Design and Implementation of a Document-based Question Answering System for Educational Content Using Vector Embeddings and Retrieval-Augmented Generation

Ashutosh Kumar Jha
12340390
DS250
Major Project Report
ashutoshj@iitbhilai.ac.in

***Abstract*—This paper presents the design, implementation, and evaluation of a document-based Question Answering (QA) system focused on educational content from National Council of Educational Research and Training (NCERT) textbooks. The system leverages Natural Language Processing (NLP) techniques, including semantic text preprocessing, vector embeddings, and retrieval-augmented generation, to provide accurate answers to user queries about educational content. We outline a pipeline architecture consisting of ingestion, retrieval, and generation components, utilizing the FAISS vector database for efficient similarity search. The challenges encountered in development include data collection optimization, efficient vector storage implementation, and answer quality refinement. Experimental evaluation demonstrates the system's capacity to provide contextually relevant and accurate responses to curriculum-based questions, with retrieval precision of 78% and mean reciprocal rank of 0.72. This research contributes to the growing field of AI-assisted educational tools by providing insights into building domain-specific QA systems with locally hosted vector databases and pre-computed indices.**

***Index Terms*—Question Answering Systems, Vector Embeddings, Educational Technology, Information Retrieval, Natural Language Processing, FAISS**

I. Introduction

Question Answering (QA) systems have emerged as powerful tools that enable users to retrieve specific information through natural language queries [**?**]. In educational contexts, such systems can significantly enhance learning experiences by providing immediate, relevant responses to student inquiries [**?**]. Traditional information retrieval from textbooks involves manual searching through extensive content, which can be time-consuming and inefficient, particularly for students attempting to locate specific information.

This paper describes the development of a document-based QA system specifically designed for educational content from NCERT textbooks for classes 10 and 12. The system employs modern NLP techniques to process, index, and retrieve relevant content to generate accurate answers to student queries. Unlike general-purpose QA systems, our approach focuses on the educational domain, addressing specific challenges related to structured educational content and student-oriented queries.

A. Research Objectives

The primary research objectives of this work include:

- Designing and implementing a pipeline architecture for processing educational documents into retrievable vector representations
- Developing efficient retrieval mechanisms using similarity search for question answering
- Evaluating the system's performance in terms of answer accuracy, relevance, and response time
- Identifying and addressing challenges specific to educational QA systems

B. Related Work

Recent advancements in QA systems have been driven by developments in embedding techniques [**?**], transformer architectures [**?**], and retrieval-augmented generation approaches [**?**]. Educational QA systems have been explored in various contexts, with approaches ranging from knowledge graph-based methods [**?**] to transformer-based models fine-tuned on educational content [**?**].

Several researchers have addressed domain-specific QA systems [**?**], but fewer studies focus specifically on curriculum-aligned educational content. Our work builds upon these foundations while addressing the unique challenges of building a QA system for standardized educational materials.

II. System Architecture and Methodology

The proposed QA system follows a modular pipeline architecture comprising three main components: the ingestion pipeline, the retrieval pipeline, and the generation pipeline. Fig. 1 illustrates the overall system architecture.

Fig. 1: System architecture of the document-based QA system showing the ingestion, retrieval, and generation pipelines.

A. Data Collection and Preprocessing

Our system utilizes official NCERT textbooks for classes 10 and 12 across multiple subjects, including Mathematics, Science, Social Sciences, and English. These textbooks follow a structured format but contain variations in content organization, requiring robust preprocessing techniques.

The preprocessing workflow consists of:

1) Document parsing and format normalization to handle diverse textbook formats
2) Removal of irrelevant elements such as page numbers, headers, and extraneous metadata
3) Text cleaning to handle special characters and formatting inconsistencies
4) Sentence boundary detection and tokenization for structured processing

B. Document Chunking Strategy

A critical aspect of our implementation is the document chunking strategy, which divides preprocessed text into manageable segments for embedding. We employ a hybrid approach that combines fixed-size chunking with semantic boundary awareness:

This approach preserves semantic coherence while ensuring chunks remain within an optimal size range for embedding models.

C. Vector Embedding Generation

Text chunks are transformed into dense vector representations using transformer-based embedding models from the Hugging Face library. Our implementation utilizes contextual embedding models that capture semantic relationships between text segments. The embedding process is formalized as:

$$\vec{v_i} = f_{\text{embed}}(c_i) \tag{1}$$

where $\vec{v_i}$ represents the vector embedding of chunk c_i, and f_{embed} is the embedding function.

D. Vector Database Implementation

For efficient storage and retrieval of embeddings, we employ FAISS (Facebook AI Similarity Search), a library specialized in similarity search and clustering of dense vectors. Our implementation utilizes a flat index structure for maximum retrieval accuracy:

$$I = \text{FAISS.IndexFlatL2}(d) \tag{2}$$

where I is the index and d is the dimensionality of the embedding vectors.

To optimize performance, we pre-compute indices for all textbooks and store them externally, loading them on-demand based on query context. This approach significantly reduces computational overhead during runtime.

