourse connectives are specified for finding the type of Wh-question (like why, where, which and when) and syntax transformations are performed. Semantic and syntactic evaluation of the system is done.

Semantic Based Automatic Question Generation using Artificial Immunity [4], both, SRL (Semantic Role Labelling) and NER (Named Entity Recognizer) are for the conversion of input text into a semantic pattern. An Artificial immune system that uses feature extraction, learning, storage and associative retrieval to classify patterns according to question type like who, when, where, why, and how. The input sentence is mapped into a pattern through SRL (which is used for feature extraction) and NER, and depending on the question type, sentence pattern is realized. 170 sentences were mapped into 250 patterns that were used for training and testing. For evaluation, Recall, Precision and F-measurement were used. The proposed model has a classification accuracy of over 95%, and 87% in generating the new question patterns.

A Combined Approach Using Semantic Role Labelling and Word Sense Disambiguation for Question Generation and Answer Extraction [5], the article introduces a joint model of question formation and answer identification using Natural Language Processing. The Question Generation part makes use of SRL and WSD (Word Sense Disambiguation) techniques while the Answer Extraction part uses NER along with SRL. Simple sentences are provided as an input to the model. The questions and answer pairs obtained for a set of sentences were analyzed to evaluate the accuracy of each question generation and answer extraction.

Automatic Question Generation from Given Paragraph [6], the paper present a web application in which simple and complex Wh-questions are generated from a paragraph. It is mapped to a set of predefined rules depending on the verb, subject, object, and prepositions that a sentence comprises of. POS Tagger is used to label the part of speech for each word, Dependency parser analyses the grammatical structure of the sentence and the relation between the words, and Support Vector Machine (SVM) is the algorithm used for performing classification. Human evaluation is done to check the semantic and syntactic accuracy of the output generated.

Similarities in words Using Different POS Taggers [7], presents a comparison of four different POS Taggers (NLTK, Freeling, NLP Tagger and Cognitive POS Tagger) to identify the proper tag for a given text. The paper analyses the results of each tagger for Wh-questions like how, what, which, where, who and why. Out of 350 wh-questions, 154 had contrasting tags by these four tools and the results can are summarized by stating that NLTK outperforms other taggers by labeling the word with the right part of speech. We use the NLTK tagger for POS tagging and other NLTK algorithms like Lancaster Stemmer and WordNet Lemmatizer that are discussed in the following sections.

III. SPACY

Spacy is one of the on-the-go-libraries of NLP enthusiasts which is specifically built to process and help us understand large volumes of text. The Spacy framework which is written in Cython is a quite fast library that supports multiple languages like English, Spanish, French, German, Dutch, Italian, Greek, etc. It comprises various models about trained vectors, vocabularies, syntaxes, and entities. These models are to be loaded based on the requirements. For the “english-core-web” package, the default package is 'en_core_web_sm' where ‘sm’ stands for small. Spacy has three models in the English language- small, medium, and large. As the name suggests, these models vary in size and accuracy. However, in the proposed system load the large package which is used for entity recognition for better accuracy and precision.

```
>>>import spacy
>>>spacy.load (“en_core_web_lg”)
```

IV. NATURAL LANGUAGE TOOLKIT

NLTK- the Natural Language Processing Toolkit, is the mother of all NLP libraries. It provides lexical resources, over 50 corpora and a set of libraries for tokenization, stemming, classification, tagging, and semantic reasoning along with many others. It is a platform to develop programs that require natural language processing in Python language. NLTK is a crucial component of the AQGS system presented in this paper.

A. Lancaster Stemmer

Stemming in Natural Language Processing refers to the process of reducing words to their stem or root word. This word stem may not be the same word as a dictionary-based root word. But, it is just a smaller or equal form of the word. For instance, ‘retrieves’, ‘retrieval’, ‘retrieved’ reduce to the root ‘retrieve’. Porter’s Stemmer, Lovins Stemmer, Dawson

