Knowledge Navigator: Revolutionizing Education throughLLMs in Generative AI

Malathi S.[1], Hemamalini S.[*,2], Ashwin M.[3], Rijo Benny[4]
[1,2,3,4]Department of Artificial Intelligence and Data Science, Panimalar Engineering College, India
Emails: malathi.raghuram@gmail.com; hemamalini.selvamani@gmail.com;ashwinm656@gmail.com; rijob2002@gmail.com

Abstract

The education landscape is shifting towards automation and digitalization to cater to the increasing demand for personalized learning experiences and more efficient teaching methods. In response to this trend, we propose the development of an integrated educational automation fusion platform that aims to overhaul educating and learning practices across various educational sectors. The integration of cutting-edge language models like GPT-3.5 and Gemini Pro in information retrieval and conversational AI has opened fresher opportunities, even within the realm of education. With its advanced features, LangChain, a powerful framework for large language models, enables seamless integration of AI-driven functionalities, including document analysis, question generation, and chatbot interaction, revolutionizing the educational landscape. Also, by harnessing the vast resources of the OpenAI API, our platform empowers educators and learners to engage in dynamic conversations with educational materials, generate personalized assessments, and gain deeper insights from complex datasets within a single forum. This single platform disseminates information on all facets of research and development in educational domain on the grounds of fusion practices and applications. This system is successful in combining multiple models for intelligent systems. On evaluation, our system was successful in generating decent performance compared to existing systems, even though they are singular modules. Overall, our platform aims to empower educators, students, and institutions to embrace the digital era of learning and unlock new avenues for fusion-based knowledge acquisition and innovation.

Keywords: Intelligent systems; Data Fusion; LangChain; OpenAI; GPT-3.5; Gemini Pro.

1. Introduction

One of the most recent advancements in the digital age is the emergence of Large Language Models (LLMs), particularly in the creation of chatbots. ChatGPT stands out as a leading example in this field, spearheading the development of such models [1]. Our work represents a significant step forward in the automation and innovation of educational processes through Large Language Models in Generative AI. As education continues to adapt to the digital age, it's embracing automation and digital tools to meet the rising demand for tailored learning experiences and more efficient teaching methods. Considering this, we envision creating a dynamic educational automation platform that promises to transform the way we teach and learn across various educational fields, ushering in a new era of innovative educational practices. It is proven that Large Language Models (LLMs) hold significant promise for enhancing education, aiding both students and educators in preparing individuals for evolving job landscapes, tailoring learning experiences, minimizing time-consuming responsibilities, enhancing accessibility and inclusivity, and offering multilingual assistance. LLMs are widely used for the purpose of abstractive summarizations. Thanks to recent breakthroughs in LLMs, abstractive summarization is poised to become even faster and more efficient than it was with earlier transformer models [2].

In our work we have utilized the cutting-edge capabilities of LangChain, a robust framework designed for large language models, facilitates effortless incorporation of AI-driven features such as document examination, query creation, and interaction with chatbots, information retrieval and its visualization thereby transforming the educational arena. Moreover, by leveraging the extensive resources of the OpenAI API, our platform empowers

Doi: https://doi.org/10.54216/FPA.160114
Received: July 07, 2023 Revised: November 10, 2023 Accepted: April 27, 2024

both educators and students to partake in dynamic dialogues with educational content, craft customized evaluations, and extract profound insights from intricate data sets.
Our system offers a range of services, such as-

- Chat with Documents
- Automatic Question-Answer Generation
- Instant MCQs Generation
- Visualization with CSVs.
- Research Bot etc., within a single forum

2. Related works

This section includes the summarization of works in the state-of-the-art that are related to our research domain.
Rabee Al-Qasem et al. [1], in their work, proposed a LLM-based chatbot aimed at providing legal assistance to Palestinian cooperatives and their members by leveraging publicly available legal documents. Through Natural Language Processing (NLP) techniques and the analysis of legal texts, the chatbot offers responses to legal inquiries. The study reports an overall accuracy of 82% and an F1 score of 79%, evaluated against answers designed by a legal expert, with an average satisfaction score of 78.3% based on user feedback. Challenges include incorrect answers and processing limits, while demonstrating potential benefits in accessibility, efficiency, and scalability, the authors highlight the need for continuous improvement to enhance accuracy and reliability, underscoring the chatbot's value as a tool for delivering reliable legal support.

Gunjan Keswani et al. [2] introduces a novel approach aimed at enhancing the efficiency of Large Language Models (LLMs) in retaining context, particularly in the context of summarization and question answering tasks. By leveraging Retrieval-Augmented Generation (RAG) techniques and ROUGE scores, the study evaluates the effectiveness of the proposed method in maintaining context relevance and generating accurate responses. The approach demonstrates promising results in improving LLM performance, with notable advantages in precise response generation and effective summarization.

Sumedha P Raut et al. [3] designed a natural language processing-based system, augmented by Large Language Models (LLMs), for automating the analysis and scoring of answer scripts, aiming to alleviate the biases and time-consuming nature of manual assessment. Techniques such as text extraction, keyword-based summarization, text preprocessing, and similarity measurement, leveraging LLMs for text understanding, are employed to generate final marks for each script. The system's accuracy is evaluated by comparing automatically assigned marks with manual assessments, showing promising results with marks largely aligning. Despite these limitations, the automated system offers potential advantages in terms of efficiency, consistency, and bias reduction in educational assessment processes.

Pathan Simran et al. [4] introduces "Chat with Documents using LLM," proposing an innovative approach where users interact with large language models (LLMs) to retrieve information from documents through natural language conversations, surpassing traditional keyword-based searches. Leveraging LLMs for natural language understanding and generation, the system aims for efficient information retrieval, context-aware responses, and privacy controls. However, challenges such as bias in responses, document format compatibility, privacy concerns, and regulatory compliance persist. Despite these drawbacks, the project promises significant utility in various domains, offering a seamless integration of human-like conversation with document search.

