AI-Driven MCQ Generation Using NLP

Dr. Rajesh Saturi[1], Gorige Anusha[2], C. Sony[3]
[1]Associate Professor, Computer Science and Engineering Vignana Bharathi Institute of Technology
[2,3] B.Tech Honors Student, Computer Science and Engineering Vignana Bharathi Institute of Technology

***Abstract*—An AI-driven framework is proposed for automating the generation of Multiple-Choice Questions (MCQs) using Natural Language Processing (NLP), with the T5 model at its core. Traditional methods of MCQ creation are labor-intensive and depend heavily on subject-matter experts, limiting efficiency and scalability. This system addresses those limitations by leveraging transformer-based NLP models capable of processing educational materials in formats such as PDF, DOCX, and TXT. It generates context-aware questions and relevant distractors through modules focused on question generation, answer identification, and semantic similarity. The backend is implemented using FastAPI, while the user interface is built with React. Evaluation based on language quality, answer accuracy, and distractor relevance shows that the AI-generated MCQs closely match the quality of manually crafted ones, making the solution highly suitable for educational purposes.**

***Index Terms*—AI-driven MCQ generation, Natural Language Processing, T5 Model, Text-to-Text, NLP-based Question Generation**

1. INTRODUCTION

As digital learning continues to expand, the need for efficient and scalable assessment tools has become increasingly important. One key challenge in this space is the generation of Multiple-Choice Questions (MCQs), which traditionally demands significant time and subject knowledge from educators. This manual process limits adaptability, especially in dynamic or large-scale learning environments.

To address this, recent developments in Natural Language Processing (NLP), particularly the use of transformer- based models, offer promising solutions. The T5 model, known for its ability to reframe diverse NLP tasks into a unified text-to-text format, plays a central role in automating question generation. When combined with semantic search techniques for generating meaningful distractors and NLP-driven answer extraction methods, it becomes possible to automate the creation of high-quality, context-aware MCQs. This paper presents a modular, AI-based framework that brings these technologies together in a streamlined pipeline. Users can upload PDF-based learning content, which is processed through layers of text parsing, concept identification, and question generation. The backend, built with Fast API, handles the processing logic, while a React-based frontend offers an accessible user experience. Designed to support both academic and professional learning scenarios, this system significantly reduces manual workload while ensuring the quality and relevance of the generated questions.

2. LITERATURE SURVEY

The automation of MCQ generation using NLP has been explored extensively in recent years, with various approaches offering their own methodologies, strengths, and limitations.

Ayushi Mathur and Dr. M. Suchithra proposed a method using abstractive summarization to simplify source content before MCQ creation. This method improved the linguistic fluency and coherence of questions, but it also posed the risk of omitting critical context from the original content (2024) [1].

Matthew Martianus Henry and Nur Adhianti Heryanto presented a broad literature review comparing rule-based, statistical, and deep learning methods for question generation. Their study offered valuable theoretical comparisons but lacked practical validation and empirical results (2024) [2].

Pochiraju and Chakilam introduced a hybrid approach that combines extractive summarization with XLNet for refining question context and accuracy. This ensured that factual content was retained, although the model relied heavily on redundant text and struggled with creativity in question formation (2023) [4].

Rao and Saha's two contributions include a survey of MCQ generation methods and a domain-specific model focused on school textbooks. Their survey categorized methods well, while the textbook model was effective for curriculum-aligned MCQs. However, both approaches were less flexible for broader or dynamic content inputs (2022) [5][7].

Altaj Virani and Rakesh Yadav developed a pipeline utilizing the T5 model for question generation and BERT for answer extraction. Although their method produced strong QA pairs, it did not address

distractor generation or UI deployment, limiting real-world applicability (2023) [6].

Hadifar and Bitew explored distractor reuse techniques by leveraging semantic similarity. This promoted diversity and efficiency but raised concerns about contextual relevance if not carefully validated (2024) [9].

Finally, Pawar and Dube implemented an end-to-end MCQ generation system using Langchain and Gemini LLM. Their model leveraged prompt engineering to generate dynamic MCQs but was resource-intensive and heavily dependent on precise prompt formulation (2022) [10].

3. PROPOSED SYSTEM ARCHITECTURE

The proposed system extracts text from uploaded documents of various PDF formats, including scanned files, text-based PDFs, and academic articles. It also supports other common document formats such as DOCX and plain TXT files, processes the extracted text, and generates multiple-choice questions (MCQs) with relevant answer choices. The backend is built using *FastAPI*, while the frontend is developed using *React*. The overall workflow consists of text extraction, MCQ generation, and final question refinement.

The structural diagram of an MCQ generation system outlines a sequential process. It begins with Text Acquisition, where the source material for questions is obtained. This is followed by Preprocessing, which likely involves cleaning and preparing the text for further analysis. Next, Question Generation algorithms create potential questions based on the processed text. Answer Selection identifies the correct answer for each generated question. The Distractor Generator then creates plausible but incorrect options. Finally, the MCQ Display presents the generated multiple-choice questions.