Stemmer, Xerox Stemmer, Snowball Stemmer are some of the many stemming algorithms developed till date. The proposed approach uses the Lancaster Stemmer algorithm provided by NLTK as it is dynamic, fast and aggressive as compared to others. It uses an iterative algorithm and saves rules externally, meaning, custom rules be added. We use the Lancaster Stemmer during question generation to stem the verb from a sentence and generate a skeleton of the question after POS tagging of the sentence from which the question is to be formed. Sometimes, it transforms words into strange roots and therefore, proper care of the spelling errors in the sentences needs to be taken while using this algorithm. In the example given below, ‘troubl’ isn’t a stem word according to the dictionary but the stemmer reduces words similar to trouble into the stem- ‘troubl.

```
>>>Lancaster = LancasterStemmer()
>>>print(“cat : ”, Lancaster.stem(“cat”))
>>>print(“trouble : ”, Lancaster.stem(“trouble”))
>>>print(“troubling : ”, Lancaster.stem(“troubling”))
```

```
Output:
cat : cat
trouble : troubl
troubling : troubl
```

B. WordNet Lemmatizer

Lemmatization groups together different forms of the word for analyzing them as a single entity in order to identify the dictionary root word, preferably called ‘lemma’. Lemmatization is similar to stemming to some extent. The key difference is that Lemmatization aims to get rid of the inflectional endings that might occur while stemming. The output, after the process of lemmatization, has some context to it and importantly, the word holds a meaning unlike stemming.

WordNet is a large, free and publicly available lexical database of the English language. It can be viewed as a thesaurus where similar words are grouped into sets (synsets), each one individually expressing a distinct concept. The main aim is to develop a structured semantic relationship between words. NLTK provides an interface to access this dictionary-WordNet corpus reader. After download and installation, an instance of WordNetLemmatizer() is needed to lemmatize words, similar to the stemming example.

```
>>>lemmatizer = WordNetLemmatizer()
>>>print(“trouble : ”, lemmatizer.lemmatize(“trouble”))
>>>print(“rocks : ”, lemmatizer.lemmatize(“rocks”))
>>>print(“corpora : ”, lemmatizer.lemmatize(“corpora”))
```

```
Output:
trouble : trouble
rocks : rock
corpora : corpus
```

C. Part-of-Speech Tagging

POS Tagging is the technique of labeling the appropriate Part-of-Speech to a token (word) in a text corpus. POS tagger is one of the most powerful aspects of the Natural Language Toolkit (NLTK). It first reads the sentence and then assigns parts of speech (such as Noun, verb, adjective, etc) to each token. Every part of speech is represented with a tag. For the question generation process, we focus majorly on NN, NNS, NNP, NNPS, VB, VBN, VBD, PRP, PRPP.

Example:

```
>>>import nltk
>>>print(nltk.pos_tag (nltk.word_tokenize("Hey, how are you doing?")))
```

```
Output:
 [('Hey', 'NNP'), (',', ','), ('how', 'WRB'), ('are', 'VBP'), ('you', 'PRP'), ('doing', 'VBG'), ('?', '.')]
```

TABLE 1. MAJOR POS TAGS LIST

Tag	Part of Speech
NN	noun, singular ‘chair’
NNS	noun, plural ‘chair’
NNP	proper noun, singular ‘Jones’
NNPS	proper noun, plural ‘Indians’
VB	verb, base form ‘take’
VBN	verb, past participle ‘taken’
VBD	verb, past tense ‘took’
VBP	verb, sing. present, non-3d ‘are’
VBG	verb, gerund/present participle ‘doing’
PRP	PRP personal pronoun ‘I, he, she’
WRB	wh-abverb ‘where’, ‘how’, ‘when’

V. BLOOM’S TAXONOMY

There are three levels of human cognition: thinking, learning and understanding. Bloom’s taxonomy is a classification system to define and distinguish the levels of knowledge acquisition and thus, acts as a guide to develop assessments and questioning strategies. The system put forward in this paper uses these categories to evaluate students in a precise way. Every question is categorized, as per 6 levels defined by revised Bloom’s Taxonomy [8], based on the example question cues and stems shown below in Table 2.