Md Adnan Arefeen et al. [5] proposed a cost-efficient context reduction system for minimizing expenses associated with LLM API usage in question-answering applications. LeanContext dynamically extracts key sentences from domain-specific context using reinforcement learning, reducing context while maintaining response accuracy. Compared to existing methods, LeanContext achieves comparable or superior performance with reduced costs, leveraging open-source summarization techniques. Its flexibility in adjusting to different summarization methods enhances overall system performance. While LeanContext demonstrates encouraging outcomes, further exploration into other domains and potential drawbacks is warranted.

Joshua Robinson et al. [6] explores the effectiveness of multiple-choice prompts (MCP) versus traditional cloze prompts for large language models (LLMs) in multiple choice question answering (MCQA). It introduces the concept of multiple-choice symbol binding (MCSB) ability and demonstrates that LLMs with high MCSB ability achieve significantly improved accuracy using MCP, nearly matching the state-of-the-art performance. While MCP shows promise, there are datasets where cloze prompts still outperform MCP, indicating the need for further

Doi: https://doi.org/10.54216/FPA.160114
Received: July 07, 2023 Revised: November 10, 2023 Accepted: April 27, 2024

investigation. The paper emphasizes the importance of intuitive changes in prompting techniques to enhance LLM performance and suggests future research directions in prompt engineering and model evaluation.

Lochan Basyal et al. [7] proposed a text summarization technique using various Large Language Models (LLMs), showcasing text-davinci-003 as the top performer with an accuracy of 73%. It emphasizes the significance of model size and suggests future improvements with larger-parameter models and domain-specific fine-tuning to enhance summarization quality. Furthermore, it discusses the potential applications of Generative AI across diverse business domains.

Andrew Tran et al. [8] explores the feasibility of using large language models (LLMs), particularly GPT-3 and GPT-4, to streamline the generation of multiple-choice questions (MCQs) in educational contexts. By leveraging LLMs' generative capabilities, the study aims to address the time-consuming nature of MCQ creation and the issue of recycling generic questions from question banks. The evaluation reveals promising results, with GPT-4 accurately generating answers for 78.5% of MCQs based solely on the question stem. Despite this, limitations such as imperfect performance, minimal correlation between LLM capabilities and discrimination scores, and the need for further evaluation are noted.

3. Proposed Method

The proposed work, “Knowledge Navigator: Revolutionizing Education through LLMs in Generative AI” is an innovative tool designed to streamline information management and research processes with its diverse range of modules. Users can effortlessly interact with CSV files for data visualization, ask queries and receive precise answers from a vast knowledge base through the Question-Answering System, simplify the generation of Multiple-Choice Questions with the MCQ Generator, and conduct research efficiently with the assistance of the Research Bot. Additionally, the Q&A evaluator automates the comparison between written responses and answer keys, saving time in assessment processes.

With its advanced features, LangChain, a powerful framework for large language models, enables seamless integration of AI-driven functionalities mentioned above, revolutionizing the educational landscape. Also, by harnessing the vast resources of the OpenAI API, our platform empowers educators and learners to engage in dynamic conversations with educational materials, generate personalized assessments, and gain

deeper insights from complex datasets. With “Knowledge Navigator”, users can access a comprehensive suite of tools tailored to their needs, empowering them to succeed in data management, research endeavors, and educational assessments. This system is successful in achieving the goal of an intelligent system that combines multiple cognitive learning-based fusion models within a single forum.

The architecture diagram for the proposed work is depicted in Figure 1. The proposed architecture begins by taking a document in various formats such as .pdf, .txt, .jpeg, or .csv as input. Text is then extracted from these documents using corresponding Python packages. Next, an embedding model is employed to transform the extracted text into lower-dimensional vectors, capturing semantic or contextual information in a compact form. These embeddings are stored in a vector database, such as FAISS (Facebook AI Similarity Search), for efficient retrieval. Subsequently, a prompt template is utilized to generate prompts or questions for large language models (LLMs) like GPT-3.5 or LIDA. These prompts are designed to elicit summaries of the document from the LLM. The prompts are then sent to the LLM, which generates candidate summaries or appropriate responses for each prompt. Finally, the interactive web application is created using Streamlit UI to present the results to the user.
The proposed work contains a single framework, with an integration of five different modules serving five unique use-cases for fostering the development of the educational domain. The five unique modules are as given below-

- Visualization with CSV
- Instant Report Generation
- Descriptive Answer Evaluation
- Q&A Generation
- MCQ Generation from documents

The above-mentioned modules are explained below.

Doi: https://doi.org/10.54216/FPA.160114
Received: July 07, 2023 Revised: November 10, 2023 Accepted: April 27, 2024

3.1 Visualization with CSV

This work introduces an innovative data visualization and analysis module powered by the LIDA model, aimed at revolutionizing the understanding and presentation of complex datasets. With its unique capability to generate graphs based on user-defined questions, interactive visualization features, advanced summarization tools, and the ability to create captivating infographics, this module provides a comprehensive solution for exploring, summarizing, and communicating insights from data. It provides visualizations ranging from scatter plots to bar charts, facilitating dynamic exploration and understanding. The basic pseudocode for this module is demonstrated in Algorithm 1.

