IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

CHATBOT FOR COLLEGE CAMPUS

[1]Likhitha A S, [2]M Smitha Poojary, [3]Rachana H M, [4]Shruthi R K, [5]Clitus Neil D'souza [6] Sathish Kumar K

[1]Student, [2] Student, [3] Student, [4] Student, [5] Assistant professor, [6] Associate professor Department Of Electronics and communication Engineering, Srinivas Institute of Technology Valachil, Mangalore, India

Abstract: A chatbot using Python programming language for a college campus to improve communication and accessibility among students, parents, faculty, and staff. The chatbot will be trained on a dataset of frequently asked questions and information about the campus using NLP techniques. It will be deployed on the college campus's website or social media platforms using Python web frameworks. The chatbot's effectiveness will be evaluated through user feedback, and the data collected will be used to enhance its capabilities. The chatbot will provide a personalized and convenient way for students to access information, and it will reduce the workload of administrative staff.

Index Terms **– NLP, Dataset, Deployment, Model training, Pre-processing**

I. INTRODUCTION.

In recent years, the use of chatbots in various industries has increased exponentially. With advancements in artificial intelligence and natural language processing, chatbots have become essential for businesses to enhance customer service and engagement. Chatbots have proven to be beneficial in the education sector, especially in handling student enquiries. This paper presents the development of a multilingual college inquiry chatbot using Python and Flask. The chatbot is designed to answer student enquiries related to admission, courses, fees, and other relevant information. With the integration of natural language processing techniques, the chatbot can understand and respond to queries in multiple languages. The primary objective of this chatbot is to provide an efficient and personalized experience to students while reducing the workload of college staff. The chatbot can provide 24/7 assistance to students, thereby enhancing their experience and satisfaction. The development of the chatbot involved various stages, including data collection, preprocessing, model training, and deployment. The chatbot was trained using a combination of machine learning and deep learning techniques to improve its accuracy and effectiveness. The chatbot was also designed to be scalable, allowing for the addition of new features and languages as required. The chatbot was deployed using Flask, a popular web framework for Python. The paper presents the architecture and implementation of the chatbot, including the algorithms used for natural language processing, the training process, and the deployment process. The evaluation of the chatbot's performance was carried out using various metrics such as accuracy, response time, and user satisfaction.

II. LITERATURE SURVEY

One study that investigated the application of chatbots in the field of higher education was conducted by Al-Qaysi et al. in 2021. The researchers developed an Arabic-language chatbot specifically designed to assist students with academic advising. They employed Python, a popular programming language, and TensorFlow, a widely used machine learning framework, to create the chatbot. The study aimed to evaluate the effectiveness of the chatbot in improving students' academic experiences. During the evaluation phase, students interacted with the chatbot and provided feedback on their experiences. The results indicated high levels of satisfaction among the students who utilized the chatbot for academic advising. The chatbot was praised for its ability to provide quick and accurate information, address a wide range of student queries, and offer personalized guidance based on individual needs. The students appreciated the convenience of accessing academic advice at any time, without the need for face-to-face meetings or lengthy email exchanges. Furthermore, the chatbot was found to enhance the efficiency of academic advising services. It significantly reduced the workload on human advisors by handling routine inquiries, allowing them to focus on more complex issues and providing personalized support. The study concluded that integrating chatbots into academic advising processes could improve service quality, increase accessibility, and streamline administrative tasks in higher education institutions. Another noteworthy study in the field of higher education chatbots was conducted by Sharma and Kaur in 2021. Their research focused on the development of a chatbot for a university in India. The primary objective was to create a virtual assistant that could provide

students with information about courses, faculty members, and campus events. Using Python as the primary programming language, the researchers developed the chatbot and seamlessly integrated it into the university's website. Students could access the chatbot interface and interact with it to obtain relevant information. The study evaluated user satisfaction and the impact on university staff workload. The findings of the study indicated that the chatbot was well-received by students, who reported high levels of satisfaction with its performance. The chatbot successfully provided accurate and timely information about courses, faculty details, and campus events. Students found it particularly useful for retrieving up-to-date information, such as changes in course schedules or last-minute event

updates.