TABLE 2. BLOOM'S TAXONOMY (FROM LOWER TO HIGHER ORDER THINKING SKILLS), QUESTION CUES AND QUESTION STEMS

Category	**Question Cues**	**Question Stems**
Knowledge (Factual recall, remembrance of major dates, events, etc.)	define, who, when, where, quote, name, identify, label	Who wrote...?, when did…?, who said…?, where did…?, who are the…?
Comprehension (understanding, compare, interpret)	differentiate, distinguish, describe, summarize, discuss, predict, list, contrast	What is the difference between…?, What is the summary of…, what is the predicted outcome of…, what is the sequence of…
Application (visualize application in real life, solve problems using methods or theories)	Demonstrate, calculate, solve, illustrate, examine, test, classify	How to solve…, what is the classification of…?, how to examine…?, Demonstrate the process of….
Analysis (identification, pattern, recognition, analysis)	Analyse, explain, classify, connect, infer, probe	What proves that…?, how is this similar/different to…?, what is the problem with…?, why did …precede/follow…?
Evaluation (choose, verify evidence, recognize, access theories)	Access, rank, grade, support, conclude, select, measure, convince, support	How effective is…?, what would you choose…? How would you rank/grade…?, what does the argument support…?
Creativity (independent creative thinking, shift perspective, innovate)	Design, innovate, hypothesise, conceive, craft, compose, invent	Can you image how…?, how would you invent…?, hwo would you respond, what design would you make for…?

VI. PROPOSED APPROACH

The main components of the proposed system are sentence selection, question formation, answer extraction, identification of Bloom's Taxonomy, and question paper generation. The details of the system flowchart depicted in Fig. 2 are discussed in the sections below:

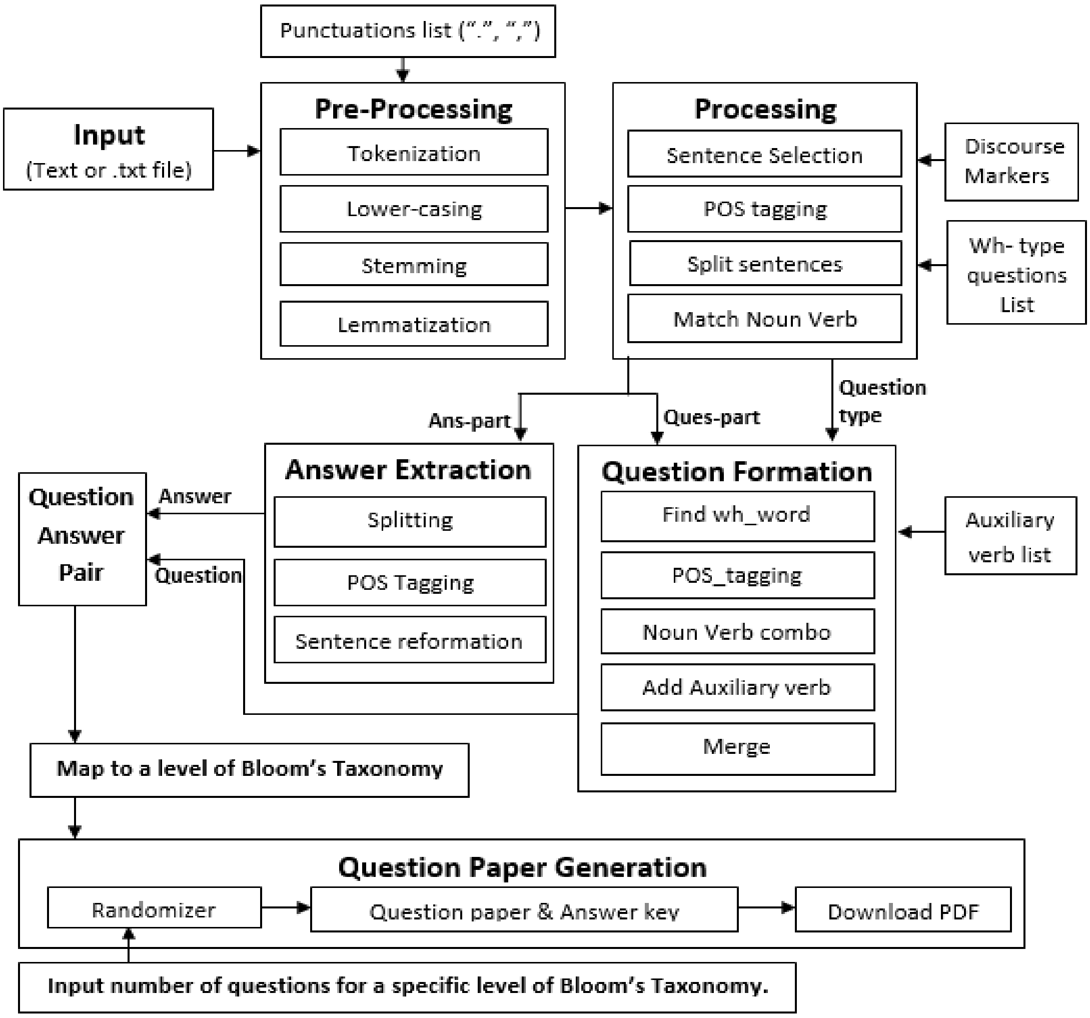

Fig. 2. Flowchart of Proposed system

A. Question formation and Question-Answer pair Generation

A piece of text consisting of one or more paragraphs needs to be given as input by the person in charge of setting the test paper. The input provided is split into small units called tokens. For tokenization, we use the NLTK Punkt Sentence Tokenizer which will segment the running text into meaningful sentences. Other preliminary steps of input data pre-processing include lowercasing, stemming, and lemmatization. The Lancaster Stemmer algorithm used for stemming and WordNet Lemmatizer is used for lemmatization.

The selection of potential sentences is done on the basis of discourse cues. A list of seven discourse markers is hardcoded and the sentences comprising of these are filtered from the rest. Once the discourse marker in a sentence is identified, the mapping to the type of question that can be formed is done. The sentence is split into two parts- the question-part and the answer-part after the process of POS tagging.

TABLE 3. SENTENCE SELECTION USING DISCOURSE MARKER

Discourse connective	**Sense**	**Q-type**	**Target argument**
because	causal	why	arg1
Since	temporal, causal	Why, when	arg1
when	causal + temporal, temporal, conditional	when	arg1
although	contrast, concession	yes/ no	arg1
as a result	result	why	arg2
for example	instantiation	give an example where	arg1
for instance	instantiation	give an instance where	arg1

Sentences are traversed to find auxiliary verbs. If the auxiliary verb exists, we tokenize the part of the sentence that will form the question, separate the auxiliary verb from the main verb and form a question using the auxiliary verb. The sentence string is again combined as it is with a single change i.e. the auxiliary verb is replaced by a wh-question at the start and a question mark in the end. The wh-word is found out by traversing the specific tagged words in sentences like date/time (when), names (who), etc. The best-suited wh-word for the sentence is put at the start of the sentence. For the contrary case, when the sentence does not comprise of any auxiliary verb, the only possibility is the use of some form of ‘do’. There can be the following combinations of nouns and verbs:

NN/NNP and VBZ =>Does
NNS/NNPS(plural) and VBP => Do
NN/NNP and VBN/VBD => Did
NNS/NNPS(plural) and VBN => Did

Therefore, we tokenize and find a relevant POS tag for all the words. The stem is determined from the verb by stemming. We check the first index of the list obtained if it is a noun, pronoun, singular or plural first person, etc. and an appropriate word (do, does, or did) is designated in place of the word at index 0. The whole sentence is combined with a question mark at the end. To find the answer pair for a particular question, a similar approach is followed in which the answer-part identified for a particular sentence undergoes POS tagging and sentence re-transformation if necessary. For instance, for the sentence- ‘He went bankrupt because he took many loans’, the question will be ‘why he went bankrupt?’ and the answer will be the same as the original sentence. Thus, no transformation is needed. However for yes/no questions formed on the basis of discourse marker ‘although’, sentence transformation is required. Both the question and answer sentences are converted into a syntax tree that is most commonly used by NLP to form sentences and carry out the similarity. The tree helps the program to formulate a proper question sentence and also at the same time look for some missing grammatical mistakes. Here we use lemmatization to find the root words and if the words have some alphabets missing they can be added. The mistakes can be some extra words like in, at, etc. which weren’t removed during question formation or some extra words left that will make the sentence grammatically correct. Below is an example of question and answer generation from a given input sentence as per the proposed approach.

