Objectives

Main objective addition: talk about the training material and how it works.

The main objective of the project is to create a fully functional chatbot that is autonomous and non-dependent of human help. The chatbot should be created in Python (a scripting language) with the help of different libraries such as: openpyxl (a Python library to read/write Excel documents), telebot (A Python implementation for the Telegram Bot API). Furthermore, the chatbot will be hosted in Telegram with help of its BOT API framework by using the mentioned ‘telebot’ library. The BOT API is an HTTP-based interface created for developers keen on building bots for Telegram.

The program should use a training system in order to build up knowledge about patterns and to understand when words are used and for what purpose. The training program should be able to train the helpbot by reading large files of text data and extracting relevant information about them. This will help the program to understand English better and it will give it an idea of how the user will talk to it.

The chatbot should have a basic AI that reads the sentences sent by the user and finds patterns or keywords that will guide the program to the correct answer that contains useful information for the user. This AI will find the patterns/keywords by comparing every word to a database of known patterns/keywords (database made in excel and managed by openpyxl), and if successfully found, the AI should select an answer suitable to that specific pattern/keyword and then display it to the user.

Input objectives:

1. The user should be able to introduce a sentence in Telegram asking the program for information. This sentence should then be stored transferred to my python program in form of a JSON file containing multiple information such as the ID of the sender and the message in text form.
2. The message should then be extracted from the JSON file and converted into a string so it can be used further on in the program.
3. If the user enters the command ‘/help’, the program should be able to detect it and output a set of explanations and other commands that could be useful for the user.
4. If the users enter an input that is not a string (eg. An image or emoji), the program should be able to detect it and ask the user to make the request again as those types of input are not supported by the chatbot.

Processing objectives:

1. Create an algorithm that copies the message from a user and saves it into a text file. After this, the algorithm should delete every type of punctuation in the sentence and save all the words line by line in a newly created text file.
2. Create an algorithm that checks every word against the primary and secondary databases. I should then decide if the word is in any of those databases. If it’s not found, then the word should be saved as a new word in the secondary database.
3. If the word is found in one of the databases, another algorithm should update the stats for that single word (the number of appearances, the topic it belongs to and the percentage number of appearances compared to the total).
4. Create a mathematical algorithm that will calculate the relevance of each word based on the number of topics that the word belongs to and the percentage number of appearances compared to the total. The value calculated should then be stored under the relevance section of each word in the primary and secondary database.
5. Create an algorithm which extracts the relevance from each word that the user writes and stored it in the topics’ 2d array depending on the topic it belongs to.
6. Create a procedure which finds the topic with the best relevance within the topics’ 2d array.
7. After the topic is found, the program should select a link from the third database depending on the most relevant topic. This should be done by an algorithm that search through the database for the link that suits the chosen topic, and then extracts it in order to send it to the user.
8. There should be a final procedure that sends the link to the user via Telegram API along with a message telling the user that the request was successful.

Storage objectives:

1. 2 databases should be created they can be used by the program:
   1. - The first database should contain:

1.1.1 - The 10000 most used words of English and saved as a string.

1.1.2 - The number of apparitions in the training data. Saved as an integer

1.1.3 - The percentage number of apparitions of every word compared to the total number of apparitions. Saved as an integer

1.1.4 - The relevance of each word. This should be saved as an integer.

1.1.5 - The number of topics specific to each word. Saved as an Integer.

1.1.6 - The topics that each word appears in, in the training material.

* 1. - The secondary database should contain:

1.2.1 - Every word that is found in the training material but it’s not in the primary database, saved as string.

1.2.2 - The number of apparitions in the training data. Saved as an integer

1.2.3 - The percentage number of apparitions of every word compared to the total number of apparitions. Saved as an integer

1.2.4 - The relevance of each word. This should be saved as an integer.

1.2.5 - The number of topics specific to each word. Saved as an Integer.

* 1. The topics that each word appears in, in the training material.

1. When handling the message from the user, the relevance of each topic should be stored in a 2d array, as an integer.
2. An array should be created to store the letters of the available cell of the database. This array will be used to manage the database and to input and output values from it.
3. The third and last database should be created to store the links that the program is going to be sending to the user by the end of the system. The database should contain:

4.1 - The links that the chatbot should output depending on the topic with most relevance.

* 1. - The name of the topic which the link belongs to.
  2. - This database should only be used to extract information, meaning that nothing within it can be changed or altered.

Output objectives:

1. When the ‘/help’ command is typed by the user, the program should show a list of other commands or a set of explanations in order to help the user use the chatbot. This should be done by outputting a string that has been pre-prepared containing all the information.
2. If the request from the user is successful, the program should output a string to the user, via Telegram’s API, containing a link that will guide the user to IBM’s website containing information of interest to the user.
3. If the user decides to click the link, the program should not stop working in case the user wasn’t satisfied with the link given and wants to make another request to the chatbot.
4. If the user enters a request that yields no results, the program should send a string containing a sentence asking the user to try again due to a failed request.

Other objectives:

1. The training material should be obtained by extracting it from IBM’s products website, and stored in a text file depending on the topic that it belongs to.
2. The program should have a procedure that exists just in case the input from the user is not valid or it doesn’t yield any results.