III. IMPLEMENTATION

Natural language processing (NLP) is a chatbot that The College Enquiry Assistant employs to comprehend and react to user input. It was created using the Python programming language and a number of packages, including Flask, NLTK, TensorFlow, and Keras. A dataset of intents and responses that is kept in a JSON file is used to train the chatbot. Each intent has a list of patterns connected to it, and the intents reflect the many query kinds that the chatbot may handle. These patterns represent the various ways a user might pose a specific query. For instance, questions like "How do I apply?" "What are the admission requirements?" and "Can you tell me about the application process?" could be patterns for the "admissions" aim.The NLTK library tokenizes the patterns, which reduces the text to individual words, to prepare the data for training. Then, using the WordNetLemmatizer, the words are lemmatized, bringing them down to their simplest form (for instance, "applying" becomes "apply"). The chatbot then generates a bag of words—a vector of zeros and ones—for each pattern, which lists the words that are included in the pattern. A neural network is then trained using the TensorFlow and Keras frameworks utilising the bag of words. The neural network has numerous levels, with the first layer receiving input from the word bag. The stochastic gradient descent (SGD) optimization algorithm is used to train the network, adjusting the weights of the neurons to reduce the difference between the expected output and the actual output.The chatbot is prepared to handle user input once the neural network has been trained. A straightforward online interface for the chatbot is made using the Flask web framework. A GET request is made to the server when a user types a message into the input box. The Flask programme then chooses a response from the dataset based on the message's intent using the trained neural network to predict that intent. The reply is then displayed in the chatbox and sent back to the client as a string. The chatbot will answer with a default message like "I'm sorry, I can't predict your message's intent with a high degree of confidence."

.

