Towards Personalized Education: Introducing NCTB Dataset in Learning Environments for Bangla Language

by

Md. Mahinuzzaman Shaan
21101208
Nuzhat Tahsin
21301206
Afridi Alam Polock
21301303
Lamia Alam Emu
21301662
Nafisa Mehreen
23241114

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
June 2025

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at BRAC University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

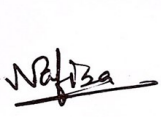
Student's Full Name & Signature:

Md. Mahinuzzaman Shaan
21101208

Nuzhat Tahsin
21301206

Afridi Alam Polock
21301303

Lamia Alam Emu
21301662



Nafisa Mehreen
23241114

Approval

The thesis titled "Towards Personalized Education: Introducing NCTB Dataset in Learning Environments for Bangla Language" submitted by

1. Md. Mahinuzzaman Shaan (21101208)
2. Nuzhat Tahsin (21301206)
3. Afridi Alam Polock (21301303)
4. Lamia Alam Emu (21301662)
5. Nafisa Mehreen (23241114)

Of Spring, 2024 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on June, 2025.

Examining Committee:

Supervisor:

Muhammad Iqbal Hossain, PhD
Associate Professor
Department of Computer Science and Engineering
BRAC University

Co-Supervisor:

Md. Sabbir Ahmed
Lecturer
Department of Computer Science and Engineering
BRAC University

Thesis Coordinator:
(Member)

Md. Golam Rabiul Alam, PhD
Associate Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Sadia Hamid Kazi
Associate Professor
Department of Computer Science and Engineering
BRAC University

Abstract

The core contribution of this paper lies in con structing and employing a high-quality, Bangla-language dataset specifically for educational settings. While AI and NLP have transformed learning environments elsewhere in the world, Bangla, as the seventh most spoken language, remains under represented, particularly in education. This piece addresses the lack by focusing on constructing curriculum-aligned dataset that comes in the form of Bangla context paragraphs and corre sponding question-answer pairs. These sources of data are mined and cleansed from national curriculum textbooks and teach ing resources, ensuring linguistic appropriateness and academic usefulness. The cleaned dataset is poised to facilitate natural language processing algorithms bespoke to Bangla, enabling the development of smart systems that can successfully understand, process, and generate context-dependent responses appropriate to education settings. With this work, we intend to clear the way for more available, localized AI-powered learning environments that are accessible for Bangla-speaking students, especially in low-resource environments.

Keywords: Dataset; Natural Language Processing; Bangla; Large Language Model; Tokenizer;

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Introduction

Artificial Intelligence (AI) is transforming education worldwide. It provides customized, efficient, and responsive solutions to common learning obstacles. Natural Language Processing (NLP) enables education apps to replicate human-like comprehension and communication with students. The apps can give customized feedback and guidance. Hasan et al.[21] stated that these technologies are revolutionizing the manner in which students interact with streams of knowledge, enabling systems to ascertain understanding, recommend resources, and guide students through the learning process. But according to Rahman et al.,[19] most of this advancement has almost catered to English and only a limited cluster of high-resource languages, while lagging behind languages like Bangla which, despite being the seventh most spoken language in the world, is significantly under-represented in AI and NLP advancements.

Vast numbers of students in Bangladesh are enrolled in Bangla-medium schools where English-based digital educational content is not capable of serving them. This pattern is most prevalent in poor and rural communities, where they have poor access to good education and technological advancement. Chowdhury and Shahnaz [14] observed that such children are exposed to numerous challenges. Besides having limited access to the internet, they also have learning materials unavailable in their native language. The absence of AI-based learning solutions in Bangla is a component of this difference and limits students' ability to self-learn, reiterate or assess their preparation. According to Alam et al. [7], an education AI product in Bangla would not just provide accessibility but empowerment, allowing learners to control learning in their own comfort language.

It has been shown through research that intelligent learning environments, when linguistically and culturally localized, could have a significant impact on learning engagement and comprehension stated by Patel. [12] A Bengali-medium curriculum content-backed intelligent system powered by artificial intelligence can serve as a virtual tutor. It can explain concepts, test knowledge, and respond to questions. In low-resource settings where human tutors may not be available or affordable, such tools are essential. Islam and Jahan [22] referenced that technology-driven interventions, especially if tailored for the national language, can assist in bridging

Bangladesh's urban-rural knowledge divide. Moreover, according to Roy et al [23], if they are made with local feedback, such systems can accommodate students from diverse dialectal and socio-economic backgrounds, thus making education inclusive.

Sarker et al. [32] believes that despite the evident need and potential, some hindrances are on the way to developing stable Bangla NLP systems. One of them is the lack of large-scale, annotated datasets for training and testing AI models . Normally, latest models like BERT or T5 have been trained on English or multilingual datasets that do not reflect the grammatical and contextual richness of Bangla. Hence, fine-tuning them on Bangla educational datasets is crucial for actual-world performance. Additionally, language-specific phenomena like compound words, free word order, and orthographic variation render Bangla NLP particularly challenging compared to structurally less complex languages like English which is stated by Ahmed et al.[6]

The limitations don't end at information, rather deployment is also limited by infrastructural factors. Even with improved smartphone availability, schools and homes in rural areas in Bangladesh continue to have weak internet connectivity and a shortage of digital equipment supplies. Rahim et al. [18] reported that these infrastructural shortcomings constrain the potential scope of AI-driven learning platforms. Even where there is connectivity, digital literacy is low, particularly among teachers and parents which makes it more difficult to infuse AI tools into the general learning environment.

This can pose issues that incline with algorithmic fairness. If AI systems are trained on non-representative datasets which are not validated from an unbiased standpoint, they are likely to generate biased results, which can lead to the making of an unfair learning system. Mehrabi et al. [17] pointed out that biases in NLP models are created because of unbalanced data or lack of dialectal variation which can then skew results and damage user's trust. In education, bias can lead to misleading criticism or incorrect levels of difficulty, damaging the performance of students. Therefore, fairness, transparency and representative training are not an option one might wish to do but essential to the researcher's moral obligation.

To address these issues, our research focuses on building a high-quality Bangla education dataset. This dataset is constructed using national curriculum textbooks and consists of context paragraphs, questions, and appropriate answers that are handcurated, maintaining the linguistic richness and pedagogical structure of Bangla-medium learning. Our dataset allows for fine-tuning BanglaT5 and similar pre trained models for the education question-answering task. By grounding the data on actual materials that Bangladeshi pupils study, we ensure contextual pertinence, language usability, and educational conformity.

In addition to filling a fundamental gap in Bangla NLP resources, our dataset supports fair and accessible AI development. It supports bias analysis, dialectal variation testing, and lightweight deployment suitable for low-resource envi ronments. Our study not only offers a much-needed dataset but also lays the groundwork for intelligent education tools that can reach Bangla-speaking learners across socio-

economic and geographical boundaries. Ultimately, the project aims to democratize AI learning by empowering students in their own language and promoting additional work on low-resource language technologies. Moreover, a RAG pipeline is also implemented to showcase the performance of our dataset.

1.2 Problem Statement

Most of the available AI-based learning solutions are designed targeting English-speaking students, and this creates a huge gap in the case of Bengali-medium students. Bengali-medium students lack access to AI-based tools that suit their curriculum and language needs. Even though English-language tools offer potent features like quizzes, practice tests, and smart feedback, these features are not available in Bangla and therefore prevent Bengali-speaking students from accessing self-learning resources. A core issue is that of the nonavailability of high-quality, annotated Bangla datasets that are specifically ones based on the National Curriculum and Textbook Board (NCTB) material used by Bangladesh's millions of students. Most datasets currently in use are too small, unstructured, or not aligned with the educational objectives of the NCTB syllabus. Thus, it is difficult to train stable NLP models that can generate proper questions or evaluate answers rightly. Without appropriate and representative training data, any AI system may be vulnerable to suboptimal performance as well as cultural or linguistic bias. Our work specifically addresses this constraint by highlighting curation and employing a Bangla NCTB-aligned dataset. By annotating and designing textbook content from the NCTB syllabus, we see the potential for building a solid foundation for downstream tasks such as automatic question generation and self-testing. This data-driven approach is essential to the development of a smarter study system for Bengali-medium students. Moreover, a tool that can effectively test understanding, provide valuable feedback, and bridge the education technology gap for Bangla-speaking students.

1.3 Research Objective

The primary aim of this research is to support Bangla language based education using automated solutions. To achieve this, the following research objectives have been set:

1. To develop and preprocess a large-scale Bengali language dataset focused on the education domain to support intelligent system development.
2. To help the learner by providing relevant questions and answers from contexts.
3. To build an intelligent system that will be centered around the Education sector which focuses on the Bangla Language.
4. To automatically generate multiple-choice questions along with some distractors (wrong answer choices) to help learners practice popular exam-style questions of educational institutions.

1.4 Thesis Structure

This thesis is organized into 6 chapters where each of the chapters focuses on addressing a specific segment of the research problem. These are:

- **Chapter 1** - This chapter provides the background of this study, defines the research problem and describes the objectives.
- **Chapter 2** - This chapter explores relevant ideas, models and previous researches to provide a foundation for our study.
- **Chapter 3** - This is the core chapter of our study where we introduce a new dataset based on NCTB Textbooks for Bangla Medium students. Dataset analysis and validation is performed accordingly.
- **Chapter 4** - This chapter discusses about finetuning the existing models that were used to test our dataset. We selected BanglaT5 as our base model, following the result analysis and evaluation of the finetunings.
- **Chapter 5** - In this chapter, a RAG pipeline based on the BanglaT5 Model is implemented. This pipeline is then used to formulate a question-answer generation application with the help of OpenAI API and Tesseract OCR.
- **Chapter 6** - This chapter concludes our entire study, mentions the limitations and provides recommendations for future work.

Chapter 2

Literature Review

2.1 Detailed Literature Review

"Alapi", a Bangla automated chat system, was proposed by Hasan et al.[15]. It can keep a long and meaningful conversation going by using its database. The system focuses on converting audio input into text and the response back into audio output after processing the input using AI. This system has been tested in both noisy and noise-free environments. It has its own Bangla library which helps the system to have a high accuracy rate for known data as well as be grammatical error-free during translation. Since the library is still in the development phase, there is a scope of improvement. Even though the paper does not clearly mention any research gap, one possible gap could be that the system lacks the ability to connect with other systems. Another gap could be the fact that the model is a simple three-layer FC model. The paper could look into the idea of using more advanced models in order to increase the system's accuracy and functionality.

This research paper by Aleedy et al.[3] describes how an interactive AI agent is built which will automatically generate conversations between a human and a computer. NLP and deep learning techniques consisting of Long Short-Term Memory, Convolution Neural Networks and Gated Recurrent Units have been used in order to predict relevant and automatic responses to the inquiries of the customers. The system has been tested and evaluated and the results show that out of all the models, the best one is the Long Short-Term Memory model in terms of performance. The goal of this paper is to increase the support of the customers by responding to the customers' inquiries as quickly and accurately as possible. The limited integration of outside information sources into the communication model is the primary research gap in this paper. This model focuses on past dialogues and overlooks other sources such as databases, internet, APIs etc. and this can result in limiting the AI agent's capability to respond with information that is correct or reliable.

According to Bommadi et al.[13], their research aimed to create an automatic learning assistant in Telugu that tests a person's knowledge and gives feedback to help them learn faster. This system mainly focuses on summarization, question generation, and answer evaluation. For the dataset, they used a Telugu stories dataset which was sourced from Kathalu Wordpress. However, they mentioned the challenges of the lack of a large Telugu dataset and the difficulty of creating rules to

cover all possible questions. According to the article, the system has limitations due to anaphora resolution. This means the system may struggle to identify the antecedent of a pronoun. Also, the question generation was tested in the domain of news articles which could produce grammatically correct answers. The document also mentions that the system may not perform well on complex sentences or sentences with words and phrases that require tags beyond the proposed set of rules. They would like to add more types of questions, improve the UD tagging model, and extend to other domains. Moreover, for further research, they want to test the system using minor details, named entities, and common nouns.

Alwahaishi [25] states that this study proposes a new model that overcomes limitations of previous dialogue systems by evaluating voice/video attributes and characteristics. This chatbot also contains emotional representation which is relate to biosignals and non-anthropomorphic expressive behavior as well. Additionally, the proposed model looks forward to having more personalized interactive agents through it. Reinforcement Learning algorithms have been proven helpful in optimizing statistical dialog managers in this paper too. These strategies have as well proved useful for dealing with problems of speech understanding. Furthermore, the system can serve as a prototype for different applications with diverse input and output emotional channels needed for them. Future work will involve further testing and refinement of the model as outlined in the paper. Moreover, they showed their interest in mental health or education where It could be applied someday if not now! Also, there will be an extension of this research into other areas such as adding new modalities to the model while considering ethics when using conversation interfaces.

In the studies, Khan et al. [16] presented an extensive comparative analysis of each default and custom component impacts. The chatbot system was constructed in Bangla to eases the communication between businesses and Bangali users. They gathered data which they later discovered was imbalanced. Therefore, they also had to address the imbalanced class problem. This Rasa framework is used in developing interactive agents for maintaining conversations in Bangla. Nevertheless, Rasa Framework has been ineffective when applied to Bangla because it is a low-resource language. This therefore necessitates the need of creating personalized components that are specific to Bengali language. Additionally, they indicated their desire for expanding and improving the overall quality of existing datasets in future work plans. In order to achieve this, they will gather more information and perform intense scrutiny and evaluation too; hence planning to have them included in their works as well. Their intentions here also include some additional personal components to be added into Rasa such as recent transformer models, multilingualism among others.

This paper by Gutiérrez [28] aims to investigate the role of Artificial Intelligence and Chat-bot-based learning based on open conversational tools such as ChatGPT. The study involves a handful of participants who engage with ChatGPT every day to learn the English language and assess their proficiency after analyzing it with questionnaires and interviews. The results of this research show how beneficial ChatGPT can be to all learners when applied in various learning contexts, making it a tool that complements and improves the field of research on technology-enhanced language learning. Moreover, The AIIA system uses OpenAI's GPT-3.5 model for

its NLP capabilities. The model is selected for its advanced text generation and few-shot learning abilities. The system architecture includes a NodeJS backend and uses text embeddings for efficient document retrieval and query response. The AIIA provides features such as personalized learning pathways, quiz and flashcard generation, and real-time feedback. However, this study has a limitation on the scale of conversations at beginner level and does not reflect how a complex pattern of questions could impact the performances of the AI chatbots. According to the author, further research on adaptive and interactive learning in AI-based chatbots is required to overcome these scenarios.

This chatbot by Dan et al. [26] implements LLM and converts the vast English Language Model to Chinese to do intelligent assessments of essays, Socratic teaching, and emotional support. It takes feedback based on human psychology and frontline teachers to assess the accuracy of the bot and fine-tunes the data for a deeper understanding of the Retrieval-Augmented open question answering, psychology-based dynamic emotional support, and stimulating critical thinking. EduChat addresses challenges in the educational applications of LLMs by pre-training on a vast corpus of educational books and diverse instructions to establish domain-specific knowledge. It further fine-tunes the model with high-quality, customized instructions to enhance educational functions. Additionally, EduChat uses retrieval-augmented techniques to incorporate real-time data, ensuring accurate and up-to-date responses The case studies for this claim to provide precise answers with relevant information and try to guide the students like a teacher. However, this chatbot is limited to the Chinese language only and it can assess only essays, and questions and provide intelligent emotional support for now. The researchers state that further study is needed to improve the LLM to integrate more educational features.