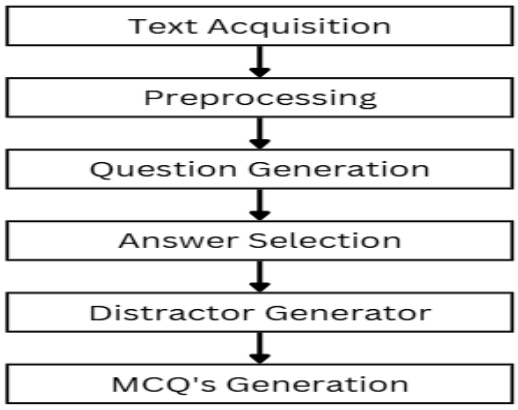

Figure 1: Structural Diagram of MCQ's Generation System

4. METHODOLOGY

The proposed system uses a multi-step process to automatically generate Multiple-Choice Questions (MCQs) from educational documents. It consists of two main components: a FastAPI-based backend for processing and a React frontend for user interaction. The backend handles file uploads, text extraction, and MCQ generation, while the frontend manages user inputs and displays the results.

4.1. Backend Implementation:

- FastAPI was chosen for the backend due to its high-performance and asynchronous capabilities. Cross-Origin Resource Sharing (CORS) middleware is added to enable communication between the frontend and backend. Uploaded files—such as PDFs, DOCX, or TXT—are processed using PyMuPDF to extract the text content. If the extraction fails or returns empty, the system notifies the user with an error message.
- After extracting the text, several NLP models process the content:
- Question Generation: The valhalla/t5-base-qg-hl model generates questions from key sentences.
- Answer Extraction: The deepset/roberta-base-squad2 model identifies accurate answers directly from the text.
- Distractor Generation: Using BAAI/bge-small-en-v1.5, a sentence-transformer model, the system finds distractors that are similar in context but incorrect, ensuring meaningful answer choices.

Once the MCQs are created, they undergo a validation phase. This includes grammar checks, difficulty evaluation, and proper formatting to ensure the questions meet quality standards. The backend also includes error handling for consistent performance at every stage.

4.2. Frontend Interface:

The frontend is developed using React, offering an intuitive user interface. Users can upload files and receive generated MCQs in a clean, readable format. React's state management (e.g., useState) is used to track the uploaded file, loading status, and server response.

- For file uploads, Axios sends a POST requestwith the document in multipart/form-data format to the FastAPI backend. The backend returns a structured response containing the question, correct answer, and distractors, which is then rendered dynamically on the screen. A loading spinner provides real-time feedback during processing.

- Supported file formats include PDF, DOCX, and TXT, ensuring broad usability across different content types.
- The internal pipeline is designed to ensure that the generated MCQs are accurate, contextually relevant, and educationally appropriate. The process includes:
- Concept Identification: Using Named Entity Recognition (NER) and dependency parsing, the system identifies important terms and concepts from the text to guide question creation.
- Question Generation: The T5-based model creates well-formed, grammatically correct questions based on the identified content, maintaining coherence with the original material.
- Answer Selection: The Roberta-based model extracts exact answers from the text, ensuring they are precise and relevant.
- Distractor Creation: Semantic similarity techniques are used to produce distractors that are related to the topic but clearly incorrect, avoiding ambiguity and ensuring fairness.
- Validation and Formatting: A final review checks grammar, structure, and difficulty level. MCQs are then formatted for clarity and usability.

5. RESULTS AND DISCUSSION

5.1. Comparative Analysis

To evaluate the effectiveness of our proposed system, we compare it with the work titled "Automatic Multiple Choice Question Generation from Text: A Survey" by Dhawaleswar Rao CH and Sujan Kumar Saha (2022) [5]. Their survey compiles and categorizes various MCQ generation approaches, including rule-based, statistical, and neural models. While comprehensive, their work primarily focuses on theoretical exploration and categorization of methodologies without presenting a complete, practical implementation pipeline.

In contrast, our system offers a full-stack solution that includes both backend (FastAPI) and frontend (React) components. It supports diverse document formats such as PDF, DOCX, and TXT, and processes both structured and scanned files. We employ transformer-based models not only for question generation (T5) and answer extraction (RoBERTa) but also introduce semantic similarity via Sentence Transformers for distractor generation—something not covered in the surveyed models.

Table:Comparative Analysis

Criteria	Existing system [5]	Proposed System
Type of Work	Theoretical review	Practical end-to-end implementation
Question Generation	General neural model discussion	T5 transformer model
Answer Extraction	Not implementedStatistical, Neural	RoBERTa (QA model)
Distractor Creation	Not covered in depth	Semantic search using Sentence Transformers
Evaluation	Qualitative	Qualitative (improved accuracy)

Unlike the survey, which evaluates techniques qualitatively, our system was evaluated through internal testing, showing promising results in generating questions that align well with expected educational outcomes. This highlights the practical viability and precision of our pipeline, which integrates question logic, validation layers, and user interactivity for real-time assessment creation

The AI-driven MCQ generator demonstrates the ability to create structured and meaningful assessment questions. Performance evaluation is based on:

- Accuracy & Precision: Measures the relevance and correctness of generated MCQs.
- Semantic Analysis: Ensures logical consistency in question-answer pairs.
- User Feedback: Reviews from educators validate the effectiveness of automated MCQs.

