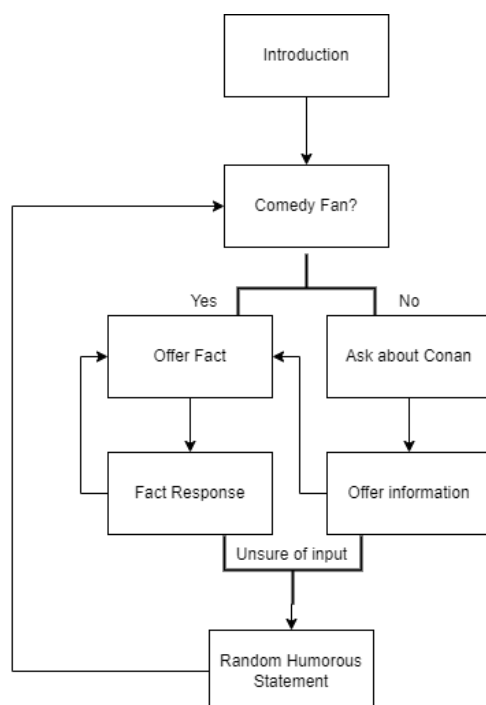


Overview

The system chosen for this project was ChatScript, a open source software that allows for the creation and utilization of chat bots by utilizing a knowledge base defined by the user. The overall topics that were decided on prior to its construction were obtained via project 1 using the NLTK classification library as well as TF/IDF methods. Using the tools provided by other libraries as well, the web scraped data was used to determine which facts should be most important to include in the database used by the chat bot. These facts were used in the knowledge base which the bot draws on and several practical as well as conversational flair rules were added to further emulate an intelligent system with the intention to give the impression of a bot that takes after the personality of Conan O'Brien himself. In accordance with how the software is capable of, different topics were split into different files based on topics and relevant facts as well as rules were grouped with each topic.

Dialog Tree Example



Sample Dialog Interactions

The following is a possible conversation to have with the bot:

If the user is new:

Coco: Hi, I'm Coco, and I'm being forced to spout Conan O'Brien facts! Yay! What's your name?

User: Danias

Coco: Hi there, Danias. Do you like Comedy?

Danias: Yes.

Coco: If you like funny people, you should probably look elsewhere... But if you want to see someone's valiant attempt at comedy, I recommend Conan O'Brien!

If the user is returning using a previously used username:

Coco: Well, well, well, if it isn't Danias again... IF THAT'S EVEN YOUR REAL NAME.

Anyway, here's a Conan fact just for you... Did you know Conan almost broke his hip trying to ride a water buffalo?

Danias: Nope!

Coco: Well, now you're filled with absolutely useless knowledge. Go out and share it with the world!

Knowledge Base Appendix

In the excerpt from a topic file below, explanations for some components of the knowledge base will be provided

topic: ~introductions keep repeat [] **(1)**

t: HI () [Hello] [Hi] [Hey], [talk] [speak] [say something] to me! **(2)**

t: (%input<%userfirstline %input>0) **(3)**

^keep() **(4)**

[Welcome back %user] [Hello, again %user] [Well, well, well, if it isn't %user again... IF THAT'S EVEN YOUR REAL NAME] . Anyway, here's a Conan fact just for you...

t: (%input<%userfirstline)

^keep()

Hi, I'm Coco, and I'm being forced to spout Conan O'Brien facts! Yay! What's your name?

a: (["Conan O'Brien" Conan]) **(5)**

That's a terrible name, but I'm sure you're a fantastic person. Do you like comedy?

b: (~yes) **(6)**

If you like funny people, you should probably look elsewhere... But if you want to see someone's valiant attempt at comedy, I recommend Conan O'Brien!

b: (~no)

Really? Have you ever watched an episode of Saturday Night Live or The Tonight Show?

t: FILLER (*) [Interesting...] [Is that so...] [Right...] (7)

u: (tell me about yourself) ^reuse(WHAT) ^reuse(WHERE) (8)

u: ([who what when where why]) Good question. (9)