In this research, Jennifer et al. [29] propose a deep learning-based bilingual bot named BilinBot which incorporates both English and Bangla Language. It implements AI and NLP for conversational purposes and uses Google BERT to process the language. Also, LSTM and GRU are used to train the datasets efficiently. To implement LSTM, they tokenized the data, built the network, and trained multiple models with different optimizers. For the deep learning model, they added an Embedding layer and a dense output layer to make the activation function work. Moreover, to evaluate the predictability score, they used Python's ROUGE score model. On trained English datasets of ROUGE models, they got 73% - 96% accuracy and got 73% to 76% for the Bangla datasets. According to their analysis, the results lie ahead of most state of art deep learning-based chatbots. The limitations of this chatbot lie in the categories in which it can perform because of the limited dataset in the Bangla Language. Also, it can only be used in two languages but further research is needed to implement this in other languages as well to make it a multilingual chatbot.

According to Thoppilan et al. [24], LaMDA is presented as an advanced AI system for conversations. This is designed for natural, meaningful dialogues. The system is trained using extensive text data. Also, different sizes are available with larger models being more capable of handling complex interactions. In the paper, researchers also provided examples that show LaMDA can make sensible and engaging conver-

sations. Along with that, researchers have kept safety and accuracy as top priorities. Also, the responses of LaMDA are designed to be respectful and show factual correctness. LaMDA can be a great part of education and content recommendation-based apps as it can execute specific tasks if finely tuned. Additionally, since LaMDA is an AI that can hold conversations, it continuously tries to improve its quality and accuracy of replies. However, the paper also talks about several research gaps in LaMDA's development. This conversational system needs continuous fine-tuning for improvement which is expensive and time-consuming. Also, subjective human judgments can cause inconsistent fine-tuning. LaMDA has learned to use external knowledge better but still sometimes misrepresents facts. This mostly happens where better evaluation metrics are needed. Bias mitigation is also hard as it needs better detection techniques. Additionally, LaMDA's high energy consumption needs more sustainable practices.

The research paper by Hasnat, Chowdhury, and Khan [1]presents an OCR for Bangla language, BanglaOCR. It is the first open-source Optical Character Recognition (OCR) system for Bangla script. The system was developed by incorporating Tesseract's optical character recognition engine with CRBLP's Bangla script processing tools. The authors of the paper included a thorough methodology that includes training data preparation, image pre-processing, segmentation, and two-level post-processing for better accuracy. Their system could achieve up to 93 percent accuracy for clean printed documents. It also includes a GUI, spell checker, and support for both Windows and Linux environments. .

The research paper by Lewis et al. [10] gives an account of BART, a denoising autoencoder developed for pretraining sequence-to-sequence models. BART is designed on the basis of a Transformer architecture with one bi-directional encoder and an auto-regressive decoder. During pre-training, BART corrupts text in different ways such as random sentence shuffling or a new kind of in-filling scheme and then learns how to reconstruct that original text. It means that this approach combines the bidirectional aspect of both GPT and BERT into one generalized model. Consequently, the paper's authors managed to provide indicators which showed its performance better than previous methods on some framing benchmarks – GLUE, SQuAD, XSum with R-1 and BLEU showing notable improvements respectively. Its flexible noising and dual-graph interaction mechanisms make it highly effective across a broad range of NLP tasks. Despite this high-performance level manifested by BART there are still gaps in its research. To begin with, the methods used for pretraining corruption should be more diverse and task specific. In addition, when generating unsupported information it tends to hallucinate showing need for improved factual accuracy again as mentioned above while also noting that it hasn't yet focused on what is relevantly related to each other within ELI5 task types. Again we can generalize about this but then we can say that even if BART works so well in most hands-free situations where you simply turn it on and speak at your command; while at the same time being perfect during all sorts of laid-back conversations like these ones going back forth between two roommates; when subjected loosely defined duties such as those belonging to ELI5 where any formulating any request takes quite long sometimes resulting into getting irrelevant answers from AI becomes quite tedious thus calling for much higher adaptability levels across task

clusters.

The paper by Sajja et al.[30] introduces the Artificial Intelligence-Enabled Intelligent Assistant (AIIA) using a novel framework. This system creates scopes for customized and adaptive learning in higher education. This assistant incorporates advanced AI and Natural Language Processing (NLP) techniques to provide easily accessible information, and customized learning assistance with interactive attributes to the users. The framework connects with Learning Management Systems and analyzes the issues and future possibilities for AI-based teaching assistants. The system does not only limit its features to question-answering features. They have incorporated functionalities like flashcards, quizzes, homework/assignment evaluation, and summarization. However, this virtual assistant faces some limitations in extracting structured data from PDF, handling scanned copies, performing OCR functionality, incorporating multimedia resources, and implementing real-time video interaction features.

Islam et al. [5] address the planning and development of a Bangla virtual AI assistant, 'Adheetee' in this paper. This proposed model can follow and carry out a wide set of commands given in Bangla to smart devices. The commands categorized into basic and core commands use techniques like Named Entity Recognition, Keyword Extraction, and Cosine Similarity. Also, the algorithms mentioned here are implemented to help the virtual assistant use its own intelligence to respond to a variety of commands. Also, the system can give responses independently or it can use external APIs to find responses for unknown commands. However, the model is somewhat dependent on other systems since it mostly needs external APIs for unknown data. This limits the model's independence and has an impact on its performance. For further research, they want to work with more complex commands and decrease the dependency on external APIs. Also, they are developing software for both iOS and Android.

In this research, Rajpurkar et al. [2] deep dived into the Stanford Question Answering Dataset (SQuAD), which provides a large, high-quality dataset that helps with the development of training machine reading comprehension models. This large dataset is already preprocessed to convert the text into a format which will help machine reading. It specifically tokenizes the texts, converts the words to numerical representations and mostly does this thing where it creates features that can be utilized by the model. But here the researchers also pointed out that there is a significant gap between the performance of the machine models and humans. Also, SQuAD gets the answers from the parsing of the sentences, so it avoids multiple choice answers. But it adjusts the model's parameters so that the error is minimized between the model's predicted answers and the correct answers.

Lewis et al. [11] proposed the concept of Retrieval-Augmented Generation (RAG) to improve the factual accuracy and transparency of language models. Traditional parametric models like BERT or GPT bundle all world knowledge inside their parameters, which is impossible to update or verify their responses. RAG addresses this by combining a pre-trained sequence-to-sequence generator (e.g., BART) with an external non-parametric memory module which is a dense retriever over a large

corpus such as Wikipedia in this example. This enables the model to pick up related documents at runtime and base its generation on them, creating wiser and more updated outputs. The paper introduces two versions of RAG: RAG-Sequence, in which the extracted documents are represented by the model and an answer is produced using their common representation, and RAG-Token, in which dynamic token-wise conditioning is allowed on diverse retrieved passages. The retriever is implemented based on Dense Passage Retrieval (DPR), and the retriever and generator are trained end-to-end within a differentiable framework. The model is evaluated on a series of open-domain QA benchmarks including Natural Questions, WebQuestions, and CuratedTREC. RAG outperforms existing pipelines that use independent retriever-reader models and even achieves state-of-the-art results on some of the datasets at the time of publication. It is a good thing that RAG has the capability to return evidence citations, which increases explainability and user trust. In addition, in contrast to purely parametric models, the RAG updates are not retraining but rather corpus or retriever index updates. However, the architecture is not without some additional computational overhead from multi-passage encoding and real-time retrieval. Despite this, RAG paved the way for an enormous diversity of retrieval-augmented language systems, and this had a subsequent impact on later work in open-domain QA, summarization, and even coding assistance. It was a milestone for NLP from solely generative models to hybrid models that combined symbolic and neural reasoning.

Asai et al. [34] leverage the deficiencies of standard RAG models with the creation of Self-RAG, a novel retrieval-augmented architecture that allows the model to self-regulate its own retrieval and evaluation process. Unlike earlier practices that retrieve a fixed amount of documents and produce afterward, Self-RAG trains the model to decide when and how to retrieve and whether to re-evaluate or rewrite using internal critique tokens. This approach reflects a paradigm change from static augmentation towards agentic reasoning, where the language model recognizes its gaps in knowledge and inaccuracies in facts. Self-RAG inserts target tokens during training time: [RETRIEVE] tells the model to initiate a retrieval query, and [CRITIQUE] tokens allow it to critique the adequacy and salience of retrieved responses. Generation is recursive, and the model is allowed to revise its response based on later critiques and retrievals. This structure more closely simulates human problem-solving and avoids depending on earlier retrieval responses. Empirical evaluation on QA, fact verification, and summarization tasks shows that Self-RAG significantly surpasses static RAG baselines, ChatGPT, and LLaMA-2 systems, particularly in terms of factual accuracy and reference trust. On open-domain QA, it reduces hallucination rates and boosts BLEU scores by considerable margins. Furthermore, qualitative assessment proves that the critique step is helpful for identifying and revising wrong outputs, showing the ability of the system for self-correction. Self-RAG has only one limitation: higher inference cost for several steps and rounds of decoding incur latency. More sophisticated training data and supervision strategies are also required in order to train retrieval timing and critique effectively. Nonetheless, Self-RAG sets a new benchmark for adaptive retrieval-augmented models by offering a scalable self-regulated pipeline for use in applications that necessitate reliable and evidence-based responses.

Gao et al. [36] offer a timely systematic survey of Retrieval-Augmented Generation (RAG) techniques adapted to Large Language Models (LLMs). With the hasty proliferation of LLMs like GPT, PaLM, and LLaMA, and the increasing need to ground their output on external knowledge, this paper categorizes the evolving architecture, processes, and application of RAG in a unifying framework. The survey uses over 200 research papers and provides a comprehensive taxonomy of the components that make up an average RAG system. The authors divide RAG's development into three major phases: Naïve RAG, using fixed retriever + generator pipelines; Advanced RAG, using re-ranking, query reformulation, and late fusion; and Modular RAG, where retrieval and generation are used as flexible, learnable modules that can be controlled by agents or policies. The retrieval module is addressed in terms of sparse vs. dense paradigms (e.g., BM25 vs. DPR), while generation approaches include Fusion-in-Decoder, retrieval-augmented prompting, and fine-tuned encoders. One of the key contributions is the authors' incorporation of evaluation criteria and standards. They note the difficulty in measuring groundedness, factuality, latency, and memory efficiency for RAG systems and suggest using a standardized evaluation matrix. The paper also identifies challenges such as index staleness, retrieval noise, and security risks such as prompt injection via returned documents. In addition, the survey explores diverse trends such as multimodal RAG (text with images or audio), long-context RAG through memory compression, and tool-augmented RAG where LLMs use retrieval as part of a broader reasoning pipeline. The authors end by establishing a definition of open research directions such as life-long retrieval learning, retrieval for instruction-tuned models, and privacy-aware RAG. The paper is a seminal guide for researchers and practitioners who want to deploy or understand retrieval-augmented systems at scale. Its extensive coverage and insightful classification make it an essential guide to navigating the fast-evolving RAG space.

The study by Zulkarnain et al. [33] introduces bbOCR, a comprehensive and open-source OCR pipeline developed to digitize printed Bengali documents across various domains. The system aims to address the lack of integrated OCR solutions for Bengali, which remains underserved despite its global ranking as the sixth most spoken language. The pipeline incorporates several modules such as geometric and illumination correction, document layout analysis, line and word detection, word recognition and HTML reconstruction. These modules help to accurately convert scanned Bengali documents into structured and editable formats. A notable contribution of the paper is the introduction of a Bengali text recognition model named APSIS-Net alongside two synthetic training datasets named Bengali SynthTIGER and SynthINDIC and a large evaluation dataset named BCD3 which contains annotated samples from 9 domains. The evaluation uses both traditional OCR metrics and new system-level reconstruction metrics to demonstrate that bbOCR significantly outperforms Tesseract in both accuracy and layout preservation. However, while the pipeline is highly effective for printed texts, it currently does not support handwritten documents or documents containing tables. It also shows limitations when dealing with blurred images. Future plans include expanding to multilingual capabilities and integrating table recognition. This paper highlights a critical advancement in Bengali document digitization but also signals the need for further improvements to handle more complex document types and handwritten scripts.

The paper by Du et al. [9] introduces PP-OCR, an ultra-lightweight Optical Character Recognition (OCR) system designed to either enhance the model ability or reduce the model size. The system is divided into three main parts: text detection using Differentiable Binarization (DB), detected box rectification, and CRNN-based text recognition. Each module is optimized using a variety of enhancement and slimming techniques such as lightweight backbone, cosine learning rate decay, FPGM pruning, PACT quantization etc. These strategies collectively reduce the total model size to as low as 3.5M for Chinese character recognition and 2.8M for alphanumeric symbol recognition without compromising much on accuracy. The system is trained on massive datasets and is validated across multiple languages such as French, Korean, Japanese and German. One key strength of this paper is its extensive ablation studies that show the effectiveness of each optimization strategy. However, the system lags behind large-scale models in terms of F-score. Nevertheless, the trade-off offers significant practical advantages for deployment on mobile or embedded devices. Future improvements could target the quantization of more complex components like LSTMs and expanding training data diversity for better multilingual generalization.

Sequential Question Generation (SQG) is addressed by Chai Wan [8] in their pa per. This helps to minimize the cascading errors as well as poor context capture that usually arises in traditional approaches. They propose a semi-autoregressive approach that groups and generates questions concurrently which reduces errors and speeds up the process. Furthermore, they have implemented dual-graph inter action for better context and relevance. An answer-aware attention mechanism was also used alongside a coarse-to-fine generation scenario. Therefore, it considerably enhances question quality and relevance, thus establishing new standards for SQG research based on a novel SQG-specific dataset. Current Sequential Question Gen eration (SQG) research faces numerous challenges including error cascades, limited context capture as well as low-quality datasets leading to lower quality and coherence of generated questions. More studies are required though to enhance question informativeness even if they introduce a semi-autoregressive model; develop more efficient clustering and generation methods; improve coverage of images and knowl edge graphs among other things.

This paper by Narayan et al. [38] proposes a Retrieval-Augmented Generation (RAG) framework integrated with the Ample LMS platform to automate the creation of multiple-choice questions from PDF-based educational documents. The authors aim to overcome the inefficiencies of manual and template-based question generation methods, which are often repetitive and inflexible. Their system extracts content from documents, embeds it into a FAISS vector store for efficient retrieval and then uses OpenAI's GPT-4 model to generate questions with four answer options. The study evaluates the output of RAG-powered models, such as ChatGPT, Gemini and Perplexity, using similarity measures like BERT, XLNet and CodeBERT, comparing them against human-created questions. ChatGPT performed the best, with the highest similarity scores and the most natural-sounding questions. One significant feature is the use of a semantic similarity check to avoid duplicate questions. While the system efficiently automates question generation and integrates seamlessly into LMS environments, it still faces limitations like high computational requirements and dependency on the quality of retrieved content. Future work

includes expanding the system to support open-ended and adaptive personalized assessments and increasing subject domain diversity. This research highlights the practical potential of RAG-based AI systems in streamlining educational content creation.

Basak et al.[37] present an effective Retrieval-Augmented Generation (RAG) pipeline specifically designed for the Bengali language. It addresses the limitations in open-domain question answering for low-resource languages. The study introduces a fine-tuned version of the ColBERT retriever on the Bengali SQuAD dataset named SQuAD BN using various embedding models such as BanglaBERT, Bangla-BERT-base and Bengali-sentence-similarity-SBERT etc. The authors benchmarked both zero-shot and fine-tuned models and found significant performance improvements in Hit Rate and MRR when using the fine-tuned ColBERT approach. Furthermore, they integrate their Bengali retriever with a Bengali LLM, BN-RAG-LLaMA3-8b, and observe an 8% improvement in F1 score over the LLM without RAG. The thesis emphasizes how integrating retrieval boosts the contextual awareness of generative models, improving accuracy in Bengali question answering tasks. However, the authors note that due to the dataset's short ground truth answers and the model's tendency to generate longer responses, the F1 scores remain relatively low. This research highlights the potential of tailored RAG pipelines in enhancing NLP capabilities for underrepresented languages and suggests further refinement in answer length control and model tuning for better performance.

Chapter 3

Data Collection and Analysis

3.1 Data Collection Procedure

The data collection process is designed in a way to develop a thorough and high-quality dataset of Bangla question-answer pairs. The main goal of this was to ensure that the dataset would be relevant and technically robust and would be fit for any tasks related to natural language processing (NLP) such as short questions or multiple choice questions or information retrieval in Bangla.