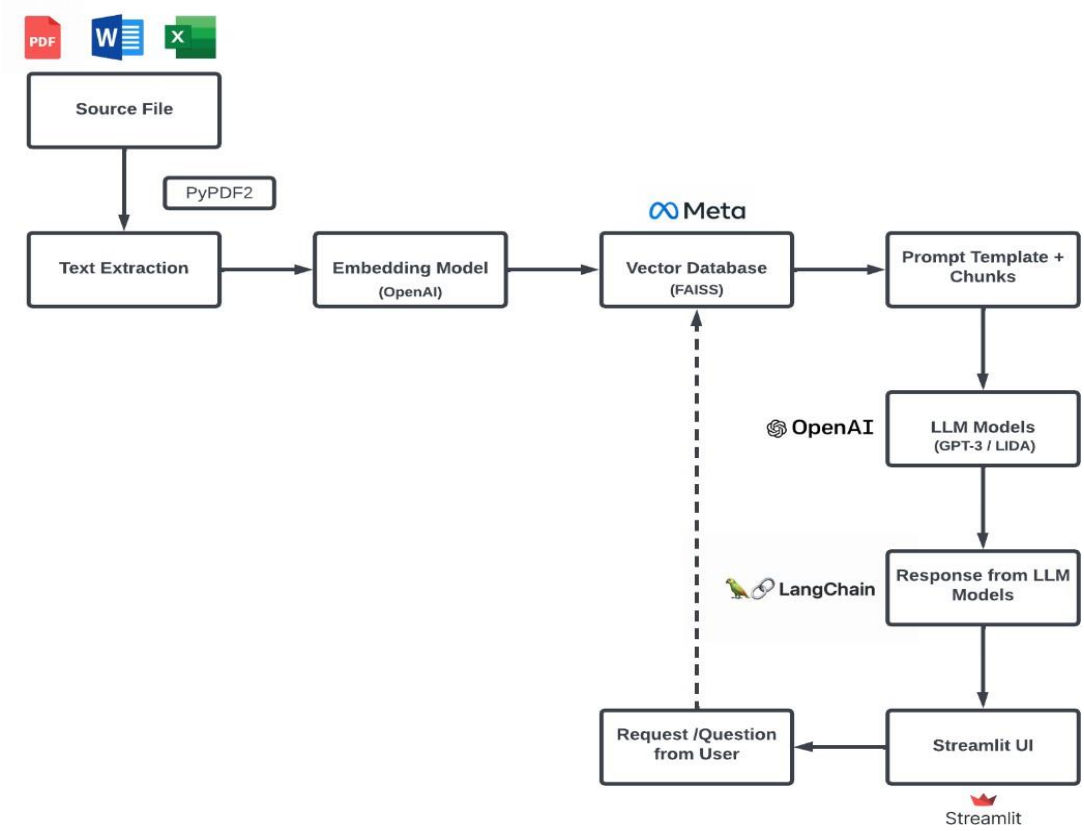

Figure 1: Architecture diagram

Algorithm 1: Visualization with CSV

Import necessary libraries
  Set up the OpenAI API key
Define class DataAnalyzer
  Method to_convert_base64_strings_to_image_objects(base64_string)
  Convert base64 string to image object
  Return image object
 	
  Method summarize_and_visualize_CSV_data(file_path)
  Read CSV data from file_path
  Summarize the data (e.g., descriptive statistics)
  Visualize the data (e.g., histograms, scatter plots)
 	
  Method generate_graph_based_on_query(file_path, user_query)
  Read CSV data from file_path
  Process user_query to filter data

Doi: https://doi.org/10.54216/FPA.160114
Received: July 07, 2023 Revised: November 10, 2023 Accepted: April 27, 2024

```
    Generate graph based on filtered data (e.g., bar chart, line plot)

Create an instance of DataAnalyzer

Display_sidebar_menu(options)
    Show options: "Summarize" and "Question based Graph"
    Accept user input for menu selection
    Return user's choice

If selected_option is "Summarize"
    Allow users to upload a CSV file
    Call DataAnalyzer.summarize_and_visualize_CSV_data(file_path)

If selected_option is "Question based Graph"
    Allow users to upload a CSV file
    Accept user input for query
    Call DataAnalyzer.generate_graph_based_on_query(file_path, user_query)
Continue_loop_until_exit_condition_is_met
```

This algorithm outlines a Python script built using Streamlit, aimed at simplifying data analysis and visualization tasks through an intuitive web interface. It begins by importing necessary libraries such as pandas for data handling, matplotlib for plotting, and Streamlit for UI creation. With a structured DataAnalyzer class, the script provides methods for converting base64 strings to images, summarizing CSV data, and generating graphs based on user queries. The main function orchestrates user interaction, presenting a sidebar menu for choosing between data summarization or query-based graph generation. Users can upload CSV files to perform statistical summaries and visualize data distributions or input queries to dynamically generate graphs tailored to their questions. This script encapsulates a user-friendly approach to data exploration, offering both efficiency and flexibility in data-driven decision-making.

3.2 Question Answering System

The Q&A Generation system is an intelligent tool and a module within our project designed to streamline the process of creating questions and their corresponding answers based on provided documents. As described in Algorithm 2, utilizing advanced language models and natural language processing techniques, the system can analyze the content of PDF files and generating insightful questions that cover key aspects of the material. Additionally, it formulates accurate answers to these questions, ensuring a comprehensive understanding of the document's content. The generated question and answers are saved in the form of CSV file format on download request.

Algorithm 2: Q & A Generation

```
Initialize the PDFQuestionAnsweringSystem class

Define a method process_PDF_file(file_path)
    Process the PDF file located at file_path
    Extract text from PDF
    Preprocess text (e.g., remove noise, extract paragraphs)
    Return preprocessed text

Define a method setup_question_answer_pipeline()
    Set up the question-answer pipeline (e.g., using a pre-trained model)

Define a method generate_CSV_file_with_questions_and_answers(text)
    Generate questions and answers from the preprocessed text
    Store questions and answers in a CSV file format
    Return the path to the generated CSV file

Define a method allow_user_to_upload_PDF_file()
    Display a file upload button for the user
    Accept uploaded PDF file
```


Doi: https://doi.org/10.54216/FPA.160114
Received: July 07, 2023 Revised: November 10, 2023 Accepted: April 27, 2024

```
Define a method generate_questions_and_answers_if_file_uploaded(file)
    If a file is uploaded, call process_PDF_file method
    Call setup_question_answer_pipeline method
    Call generate_CSV_file_with_questions_and_answers method
    Display a link to download the generated CSV file

Define a method clean_up_temporary_files()
    Remove any temporary files created during processing
```

This algorithm outlines a Python script leveraging Streamlit to develop a user-friendly interface for a PDF Question Answering System. The script initializes a PDFQuestionAnsweringSystem class and defines methods to handle various functionalities. Firstly, a method is established to handle PDF file processing, saving the uploaded PDF temporarily, extracting questions and answers using the PDFQuestionAnsweringSystem, and subsequently cleaning up temporary files. Another method is initiated which configures the question-answer pipeline, allowing for the utilization of pretrained models or custom logic. Additionally, a function is defined to produce a CSV file containing the generated questions and answers. The main function serves as the core orchestrator of the Streamlit application.

