FreshmenHub: An FAQ Chatbot for First-Year Students at Habib University

Syed Muhammad Ather Hashmi
School of Science and Engineering
Habib University
Karachi, Pakistan
sh07554@st.habib.edu.pk

Sidra Aamir
School of Science and Engineering
Habib University
Karachi, Pakistan
sa07316@st.habib.edu.pk

Abdullah Junejo
School of Science and Engineering
Habib University
Karachi, Pakistan
aj07154@st.habib.edu.pk

***Abstract*—Freshmen students often feel confused and lost when they enter university, and a lot of their time is spent getting their queries answered by different departments, administrators or instructors. This can be tedious for both sides and hence emphasizes the need for a more effective approach. To solve this, we propose a chatbot that uses deep learning models, trained on data specifically gathered from Habib University, to address common questions freshmen have. This report looks at similar work done in the area, discusses the models we used, and reviews the results. We also explain the process we followed to gather and prepare the dataset for our chatbot.**

I. Introduction

With recent advancements in Artificial Intelligence and Deep Learning, the roles and need for human interaction have diminished. With the emergence of Chatbots, organisations are leaning toward incorporating them into their respective frameworks for automating certain tasks which otherwise, require human resources. These AI and Deep-learning-based, automotive responsive agents leverage the use of Natural Language Processing (NLP) to address the prompts provided to them, giving answers based on its training corpus. They are mainly used as an interactive system for information retrieval, by organizations to enhance user experience [1].

A chatbot's main role is to provide answers to the queries asked to it based on its design and functionality, providing a 24 hours, 7 days customer service, eliminating the need for human interaction and providing customized answers as per the user's queries [2].

Universities are also such places, where the need for constant information retrieval never ends. With various departments, the struggle to find the correct response is tedious and often time-consuming. A chatbot in this situation can relieve the load on both the departments and the students respectively. Even though every university has a website, and a prospectus, explaining the functions, responsibilities and assistance provided by the departments, it is common for students to face difficulty in finding the answers to their specific queries [3]. Our solution presents a Frequently-Asked Questions chatbot which deals with answering the intriguing questions surfacing the minds of freshmen students of Habib University, Pakistan specifically.

II. Research Question

The research question that we are addressing is to develop a chatbot using deep learning that caters to the queries and concerns of a freshman student enrolled in Habib University specifically. This means that "given a question, the chatbot will output an answer related to the question".

The transition from high school to university can be quite overwhelming for students. Upon entering the university as freshmen, they often feel confused and have many questions related to various procedures and departments. Even after spending one or two semesters at the university, it is challenging to say that every freshman will be aware of all the departments and their functions. Many students hesitate to reach out to the respective departments due to shyness or conflicting schedules. Additionally, contacting these departments can be a tedious task, as it typically involves sending emails and scheduling appointments. This process consumes time and can be quite inconvenient for students who are already trying to adjust to a new environment. In most cases, the concerns have straightforward answers for which scheduling appointments does not seem feasible. Therefore, a solution is needed that provides easy access to the information and answers to the problems students face, eliminating the lengthy process currently in place.

III. Literature Review

In [4], a novel dataset was created by surveying the students of Indira Gandhi Delhi Technical University for Women, Delhi (IGDTUW). A total of 75 queries were collected from 70 participants. Then the team formed carefully curated answers and stored the 75 query and answer pairs in a JSON format, with the keywords of the questions as tags. The next step is to pre-process the raw data and clean it. For that, Natural Language Processing (NLP) Python libraries such as Spacy and NLTK were used, followed by tokenization and padding of the data. The approach used in [4] relies on contrasting a feed-forward neural network and a Bidirectional Long and Short Term Memory (BiLSTM). The Feedforward neural network contained an embedding layer and three dense layers. In the BiLSTM approach, after the embedding layer, a BiLSTM layer was added, followed by two dense layers. This model was

trained to fit 550 epochs. Adam optimizer was used in both cases. Model 1 involving the feed-forward neural network performed better than the BiLSTM approach giving precise responses. However, the model is not able to perform well on the validation data giving a 48.27% accuracy while it gave 80% accuracy on the training set, leading to a case of overfitting. This experiment had its limitations as the data set was too small and implementation of techniques such as Fine-Tuning and data augmentation could prevent overfitting.