3.1.1 Primary Data Source and Extraction

Primary source used for data collection was the Bangla version of the NCTB textbook "Bangladesh and Global Studies" of class 9-10. This book was chosen based on the fact that it provides well-structured, factual content that gives better results when generating question answers within a clear contextual boundary. Also, students struggle with this particular subject as well.

For extracting the data, the OCR method was used. By using this method, the texts are broken down into meaningful contexts which have the constraint of less than 1000 characters. The purpose of this constraint was to make the context manageable for generating question and answer extraction. The architecture that is followed for the dataset is the SQuAD (Stanford Question Answering Dataset) format.

```
[4] ds.head()
```

	title	context	question	answer	chapter	answer_start
0	জাতিসংঘ ও বাংলাদেশ	মানবাধিকার ছাড়া কোনো মানুষ পূর্ণাঙ্গভাবে বিকশ...	মানবাধিকার কি?	মানুষের বেচে থাকার জন্য প্রয়োজনীয় কিছু সুযোগ...	9	86
1	জাতিসংঘ ও বাংলাদেশ	মানবাধিকার ছাড়া কোনো মানুষ পূর্ণাঙ্গভাবে বিকশ...	পৃথিবীতে কয়টি বড় যুদ্ধ হয়েছে?	দুটি	9	165
2	জাতিসংঘ ও বাংলাদেশ	মানবাধিকার ছাড়া কোনো মানুষ পূর্ণাঙ্গভাবে বিকশ...	প্রথম বিশ্বযুদ্ধ কবে শুরু হয়েছিল?	১৯১৪ সালে	9	316
3	জাতিসংঘ ও বাংলাদেশ	মানবাধিকার ছাড়া কোনো মানুষ পূর্ণাঙ্গভাবে বিকশ...	প্রথম বিশ্বযুদ্ধের পর কোন আন্তর্জাতিক প্রতিষ্ঠ...	লীগ অব নেশনস	9	468
4	জাতিসংঘ ও বাংলাদেশ	মানবাধিকার ছাড়া কোনো মানুষ পূর্ণাঙ্গভাবে বিকশ...	দ্বিতীয় বিশ্বযুদ্ধ কেমন ছিল?	আরও কলঙ্কিত ও বিভীষিকাময়	9	649

Figure 3.1: Start of Dataset

```
[7] ds.tail()
```

	title	context	question	answer	chapter	answer_start
2253	বাংলাদেশের তথ্য অধিকার আইন	আবেদনকারী আপিল কর্তৃপক্ষের নিকট সুবিচার না পেল...	তথ্য সরবরাহের ক্ষেত্রে দায়িত্বপ্রাপ্ত কর্মকর্...	সংরক্ষক	6	253
2254	বাংলাদেশের তথ্য অধিকার আইন	আবেদনকারী আপিল কর্তৃপক্ষের নিকট সুবিচার না পেল...	আবেদন প্রক্রিয়া সম্পর্কে জনগণকে কে অবহিত করবেন?	দায়িত্বপ্রাপ্ত কর্মকর্তা	6	218
2255	বাংলাদেশের তথ্য অধিকার আইন	তথ্য প্রাপ্তির ক্ষেত্রে আমাদের তথ্য অধিকার আইন...	তথ্য অধিকার আইনের মাধ্যমে কী বৃদ্ধি পাবে?	প্রতিটি সংস্থার কার্যক্রমে স্বচ্ছতা ও জবাবদিহিতা	6	165
2256	বাংলাদেশের তথ্য অধিকার আইন	তথ্য প্রাপ্তির ক্ষেত্রে আমাদের তথ্য অধিকার আইন...	তথ্য অধিকার আইনের মাধ্যমে কী নির্মূল করা সম্ভব...	দুর্নীতি	6	253
2257	বাংলাদেশের তথ্য অধিকার আইন	তথ্য প্রাপ্তির ক্ষেত্রে আমাদের তথ্য অধিকার আইন...	তথ্য অধিকার আইন বাস্তবায়নে কী মজবুত হবে?	গণতন্ত্র	6	387

Figure 3.2: End of Dataset

3.1.2 Contextual Analysis and Question Generation

Each context was carefully reviewed to identify key information that could be posed as a question. The contexts that have more facts or in depth concepts were flagged and different types of questions were generated from the same context to get varieties of question-answers. Our main focus was to create simple, easy-to-read answer questions as this was based on Secondary level students' understanding. The answers to the generated questions were formulated directly from the context, making sure that the answer was detectable from the text. Again, the answers were double-checked for accuracy. Also, we tried to be careful enough to make sure that the answers were precise and directly related to the information from the source material.

3.1.3 Indexing and Structuring

Moreover, all the answers were indexed to the location in the original text. The index was generated to ensure that the answer's starting point in the text was properly documented, which allowed for traceability and accuracy in referencing. After reviewing and rechecking the question-answer pairs, the raw dataset was structured in a tabular format. This raw dataset was combined into a 2,258 line csv file. Each entry consisted of following fields:

- **"ID"** - Unique identifier for each entry
- **"Title"** - Title of the chapter the context is taken from
- **"Context"** - The passage from which the questions and answers will derive
- **"Question"** - The generated question
- **"Answer"** - The answer to the particular question
- **"Chapter"** - The chapter in the textbook
- **"Answer-start"** - Character-level index of the answer's starting point in the context.

ID	Title	Context	Question	Answer	Chapter	Answer_start

Table 3.1: Table of collected Dataset Format

This structured format ensured the integrity and accuracy of the data, making it suitable for subsequent stages of the research such as analysis, evaluation, and potential model training in natural language processing tasks involving the language Bangla.

3.1.4 Expansion with Literature-Based Dataset

Here we focused on expanding the dataset by collecting data from the secondary Bangla literature book "Maddhomik Bangla Shahitto" and the data is extracted from the literature section only. Bangla is already very hard in the secondary level as there are many questions and answers that are very unpredictable. So, we decided to make the context more concise and extensive which allowed us to get more short question answers. This one follows the RAG (Retrieval-Augmented Generation) pipeline closely. Collecting more extensive contexts allowed the model to create a broader spectrum of question types such as multiple choice questions and it's known how important MCQs are for secondary level. Even though the context is extensive, the SQuAD style is still followed, comprising 'ID', 'Title', 'Context', 'Question', 'Answer', 'Chapter' and "answer-start"- which made sure that consistency and compatibility was maintained. Because of following the same structure, it helped us merge both the datasets which are rich in educational diversity. An additional 1,215 pairs were generated from supplementary Bangla literature content and merged with the initial dataset, resulting in a combined total of 3,473 entries.

3.2 Data Analysis

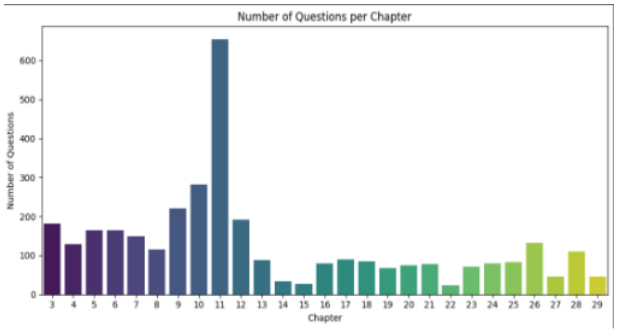


Figure 3.3: Chapter Distribution in Dataset

This provides us the idea for "Number of questions per Chapter" where x-axis represents "Chapter", while y-axis represents "Number of Questions" associated with each chapter. The number of questions differs from chapter to chapter like some have less than 100 questions but some have more than 300 questions. But mostly the number of questions has a range of 200-300. The highest number of data is found from chapter 11 which is from the Bangladesh and Global Studies book and more than 400 questions derived from that single chapter.

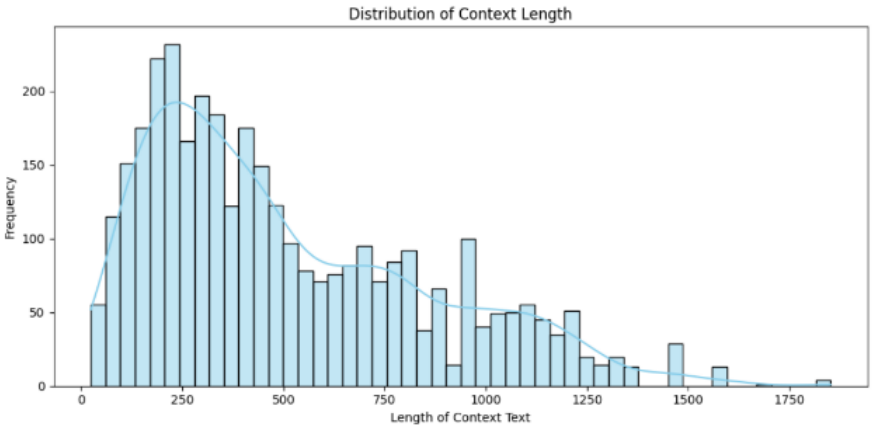


Figure 3.4: Context vs Frequency Length in Dataset

This histogram indicates the distribution of context length which helps us see how frequently different lengths of context occur in the dataset. The x-axis indicates the 'length of context text' while the y-axis represents the frequency of the texts. The length of the context text is likely in words, tokens or characters. The range of x-axis is 0-1750. The shortest texts are near 0 and the longest is near 1750. We see in the area of 250-500 where the frequency count is around 200.

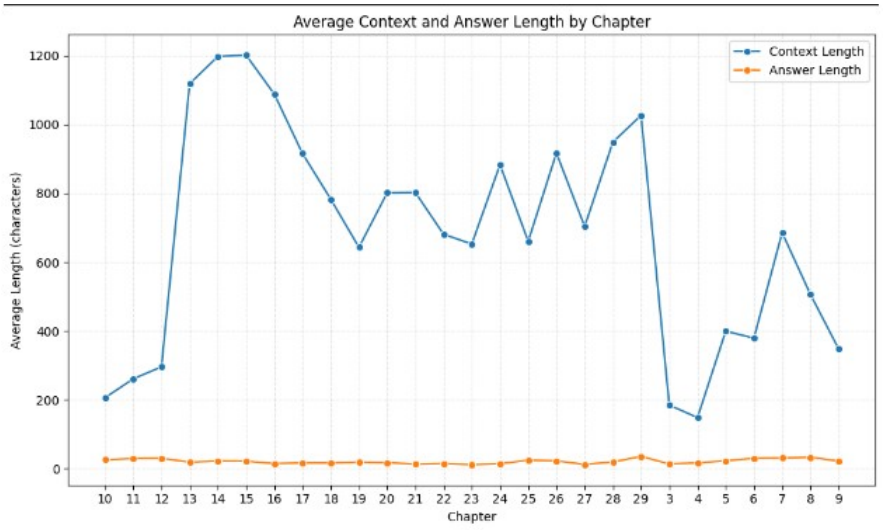


Figure 3.5: Context vs Answer Length in Dataset

This one represents the average context and answer length by chapter where the x-axis represents chapters and y-axis represents average length (characters) as the answer start is based on characters. The context length is in some cases 1200 characters long where the answer lengths never even crossed more than 20 characters. The context length is not less than 200 characters on the other hand.

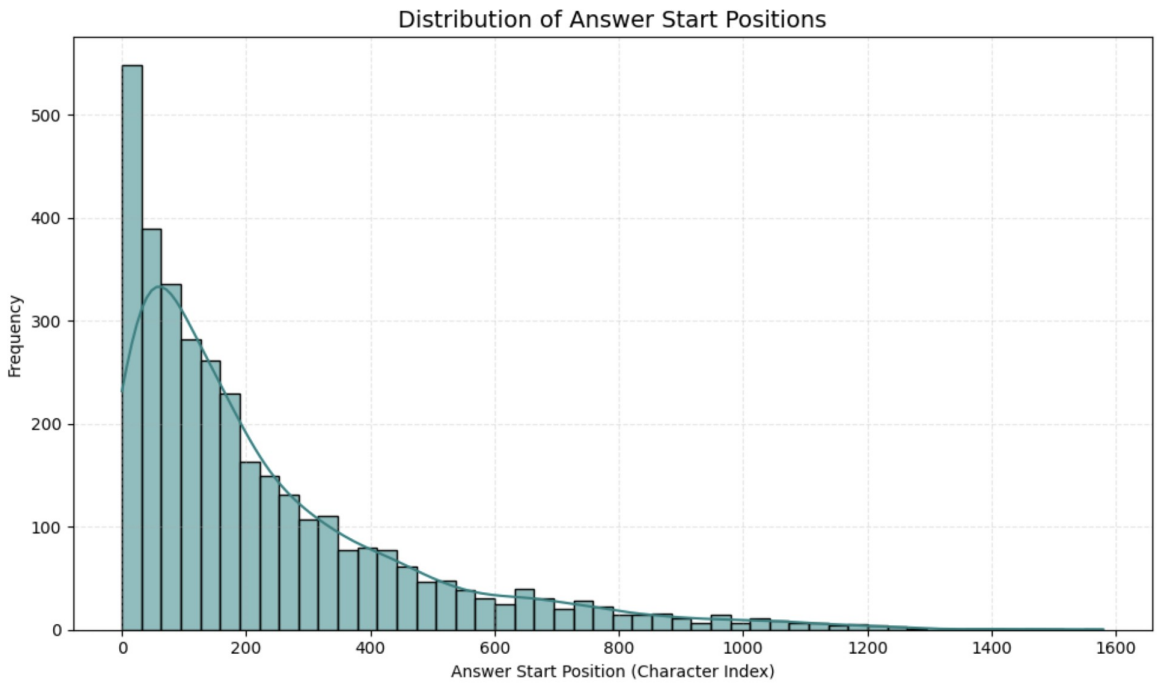


Figure 3.6: Distribution of Answer Start Position

The bar chart represents the distribution of answer start position where the x-axis shows answer start position (character index wise) and the y-axis shows the frequency. This is a right skewed chart where 0 has a frequency as high as 500. There are very few answer start positions in a range between 800 and 1600.

3.3 Dataset Validation and Cleaning

For ensuring integrity of the dataset in different dimensions - structural completeness, schema conformance, language consistency, duplicate entry and text length analysis - before creating a cleaned version suitable for machine learning applications.

3.3.1 Initial Structural Analysis

After finishing the dataset, we found that it consists of 3,473 rows and 6 columns which had the followings:

1. **Title:** Name of the chapter which is a bangla phrase/name representing the topic
2. **Context:** A longer bangla passage containing possible questions and answers
3. **Question:** Bangla sentence which represents a question
4. **Answer:** the answer span, which has to have relevance to the context

5. **Chapter:** An integer referring to a chapter
6. **Answer-start:** The index in context where the answer begins.

The first test revealed that there were no missing or null values in any of the columns. But somehow the answer start part was saved as a string/object instead of being an integer. That's why it was marked for modification so that there wouldn't be any alignment validation issue.

3.3.2 Schema Validation

All schema tests were manually done using pandas. For validation, these were the rules that was followed:

1. There were no failing rows for title length not between 3 and 60.
2. There were 56 rows that had context strings that were too short to support meaningful question answering.
3. The questions length are between 4 and 120.
4. 26 rows had answer fields that are either too short or very long. So answer length is not between 1 and 100.
5. There were 7 rows that had non-integer answer values which broke the span annotation logic.

Checking these rules created the core criteria for filtering during the cleaning process.

3.3.3 Textual Content and Language Consistency

When analysing there are more insights such as all text columns were confirmed to be Bangla using language detection, 1 duplicate row was found which was removed later, 1404 context entries were longer than 1000 characters and 143 answer entries were really short like less than 4 characters.

3.3.4 Dataset Cleaning

After checking all the validation rules, a clean version of the dataset was made applying some filters which were:

- Removed the rows where the context consists of fewer than 60 characters.
- If the answer length was not between 1 and 100, then those were removed.
- Non-numeric or missing answer values were removed.
- By applying these filters, a total of 88 rows were eliminated.

