Beyond ChatGPT: A Hybrid Chatbot Model for Reliable Educational Administration

Kanaan Al-Jaf *, Cemil Öz, Tarik A. Rashid *

Posted Date: 4 October 2024

doi: 10.20944/preprints202410.0276.v1

Keywords: Chatbots; ChatGPT; Rule-based; Retrieval-based; Generative-based; Educational Administration; Hybrid model

 doi:10.20944/preprints202410.0276.v1

Article

Beyond ChatGPT: A Hybrid Chatbot Model for Reliable Educational Administration

Kanaan Al-Jaf [1,2,*], Cemil Öz [1] and Tarik A. Rashid [3,*]

[1] Department of Computer Engineering, Sakarya University, 54050 Sakarya, Turkey; coz@sakarya.edu.tr
[2] Department of Information Technology, University of Human Development, Sulaymaniah 46001, Iraq;
[3] Computer Science and Engineering, University of Kurdistan Hewler, Erbil 44001, Iraq
* Correspondence: kanaan.jaf@ogr.sakarya.edu.tr; tarik.ahmed@ukh.edu.krd

Abstract: The ever-growing student population and limited administrative resources strain traditional communication channels within educational institutions. Chatbot technology offers a promising solution, but current limitations can impede effective student-administration interaction. This paper proposes a hybrid chatbot model to enhance reliability in educational administration by addressing the limitations of generative models like ChatGPT. We investigate three individual chatbot approaches (rule-based, retrieval-based, and generative-based) to handle educational inquiries. An evaluation assessed their strengths, weaknesses, and user experience. Based on these findings, a hybrid model with an intelligent classifier for query routing was developed. This model leverages each approach's strengths and addresses uncertainty in the generative model to achieve increased accuracy and reliability. Our results demonstrate an accuracy improvement compared to individual models. This work contributes to developing adaptable chatbot systems capable of addressing a wider range of user inquiries, which is particularly beneficial in resource-constrained educational settings. Additionally, the proposed model demonstrates competitive performance when benchmarked against existing state-of-the-art administrative educational chatbots. Furthermore, by automating routine inquiries and offering 24/7 access to information and support, this model has the potential to reduce administrative workload and improve the overall student experience significantly. Moreover, the modular design of our hybrid model offers a foundation for future research on personalizing chatbot interactions and integrating domain-specific knowledge for enhanced communication within educational environments.

Keywords: Chatbots; ChatGPT; Rule-based; Retrieval-based; Generative-based; Educational Administration; Hybrid model

1. Introduction

Educational institutions rely heavily on clear and efficient communication between students, faculty, and administrators to ensure a smooth learning experience. Timely access to information and resources is essential for academic success. Traditionally, communication has depended on in-person interactions, emails, and phone calls, which can be time-consuming and resource-intensive [1]. The advent of chatbot systems offers a promising solution to streamline administrative tasks and enhance user experiences by providing prompt and accurate responses to a wide range of inquiries [2, 3]. These intelligent conversational agents can reduce the administrative burden and improve accessibility for students, faculty, and staff by providing readily available information 24/7 [4].

Despite their potential, current chatbot technologies face significant limitations. Existing approaches, such as rule-based, retrieval-based, and generative-based models, struggle to meet the diverse administrative needs of educational institutions effectively [5]. For instance, generative models like ChatGPT can handle open-ended questions but may sometimes produce inaccurate or uncertain responses [6]. This issue is particularly critical in educational settings, where the accuracy and reliability of information are paramount.

To address these limitations, we propose a novel hybrid chatbot model specifically designed for reliable educational administration. This model integrates the strengths of three distinct

 doi:10.20944/preprints202410.0276.v1

approaches—rule-based, retrieval-based, and generative-based chatbots—using a smart classifier system for intelligent query routing. By combining these methods, our hybrid model aims to provide superior performance and a dependable source of information within educational environments. Our research evaluates the effectiveness of individual chatbot models in handling educational inquiries, assessing their strengths and weaknesses, user satisfaction, and overall usefulness. Based on these evaluations, we have developed a hybrid model with an intelligent classifier to route queries to the most suitable approach. Our objective is to demonstrate the superiority of this hybrid model in terms of accuracy and reliability compared to individual models. This work contributes to the development of adaptable chatbot systems capable of addressing a broader range of administrative inquiries, which is particularly beneficial in resource-constrained educational environments. It also aims to inform future research on hybrid chatbots for educational settings.

The structure of the remaining sections of this paper is as follows: Section 2 provides a review of relevant related works; Section 3 discusses the methodology employed; Section 4 outlines the results and discussion of the research; and Section 5 concludes the research.

2. Literature Review

The amount of research exploring the use of chatbots in the educational field is increasing [1], and the educational landscape is witnessing a surge in interest in chatbots as valuable tools for enhancing student learning experiences and streamlining administrative tasks. This growing adoption is driven by the potential of chatbots to provide 24/7 accessibility to information and resources [7], fostering increased student engagement [8], and reducing the workload on administrative staff [9]. Studies have shown that chatbots can effectively address various needs within educational settings, which can be broadly categorized into information provision and administrative support. Regarding information provision, chatbots can address FAQs and provide users with general information regarding various aspects of educational institutions, including activities [16-19], admissions [3, 9, 20, 21], academics [8, 22-24], libraries [25, 26], and other topics like tuition fees and campus life [12, 27] . The second type of chatbot is designed to assist with administrative tasks, aiming to improve efficiency and user experience through automation. This automation benefits students, educators, and administrative staff by saving time [28, 29], reducing workload and labor costs [3, 23, 30-32], offering 24/7 availability [4, 7, 15], increasing user experience [33, 34], engagement [4, 8, 35], and even automating the admission process [29].

Chatbot Architectures: Based on their underlying architecture or approach, chatbot models can be classified as rule-based, retrieval-based, and generative chatbots. Rule-based chatbots follow a predefined set of rules and are designed to respond to specific keywords or phrases. Retrieval-based chatbots use a combination of predefined responses and machine learning algorithms to select the best response to a user's input. They work by matching the user's input to a database of pre-existing responses. Generative chatbots, on the other hand, use deep learning techniques like neural networks to generate responses from scratch. When it comes to chatbot development for educational institutions, the literature reflects a variety of approaches. For rule-based chatbots, some studies utilized development tools like Artificial Intelligence Markup Language (AIML) [8,30,37]. Generative-based chatbots were created using techniques such as Recurrent Neural Network (RNN) [38, 39] and Unsupervised Learning [33]. Retrieval-based chatbots found application in educational settings through platforms like DialogFlow, a natural language understanding platform [17, 31, 32, 41- 43], as well as the RASA framework [44-46], and Chatterbot platform [11, 47].

Comparative Analysis: While individual studies have explored different chatbot approaches, there remains a gap in the literature concerning the development of a comprehensive and adaptable chatbot model capable of effectively handling various administrative tasks. While individual chatbot approaches have shown promise, there remains a limited understanding of their relative strengths and weaknesses in educational settings. Comparative analyses can address this gap by directly evaluating different models. One study [42] compared rule-based (AIML) and generative (Sequence-to-Sequence) approaches, revealing a trade-off between user satisfaction and task completion for rule-based models, and superior information retrieval for generative models. Building upon the idea of

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 October 2024 doi:10.20944/preprints202410.0276.v1

comparative analysis, another study [1] developed five chatbot models using various techniques and found limitations in pattern-matching approaches due to overfitting.

Generative models: The emergence of large language models like ChatGPT, have garnered significant interest for their ability to handle open-ended questions and generate creative text formats [43]. In the context of educational administration, this translates to the potential for providing students with explanations, summaries, and even creative writing prompts. In their work, the authors of [44] identified several key advantages of using ChatGPT in education. These include the ability to tailor learning to individual students, develop innovative assessment methods, and enhance students' critical thinking abilities. They suggest that by carefully considering both the potential benefits and drawbacks of ChatGPT, it can be used to improve the overall learning experience for university students.

Research Gap: The studies reviewed in this section showcase the potential of various chatbot approaches in educational settings. However, some limitations exist. Rule-based models can be inflexible for open-ended questions, while retrieval-based models may struggle with complex inquiries. Generative models, like ChatGPT, due to their reliance on statistical patterns in vast datasets can lead to inaccuracies, particularly in factual inquiries [45]. Additionally, generative models may struggle to understand the nuances of human language and context, potentially leading to misleading or irrelevant responses. Our research proposes a hybrid chatbot model that leverages the strengths of these approaches to overcome these limitations. By combining rule-based, retrieval-based, and generative-based approaches, we aim to develop a more robust and adaptable model for handling diverse administrative tasks in educational institutions.

3. Materials and Methods

This section details the methods employed in the comparative analysis of rule-based, retrieval-based, and generative-based chatbots designed to address administrative inquiries within educational institutions, it also proposes the development and evaluation of a hybrid model that leverages the strengths of these individual approaches. The methodology, further illustrated in Figure 1, encompasses five key steps: data collection, model development, model evaluation and comparison, classifier implementation, and model integration.

