A Project Report

on

Text and Voice based Chatbot for Website

carried out as part of the course CS1634 Submitted by

By

Kumar Nilind

189301121

Shubham Maheshwari

189301129

VI-CSE

in partial fulfilment for the award of the degree

of

BACHELOR OF TECHNOLOGY

In

Computer Science & Engineering

MANIPAL UNIVERSITY
JAIPUR

Department of Computer Science & Engineering,

School of Computing and IT,

Manipal University Jaipur,

April 2021

Jaipur, Rajasthan

Abstract:

Chatbot is an AI based or Rule based application used to communicate to a user via a computer program which makes it feel like the user is conversing to a human. Chatbots usually require a huge dataset of queries and responses to work well, with a reasonable amount of data for each type of query.

A small organization or an organization new to this approach might not have this data, our project is focused on these kinds of businesses. This Project uses a pattern matching approach to deal with the lack of data. Most of the data is already built into the Chatbot, any different kind of data like business related queries and other new intents can be added to the program with an excel file in a specified format.

Currently, the Chatbot system is built for MUJ Website for navigation. This Chatbot helps in providing course related details and helping in website navigation using links to the students who visit the website. The bot can reply according to the visitor's query, including Course details, links to webpages, and can collect Contact information in case the visitor wants to contact the administration using Email or phone number. The bot has prebuilt intents to provide basic common functionality to the administrator.

Table of Contents

List of Figures

1. Introduction:

This document aims at defining the overall description of all the functions, and specifications for Text and Voice based Chatbot.

The Chatbot is developed for organizations or businesses which lack the data and resources to build an AI Based Chatbot. The expected audience of this document is the faculty of Manipal University Jaipur (MUJ), including the person in a role of developing the Chatbot, database administrator and students of all disciplines of MUJ.

1.1 Scope of Work:

This Chatbot is designed for small companies, organizations or businesses looking to build a chatbot for their website to reduce the workload of their customer care department and provide a satisfactory experience to the user. The chatbot can handle basic requests like fetching certain information or a webpage required by the user to access certain information. The Chatbot uses pattern matching and deep learning to perform these tasks.

1.2 Product Scenarios:

Customers of those given companies which have chatbot implemented on their website will engage in a conversation with the chatbot to resolve a particular query for which the company has uploaded the data. If there is an unresolved query, then the user will be requested for the required response for the Self-Learning Module, if it is enabled. The Responses will be saved in a database and the admin can then check the queries and is responsible to update the data to solve those unresolved queries or to filter out the garbage useless data that the User may provide, or the business might not require.

2. Requirement Analysis:

2.1 Functional Requirements:

2.1.1 Chatbot Input and Interface:

Chatbot should be able to receive voice or text-based input and should be easy to deploy on a website for which it is developed, without causing too much load on the server for loading a webpage. The Chatbot should also retain its text messages until the connection is dropped or is timed out, change of webpage should not affect the previously given input.

2.1.2 Chatbot Response:

Chatbot should be able to receive a textual or a verbal query, which can then be processed using a pattern matching algorithm and according to the matched intent and the current state of

conversation, the bot will respond to the query with some predefined responses or perform an action based on the query.

2.1.3 Dataset Management:

Chatbot will allow the admin to upload the data in a specific format to formulate queries and responses and set the conversation flow for the Chatbot. The dataset for business specific queries should be available while developing the Chatbot if the organization wants the Chatbot to respond to specific queries according to their business.

2.1.4 Data Collection for Self-Learning:

Chatbot will store all the unresolved and new queries in an excel sheet and present them to the admin, then the admin will have to assign an intent to those queries or reject a response if it is unrelated to the company. This is done because the User might inexplicably give incorrect responses which will also be saved to the table, but these inputs cannot be used to improve the Chatbot and might also cause it to return invalid responses.

2.1.5 Multiple Algorithm Support:

The Chatbot can work with Pattern Matching or using Artificial Neural Networks, for intent classification of the queries. This makes the Chatbot more capable to work with different sizes and different types of datasets available to a business.