- In the end, the final dataset came down to 3,385 rows.

By following all the systematic rules and regulations, a reliable resource for building Bangla questions and answering was made. Issues related to type mismatches. Invalid spans and length outliers had been resolved.

Chapter 4

Models Architecture and Results Analysis

4.1 Models

4.1.1 Bangla T5

BanglaT5[20] model is made by following a transformer-based encoder-decoder architecture. This architecture is based on the model of Text-to-Text Transfer Transformer. The processing of the encoder starts when a sentence enters as an input. Then it processes through multiple levels of self attention, converts to latent representation and proceeds to feed forward networks. On the other hand, a decoder is used for generating the output sequence. It generates the output using the encoder's latent representation and tokens. Also, to focus on different portions of the sequence, the model uses a multi-head self-attention in encoder and decoder.

Since the words of a sentence follow a sequential order and BanglaT5 model doesn't have the capacity to understand the sequence itself. That's why positional information or encoding are added to the input encodings before each word of the sequence. As a result, it carries the order of that particular word along with it. The BanglaT5 model generally works with 12 encoder and decoder layers. Each of the layers have 12 attention heads and also has a feed-forward network with 3072 units.

The model is trained using Bangla specific sentencepiece tokenizer which helps to breakdown Bangla texts into smaller units or tokens. Bangla language has a complex combination of characters and that's why the tokenizer splits the text into meaningful sub-parts. This helps the model to handle unseen texts.

So if the model hasn't seen this word while training, it will still be able to handle this text by detecting the tokens. Also, BanglaT5 has almost 223 million parameters so it can easily handle complex Bengali text generation tasks.

T5 Tokenizer

For BanglaT5 model, tokenizers work similarly like other transformer-based models. It uses a Sentence-Piece Model. It is a type of subword tokenizer model, which splits the words into smaller units called subwords. This way it becomes significantly

efficient in terms of handling rare and unseen words.This method is based on Byte-Pair Encoding (BPE), which allows the tokenizers to evaluate the whole Bangla words as single tokens. It also makes sure that the single tokens break less frequent words into parts. Special tokens like ¡pad¿, ¡unk¿, ¡s¿ and ¡/s¿ are used for padding, unknown words, and sequence boundaries.

4.1.2 BanglaGPT

BanglaGPT[31] follows a transformer-based architecture, specially optimized for Bangla-related services. It is built on a similar structure to GPT (Generative Pre-trained Transformer), where the model is designed to predict the next word in sequence in order to handle tasks such as text generation, completion and comprehension.

The model only works through the decoder architecture, that is, it focuses on processing tokens in order from left to right. It uses multihead auto-attentive mechanisms, which help consider different parts of the input at each decoding step. These focused approaches enable the model to capture long-term dependencies and contextual meanings in Bangla.

Bangla GPT also uses a positional coding system to preserve word order, which is important for contextual understanding in Bangla, where word order affects meaning. Provided a Bangla-specific corpus and phrase fragment tokenizer Bangla scripts tokenize into sub -word groups. This helps to deal with difficult and previously unseen words, allowing them to perform well in a wide range of Bangla language tasks.

The model's ability to produce smooth and consistent Bangla text comes from its extensive preliminary training on large Bangla datasets. BanglaGPT has hundreds of thousands of parameters, enabling it to perform tasks such as text generation, collecting questions and answering them in Bangla.

GPT Tokenizer

The BanglaGPT model uses subword tokenization. This is based on the BPE which proves to be very efficient in handling Bangla text. It handles Bangla's complex script like compound characters and diacritic marks by evaluating them as individual tokens and splitting them into subwords.

4.1.3 XLM-RoBERTa

XLM-RoBERTa[4] is a multilingual model based on the RoBERTa framework, which is itself an improved version of BERT (Bidirectional Encoder Representations from Transformers). Previously trained on writing in several languages including Bangla using masked language (MLM) objectives. In MLM, it is used to mask some words in a sentence, and the model is trained to predict these masked words based on context.

The model is a multi-layered transformer-based encoder that focuses on itself, where it processes input sequences in both directions, taking into account dependencies between words rather than sorting them. This bidirectional characteristic is needed for understanding complex syntactic relations in Bengali sentences.

XLM-RoBERTa recognizes the structure of words in Bangla writing and applies positional rules to ensure that it maintains the smooth flow of meaning. The example uses a multilingual fragment tokenizer that decomposes the Bangla text into subwords, which can use words not found in the language or unusual linguistic structures With 55 crore criteria, XLM-RoBERTa is well suited for various Bengali specific tasks including text segmentation, named entity recognition and bilingual tasks Its multilingual characteristics also enable the model to perform multilingual tasks, such as translation between Bangla and other languages or understanding multilingual contexts.

XLM RoBERTa Tokenizer

The XLM-RoBERTa model uses subword models like Sentence-Piece Model which is based on BPE. This tokenizer works by breaking down the words into smaller subwords and effectively handles the complex compound words as well. After tokenization each subword is mapped into unique individual ID extracts from the vocabulary. In the time of generating texts, the tokenizer decodes back to the human level texts so that we can read the words or the sentence.

4.2 Model Finetuning and Evaluation

4.2.1 Evaluation Metrics

F1 Score

To evaluate the machine learning models, F1 score is widely used as the performance metric. It is particularly specialized in the classification tasks. It is basically the harmonic mean of precision and recall. The formula for F1 score is:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.1}$$

This metric is very much helpful in the field of Natural Language Processing (NLP) by giving us the idea of measurement of accuracy and identifying relevant instances.

Exact Match

Exact Match is a common metric used in the field of NLP. It is especially used in the question answering models or text generation models, by evaluating the accuracy of the model predictions. It is calculated by the ratio of the numbers of the correct predictions with the number of total predictions. It is very helpful in the cases where precise answers are very crucial. It takes into account partial matches, which is helpful in measuring strict ways of correctness, which makes it stand out from the other metrics like F1 score.

4.2.2 Model Finetuning

Initially, we have selected 3 models which are very efficient for the Natural Language Processing tasks that we are going to implement for the Bangla Language. These models are BanglaT5, BanglaGPT and XLM-RoBERTa. These models are well structured to do the tasks such as summarization, question-answering and text2text generation via their tokenization tools. All of these models are pre-trained in Bangla to do these tasks. Now we have to fine tune these models using our own dataset that we have created from scratch from the 'Bangladesh and Global Studies' book in SQAuD 2.0 format. For the evaluation of fine-tuning these models, we are going to use F1 and Exact Match scores as metrics.

4.2.3 Finetuning BanglaT5

At first, we fine tuned the BanglaT5 model using the 'Epoch' evaluation method. We have divided the dataset into 'Training' and 'Evaluation' with a split of 0.1. We used Adam optimizer for the process with weight decay of 0.01 and learning rate of 0.0001. The training and evaluation data were divided into 4 batches per device for the training function. The finetuning process ran for 10 epochs and it took a total time of almost 2 hours to complete with our GPU. Here are the results we got from our finetuning process with visualization.

Epoch	Training Loss	Validation Loss
1	3.55005	1.36557
2	1.59984	0.83164
3	1.08870	0.62620
4	0.86933	0.41478
5	0.67437	0.25177
6	0.55893	0.24301
7	0.46986	0.17411
8	0.42095	0.21285
9	0.36137	0.15299
10	0.29431	0.11133

Table 4.1: Training and Validation Loss over Epochs for Fine-tuned BanglaT5

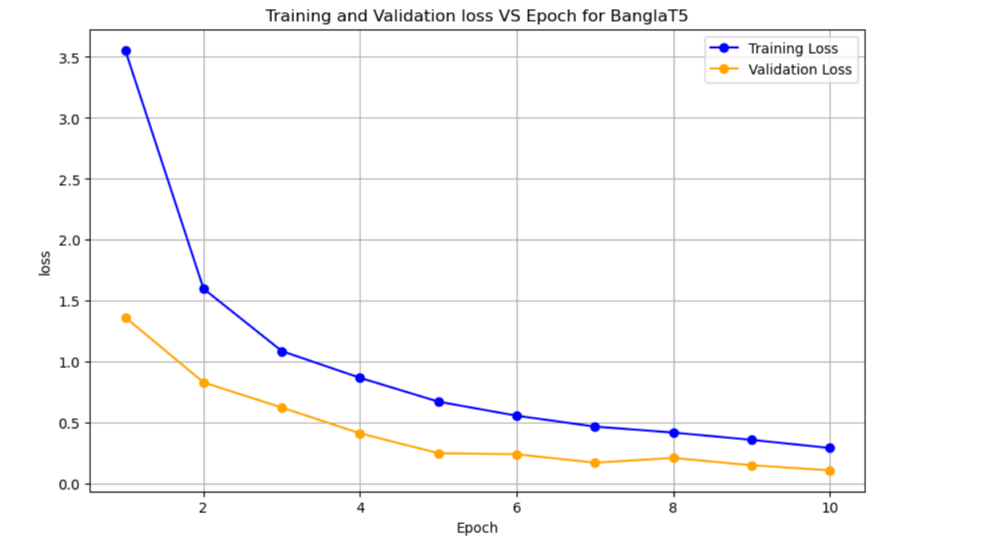


Figure 4.1: Fine-tuned BanglaT5

This shows that the training data loss and validation data loss for our dataset was initially higher but it gradually decreased to a stable state over the course of the fine tuning process. At the 10th epoch it was giving the minimal amount of data loss indicating that the model was fitting properly with our dataset.

Then, we ran a test for question-answering with the fine tuned BanglaT5 model alongside F1 and Exact Match score.

```
context1 = data.iloc[4]['context']
question1 = data.iloc[4]['question']
answer1 = data.iloc[4]['answer']
predict_answer(context1,question1,answer1)
```

```
Context:
 যুদ্ধ কখনও জাতিতে জাতিতে সংকট নিরসনের পথ হতে পারে না।


Question:
 যুদ্ধ কি সংকট নিরসনের উপায় হতে পারে?
{'Reference Answer: ': 'না',
 'Predicted Answer: ': 'না',
 'BLEU Score: ': {'google_bleu': 1.0}}
```

Figure 4.2: Example Question Answer Set in Fine-tuned BanglaT5

Here, we got 87.82% score in F1 metrics 78.23% score in Exact Match metrics using the test split of our dataset.

4.2.4 Finetuning BanglaGPT

Secondly, we fine tuned the BanglaGPT model using the 'Epoch' evaluation method. We have divided the dataset into 'Training' and 'Evaluation' with a split of 0.1. We used Adam optimizer for the process with weight decay of 0.01 and learning rate of 2e-5. The training and evaluation data were divided into 4 batches per device for

the training function. The finetuning process ran for 10 epochs and it took a total time of almost 3 hours to complete with our GPU. Here are the results we got from our finetuning process with visualization.

Epoch	Training Loss	Validation Loss
1	4.86232	2.51237
2	4.30125	2.10184
3	3.74019	1.68951
4	3.18120	1.37002
5	2.72645	1.12066
6	2.28178	0.90412
7	1.84563	0.71143
8	1.47892	0.58056
9	1.21987	0.49321
10	1.04522	0.38947

Table 4.2: Training and Validation Loss over Epochs for Fine-tuned BanglaGPT

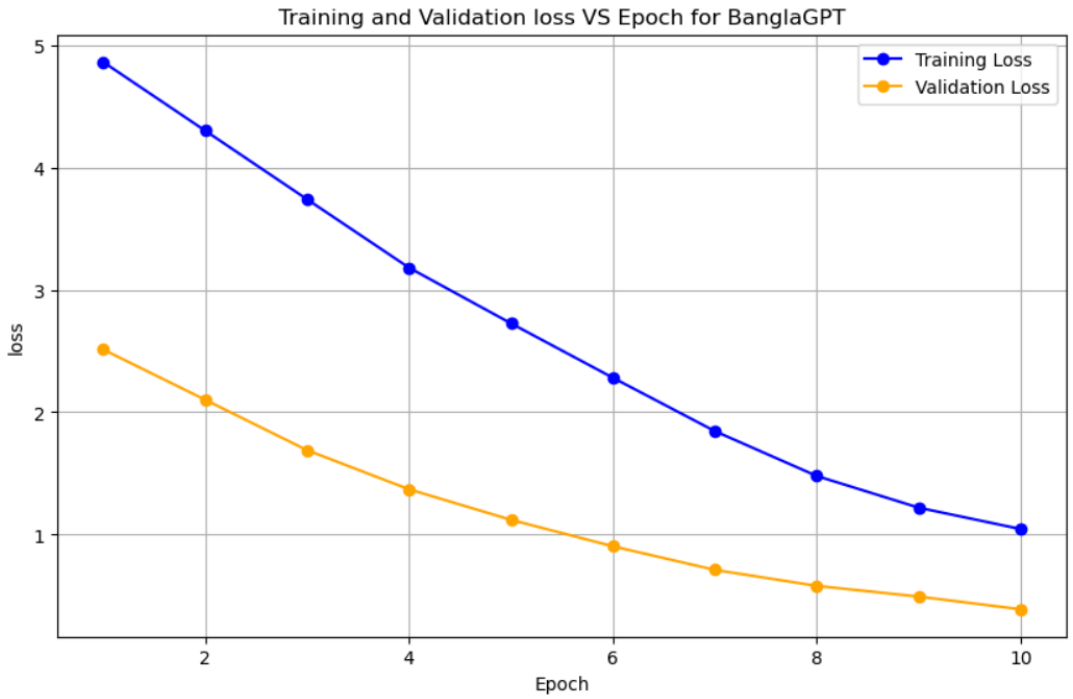


Figure 4.3: Fine-tuned BanglaGPT

This data shows that the starting training and evaluation data loss while fine tuning this model was quite high but it gradually decreased till the 10th epoch but the training loss was still in a higher state. Also, the evaluation metrics score for this model was 74.19% for F1 and 67.53% for Exact Match. Here is an example of a question-answer function.