Input Sentence:
>>> He went bankrupt because he took too many loans.
Tokenization:
>>> ['He', 'went', 'bankrupt', 'because', 'he', 'took', 'too', 'many', 'loans', '.']
Lowercasing, Stemming and Lemmatization:
>>> ['he', 'went', 'bankrupt', 'because', 'he', 'took', 'too', 'many', 'loan', '.']
Preprocessed Sentence:
>>> ['he', 'went', 'bankrupt', 'because', 'he', 'took', 'too', 'many', 'loan', '.']
Discourse Marker Identified:
>>> ‘because’
Map the discourse marker to Wh-type Question according to Table 3:
>>> ‘why’
Target arguments according to Table 3:
>>> arg1 (question part): ‘he went bankrupt’
>>> agr2 (answer part): ‘he took too many loans’
POS Tagging question part:
>>> [('he', 'PRP'), ('went', 'VBD'), ('bankrupt', 'NN')]
Identification of verb based on Noun-Verb combination:
>>> no auxilary verb case: NNP+VBD= ‘did’
Question Formation:
>>> Why did he went bankrupt?
Answer POS Tagging and Formation:

>>> [('He', 'PRP'), ('went', 'VBD'), ('bankrupt', 'RB'), ('because', 'IN'), ('he', 'PRP'), ('took', 'VBD'), ('too', 'RB'), ('many', 'JJ'), ('loans', 'NNS'), ('.', '.')]
QnA Pair Generated:
>>> [['He went bankrupt because he took too many loans.', 'Why did he went bankrupt ?']]
QnA pair with Bloom's Taxonomy level identified for Question Stem (Why did…) according to Table 2:
>>> [['He went bankrupt because he took too many loans.', 'Why did he went bankrupt ?', 'Analysis']]

Finally, the question and answer pairs generated are stored in a database with details about the course, module along with which level of Bloom's Taxonomy a particular question belongs to. Additionally, the spreadsheet or existing question bank of a particular professor for a particular test can be integrated into the database as the more the number of questions, the less is the chance of question repetition.

B. Question Paper Generation Using Bloom's Taxonomy

The system is provided with a predefined list of question cues and question stems for each category of Bloom's Taxonomy. Using POS tags of the tokens in the question, the accurate level of Bloom's Taxonomy is identified for the question. The examiner has to specify the number of questions for a specific category on the UI of the system developed such as "Knowledge"-based 5 questions, "Application"-based 3 questions and "Analysis"-based 2 questions. Out of the question repository, 5 random questions pertaining to the "Knowledge" category, 3 random questions pertaining to the "Application" category and 2 random questions pertaining to the "Analysis" category are chosen to generate the question paper. The random module of python is used for the random selection of questions stored in the database. The Mersenne Twister PRNG algorithm is used by the 'rand' function and has a period of 2**19937-1, ensuring a purely random and unbiased question paper. When the pdf of the question paper is generated, appropriate answers for the selected question are simultaneously inserted in a separate pdf file.

VII. RESULTS

We evaluate the performance of the system by providing paragraphs with a varied number of sentences as in input. The questions generated were checked and compared with questions generated with human English proficiency. Following this iterative process 10 times, the attributes of a confusion matrix (TP, TN, FP, and FN) were calculated. Further, using these values, performance parameters- accuracy, precision, and recall are calculated and represented graphically. Accuracy can be defined as the proximity of given values to the true value. By analyzing the results we can say that the system works with an accuracy of 72.9%. Precision shows us the closeness of different measured values to each other and Recall depicts the fraction of relevant instances to the total number of values retrieved. Figure 3 depicts the graphical representation of the performance measures calculated (where the X-axis denotes the number of sentences and the Y-axis represents accuracy, precision, recall respectively).