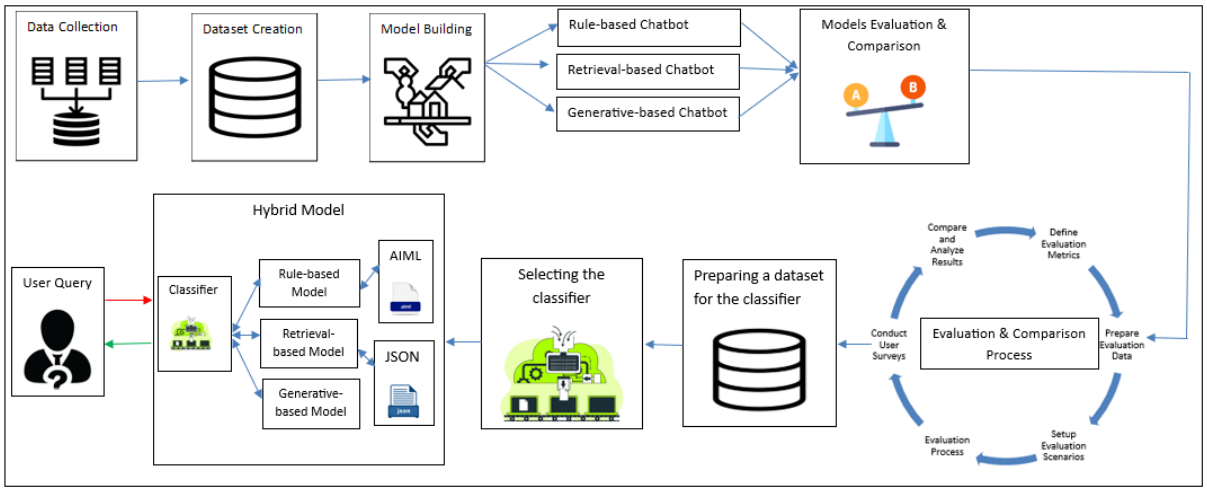

Figure 1. Simplified Workflow Diagram.

3.1. Data Collection and Dataset Creation

A dataset is crucial for training any model. Due to the lack of datasets specifically tailored to Sakarya University, we constructed a new dataset from scratch. This dataset comprises a collection of questions and corresponding answers stored in a text file format. To ensure clarity and organization, a consistent notation system has been adopted. Question sentences are marked with a "Q:" prefix, while answer sentences are denoted with an "A:". The data has been sourced from various

 doi:10.20944/preprints202410.0276.v1

channels: Sakarya University's FAQ pages, and social media groups where students posed questions about the university and questions were generated based on information available on the university's official websites. This comprehensive approach ensured the dataset captured a wide range of inquiries typically encountered by students. Additionally, incorporating valuable feedback from students further enriched the dataset's relevance. data pre-processing steps have been performed (removing punctuation and duplicate questions, segmenting long sentences into smaller units) to clean and organize the collected information. The resulting dataset encompasses 2253 lines of conversation, categorized into various domains such as Admissions and Registrations, Academic Affairs, Financial Matters, and Student Services. This structure ensures the dataset's broad coverage and addresses the needs of prospective, current, and foreign students at Sakarya University.

3.2. Models Creation

The core of this research hinges on the development and application of diverse models. This subsection, 3.2 Models Creation, delves into the specific steps taken to construct these models. We will detail the implementation of all three models: a rule-based approach, a retrieval-based approach, and a generative-based approach.

3.2.1. Rule-Based Chatbot

To develop a rule-based chatbot, Artificial Intelligence Markup Language (AIML) was selected. AIML is a widely used programming language for creating chatbots and virtual assistants. It is an XML-based language that uses pattern matching and natural language processing techniques to simulate conversation with human users [46]. It has been opted for due to its simplicity, flexibility, and user-friendly syntax. AIML's built-in features enable efficient handling of diverse user queries [46]. The Pandorabots platform has been utilized. Pandorabots is a platform that utilizes AIML to build and deploy chatbots which provides a user-friendly interface for defining AIML rules, managing the knowledge base, and integrating the chatbot [47]. 637 categories were established from the dataset described in section 3.1. The development process can be summarized as follows:

1. AIML File Generation: The Pandorabots playground was used to generate AIML files from the dataset, creating the initial chatbot prototype.
2. Prototype Integration: The prototype was then integrated into the Pandorabots framework, enabling user interaction.
3. User Testing: A group of 113 students (undergraduate and postgraduate) interacted with the chatbot.
4. Evaluation and Improvement: User interactions were logged and analyzed (elaborate on the specific aspects evaluated, e.g., response accuracy, user satisfaction). This analysis led to a review and enhancement of the chatbot's functionalities.
5. Redeployment: The improved version of the chatbot was redeployed for further user interaction.

Figure 2 shows the pseudocode for the built rule-based chatbot model, and Figure 3 represents its graphical interface.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 October 2024 doi:10.20944/preprints202410.0276.v1


```
FUNCTION GENERATE_AIML_FILES(dataset)
  USE_PANDORABOTS_PLAYGROUND(dataset)
  RETURN generated_aiml_files
END FUNCTION
FUNCTION INTEGRATE_PROTOTYPE(aiml_files)
  INTEGRATE_AIML(aiml_files)
  ENABLE_USER_INTERACTION()
END FUNCTION
FUNCTION CONDUCT_USER_TESTING(num_users)
  RECRUIT_USERS(num_users)
  FACILITATE_USER_INTERACTION()
  LOG_USER_INTERACTIONS()
END FUNCTION
FUNCTION EVALUATE_AND_IMPROVE(user_interactions)
  ANALYZE_INTERACTIONS(user_interactions)
  REVIEW_AND_ENHANCE_FUNCTIONALITIES()
END FUNCTION
FUNCTION REDEPLOY_CHATBOT(improved_version)
  DEPLOY(improved_version)
END FUNCTION
LET aiml_files = GENERATE_AIML_FILES(dataset)
INTEGRATE_PROTOTYPE(aiml_files)
CONDUCT_USER_TESTING(113)  // Assuming 113 students
LET user_interactions = GET_LOGGED_INTERACTIONS()
EVALUATE_AND_IMPROVE(user_interactions)
REDEPLOY_CHATBOT(improved_version)
```

Figure 2. Rule-based model algorithm.

Figure 3. Rule-based chatbot GUI.

3.2.2. Retrieval-Based Chatbot

The retrieval-based chatbot leverages a JSON file to store information relevant to user inquiries. This file maintains data regarding Sakarya University, categorized into the same main categories as the dataset (section 3.1). The JSON structure includes:

1. intents: An array of intent objects, each representing a specific category such as 'weather', 'courses', and 'tuition fees'.
2. Questions: An array of user queries or inputs associated with the intent.
3. Responses: An array of possible responses or answers the chatbot can provide related to the given intent.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 October 2024 doi:10.20944/preprints202410.0276.v1

Data pre-processing is crucial for the retrieval-based model. This involves:

1. Tokenization: Breaking down text into individual words or units.
2. Stopword removal: Eliminating common words that hold little meaning such as 'the', and 'a'.
3. Stemming: Reducing words to their base form such as 'running' -> 'run'. Porter Stemmer has been employed for this purpose.
4. Processed dataset creation: Combining the preprocessed questions (patterns) and corresponding preprocessed answers (randomly chosen from the response options).

TF-IDF (Term Frequency-Inverse Document Frequency) is a core technique used in this model. It is a numerical statistic used to evaluate the importance of a word to a document in a collection or corpus [48]. A TF-IDF vectorizer is initialized to calculate the similarity matrix using cosine similarity. This step aims to determine the closest match between the user's input and the preprocessed questions within the dataset.

Entity extraction involves identifying specific elements within the user's input that are relevant to the query. This is achieved through a function named extract entities. It performs the following actions:

1. Tokenization: Breaking down the user input into individual words.
2. Pattern matching: Checking the user input against predefined patterns associated with the intent.
3. Entity extraction: Utilizing regular expressions to extract specific entities from the matched patterns.

Finally, the generated response function takes the user input and generates a response. This function:

1. Preprocesses the user input.
2. Calculate the similarity between the preprocessed input and the preprocessed dataset questions using cosine similarity.
3. Identifies the most similar question (intent) based on the calculated similarity scores.
4. Extract entities relevant to the identified intent using the extract_entities function.
5. Returns the corresponding response associated with the most similar question and extracted entities.