2.1.6 Voice based Input and Responses:

Chatbot will be able to communicate to the user using voice-based input and respond back in voice-based output using an API to convert Text to Speech and Speech Recognition.

2.2 Non-functional Requirements:

2.2.1 Compactness:

Chatbot should be small to not hinder the functionality of the website itself. Addition of new data should be retained to enter into the chatbot database if it makes the website work slower.

2.2.2 Security:

Only the admin should be allowed to update the chatbot's data. No one other than admin should be allowed to tamper the data.

2.2.3 Performance:

The response time should be low, and throughput of the system should be high for the customers that have a conversation with the chatbot.

2.3 Use Case Scenarios:

2.3.1 Querying Chatbot and returning a response, Text based:

Customers will ask a query to the chatbot in Textual form and wait for a few milli-seconds for the chatbot to process and respond to take the conversation further. In case the query is out of scope for the Chatbot, users can then assist in improving that query by submitting a response and query to the Chatbot, which will be stored in a database table. If the query is found then the Chatbot will ask further information from the user, if required or not already provided.

2.3.2 Querying Chatbot and returning a response, Voice based:

Customers will ask a query to the chatbot in Vocal form and wait for a few seconds for the chatbot to process and respond to take the conversation further. In case the query is unrecognizable then the User is asked to say the query again. If the Query is out of scope for the Chatbot, users can then assist in improving that query by submitting a response and query to the Chatbot, which will be stored in a database table. If the query is found then the Chatbot will ask further information from the user, if required or not already provided.

2.3.3 Upload Chatbot Data:

Admin will upload the chatbot business specific data from which the queries and responses will be formed and decide the conversation flow for the chatbot, using the helper functions of the module. The data must be in a specific predefined format to be uploaded correctly. The Chatbot is built for dataset of two types, ones that contain conversations or ones the contain tokens or tags, support of various algorithms is made to make the process of uploading data easier.

2.3.4 Update Chatbot Data:

Admin can update the chatbot data when it has been collected. The Admin will work as a filter to remove false or Out of Scope queries to make sure the Bot responds in the correct way.

3. System Design:

3.1 Design Goals:

The Chatbot is designed in such a way that it could handle the following goals:

1. Chatbot should be designed in a way to quickly respond to the customer.
2. Chatbot should be designed in a way to have less memory load on the system/ website.
3. Chatbot Interface should be user-friendly for both Admin and the User.

4. Data Collection should be available for Out-of-Scope user queries to develop a better and more flexible response system.

3.2 System Architecture:

System will be designed for a website in a client-server manner, like a messaging app. Client and server will be connected using WebSockets to speed the process of communication using asynchronous requests and remove the slow and synchronous process of http requests. Asynchronous means anyone can communicate with the other directly, without any delay or requests. Major Processing tasks will be done at backend, only the response will be sent to the Users in an asynchronous manner to the interface.

The backend can perform basic pre-processing of the data using Python Natural Language Processing libraries, and the result is pattern matched using regular expressions or entered into the tensorflow model using the bag of words approach. The patterns can be loaded from a json file which is already implemented to work for basic queries for the user. The patterns found in the database are also checked for synonyms using the Wordnet module from Python's Natural Language Toolkit Library (nltk). Wordnet is used to provide more flexibility in detecting the user query. Since the backend also supports Self-Learning data collection, the Chatbot will be able to save certain interactions from the user in a SQLlite3 Database table.

The Chatbot uses a well-defined conversation flow to maintain a conversation, which is shown in the given figure:

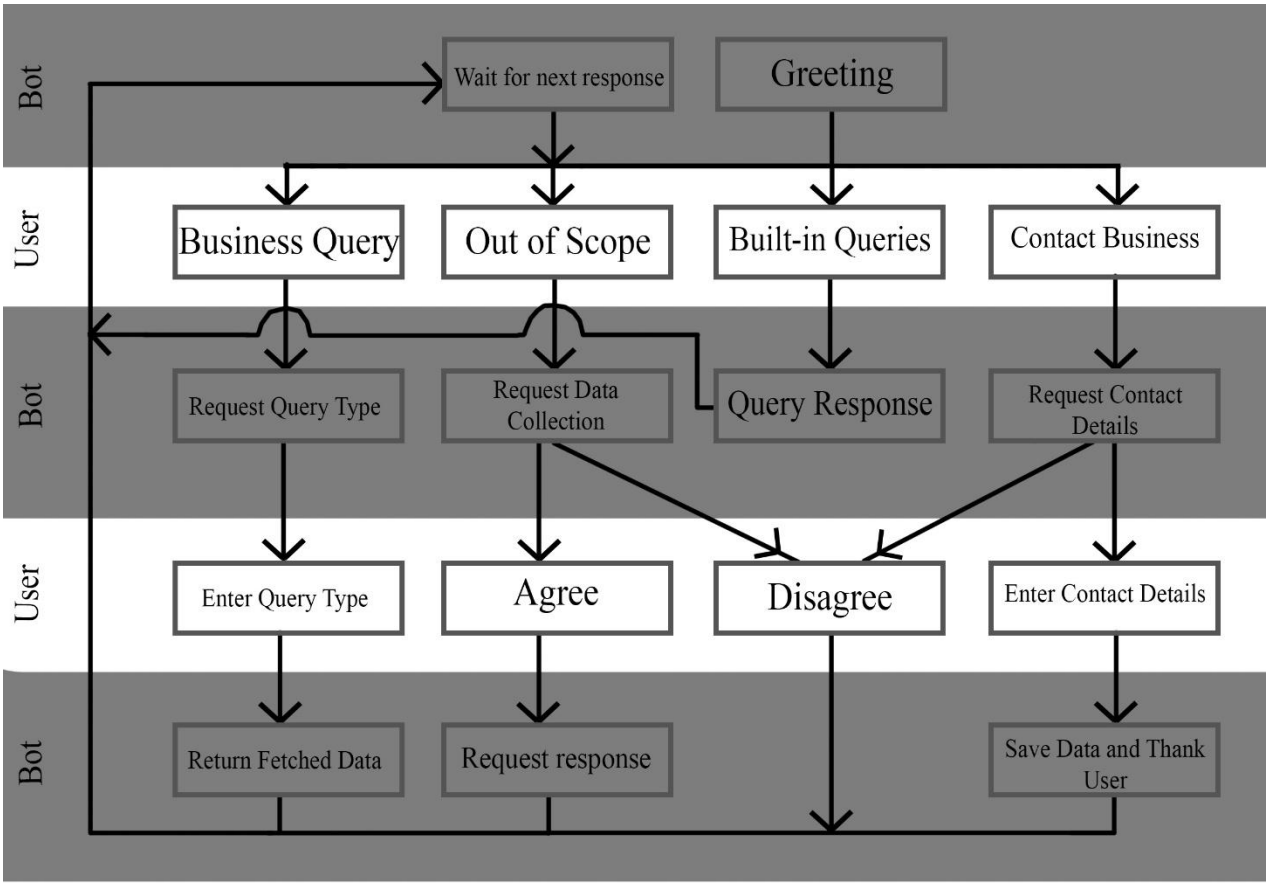

Figure 1: Chatbot Conversational Flow

The Built-in Queries are meant to represent all the queries like, User Greetings, Thanks, Goodbye, or any other type of query that will not have a functionality and is already built in into the system.

3.3 Detailed Design Methodologies:

The Chatbot works as a Module which is demonstrated as if it has been deployed on the MUJ Website. This Module can be programmed to be deployed for any website if we have designed a conversational flow for the Chatbot and has all functions built-in to load dataset and train it. Chatbot recognizes a given intent in the dataset and produces a random output from it. The Dataset is already available in the built-in system, but for every instance of the Chatbot, a copy of this built-in dataset is created. The Chatbot uses two approaches to work:

3.3.1 Approach #1: Pattern Matching:

Pattern Matching Algorithm is used to check a query for certain patterns in each query using Regular Expressions or Natural Language Processing. This approach is especially useful if we want the Chatbot to react to certain specific keywords, or if the dataset only contains keywords instead of Natural Language Sentences. However, it is also designed to handle natural sentences. The approach follows the following steps:

1. Tokenization is done to convert sentences to words.
2. For every token, we remove special symbols and convert it to lower case.
3. Now we remove all the stop words in the tokens.
4. Next to conclude preprocessing of the data, we pass the words through a Lemmatizer to convert similar words in same words and remove any words that are specified as ignore words.
5. Next, we pass the words through Wordnet (A Database of similar words available in NLTK Library in Python) to find similar words for a given token.
6. After similar words are found in each intent,
    a. if the query is a value in the dataset, then we count the number of unique words found and save that value with "VocabSize" key in the json file. Along with the rest of the data.
    b. Else if the query is a response to the user, we iterate through intents given in by the user, and find number of matched patterns of each intent, and divide that number by the "VocabSize" of the given intent. This is done to Normalize the Data. The formula for which can be written as:

$$\text{Normalized Score for an Intent} = \frac{\text{No. of Matched Patterns in an intent}}{\text{Vocabulary size of that intent}}$$

7. The value which is the highest for a given intent is selected as the Intent of the user. The Intent should also be from a given set of Intents provided beforehand, to maintain the conversation flow, if not then the OutOfScope Intent is automatically predicted.
8. For a given intent,
    a. If the intent is a 'generic' intent, then a random response also available in the dataset for that intent is selected. (Generic Intents, refer to things like Greetings, Thanking etc.)
    b. If the intent is "Query" intent, then the pattern matching Algorithm works again for finding out the type of Query it is, in the Query dataset provided by the Business.
    c. If the intent is "OutOfScope" intent and the self-learning system is enabled, then bot asks the user for a possible response which is then saved into a database. If self-learning is not enabled, then this intent is treated as a generic intent.
    d. If the intent is "Contact" intent, then the chatbot asks for the contact information from the User and saves it in another table.
    e. If it is "Agree" or "Disagree" intent then the response is passed on to a verification function which return if the value is positive/negative or OutOfScope. OutOfScope values are treated as the default value provided in the function.
    f. Rest all can be defined as per the conversation flow of the Chatbot.

3.3.2 Approach #2: Neural Networks:

For our Hybrid Approach we have implemented a 3-layer Neural Network with 1 hidden layer, 1 input layer and 1 output layer. These layers work together with the pattern matching system for predicting output of a given user-query.

1. Tokenization is done to convert sentences to words.
2. For every token, we remove special symbols and convert it to lower case.
3. Now we remove all the stop words in the tokens.
4. Next to conclude preprocessing of the data, we pass the words through a Lemmatizer to convert similar words in same words and remove any words that are specified as ignore words.
5. After this,
    a. If model is training, we use the Bag of Words approach which is followed in the given manner:
        i. A list of unique words is created, in which every index is used as a key for that word in the list, this is called a word vector. These labels are then stored in a file for prediction.
        ii. The model is trained after converting each sentence in the word vector form, which is done by initializing an array of zeros with a size of the vocabulary of the dataset and setting the value of all indices to 1 where a word from a given sentence occurs.

    The model uses an Adam optimizer with SoftMax activations and binary cross-entropy loss function, and after training the model is saved for prediction purposes later with the indexes of the classes and their labels.

    b. If the query is for prediction, then the query is converted into a word vector, using the same approach discussed above, and fed into the Neural Network for prediction. The highest value of the prediction is used from a given list of intents is selected and if it is higher than a given threshold value, then that intent is classified. Otherwise, OutOfScope intent is classified as the intent and is handled like in pattern matching approach.