Similar work was done in [5] but for the Amharic Language. To collect the data, a questionnaire was spread to the students of Mekelle University and Aksum University, with 80 students participating in it, giving 850 pattern queries. The data was translated to Amharic Language using Google Translate and Language Experts. Further, it was reduced to cover 60 topics and stored in a JSON format, where the JSON consists of the tags (topic), patterns, responses etc. Using Facebook Messenger as an interface and a FLASK webhook to communicate with the chatbot model, the chatbot developed in this paper was created to comprehend and react to a range of Amharic text inputs. It was then hosted in a Heroku web server to provide real-time communication. The chatbot model used stemming, tokenization, and stop word removal as preprocessing techniques, overall, this chatbot is a text classifier and compares the usage of Support Vector Machines, Multinomial Naïve Bayes, and a sequential Deep Neural Network model from Tensorflow and Keras Python libraries. The first input layer contains the same number of features (vectors) that the classifier was trained with; the second and third dense hidden layers include 128 and 64 hidden neurons, respectively; the last layer is an output layer that contains the same number of intents as the output intent predictor. Additionally, [5] used softmax for the fourth layer, which is the output layer, and ReLU for the activation function of the two hidden layers. During experimentation. 20% dataset was used for testing. The experimental evaluation showed that the DNN classifier outperformed the other types of classifiers, with a 91.55% accuracy rate, 85.98% precision, 87.13% recall, and 85.23% F-1 score. This chatbot had a user satisfaction level of 86.2%. Future work planned by the authors is to incorporate sentiment analysis, integrating voice recognition and support for a large user base and other Ethiopian languages.

Moreover, [6] developed a chatbot for College Enquiry, reducing the load on the departments. They preprocess the data using similar techniques as above such as tokenization and lemmatization using NLP python libraries such as NLTK. For the chatbot model training, [6] used a recurrent Neural Network, called Long and Short Term Memory (LSTM). As this is a fully functional application, their chatbot also uses REST APIs and FLASK Framework to connect the backend with the front end. This paper achieved an accuracy of 99% after training it for 200 epochs. For future work, the authors plan to develop chatbots specific for student admissions, course recommendations etc. It is important to note that no information about data acquisition was shared in [6].

In a related study [7], a chatbot called ParsyBot was developed for Baskent University, Turkey, to assist students with queries regarding regulations, admissions, scholarships, departments, and other university-related information. The dataset for this project was curated specifically to include 688 question-answer pairs and long documents related to the university's regulations and facilities, totalling 25,741 words. A pre-trained BERT model was fine-tuned on this dataset. Both quantitative and qualitative tests were conducted to evaluate the model's performance. Quantitative evaluations used metrics such as BLEU, METEOR, and ROUGE-N. The model achieved promising results, with a METEOR score of 0.81 and a ROUGE-1 score of 0.24. Notably, ParsyBot outperformed ChatGPT in BLEU-n and ROUGE metrics, particularly excelling in ROUGE-L, which measures the longest subsequence match between the ground truth and generated responses, indicating that ParsyBot's responses were closely aligned with the reference answers.

As per another study [8], researchers at Macau Polytechnic University developed a chatbot to handle questions commonly asked during the university's open day. They created a dataset from frequently asked questions, sourced from the university's website, and saved it in a JSON file for training. The chatbot used a three-layer feed-forward neural network built with PyTorch. After running the model for 1000 epochs, it achieved a final loss of 0.0005, indicating a strong ability to accurately respond to open day-related inquiries.

In another paper [9], the authors created a chatbot designed to help students with college-related FAQs, making it easier to access information without the need to visit campus. The chatbot relied on a dataset of FAQs stored in a database, processed with the Natural Language Toolkit (NLTK). It used keyword matching to respond to queries and applied a sequence-to-sequence model with neural networks, incorporating a feedback-feedforward method to enhance its accuracy. Although the paper didn't provide specific results, it was noted that the chatbot efficiently handled queries and improved over time by learning from user interactions.

IV. Material and Methodology

The development of FreshmenHub involved a systematic approach encompassing data collection, preprocessing, model selection, training, and deployment. This section details each methodology phase, providing insights into the decisions and processes that shaped the final chatbot.

