Received Date: 31st May, 2024
Revision Date:7th June, 2024
Accepted Date: 15th July, 2024

College Chatbot Using RASA

Yubraj Sigdel [1*], Sujan Shrestha [2], Nabin Neupane [3]

[1]Dept of Computer Engineering, Kathmandu Engineering College, E-mail: yubrajsigdel1@gmail.com
[2]Assoc Professor, Dept of Electronics, Communication, and Information Engineering, Kathmandu Engineering College, E-mail: sujan.shrestha@kecktm.edu.np
[3]Assoc Professor, Dept of Computer Engineering, Kathmandu Engineering College, E-mail: nabin.neupane@kecktm.edu.np

***Abstract*— The rapid advancement of Artificial Intelligence, Machine Learning, etc., has brought about significant changes in the technological sphere for decades. One such transformative technology is a Chatbot, a Conversational AI capable of addressing users' queries. These chatbots are tailored to serve various objectives, one of which is to streamline tasks in education through automation. In line with this, a paper titled "College Chatbot Using RASA" has been written, which can effectively respond to questions based on college and other college-related information. The chatbot has been made accessible via an API utilizing the Streamlit framework. The RASA framework, a key open-source technology, has been harnessed to enable the chatbot. RASA plays a vital role in the chatbot's functionality by combining two main components, Rasa Core and Rasa Natural Language Understanding (NLU). Rasa NLU component is used to deduce intent and extract necessary entities from user input, while Rasa Core generates output by constructing a probabilistic model. The model is evaluated by getting a confusion matrix and prediction confidence distribution. The chatbot, upon testing, showcases the best result of 1 for each accuracy, precision, recall, and F1 score.**

***Keywords*— "Chatbot", "Conversational AI", "RASA", "Streamlit", "API", "Rasa NLU", "Rasa Core"**

Introduction

Conversational systems are becoming more pervasive for human-computer interaction as we seek more natural ways to integrate automation into everyday life [1]. Integrating Artificial Intelligence (AI) has revolutionized various aspects of today's rapidly advancing world. One of such aspects revolutionized by AI is educational institutions, which help enhance efficiency, accessibility, and user experience in the educational landscape. Chatbots and their use within educational institutions have gained significant attention among the many applications, as they leverage Artificial Intelligence (AI) and Natural Language Processing (NLP) to guide users to desired output.

The College Chatbot is a versatile tool that caters to diverse stakeholders within the college community, including prospective students, current students, faculty, staff, and alumni. By harnessing natural language processing (NLP) algorithms and machine learning techniques, the chatbot aims to streamline various processes, facilitate access to information, and provide personalized assistance across various domains, including admissions, academic advising, campus resources, and engagement.

This paper explores the design, development, and implementation of a chatbot tailored for a college environment, leveraging the Rasa framework—a widely used open-source platform for building AI-powered conversational agents. Through a detailed examination of the functionalities and features of the College Chatbot, this research elucidates the potential benefits, challenges, and implications of integrating such technology within academic institutions.

Additionally, this paper digs into the underlying architecture, dialogue management strategies, training data preparation, and deployment considerations in developing the College Chatbot using the Rasa framework.

Talking about chatbots there are 5 types of chatbots available, and they are as follows [2]:

A. Menu or button-based chatbots:

Menu-based or button-based chatbots are the most basic form where users can interact with them by clicking the button from the predefined menu that best represents the user's choice. The chatbot prompts another set of predefined menus and shows until users get to the suitable options.

B. Rule-based chatbots:

Apart from Menu-based chatbots, rule-based chatbots employ a bunch of if/else conditional logic to develop conversation automation flows. This chatbot acts as an FAQ, where the chatbot programs predefined combinations of questions and answers so that the chatbot understands users' input and responds accordingly.

C. AI-powered chatbots:

While the rule-based chatbot's conversational flow only supports predefined questions and answer options, an AI chatbot can understand users' questions no matter how they're rephrased. With AI and NLU capabilities, the bot can understand all the contextual information the user shares.

* *Corresponding Author*

D. Voice-based chatbots:

A voice chatbot is another conversational tool that allows users to interact with the bot by speaking to it rather than typing all its queries.

