A Project/Dissertation Review-1 and Review-2 Report

on

Intelligent Chatbot using Natural Language processing

Submitted in partial fulfillment of the

requirement for the award of the degree of

B. Tech in CSE

GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Under The Supervision of

Ms. Priyanka Shukla

Assistant Professor

Submitted By

Mohammad Taliv

19SCSE1010028

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GALGOTIAS UNIVERSITY, GREATER NOIDA

INDIA

APRIL ,2023

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the project, entitled **"Intelligent Chatbot using Natural Language Processing."** in partial fulfillment of the requirements for the award of the Bachelor of Technology in Computer Science Engineering submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of August 2022 to December 2002, under the supervision of Ms. Priyanka Gosh, Asst. Professor Department of Computer Science and Engineering/Computer Application and Information and Science, of School of Computing Science and Engineering, Galgotias University, Greater Noida

The matter presented in the thesis/project/dissertation has not been submitted by me/us for the award of any other degree of this or any other places.

Mohammad Taliv, 19SCSE1010028

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Ms. Priyanka Shukla

Asst. Professor

CERTIFICATE

The Review-1 and Review-2 Thesis/Project/ Dissertation Viva-Voce examination of Mohammad Taliv – 19SCSE1010028 has been held on _________________ and his/her work is recommended for the award of

Signature of Examiner(s) **Signature of Supervisor(s)**

Signature of Project Coordinator **Signature of Dean**

Date: April, 2023

Place: Greater Noida

Abstract

A chatbot or colloquial agent may be a software system that may communicate with an individual by victimization language. One of the essential tasks in the AI and Natural language process is the modeling of language. Since the start of AI, it's been the most formidable challenge to form a decent chatbot. Through chatbots, humans will perform several tasks. The first operation they need to perform is to know the words of humans and to reply to them suitably. In the past, straightforward data ways or written templates and rules were used for the construction of chatbot architectures. With increasing learning capabilities, end-to-end neural networks replaced those models in around 2015. To be specific, encoder-decoder repetition models are currently dominant in dialogue modeling.

This design is taken from the neural artificial intelligence domain, and it performed well there. Until now, lots of options and variations have been introduced that have remarkably increased the colloquial capabilities of chatbots. In this paper, we tend to perform an in-depth survey of recent literature. We tend to examine several publications from the last 5 years associated with chatbots. At that time, we had a tendency to confer completely unusual linked works to our subject, and therefore the AI ideas required to create associated intelligent colloquial agents supported by deep learning models Finally, we tend to propose a practical design that we have a tendency to create an intelligent chatbot for health care help.

Table of Contents

List of Tables

List of Figures

Acronyms

B.Tech.	Bachelor of Technology
M.Tech.	Master of Technology
BCA	Bachelor of Computer Applications
MCA	Master of Computer Applications
B.Sc. (CS)	Bachelor of Science in Computer Science
M.Sc. (CS)	Master of Science in Computer Science
SCSE	School of Computing Science and Engineering

CHAPTER-1
Introduction

Chatbots are intelligent conversational computer programs that mimic human conversation in its natural form. A chatbot can process user input and produce an output. Usually, chatbots take natural language text as input, and the output should be the most relevant output to the user input sentence. Chatbots can also be defined as "online human-computer dialogue system(s) with natural language" . Chatbots constitute therefore an automated dialogue system, that can attend to thousands of potential users at once. Chatbots are currently applied to a variety of different fields and applications, spanning from education to e-commerce, encompassing healthcare and entertainment. Therefore, chatbots can provide both support in different fields as well as entertainment to users; this is the case for chatbots such as Mitsuku and Jessie Humani, "small talk" oriented chatbots that could provide a sense of social connection. Chatbots appear, in fact, to be more engaging to the user than the static Frequently Asked Questions (FAQ) page of a website. At the same time, chatbots can simultaneously assist multiple users, thus resulting more productive and less expensive compared to human customer supports services. In addition to support and assistance to customers, chatbots can be used for providing entertainment and companionship for the end user. Nonetheless, different levels of embodiment-the way chatbots are human-like -and disclosure-how and when the nature of the chatbot is revealed to the user-seem to impact users' engagement with and trust in chatbots . In recent years, with the commoditization and the increase of computational power and the sharing of open source technologies and frameworks, chatbots programmes have become increasingly common. Recent developments in Artificial Intelligence and Natural Language Processing techniques have made chatbots easier to implement, more flexible in terms of application and maintainability, and increasingly capable to mimic human conversation. However, human-chatbot interaction is not perfect; some areas for improvements are contextual and emotional understanding and gender biases. Chatbots are, in fact, less able to understand conversational context and emotional linguistic cues compared to humans, which affects their ability to converse in a more entertaining and friendly manner. At the same time, chatbots tend to take on traditionally feminine roles which they perform with traditionally feminine features and often displaying stereotypical behaviour, revealing a gender bias in chatbots' implementation and application . Since chatbots are so widespread and applied to many different fields, improvements in their implementations and evaluation constitute important research topics. The main contributions of this paper are: (i) extensive survey of the literature work on chatbots as well as the state of the art on chatbots' implementation methods, with a focus on

Deep Learning algorithms, (ii) the identification of the challenges and limitations of chatbots implementation and application, and (iii) recommendation for future research on chatbot. The rest of this article is organized as follows: first provides some background on chatbots and their evolution through time, then describes the methodology, then presents an analysis of the state of the art in terms of chatbots Deep Learning algorithms; including the datasets used for training and evaluation methods, then we will discuss related works and we conclude the paper in last.