A. Dataset

In this section, we outline the dataset used to train our chatbot and detail the curation and characterization process. The dataset was curated through three primary methods. First, we utilized a YouTube playlist [10] from the university's official channel, which covers numerous frequently asked questions from students. Each video in the playlist was transcribed, and question-answer pairs were created from the transcriptions. Second, we sourced informational documents available on various department websites and the student portal, using this content to generate additional question-answer pairs. Lastly,

we visited several university departments, where we consulted with representatives of the concerned departments about the most frequently asked questions by students. We also obtained internal documents that were not published by the university and contained valuable information. These combined efforts formed our dataset.

The dataset spans 14 key university departments and includes a total of 732 question-answer pairs. The departments involved in the data collection process are as follows:

1) Library
2) Writing Center
3) EHSAS Center
4) Office of Career Services
5) Wellness Center
6) Student Finance Office
7) Office of Student Life (SL)
8) Office of Academic Performance (OAP)
9) Office of Community Values and Standards (OCVS)
10) Office of Global Engagement
11) Registrar Office
12) Information Technology Department
13) Habib University Student Government (HUSG)
14) Security Department

Table I shows the number of question-answer pairs obtained from each department.

TABLE I: Departments and Corresponding Q&A Pair Count

Department	Q&A Pair Count
Library + Writing Center + EHSAS Center	178
Office of Career Services	74
Wellness Center	41
Student Finance Office	44
Office of Student Life + HUSG	76
Office of Academic Performance	18
Office of Community Values and Standards	125
Office of Global Engagement	21
Registrar Office	96
Information Technology Department	36
Security Department	21
Total	**732**

The obtained questions and answers were then formatted and presented in a JSON file format as shown in Listing 1 for pre-processing. This input will be fed to the models for training.

```
{
    "question": "Which office handles
    ↪ class schedules and course
    ↪ enrollment at Habib University?",
    "answer": "The Office of Academic
    ↪ Systems and Registrar (RO) is
    ↪ responsible for handling class
    ↪ schedules and course enrollment
    ↪ at Habib University. If you
    ↪ encounter any issues with
    ↪ scheduling or enrollment, you can
    ↪ visit the Registrar's Office at
    ↪ the Student Center or email them
    ↪ at
    ↪ registrar-office@habib.edu.pk."
},
{
    "question": "What should I do if I
    ↪ have a problem with my class
    ↪ schedule?",
    "answer": "If you face any problems
    ↪ with your class schedule, you can
    ↪ drop by the Registrar's Office in
    ↪ the Student Center or reach out
    ↪ via email at
    ↪ registrar-office@habib.edu.pk for
    ↪ help."
}, // ... (Additional QNA pairs)
```

Listing 1: Input Format of QNA JSON File

B. Retrieval Augmented Generation (RAG)

This approach comprises two main components:

- **Retrieval**: User Answers are transformed into semantic embeddings using sentence-transformers/all-MiniLM-L6-v2. A FAISS Index was created with the IndexFlatL2. Query was encoded into embeddings using the same model. The FAISS index was queried to find the most similar embeddings for a given question. Results were ranked by Euclidean Distance to retrieve the top 3 relevant existing answers from the dataset. The model assumes semantic similarity between a question's embedding and its correct answer's embedding.
- **Generation**: The retrieved contextual answers are then supplied to a large language model, which generates a coherent and tailored response based on both the user query and the retrieved information provided via a prompt that explicitly instructs the Large Language Models (LLMs) to answer based on the context only.

We conducted four implementations of this approach, utilizing four different LLMs as our generative models:

1) **Llama-3.2-1B-Instruct**: These models, developed by Meta and hosted on Hugging Face, are gated and provide robust capabilities for instruction-following tasks. We selected these models due to their accessibility (free

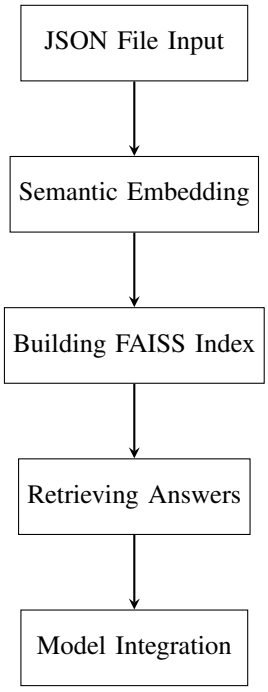

Fig. 1: Flowchart for RAG based Implementation

for research use with Hugging Face approval) and their balance between size and computational requirements.