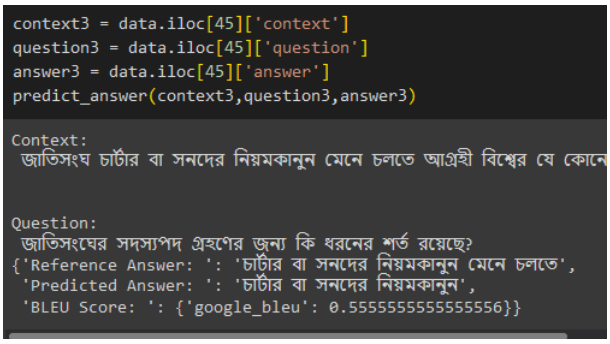


Figure 4.4: Example Question Answer Set in Fine-tuned BanglaGPT

4.2.5 Finetuning XLM-RoBERTa

We have fine tuned the XLM RoBERTa also using the 'Epoch' evaluation method. For this, we split our dataset into Training 80% and Evaluation 20%. The process was executed with weight decay of 0.01 and learning rate of 2e-5. Total save limits was 2 and the process ran for 10 epochs. The process took almost 3 hours to complete with our GPU. The results are given below:

Epoch	Training Loss	Validation Loss
1	1.45231	1.05534
2	1.28347	0.91313
3	1.06124	0.67847
4	0.76570	0.42579
5	0.73433	0.44322
6	0.70116	0.43247
7	0.65159	0.45681
8	0.62955	0.44391
9	0.61238	0.43459
10	0.60124	0.42568

Table 4.3: Training and Validation Loss over Epochs for Fine-tuned XLM-RoBERTa

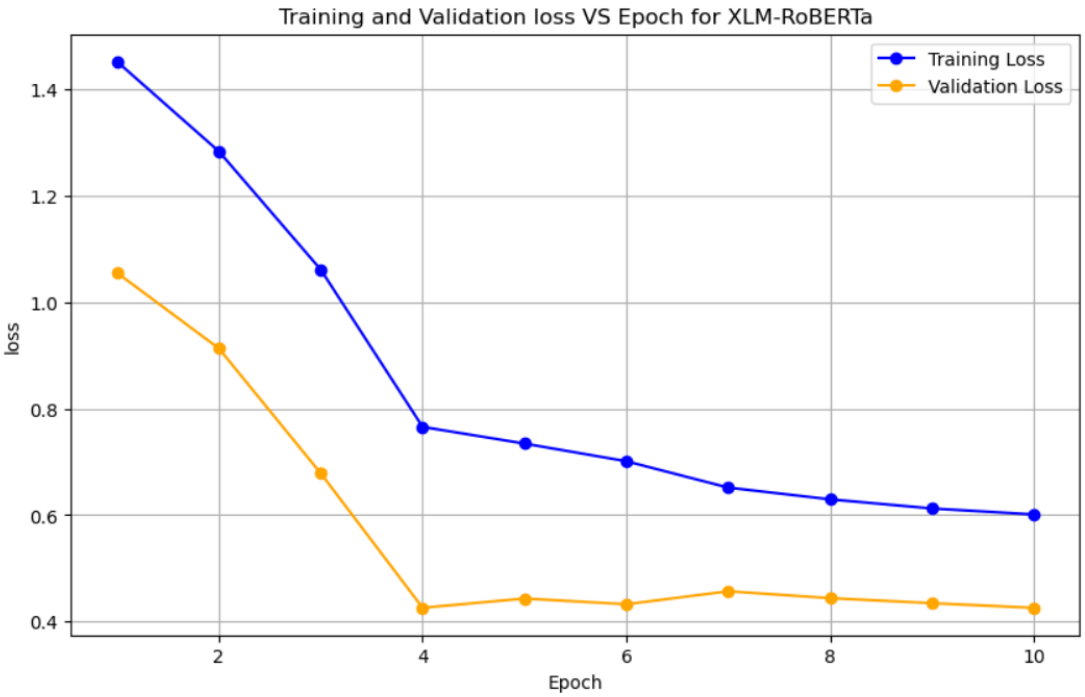


Figure 4.5: Fine-tuned XLM-RoBERTa

From this we can get to know that, the training loss and validation loss was initially much higher. Over the epochs, the both losses decreased almost like a straight slope. But after the 4th epoch the training loss was still decreasing but with the lower slope. But in the case of validation slope, we can see the loss was not stable. After the 4th epoch, the validation loss got slightly increased till the 5th epoch, then it was slightly decreased till the 6th epoch. But after that, it was again slightly increased in the 7th epoch, after that it gradually decreased but with a very little slope. After the 10th epoch, the training loss and validation loss was in a lower state. But it was still higher than the BanglaT5 model. The F1 score for this model is 81.03% and Exact Match is 73.92%. We have also tasted with the question answering and here is the result:

```
context2 = data.iloc[15]['context']
question2 = data.iloc[15]['question']
answer2 = data.iloc[15]['answer']
predict_answer(context2,question2,answer2)
```

```
Context:
 জাতিসংঘ সৃষ্টির পটভূমিপ্রতিটি দেশ হারায় তাদের কর্মক্ষম যুবসম্প্র


Question:
 জাতিসংঘ আনুষ্ঠানিকভাবে কবে আত্মপ্রকাশ করে?
{'Reference Answer: ': '১৯৪৫ সালের ২৪শে অক্টোবর',
 'Predicted Answer: ': '১৯৪৫ সালের ২৪শে অক্টোবর',
 'BLEU Score: ': {'google_bleu': 1.0}}
```

Figure 4.6: Example Question Answer Set in Fine-tuned XLM-RoBERTa

4.3 Result Analysis and Model Selection

Based on the evaluation metrics, here BanglaT5 stands the top for the Bangla intelligent system we are trying to build, as it showed the best performance in average low training loss (0.8742) and high F1 (87.82%). It also stands on higher ground in the Exact Match (78.63%). Without finetune, BanglaT5 did not show any strong position. But once it was fine tuned, BanglaT5 showed its strongest position; Exact Match (78.23%) and F1 score (87.82%). On the other hand, BanglaGPT has a larger parameter count (1.2B) and it showed

Models	**Params**	**EM**	**F1**
mBERT	180M	67.12	72.64
XLM-R (base)	270M	68.09	74.27
XLM-R (large)	550M	73.15	79.06
sahajBERT	18M	65.48	70.69
BanglishBERT	110M	72.43	78.40
BanglaBERT	110M	72.63	79.34
BanglaT5	247M	68.50	74.80
BanglaGPT	1.2B	65.23	73.72
Fine-tuned BanglaT5	**247M**	**78.23**	**87.82**
Fine-tuned XLM-R (Large)	**550M**	**73.92**	**81.03**
Fine-tuned BanglaGPT	**1.2B**	**67.53**	**74.19**

Table 4.4: Performance of various models

higher average training loss (2.6683) and lower Exact Match (67.53) and F1 score (74.19), which does not stand very strong in the line. So, it might not generalize as the other models. But it is suitable for large-scale data processing and contextual understanding. Then again, XLM-RoBERTa exhibited much better performance than BanglaGPT with moderately less training loss and higher Exact Match (73.92) and F1 score (81.03), which shows it falls behind the fine tuned BanglaT5. This model can be a good option, if we require higher accuracy without a relatively increased training loss.

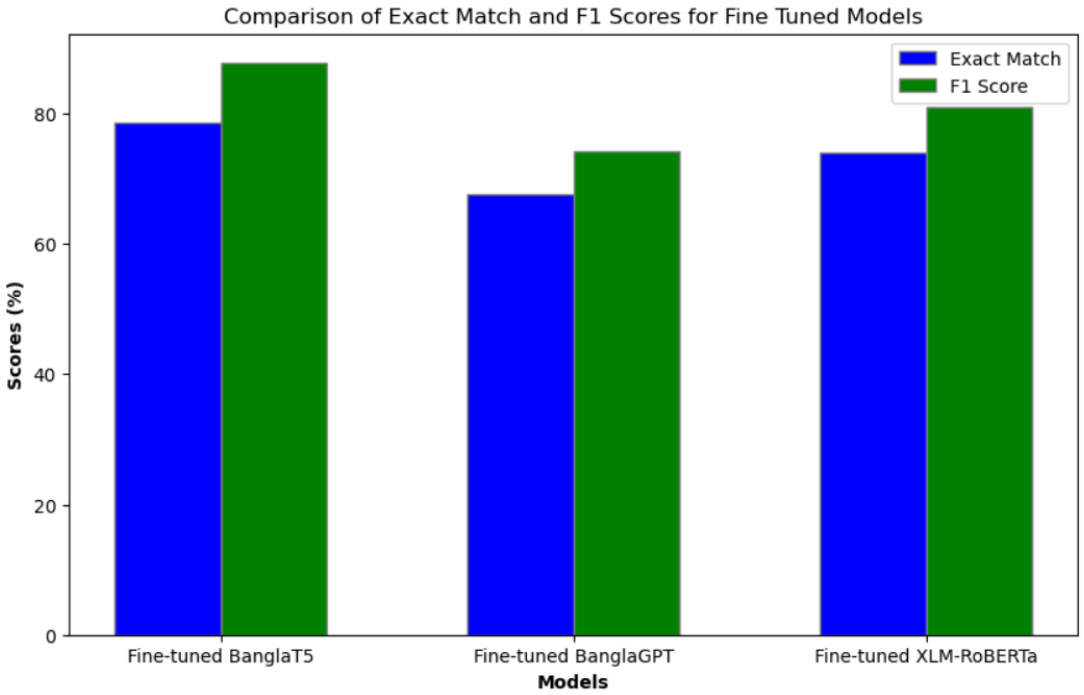


Figure 4.7: Comparison of The Fine-tuned Models

To sum up, BanglaT5 has the superior metrics among the models and showed its ideal position for selecting it for our work. Especially for task-specific objectives like ours, fine tuned BanglaT5 stands at the top.

Chapter 5

RAG Implementation and User Interface

5.1 RAG Implementation

The educational question answering system created during the present research is designed as a modular, sequential pipeline with three main steps: (1) Pre-processing of input and parsing of a query, (2) knowledge discovery, (3) generation of responses. Any of the stages has good subcomponents that provide linguistic accuracy and computational efficiency. The system is implemented to be operating smoothly across modalities (text and image), which makes it malleable to the real world conditions of education in Bangladesh; it can be expected that students can be engaged with either digital textbooks or printed material, hence the need to account for modalities.

5.1.1 Input Acquisition and Preprocessing

Our RAG-based educational QA system is based on the input acquisition and pre-processing module as its fundamental layer. Its main task is to work with the data provided by the users: either in text or image format and normalize it to the format suitable to be embedded and retrieved downstream. This module provides strong flexibility of inputs with regard to semantic intactness and linguistic consistency of Bangla language contents.

Input Modalities

There are two major input options of the system: (a) a direct textual system to enter user questions and (b) an image-based system which explicitly links the system to the textbook in printed form or in prints. Such backup mode ability becomes very important in terms of real life applicability of this technology in the education sector of Bangladesh, where students commonly use hard copy text books, printed hand outs and exam papers, which are not digitally present.

Direct Textual Input Handling

As the use enters a question in Bangla using the Streamlit interface, the system starts a light weight preprocessing pipeline, which standardizes the inputs without

changing their meaning. This starts with Unicode normalization under NFC (Normalization Form Canonical Composition), which allows consistent representation of Bangla characters, which are especially complex because of conjunct forms, as well as diacritics used in the script. The system then cleans by deleting redundant internal spacing, leading, and trailing whitespace. It also gets rid of extraneous or non-Bangla characters that could make embedding model confused, which include special characters or emojis. Syntactically and semantically significant marks are nevertheless kept in with full stops, commas or question marks to switch lines. Lastly, the cleaned, normalized question is simply tokenized with a user defined Bangla tokenizer that is compatible with the downstream sentence embedding model.

Image-Based Input Handling and OCR

In case the user uploads an image file (JPG, PNG or multi-page PDF), the system undergoes a more involved multi-step Optical Character Recognition (OCR) preprocessing process. This is designed to deal with a wide variety of images, and especially noisy real-word image situations, poor light, skewed scans, low-resolution pages or overlapping annotations. The procedure starts with conversion into grayscale that minimises complexity of channel and enhances contrast. It is succeeded by noise suppression by employing median filtering; it is better at removing salt-and-pepper like noise. It also does not blur the edges of characters, which in the case of Bangla font where very thin curves are parts of distinct letters is a key advantage to implementing this procedure. It is followed by adaptive Gaussian thresholding that is used to binarize the image so that the foreground text can be separated even in non-uniform light. The system will then carry out skew detection and correction via Hough Line Transform that computes the angle of the most pronounced text line and rotate the image horizontally to produce perfect match. This is important in maximizing OCR accuracy especially with handwritten or poorly scanned or aligned scans. When the image is cleaned and deskewed, the image is sent through the Tesseract OCR engine (v5.3), which is set to use the traineddata (ben) OCR applicative for Bangla script, as well as the Math/OCR equation support. Tesseract renders extracted text plus bounding box metadata and confidence scores, and font features, in hOCR or JSON.

Tokenization and Encoding Compatibility

After chunking, tokens of the text are created with the Bangla-compatible tokenizer that comes along with our sentence embedding model of choice. Compound consonants, vowel diacritics, and zero-width joiners are distinctive Bangla tokenization rules that are commonly misprocessed by models trained on Indo-European languages; tokenizer takes those into account. Erroneous tokenization in Bangla may heavily affect the meaning of a word (e.g., Romoni vs. Ro moni), so it is important to have a tokenizer written specifically to a language to preserve the language integrity. To help with this, the tokenizer is trained on corpus of Bangla textbook and exam-type text, such that named entities, technical terms and compound verbs are not faultily split. The outputs generated, those which are tokenized, are transformed into dense vectors in the embedding-stage, and the tokenizer allows the use of sequences whose length does not exceed the maximum possible length (which is usually 512 tokens). The segments which will be more than this limit will either

be further subdivided or cut in a strategic fashion preferring the semantic boundaries instead of cutting in a fixed length and cutting in the middle of a meaningful sentence.

5.1.2 Embedding and Vector Indexing

Embedding and Vector Indexing is a module that comprises the center of the retrieval component of the Retrieval-Augmented Generation (RAG) framework. It encodes textbook content in some semantically meaningful representation and organizes this content so that similarity search can be done efficiently. This section expounds on the steps and design choices that powered the conversion of Bangla educational text to humans to dense vector embeddings and indexing those vectors to a scalable, searchable index using FAISS [35].

Embedding Model Selection and Justification

Since we worked on the domain that requires high levels of language specificity, i.e., Bangla textbook and exam-style material, it was paramount to choose an embedding model that does not lose semantic connections and, additionally, course-specific syntax peculiarities of Bangla. The first set of benchmarking involved multilingual sentence embedding models including multilingual MiniLM, LaBSE, and multilingual universal sentence encoder (mUSE). Although such models displayed the general semantic skill, they performed poorly in the fine grained linguistic similarity tasks with Bangla constructs (conjunct verbs, postpositions, honorific pronouns). In turn, the system uses the model l3cube-pune/bengali-sentence-similarity-sbert, a fine-tuned algorithm SBERT specifically working on Bangla sentence similarity data. This is a model built on the siamese architecture of BERT to simulate cosine similarity on sentence-level embeddings and it has proven to be better in applications related to semantic paraphrase identification, reading comprehension, and QA matching in Bangla. Each input chunk is processed in the model, encoded as a 768-dimensional dense vector deriving deep contextualized meaning beyond the surface overlap of words.

Text-to-Vector Encoding Pipeline

At this point, each text chunk produced by the input preprocessing module, usually a coherent paragraph, or 3-5 consecutive sentences, will undergo training through the SBERT model and its tokenize-encode-infer preprocessing chain. The tokenizer will take care of the correct segmentation of Bangla script into tokens that will not violate compound characters and assign a fixed-sized vector to each chunk by the embedding layer. In order to have consistency, the final state that is about to be hidden from the [CLS] token will be taken out as the representation of the whole chunk. This is empirically justified method that has been later validated on Bangla sentence similarity tasks and has a good performance in retrieval tasks on top of the cosine similarity. Before the storage, vectors are unit-length normalized (L2-normalization), i.e. the cosine similarity and Euclidean distance are used to generate similar ranking outputs. Such normalization step is critical, particularly in case of approximate nearest neighbor (ANN) methods since it minimizes directionality bias and enhances the geometric meaning of the embedding space.

Vector Indexing using FAISS

To enable efficient large-scale search the system is based on Facebook AI Similarity Search (FAISS), a library optimized to perform efficient similarity search of dense vectors. The index is created in the form of FAISS index constructed by IndexIVF-Flat structure that is a combination of Inverted File System (IVF) and a flat inner product quantizer. Such a hybrid solution divides the embedding space into several clusters (k-means clustering is used to cluster the space during training), and new vectors are added to the posting list of the nearest centroid. When the retrieval is done, a collection (subsets) of these posting lists are searched and specifically narrows the time consumed during the look-up. A representative sample of all the embedded chunks is used in training the index; it is used in coming up with the initial centroids. The value of nlist (number of clusters) is usually selected empirically, so as to trade off retrieval latency and recall: it is often set to the square root of the number of chunks. Similarity between inner products is applied with aid of vectors normalized using cosine to estimate semantic distance successfully.