Figure 4 illustrates the pseudocode for the constructed retrieval-based chatbot model, while Figure 5 displays its graphical interface.

```
DEFINE_STRUCT Intent {
  questions: ARRAY OF STRING
  responses: ARRAY OF STRING
}

LET chatbot_data = LOAD_JSON_DATA("data.json")
LET intents = chatbot_data["intents"]
FUNCTION PREPROCESS_TEXT(text)
  LET tokens = TOKENIZE(text)
  REMOVE_STOPWORDS(tokens)
  STEM_WORDS(tokens, PORTER_STEMMER)
  RETURN tokens
END FUNCTION
FUNCTION CREATE_PROCESSED_DATASET()
  LET processed_dataset = []
  FOR EACH intent IN intents
    LET preprocessed_questions = []
    FOR EACH question IN intent.questions
      LET preprocessed_question = PREPROCESS_TEXT(question)
      APPEND(preprocessed_questions, preprocessed_question)
    END FOR
    LET random_response_idx = RANDOM_INT(0, SIZE(intent.responses) - 1)
    LET random_response = intent.responses[random_response_idx]
    APPEND(processed_dataset, {questions: preprocessed_questions,
        | response: random_response})
  END FOR
  RETURN processed_dataset
```

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 October 2024 doi:10.20944/preprints202410.0276.v1


```
END FUNCTION
LET tfidf_vectorizer = CREATE_TFIDF_VECTORIZER()
LET processed_dataset = CREATE_PROCESSED_DATASET()
LET question_vectors = tfidf_vectorizer.fit_transform(
[q for q in (item["questions"] for item in processed_dataset)])
FUNCTION EXTRACT_ENTITIES(user_input)
  LET tokens = TOKENIZE(user_input)
  LET entities = EXTRACT_ENTITIES_REGEX(tokens)
  RETURN entities
END FUNCTION
FUNCTION GENERATE_RESPONSE(user_input)
  LET preprocessed_user_input = PREPROCESS_TEXT(user_input)
  LET similarities = COSINE_SIMILARITY(tfidf_vectorizer.transform(
        [preprocessed_user_input]), question_vectors)
  LET most_similar_idx = FIND_MAX_INDEX(similarities)
  LET extracted_entities = EXTRACT_ENTITIES(user_input)
  LET response = processed_dataset[most_similar_idx]["response"]
  RETURN response
END FUNCTION

// Main Loop
LOOP
  LET user_input = GET_USER_INPUT()
  LET response = GENERATE_RESPONSE(user_input)
  PRINT(response)
END LOOP
```

Figure 4. Retrieval-based model algorithm.

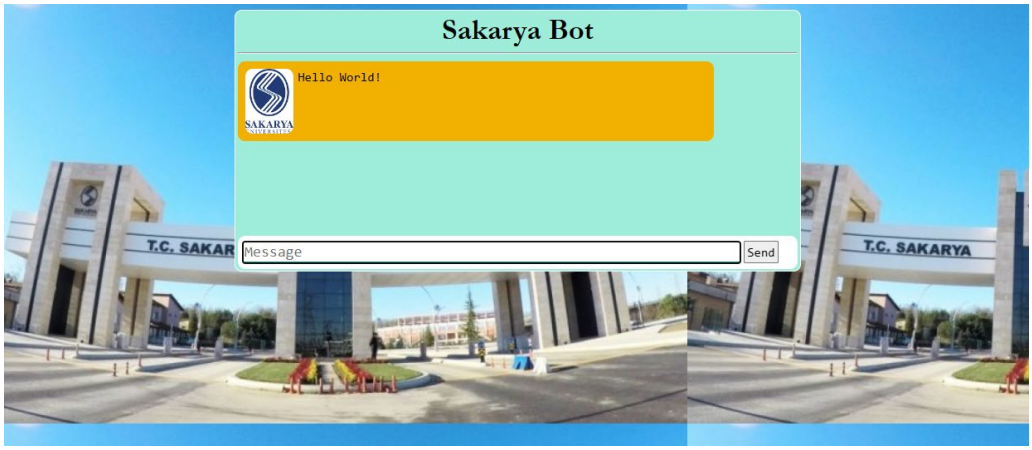

Figure 5. Retrieval-based chatbot GUI.

3.2.3. Generative-Based Chatbot

This section clarifies the fine-tuning process employed to convert a pre-trained language model into a functional chatbot using the Hugging Face transformers library. The objective is to fine-tune the model on a custom Q&A dataset (section 3.1) and leverage it to generate responses to user queries. The generative-based chatbot implementation can be summarized as follows:

1. Library Imports: Necessary libraries are imported, including torch for PyTorch functionalities and AutoTokenizer and AutoModelForSeq2SeqLM from the transformers library for handling the pre-trained model and tokenizer.
2. GPU Availability Check: The code checks for a CUDA-enabled GPU to accelerate computations. If available, the GPU is used; otherwise, the CPU takes over.
3. **Loading the Dataset:** The dataset created in section 3.1 is loaded line by line, separating each line into individual questions and answers.
4. Separating Questions and Answers: The entire dataset is traversed to split questions and answers into distinct lists.
5. Loading Pre-trained Model and Tokenizer: The pre-trained model ("tuner007/pegasus_paraphrase") is specified, and the corresponding tokenizer and model are

 doi:10.20944/preprints202410.0276.v1

loaded using AutoTokenizer and AutoModelForSeq2SeqLM. The loaded model is then transferred to the chosen device (GPU or CPU).

6. Tokenization and Encoding: Both questions and answers are tokenized and encoded using the loaded tokenizer. Padding and truncation are applied to ensure sequences adhere to specific length requirements. These encoded sequences are then converted into PyTorch tensors.
7. Fine-tuning the Model: Fine-tuning involves the following sub-steps:

a. **Optimizer Selection:** An Adam optimizer is chosen and initialized with a learning rate of 5e-5. The fine-tuning process iterates through 40 epochs.
b. **Encoding Input and Output Sequences:** For each epoch, the model encodes the input and output sequences for each Q&A pair.
c. **Generating Output and Calculating Loss:** The model generates an output sequence based on the input sequence and calculates the loss against the provided target output sequence.
d. **Backpropagation and Updating Model Parameters:** The calculated loss is accumulated and used to update the model's parameters through backpropagation.
e. **Saving Model State:** After each epoch, the model's state (including both model weights and tokenizer) is saved whenever a decrease in the average loss is observed.

8. Model Evaluation: The fine-tuned model's performance is assessed using the training set.
9. User Interaction with Fine-tuned Model: The final step involves user interaction with the chatbot:

- **Optimizer Selection:** An Adam optimizer is chosen and initialized with a learning rate of 5e-5. The fine-tuning process iterates through 40 epochs.
- **Encoding Input and Output Sequences:** For each epoch, the model encodes the input and output sequences for each Q&A pair.
- **Generating Output and Calculating Loss:** The model generates an output sequence based on the input sequence and calculates the loss against the provided target output sequence.
- **Backpropagation and Updating Model Parameters:** The calculated loss is accumulated and used to update the model's parameters through backpropagation.
- **Saving Model State:** After each epoch, the model's state (including both model weights and tokenizer) is saved whenever a decrease in the average loss is observed.

 doi:10.20944/preprints202410.0276.v1


```
IMPORT torch
FROM transformers IMPORT AutoTokenizer, AutoModelForSeq2SeqLM
IF HAS_CUDA_GPU() THEN
  SET DEVICE = "cuda"
ELSE
  SET DEVICE = "cpu"
END IF
LET dataset = []
OPEN_FILE("dataset.txt")
LOOP (UNTIL_END_OF_FILE)
  LET line = READ_LINE()
  LET [question, answer] = SPLIT_LINE(line)
  APPEND(dataset, {question: question, answer: answer})
END LOOP
CLOSE_FILE()
LET questions = []
LET answers = []
FOR EACH item IN dataset
  APPEND(questions, item["question"])
  APPEND(answers, item["answer"])
END FOR
LET model_name = "tuner007/pegasus_paraphrase"
LET tokenizer = AutoTokenizer.from_pretrained(model_name)
LET model = AutoModelForSeq2SeqLM.from_pretrained(model_name)
MODEL.to(DEVICE)
LET encoded_questions = []
LET encoded_answers = []
FOR EACH question, answer IN ZIP(questions, answers)
  LET tokenized_question = tokenizer(question, padding="max_length", truncation=True)
  LET tokenized_answer = tokenizer(answer, padding="max_length", truncation=True)
  CONVERT_TO_TENSOR(tokenized_question, DEVICE)  # Convert to PyTorch tensor on chosen device
  CONVERT_TO_TENSOR(tokenized_answer, DEVICE)
  APPEND(encoded_questions, tokenized_question)
  APPEND(encoded_answers, tokenized_answer)
END LOOP
LET optimizer = ADAM(model.parameters(), lr=5e-5)  # Adam optimizer with learning rate
LET num_epochs = 40
FOR epoch = 1 TO num_epochs
  LET total_loss = 0
  FOR EACH question, answer IN ZIP(encoded_questions, encoded_answers)
    Encode input and output sequences
    Generate output, calculate loss, backpropagate, and update parameters
    ADD_LOSS_TO_TOTAL(loss)
  END FOR
  LET avg_loss = total_loss / SIZE(dataset)
  IF (IS_LOSS_DECREASED(avg_loss)) THEN
    SAVE_MODEL(model, tokenizer)
  END IF
  PRINT("Epoch", epoch, "Average Loss:", avg_loss)
END FOR
  Model Evaluation
LOOP
  LET user_question = GET_USER_INPUT()
  LET encoded_question = tokenizer(user_question, padding="max_length", truncation=True)
  CONVERT_TO_TENSOR(encoded_question, DEVICE)
  LET generated_answer = GENERATE_RESPONSE(model, encoded_question)
  LET decoded_answer = tokenizer.decode(generated_answer, skip_special_tokens=True)
  PRINT("Bot:", decoded_answer)
END LOOP
```

Figure 6. Generative-based model algorithm.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 October 2024 doi:10.20944/preprints202410.0276.v1

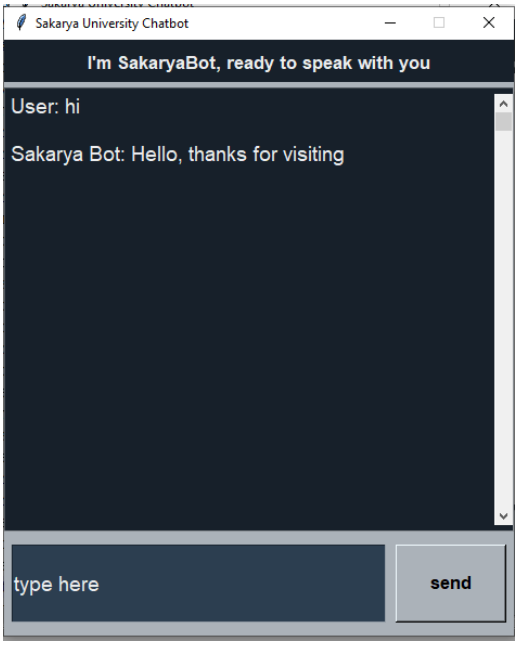

Figure 7. Generative-based chatbot GUI.