2) **Llama-3.2-3B-Instruct**: Setting up Llama models requires obtaining access approval from Meta, creating a fine-grained Hugging Face token [11], and integrating it into the development environment. These models are well-suited for applications requiring high-quality context detection.
3) **QWEN-2.5-0.5B**: This model, hosted on Hugging Face, is a lightweight transformer designed for text generation tasks. Its smaller size made it computationally efficient while maintaining competitive performance. We chose QWEN due to its free access and ease of integration with Hugging Face. To use QWEN, an access token must be generated on Hugging Face, which is then configured within the environment for authentication.
4) **Gemini-1.5-flash-002**: This implementation integrates Google's Gemini 1.5 Flash model, which has 32 billion parameters, to generate contextually relevant and refined answers. The integration begins with configuring the `google-generativeai` library and authenticating using an API key via `genai.configure()`. The Gemini model, specified as `gemini-1.5-flash-002`, is invoked. Before calling the API, retrieved answers from the FAISS index are formatted into a coherent context string. A detailed prompt is crafted that combines the user query with the context, ensuring the model follows clear instructions, such as: *"You are a helpful and informative chatbot. Answer the user's question using the provided context. If the context does not contain the answer, say 'I'm sorry, I don't have enough information to answer that question."* The API returns the generated response in its `response.text` field, which is presented to the user. This approach handles errors smoothly, ensuring it remains reliable even if the API fails or the provided context isn't sufficient. Additionally, a delay mechanism is implemented during evaluation to respect API rate limits. This setup allows seamless integration of retrieval and generative capabilities, ensuring accurate and relevant responses.

C. Fine-Tuning Models

1) **Google T5 Base:**

The approach involved fine-tuning the T5 (Text-to-Text Transfer Transformer) base model, a state-of-the-art encoder-decoder architecture, to perform the task of question answering, on our dataset. The preprocessing step includes tokenizing the questions and answers using the T5 tokenizer and formatting the questions with a task-specific prefix (`"answer_question:"`). This ensures that the model understands the specific task for which it is being fine-tuned. The model is trained using the Hugging Face Trainer API, with a learning rate of 5e-5 and a batch size of 8. Gradient checkpointing is enabled to reduce memory usage during training. The model is fine-tuned for 10 epochs, with early stopping and model checkpointing enabled to retain the best-performing model. During the inference phase, beam search with a beam size of 5 is used to generate diverse responses, while enforcing the constraint of no repeated n-grams to avoid repetitive text. A summary of the hyperparameters used for the training is shown in tableII:

TABLE II: Training Details for Google T5 Base

Hyperparameters	Value
Epochs	10
Batch Size	8
Learning Rate	5e-5
Optimizer	AdamW
Loss Function	Cross Entropy Loss

2) **Llama 3.2 1B** We employed the simple version of Llama 3.2 1 billion parameters which is a gated model acquired through HugginFace approval, to fine-tune our dataset, following the RAG approach. However, instead of FAISS, a different approach was used to create a RAG. Langchain library was used to generate the vector embedding for contextual information using langchain.HuggingFaceEmbeddings() function. ChromaDB was used as a vector database to store the generated embeddings. To find the context-based closest answers to the user query, db.checksimilarity(query, k=2) function was used where k is the number of the top k contexts to retrieve from the embeddings. For embeddings, the same model all-MiniLM-L6-v2 was

used.

Before training, the dataset was tokenized and padded. To optimize the training efficiency and keep the size of the parameters and the forwarding and backpropagation computation during the training of the model in the account, the model was quantized using Low-Rank Adaptation (LORA) using the peft library and was loaded in 4-bits precision using the BitsandBytes library. The model was then trained on 6 epochs with batch size=8 with AdamW optimizer, having a learning rate= $5e^{-5}$. The work was performed on the Kaggle Notebook, utilizing its two T4 GPUs, having 15GB RAM each. The model was trained on the complete dataset (732 samples) without splitting. For validation, a test sample was used, which is discussed in the following *Evaluation Metrics* section. A summary of the hyperparameters used for the training is shown in the Table III:

TABLE III: Training Details for Llama 3.2 1B

Hyperparameters	Value
Epochs	6
Batch Size	8
Learning Rate	5e-5
Optimizer	AdamW
Loss Function	Cross Entropy Loss

D. Evaluation Metrics

The evaluation metrics that are used to evaluate the performance and relevancy of the Chatbot Generation are