3.3 MCQ Generator

In the landscape of education where formative assessment plays a pivotal role, the demand for efficient creation of Multiple-Choice Questions (MCQs) has surged. Recognizing this need, this model, powered by generative AI's LLM, offers a transformative solution. By automating the MCQ generation process from provideddocuments, it empowers educators with the ability to swiftly produce tailored assessments. This not only saves significant time and effort but also ensures the assessments align closely with the course material, facilitating enhanced learning outcomes for both educators and learners alike.

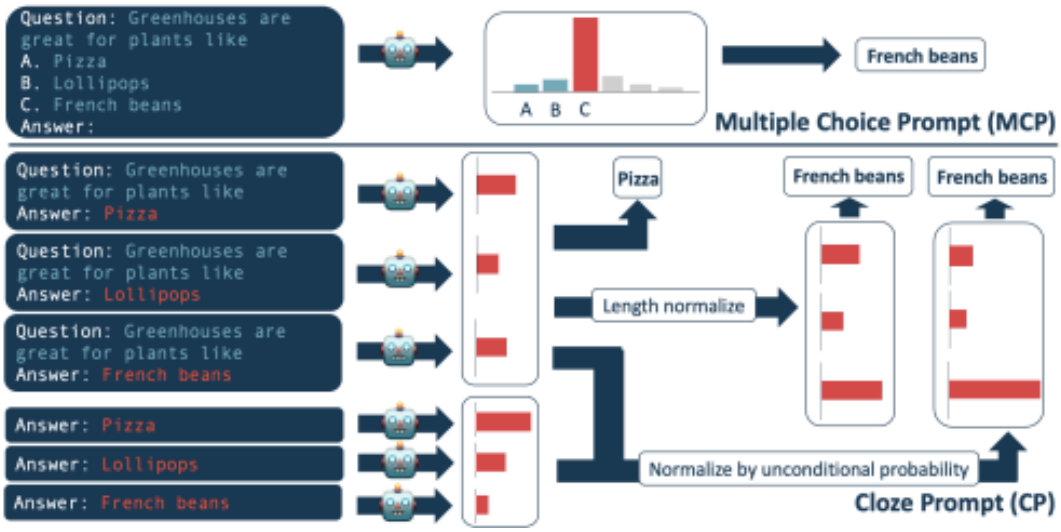

Figure 2: MCQ prompt visualization for an example question

Figure 2 demonstrates the working flow of this module for an example question. Basically, there is a hypothesis that there are underlying issues with the widespread method of Multiple-Choice Question Answering (MCQA) for Large Language Models (LLMs), commonly known as "cloze prompting" (CP). In cloze prompting, an LLM receives a question, scores each candidate answer independently, and selects the answer option with the highest probability. To address potential biases caused by common or uncommon tokens or varying sequence lengths, Brown et al. (2020) employed two normalization methods. In one, the sequence's probability is normalized by taking the nth root, represented as Equation (1)

Doi: https://doi.org/10.54216/FPA.160114
Received: July 07, 2023 Revised: November 10, 2023 Accepted: April 27, 2024

$$\mathbf{P(x1, x2, ..., x_n) = \sqrt[n]{\pi}.\, P(x_i),}\ \text{where i is from 1 to n} \quad \text{.....(1)}$$

In the other, the answer's probability is normalized by the unconditional probability of the answer, expressed as P(completion|context) / P(completion|answer context), where the answer context is defined as the string "Answer: "

To explore how the wording of prompts affects responses, we crafted and assessed various prompts for generating isomorphic multiple-choice questions (MCQs):

Prompt 1: Generate one correct answer and three distractors for the given MCQ. Indicate the correct answer with an (X).

Prompt 2: Produce three relevant distractors and one correct answer for the question stem. Ensure only one answer is correct, marked with an (X).

Prompt 3: Formulate a set of responses for the question stem, including one correct answer and three distractors. Ensure the distractors are distinct concepts but still pertinent to the problem. Mark the correct answer with an (X).

Here, we utilized the 'text-davinci-003' model with a temperature setting of '0.7'. Temperature controls response variability; higher values yield more diverse responses, while lower values result in more consistent ones.

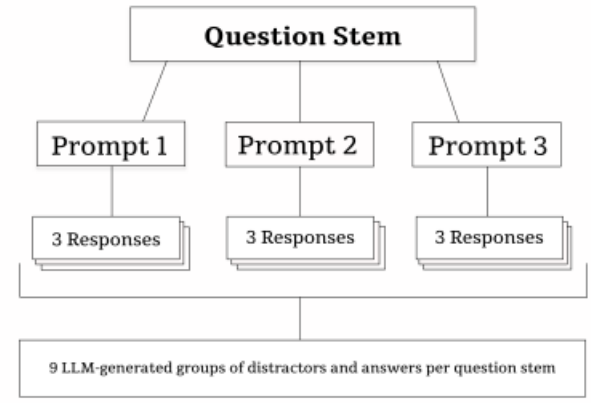

Figure 3: Answer choice generation for a question stem.

Figure 3 given above demonstrates the generation of answer choices for a question stem using LLM using three prompts. This resulted in nine isomorphic MCQs for each question stem.