E. Generative AI-powered chatbots:

Generative AI-powered chatbots can be referred to as the next generation of chatbots. Their generative AI capabilities offer enhanced functionality via their fluency in understanding common language, ability to adapt to users' conversation style, and use of empathy while answering questions.

Literature Review

The section below discusses previous work while building chatbots in different domains. Dinesh et al. proposed an AI Bot for Academics to schedule online classes while affected by the COVID-19 pandemic, and their primary objective was to provide an on-demand and updated schedule for each class of the day. The authors further mentioned that the chatbot is useful for those students who were in isolation due to COVID-19 and need to get accurate and real-time information [3].

Swati et al. proposed a chatbot to provide the ground truth regarding COVID-19 as false information arose. Their chatbot also included the functionality of Rapid API, designed to provide real-time information regarding COVID-19. The bot developed by Swati et al. is used to fetch real-time COVID-19 data from the source based on the location the user provides [4].

The paper titled "Rasa: Introduction to Open Source and Dialogue Management" by Bocklisch et al. introduces Rasa Core and Rasa NLU, which can be used to create chatbots. The authors have developed this library to enable anyone with limited knowledge to build a customized chatbot for any field. However, the library is still under development to make Rasa NLU more robust to spelling mistakes and to support more languages [1].

The paper proposed by Meshram et al. explains how to collect inquiries using a chatbot using Rasa, reducing the tedious load on college staff. They proposed a web-based chatbot, where users can interact with the chatbot and get their queries answered instead of doing inquiries with college staff either physically or via email [5].

The paper proposed by Fauzia et al. explains how a Chatbot was implemented on the university website. The authors stated that the chatbot was developed to transfer information regarding admissions to its users conveniently, and they even dockerized the application so that it could be shipped or used as a widget [6].

An analytical study and review of Rasa – An open-source technology for creating conversational AIs is presented in this paper [7]. The authors stated that Rasa's core features, like slots, forms, supervised interactive learning, API integration, and so on, make it more capable than any other open-source alternative.

Methodology

RASA is an open-source framework designed to create conversational systems that leverage NLP and AI. This conversational framework allows us to understand, hold conversations, and connect to messaging channels and third-party systems via APIs. It is the building block for creating virtual assistants or chatbots [8].

F. Architecture

RASA's architecture is modular in design and contains several components that work together to enable the creation of conversational AI applications. The figure below is a detailed deception of RASA Open Source's architecture. It consists of two major components: Rasa NLU and Rasa Core.

Rasa SDK is an action server that handles custom actions, such as querying the database and making some API calls. Other components are related to trackers and trained models necessary for the system [3].

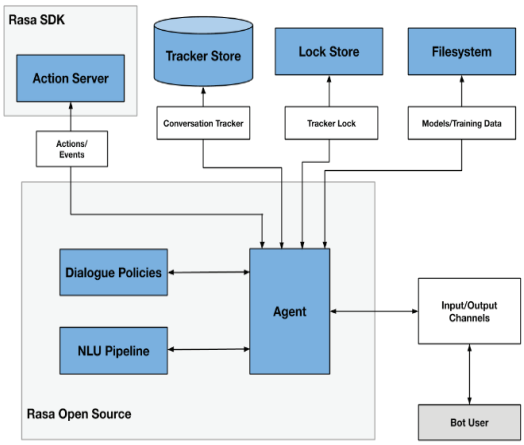

Fig. 1 RASA Architecture [9]

The diagram below shows the basic steps of how an assistant built with Rasa responds to a message:

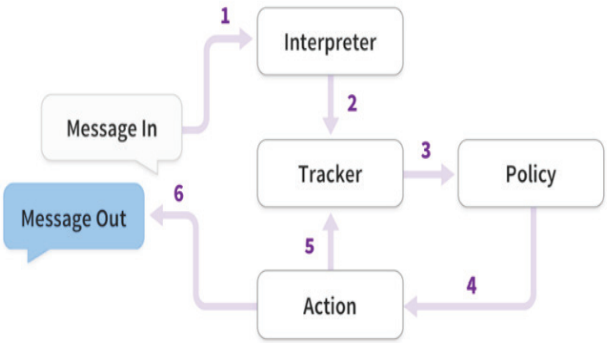

Fig. 2 RASA Message Handling [1]