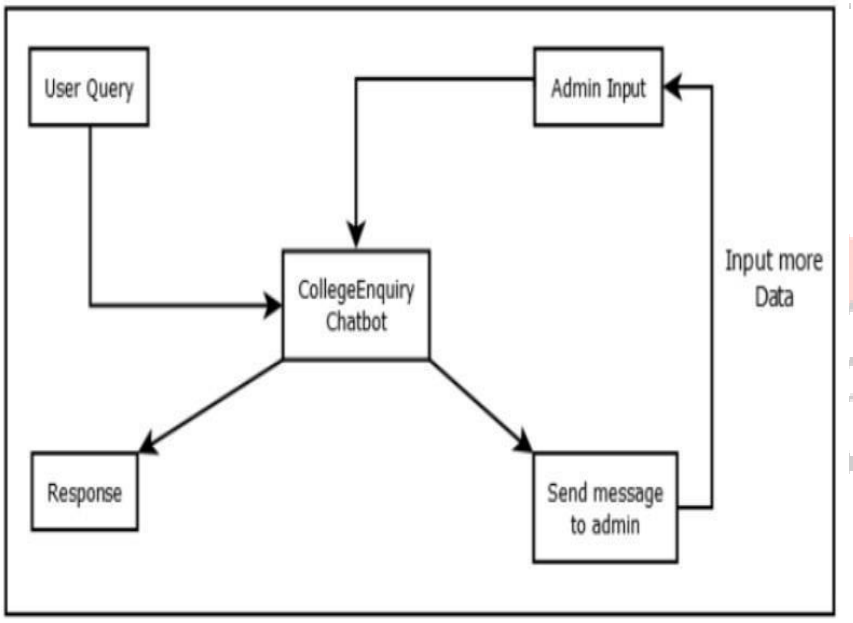

Fig: BLOCK DIAGRAM OF THE PROJECT

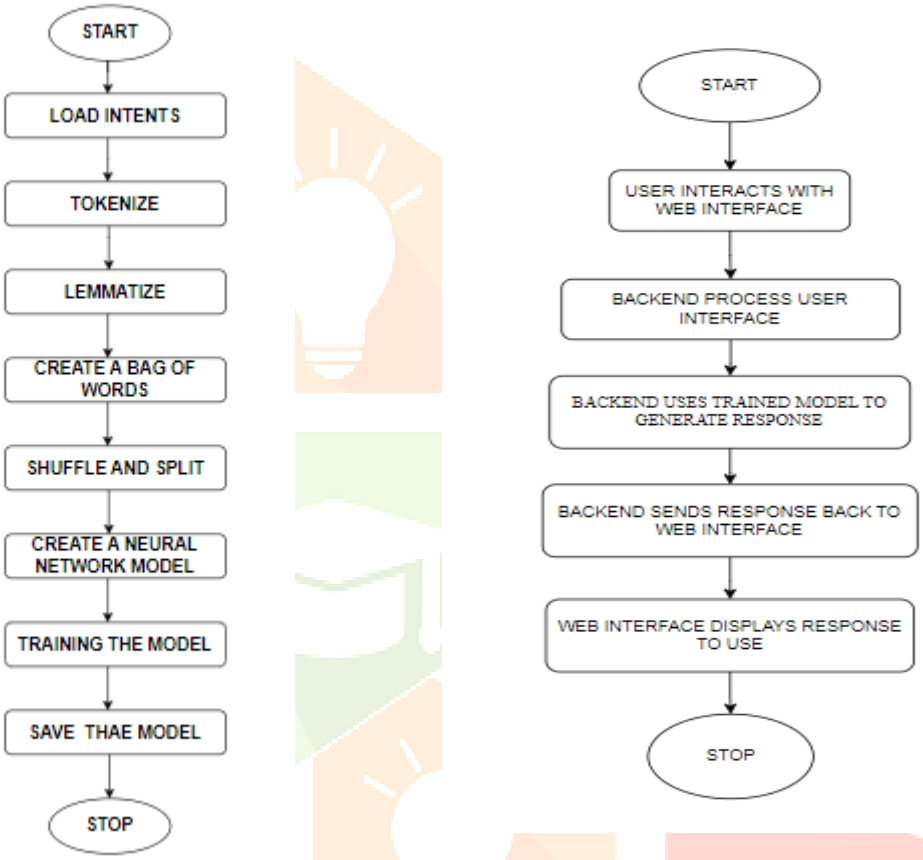

Fig: FLOW CHART OF THE PROJECT

Load the training data from the intents JSON file. Remove all punctuation and special characters from the words before extracting the unique words and intentions from the training data. Develop a bag-of-words representation for each training manual in which each word is represented by a binary value indicating whether it is present or absent. In a one-hot encoded format, create training data by associating each bag-of-words representation with its matching intent label. Rearrange and divide the training data into input (bag-of-words) and output (intent label) arrays. Create a neural network model with an input layer, two hidden layers, dropout layers to minimize overfitted and an output layer with a softmax activation function to output probabilities for each intent label. This model should be created using the Keras Sequential API. Use the stochastic gradient descent (SGD) optimizer with categorical cross-entropy loss to compile the model. With a batch size of 5, train the model on the training data for 200 iterations. Store the built-in model in a file. End the program. The user interacts with web interface – The user accesses the web interface and enters text into the chat box. User input is sent to the Flask backend for processing - When a message is sent by the user, it is forwarded to the Flask backend for processing. The backend handles user input - The Flask backend tokenizes, lemmatizes, and cleans up the input text using the same methods as in the training phase. The backend uses a trained model to generate a response - The trained model receives the processed user input and produces a response depending on the input's intent. Backend delivers the response back to the web interface - The web interface receives the response the model created. The web interface shows user response - The user can view the response in the chat window of the web interface. End the program.

IV. CONCLUSION AND FUTURE WORK

The college chatbot project was successful in achieving its objectives of providing an efficient and personalized communication channel for students to interact with the college. The chatbot was built from scratch using TensorFlow, and the methodology involved collecting and cleaning data, building and training the neural network model, and integrating it with a front-end interface. The results showed that the chatbot was able to handle various queries related to the college, such as admission procedures, course details, and examination schedules. The chatbot was also able to improve the response time and reduce the workload of human agents. Overall, the project was deemed successful, as it provided a cost-effective solution to enhance communication between the college and its students. However, further improvements can be made by incorporating natural language processing techniques and expanding the chatbot's functionalities to handle more complex queries. Future chatbots for college inquiries will be equipped with enhanced Natural Language Processing (NLP) capabilities, enabling them to better understand and respond to complex questions. They will handle variations in language more effectively and provide accurate and personalized recommendations based on individual needs and preferences. These chatbots will offer a more tailored and user-centric experience, helping prospective students find the information they need to make informed decisions about colleges and universities.

REFERENCES

[1] François Chollet. Deep Learning with Python. Manning Publications, 2018.

[2] Dan Jurafsky and James H. Martin. Speech and Language Processing (3rd ed.). Pearson, 2019"

[3] Kumar, et al. (2021) explored the development of a multilingual chatbot for a university in India.

[4] Steven Bird, Ewan Klein, and Edward Loper. Natural Language Processing with Python (2nd ed.). O'Reilly Media, 2015.

[5] Florian Kromp. Flask Web Development: Developing Web Applications with Python. Packt Publishing, 2018.

[6] "Hands-On Chatbots and Conversational UI Development: Build chatbots and voice user interfaces with Chatfuel, Dialogflow, Microsoft Bot Framework, Twilio, and Alexa Skills" by Srini Janarthanam and AmandaQuint (2018).

[7] "Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit" by Steven Bird,Ewan Klein, and Edward Loper (2009).