From the above:

- (1) Every file will need to have a topic marked with a tilde (~) along with the definitions of certain functions to be used in the file such as “keep” and “repeat”. The brackets indicate what words or phrases will trigger the driver program to scan this particular file to find a best response to a sentence from the user.
- (2) In this statement, having a single word in all uppercase before a rule assigns that rule to the word as a variable. Each sequence of spaced brackets represents a pool of potential word choices the program chooses from each time that rule is called.
- (3) This line can be likened to an if statement and checks whether or not the current user has interacted with the bot previously.
- (4) This function is put in statements that may be used later on, since the default setting is to delete rules once they are used initially. Putting the “keep” key word in the topic declaration will set this as the default for all rules under it.
- (5) Multiple words or phrases within brackets separated by spaces indicate that if any one of them is input by the user, the following rule will then apply. In this instance, if the user

answers that their name is either “Conan” or “Conan O’ Brien”, it will then output the statement below it.

- (6) Subsequences under rules labeled a:, b:, c:, ...q: represent rules which are only executed if the rules above them are also executed, somewhat similar to an if statement.
- (7) In this line, the * character represents a wildcard that can be 0 or more words, which in this instance means if something is said that doesn’t match any other current rule, it will default to this rule and select one of the 3 provided filler phrases.
- (8) This rule responds to inquiries about the bot itself and calls functions to reuse previously defined rules by using the uppercase word assigned to them.
- (9) This rule will activate if any of the words within the brackets are detected, treating the grouping as a singular instance.

Sample User Model Appendix

Within the ChatScript files there is a directory for the different users who accessed the software overall and then profiles for each individual bot with specific settings. Below is one such example of the user most often used in the testing and development of this project:

jul2116 Apr10'22-01:39:03 Apr10'22-04:49:02 1649574266-4.9.0 | - Represents most recent edit date

\$cs_language=ENGLISH – Indicates language used by user when conversin

\$cs_randindex=2517

\$cs_bot=Harry – The name of the prototype bot used in compilation

\$cs_login=Danias – The username chosen by user when starting script

\$cs_token=75726583 – Unique token generated for use by user

\$cs_control_main=~control – Indicates which file was used as driver for selected bot

\$userprompt=danias: > - Indicates what user is referred as when speaking

\$botprompt=HARRY: - Indicates bot name

\$singleresponse=1 – Boolean value to indicate whether the user offered one response or multiple in most recent rally.

\$shutup=1 – Special value that indicates user has told the bot to “shut up” and thus no longer offer output

#`end variables

#context ~introductions 53

The list below indicates different names taken on by the same user across different sessions.

Dan

Ree

Den

Danias

#`end user

Evaluation of Strengths and Weaknesses

While outside parties were not consulted for the gathering of this insight, I as a single project worker saw the entire process from start to finish and from the whole experience, there was plentiful opportunity to apply NLP methods and concepts to this chat bot. As for the specific

software used, ChatScript, there were several upsides and some downsides that were observed during the span of this project. In regards to the former, the bot software is relatively well-documented and tutorials/manuals provide large amounts of information as well as helpful asides such as use cases. The knowledge base was simple enough to develop and the syntaxes of statements and rules were made clear within the manual. However, a factor that became a detriment the more complex rules was the unclear reasoning behind spacing rules as well as variable and function storage. The bot itself is limited by a driver which, while well-documented, can be difficult to imitate when attempting to build a new bot not based on it. The manipulation of files required to begin a new bot as well as a somewhat inconvenient entry point to the program itself was also a con in the overall experience. However, once the initial learning curve is surpassed, the development process is rather straightforward and the option to split topics by file and have them cross-reference one another by default was a major upside that significantly contributed to the ease of access to the bot's development. On the opposite of that, the lack of clear debugging tools or methods did hinder development efforts, but error messages were usually informative and fixable even with limited base knowledge. Overall, this software was an ideal mix of powerful and understandable for a beginner such as myself, and while there are likely simpler or more capable software frames available, ChatScript showed much potential for future and more complex projects.