Index Quantization and Optimization

The possible size of the system (textbook pages and tens of millions of embedded fragments) makes memory efficiency a major issue. FAISS offers a choice of three methods of quantizing the high-dimensional input vectors into concise binary codes: Product Quantization (PQ) and Scalar Quantization (SQ). In this system, scalar quantization may be used to compress the float32 768-dimensional embeddings down to int8 representations. This saves a lot of RAM (reducing it by more than 75 percent) at the little cost of retrieval accuracy. Also, the index gets serialised to disk and cached in memory throughout system boot up. This method lowers the cold-start latency and allows deployment to be fast during research and production. The index will be re-trained and re-indexed regularly in order to accommodate new textbook content or embedded refinements, so each indexed entry will always include newer content. Our RAG system, in summary, forms a semantically rich, efficiently searchable representation of raw educational content and this is a process implemented at embedding and vector indexing. The system can retrieve fast, contextually relevant information with the help of a Bangla optimized SBERT model, and a highly scalable FAISS ANN based index. Such an infrastructure preconditions the use of reliable real-time context selection and the subsequent generative components of QA system.

5.1.3 Context Retrieval Mechanism

User Query Embedding

Knowledge retrieval in our system RAG system occurred when a user entered question either typed manually or transcribed by an OCR to an image. This paragraph is devoted to the embedding and retrieval of the user query which seems to be critical to finding useful textual pieces in the knowledge base that will be forwarded to the generators model. The most important design goal here is making sure that

there is semantics matching between the query vector and document chunks that were embedded previously into the vector database.

Preprocessing for Consistency

The system is pre-conditioned to apply the same preprocessing pipeline to the question as it applied when ingesting the documents in order to create a similarity in representation between the query requested by a user and the ones stored as document embeddings. This involves Unicode normalization to NFC, which is especially significant in Bangla script, because it contains several code points that can indicate the same grapheme. Moreover, whitespace gets normalized by condensing multiple spaces to one token and unnecessary textual punctuations, not linguistically meaningful, is eliminated. These normalizations are done to prevent spurious mismatches of embeddings due to formatting differences but not semantic differences.

Encoding the Question as a Vector

The preliminary query is normalised and then is run through the same model of sentence embedding employed in the indexing of the documents: l3cube-pune/bengali-sentence-similarity-sbert. [27] This will mean that the question is contained in the same space as the document chunks. The input is tokenized with the Bangla-compatible tokenizer of the model and the output of the final encoder layer corresponding to [CLS] token is taken as the 768-dimensional representation of the whole question. Instead of using surface-level details such as lexical features, this vector uses contextual semantics of the question, and can therefore match relevant material on a high-quality level even in cases of phrasal variation. The reason why the SBERT model used in both question and context embedding is the same is homogeneity in the embedding space and negating the domain drift that may arise as a result of asymmetric encoders (e.g., Bi-encoders vs. Cross-encoders). Such symmetry is essential to cosine-based similarity search and ensure that retrieval is done by relational closeness and not absolute coordinate location in the space.

Querying the FAISS Vector Index

The resulting vector of questions was then presented as a query to the FAISS index that was used in using the textbook chunks as embedding. The FAISS index, built as IndexIVFFlat, is optimized rapid approximate nearest neighbor (ANN) query in high dimensional spaces On query vector FAISS will first determine the most relevant centroids (precomputed when the index is trained with k-means clustering) and search accordingly in the corresponding posting lists based on their similarity. This cuts the comparison and sorting by a literal order of magnitude, making an O(n) brute-force to an O(root-over(n)), with n document chunks in the index. By default, the cosine similarity is used because it is highly applicable in comparing the normalized unit embeddings. The cosine similarity measures an angular distance between two vectors thus retains semantic similarity irrespective of the magnitude of the vectors. It renders it resistant to changes in length or intensity or frequency of some words of sentences, criteria that are common in natural Bangla phraseology. As an example, a user can ask, “Who are the writers?” and the textbook itself might

talk of the same thing in the words "Poets and novelists of Bengal" but lexical gap can be bridged by the cosine-based similarity search.

Top-k Context Selection

After computing the similarity scores, the index will provide the top-k highest semantically ranked document chunks with a natural number of k predetermined by empirical experiments, usually k=8. This value gives a moderate tradeoff of richness of context and computational tractability in the generation phase. The result of the retrieval is sorted in the order of decreasing cosine similarity score and filtered up to rid itself of near-duplicate entries (e.g. overlapping or repetitive paragraph chunks). Optionally, diversity-conscious reranking may be optionally fixed to require that the chosen passages cover different subtopics or views in the scope of retrieved passages by employing Maximal Marginal Relevance (MMR). With every selected chunk, there is its associated metadata: textbook, chapter name, and location is kept to be shown and tracked. This is to enable that the generated answer can subsequently direct or recount which pieces of the curriculum were drawn upon strengthening transparency and academic reliability.

Embedding Caching and Efficiency Considerations

In order to perform optimally, the system also includes an optional faq-like caching system based on key values where the normalized (key) question string will be null and the value is a set of FAISS document results. This prevents duplication of calculation when the same query is made again and makes the response time much faster especially where a query might be repeated by various users in the classroom or the preparation of exams. Also, in multi-user set ups the embedding model and FAISS index will be buffered (through a persistent server session or API backend) to prevent recurrent cold starts. Any embedding operation is batch and accelerated with GPU inference whenever possible. To conclude, the user query embedding is a closely juxtaposed system, which converts natural-language Bangla questions into a dense and semantically rich document embedding space that aligns to a dense and semantically rich document embedding space. By using a combination of prepossessing, parts-of-speech encoding where one model is shared across and simple retrieval through FAISS, the system will expose the user to highly relevant context information irrespective of superficial linguistic differences. The beauty of this mechanism is its foundation to the retrieval mechanism and it directly affects the quality and relevance of the answers ready to be produced in the later stages of retrieval.

Context Retrieval and Re-ranking

The first attempt at searching the FAISS index using the nearest neighbor (ANN) method is not the concluding point of the retrieval pipeline. Although the top-k list provided by FAISS provides kinds of coarse-grained filtering of relative content, the passages themselves are frequently semantically similar, but unsuitable or too contextually broad to the objective at hand, or insufficiently particular to the user objective. This is more so in open-domain learning environment with textbook material that may be thematically sparse, yet semantically repetitive. To cope with that, a multi-stage reranking and filtering system is included in the system to

enhance the relevance, specificity and diversity of contextual information feeding into the generative model. **Initial Lexical Filtering using Jaccard Similarity** The initial nugget of this re-ranking pipeline is the removal of outlier passages achieved via lexical level pruning with Jaccard similarity. Jaccard similarity is a calculation, which determines how many token sets are the same between a question the user types in and a given retrieved chunk. This measure is not precise, but an efficient measure of lexical overlap, and when both the lexical overlap and the measure of similarity are small, the retrieved chunk might contain no shared vocabulary and no core concepts but might be strongly correlated with the same topic. As an example, a question concerning "Bangla Independence Movement" can mistakenly recall a text on "Modern Literature Movement" they would be irrelevant to each other, and such dissimilarity would be reflected in lexical embeddings. Any chunks that have Jaccard scores of less than a number that can be configured (0.2-03) are eliminated to avoid misleading or noise context being introduced.

Semantic Re-ranking with Cross-Encoder

The unfiltered list of candidate chunks that are typically 8-12 chunks after lexical filtering are then submitted to a cross-encoder model to receive a fine-grained semantic relevance score. As opposed to bi-encoder systems (like SBERT) where the question and the passage are embedded separately, in a cross-encoder, both sequences are processed together in a stack of transformers, which allows it to learn fine-grained interactions, and positional correlations, and attention patterns between the two sequences. In doing this task, the system employs a cross-encoder fine-tuned with MiniLMv2 on Bangla question-passage relevance data. This model is trained on finetuned examples of QA-relevance based on Bangla school curricula, national examination questions and annotations based on comprehension. In inference, the model is fed with a (question, chunk) pair and generates a scalar relevance score on the interval of 0-1 reflecting the probability that the chunk bears meaning regarding the question. Each of the passages of all the candidates is marked separately and are ranked according to their relevance, decreasing in order. This reranking-based semantics is particularly successful at distinguishing such tenuous (and sometimes subjective) relevance borders, e.g. telling the difference between a chunk that only discussed a subject and a chunk that offered explanations or a piece of the answer. Besides, it enhances robustness when it comes to paraphrased questions, negation, and implicit context questions.

Context Consolidation and Delimiter Encoding

Upon reranking, the N best passages (usually 2 or 3) with highest cross-encoder scores are chosen to make a final inclusion to the prompt passed in the generative model. These passages can be fed rather than as a single undifferentiated string but as an enumeration, joined together by a special context-separation token which can be used in multiple contexts during downstream generation. To begin with, it gives structural direction to the generative model which means logical limits of different sections of contexts. Second, it allows the model to focus individually on each passage and determine their respective or collective significance to the various aspects of the user question. The empirical evaluations have shown that context size should be limited to 2-3 passages because the further context growth tends

to produce less and less improvement in the quality of the generated answers and makes them more and more inefficient in terms of generation time and noise. The chosen passages are token-trimmed, when needed, to the maximum 512-768 tokens of the context window of the BanglaT5 model (depending on the vocabulary of the tokenizer). Syntactic and semantic completeness is maintained by, where possible, truncating passages at sentence boundaries.

Handling Noisy or Ambiguous Queries

The cross-encoder scoring mechanism also helps in resilience in the scenarios when there is an undescribed query by the user, semantic ambiguity, or OCR noise (e.g., because of image quality). Because the cross-encoders can be trained to downrank irrelevant content even in the context of syntactic variance, they serve as extra precaution against copycat injection into the context. In the case that no candidate passage has a minimum relevance score (e.g. the cutoff is above 0.5), the system will either ask the user to rephrase the question or give a fallback response that there is no context to produce an answer with confidence. As a wrap-up, the context retrieval and reranking module highly improves the performance of the generative phase by pruning and re-arranging the FAISS-recruited candidates, effectively into an annotated, ranked and logically-organized embodiment of available passages. Comprising fast approximate search and deterministic lexical filtering with transformer-based semantic reranking, the system provides high precision in context selection, with efficiency in application. Such a multi-tired system means that the generative model is presented not only with similar-appearing text, but with pedagogically sound and contextually anchored text that actually promotes correct and educationally meaningful responses.

5.1.4 Generative Answer Formulation (BanglaT5)

After retrieving and reranking the most relevant contextual passages, the system proceeds to the generative phase, wherein a linguistically fluent and pedagogically sound answer is composed in Bangla. At the core of this stage lies a generative transformer model "BanglaT5", a powerful encoder-decoder architecture optimized for educational and question-answering tasks in the Bangla language. Unlike traditional extractive approaches, this generative module synthesizes answers by jointly reasoning over multiple context passages, ensuring coherent and contextually aligned responses to diverse student queries.

Pretraining and Fine-Tuning Corpus

The model was based on the BanglaT5 [20] that was pretrained in large-scale datasets like OSCAR and Common Crawl but filtered by Bangla and subsequently fine-tuned on a curated hybrid data built-in especially to serve the educational QA. The three main components of this dataset includes:

1. Synthetically constructed question-answer pairs based on national curriculum textbook text based on weakly supervised generation templates, grammar aware paraphrasing and sentential expansion.

2. Question-answer sets composed by people, which translate the formal, factual, pedagogical emphasis of textbook responses. They are extensive and touch all sorts of topics such as Bangla literature, history, general science and social studies.

3. Adversarial and null context examples that are intended to make the model learn to determine that a question may not be answered in the input. These examples make sure that the model learns to avoid answering certain questions (i.e. learns to abstain) instead of hallucinating answers to questions that it is not certain about.

This solution-efficient fine-tuning approach enables the model with resilience into a wide range of educational circumstances, such as vague inquiries, partially matched conditions, as well as question-context paucity.

Input Formatting and Fusion-in-Decoder Architecture

When inferencing, during processing the BanglaT5 receives a combination of original user query followed by the two or three most relevant context passages (separated by special delimiter token, e.g.,). This formatting approach assists the model to differentiate various sources of contexts.Both the sequences are tokenized in the form of a Bangla-specific SentencePiece tokenizer that can effectively work with ligatures, compound characters, and subword fragments that appear in the Bangla writing system. The BanglaT5 takes a different architecture (fusion-in-decoder) to that of

Dracula, in that the encoder operates over the complete input sequence, whereas the decoder focuses over the entire set of encoded tokens during generation. Such an arrangement allows the model to combine the information of multiple passages together, without fragmentation of information, unlike the retrieval models, where processing only passages or context windows happen in isolation.

Decoding Strategy: Contrastive Search

The system uses contrastive search to decode answers in order to maximize quality of answers. Instead of picking the token with the highest probability (relying purely on what is called maximum likelihood) according to the greedy decoding sense, as when the technique makes a choice only according to the most probable token, but in addition to these diversification selection criteria are taken into account by the technique. In particular, the candidate tokens are scored according to confidence and dissimilarity in order to exclude redundancy and to enhance the factual informativeness. The use of contrastive search addresses the generic or repetitive outputs that usually occur in language models in general and it avoids claims that lack support especially in education. Also the decoding is made subject to two constraints, namely a token budget, usually 80 tokens, and the presence of an early stopping mechanism, implemented with the "dari"(Fullstop for Bangla). This is a type of punctuation mark of ultimate global prevalence in Bangla since it naturally (but perceptually) marks sentence boundaries. It makes sure that the responses will be brief, relatively context-based, and of similar styles to the typically expected responses in the classroom.

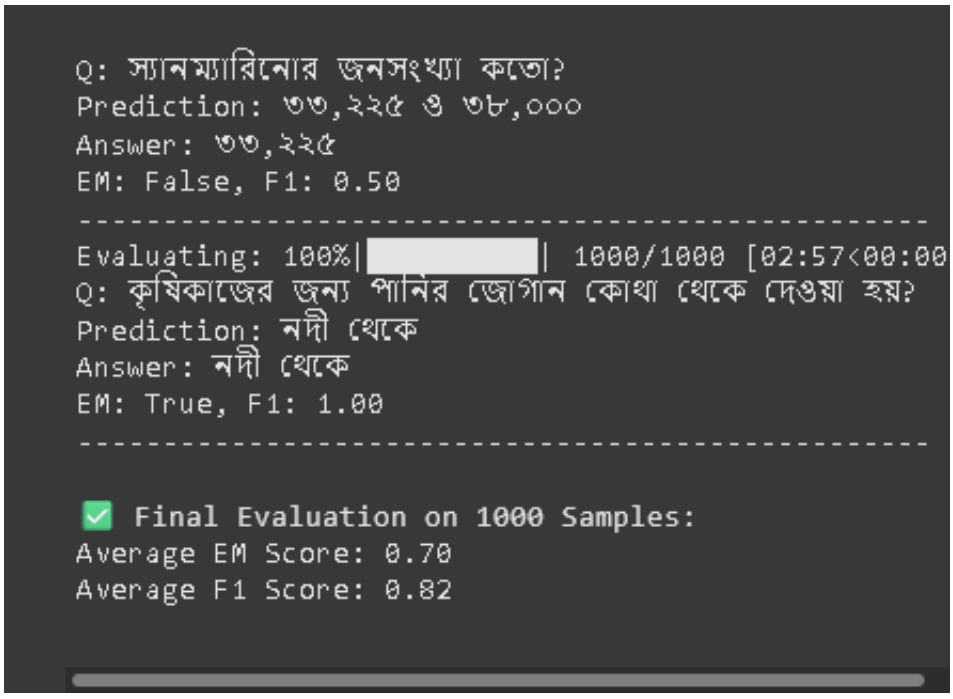


Figure 5.1: BanglaT5 EM and F1 Metrics with Test Dataset

5.2 Comparison Between Finetuned BanglaT5 and RAG Pipeline

We selected 1000 question-answer pairs from our dataset to evaluate the finetuned BanglaT5 model and RAG pipeline and found these results.

We've observed that the discrepancy in EM (Exact Match) and F1 scores between our base fine-tuned model and our RAG (Retrieval-Augmented Generation) pipeline arises from key architectural and functional differences. Firstly, EM evaluates if the predicted answer exactly matches the context after normalization. However, the F1 score calculates the overlap at the token level by combining precision and recall.