3.3. Models' Evaluation and Comparison

This section presents a comparative analysis of three prominent chatbot models utilized in educational institutions: rule-based, retrieval-based, and generative-based models. Through a rigorous evaluation process to:

1. Unveil the strengths, weaknesses, and unique performance characteristics of each approach.
2. Lay the groundwork for identifying advancements and exploring the feasibility of a hybrid model that leverages the most advantageous features of all three approaches.

Figure 8 visually depicts the lifecycle of the comparison process, providing a roadmap for the subsequent sections.

Figure 8. Models Comparison Life Cycle.

3.3.1. Define Evaluation Metrics

The initial step involves determining relevant metrics to assess the chatbot models' performance, the core metrics include:

1. Accuracy: Percentage of inquiries answered correctly and relevantly.
2. Precision: Proportion of responses that are topical and pertinent to the user's query.

 doi:10.20944/preprints202410.0276.v1

3. Recall: Proportion of relevant questions accurately answered.
4. F1 Score: Harmonic means of precision and recall, offering a balanced view.

In addition to the core metrics, additional metrics have been considered such as:

a. Response Time: Average time taken to generate a response.
b. User Satisfaction and Engagement: Subjective metrics assessed through surveys to gauge user perception of helpfulness, ease of use, and overall effectiveness.
c. Flexibility: Ability to adapt to new information or situations such as ease of adding new knowledge, handling unexpected queries.

3.3.2. Prepare Evaluation Data

To ensure an accurate evaluation, a dedicated dataset was compiled using real-world data, the factors have been considered are:

1. Data Source: Extracted from the last updated FAQ page on Sakarya University's main website.
2. Selection Criteria: 300 representative queries and corresponding answers, representing approximately 20% of the total training dataset size, were chosen to capture the diverse range of user inquiries typically encountered in the university setting.
3. Rationale: Incorporating these authentic and up-to-date queries aims to assess the models' performance in accurately understanding and responding to inquiries specific to Sakarya University.

3.3.3. Setup Evaluation Scenarios

Three distinct scenarios or use cases are defined to represent the digital assistant's functionalities:

1. Scenario 1 (Prospective Students): Admissions, academics, and general inquiries.
2. Scenario 2 (Enrolled Students): Registrations, tuition fees, courses, and campus facilities.
3. Scenario 3 (Foreign Students): Admission and registration, tuition fees, weather, language proficiency requirements, and accommodation.

These scenarios encompass various domains and topics relevant to Sakarya University.

3.3.4. Evaluation Process

In this step, the specific evaluation approaches for each model have been outlined:

1. Rule-based Chatbot:
a. Focus: Ability to match user queries to predefined rules and generate appropriate responses.
b. Metrics: Measure the accuracy of matching and rule coverage.
2. Retrieval-based Chatbot:
a. Focus: Selecting the most relevant response from a predefined set.
b. Metrics: Use accuracy, precision, recall, and F1 score to evaluate response selection accuracy.
3. Generative-based Chatbot:
a. Focus: Assess the quality of generated responses.
b. Factors: Consider response coherence, grammar, and relevance to the user query.

3.3.5. Conduct User Surveys

User surveys were conducted to gather feedback on:

1. Satisfaction Level: How satisfied users were with the interaction.
2. Perceived Usefulness: How helpful users found the chatbot.
3. Overall User Experience: Ease of use and overall impression.

A total of 267 undergraduate students participated by responding to the 10-question survey (Table 1). All user interactions and feedback were recorded for further analysis.

Table 1. User Survey Questions with Rating Scales.

No.	Questions	Rating scales

 doi:10.20944/preprints202410.0276.v1

Q1	How would you rate the accuracy of the responses?	(1-5)
Q2	How would you rate your level of acceptance of the chatbot's suggestions or recommendations?	(1-5)
Q3	How would you rate the usefulness of the chatbot in addressing your queries or providing information?	(1-5)
Q4	How would you evaluate the context of the conversations?	(1-5)
Q5	Would you be interested in using the chatbot again in the future?	(1-5)
Q6	How satisfied are you with the chatbot's ability to understand your queries?	(1-5)
Q7	How satisfied are you with the chatbot's response time?	(1-5)
Q8	How would you rate the chatbot's ability to provide relevant and helpful information?	(1-5)
Q9	How would you rate the chatbot's overall performance in assisting you with your needs?	(1-5)
Q10	How likely are you to recommend the chatbot to others?	(1-5)

3.3.6. Compare and Analyze Results

This final step involves:

1. Comparing performance: Analyze the performance of each model across the different evaluation metrics and user feedback.
2. Identifying strengths and weaknesses: Examine each model's effectiveness in terms of accuracy, response quality, user satisfaction, and other relevant factors.

The purpose of evaluating the models was to:

1. Review and improve: The evaluation aims to identify areas for improvement in both the dataset and the design of the models.
2. Hybrid model development: By comparing the models' strengths, the analysis seeks to explore the possibility of integrating them into a single, enhanced hybrid model.

3.4. Classifier Implementation

The evaluation process revealed the distinct strengths of each chatbot model in handling administrative tasks across various university-related categories. To efficiently route user queries to the most suitable model, a classifier is implemented. This section details the chosen approach and its implementation steps. Figure 9 presents the pseudocode for the Multinomial Naive Bayes Classifier, while Figure 10 displays a sample of the classifier dataset.

3.4.1. Classifier Selection:

A Multinomial Naive Bayes classifier is employed for category classification. A Multinomial Naive Bayes Classifier is a probabilistic machine learning algorithm used for text classification [49]. It assumes that the occurrence of a word in a document is independent of the occurrence of other words. due to its:

1. Simplicity: Easy to understand and implement.
2. Efficiency: Performs well with large datasets and requires minimal training data compared to some complex models.
3. Effectiveness: Often yields competitive results for text classification tasks.

3.4.2. Data Preparation:

A dedicated dataset (Figure 10) was created for classifier training, containing:

1. Question Column: Represents the input user query as text.
2. Category Column: Represents the corresponding category or class such as admissions, registration, and fees.

3.4.3. Implementation Steps:

 doi:10.20944/preprints202410.0276.v1

1. Data Loading: The Question and Category columns are loaded into separate lists, typically named corpus and labels, respectively, for easier processing.
2. Text Classification Pipeline:

a. The Pipeline class from scikit-learn is utilized to create a text classification pipeline.
b. The pipeline consists of two crucial steps:
i. Text Vectorization: The CountVectorizer transforms textual data into numerical features, enabling the classifier to process the text.
ii. Classification: The MultinomialNB algorithm implements the Naive Bayes classification model to categorize the text data based on the learned patterns.

3. Data Splitting:

a. The train_test_split function from scikit-learn is used to divide the dataset into training and testing sets.
b. A common practice is to allocate 40% of the data for testing (test_size=0.4) to assess the classifier's performance on unseen data.
c. Setting the random_state parameter to a fixed value (e.g., 42) ensures reproducibility, meaning the same data split will occur when the code is run multiple times.

4. Classifier Training:

a. The training set is used to train the classifier using the fit method of the pipeline.
b. During this process:
i. The text data from the training set is vectorized using CountVectorizer.
ii. The transformed numerical features are then used to train the Naive Bayes model, enabling it to learn the relationships between text patterns and their corresponding categories.

5. Model Saving:

a. The trained classifier is saved to a file (classifier.pkl) using the joblib.dump function.
b. This step allows for reusing the trained model for future predictions without the need for retraining, saving computational resources.

6. Evaluation:

a. The performance of the trained classifier is assessed using the testing set (X_test).
b. The predict method is employed to predict category labels for the unseen test data.
c. The accuracy_score function is used to evaluate the model's accuracy by comparing the predicted labels with the actual labels (y_test).
d. Accuracy represents the percentage of correctly classified instances.

```
LET corpus = []  # List to store question texts
LET labels = []  # List to store category labels
LOAD_DATA("data.csv", corpus, labels)

LET pipeline = Pipeline([
  ("vectorizer", CountVectorizer()),
  ("classifier", MultinomialNB())
])
LET X_train, X_test, y_train, y_test = train_test_split
        (corpus, labels, test_size=0.4, random_state=42)
pipeline.fit(X_train, y_train)
SAVE_MODEL(pipeline, "classifier.pkl")
LET predicted_labels = pipeline.predict(X_test)
LET accuracy = accuracy_score(y_test, predicted_labels)
PRINT("Model Accuracy:", accuracy)
```

Figure 9. Multinomial Naive Bayes Classifier algorithm.

 doi:10.20944/preprints202410.0276.v1

A	B
Question	Category
What are the available exchange programs in Sakarya University?	Rule-based
is the library working hours change during the exam period ?	Retrieval-based
Which students have to pay the tuition fee?	Rule-based
Does Sakarya experience any extreme weather conditions, such as tornadoes or hurricanes?	Generative-based
What are the different types of dormitories available at Sakarya University?	Retrieval-based

Figure 10. Sample of Classifier Dataset.