3.3.3 Data Handling:

Data handling is done using Excel sheets, Json files and SQLite3 Databases. We are using JSON files as it uses Hashing to find indexes faster which enables the Chatbot to work faster. The Data is loaded in a JSON file from an Excel File using the following approach:

1. The user provides the directory path of the Excel sheet on their device, the name of the sheet that contains data (whether it be new intents or business queries), and columns which contains the Queries, Intents and Responses/Tokens.
2. We then use the Pandas library to read the excel sheet in the form of a Pandas DataFrame and iterate through rows and save the queries along with their intents in the json file format, which each intent is a key to the dictionary and contains text, responses and other metadata for processing.
3. If the queries are token specific, they are classified using the “Query” intent, that is, if the Query intent is predicted then the system works its way to find the Type of Query. These queries are saved in their own dictionary. The value of which is another dictionary,

containing “QueryNames” and “QueryTypes” keys, QueryTypes are specific subtypes of Queries which can be asked and QueryNames is the name of the Query. These Query are also saved like normal Intents using Excel files.

For example, If the user wants to know about the Eligibility of CSE in B. Tech then the system identifies it as Query intent and the “QueryName” would be CSE and B. Tech, and “QueryType” will be Eligibility.

Data Collection can also be from a User if the self-learning/contact sharing system is enabled. This is done in following steps:

1. The System identifies the “OutOfScope” intent for a given User Query.
2. The System asks the user for a response/contact information.
3. If the Contact Information system is trigged, it checks for mail/phone number in the response and if it is correct.
4. Then the system accesses the Self-learning/Contact Database and saves the user responses.

3.3.4 Voice Interface:

The Voice interface of the Chatbot works as an Encapsulation of the Text based system. It follows the two given rules:

1. If the response is coming the user side in the form of Speech, then it translates it to text and passes it on to the text-based system.
2. If the response is coming from text-based system, in the form of text, then it is converted to speech and passed on to the speakers of user.

4. Work Done:

4.1 Development Environment:

The following development software have been used:

1. Python Version 3.7 is used to develop the Chatbot.
2. The intents are stored in a json file, to produce a dictionary to provide faster results.
3. Database used is SQLlite3 for faster and simple usage as it is already commonly available and easily imported in Python.
4. Flask will be used with the Chatbot using WebSockets to create a user-friendly interface for chatbot responses and user queries.
5. Tensorflow is used for creating the neural network and the Chatbot model.
6. Natural Language Toolkit (nltk) is used for Natural Language Processing of the data for stuff like Tokenization and Lemmatization.
7. Web Speech API has been used for Voice based System.

4.2 Results and Discussion:

This Project develops a generalized framework to produce a working Chatbot, which can be easily customized using built-in methods. The Project can produce output of basic general queries like

Greetings and Thanks on its own. Business specific queries data can be added using the built-in functions of the Chatbot module.
To demonstrate its capabilities, we have developed a basic textual chatbot to operate on Manipal University Jaipur's official website containing information about the courses available and their eligibility and duration for each course.

4.3 Individual contribution of project members:

Work Done by:

- Kumar Nilind
    - Literature Review for the best methodologies
    - Designing the Conversation flow
    - Developing the frontend of the Interface
    - Text based Chatbot (Pattern Matching and ANN Modules)
- Shubham Maheshwari
    - Literature Review for the best methodologies
    - Business facing queries handling
    - Backend interface
    - Voice based extension
    - Admin Interface and Management System

5. Conclusion and Future Work:

Chatbot is still a developing field with much yet to be discovered, therefore we are still applying some of the recently developed approaches of these systems for businesses that are still new to this field. This project is based on Voice and Text input which can be integrated with a website to function as a customer care/ enquiry bot for a business or organization. The Chatbot is aimed specifically at organizations that do not have the data to develop a complex system but would also like to have a system which can reduce the mundane and repetitive task of customer care for basic requests from customers.
In future, this module can be extended to have its own response generation using LSTMs and Natural Language Generation after collection of a good amount of data, which can be used to have an AI based approach, which is faster and more effective in replying to users. The self-learning module can also be extended to receive data from external sources which can also be used to train a Neural Networks to classify the queries of users. Furthermore, the voice recognition system can be made more robust and quicker if the company has access to their own or a faster API which can be used to convert voice to text on the server side itself. Sentiment Analysis can be applied to generate response based on the emotions of the user, which can be helpful in generating good customer satisfaction for the company.