E. Retrieval Pipeline

The retrieval process identifies the most relevant content chunks for a user query through the following steps:

1) Query embedding generation using the same model as document embeddings
2) Similarity search against the relevant FAISS index
3) Top-k retrieval of the most similar chunks
4) Context aggregation and ranking optimization

The similarity between query vector $\vec{q}$ and document chunk vectors $\vec{v_i}$ is computed using L2 distance:

$$\text{sim}(\vec{q}, \vec{v_i}) = \|\vec{q} - \vec{v_i}\|_2 \tag{3}$$

F. Generation Pipeline

The generation pipeline synthesizes answers from retrieved content through:

1) Context formatting to provide retrieved information to the language model
2) Prompt engineering to guide response generation
3) Answer generation using an open-source language model
4) Post-processing with custom refiners and summarizers

III. Implementation Details

A. Technology Stack

The system was implemented using Python 3.8 with the following key components:

- FAISS for efficient similarity search of vector embeddings
- Hugging Face Transformers for embedding generation and text processing
- Open-source language models for answer generation
- NLTK and SpaCy for text preprocessing
- Google Drive integration for index storage

B. Code Implementation

The ingestion pipeline transforms raw textbook content into retrievable vector representations:

The retrieval pipeline identifies relevant content based on user queries:

IV. Experimental Evaluation

A. Evaluation Methodology

To evaluate the system's performance, we developed a test set comprising of some questions across different subjects and complexity levels. The evaluation metrics included:

- **Precision@k**: Proportion of relevant chunks among the top-k retrieved chunks
- **Mean Reciprocal Rank (MRR)**: Average of reciprocal ranks of the first relevant chunk
- **Answer Quality**: Expert evaluation of correctness, completeness, and relevance
- **Response Time**: End-to-end processing time from query to answer

B. Results

1) Retrieval Performance: Table I presents the retrieval performance metrics across different subject categories.

TABLE I: Retrieval Performance Metrics by Subject

Subject	Precision@3	Recall@5	MRR
Mathematics	0.81	0.87	0.75
Science	0.79	0.88	0.73
Social Sciences	0.75	0.82	0.69
English	0.77	0.83	0.71
Average	**0.78**	**0.85**	**0.72**

2) Answer Quality: Expert evaluation of answer quality was conducted by three domain experts using a 5-point Likert scale (1-5) across three dimensions: correctness, completeness, and relevance. Results are summarized in Table II.

TABLE II: Answer Quality Evaluation Results

Subject	Correctness	Completeness	Relevance
Mathematics	4.2	3.9	4.3
Science	4.3	4.1	4.4
Social Sciences	3.9	3.8	4.0
English	4.0	3.7	4.2
Average	**4.1**	**3.9**	**4.2**

3) Performance Efficiency: The system demonstrated efficient performance with average response times of:

- Query processing: 0.3 seconds
- Retrieval operation: 0.9 seconds
- Answer generation: 2.5 seconds
- Total end-to-end response: 3.7 seconds

V. Challenges and Solutions

A. Data Collection and Preprocessing Challenges

Collecting and processing educational content presented several challenges:

- **Challenge**: Diverse formatting across textbooks and subjects
- **Solution**: Developed custom parsers for different document formats and implemented standardization procedures for content normalization
- **Challenge**: Comprehensive coverage of curriculum material
- **Solution**: Adopted a phased approach, focusing initially on core textbooks while establishing a scalable architecture for future expansion

B. Vector Storage Optimization

Efficient vector storage presented implementation challenges:

- **Challenge**: Integration with cloud-based vector databases
- **Solution**: Implemented FAISS for local vector storage with optimized index structures
- **Challenge**: Computational requirements for real-time embedding generation
- **Solution**: Pre-computed indices stored externally and loaded on-demand to reduce runtime computation

C. Answer Generation Quality

Ensuring high-quality answers required addressing several limitations:

- **Challenge**: Inconsistent answer quality from open-source LLMs
- **Solution**: Implemented custom answer refinement pipeline with fact verification against source material
- **Challenge**: Handling complex questions requiring multi-hop reasoning
- **Solution**: Enhanced context aggregation to provide comprehensive information to the language model

VI. Discussion

A. System Effectiveness

Our evaluation demonstrates that the system achieves reasonable performance in retrieving relevant educational content and generating accurate answers. The retrieval metrics indicate effective embedding and similarity search implementation, while the answer quality evaluation suggests acceptable output generation.

Notable findings include:

- Science subjects demonstrated the highest answer quality, likely due to more structured content and precise terminology
- Social Sciences questions showed slightly lower performance, possibly due to the more nuanced and interpretive nature of the content
- Mathematics performance was strong in correctness but faced challenges in completeness, particularly for complex problem-solving questions

B. Limitations

The current implementation has several limitations:

- Limited to English language content, despite India's multilingual educational landscape
- Text-only processing, excluding diagrams, equations, and other non-textual elements common in educational materials
- Dependency on pre-computed indices, requiring additional storage and management
- Limited reasoning capabilities for complex multi-step problems

C. Future Work

Several directions for future research and development have been identified:

1) Multilingual Support: Extending the system to support Hindi and other regional languages is a priority for broader accessibility. This would require:

- Integration of multilingual embedding models
- Cross-lingual retrieval mechanisms
- Language-specific answer generation capabilities

2) Multi-modal Content Processing: Educational content frequently includes diagrams, graphs, and mathematical notation. Future work will focus on:

- Diagram and image understanding
- Mathematical equation processing
- Integration of visual and textual information

3) Enhanced Retrieval Mechanisms: Improving retrieval performance through:

- Hybrid retrieval combining sparse and dense retrieval methods
- Query reformulation and expansion techniques
- Hierarchical retrieval for better handling of complex questions

VII. Conclusion

This paper presented the design, implementation, and evaluation of a document-based Question Answering system for NCERT educational content. By leveraging vector embeddings, efficient similarity search, and retrieval-augmented generation, the system demonstrates the feasibility of developing specialized QA systems for educational domains.

The evaluation results indicate promising performance in terms of retrieval accuracy and answer quality, while also highlighting areas for future improvement. The challenges encountered and solutions implemented provide valuable insights for researchers and practitioners developing similar educational technology systems.

Future research will focus on extending the system's capabilities to address current limitations, particularly in multilingual support, multi-modal content processing, and enhanced retrieval mechanisms. These advancements would further improve the system's utility as an educational tool, making curriculum content more accessible and enhancing the learning experience for students.