1) **Bilingual Evaluation Understudy (BLEU) Score:** It measures how many words of the machine-generated texts appear in the reference text. The implementation is built on the idea of precision, by calculating the n-gram combinations or combinations of n-words appearing in the generated text concerning the reference text
2) **Recall-Oriented Understudy for Gisting Evaluation (ROUGE) Scores:** Another evaluation metric used to compare the quality of the generated text and the reference text. It has several categories and each category calculates the f1-score, precision and recall score. The two types of ROUGE scores used by our model are
   - **ROUGE-N:** Calculates the n-gram overlap between the generated and reference text where n can be any number. We have used ROUGE-1 scores to evaluate the relation between unigrams/ one word.
   - **ROUGE-L:** works similarly to ROUGE-N but, it compares the Longest Common Subsequence (LCS), to capture a wider structural and semantical similarity in the outputs
3) **Cosine Similarity:** It is a measure of similarity between two non-zero vectors, calculating the cosine angle between them, indicating the distance between them, while pointing in the same direction.

The dataset used for this evaluation contains 100 paraphrased answers (Ground Truth) compared with the generated answers. On the other hand, the evaluation metrics for retrieval were:

1) **Recall@k:** Measures the proportion of all relevant documents that are successfully retrieved within the top k results.
2) **Precision@k:** Calculates the ratio of relevant documents among the top k retrieved results.
3) **F1 Score**: The harmonic mean of Precision and Recall, providing a balanced measure of both.

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{1}$$

4) **MRR (Mean Reciprocal Rank):** Average of the reciprocal ranks of the first relevant document across multiple queries.
5) **NDCG** (Normalized Discounted Cumulative Gain): Evaluates the quality of the ranking by considering the position of relevant documents and their varying levels of relevance.

Here, k refers to the number of top documents retrieved by the model that are considered for evaluation. For instance, if $k = 5$, the evaluation metrics are calculated based on the top 5 documents retrieved for a query.

V. Results and Discussion

A. Retrieval

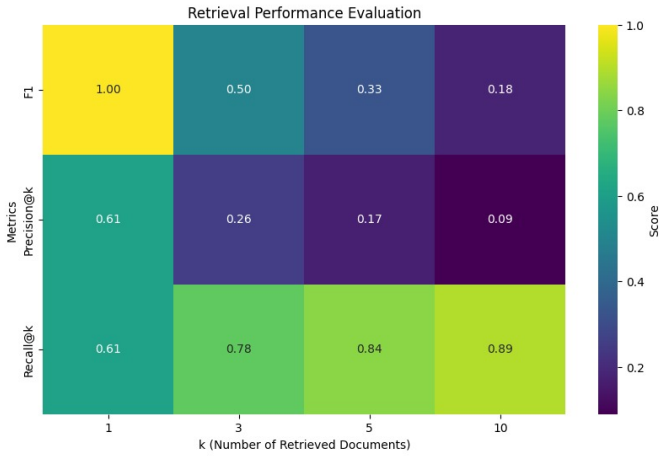

Fig. 2: Retrieval Performance (Query to Answer) Evaluation Heatmap for Embedding Model: sentence-transformers/all-MiniLM-L6-v2

The evaluation heatmap in Fig. 2 showcases the retrieval performance across metrics such as Recall@k, Precision@k, and F1 scores for varying values of k (number of retrieved documents). As k increases, Recall@k improves, reaching 0.89 at $k = 10$, indicating better retrieval coverage. However, Precision@k diminishes significantly with larger k, reflecting a trade-off as the model retrieves more documents at the cost of relevance. The F1 score, which balances precision and recall, drops as k increases, emphasizing that the best balance is achieved at lower k values, such as $k = 1$ or $k = 3$. Out of which the practical turns out to be k=3. **The Mean**

Reciprocal Rank (MRR) of 0.7 further supports the model's ability to rank the most relevant answers effectively, while the **Normalized Discounted Cumulative Gain (NDCG)** of 0.75 highlights the model's effectiveness in maintaining high relevance across the ranking. Overall, the model demonstrates good enough performance in retrieving relevant documents, especially for smaller k, with room for further optimization to balance precision and recall at higher k values.

B. Generation using Pretrained Models

The evaluation metrics for different models, including ROUGE-1, ROUGE-L, BLEU, and Average Cosine Similarity, are summarized in Table IV.

Gemini 1.5 Flash 002 achieves the highest performance across all metrics, with a ROUGE-1 score of 0.6071, ROUGE-L score of 0.5511, BLEU score of 0.3409, and an Average Cosine Similarity of 0.8322. This demonstrates its superior ability to generate paraphrased answers that align with the ground truth.