CHAPTER-2
Literature Survey

3.1 A review on Chat-

Interface This unit is the front end of the system. It is responsible for collecting the user queries from the user which are the input to the system. It is also responsible for displaying the system generated results to the user. Therefore, it can be said that the chat interface is the face of the system through which the entire communication takes place. It is the mediator of conversation between the system and the user. The query that user fires on the chat interface is passed on to the chatting backend which acts as a message delivering system between the Chat interface and the Machine Learning Layer. This interface can be accessed either as a website or as a smart phone app. The type of interface depends on the requirements of the user that are to be satisfied by the system. If the system is accessed from a smartphone, the interface will be in the form of an app and if the system is accessed from a website, then the interface will be in the form of a website. For building apps on the smartphone, it will require to use android for android phones or Swift for iOS. In this case, only the interfacing platform will be programmed on android and the complete backend processing of the system will take place on a server on which the system will be deployed. For making a website, either Java or Python web frameworks can be used. Java provides Spring and Struts as the most advanced and latest web frameworks. Similarly, Python allows usage of Django and Flask frameworks for building of a website. The criteria for selection of the programming language depends upon the functionalities that the system intents to provide, the requirements of the users that will use the system, the algorithms that are to be used by the system, etc. Selection of an appropriate programming language makes it simpler for developers to develop a system which provides maximum functionality to the user with high accuracy and minimum complexity.

3.2 A review on NLU Engine-

NLU i.e. Natural Language Understanding is a subpart of NLP (Natural Language Processing) which enables the system to understand the natural language or the conversational language spoken by the users. The conversational language used by humans for day to day conversations is not as perfect as the formal language. It does not focus much on the vocabulary and the grammar. Hence, it becomes difficult for a system to understand the intent of the sentence. The input received from the user is in unstructured text format which cannot be understood by the system directly. It understands input only in structured formats. The unstructured text received from the user is converted to structured format by extracting important words and patterns from the user text using the NLU techniques. Humans are capable of understanding any mispronunciations, homophones, swapped words, shortened form of words (like „it‟s‟ for „it is‟), slang words or phrases and also words which are not used in formal vocabulary but exist in regular conversations. NLU techniques enables the system to identify these twerks if the user makes use of them while conversing with the chatbot, so as to make the user feel that the conversation is taking place between two humans and not between a human and a bot. NLU systems do not directly understand the meaning of the user sentences. It involves a sequence of processes to derive the actual intent of the sentence. To understand a complete sentence, the NLU system needs to understand each word of that sentence. It means that the initial task is the segmentation of the sentences into individual words. Next, to understand the word, the system needs to understand the grammar of the sentence. This can be done by knowing the parts of speech of each word in that sentence. Here comes the POS (Parts-

Of-Speech) tagger into picture. After knowing the grammatical weightage of each word, all of them are parsed to know the dependency among them. This is the most important step wherein the word with the highest dependency is extracted, from which the intent of the system can be known. It is not possible that the knowledge base would contain the exact sentence that the user has sent. It might contain a sentence with the same intent but with different words used in it. To match these types of synonymic sentences, synonym determination and sentence matching are required. The different tasks to be implemented under the NLU Engine and the methods to do the same have been discussed further.

3.2.1 A review on Word Segmentation

Segmentation, also referred to as tokenization is the process of splitting text into smaller and meaningful units. These units could be paragraphs, sentences, clauses, phrases, words or letters. The smallest unit are the letters. Word segmentation is the splitting of sentences into individual words separated by blank spaces. The tokenized units of the sentences are called as tokens. The tokenizers split the sentences into words and punctuations marks as independent units. The most commonly used tokenizer is of space type, i.e. it splits the sentences into words at the blank spaces. It is also required that the tokenizer should consider abbreviations, acronyms, dates, numbers in decimal formats, etc., which cannot split at punctuations and blank spaces, as they will lose their meaning if done so. **Mohammed Javed et al**. [1] [2015] explained a method to implement word segmentation. He proposed in his algorithm to calculate character spaces in the sentences. The character spaces should include all types of gaps between characters. They include the gaps between letter, punctuations and the words. The algorithm functions on the basis of the amount of gap or character space between each unit in the sentence. After the calculation of character spaces, an average of the gaps is calculated to know the mean average between characters in the sentence. This average gap distance is then applied to the sentence which is to be segmented. The places at which the character space is more than the average character space are said to be the points of tokenization. The gap between words is always more than the average gap and hence tokenization takes place at the blank spaces between words in the sentences.

Naeun Lee et al. [2] [2017] proposed the implementation of word segmentation using NLTK. Natural Language ToolKit (NLTK) is a python package which caters to provide services for NLP. It has inbuilt tokenizers. Users need to import the package and use the required type of tokenizer which is present in the form of functions. The NLTK includes a wide range of tokenizers which are as follows standard, letter, word, classic, lowercase, N-gram, pattern, keyword, path, etc. The most commonly used tokenizer is the word-punkt tokenizer which splits the sentences at the blank spaces. The accuracy, speed and efficiency of the NLTK tokenizers is commendable. Also, it does not require any algorithm implementation as the package executes them at the backend. **Tao Jaing [3]** [2011] explains the usage of CRF (Conditional Random Fields) Algorithm for word segmentation. This algorithm trains the system for spaces between the characters. Using this training, the system identifies the gap between characters in the test sentence. The system keeps a threshold value for the gap distance. If the value of gaps in the test sentence is more than the specified threshold, then the sentence splits at those points. CRF requires a lot of training to be given to the system, which makes the process time consuming. Comparing the three methods illustrated above, the NLTK proves to be more efficient in all aspects as compared to the other two. The usage of NLTK does not require the implementation of any algorithm as everything is taken care by the package itself. Also, the accuracy, speed and diversity provided by the package is better than the two algorithms.

3.2.2 A review on POS Tagging

POS Tagging is the process of assigning grammatical annotations to individual words in the sentences. These annotations include the Parts-Of-Speech Tags. They denote the grammatical importance of the word in the sentence based on the dependency of that word with other words in that phrase, clause, sentence, paragraph, etc. The common POS tags are noun, verb, pronoun, etc. There are number of ways which can be used to perform the POS Tagging. Some of them are explained below.