3.5. Models Integration (Hybrid Model)

Building upon the individual strengths of the rule-based, retrieval-based, and generative-based models, this section describes their integration into a cohesive framework, forming a hybrid chatbot system.

The integration process revolves around two key components:

1. Classifier: The trained Multinomial Naive Bayes classifier, as explained in Section 3.4, plays a crucial role in efficiently routing user queries.
2. Main Program Loop: This core loop manages the user interaction flow within the system.

Figure 11 Represents the pseudocode for the proposed hybrid model, and Figure 12 Describes it's process flow which can be summarized as:

1. User Query Input: The user submits their query through the system's interface.
2. Query Processing Function (process_user_query(query))

a. Category Prediction: The function leverages the trained classifier to predict the category (e.g., admissions, registration, fees) that the query most likely belongs to.

b. Model Selection: Based on the predicted category, the function directs the query to the most suitable model:

i. Rule-based model: Well-suited for well-defined queries with a clear answer in the knowledge base.

ii. Retrieval-based model: Effective for retrieving relevant responses from a predefined set when the answer might involve variations in phrasing.

iii. Generative-based model: Ideal for handling open-ended questions, complex inquiries, or situations where a creative response is desired.

3. Model Response Generation: The selected model processes the query and generates a response tailored to the user's needs.
4. Response Delivery: The generated response is returned from the process_user_query(query) function to the main program loop.
5. User Interface Interaction: The main program loop displays the response to the user through the system's interface, completing the interaction cycle.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 October 2024 doi:10.20944/preprints202410.0276.v1


```
LET classifier = LOAD_MODEL("Multinomial_Naive_Bayes_Classifier")
LET rule_based_model = LOAD_MODEL("Rule_Based_Model")
LET retrieval_based_model = LOAD_MODEL("Retrieval_Based_Model")
LET generative_based_model = LOAD_MODEL("Generative_Based_Model")

// Main Program Loop
LOOP
  LET user_query = GET_USER_INPUT()
  FUNCTION process_user_query(query)
    LET predicted_category = PREDICT_CATEGORY(classifier, query)
    LET selected_model = CHOOSE_MODEL(predicted_category)
    LET response = GENERATE_RESPONSE(selected_model, query)
    RETURN response
  END FUNCTION
  LET response = process_user_query(user_query)
  PRINT(response)
END LOOP
FUNCTION CHOOSE_MODEL(category)
  IF category IN CATEGORIES_FOR_RULE_BASED_MODEL THEN
    RETURN rule_based_model
  ELSIF category IN CATEGORIES_FOR_RETRIEVAL_BASED_MODEL THEN
    RETURN retrieval_based_model
  ELSE
    RETURN generative_based_model
  END IF
END FUNCTION

FUNCTION GENERATE_RESPONSE(model, query)
  response = call_model(selected_model, query)
  return response
END FUNCTION
```

Figure 11. Hybrid model algorithm.

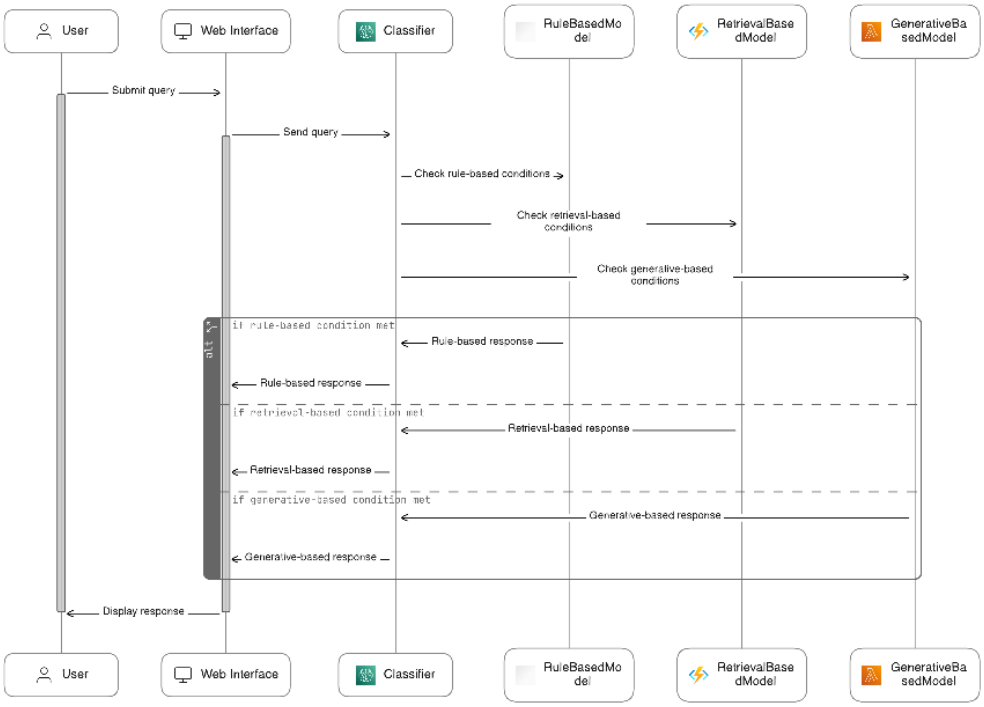

Figure 12. Hybrid Model Flow Diagram.

4. Results And Discussion

For randomly selected 300 different queries, the performance of all three chatbot approaches was evaluated. The rule-based chatbot achieved a matching accuracy of 73.33% for user queries to

 doi:10.20944/preprints202410.0276.v1

predefined rules, effectively generating appropriate responses. It covered 63.33% of the rules defined within AIML categories (Table 2).

Table 2. Rule-based evaluation.

Metrics	Results %
Accuracy of Matching	210/300 = 73.33 %
Rule Coverage	191/300 = 63.33 %

On the other hand, the retrieval-based chatbot demonstrated a higher performance with a score of 83.33% in selecting the most relevant response from a predefined set of responses. Additionally, it provided a wide range of response diversity, enhancing the overall user experience (Table 3).

Table 3. Retrieval-based evaluation.

Metrics	Results %
Response Selection Accuracy	252/300 = 83.33 %
Diversity of Responses	qualitative assessment of the chatbot's ability to provide varied and appropriate responses is required.

To evaluate the generative-based chatbot, the chatbot's responses for the same set of queries were assessed by an English specialist annotator. The manual evaluation using the Likert scale (strongly agree(5), agree(4) , neutral(3), disagree(2), strongly disagree(1)) revealed that the model achieved scores of 72.12%, 75.42%, 87.23% for response quality, coherence, grammar, and language fluency, respectively (Table 4).

Table 4. Generative-based evaluation.

Metrics	Metric	Results %
Response Quality	BLEU (0-1)	0.7812 = 78.12 %
Coherence and Relevance	Manual (1-5)	75.42 %
Grammar and Language Fluency	Manual (1-5)	87.23%

The evaluation results highlighted the need for improvements in all three chatbot models and the dataset as well. For the rule-based model, an expansion of the rules defined within AIML categories was necessary to cover a wider range of user queries. As a result, an additional 112 categories were added to enhance the model's coverage. In the case of the retrieval-based chatbot, the random() function was activated to introduce more response diversity, especially for queries that could have multiple valid answers. This adjustment aimed to enhance the user experience and provide varied responses. Furthermore, to improve the performance of the generative-based model, the dataset was modified to include more examples that consider comprehension and context, thereby ensuring the generation of more contextually appropriate and coherent responses. Table 5 summarizes the results of the questionnaire administered to users who interacted with all three models. The scores were calculated for each question and for all three models using Equation 1. User satisfaction rates were determined based on questions 2, 6, and 7. The retrieval-based model received the highest score in user satisfaction. Model usefulness was assessed through questions 3, 8, and 9, and the rule-based model achieved the highest score in this category. User engagement was evaluated using questions 5 and 10, and the rule-based model also scored highest in this aspect. Question 4 focused on measuring human-like conversation, and the generative-based model received the highest score in this area.

$$\text{Score} = \frac{(summation\ of\ all\ 67\ students)}{10\ *\ 67} * 2 \qquad (1)$$

Table 5. Conduct User Surveys Results.

 doi:10.20944/preprints202410.0276.v1

User Experience	Rule-based (out of 10)	Retrieval-Based (out of 10)	Generative-based (out of 10)
User Satisfaction	7.29	7.53	6.78
Usefulness/Effectiveness	7.66	7.11	7.45
User Engagement	8.33	7.56	8.10
Human-like Conversation	5.79	6.22	7.24

The evaluation results and user feedback revealed distinct advantages for each chatbot model. The rule-based model demonstrated superior performance in terms of model usefulness and user engagement. This was primarily attributed to its ability to provide accurate answers in specific categories such as tuition fees and course schedules. Additionally, the rule-based model offered an attractive user interface through the utilization of rich media tags provided by the AIML playground (Pandorabots). The retrieval-based model outperformed the others in terms of response selection, leading to higher user satisfaction. The generative-based model excelled in delivering a more human-like conversation experience, owing to its implementation of a generative pre-trained model with a transformer architecture. To assess the strengths of each model in handling various administrative tasks within Sakarya University and to propose the hybrid model, a detailed analysis was conducted on all the conversations. The task completion rates for each model are summarized in Table 6. The results indicate that the rule-based model performed well in the categories of Admissions and Registrations (87.33%) and Financial Matters (89.56%). The retrieval-based model achieved the best results in the categories of Academic Affairs (84.67%), Student Services (88.94%), and Campus Life (91.44%). On the other hand, the generative-based model excelled in the categories of General Information (100%) and Cross-domain Queries (73.12%).