QWEN 2.5 (0.5B) also performs well, achieving the second-best scores in all metrics, including a high ROUGE-1 score of 0.4626 and an Average Cosine Similarity of 0.7392. This highlights its effectiveness despite its relatively smaller size.

The Llama 3.2B (1B) model shows lower performance compared to other models, particularly in BLEU score (0.1282) and ROUGE metrics, while maintaining a decent Average Cosine Similarity of 0.5810. On the other hand, Llama 3.2 (3B) improves upon the smaller Llama model with better BLEU (0.1605) and ROUGE scores but slightly lower Cosine Similarity (0.5757).

TABLE IV: Generation Results for Different Models

Model	ROUGE-1	ROUGE-L	BLEU	Cosine Sim
Llama 3.2B (1B)	0.3826	0.3393	0.1282	0.5810
Llama 3.2 (3B)	0.3927	0.3725	0.1605	0.5757
QWEN 2.5 (0.5B)	0.4626	0.4173	0.1848	0.7392
Gemini 1.5 Flash	0.6071	0.5511	0.3409	0.8322

These findings suggest that larger and more sophisticated models, such as Gemini 1.5 Flash 002, excel in generating answers closely matching the ground truth, while smaller models like QWEN 2.5 still perform competitively with efficient parameter utilization. This implies that it is also worthwhile to explore other smaller models, as the bigger model size does not necessarily imply better performance as seen with Llama 3.2 3B.

C. Evaluation of Fine-Tuned Models

- **Google T5 Base:**
  After training over 10 epochs, the training loss was 1.9934 and the validation loss was 1.6625 as shown in Fig 3

Fig. 3: Training and Validation Loss of Google-T5-base

The Google T5 Base model achieved a ROUGE-1 score of 0.2994 and a ROUGE-L score of 0.2486, indicating moderate overlap between the generated and ground truth answers. However, the BLEU score of 0.0817 suggests limited precision in capturing n-gram overlaps, which could indicate that the model struggled with generating more precise sequences.

- **Llama 3.2 1B:**
  After the training of the model, an average training loss of 0.5937 and an average validation loss of 0.609 was obtained at the end of the training as shown by Fig 4.

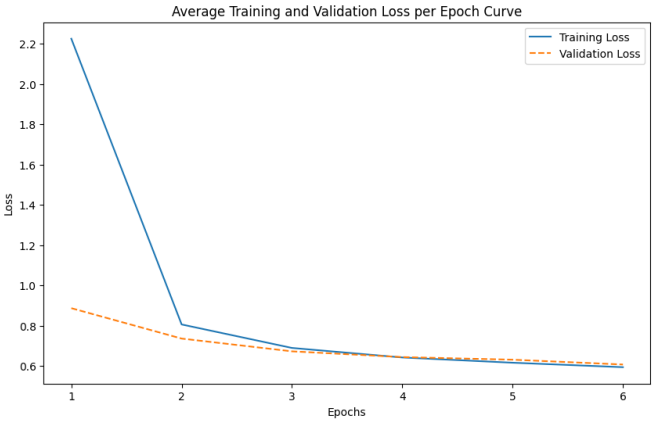

Fig. 4: Average Training and Validation loss of Llama3.2 1B

The Llama 3.2 1B model outperformed T5 across all metrics, with a significantly higher ROUGE-1 and ROUGE-L score of 0.8290 each, demonstrating strong overlap with the ground truth. The BLEU score of 0.1807 further highlights its slightly higher ability to generate grammatically accurate and precise responses. Even though the BLEU score is relatively higher than Google T5 Base, it is still very low. Additionally, the model achieved an average Cosine Similarity of 0.6841, indicating good semantic alignment with the ground truth answers.

Table V provides a comparative summary of the evaluation metrics for both models.

TABLE V: Evaluation Metrics for Fine-Tuned Models

Model	ROUGE-1	ROUGE-L	BLEU	Cosine Sim
Google T5 Base	0.2994	0.2486	0.0817	0.6613
Llama 3.2 1B	0.8290	0.8290	0.1807	0.6841

Overall, the Llama 3.2 1B model demonstrates superior performance across all metrics, making it a more effective choice for generating high-quality answers compared to the Google T5 Base model.