6. CONCLUSION

This study presents an automated approach to MCQ generation using NLP techniques, particularly T5 and text-to-text transformations. By utilizing AI, the process of question generation is enhanced in terms of efficiency, scalability, and accuracy. Future work will focus on domain-specific customization and

improving the quality of generated distractors for broader applications in education and training.

7. FUTURE SCOPE

In the future, this system can be extended to generate questions from visual elements such as diagrams, flowcharts, and tables. Integrating image processing and OCR techniques can allow the extraction of visual context for question formation. Additionally, implementing adaptive difficulty levels by categorizing questions into easy, moderate, and advanced can enhance learner engagement and personalized assessment. These improvements would further broaden the applicability of the system across a variety of educational settings.

REFERENCES

[1]. Pratik Pawar, Raghav Dube (2024). Automated Generation and Evaluation of Multiple-Choice Quizzes using Langchain and Gemini LLM.

[2]. Matthew Martianus Henry, Nur Adhianti Heryanto (2024). Automatic Multiple Choice Question Generation: A Systematic Literature Review.

[3]. V. Shobha Rani (2024). The Future of Assessment: Automating Multiple Choice Question Generation.

[4]. Dhanamjaya Pochiraju, Abhinav Chakilam (2023). Extractive Summarization and Multiple-Choice Question Generation using XLNet.

[5]. Dhawaleswar Rao CH, Sujan Kumar Saha (2022). Automatic Multiple Choice Question Generation from Text: A Survey.

[6]. Altaj Virani, Rakesh Yadav (2023). Automatic Question Answer Generation using T5 and NLP.

[7]. Dhawaleswar Rao CH, Sujan Kumar Saha (2023). Generation of Multiple-Choice Questions from Textbook Contents of School-Level Subjects.

[8]. Mohammed Ashraf Mohammed, Mostafa Ashraf Borhamy (2023). End-to-End Quiz-Style Question Generation for Educational Purposes.

[9]. Amir Hadifar, Semere Kiros Bitew (2024). Learning to Reuse Distractors to Support Multiple-Choice Question Generation in Education.

[10]. Ayushi Mathur, Dr. M. Suchithra (2022). Application of Abstractive Summarization in Multiple Choice Question Generation.

[11].]Manning, C.D, and Schutze, H., 2007 "Foundations of Statistical Natural Language Processing" The MIT Press.

[12]. Goyvaerts, P. 1996 "Controlled English, curse or blessing? A user perspective" Proceedings of the 1st International workshop on Controlled Language Applications (CLAW '96) (Leuven). 137-42 .

[13]. Foster, R.M. 2009 "Improving the Output from Software that Generates Multiple Choice Question (MCQ) Test Items Automatically using Controlled Rhetorical Structure Theory" In proceedings of RANLP 2009, Borovets – Student Conference.

[14]. Mitkov, R., and L. A. Ha. 2003. "Computer-Aided Generation of Multiple-Choice Tests." In Proceedings of HLT-NAACL 2003 Workshop on Building Educational Applications Using Natural Language Processing, pp. 17-22. Edmonton, Canada.

[15]. Mitkov, R., L. A. Ha, and N. Karamanis. 2006. "A computer-aided environment for generating multiple-choice test items." Natural Language Engineering 12(2): 177-194.

[16]. Reiter and R. Dale. 2000. "Building natural language generation systems." Cambridge University Press.

[17]. Mitkov, R., 2004. "The Oxford Handbook of Computational Linguistics "Oxford University press

[18]. Rodriguez-Torrealba, R., Garcia-Lopez, E., & Garcia-Cabot, A. (2022). End-to-end generation of multiple-choice questions using text-to-text transfer transformer models. Expert Systems with Applications, 208, 118258.

[19]. Tami, M., Ashqar, H. I., & Elhenawy, M. (2024). Automated Question Generation for Science Tests in Arabic Language Using NLP Techniques. arXiv preprint arXiv:2406.08520.

[20]. Vichare, S., Gawade, A., & Mangrulkar, R. (2024). Qgen: A Unique Question Generation and Answer Evaluation Technique Using Natural Language Processing. Journal Transformations, 38(1). of Engineering Education

[21]. Ling, J., & Afzaal, M. (2024). Automatic question-answer pairs generation using pre-trained large language models in higher education. Computers and Education: Artificial Intelligence, 100252.

[22]. Barlybayev, A., & Matkarimov, B. (2024). Development of system for generating

questions, answers, distractors using transformers. International Journal of Electrical & Computer Engineering (2088-8708), 14(2).

[23]. Virani, A., Yadav, R., Sonawane, P., & Jawale, S. (2023, June). Automatic question answer generation using T5 and NLP. In 2023 International Conference on Sustainable Computing and Smart Systems (ICSCSS) (pp. 1667-1673). IEEE.