Table 6. Task completion rate.

Tasks	Query's Category	Rule-based	Retrieval-based	Generative-based
Task1	Admissions and Registrations	87.33	82.13%	81.21%
Task2	Academic Affairs	78.18%	84.67%	77.57%
Task3	Financial Matters	89.56%	82.33%	79.77%
Task4	Student Services	55.51%	71.18%	78.67%
Task5	Campus Life	84.35%	91.44%	83.09%
Task6	Greeting and General Information	82.57%	76.13%	97.89%
Task7	Cross-domain Queries	44.21%	51.16%	73.12

This categorization resulted from the analysis of the study, where the specific chatbot implementation, quality of data, and training provided to the chatbot models contribute to the different strengths exhibited by each chatbot model in specific task categories. As part of the evaluation process, the individual responses of the chatbots have been analyzed. Table 7 presents the various response categories.

 doi:10.20944/preprints202410.0276.v1

Table 7. Response types.

Response type	Description
True Positive	refers to the number of instances where the chatbot correctly predicts a positive response.
False Positive	This refers to the number of instances where the chatbot incorrectly predicts a positive response.
True Negative	refers to the number of instances where the chatbot correctly predicts a negative response.
False Negative	refers to the number of instances where the chatbot incorrectly predicts a negative response.

Using the information provided in Table 8, Accuracy, Precision, Recall, and F-Measure were computed utilizing the following formulas [50]:

$$Accuracy = \frac{\text{True Positives} + \text{True Negative}}{\text{True Positives} + \text{True Negative} + False\ Positives + False\ Negative} \quad (2)$$

$$Precision = \frac{\text{True Positives}}{True\ Positives + False\ Positives} \quad (3)$$

$$Recall = \frac{\text{True Positives}}{True\ Positives + False\ Negative} \quad (4)$$

$$F - score = 2 * \frac{\text{Precision} * \text{Recall}}{Precision + Recall} \quad (5)$$

Table 8. Accuracy, Precision, Recall, F1-measure.

Response types	Rule-based	Retrieval-Based	Generative-based
True Positive	302	356	323
False Positive	132	98	111
False Negative	73	47	57
True Negative	11	17	25
Accuracy	76.11%	87.64%	83.78 %
Precision	0.70	0.78	0.74
Recall	0.81	0.88	0.85
F1-Measure	0.62	0.67	0.65

Accuracy measures the overall correctness of the model's predictions. the retrieval-based model still has the highest accuracy among the three models. The accuracy alone is not sufficient to evaluate a model's performance comprehensively. Precision and recall provide additional insights into the model's ability to correctly identify positive instances and minimize false positives or false negatives, respectively. Precision is a measure of how many of the responses predicted as positive are actually correct, the retrieval-based model still has the highest precision, meaning that when it predicts a response as positive, it is correct 78% of the time. Recall measures the ability of the model to correctly

 doi:10.20944/preprints202410.0276.v1

identify positive responses, the retrieval-based model still has the highest recall, indicating that it can correctly identify positive responses 88% of the time. F1-Measure is the harmonic mean of precision and recall and provides a single metric to evaluate the overall performance, The retrieval-based model still has the highest F1-Measure, indicating a balanced performance in terms of precision and recall.

The purpose of evaluating the models was to review and make the necessary modifications to both the dataset and the design of the models. Additionally, the purpose of comparing the models was to identify their respective strengths to integrate them into a single hybrid model. The models' evaluation has been completed and required modifications have been made to both the dataset and the design of the models. Based on the models' comparison, the polynomial Naïve Bayes classifier has been implemented to forward specific types of queries to different models. For instance, queries related to 'Admissions and Registrations' and 'Financial Matters' are directed to the rule-based model. 'Academic Affairs', 'Student Services', and 'Campus Life' queries are handled by the retrieval-based chatbot, while 'General Information' and 'Cross-domain' queries are addressed by the generative-based model. Table 9 demonstrates the advancements and accuracy achieved by the hybrid model. The evaluation of the model utilized the updated dataset, with an equal number of queries employed for all three models. The results showcased a noteworthy improvement in comparison to the rule-based, retrieval-based, and generative-based models. This progress serves as evidence that the hybrid model effectively utilizes the strengths of all three models. This achievement stands as the primary contribution of this work.

According to [2] assessing the practicality of Chatbot systems requires employing a suitable method to evaluate the efficiency of the software engineering product, along with a broader and statistically meaningful sample of users.

Table 9. Accuracy, Precision, Recall, F1-measure for Hybrid model.

Tasks	Hybrid model
True Positive	477
False Positive	7
False Negative	13
True Negative	21
Accuracy	96.13%
Precision	0.9855
Recall	0.9734
F1-Measure	0.9749

A detailed comparison in Table 10 reveals the proposed hybrid model surpasses all existing state-of-the-art administrative educational chatbots. This superiority is achieved through combining the strengths of each model (rule-based, retrieval-based, and generative-based models). For example, the hybrid model leverages the rule-based system's efficiency for handling common tasks, the retrieval-based system's ability to access and deliver relevant information, and the generative-based system's capacity for natural language understanding and response generation.

Table 10. Summary of comparison between related works and the proposed hybrid model.

 doi:10.20944/preprints202410.0276.v1

Reference	Objective	Dataset used	Dataset Size	Model type	Evaluation Approach	Accuracy obtained
[34]	administrative and learning support	Ho Chi Minh City University of Science, Vietnam (FIT-HCMUS)	1560 user messages	Retrieval-based	precision, recall measures, F-score	82.33%
[51]	intelligent customer service system for university admissions	encyclopedia question-and-answer, Tamkang University Dataset	1.5 million pre-filtered questions and answers, 210 questions and answers about Tamkang University	generative-based	BLEU, questionnaire-based evaluation	80.02%
[52]	Chatbot for academic calendar, schedule, and grade		1400 sentences	Retrieval-based	Manual evaluation	85.5%
[53]	support the admission process	College-related dataset	8,500 questions and 6,500 answers.	Retrieval-based	Precision, Recall, and F-score	89.0%
[54]	addressing questions about new student admittance.	University-related dataset		Retrieval-based	precision, Recall, and F1-score	90.62 %
[31]	Addressing FAQ about University	Dataset about University of Computer Studies, Yangon, Myanmar	5000 question-answer pairs	Generative-based	BLEU	82.0%
[32]	Chatbot to answer queries related to Telkom University admission	Telkom University dataset	2,903 Conversations	Generative-based	BLEU	89.36%
[3]	Addressing queries about admission for new students, tuition fees, IELTS test, scholarships, or deadlines.	National Economics University dataset	1500 examples	Retrieval-based	F1-score, accuracy, and precision	95.1%
[55]	Addressing FAQ, student support	National University of Science & Technology (NUST), Islamabad, Pakistan dataset	1000 examples	Retrieval-based	Precision, fl-score, and accuracy	76.8%

 doi:10.20944/preprints202410.0276.v1

Proposed Hybrid Model	Proposing hybrid educational administrative chatbot	Sakary University Dataset	2253 Question-Answer pairs	Hybrid approach	Accuracy, Precision, Recall, F-1 measure	97.57%

5. CONCLUSIONs

This study investigated the potential of hybrid chatbot models for enhancing educational administration. By comparing rule-based, retrieval-based, and generative-based approaches, we identified distinct strengths and weaknesses in handling student inquiries. Our findings revealed that rule-based models excel in domains such as Admissions and Registrations, and Financial Matters, while retrieval-based models demonstrate superior performance in Academic Affairs, Student Services, and Campus Life. Generative models proved effective in handling Greeting and General Information inquiries as well as cross-domain queries. Our proposed hybrid model, which combines these approaches and employs a query routing classifier, demonstrated an accuracy improvement over individual models. This enhancement underscores the model's potential to optimize efficiency and user experience within educational settings. Benchmarking against state-of-the-art solutions further solidified the model's competitive performance. Future research should focus on expanding the dataset for more robust model training, exploring advanced classification algorithms such as deep learning-based techniques that could further enhance query routing and overall system performance, and conducting comprehensive user studies to assess the model's impact on student satisfaction and administrative efficiency.