1. The user's message is received and passed to the interpreter (Rasa NLU) to extract intent, entities, and other structured information.
2. The tracker maintains a conversation state, including the dialogue history, user inputs, and bot responses. It stores information about the context of the current conversation and is used by Rasa Core to decide on the next best action.
3. The policy receives the current state of the tracker.
4. The policy decides which action will be invoked next.
5. The tracker logs the chosen action.
6. The action is then executed. This may be uttering a simple message or performing custom actions, such as querying the database and providing the response.
7. If the predicted action is not "listen," go to step 3.

G. Actions

We classify dialogue management as a classification problem. Rasa Core predicts which action to take from a predefined action list at each iteration [1]. An action can be a simple utterance or a custom action that can run any code we want, such as an API call or querying the database [8]. When an action is executed, it is passed with the tracker instance. It can use any relevant structured information captured over the conversation history, such as entities and previous utterances.

H. Natural Language Understanding

This is the major component invoked after receiving a user message. Rasa NLU is responsible for understanding and processing the user's input, extracting structured data such as intents (the user's intention) and entities (relevant pieces of information). Rasa NLU uses DIETClasifier, a multi-task model for intent classification and entity extraction [8]. DIET, Dual Intent, and Entity Transformer are based on the transformer shared for both tasks. The pipeline represents the configuration for training a machine learning model for Rasa's NLU component. It defines a series of steps that transform raw text input into features the machine can use for classification.

1) WhiteSpaceTokenizer: This component is responsible for tokenizing the input text by splitting it into tokens based on white space characters, such as spaces, tabs, and newlines.

2) RegexFeaturizer: This component extracts features from the text using regular expressions.

3) LexicalSyntaticFeaturizer: This component extracts lexical and syntactic information about sentence structure, such as part of speech and syntactic dependencies.

4) CountVectorsFeaturizer: This component converts the text into numerical feature vectors using a bag-of-words representation. It counts the frequency of each word in the text and represents it as a vector.

5) CountVectorsFeaturizer (with char_wb analyzer): This is another instance of CountVectorsFeaturizer component, but it uses a character-based analyzer with word boundaries. It captures character-level features in addition to word-level features.

6) DIETClassifier: This is the main classifier component responsible for training and making predictions based on input data. It uses Dual Intent and Entity Transformer (DIET) architecture, which combines intent classification and entity extraction into a single model.

7) ResponseSelector: This component predicts the best response from a predefined set of responses based on the input message.

8) FallbackClassifier: This component serves as the backup to the primary classifier, i.e., DIETClassifier, when the primary classifier is uncertain about the correct action to choose.

I. RASA Core

This component handles dialogue management and determines how the chatbot should respond to user inputs based on the current conversation context. It employs a machine learning-based approach to learn from conversational data and predict the next best action.

J. Training Data

To train our chatbot effectively, we have gathered data specific to Kathmandu Engineering College from various sources, including faculty, guardians, and students. This dataset is crucial for teaching the chatbot to understand and respond to common queries related to the college.

The dataset creation process involves organizing the data into three main components: NLU data, rules, and stories. These components are written in YAML file format, making it easy for the chatbot to learn from them.

The NLU data consists of examples of users' probable inputs along with their corresponding intents and entities. Intents represent the user's intention and goal behind the input message, while entities represent the most valuable information present in the message. This data is used for training the NLU model, which is then responsible for

extracting intents and entities from the user's message while inferring. Example:

- intent: Ask_syllabus

  examples: |

- provide me the syllabus for the [bct](program)

- can I get the syllabus for [bei](program)

Similarly, rules define the conversation flow and specify how the chatbot should respond to certain user inputs or events. They are written in a declarative format and specify conditions (triggers) and their corresponding actions. Example:

- rule: Say goodbye anytime the user says goodbye

steps:

  - intent: goodbye

  - action: utter_goodbye

Stories represent example dialogues or conversations between the user and the chatbot. Each story consists of user inputs (intents) and corresponding bot responses (actions). Stories train the dialogue management model, which learns to predict the next best action based on the current conversation context. Example:

- story: Admission Probability Positive Path

steps:

- intent:

admission_with_rank_in_particular_program

    entities:

    - program: bct

    - rank: 500

    - action: action_admission_confidence

K. Visualization of Dialogue Graph

Rasa core can visualize a graph of training dialogues. A story graph is a directed graph with actions as nodes. Edges are labeled with the user utterances that occur in between two consecutive actions; the edge label is omitted. Each graph consists of an initial node called START and a terminal node called END [1]. The graph does not capture the full dialogue state, and not all possible walk-along edges necessarily occur in the training set. An instance of the dialogue graph visualization is shown below:

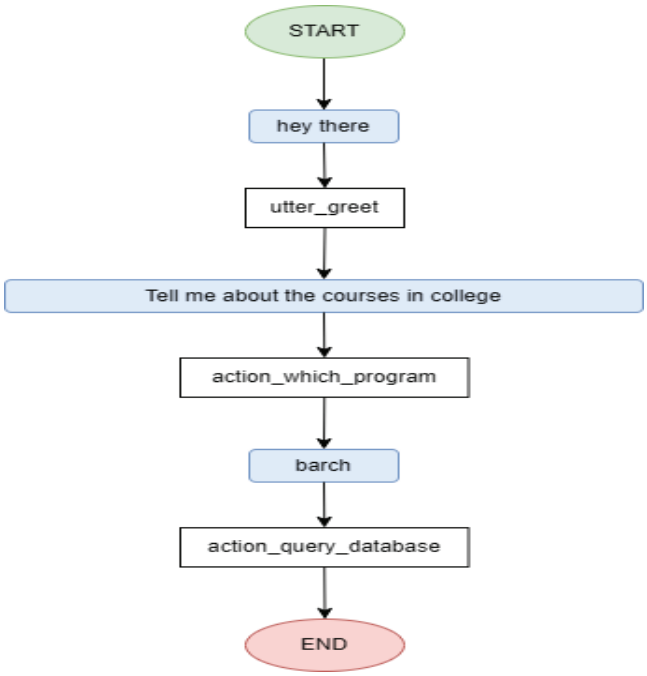

Fig. 3 Story graph visualization

Results

After rigorous testing, our Rasa chatbot has undergone thorough evaluation. Below are the corresponding confusion matrix and confidence histograms, providing insights into its performance. These visual representations comprehensively understand how our chatbot interprets user inputs

and makes predictions effectively. Test stories can test the performance of the intent classifier and response selector.

Rasa's built-in feature generates confusion matrices and confidence histograms for each major component in the pipeline to evaluate the performance of both the NLU and core model.