TABLE 4. CALCULATION OF PERFORMANCE MEASURES

Sr. No.	Number Of Sentences	TP	TN	FP	FN	Accuracy =(TP+TN) / (TP+TN+FP+FN)	Precision = TP / (TP+FP)	Recall = TP / (TP+FN)
1	0	0	0	0	0	0	0	0
2	1	1	1	0	0	1	1	1
3	2	3	1	1	1	0.667	0.75	0.75
4	3	3	2	2	1	0.625	0.6	0.75
5	5	4	2	2	1	0.667	0.667	0.8
6	10	8	3	3	2	0.688	0.727	0.8
7	15	11	6	5	3	0.68	0.688	0.786
8	20	20	5	7	2	0.735	0.741	0.909
9	25	22	7	6	4	0.744	0.786	0.846
10	30	24	7	6	4	0.756	0.8	0.857

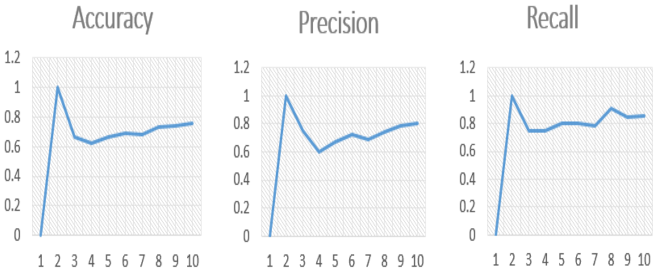

Fig. 3. Graph of Accuracy, Prescision and Recall calculated for the proposed system

VIII. CONCLUSION

Many processes in the educational domain are carried out manually. The main aim of the paper is to reduce the gap between manpower and technology by focusing on automating the task of question paper generation. Various approaches and methodologies adopted by existing papers were studied and analyzed to propose an AQGS system using NLTK in python language. The system accepts text passages as input that is subjected to tokenization, lemmatization and stemming for pre-processing. Potential sentences are selected from these processed phrases with help of discourse markers which undergo syntactic analysis using POS tagging and semantic analysis using NER. Grammatically sound questions are formed using NER and syntax tree and are stored in the database after mapping each to an appropriate Bloom's taxonomy. The test paper is generated by random selection of questions for a specific category of the taxonomy used. Future work of the system includes increasing the accuracy of the system by enhancing question framing. Questions other than wh-questions (like true/false, MCQs, etc.) can be incorporated. An Answer evaluation module can be integrated to evaluate and score the test answers submitted by students by calculating semantic similarity with the correct answer. Out of the numerous papers on approaches for question generation, this paper focuses on the implementation of AQGS system in python to contribute to automated, quick, unbiased question paper generation.

ACKNOWLEDGMENT

We would like to thank our project guide Prof. Ravindra Divekar and Project Committee for their indispensable support, suggestions and timely guidance.

REFERENCES

[1]. D. R. CH and S. K. Saha, "Automatic Multiple Choice Question Generation From Text: A Survey," in IEEE Transactions on Learning Technologies, vol. 13, no. 1, pp. 14-25, 1 Jan.-March 2020, doi: 10.1109/TLT.2018.2889100.

[2]. Narendra, A., Manish Agarwal and Rakshit shah, "Automatic Cloze-Questions Generation." RANLP, 2013.

[3]. Agarwal, Manish & Shah, Rakshit & Mannem, Prashanth, Automatic question generation using discourse cues, 2011, pp. 1-9.

[4]. Eldesoky, Ibrahim. (2015). Semantic Question Generation Using Artificial Immunity. I.J. Modern Education and Computer Science. 7. 1-8. 10.5815/ijmecs.2015.01.01.

[5]. L. R. Pillai, V. G. and D. Gupta, "A Combined Approach Using Semantic Role Labelling and Word Sense Disambiguation for Question Generation and Answer Extraction," 2018 Second International Conference on Advances in Electronics, Computers and Communications (ICAECC), Bangalore, India, 2018, pp. 1-6, doi: 10.1109/ICAECC.2018.8479468.

[6]. Deokate Harshada G., Jogdand Prasad P., Satpute Priyanka S., Shaikh Sameer B., Automatic Question Generation from Given Paragraph, IJSRD - International Journal for Scientific Research & Development| Vol. 7, Issue 03, 2019 | ISSN (online): 2321-0613

[7]. Kalpana B. Khandale, Ajitkumar Pundage ,C. Namrata Mahender, Similarities in words Using Different Pos Taggers, IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661,p-ISSN: 2278-8727, PP 51-55

[8]. Anderson LW, Krathwohl DR. A taxonomy for learning, teaching, and assessing: a revision of Bloom's taxonomy of educational objectives. New YorkNY: Longmans; 2001