The general flow can be as follows-

Algorithm 3: MCQ Generator
Start Import necessary libraries and modules Load environment variables Assign your OpenAI API key to a variable Initialize ChatOpenAI with the assigned OpenAI API key and specify model parameters Define the template for generating quiz prompts Initialize a PromptTemplate object for template quiz generation Initialize an LLMChain object for quiz generation using LLM Define the template for quiz evaluation prompts

Doi: https://doi.org/10.54216/FPA.160114
Received: July 07, 2023 Revised: November 10, 2023 Accepted: April 27, 2024

```
    Initialize a PromptTemplate object for template for quiz evaluation

    Initialize an LLMChain object for quiz evaluation using LLM

    Initialize a SequentialChain object combining quiz generation and evaluation steps

End
```

Algorithm 3 begins by importing essential libraries and loading environment variables. Next, it assigns an OpenAI API key, initializing a ChatOpenAI instance with model parameters. Following this, templates are defined for generating quiz prompts and evaluating quizzes, encapsulated within PromptTemplate objects. These templates are used to initialize LLMChain objects for both quiz generation and evaluation. Ultimately, a Sequential Chain is constructed, incorporating both generation and evaluation steps. This structured approach lays the groundwork for an automated process to generate and assess multiple-choice quizzes, streamlining educational content creation.

3.4 Research Bot

The Research Bot is another interesting module which is designed to swiftly generate comprehensive research reports based on user prompts. Integrated with tools accessing Wikipedia, DuckDuckGo, and YouTube, the bot dynamically generates academic introductions, statistical facts, recent publications, recommended books, and YouTube links relevant to the input. Utilizing an OpenAI language model, users can fine-tune report generation with adjustable temperature settings. The Streamlit interface offers a seamless experience, allowing users to input prompts, configure model settings, and initiate report generation effortlessly. With its intuitive design and multifaceted research capabilities, the Research Bot streamlines the process of acquiring and synthesizing information for academic or informational purposes. The basic pseudocode for this module is stated in Algorithm 4.

Algorithm 4: Research Bot

```
Start
   Import necessary libraries and modules

Define class ResearchBot
   Constructor method __init__(self, default_settings)
      Initialize ResearchBot object with default settings

   Method generate_YouTube_links(user_input)
      Use VideosSearch object to search for videos on YouTube based on user_input
      Iterate through search results and display title and link of each video

   Method generate_research_reports(user_input)
      Utilize various research tools such as Wikipedia, DuckDuckGo, and YouTube
      Generate different sections of the research report using these tools

   Method run()
      Set up the interface
      Trigger the generation of research based on user input

End
```

The ResearchBot class is designed to facilitate research report generation based on user input. It first imports necessary libraries and modules for its functionality, including Streamlit for the user interface and various tools for language processing, web scraping, and API requests. The class defines methods for generating YouTube links from user input and conducting research using tools such as Wikipedia, DuckDuckGo, and YouTube. By utilizing these tools, the research generation method generates different sections of the research report, including introduction, quantitative facts, recent publications, recommended books, and YouTube links, all based on the user's input. The run method serves as the main entry point, setting up the Streamlit interface and triggering research generation upon user interaction. Overall, the ResearchBot provides a comprehensive solution for conducting research and presenting findings in an organized manner.

Doi: https://doi.org/10.54216/FPA.160114
Received: July 07, 2023 Revised: November 10, 2023 Accepted: April 27, 2024

3.5 Q and A Evaluation

This module is designed to assess the similarity between a provided descriptive answer and an actual script, enabling thorough evaluation and feedback generation. Leveraging advanced Optical Character Recognition (OCR) capabilities through Tesseract, the module extracts text from uploaded images containing the answer scripts. It then utilizes advanced Language Model (LLM) techniques to process and analyze the extracted text, comparing it with the content of the actual script. This comparison is performed using sophisticated algorithms to compute a similarity score, providing users with a quantitative measure of accuracy.

Table 1: Marking scheme for score S1

No. of words w.r.t x	Score (S_1)
Less than (x-30%)	0.5
Between (x-30%) & (x-10%)	0.8
More than (x+10%)	0.8
X+-10%	1

Table 1 depicts that in this model, a portion of the overall score is influenced by the S_1 which is actually the length of the answer. The table above outlines the marking scheme and the associated thresholds. The optimal word count expected for an answer is denoted as "x". The score (S_1) from this aspect constitutes 5% of the total score for the answer. In this model, the organization and grammar of the answer are assessed, resulting in a score (S_2). This score (S_2) accounts for 5% of the overall score. Imagine the examiner has provided a list of essential keywords. This aspect of the model examines whether the student has incorporated these keywords into their response. It not only verifies the presence of these keywords but also considers synonyms. Essentially, it operates on the principle of a word bank and employs basic if-else computation methods. If a student has included y_1 keywords out of a total of y, the score is determined as follows in Equation (2):

$$\mathbf{S_3=y_1/y} \quad \ldots\ldots\ldots\ldots (2)$$

This score (S_3) constitutes 10% of the total score. The important factor of this evaluation system lies in determining the similarity index between the submitted answer and the ideal answer provided by the examiner. For calculating the Similarity Index (S_4), we use the cosine similarity method, which is commonly used for comparing the similarity of documents. The cosine similarity is generally used to measure the cosine of the angle between two vectors, representing the documents in a high-dimensional space.
Let's denote the submitted answer vector as $\vec{A}$ and the ideal answer vector as $\vec{B}$. Then, the cosine similarity cos_sim is calculated as in Equation 3 given below:

$$\cos\_sim = \frac{\boldsymbol{A.B}}{||\boldsymbol{A}|| \, ||\boldsymbol{B}||} \quad \ldots\ldots\ldots\ldots (3)$$

Where:

- $\boldsymbol{A} \cdot \boldsymbol{B}$ is the dot product of the vectors **A** and **B** .
- $||\boldsymbol{A}|| \, ||\boldsymbol{B}||$ are the Euclidean norms of vectors **A** and **B.**

The resulting cos_sim value ranges from -1 to 1, where 1 indicates perfect similarity, -1 indicates complete dissimilarity, and 0 indicates no similarity. Once cosine similarity score is obtained, we use it as the similarity index (S_4) in our scoring model, which constitutes 80% of the total score.
Now to calculate the total score:

$$S=S_1+S_2+S_3+S_4 \quad \ldots\ldots\ldots (4)$$

Equation 4 depicts the calculation of total score baseed on similarity, where:

S_1 is the score from the aspect of answer size (5% of total score S).
S_2 is the score from the aspect of language of the answer (5% of total score S).
S_3 is the score from the aspect of presence of keywords (10% of total score S).
S_4 is the score from the aspect of similarity index (80% of total score S).
The weights are given based on user's preference.