Jerome R. Bellegarda [4] [2010] proposed a method called latent analogy for POS Tagging. In this algorithm, latent semantic mapping (LSM) technique is used. It requires the training on the available corpus. The LSM maintains a feature space of the trained corpus which has been tagged. Now, new sentences are provided to the LSM for tagging and the analysis is performed so as to determine the sentences from the training data which are closest to the test sentence. This is called as sentence neighborhood. Sentence neighborhood holds true for two sentences if they share the same intent matter. Once the intent matching sentences are found from the trained data, the POS tags attached to those sentences are then mapped to the test sentences**. Liner Yang et al. [5]** [2018] put forth the technique of implementing the POS Tagger using Neural Networks. This algorithm consists of „n" numbers of hidden layers. Theses layers are determined by the number of iterations or combinations required to tag the required sentence correctly. At each layer of the algorithm, each word in the sentence is tagged with an appropriate POS tag and then passed to the next later for checking the correctness of the tags. This keeps happening unless the next layer provides the same tags as provided by the previous layer. Another technique to implement the POS tagger is following the traditional approach i.e. of maintaining a dictionary of tags for the given language. Python NLTK provides an inbuilt Tagger which can be used just by importing the NLTK package. The NLTK has a predefined set of tags and a trained data of its own. It tests the sentence and applies an appropriate tag to it. On comparing the above three algorithms, the NLTK tagger proves to be speed and usage efficient. But highest accuracy is provided by the neural network algorithm as it undergoes many iterations.

3.2.3 A review on Dependency Parsing

A dependency parser is used to establish the relationship between words in a sentence based on the grammatical tags attached to it. It is the next step after parsing. A dependency tree or graph is created for every sentence. This tree is called as the parsing tree or the dependency tree. There are a number of ways by which the parsing can be implemented. The comparison of the same is expressed below. **Bo Chen [6]** [2011] proposed a method for implementing the dependency tree. It initially finds out the dependencies among the words in the sentence. Each word is checked for its relationship or dependency with the other word. The word with the highest dependency is selected to be the root. The other words with a relation with the root node are attached to it as the child nodes. This keeps on continuing until all the words are placed in the tree. The tree form of the sentence is called the dependency parser tree. The dependencies among the words are found out by using the POS tags**. Zhenghua Li [7]** [2014] provided a further improvised model of the dependency parser. In the traditional method mentioned above the parser creates a parsed tree for the required sentence. In the graph-based dependency parser, the tree created is converted to a graph where the words in the sentences are the vertices and the dependency between the words are the represented by the edges. This data structure shows a better representation of the parsed sentence. Parsing is always to be performed by the traditional method. But graph-based parser improves the visibility, readability and understandability of the parser.

3.2.4 A review on Synonym and Pattern Recognition

For information retrievals, no matter how big our data is, no sentence sent by the user can be perfectly same to any sentence in the database. But there can be sentences with the same intent. After understanding the intent of the user sentence, the database is checked for a sentence with the same intent. The matched sentences have difference of words which are used to express the same content. They use alternative words or synonyms. This makes synonym detection necessary for the system. Synonyms for a particular word may be domain independent or domain dependent. Domain independent synonyms are synonyms for a word in the entire vocabulary. But domain-dependent synonyms are synonyms for a word in that respective domain only. There are various algorithms used for the detection and extraction of synonyms, some of which are reviewed below**.** **LinHua Gao et al. [8]** [2018] explains the traditional dictionary method of synonym extractions. In this method, the system database maintains a dataset of synonyms for important keywords in that domain. The sentence sent by the user is then mapped on to that synonym dataset. The keywords detected from the sentence are then checked in that synonym set to check for same intent. All possible synonyms of that keyword are then looked out for a match in the main database. The sentence which is closest to the user sentence is extracted. This method is time consuming and requires more of storage and complexity. **Sijun Qin [9]** [2015] proposed a feature selection method for synonym extraction. In this method, among all the parts of speech tags, words having the tags as noun, verbs and adjectives are marked as positive tags and the others as negative tags. The polarity for each feature (word) is then carried out by using the POS tags. If the overall feature polarity is positive, then it can be identified categorically. All the positive features are then grouped together and the synonyms detection for the group of features will be relatively strong, as an entire clause is checked for its synonymic meaning. The synonym sets which are extracted for that clause of features is then calculated for information gain. The one with the highest information gain is the strongest synonym extracted.

3.3 A review on Decision or ML Engine

Scripted or monotonous chatbots have predefined replies to be given. They provide replies to the user from a set of predefined replies categorized on the basis of the query given by the user. Inclusion of ML in chatbots enables it to compute the replies from scratch. It is used to make predictions to predict the responses for the user queries and also to update the system from its experiences. It keeps updating the databases as and when it encounters something new from the user. This engine uses supervised or unsupervised or both techniques to analyze what the user requires. It further uses a model to interpret the intent of the user and provides the appropriate results. The results may be in the form of predictions or any form of analysis which are based on the execution and analysis of mathematical models. Most of the machine learning models are based on statistical and probabilistic evaluations of the instances occurring and the calculations makes a prediction for the test instance. The decision engine not only includes models for predictions, but also includes algorithms for information retrievals like entity extractions, multiple text classifications, etc. Also, the inclusion of a machine learning layer in a chatbot system, is used to create an ontological relationship for entities extracted, and also associate them with context-specific questions along with their alternatives, synonyms and machine-enabled classes. These features of machine learning, converts a static and basic FAQ system to a smart and more personalized communicating experience. For chatbots that provide services in diverse domains, the machine learning layer adds on to the services that it can provide. It intends to increase the accuracy of the responses provided by the system to the users and also extends the scope of the system. The system is enabled to update itself by learning from its experiences. This makes the system less prone to false predictions. The chatbots that are used in healthcare domain for disease predictions can use a wide range of algorithms, some of which are clustering, Bayesian networks, decision trees, etc. The methods of their execution and the comparison of the algorithms for the appropriate selection of the same is briefed here. A decision engine is the brain of the system. It includes the incorporation of ML algorithms for predictions, statistical and probabilistic calculations, etc. Also, ML enables the system to learn from its past experiences, so as to provide better and revised results. The chatbots for health care domain require disease predictions algorithm. Prediction can be carried out in many ways some of which are reviewed below. Sachin S. Gavankar et al. [10] [2017] proposed the eager decision tree algorithm for prediction. This type of decision tree is the improvised version of the traditional decision tree. It creates this tree at runtime, based on the user"s queries and keeps updating the tree on new user messages. Consider its working for disease prediction. In this algorithm, the symptoms detected in the user query are added as child nodes to the root node. The nodes keep on getting added for new symptoms detected. Further for every symptom, the algorithm checks for the second symptom which has the highest occurrence with the earlier symptom and asks the user for that symptom. If he says yes, then the system traces that path to check for the disease present at the root node. This will keep iterating for all users and the tree keeps getting updated for new entries or traces the path available. **Naganna Chetty et al. [11]** [2015] put forth a fuzzy approach for predictions. In this algorithm, the system follows the clustering mechanism. It means that, the algorithm extracts that data from the knowledge base which is the closest to the user query. When the user fires a query, the algorithm searches for the best matches in the knowledge base and provides the same to the user. In the next iteration, when the user gives the second query, the best matches are further searched for relevance. Each query of the user, filters the matches on every iteration. This keeps on continuing until a single best match is found and that match is provided to the user as the result of prediction. Comparing the two algorithms we come to know that prediction using fuzzy logic (clustering) is easy to implement and involves less complexity. On the other hand, eager decision tree algorithm involves more complexity and requires more time for execution. But the accuracy provided by eager decision trees is more as compared to the fuzzy approach.