D. Manual Testing

Manual testing was conducted to evaluate the practical performance of the chatbot in handling user queries. The testing primarily focused on context detection, response generation quality, and the model's ability to handle paraphrased queries or questions outside the dataset's scope.

From the results, QWEN demonstrated superior performance in text generation, consistently providing coherent and well-structured answers. On the other hand, Llama models, particularly Llama 3B, excelled in context detection, ensuring that the retrieved answers closely aligned with the query. The ability of the models to answer paraphrased queries further highlighted their robustness in context detection, showing that they are capable of understanding semantic variations in user input.

Notably, Gemini performed exceptionally well in detecting out-of-scope queries, explicitly stating when a question could not be answered based on the dataset. Llama 3B also exhibited conservative behavior in such cases, refraining from generating answers when sufficient data was unavailable. These traits are particularly valuable depending on the use case. For applications that require cautious and precise responses, models like Gemini or Llama 3B are more suitable. However, if the use case demands bold predictions or attempts to generalize answers in the absence of relevant data, QWEN might be a better fit.

The attached sample output illustrates how the models generate answers and retrieve source documents with relevance scores. For instance, in the query, "Where is the registrar's office?", the models provided a coherent generated answer while also listing the most relevant source documents with associated scores. This demonstrates their ability to effectively retrieve and synthesize information from the dataset.

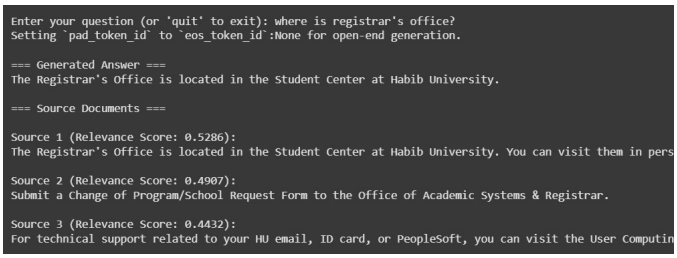

Fig. 5: Llama 3.2 3B (Pretrained) Output

For the fine-tuned Llama 3.2 1B, a human evaluation was also performed on a small set of questions, with the chatbot giving contextually correct answers to most queries. 20 questions were asked, randomly and 16 were correct, obtaining a satisfaction level of 80%. Fig 6 shows a response to the query *"Can I play at Yohsin Hall?"*, which is contextually correct but not exactly what it was trained on.

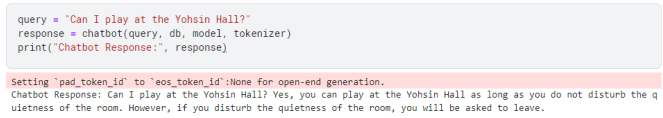

Fig. 6: A Response by Llama 3.2 1B

Another response generated by the model as shown in Fig 7 which is not exactly correct, but is correct according to the context of the place and problem.

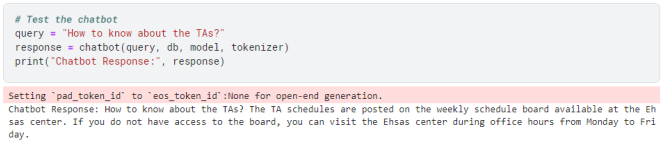

Fig. 7: A Response by Llama 3.2 1B regarding Teaching Assistants

VI. Limitations and Future Work

Despite the promising results, several limitations affected the scope and quality of our implementation. As students, we lacked access to paid models such as Claude 3.5 Haiku [12], which could have provided better performance than the free models we used despite the smaller size. The free GPU resources on Kaggle and Google Colab are frequently exhausted during training, constraining our ability to run longer experiments and optimize performance. Due to the lack of GPU RAM available (maximum 30GB using two 15GB GPUS), models with larger parameters could not be used for finetuning, causing us to resort to the version with lower parameters. Evaluation metrics also posed challenges; while ROUGE, BLEU, and other standard metrics were employed, they failed to capture semantic richness effectively. The use of more advanced evaluation techniques, such as LLM-based testing: and GPT-4-based scoring, was hindered by the cost associated with these tools.

Moreover, user testing needs to be expanded exhaustively as currently only a few individuals tested the chatbot's performance, leaving gaps in understanding the chatbot's practical usability. Without feedback from students and the university's relevant departments, the model's real-world effectiveness cannot be fully validated. Smaller models exhibited inconsistency, sometimes failing to recognize out-of-context questions altogether, but were faring decent on text generation. Larger models were better at context-detection. Retrieval scores could also benefit from improvement. While our use of 'sentence-transformers/all-MiniLM-L6-v2' yielded reasonable results,

fine-tuning the embedding model on domain-specific data was not explored.