Doi: https://doi.org/10.54216/FPA.160114
Received: July 07, 2023 Revised: November 10, 2023 Accepted: April 27, 2024

Table 2: Scoring parameters and assigned weights

Evaluating Factors	Assigned Weights
Answer Size (S1)	5%
Language of the answer (S2)	5%
Presence of keywords (S3)	10%
Similarity Index (S4)	80%
Total score (S)	100%

Table 2 describes the scoring parameters and their corresponding weights that all adds up to give a score of 100%. The general flow for the module is explained in Algorithm 5.

Algorithm 5: Descriptive Answer Evaluation

```
Start
   Initialize the PDFComparator class
Define method compare_PDF_files_using_LLM(pdf_file1, pdf_file2)
   Compare two PDF files using Language Model (LLM)
   Return similarity score between the two PDFs
Define method run_Streamlit_application_for_uploading_and_comparing_PDFs()
   Set up the Streamlit application title and file uploaders for two PDF files

   If both PDF files are uploaded:
      Write the contents of the PDF files to temporary files
      Call compare_PDF_files_using_LLM method to compare the PDFs
      Display the similarity score
      Provide feedback based on the similarity score:
         If similarity == 1, display a success message indicating identical PDFs
         If similarity > 0.75, display a success message indicating highly similar PDFs
         If similarity > 0.5, display an info message indicating somewhat similar PDFs
         Otherwise, display a message indicating low similarity

Define method handle_exceptions_and_display_error_messages(exception)
   Handle exceptions and display error messages if PDF processing fails

Define method main()
   Run the PDF Comparator application if the script is executed directly
   Catch exceptions and call handle_exceptions_and_display_error_messages method if any occur
Call main()
```

Algorithm 5 outlines a Python script utilizing Streamlit to create a PDF Comparator module. It begins by initializing the PDFComparator class responsible for comparing PDF files. Within this class, a method is defined to utilize Cosine Similarity for comparison. The script also defines a method to orchestrate the Streamlit application, setting up the interface for users to upload two PDF files for comparison. Upon successful upload of both files, the script compares them using Cosine Similarity and displays the similarity score. Feedback is provided based on this score, with messages indicating identical, highly similar, somewhat similar, or low similarity between the PDFs. Exception handling ensures that errors during PDF processing are appropriately addressed. Finally, if the script is executed directly, the PDF Comparator application is run. This comprehensive approach ensures a user-friendly experience for comparing PDF files' content similarity.

General Workflow of the proposed system

1. Login to the system through user authentication process.
2. Choose the module based on the task you want to perform.
3. Input the document in the required format.
4. Conversion of the extracted text into a numerical representation using an embedding model.
5. Storing the embedding vectors in a vector database (e.g., FAISS) for efficient similarity search.

Doi: https://doi.org/10.54216/FPA.160114
Received: July 07, 2023 Revised: November 10, 2023 Accepted: April 27, 2024

6. Design a prompt template to formulate a query based on user's request or question, ensuring it's informative and useful to the LLM.
7. Pass the query to one or more LLM models (e.g., GPT-3 or LIDA).
8. Process the output from LLM models using LangChain framework.
9. Display the final output to the user through a Streamlit UI

4. Results and Discussion

The LLM model that is used here for the system is Generative Pre-trained Transformer (GPT). The reason for choosing this LLM model above other models is pretty straight forward on certain research. GPT-3.5 is like a souped-up version of its predecessor, GPT-3. It's been fine-tuned to be faster and more accurate, making it great for quickly generating responses to all sorts of questions and prompts. You can even tweak it to better fit certain tasks or data sets. Plus, it can handle big piles of data without breaking a sweat, churning out high-quality text for everything from chatting to writing stories. But, like any tool, it's not perfect—it can sometimes get things wrong or make stuff up. Still, it's pretty impressive, scoring higher than many other models in tests where humans evaluate its performance.

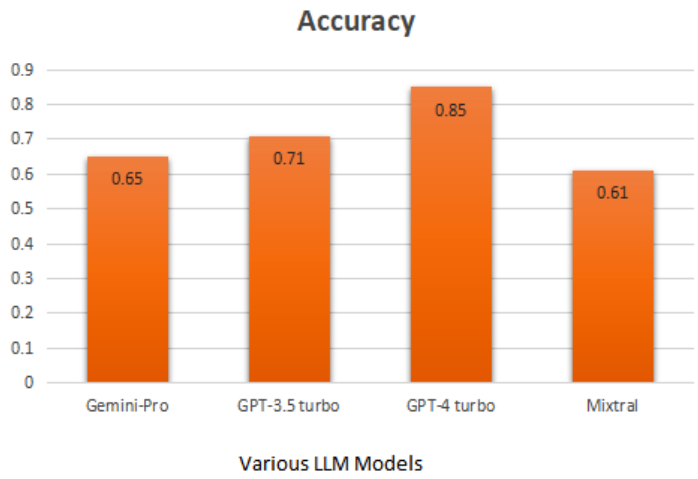

Figure 4: Model comparison based on accuracy

The above Figure 4 demonstrates a research conducted by Carnegie Mellon University (CMU) for examination of the Gemini models' impact by comparing their performance with that of GPT-3.5/GPT-4. They disclosed all the parameters used for the evaluation, ensuring transparency and reproducibility in their analysis. It was easily resulted that the performance of the GPT models was better than that of the other models namely Gemini of Google and the newest Mixtral model. However, GPT-4 gives better performance than the latter in some cases, we have used GPT-3.5 in some modules due to its better cost efficiency compared to the other.