3.4 A review on NLG Engine

NLG performs the reverse task of NLU. It is the process of converting the system produced results into natural language representations which can be easily understood by the user. In other words, NLG is the process of generating text/speech from system generated patterns. The results produced by the system are in the structured format so that they can be easily understood and processed by the system. NLG represents the system knowledge base in a natural or conversational language representation which can be easily understood by the user. There can be a number of ways in which in which a same sentence can be said. The sentences can have two voices i.e. active or passive voice. Also, there can be similarity between two sentences, but they might involve the usage of synonyms. Hence, while providing a response to the user, the NLG unit needs to calculate all the possibilities to interpret the same sentence, and then select the most appropriate one. NLG engine also performs a sequence of tasks to generate sentences. The initial task is to determine the content. It involves the selection of response to be given to the user. This step decides the appropriate content (or set of words) that should be present in the sentence. Also, it deals with the position of the words in the sentences based on its POS Tag (placements of verbs, nouns, adjectives, prepositions, etc.). In all, this step deals with the organization of a basic sentence right from the choice of words to their placement in the sentence. The next task is the choice of sentences. As already said, there can be a variety of sentences that can be used to express the same situation, this step deals with the selection of the appropriate sentence, which is the best for that instance. The sentences taken into consideration for possibilities are in their abstract format and are not perfect sentences. They require the addition of grammar rules to make them grammatically correct. This section checks the semantic correctness of sentences based on the grammar rules defined by the system. Last and the most important is the morphology check, wherein the sentence generated from the previous steps is checked upon for its correctness. This step validates the correctness of the sentence.

Summary of Literature Review

Table 1 A Summary of Literature Review

S.N.	Authors	Problem discussed and solved	Method/ Algorithm /Tools Used	Results
1	Mohammed Javed et al. [1],[15]	To implement word segmentation (tokenization)	Calculating all character spaces	It involves mathematical calculations hence proves to be slower than the others.
2	Naeun Lee et al. [2], [17]	To implement word segmentation (tokenization)	Using NLTK package which involves inbuilt tokenizer	Easy to implement, as does not require any coding. Faster and more accurate
3	Tao Jiang et al. [3], [11]	To implement word segmentation (tokenization)	Using Conditional Random Fields	This algorithm proves to be more accurate and less complex than the first but less efficient as compared to NLTK.
4	Jerome R. Bellagarda [4],[10]	To implement POS Tagging	Using the latent analogy algorithm	Requires training of large amount of data. Hence involves complexity.

5	Liner Yang et al. [5], [18]	To implement POS Tagging	Using neural network algorithm	As the algorithm works in layers, it provides high accuracy, but is not time efficient.
6	None	To implement POS Tagging	Using NLTK	Provides above average accuracy at minimum complexity.
7	Bo Chen et al. [6], [1]	To create a dependency parser	Using a dependency tree to understand the dependencies.	Traditional method. Accuracy depends on the training of the data.
8	Zhenghua Li et al. [7], [14]	To create a dependency parser	Using a graph data structure for the implementation of the parser	Improvised version of the above- mentioned algorithm. Provides higher visibility, understandability and improves accuracy.
9	LinHua Gao et al. [8],	Synonym detection and	Dictionary method	Traditional method. Requires to maintain a

CHAPTER -3

Functionality

Language Understanding: NLP-powered chatbots can understand natural language input from users, including their intentions, sentiments, and preferences.

Information Retrieval: Chatbots can use NLP to retrieve information from a knowledge base or a database, and provide users with relevant answers to their queries.

Personalization: NLP-based chatbots can tailor their responses and recommendations based on user preferences and past interactions.

Conversation Management: NLP-powered chatbots can maintain a context of the conversation and respond appropriately based on the context.

Sentiment Analysis: Chatbots can use NLP to analyze the sentiment of the user's input and respond accordingly.

Natural Language Generation: Chatbots can generate natural language responses that are personalized and contextually appropriate using NLP techniques.

Multi-Language Support: NLP-based chatbots can support multiple languages and can be programmed to respond in the language of the user's choice.

Speech Recognition: NLP-powered chatbots can recognize speech input from users and respond with appropriate actions or responses.

Intent Classification: NLP-based chatbots can classify user input into different intents or actions, allowing them to understand the user's request and respond accordingly.

Entity Recognition: Chatbots can use NLP to identify important entities in the user's input, such as names, dates, locations, and other relevant information.