Authors' Contribution: KA: Conceptualization, Methodology, Writing the Initial Draft, and Revision; CO: Conceptualization and Supervision. TA: Methodology, Restructuring the Initial Draft, and Revision. All authors have reviewed and approved the final manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data used and analyzed during the current study are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. G. Attigeri and A. Agrawal, "Advanced NLP Models for Technical University Information Chatbots : Development and Comparative Analysis," *IEEE Access*, vol. 12, no. February, pp. 29633–29647, 2024, doi: 10.1109/ACCESS.2024.3368382.
2. C. W. Okonkwo and A. Ade-Ibijola, "Chatbots applications in education: A systematic review," *Comput. Educ. Artif. Intell.*, vol. 2, 2021, doi: 10.1016/j.caeai.2021.100033.
3. T. T. Nguyen, A. D. Le, H. T. Hoang, and T. Nguyen, "NEU-chatbot: Chatbot for admission of National Economics University," *Comput. Educ. Artif. Intell.*, vol. 2, p. 100036, 2021, doi: 10.1016/j.caeai.2021.100036.
4. M. Dibitonto, K. Leszczynska, F. Tazzi, and C. M. Medaglia, "Chatbot in a Campus Environment : Design of LiSA , a Virtual Assistant to Help Students in Their University Life," in *Human-Computer Interaction. Interaction Technologies*, Cham., M. Kurosu, Ed. Springer International Publishing, 2018, pp. 103–116. doi: 10.1007/978-3-319-91250-9.
5. T. Lopez and M. Qamber, "The Benefits and Drawbacks of Implementing Chatbots in Higher Education," no. March, 2022.
6. P. P. Ray, "ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope," *Internet Things Cyber-Physical Syst.*, vol. 3, no. March, pp. 121–154, 2023, doi: 10.1016/j.iotcps.2023.04.003.
7. M. Rukhiran and P. Netinant, "Automated information retrieval and services of graduate school using chatbot system," in *International Journal of Electrical and Computer Engineering (IJECE)*, 2022, vol. 12, no. 5, pp. 5330–5338. doi: 10.11591/ijece.v12i5.pp5330-5338.
8. D. Al-ghadhban and N. Al-twairesh, "Nabiha : An Arabic Dialect Chatbot," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 3, pp. 452–459, 2020, doi: 10.14569/IJACSA.2020.0110357.

9. W. El Hefny, Y. Mansy, M. Abdallah, and S. Abdennadher, "Jooka : A Bilingual Chatbot for University Admission," in *Trends and Applications in Information Systems and Technologies*, 2021, pp. 671–681. doi: 10.1007/978-3-030-72660-7.
10. R. Lucien and S. Park, "Design and Development of an Advising Chatbot as a Student Support Intervention in a University System," *TechTrends*, vol. 68, no. 1, pp. 79–90, 2024, doi: 10.1007/s11528-023-00898-y.
11. K. N. Lam, L. H. Nguy, V. L. Le, and J. Kalita, "A Transformer-Based Educational Virtual Assistant Using Diacriticized Latin Script," *IEEE Access*, vol. 11, no. August, pp. 90094–90104, 2023, doi: 10.1109/ACCESS.2023.3307635.
12. S. C. Man, O. Matei, T. Faragau, L. Andreica, and D. Daraba, "The Innovative Use of Intelligent Chatbot for Sustainable Health Education Admission Process: Learnt Lessons and Good Practices," *Appl. Sci.*, vol. 13, no. 4, 2023, doi: 10.3390/app13042415.
13. H. Agus Santoso *et al.*, "Dinus Intelligent Assistance (DINA) Chatbot for University Admission Services," *Proc. - 2018 Int. Semin. Appl. Technol. Inf. Commun. Creat. Technol. Hum. Life, iSemantic 2018*, pp. 417–423, 2018, doi: 10.1109/ISEMANTIC.2018.8549797.
14. S. Sakulwichitsintu, "ParichartBOT: a chatbot for automatic answering for postgraduate students of an open university," *Int. J. Inf. Technol.*, vol. 15, no. 3, pp. 1387–1397, 2023, doi: 10.1007/s41870-023-01176-z.
15. P. Udupa, "Application of artificial intelligence for university information system," *Eng. Appl. Artif. Intell.*, vol. 114, p. 105038, 2022, doi: 10.1016/j.engappai.2022.105038.
16. N. Thalaya and K. Puritat, "BCNPYLIB CHAT BOT: The artificial intelligence Chatbot for library services in college of nursing," in *2022 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT & NCON)*, 2022, pp. 247–251. doi: 10.1109/ECTIDAMTNCON53731.2022.9720367.
17. S. Rodriguez and C. Mune, "Uncoding library chatbots: deploying a new virtual reference tool at the San Jose State University library," *Ref. Serv. Rev.*, vol. 50, no. 3–4, pp. 392–405, 2022, doi: 10.1108/RSR-05-2022-0020.
18. R. Isaev, R. Gumerov, G. Esenalieva, R. R. Mekuria, and E. Doszhanov, "HIVA: Holographic Intellectual Voice Assistant," *Proc. - 2023 17th Int. Conf. Electron. Comput. Comput. ICECCO 2023*, pp. 1–6, 2023, doi: 10.1109/ICECCO58239.2023.10146600.
19. J. Barzola-monteses, S. Member, and J. Guillén-mirabá, "CSM : a Chatbot Solution to Manage Student Questions About payments and," vol. 4, pp. 1–12, 2024, doi: 10.1109/ACCESS.2024.3404008.
20. R. Parkar, Y. Payare, K. Mithari, J. Nambiar, and J. Gupta, "AI and Web-Based Interactive College Enquiry Chatbot," *Proc. 13th Int. Conf. Electron. Comput. Artif. Intell.*, pp. 1–5, 2021, doi: 10.1109/ECAI52376.2021.9515065.
21. A. K. Nikhath *et al.*, "An Intelligent College Enquiry Bot using NLP and Deep Learning based techniques," *2022 Int. Conf. Adv. Technol. (CONAT)*, pp. 1–6, 2022, doi: 10.1109/ICONAT53423.2022.9725865.
22. R. B. K. Md. Abdullah Al Muid, Md. Masum Reza, M. M. Reza, R. Bin Kalim, N. Ahmed, M. T. Habib, and M. S. Rahman, "Edubot: An unsupervised domain-specific chatbot for educational institutions," *Lect. Notes Networks Syst.*, vol. 144, pp. 166–174, 2021, doi: 10.1007/978-3-030-53970-2_16.
23. N. A. Al-Madi, K. A. Maria, M. A. Al-Madi, M. A. Alia, and E. A. Maria, "An Intelligent Arabic Chatbot System Proposed Framework," *2021 Int. Conf. Inf. Technol.*, pp. 592–597, 2021, doi: 10.1109/ICIT52682.2021.9491699.
24. A. Agung and S. Gunawan, "Ava: knowledge-based chatbot as virtual assistant in university," *ICIC Express Lett. Part B Appl.*, vol. 13, no. 4, pp. 437–444, 2022, doi: 10.24507/icicelb.13.04.437.
25. H. Dinh and T. K. Tran, "EduChat: An AI-Based Chatbot for University-Related Information Using a Hybrid Approach," *Appl. Sci.*, vol. 13, no. 22, p. 12446, 2023, doi: 10.3390/app132212446.
26. T. Lalwani, S. Bhalotia, A. Pal, S. Bisen, and V. Rathod, "Implementation of a Chat Bot System using AI and NLP," *Int. J. Innov. Res. Comput. Sci. Technol.*, vol. 6, no. 3, pp. 26–30, 2018, doi: 10.21276/ijircst.2018.6.3.2.
27. J. P. and J. S. L. Galko, "Improving the user experience of electronic university enrollment," in *2018 16th International Conference on Emerging eLearning Technologies and Applications (ICETA)*, 2018, pp. 179–184. doi: 10.1109/ICETA.2018.8572054.
28. W. Villegas-ch, J. Garc, K. Mullo-ca, S. Santiago, and M. Roman-cañizares, "Implementation of a Virtual Assistant for the Academic Management of a University with the Use of Artificial Intelligence," *Futur. Internet*, vol. 13, no. 4, p. 97, 2021, doi: doi.org/10.3390/fi13040097.
29. P. Giannos and O. Delardas, "Performance of ChatGPT on UK Standardized Admission Tests: Insights from the BMAT, TMUA, LNAT, and TSA Examinations," *JMIR Med. Educ.*, vol. 9, pp. 1–7, 2023, doi: 10.2196/47737.
30. S. S. Sushmitha, S. Ramya, and G. M. Gandhi, "College Chatbot using Natural Language Processing for Student Queries," vol. 10, no. 5, pp. 25549–25553, 2020.
31. N. N. Khin and K. M. Soe, "Question Answering based University Chatbot using Sequence to Sequence Model," *2020 23rd Conf. Orient. COCOSDA Int. Comm. Co-ord. Stand. Speech Databases Assess. Tech.*, pp. 55–59, 2020, doi: 10.1109/O-COCOSDA50338.2020.9295021.