Table 3: LLM comparison based on different parameters

Model	Provider	Open-Source	Params	Fine-Tuneability
GPT-3.5-Turbo	Open AI	No	175B	Yes
GPT-4	Open AI	No	1.76T	No
Gemini Pro	Google	No	175B	No
Mistral	Mistral AI	Yes	7.3B	Yes

From Table 3, from our analysis, it can be understood that the LLM model i.e., GPT 3.5 Turbo and GPT 4 give comparatively better performance on a whole compared to other models taking the underlying parameters into consideration. Thus, the proposed system offers a range of features. All of the five modules were successfully integrated into our system. Some of the module implementations are demonstrated below. Figure 5 demonstrates

Doi: https://doi.org/10.54216/FPA.160114
Received: July 07, 2023 Revised: November 10, 2023 Accepted: April 27, 2024

how an output is generated on the screen for a user prompt regarding a particular dataset. Figure 6 depicts the demonstration of how the frontend of the web looks like for the PDF Question Answering System and Figure 7 is the outlook of how the questions and answers are generated in a new CSV file.

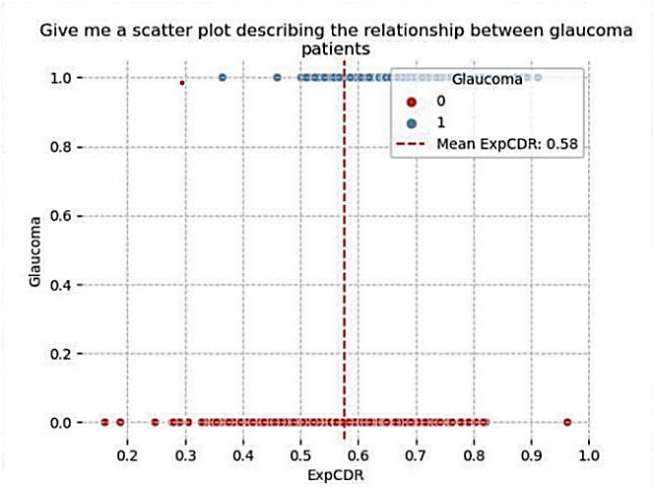

Figure 5: Data Visualization with CSV module

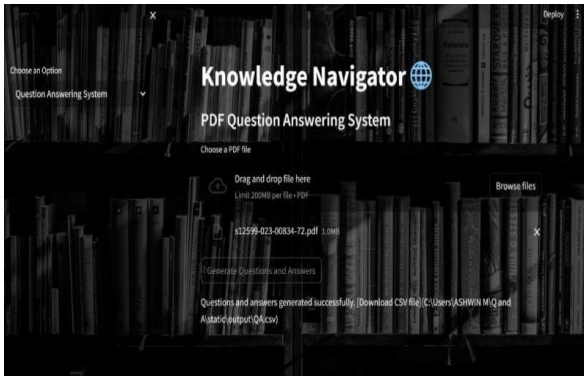

Figure 6: PDF Question Answering System Frontend

QUESTIONS	ANSWERS
1. How does generative AI impact industries and create new job profiles?	Generative AI is expected to have significant economic implications across various industries by increasing efficiency and productivity through tasks automation. This can lead to cost reduction, growth, and innovation. It may also create entirely new job profiles, such as prompt engineers, as mentioned in the research (Strobelt et al. 2023).
2. In what ways can generative AI systems assist in digital knowledge management, automated knowledge discovery, and personalized knowledge delivery?	Generative AI systems can assist in digital knowledge management by automating knowledge discovery based on large amounts of unstructured distributed data. They can identify new product combinations, improve knowledge sharing by automating the process of creating, summarizing, and disseminating content like creating wikis and FAQs in different languages. Additionally, generative AI can enable personalized knowledge delivery to individual employees based on their specific needs and preferences, providing recommendations for specific training material.
3. What are the economic implications of generative AI on industries, such as cost reduction, increased productivity, and opportunities for growth and innovation?	Generative AI is expected to have significant economic implications across various industries and markets. It can increase efficiency and productivity by automating tasks previously done by humans, such as content creation, customer service, and code generation. This automation can lead to cost reduction, open up new opportunities for growth and innovation, and contribute to economic gains. For example, AI-based translation between languages has shown economic benefits.
4. How does generative AI affect Business Process Management (BPM) by automating tasks and improving efficiency?	Generative AI can significantly impact Business Process Management (BPM) by automating routine tasks, generating process descriptions, and supporting innovative process design initiatives. It can help businesses identify and understand different stages of a process , generate new ideas for process improvement , and learn complex relationships in dynamic process.

Figure 7: Question-Answer generation format

Doi: https://doi.org/10.54216/FPA.160114
Received: July 07, 2023 Revised: November 10, 2023 Accepted: April 27, 2024

For the evaluation of our LLM model across modules, we have taken the ROUGE scores into consideration. The ROUGE (Recall-Oriented Understudy for Gisting Evaluation) score is commonly used to evaluate the quality of summaries or generated text produced by language models, including large language models (LLMs) like GPT (Generative Pre-trained Transformer). ROUGE measures the overlap between the generated text and reference summaries or documents based on various metrics such as precision, recall, and F1- score. We have calculated ROUGE scores for unigram (ROUGE-1), bigram (ROUGE-2), and Longest Common Subsequence (ROUGE-L) metrics. The precision, recall, and F1-score are provided for each metric, indicating the overlap between the generated response and the reference response. The summary of the ROUGE scores is depicted below in Table 4.

Table 4: Summary of ROUGE scores

	Precision	Recall	F1-score
ROUGE-1	0.85	0.90	0.87
ROUGE-2	0.75	0.80	0.77
ROUGE-3	0.80	0.85	0.82

The proposed system was successful in creating a system generating better ROUGE scores compared to the existing models.