Contextual Understanding: Chatbots can use NLP techniques to understand the context of the conversation and respond appropriately based on the conversation history and user preferences.

Feedback and Learning: NLP-based chatbots can use user feedback to learn and improve their responses, becoming more accurate and effective over time.

Integration with other services: Chatbots can integrate with other services, such as customer relationship management (CRM) software or payment gateways, to provide a seamless user experience.

Conversational UX Design: Chatbots can use NLP to design conversational user interfaces that are intuitive, engaging, and easy to use.

Personal Assistants: NLP-based chatbots can act as personal assistants, helping users manage their schedules, book appointments, order food, and perform other tasks.

Customer Support: Chatbots can provide customer support by answering common questions, resolving simple issues, and escalating complex issues to human agents.

PROJECT DESIGN

Architecture of Chatbot System

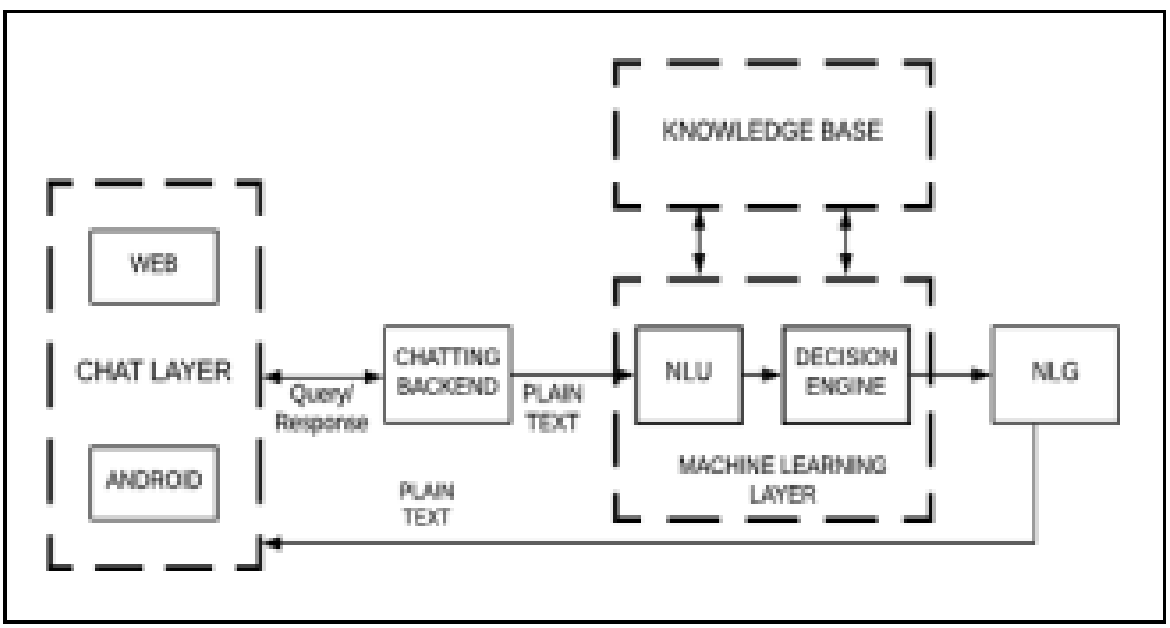

Figure 1 Architecture of Chatbot System

Use Case Diagram:
(User and admin roles)