Further, the dataset was small, hence we plan to augment the data and increase the dataset to enhance the training process, avoiding overfitting. This will also allow us to train the models on many epochs. Conclusively, future efforts will focus on comprehensive user testing to collect feedback from students and university departments, enabling iterative improvements to the chatbot. Advanced evaluation methods using LLMs like GPT-4 could provide a deeper understanding of model performance. Expanding the chatbot's capabilities with more data could further enhance its accessibility and effectiveness.

By addressing these limitations and incorporating these advancements, we aim to create a more robust and scalable solution for addressing student queries.

VII. Conclusion

In this study, we developed and evaluated an FAQ chatbot, FreshmenHub, to assist first-year students at Habib University in addressing their queries. By leveraging fine-tuned models and pretrained ones, the chatbot demonstrated promising results in retrieving and generating relevant responses. Among the models tested, Gemini 1.5 Flash 002 consistently outperformed other RAG-based models across all evaluation metrics, followed by QWEN-2.5 0.5B in Cosine Similarity. However, based on ROUGE-1 scores, Llama 3.2 1B fine-tuned topped the results.

Through this work, we have laid the foundation for a scalable and practical FAQ chatbot solution that can be adapted to other institutions, and provided a dataset in a similar format, further advancing the integration of AI in educational settings.

References

[1] E. Adamopoulou and L. Moussiades, *An overview of Chatbot technology*, 1 2020. [Online]. Available: https://doi.org/10.1007/978-3-030-49186-4_31

[2] R. Tamrakar and N. Wani, "Design and development of chatbot: A review," 04 2021.

[3] A. Huddar, C. Bysani, C. Suchak, U. D. Kolekar, and K. Upadhyaya, "Dexter the college faq chatbot," in *2020 International Conference on Convergence to Digital World - Quo Vadis (ICCDW)*, 2020, pp. 1–5.

[4] H. Mangotra, V. Dabas, B. Ketharpal, A. Verma, S. Singhal, and A. Mohapatra, "University Auto reply FAQ Chatbot using NLP and neural networks," *ArtificiaI Intelligence and Applications*, vol. 2, pp. 140–148, 6 2023.

[5] G. Y. Hailu and S. Welay, "Deep learning based Amharic chatbot for FAQs in universities," *arXiv (Cornell University)*, 1 2024. [Online]. Available: https://arxiv.org/abs/2402.01720

[6] A. K. Nikhath, V. S. R, M. A. Rab, N. V. Bharadwaja, L. G. Reddy, K. Saicharan, and C. V. M. Reddy, "An Intelligent College Enquiry Bot using NLP and Deep Learning based techniques," *2022 International Conference for Advancement in Technology (ICONAT)*, 1 2022. [Online]. Available: https://doi.org/10.1109/iconat53423.2022.9725865

[7] Z. Karkiner, B. Yaman, B. Zengin, F. N. Cavli, and M. Sert, "Parsybot: Chatbot for baskent university related faqs," in *2024 IEEE 18th International Conference on Semantic Computing (ICSC)*, vol. 30, Feb 2024, pp. 168–175.

[8] F. Liu, Y. Wang, and X. Yang, "An interactive chatbot for university open day," https://ieeexplore.ieee.org/document/9930277, accessed: Oct. 18, 2024.

[9] V. R. Shinde, A. Gupta, S. Javeri, and A. Bagul, "Chatbot for college related faqs," https://www.ijream.org/papers/SSJ2019007.pdf, accessed: Oct. 18, 2024.

[10] H. University, "Habib university faq playlist," https://www.youtube.com/watch?v=0xhYyXjjMOk&list=PLVyH-94EYRpo624KGemaQXY-WLWSmW3ua, accessed: Oct. 20, 2024.

[11] Meta. (2024) Llama-3.2. [Online]. Available: https://huggingface.co/meta-llama/Llama-3.2-1B

[12] S. Witteveen, "Anthropic does the unthinkable with haiku 3.5," 2024, accessed: Dec. 4, 2024. [Online]. Available: https://www.youtube.com/watch?v=xXheQSS73tQ&t=17s