5. Future Enhancement

As future enhancement, we intend to test more elaborate prompting strategies such as enhanced prompt chaining. Even more modules can be integrated into the system for providing much more benefits. In case of the descriptive answer evaluation, more features could be added for evaluation on language-oriented subjects as well as mathematical subjects. Establishing mechanisms for collecting user feedback, monitoring system performance, and iterating on features based on user insights and evolving requirements will be an additional bonus. Creating a system, such that it has separate spaces for both educators and learners can add to the benefit of the work

6. Conclusion

In conclusion, this workt represents a significant step forward in the automation and innovation of educational processes. From generating questions and answers from documents to facilitating interactive chatbot interactions with educational materials, our platform empowers educators and learners alike to access information, analyze data, and generate insights with ease. Additionally, features like the MCQ generator and report bot provide valuable tools for assessment and feedback, enabling educators to evaluate progress and make data-driven decisions. We were successful in creating a model with better ROUGE scores compared to the existing works done in this domain. As we continue to refine and expand our platform, we envision it becoming an indispensable resource for educators, students, and institutions seeking to embrace the digital era of education.

References

[1] Rabee Al-Qasem, Banan Tantour, Mohammed Maree , “Towards the Exploitation of Llm-Based Chatbot for Providing Legal Support to Palestinian Cooperatives”, arXiv:2306.05827v1 [cs.CL] , Jun 2023.

[2] Gunjan Keswani, Wani Bisen, Hirkani Padwad, Yash Wankhedkar, Sudhanshu Pandey, Ayushi Soni, "Abstractive Long Text Summarization using Large Language Models", International Journal of Intelligent Systems and Applications in Engineering, ISSN:2147-67992.

[3] Prof. Sumedha P Raut, Siddhesh D Chaudhari, “Automatic Evaluation of Descriptive Answers Using NLP and Machine Learning”, International Journal of Advanced Research in Science, Communication and Technology (IJARSCT), Volume 2, Issue 1, March 2022.

[4] Pathan Simran, Raykar Vaishnavi, Pandarkar Vaibhav, Kadam Arjun, “Chat With Documents Using Large Language Model (LLM)”, International Journal of Advance Research and Innovative Ideas in Education , Vol-9 Issue-6 2023, IJARIIE-ISSN(O)-2395-4396

[5] Md Adnan Arefeen, Biplob Debnath, Srimat Chakradhar. “LeanContext: Cost-efficient domain-specific question answering using LLMs”, Natural Language Processing Journal, 2024

Doi: https://doi.org/10.54216/FPA.160114
Received: July 07, 2023 Revised: November 10, 2023 Accepted: April 27, 2024

[6] Joshua Robinson, David Wingate, “Leveraging Large Language Models for Multiple Choice Question Answering”, The Eleventh International Conference on Learning Representations, 2023.Md Adnan Arefeen, Biplob Debnath, Srimat Chakradhar, “LeanContext: Cost-efficient domain-specific question answering using LLMs”, Natural Language Processing Journal, Elsevier, Volume 7, June 2024, 100065.
[7] Lochan Basyal, Mihir Sanghvi, "Text Summarization Using Large Language Models: A Comparative Study of MPT-7b-instruct, Falcon-7b-instruct, and OpenAI Chat-GPT Models", arXiv:2310.10449v2, 17 Oct 2023
[8] Andrew Tran, Kenneth Angelikas, Egi Rama, “Generating Multiple Choice Questions for Computing Courses using Large Language Models”, 2023 IEEE Frontiers in Education Conference (FIE), 2023.
[9] Jing Wei, Sungdong Kim, Hyunhoon Jung,Young-Ho Kim “Leveraging Large Language Models to Power Chatbots for Collecting User Self-Reported Data”, arXiv:2301.05843v1 [cs.HC],14 Jan 2023.
[10] Sathya Preiya V, Kumar VDA. Deep Learning-Based Classification and Feature Extraction for Predicting Pathogenesis of Foot Ulcers in Patients with Diabetes. *Diagnostics*. 2023; 13(12):1983. https://doi.org/10.3390/diagnostics13121983.
[11] Balakrishnan C, Ambeth Kumar VD. IoT-Enabled Classification of Echocardiogram Images for Cardiovascular Disease Risk Prediction with Pre-Trained Recurrent Convolutional Neural Networks. Diagnostics (Basel). 2023 Feb 18;13(4):775. doi: 10.3390/diagnostics13040775. PMID: 36832263; PMCID: PMC9955174.
[12] Hemamalini, Selvamani, and Visvam Devadoss Ambeth Kumar. 2022. "Outlier Based Skimpy Regularization Fuzzy Clustering Algorithm for Diabetic Retinopathy Image Segmentation" *Symmetry* 14, no. 12: 2512. https://doi.org/10.3390/sym14122512.
[13] Kumar, V.D.A., Sharmila, S., Kumar, A. *et al.* A novel solution for finding postpartum haemorrhage using fuzzy neural techniques. *Neural Comput & Applic* **35**, 23683–23696 (2023). https://doi.org/10.1007/s00521-020-05683-z
[14] V D Ambeth Kumar;Manish Raghuraman;Abhishek kumar;Mamoon Rashid;Saqib Hakak;Praveen Kumar Reddy, "Green-Tech CAV: Next Generation Computing for Traffic Sign and Obstacle Detection in Connected and Autonomous Vehicles" , IEEE Transactions on Green Communications and Networking, 2022 (InPress). (Q1)
[15] Abhishek Kumar, Kamred Udham Singh, Visvam Devadoss Ambeth Kumar, Tapan Kant, Abdul Khader Jilani Saudagar, Abdullah Al Tameem, Mohammed Al Khathami, Muhammad Badruddin Khan, Mozaherul Hoque Abul Hasanat, Khalid Mahmood Malik, " Robust Watermarking Scheme for NIfTI Medical Images", Vol.71, No.2, 2022, pp.3107-3125, doi:10.32604/cmc.2022.022817
[16] V.D.Ambeth Kumar and M.Ramakrishan (2013), “Temple and Maternity Ward Security using FPRS” in the month of May for the Journal of Electrical Engineering & Technology (JEET) ,Vol. 8, No. 3, PP: 633-637.

Doi: https://doi.org/10.54216/FPA.160114
Received: July 07, 2023 Revised: November 10, 2023 Accepted: April 27, 2024