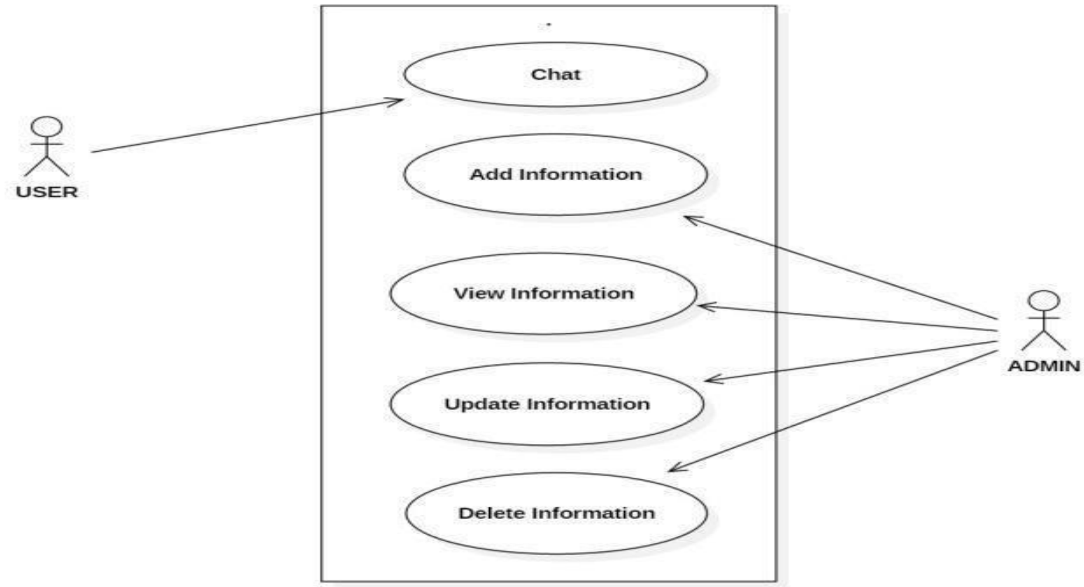

Figure 2 Use Case Diagram of user and admin roles

Data Flow Diagram

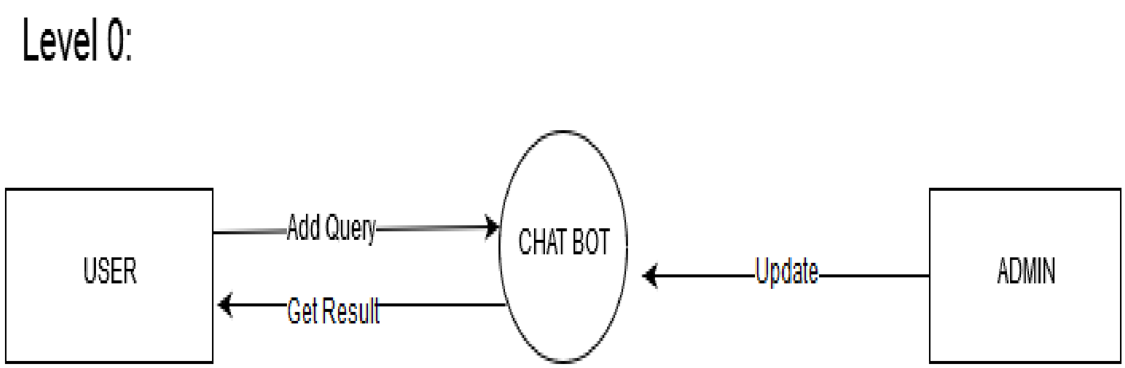

Figure 3 Zero level DFD of Chatbot system

Level 1:

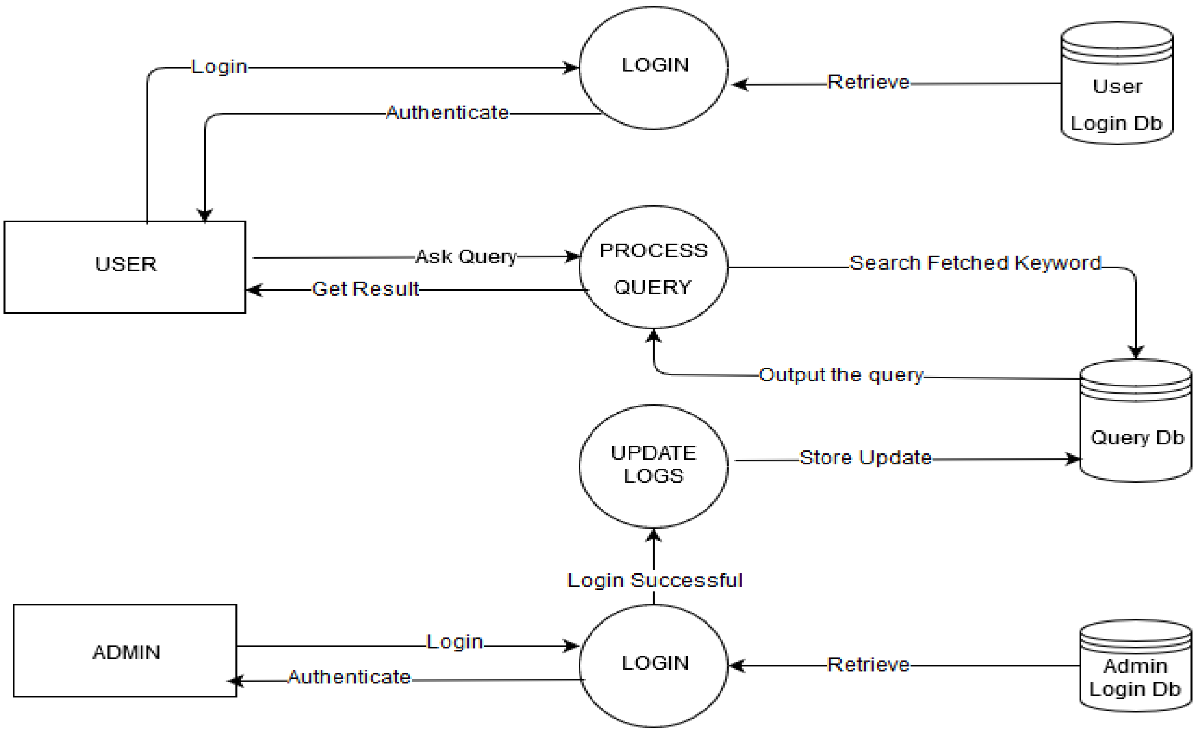

Figure 4 First level DFD of Chatbot system

Use Case Diagram:
(Context identification)

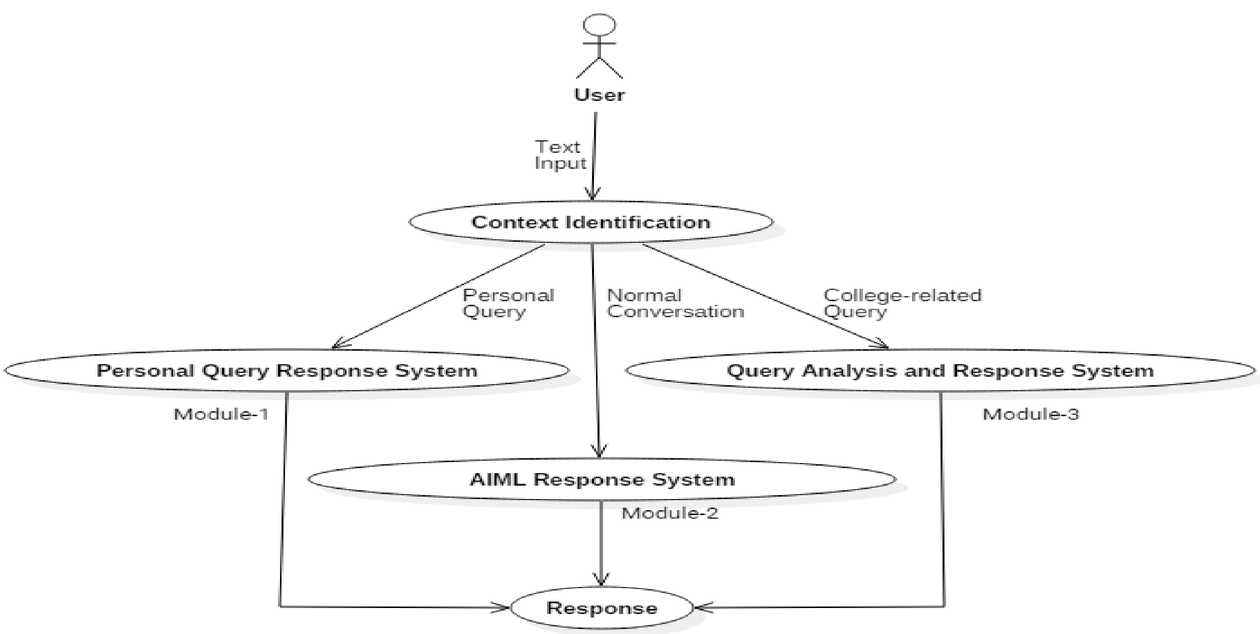

Figure 5 Use Case of context identification

Activity Diagram: Personal Query Response Activity (Module-1):