32. Y. W. Chandra and S. Suyanto, "Indonesian chatbot of university admission using a question answering system based on sequence-to-sequence model," *Procedia Comput. Sci.*, vol. 157, pp. 367–374, 2019, doi: 10.1016/j.procs.2019.08.179.
33. S. Sinha, S. Basak, Y. Dey, and A. Mondal, "An Educational Chatbot for Answering Queries," in *Emerging Technology in Modelling and Graphics*, 2020, pp. 55–60. doi: 10.1007/978-981-13-7403-6.
34. H. T. Hien, P.-N. Cuong, L. N. H. Nam, H. L. T. K. Nhung, and L. D. Thang, "Intelligent Assistants in Higher-Education Environments: The FIT-EBot, a Chatbot for Administrative and Learning Support," in *Proceedings of the 9th International Symposium on Information and Communication Technology (SoICT '18)*, 2018, pp. 69–76. doi: 10.1145/3287921.3287937.
35. K. Lee, J. Jo, and J. Kim, "Can Chatbots Help Reduce the Workload of Administrative Of fi cers ? - Implementing and Deploying FAQ Chatbot Service in a University," in *21st International Conference on Human-Computer Interaction, HCI International 2019*, 2019, pp. 348–354. doi: 10.1007/978-3-030-23522-2.
36. M. Mekni, Z. Baani, and D. Sulieman, "A Smart Virtual Assistant for Students," in *Proceedings of the 3rd International Conference on Applications of Intelligent Systems (APPIS 2020)*, 2020, pp. 1–6. doi: 10.1145/3378184.3378199.
37. S. Alqaidi, W. Alharbi, and O. Almatrafi, "A support system for formal college students: A case study of a community-based app augmented with a chatbot," *2021 19th Int. Conf. Inf. Technol. Based High. Educ. Train. (ITHET)*, vol. 978, no. 1, 2021, doi: 10.1109/ITHET50392.2021.9759796.
38. T. Dinesh, M. R. Anala, T. T. Newton, and G. R. Smitha, "AI Bot for Academic Schedules using Rasa," *Proc. 2021 IEEE Int. Conf. Innov. Comput. Intell. Commun. Smart Electr. Syst. ICSES 2021*, 2021, doi: 10.1109/ICSES52305.2021.9633799.
39. Y. Windiatmoko, A. F. Hidayatullah, and ..., "Developing FB chatbot based on deep learning using RASA framework for university enquiries," in *IOP Conf. Ser.: Mater. Sci. Eng*, 2020, p. 1077 012060. doi: 10.1088/1757-899X/1077/1/012060.
40. N. Alzaabi, M. Mohamed, H. Almansoori, M. Alhosani, and N. Ababneh, "ADPOLY Student Information Chatbot," *2022 5th Int. Conf. Data Storage Data Eng.*, pp. 108–112, 2022, doi: 10.1145/3528114.3528132.
41. A. Roy, D. Singh, and S. Sahana, "Educational assistance bot," *J. Phys. Conf. Ser.*, vol. 1797, no. 1, p. 012062, 2021, doi: 10.1088/1742-6596/1797/1/012062.
42. N. Teckchandani, A. Santokhee, and G. Bekaroo, "AIML and Sequence-to-Sequence Models to Build Artificial Intelligence Chatbots: Insights from a Comparative Analysis," *Lect. Notes Electr. Eng.*, vol. 561, no. May, pp. 323–333, 2019, doi: 10.1007/978-3-030-18240-3_30.
43. F. Khennouche, Y. Elmir, Y. Himeur, N. Djebari, and A. Amira, "Revolutionizing generative pre-traineds: Insights and challenges in deploying ChatGPT and generative chatbots for FAQs," *Expert Syst. Appl.*, vol. 246, no. January, 2024, doi: 10.1016/j.eswa.2024.123224.
44. T. Rasul *et al.*, "The role of ChatGPT in higher education: Benefits, challenges, and future research directions," *J. Appl. Learn. Teach.*, vol. 6, no. 1, pp. 41–56, 2023, doi: 10.37074/jalt.2023.6.1.29.
45. F. Doshi-Velez and B. Kim, "Towards A Rigorous Science of Interpretable Machine Learning," no. Ml, pp. 1–13, 2017, [Online]. Available: http://arxiv.org/abs/1702.08608
46. M. das G. Bruno Marietto *et al.*, "Artificial Intelligence Markup Language: A Brief Tutorial," *Int. J. Comput. Sci. Eng. Surv.*, vol. 4, no. 3, pp. 1–20, 2013, doi: 10.5121/ijcses.2013.4301.
47. "Pandorabots." https://home.pandorabots.com/dash/help/tutorials (accessed Apr. 12, 2024).
48. A. Jalilifard, V. F. Caridá, A. F. Mansano, R. S. Cristo, and F. P. C. da Fonseca, "Semantic Sensitive TF-IDF to Determine Word Relevance in Documents," *Lect. Notes Electr. Eng.*, vol. 736 LNEE, pp. 327–337, 2021, doi: 10.1007/978-981-33-6987-0_27.
49. M. Abbas, K. Ali, A. Jamali, K. Ali Memon, and A. Aleem Jamali, "Multinomial Naive Bayes Classification Model for Sentiment Analysis," *IJCSNS Int. J. Comput. Sci. Netw. Secur.*, vol. 19, no. 3, pp. 62–67, 2019, doi: 10.13140/RG.2.2.30021.40169.
50. C. Goutte and E. Gaussier, "A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation," *Lect. Notes Comput. Sci.*, vol. 3408, no. June, pp. 345–359, 2005, doi: 10.1007/978-3-540-31865-1_25.
51. M. Y. Day and S. R. Shaw, "AI Customer Service System with Pre-trained Language and Response Ranking Models for University Admissions," *Proc. - 2021 IEEE 22nd Int. Conf. Inf. Reuse Integr. Data Sci. IRI 2021*, pp. 395–401, 2021, doi: 10.1109/IRI51335.2021.00062.
52. M. Jaiwai, K. Shiangjen, S. Rawangyot, S. Dangmanee, T. Kunsuree, and A. Sa-Nguanthong, "Automatized Educational Chatbot using Deep Neural Network," *2021 Jt. 6th Int. Conf. Digit. Arts, Media Technol. with 4th ECTI North. Sect. Conf. Electr. Electron. Comput. Telecommun. Eng. ECTI DAMT NCON 2021*, pp. 85–89, 2021, doi: 10.1109/ECTIDAMTNCON51128.2021.9425716.
53. M. T. Nguyen, M. Tran-Tien, A. P. Viet, H. T. Vu, and V. H. Nguyen, "Building a Chatbot for Supporting the Admission of Universities," *2021 13th Int. Conf. Knowl. Syst. Eng.*, pp. 1–6, 2021, doi: 10.1109/KSE53942.2021.9648677.

 doi:10.20944/preprints202410.0276.v1

54. L. Fauzia, R. B. Hadiprakoso, and Girinoto, "Implementation of Chatbot on University Website Using RASA Framework," *2021 4th Int. Semin. Res. Inf. Technol. Intell. Syst. ISRITI 2021*, pp. 373–378, 2021, doi: 10.1109/ISRITI54043.2021.9702821.
55. Q. Jhqwv *et al.*, "Intelligent Agents in Educational Institutions NEdBOT-NLP-based Chatbot for Administrative Support Using DialogFlow," in *2022 IEEE International Conference on Agents (ICA)*, 2022, pp. 30–35. doi: 10.1109/ICA55837.2022.00012.

Kanaan Mikael holds a B.Sc. in Computer Science from Sulaimani University, Iraq, in 2006 and an M.Sc. in Information Technology from BAMU University, India in 2012. He is currently a Ph.D. student at Sakarya University, Turkey. With 12 years of teaching experience, Kanaan is a lecturer at the University of Human Development in Iraq. He was awarded second place in the Huawei ICT Competition 2023-2024, a Middle East-wide event held in Bahrain. His research interests encompass Natural Language Processing (NLP), Machine Translation, Machine Learning, and Dialogue systems.

56. Cemil Oz was born in Cankiri, Turkey, in 1967. He received his B.S. degree in Electronics and Communication Engineering in 1989 from Yildiz Technical University and his M.S. in Electronics and Computer Education in 1993 from Marmara University, Istanbul. During the M.S. studies, he worked as a lecturer at İstanbul Technical University. In 1994, he began his Ph.D. in Electronics Engineering at Sakarya University. He completed his Ph.D. in 1998. He worked for three and half years as a research fellow at the University of Missouri-Rolla, MO, USA. He has been working as a professor in the Faculty of Computer and Information Science, Department of Computer Engineering at Sakarya University. His research interests include computer vision, artificial intelligence, virtual reality, and pattern recognition.

TARIK A. RASHID received a Ph.D. degree in computer science and informatics from the College of Engineering, Mathematical and Physical Sciences, University College Dublin (UCD), Ireland in 2006. He joined the University of Kurdistan Hewlêr, in 2017. He is a Principal Fellow for the Higher Education Authority (PFHEA-UK) and a professor in the Department of Computer Science and Engineering at the University of Kurdistan Hewlêr (UKH), Iraq. He is on the prestigious Stanford University list of the World's Top 2% of Scientists for the years 2021, 2022, and 2023. His areas of research cover the fields of Artificial Intelligence, Nature Inspired Algorithms, Swarm Intelligence, Computational Intelligence, Machine Learning, and Data Mining.