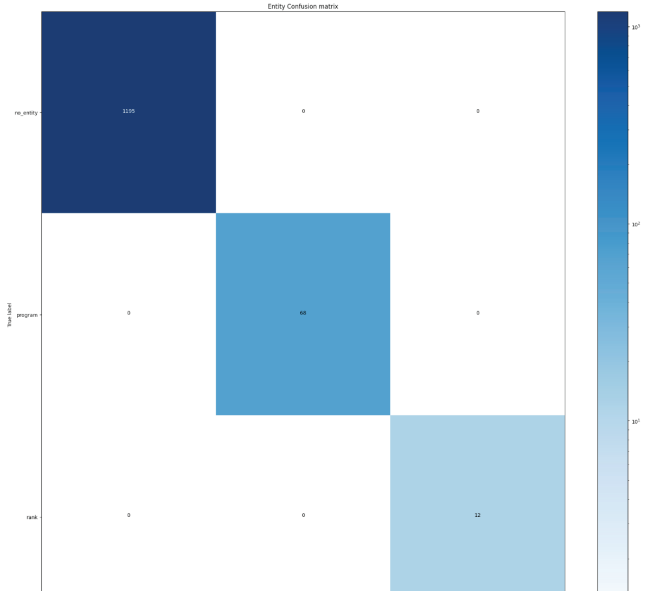

Fig 4. Entity Confusion Matrix

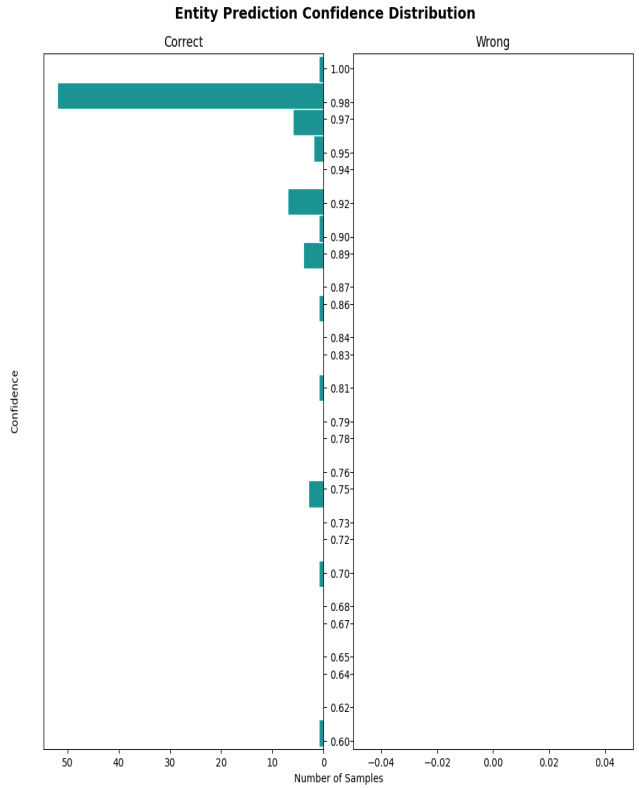

Fig. 5 Entity Prediction Confidence Histogram

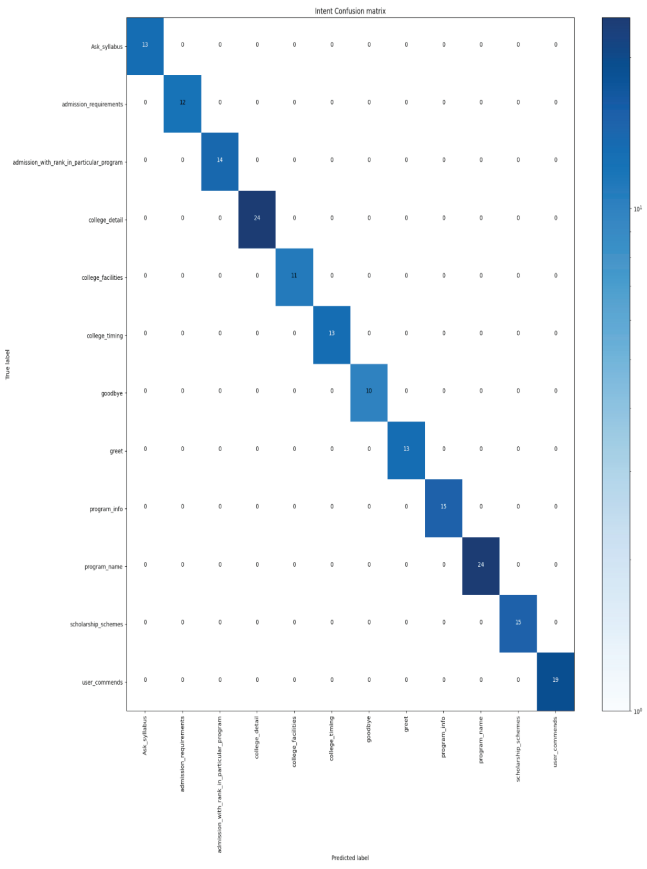

Fig. 6 Intent Confusion Matrix

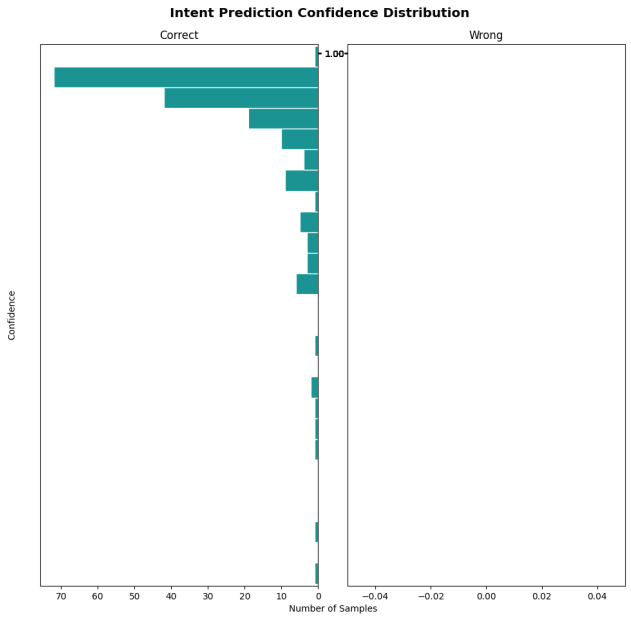

Fig. 7 Intent Prediction Confidence Histogram

TABLE 1

MACRO AVERAGE TEST RESULTS

Precision	1.0
Recall	1.0
F1 score	1.0
Support	80

Discussion and Conclusion

College Chatbot was developed using Rasa, an open-sourced framework that provides great functionalities and an NLP toolbox through its Rasa stack. Rasa provides ease of task deployment and server creation, helping to focus more on adding custom functionalities tailored to each specific domain, in our case, the education domain. The chatbot designed for Kathmandu Engineering College demonstrated strong performance during testing conducted within the framework. Evaluation metrics, including precision, recall, and F1 score, yielded average values of 1 each, indicating excellent performance.

However, there is room for improvement. The scope of improvement lies in the dataset, as we can add more training examples covering users' diverse questions. The functionalities can also be increased by providing results, due fees, class schedules, college events, and more. Currently, the chatbot assists users in getting admission and other college-related information.

Currently, the chatbot is accessible via Streamlit API. In the future, the chatbot can be embedded inside the college's website so that users can ask their queries while accessing the college website, as the information on the website could be vague for most users.

Acknowledgement