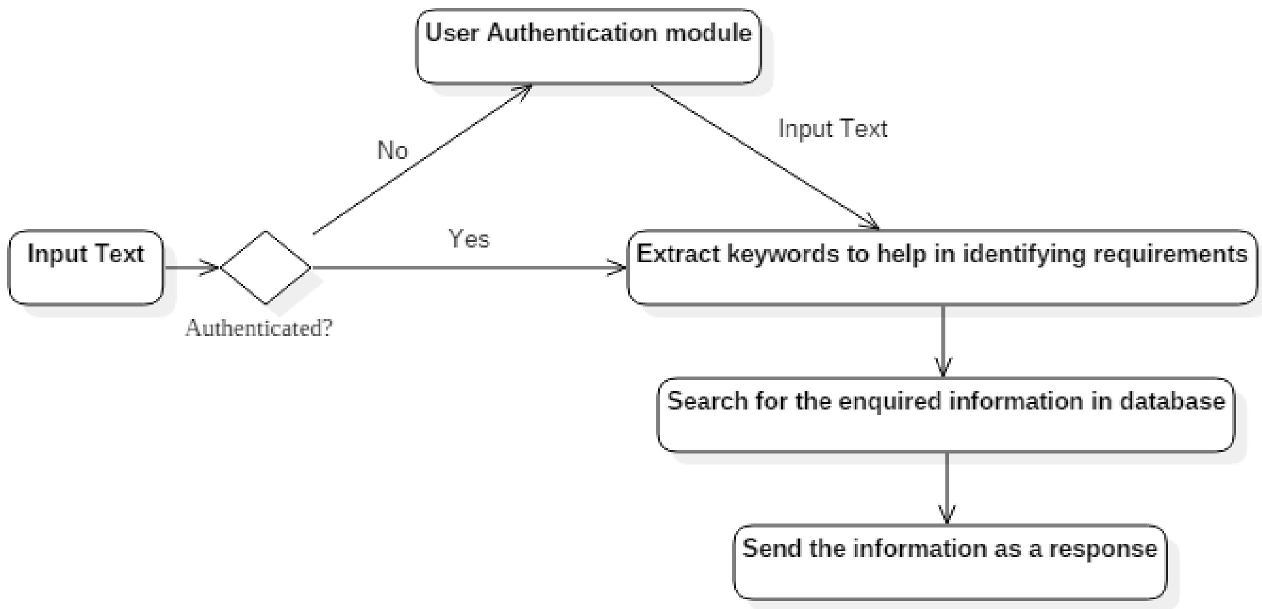

Figure 6 Activity Diagram of Personal Query Response Activity

College Related Query Response Activity

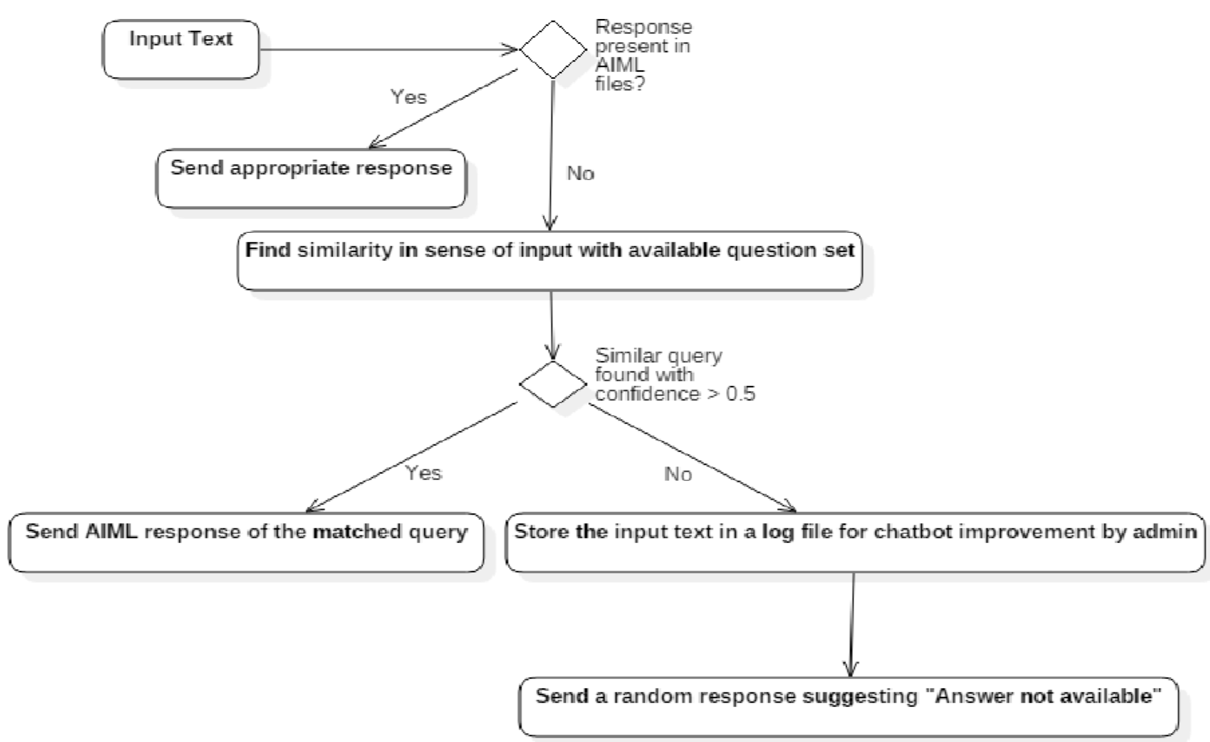

Figure 7 Activity Diagram of College Related Query Response

CHAPTER-4

Result

Intent – Greeting
Entity – welcome (classified as a simple entity)
Utterance – Hi
Hello
Hey etc
Response – Hello, how may I help you?