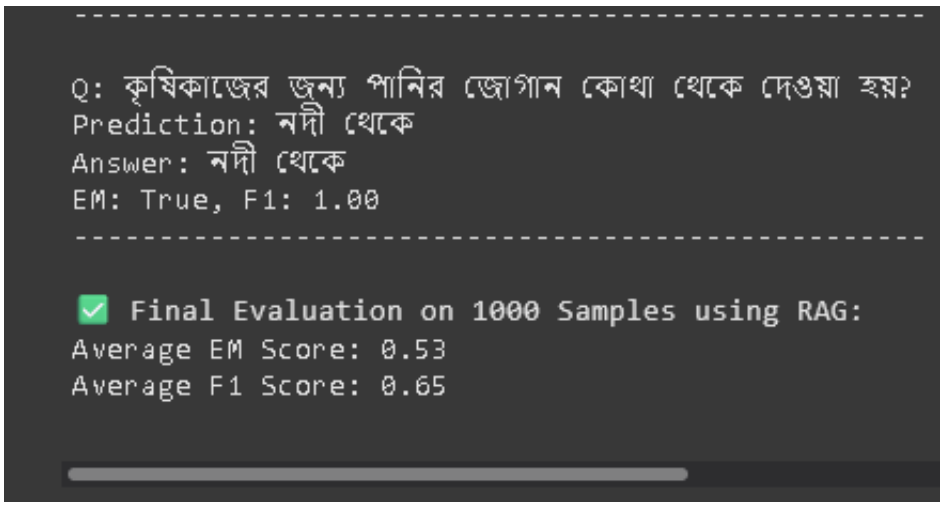


Figure 5.2: RAG Pipeline EM and F1 Metrics with Test Dataset

Our base fine-tuned mode - BERT, RoBERTa, or T5 was trained end-to-end on a supervised QA dataset such as SQuAD or Natural Questions. As the context passages are fixed and provided during training, the model learns to extract or generate answers in a controlled format. This direct supervision helps to consistently produce precise outputs. As a result, it leads to strong EM and F1 scores on evaluation benchmarks.

However, the RAG pipeline adopts a totally different two-stage architecture. Firstly, a retriever component dynamically searches a large corpus (using FAISS or similar vector-based methods) to find top-k relevant passages. Then, a generator like BART or T5 creates an answer conditioned on the retrieved documents. But RAG isn't trained in a fully end-to-end manner unlike our base model. This happens especially if the retriever is frozen or independently tuned. This disjointed training often introduces challenges.

A major factor behind RAG getting lower EM and F1 performance is retrieval noise. If documents given to the retriever are irrelevant or only partially related, the generator might produce inaccurate answers. Even when working with reasonably relevant content, the generative model often produces answers that are more verbose or paraphrased. These answers may be semantically correct but deviate from the exact ground-truth phrasing. As a result, it gets lower EM and, in some cases, reduced F1 due to token mismatches.

Another challenge is found in the generative nature of RAG's outputs. On one hand, our fine-tuned extractive model typically returns short spans with high lexical overlap. On the other hand, RAG tends to produce full natural language sentences. Though these responses may feel more human-like or informative, they are penalized by strict EM/F1 scoring due to auxiliary content, rewording, or reordered phrases. Moreover, the use of stochastic decoding techniques (like top-k or nucleus sampling) brings additional variability at inference time unlike the more deterministic outputs of the extractive models.

Also it's important to notice that RAG has clear strengths. It excels in open-domain QA and real-world applications where information needs to be retrieved dynamically. It's important when the questions require knowledge not present in the training set. In such scenarios, RAG can outperform closed-book models in terms of semantic accuracy, even if the EM/F1 scores appear lower.

To illustrate the point: given the question, "Who developed the theory of relativity?", our fine-tuned model might return "Albert Einstein," yielding perfect EM and F1. On the other hand, RAG will probably generate: "The theory of relativity was developed by physicist Albert Einstein in the early 20th century." Even though the answer is factually accurate and even more informative, this answer fails the exact match criterion and includes extra tokens that reduce the F1 score.

In summary, our fine-tuned model tends to achieve higher EM and F1 scores due to its precise and concise outputs. On the other hand, our RAG pipeline provides retrieval imperfections and the generative output format. To improve RAG's

performance,more flexible and potentially informative answers at the cost of lower metric scores, mainly we are exploring strategies like fine-tuning the retriever on in-domain data, applying reranking methods to filter more relevant documents, and using post-processing techniques to align generated answers more closely with expected outputs.

5.3 OpenAI API Integration

5.3.1 Bangla Question Generation Function

The question generation function generates Bangla-language questions from the input paragraph through the OpenAI API with the `gpt-4.1-nano` model. This model is token-efficient while optimizing high instruction-following capability in Bangla. The function uses a properly structured prompt:

```
    prompt = f"""
এই অংশ পড়ে {num_questions}টি প্রশ্ন লেখো:
\"{paragraph.strip()}\"
শুধু প্রশ্ন দাও:
১.
"""
```

Figure 5.3: OpenAI Prompt for Question Generation

The API call has parameters such as `temperature=0.5` to balance creativity and focus, and `max_tokens=200` to limit response size. After receiving the model's output, the function converts Bangla-numbered query inputs (1., 2., 3., etc.) to normal form and returns them as a list. This makes it ideal for creating tidy, contextually appropriate comprehension or passage-based questions for educational exams, e-learning software, or quiz programs.

5.3.2 Bangla Distractor Generation Function

The `generate_distractors_bangla` function creates three plausible but incorrect Bangla answer options—distractors—for a given factual sentence, also using the `gpt-4.1-nano` model via OpenAI's API. The function sends the prompt:

```
    """
    prompt = f"""
"{sentence}" এই বাক্যের জন্য সঠিক উত্তরের কাছাকাছি কিন্তু ভুল ৩টি বিভ্রান্তিকর বিকল্প লেখো।
শুধু সংখ্যা সহ বিকল্পগুলো দাও:
১. ...
২. ...
৩. ...
"""

    try:
        response = client.chat.completions.create(
            model=model,
```

Figure 5.4: OpenAI Prompt for MCQ Distractor Generation

It inserts parameters like `temperature=0.7` to make the distractors a little more imaginative, and `max_tokens=200` to keep the answers short. The method extracts just the Bangla-numbered alternatives from the model's reply, producing good-quality distractors that are contextually appropriate but wrong. This is helpful in generating challenging MCQs for exams, quizzes, or online learning platforms in Bangla.

5.4 User Interface

5.4.1 OCR Integration

The OCR system integrates with the user application to start a parallel and lightweight OCR processing pipeline whenever a user uploads a new image or document, e.g., a page of a reference book, a worksheet printed off, hand-written class notes. This works the same way as the default OCR module in offline corpus ingestion, going through grayscale conversion, binarization, noise filtering, and skew correction before Parsing by means of Tesseract OCR v5.3, properly configured to support Bangla script and equation types.

The text triggered (or selected) is, unconditionally, preprocessed (cleaned up) and chunked (constrained to a fixed number of tokens so as to suit follow-on models), with the maximum length of tokenization restricted to suit the downstream models. The embedding of each chunk is then also done in real time with the sentence transformer that was used during the offline indexing of the corpus (l3cube-pune/bengali-sentence-similarity-sbert), so that the vector embedding is comparable to the already constructed index.

These newly embedded vectors are not written to disk-resident FAISS stores so as not to jeopardize the integrity of the underlying master list that is persistent. They are rather put in a volatile, in-memory FAISS IndexFlatIP instance that is dedicated to the running session. This architecture provides maximum isolation of the user-provided documents and avoids both contamination of the global data set and data reproducibility and versioning. Here is an example of the ORC input:

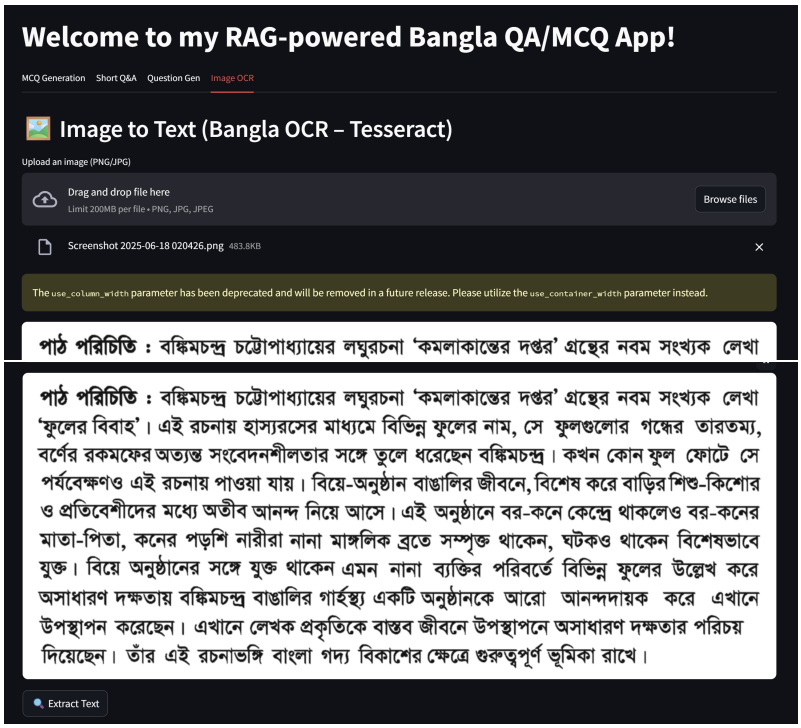


Figure 5.5: OCR Input

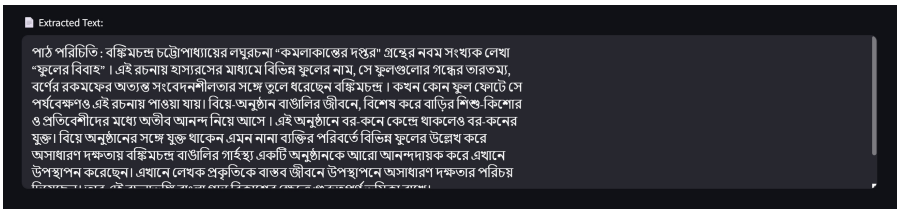


Figure 5.6: OCR output

The first two screenshots show how the user can upload an image file in JPG, PNG, or JPEG format containing a block of bangla text. This can be a scanned page, a PDF screenshot, or a photo of printed material. The picture needs to be clear. This image works as the input for our OCR (Optical Character Recognition) function. The second screenshot shows the result after processing the image. In the extracted text box, the embedded text gets extracted and converted into plain, editable text. This feature helps users quickly retrieve bangla texts from a picture and reuse content from printed or image-based documents without manual typing. This can be especially useful for students, teachers, researchers, or anyone working with scanned academic or reference materials. Also this extracted text can be used to provide context for the application's other function - question-answer generation and MCQ generation.

5.4.2 Question Generation

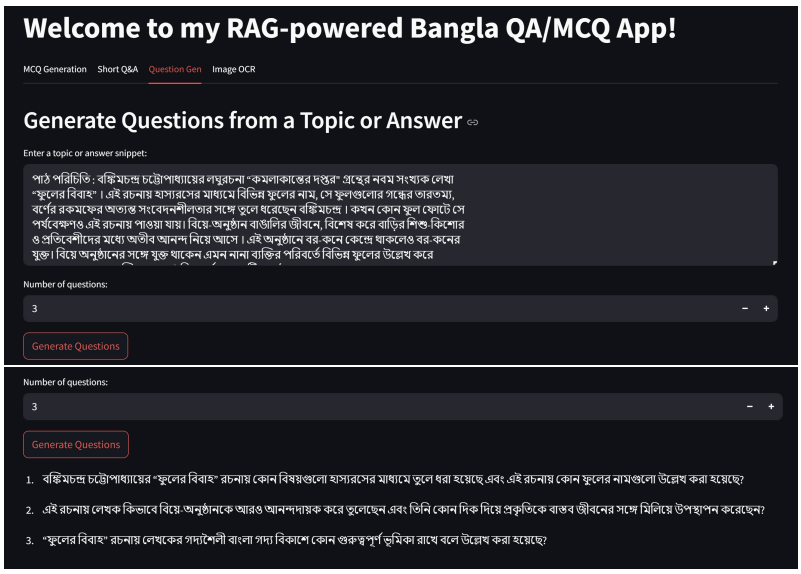


Figure 5.7: Question Generation

This interface shows the Question Generation – Input and Output Interface. This takes a Bangla context and the user provides the number of questions they want as input. It then generates that number of meaningful and context-relevant questions directly from the provided context. This helps users to practice exam-style questions. It also makes the process efficient for educators and students to generate questions and practice quickly.

5.4.3 Short Question/Answer Generation

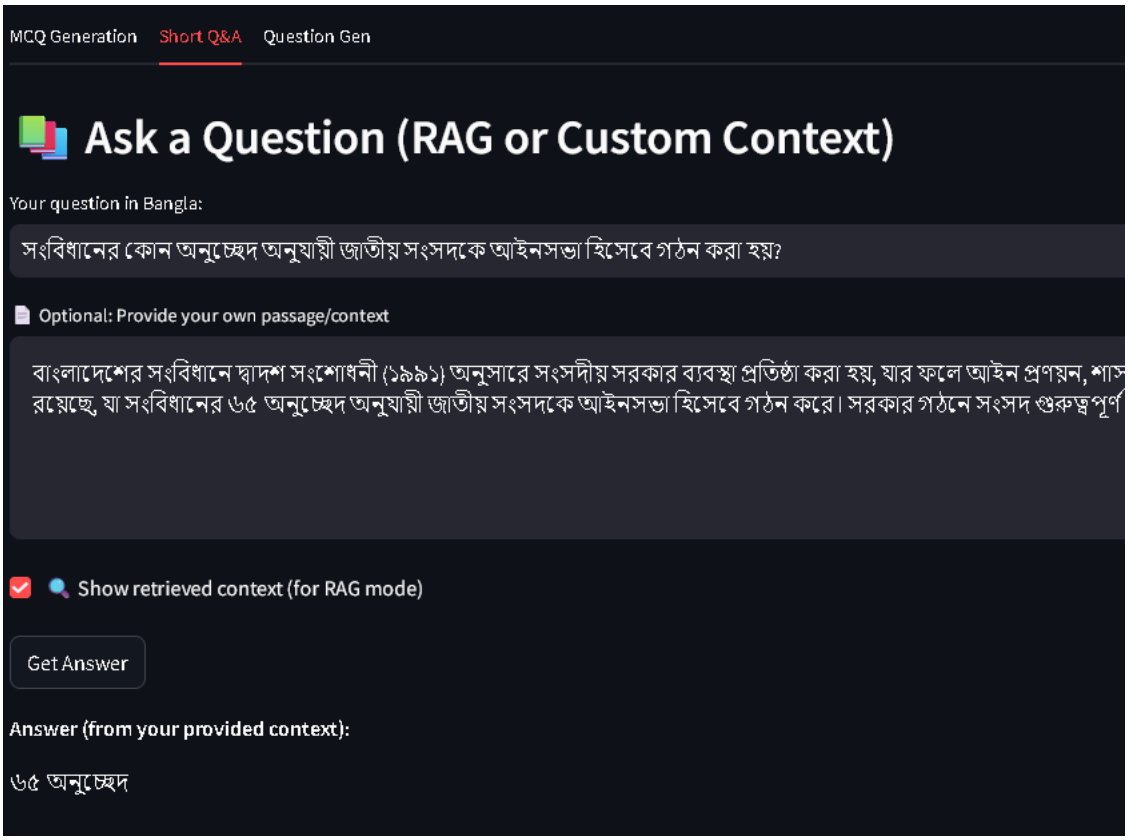


Figure 5.8: Short Question/Answer

This screenshot demonstrates the Short Question/Answer Generation – Input and Output Interface. This function asks the users to input a question, either generated by the system or written manually in Bangla. An optional context box is also given here to improve accuracy, especially for questions that need specific information. The system then generates a short answer. However, if the answer is not directly present in the context, the output may be incomplete or less accurate.