I sincerely thank Associate Professor Sujan Shrestha for their invaluable guidance, encouragement, and support throughout this research project. Their expertise and mentorship have been instrumental in shaping the development and direction of this work. Additionally, I appreciate the Department of Computer Engineering and Electronics Engineering at Kathmandu Engineering College for providing the necessary resources and facilities to complete this study. Their assistance has been invaluable.

References

[1] J. F. N. P. A. N. Tom Bocklisch, "Rasa: Open Source Language Understanding and Dialogue Management," *arXiv preprint arXiv:1712.05181,* 2017.

[2] B. Church, "5 types of chatbot and how to choose the right one for your business," IBM, 05 September 2023. [Online]. Available: https://www.ibm.com/blog/chatbot-types/.

[3] T. a. R. A. M. a. N. T. T. a. R. S. G. Dinesh, "AI Bot for Academic Schedules using Rasa," in *2021 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)*, 2021, pp. 1-6.

[4] S. a. L. D. a. A. A. a. S. J. a. L. R. Nikam, "Covid-19 Android chatbot using RASA," in *2022 3rd International Conference for Emerging Technology (INCET)*, 2022, pp. 1-7.

[5] N. N. M. V. T. M. S. K. Siddhant Meshram, "College Enquiry Chatbot using Rasa Framework," in *Asian Conference on Innovation in Technology*, Pune, 2021.

[6] R. B. H. G. Lia Fauzia, "Implementation of Chatbot on University Website Using RASA Framework," *International Seminar on Research of Information Technology and Intelligent Systems,* 2021.

[7] M. J. Rakesh Kumar Sharma, "An Analytical Study and Review of open Source Chatbot framework, RASA," *International Journal of Engineering Research & Technology,* vol. 9, no. 06, 2020.

[8] "Introduction to Rasa Open Source & Rasa Pro," RASA, [Online]. Available: https://rasa.com/docs/rasa/.

[9] "Rasa Architecture Overview," RASA, [Online]. Available: https://rasa.com/docs/rasa/arch-overview/.