Discussion

The discussion of a chatbot using NLP in Python will depend on the specific application and use case. However, here are some general points to consider:

- Accuracy: One of the most important factors in evaluating the effectiveness of a chatbot using NLP is its accuracy in understanding user input and providing appropriate responses. You can measure accuracy by comparing the chatbot's responses with the expected output or ground truth.

- Training Data: The accuracy and effectiveness of the chatbot depends heavily on the quality and quantity of training data used to train the NLP model. The more relevant and diverse the training data, the better the chatbot's performance.

- User Experience: A chatbot using NLP should provide a seamless and engaging user experience. This includes using appropriate language, responding quickly, and providing helpful information.

- Integration: The chatbot should integrate well with other software and services, such as messaging platforms, APIs, and databases.

- Maintenance and Improvement: Like any software application, a chatbot using NLP requires ongoing maintenance and improvement. This includes monitoring its performance, updating the training data, and incorporating user feedback.

- Use Cases: Chatbots using NLP can be used in a wide range of applications, including customer support, personal assistants, sales and marketing, and education. The specific use case will determine the required functionality, training data, and integration requirements.

CHAPTER-5

CONCLUSIONS

It is often impossible to get all the data on a single interface without the complications of going through multiple forms and windows. The college chatbot aims to remove this difficulty by providing a common and user-friendly interface to solve queries of college students and teachers. The purpose of a chatbot system is to simulate a human conversation. Its architecture integrates a language model and computational algorithm to emulate information online communication between a human and a computer using natural language. The college student and employees can freely upload their queries. The chatbot provides fast and efficient search for answers to the queries and gets the relevant links to their question. A background research took place, which included an overview of the conversation procedure and tries to find proper link. The database storage includes information about questions, answers, keywords, and logs. We have also developed an interface. The interface developed will have two parts, one for users and the other for the administrator.

Future Scope

The future scope of chatbots is vast and exciting, with ongoing advancements in technology and increasing adoption across industries. Here are some potential future developments for chatbots:

- Natural Language Understanding: Chatbots will continue to improve their ability to understand and interpret natural language input, allowing for more accurate and engaging interactions with users.

- Personalization: Chatbots will become even more personalized, using machine learning algorithms and user data to provide tailored responses and recommendations.

- Multi-Platform Support: Chatbots will be designed to work across multiple platforms, including messaging apps, websites, and social media platforms, providing a seamless user experience.

- Voice Interfaces: With the growing popularity of voice assistants like Siri and Alexa, chatbots will increasingly incorporate voice recognition and response capabilities.

- Emotional Intelligence: Chatbots will be designed to recognize and respond appropriately to emotions expressed by users, providing more empathetic and human-like interactions.

- Integration with AI and IoT: Chatbots will be integrated with other emerging technologies like artificial intelligence and the Internet of Things, allowing for even more intelligent and efficient automation.

- Healthcare Applications: Chatbots will increasingly be used in healthcare, providing support for patients, answering common questions, and assisting with diagnoses.

- Virtual Assistants: Chatbots will become more sophisticated and integrated into daily life, acting as virtual assistants for individuals, helping them manage their schedules, automate routine tasks, and provide helpful information.

- Overall, the future of chatbots is promising, with endless possibilities for improving user engagement, automating routine tasks, and advancing the field of artificial intelligence.

References

[1] Yuhua Li, David McLean, Zuhair A. Bandar, James D. O'Shea, Keeley Crockett, "Sentence Similarity Based on Semantic Nets and Corpus Statistics", IEEE Transactions on Knowledge and Data Engineering, Volume 18 - No. 8, August 2006.

[2] Emanuela Haller, Traian Rebedea, "Designing a Chat-bot that Simulates an Historical Figure", IEEE Conference Publications, July 2013.

[3] Pratik Slave, Vishruta Patil, Vyankatesh Gaikwad, Girish Wadhwa, "College Enquiry Chat Bot", International Journal on Recent and Innovation Trends in Computing and Communication, Volume 5, Issue 3, March 2015.

[4] "AIML Based Voice Enabled Artificial Intelligent Chatterbot", International Journal of u- and e- Service, Science and Technology Volume 8 - No. 2, 2015.

[5] Amey Tiwari, Rahul Talekar, Prof. S. M. Patil, "College Information Chatbot System", International Journal of Engineering Research and General Science, Volume 2, Issue 2, April 2017.

[6] Rachit Kulkarni, Ankit Methwani, Nakul Pawar, Charmi Valecha, Pooja Shetty, "College Chat-bot", International Journal of Advanced Research in Computer Engineering & Technology, Volume 6, Issue 4, April 2017.

[7] Chaitrali S. Kulkarni, Amruta U. Bhavsar, Savita R. Pingale, Prof. Satish S. Kumbhar, "BANK CHATBOT - An Intelligent Assistant System Using NLP and Machine Learning", International Research Journal of Engineering and Technology, Volume 4, Issue 5, May 2017.

[8] Yash Mehta, Shreya Sawkar, "The college chatbot", International Journal of Computer Applications, Volume 173 - No. 7, September 2017.

[9] Prof. K. Bala, Mukesh Kumar, Sayali Hulawale, Sahil Pandita, "Chat-Bot For College Management System Using A.I", International Research Journal of Engineering and Technology, Volume 4, Issue 11, Nov 2017.