5.4.4 MCQ Generation

In this section, two screenshots of the interface of MCQ generation function has been provided. It allows the users to input a Bangla context the desired number of multiple-choice questions that will be generated. Once the context and question number is submitted, the system processes the context and automatically generates each question with four options—three distractors and one right answer. The distractors are designed to be contextually similar, however they will be incorrect. Also, the right answer will be shown under the options. This ensures that the generated MCQs are meaningful.

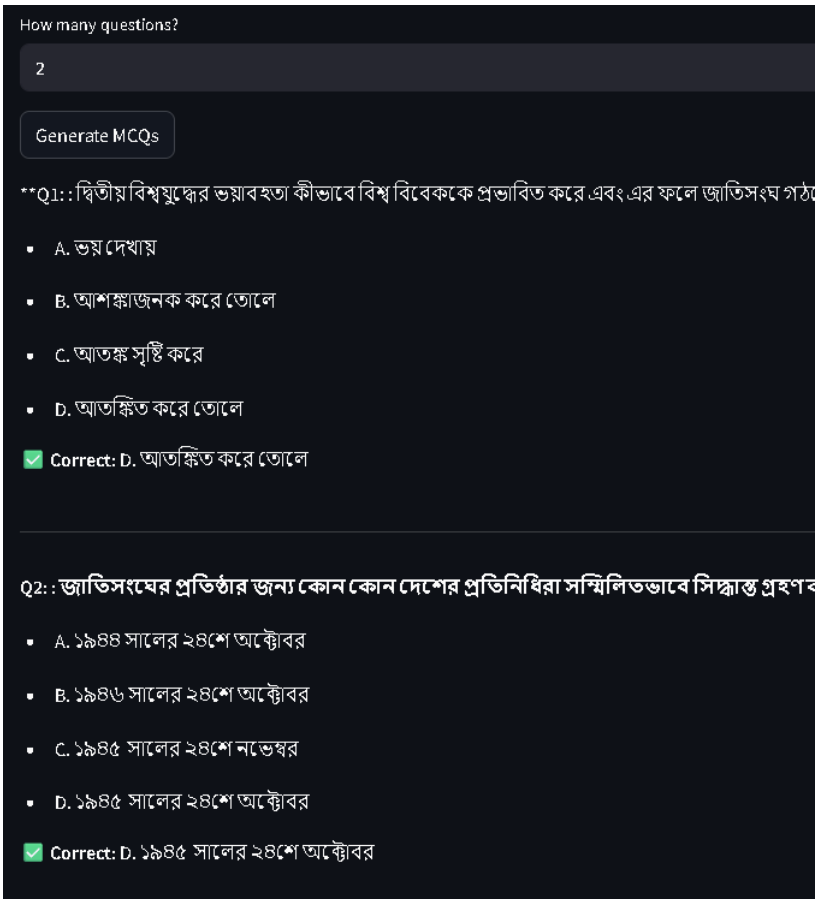


Figure 5.10: MCQ Generation Output

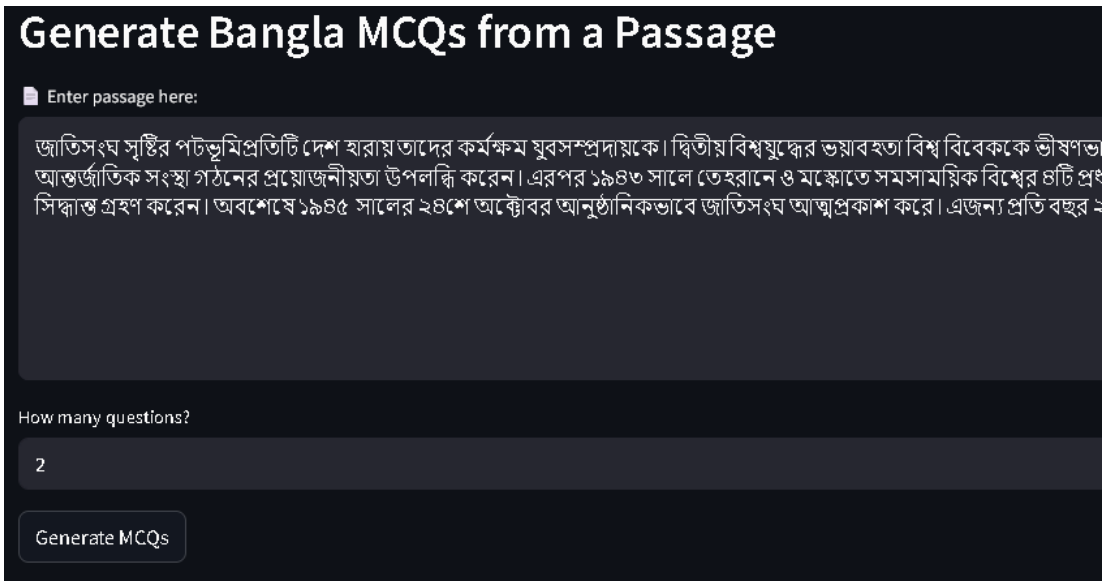


Figure 5.9: MCQ Generation Input

Chapter 6

Conclusion

6.1 Conclusions

To put it briefly, our task is to prepare a quality, curriculum based Bangla dataset to facilitate the construction of intelligent question-answering systems for Bangla-medium students. By structuring the dataset in the SQuAD format that consists of carefully created context passages, questions, and answers. We also provided the foundation resource required to train and fine-tune models like BanglaT5. This dataset enables developing AI-based tools that have the potential to assist students in contextual question answering, self-evaluation, and personalized learning. We believe that our dataset is a useful contribution to the low-resourced field of Bangla NLP for education and opens the door to developing inclusive, effective tools for Bangla-speaking students.

6.2 Limitations

No research is without limitations and this paper is not different from those either. These are the limitations that were found out-

1. One of the main limitations of the system is the limited dataset for training and fine-tuning the model. There are very few resources when it comes to Bangla language compared to other languages.

2. If the given context is vague and irrelevant, the intelligent system fails to give a feasible response. It is trained for more fact based answers hence it's incapable of critical thinking or data analysis.

3. The system is built on BanglaT5 which is a transformer based model specifically for Bangla. As it's based on Bangla language, BanglaT5 is not as optimized as its English counterparts such as T5 or BERT due to scarcity of training resources.

6.3 Future Works

Our study primarily focuses on the BanglaT5 model and data collected from NCTB curriculum textbook for class 9-10 for Bangladesh and Global studies and Bangla

First Paper. For future improvement of the system, more data can be collected from other textbooks to enhance the knowledge base for the model. Not to mention, the enhanced dataset can be used to train LLMs such as Llama or GPT4 for better generative performance and accuracy of this question-answering system. These models can handle complex queries, offer better understanding of Bangla language and smarter generation of context based answers which the BanglaT5 model is limited in. Moreover, the RAG pipeline needs to be evaluated using a sementic approach for proper evaluation since the EM and F1 metrics are not suitable for this task.

Bibliography

[1] M. A. Hasnat, M. R. Chowdhury, and M. Khan, "An open source tesseract based optical character recognizer for bangla script," in *2009 10th International Conference on Document Analysis and Recognition*, 2009, pp. 671–675. DOI: 10.1109/ICDAR.2009.62.

[2] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ questions for machine comprehension of text," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, J. Su, K. Duh, and X. Carreras, Eds., Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 2383–2392. DOI: 10.18653/v1/D16-1264. [Online]. Available: https://aclanthology.org/D16-1264.

[3] M. Aleedy, H. Shaiba, and M. Bezbradica, "Generating and analyzing chatbot responses using natural language processing," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 10, no. 9, 2019. DOI: 10.14569/IJACSA.2019.0100910.

[4] A. Conneau, K. Khandelwal, N. Goyal, *et al.*, "Unsupervised cross-lingual representation learning at scale," *CoRR*, vol. abs/1911.02116, 2019. arXiv: 1911.02116. [Online]. Available: http://arxiv.org/abs/1911.02116.

[5] S. M. Islam, M. F. A. Houya, S. M. Islam, S. Islam, and N. Hossain, "Adheetee: A comprehensive bangla virtual assistant," in *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*, Dhaka, Bangladesh, 2019, pp. 1–6. DOI: 10.1109/ICASERT.2019.8934903.

[6] N. Ahmed and S. Karim, "Complex syntax and nlp challenges in bangla," *Linguistics and AI*, vol. 2, no. 3, pp. 93–108, 2020.

[7] M. Alam, F. Kabir, and F. Zaman, "Personalized learning through ai: A south asian perspective," *International Journal of Educational Technology*, vol. 37, no. 3, pp. 115–127, 2020.

[8] Z. Chai and X. Wan, "Learning to ask more: Semi-autoregressive sequential question generation under dual-graph interaction," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 225–237. DOI: 10.18653/v1/2020.acl-main.21.

[9] Y. Du, C. Li, R. Guo, *et al.*, *Pp-ocr: A practical ultra lightweight ocr system*, 2020. arXiv: 2009.09941 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2009.09941.

[10] M. Lewis, Y. Liu, N. Goyal, *et al.*, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 7871–7880. DOI: 10.18653/v1/2020.acl-main.703.

[11] P. Lewis, E. Perez, A. Piktus, *et al.*, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. DOI: 10.48550/arXiv.2005.11401. [Online]. Available: https://arxiv.org/abs/2005.11401.

[12] A. Patel and A. Ghosh, "Localized learning tools: The missing link in educational ai," *AI for Inclusive Education*, vol. 5, no. 1, pp. 41–56, 2020.

[13] M. Bommadi, S. Terupally, and R. Mamidi, "Automatic learning assistant in telugu," in *Proceedings of the 1st Workshop on Document-Grounded Dialogue and Conversational Question Answering (DialDoc)*, 2021, pp. 29–37. DOI: 10.18653/v1/2021.dialdoc-1.4.

[14] M. H. Chowdhury and R. Shahnaz, "Digital divide in bangladesh: Challenges and policy implications," *Journal of Information Policy*, vol. 11, pp. 202–223, 2021.

[15] M. M. Hasan, A. Roy, and M. T. Hasan, "Alapi: An automated voice chat system in bangla language," in *2021 International Conference on Electronics, Communications and Information Technology (ICECIT)*, Khulna, Bangladesh, 2021, pp. 1–4. DOI: 10.1109/ICECIT54077.2021.9641323.

[16] F. S. Khan, M. A. Mushabbir, M. S. Irbaz, and M. A. A. Nasim, "End-to-end natural language understanding pipeline for bangla conversational agents," in *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Pasadena, CA, USA, 2021, pp. 205–210. DOI: 10.1109/ICMLA52953.2021.00039.

[17] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A survey on bias and fairness in machine learning," *ACM Computing Surveys (CSUR)*, vol. 54, no. 6, pp. 1–35, 2021.

[18] F. Rahim and S. Munna, "Technology access and digital literacy in rural education," *Journal of ICT and Development*, vol. 9, no. 2, pp. 12–26, 2021.

[19] O. Rahman and S. Talukder, "Underrepresentation of bangla in modern nlp and its consequences," *Journal of South Asian AI Research*, vol. 3, no. 2, pp. 55–70, 2021.

[20] A. Bhattacharjee, T. Hasan, W. U. Ahmad, and R. Shahriyar, "Banglanlg: Benchmarks and resources for evaluating low-resource natural language generation in bangla," *CoRR*, vol. abs/2205.11081, 2022. arXiv: 2205.11081. [Online]. Available: https://arxiv.org/abs/2205.11081.

[21] T. Hasan, N. Alam, R. U. Ahmed, *et al.*, "Banglaai: Building nlp tools for low-resource bangla language," in *Proceedings of the Language Resources and Evaluation Conference*, 2022.

[22] N. Islam and S. Jahan, "Ai and education in rural bangladesh: Opportunities and infrastructural gaps," *Bangladesh Journal of ICT*, vol. 5, no. 2, pp. 33–45, 2022.

[23] S. Roy, T. Sultana, and J. Islam, "Banglanlp 2.0: Challenges and progress," *Journal of Asian Computational Linguistics*, vol. 4, no. 1, pp. 66–79, 2022.

[24] R. Thoppilan, D. D. Freitas, J. Hall, *et al.*, "Lamda: Language models for dialog applications," *arXiv*, 2022. DOI: https://doi.org/10.48550/arXiv.2201.08239.

[25] S. Alwahaishi, "A smart interactive behavioral chatbot: A theoretical prototype," in *2023 8th International Engineering Conference on Renewable Energy Sustainability (ieCRES)*, Gaza, Palestine, State of, 2023, pp. 1–5. DOI: 10.1109/ieCRES57315.2023.10209534.

[26] Y. Dan, Z. Lei, Y. Gu, *et al.*, "Educhat: A large-scale language model-based chatbot system for intelligent education," *arXiv*, Jan. 2023. DOI: 10.48550/arxiv.2308.02773.

[27] S. Deode, J. Gadre, A. Kajale, A. Joshi, and R. Joshi, "L3cube-indicsbert: A simple approach for learning cross-lingual sentence representations using multilingual bert," *arXiv preprint arXiv:2304.11434*, 2023.

[28] L. Gutiérrez, "Artificial intelligence in language education: Navigating the potential and challenges of chatbots and nlp," *RSELTL*, vol. 1, no. 3, pp. 180–191, Sep. 2023. DOI: https://doi.org/10.62583/rseltl.v1i3.44.

[29] S. S. Jennifer, S. A. Islam, S. S. Koly, R. A. Tuhin, M. S. H. Khan, and M. M. Uddin, "Bilinbot: A bilingual chatbot using deep learning," in *2023 5th International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, Istanbul, Turkiye, 2023, pp. 1–6. DOI: 10.1109/HORA58378.2023.10156681.

[30] R. Sajja, Y. Sermet, M. Cikmaz, D. Cwiertny, and I. Demir, "Artificial intelligence-enabled intelligent assistant for personalized and adaptive learning in higher education," *arXiv*, 2023. DOI: https://doi.org/10.48550/arXiv.2309.10892.

[31] M. S. Salim, H. Murad, D. Das, and F. Ahmed, "Banglagpt: A generative pre-trained transformer-based model for bangla language," in *2023 International Conference on Information and Communication Technology for Sustainable Development (ICICT4SD)*, 2023, pp. 56–59. DOI: 10.1109/ICICT4SD59951.2023.10303383.

[32] S. Sarker, A. Rahman, and M. Khandakar, "Building annotated datasets for bangla nlp: Issues and directions," *Transactions in Asian Language Processing*, vol. 12, no. 1, pp. 1–15, 2023.

[33] I. M. Zulkarnain, S. B. Islam, M. Z. A. Z. Farabe, *et al.*, *Bbocr: An open-source multi-domain ocr pipeline for bengali documents*, 2023. arXiv: 2308.10647 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2308.10647.

[34] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, "Self-rag: Learning to retrieve, generate, and critique through self-reflection," in *Proceedings of the 12th International Conference on Learning Representations (ICLR)*, 2024. DOI: 10.48550/arXiv.2310.11511. [Online]. Available: https://arxiv.org/abs/2310.11511.

[35] M. Douze, A. Guzhva, C. Deng, *et al.*, "The faiss library," *arXiv preprint arXiv:2401.08281*, 2024. arXiv: 2401.08281 [cs.LG].

[36] Y. Gao, Y. Xiong, X. Gao, *et al.*, "Retrieval-augmented generation for large language models: A survey," *arXiv preprint arXiv:2312.10997*, 2024. DOI: 10.48550/arXiv.2312.10997. [Online]. Available: https://arxiv.org/abs/2312.10997.

[37] N. Nawal, S. Basak, and R. Shahriyar, "Effective retrieval-augmented generation for open domain question answering in bengali," Ph.D. dissertation, Jul. 2024. DOI: 10.13140/RG.2.2.15361.06248.

[38] P. Narayan, R. T, T. Mg, K. Krishna, and P. V, "Retrieval-augmented generation for multiple-choice questions and answers generation," vol. 259, Jan. 2025. DOI: 10.1016/j.procs.2025.03.352.