|  |
| --- |
| **Creating a Chatbot Using NLTK and Chatterbot** |
|  |
| **Cypress Payne** |
|  |
|  |
|  |
|  |

Abstract

|  |
| --- |
| 000  001  002  003  004  005  006  007  008  009  010  011  012  013  014  015  016  017  018  019  020  021  022  023  024  025  026  027  028  029  030  031  032  033  034  035  036  037  038  039  040  041  042  043  044  045  046  047  048  049 |

This document contains the details on how I created Barold the chatbot using python and specifically the NLTK and chatterbot libraries. It reviews chatbots as well as the libraries used before examining how a chatbot can be created and the results from my project.

Introduction –What are Chatbots

Chatbots are conversational systems created using computational linguistics that use natural language processing and machine learning in order to simulate responses to the user’s input. Some chatbots have specific purposes, while others are simply created for conversation. These purposes can include online dialog bots made to complete tasks such as setting up an appointment, or even a psychologist bot made to aid people going through a difficult situation.

Dialog bots include AIs such as Siri and Alexa, who are able to understand a command and then execute it such as playing music or sending a text. Conversational bot examples include Cleverbot and Mitsuku, both online chatbots that people can visit and talk to, allowing them to continue to learn.

Chatbots can be made through two main methods, rule-based and corpus-based methods. Rule-based chatbots, such as ELIZA, a famous psychologist chatbot, use a list of words and phrases that it searches for in the input. When these words are found, regular expressions are used in order to compute a response and output it. Corpus-based chatbots, on the other hand, are created using machine learning and a database full of information that the chatbot learns from and can then pull from during conversations.

Creating a chatbot

In order to create a chatbot, I used the python language along with the NLTK and chatterbot packages. Building a chatbot using machine learning involved two main steps: Creating the bot itself and training it through both corpora and actually talking to it.

|  |
| --- |
| 050  051  052  053  054  055  056  057  058  059  060  061  062  063  064  065  066  067  068  069  070  071  072  073  074  075  076  077  078  079  080  081  082  083  084  085  086  087  088  089  090  091  092  093  094  095  096  097  098  099 |

* 1. Chatterbot

Chatterbot is a python library made to create chatbots easily. It acts as a framework in order to write a chatbot that will work to the users’ expectations. Chatterbot is language independent, meaning it can support a chatbot in any language, and it comes with preprocessors and logic adapters that can be used for the chatbot already coded (About Chatterbot).

Chatterbot is set up to create a very simple chatbot that receives an input, processes the statement using preprocessing and logic adapters, and then outputs a response. The library includes a number of corpora containing prewritten chatbot conversations in order to make training the chatbot faster and easier. These include many different languages and types of conversation such as humor, greetings, and sports.

* 1. NLTK

The NLTK library, or Natural Language Tool Kit, is a python library created to aid with natural language processing. For my chatbot, the toolkit was used to aid in preprocessing the input statement in order to make it easier to read. The NLTK library comes with a tokenizer that helps with dividing sentences and paragraphs into shorter groups such as individual words. This aids in creating phrases that the chatbot can understand better.

* 1. Basic Python Input/Output

For the input and output of my chatbot, a simple while loop was utilized. The while loop receives an input, sends it to the preprocessing function and then gets the response of the preprocessed input to return and output. In each loop, it checks whether the final processed input is “goodbye” and if it is, the bot prints “Goodbye” in return and exits the while loop, ending the conversation with the chatbot.

* 1. Preprocessing with NLTK

Preprocessing for my chatbot involves both the NLTK libraries and the re library, a library which allows for substitutions and replacement of characters and phrases. While chatterbot does provide some preprocessing, I found them to be subpar and decided to instead write my own using the NLTK library which allowed for more freedom and functionality during the preprocessing stages of my chatbot.

Using re, my preprocessing function replaced all line breaks, tabs, and extra spaces with just one space. Once this is completed, the NLTK tokenization library can be used to divide up each word, creating a list of words from the input statement. This list was then assessed using NLTK. All uppercase characters are changed to lowercase and all non-alphabetic characters are removed. Once this list has been processed it is then converted back into a string and returned to the while loop to go through logic adapters.

* 1. Logic Adapters

Chatterbot comes with pre-written logic adapters that can be used on chatbots built with the library. The adapters include a best match adapter, a time logic adapter, a mathematical evaluation adapter, and a specific response adapter. Multiple adapters can also be used and in that case the adapter that produces the highest confidence rating is the response that is returned.

My chatbot uses the best match adapter as it is simply a conversational bot. The best match adapter takes the input statement and assesses the closest previous input using a number called confidence. Based on this confidence it finds the best response from previous training and returns that response.

* 1. Training

As previously discussed, the chatterbot library contains a wide selection of corpora in many different languages that can be used to train chatbots. The English selection alone includes corpora on emotion, food, greetings, health, and humor.

These corpora are made using the structure shown below:

* - input phrase

- output phrase

A provided training program in chatterbot then goes through the corpora specified and trains the bot on them, storing it in a separate file specific to the chatbot.

|  |
| --- |
| 150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181  182  183  184  185  186  187  188  189  190  191  192  193  194  195  196  197  198  199 |

For my chatbot, named Barold, I wrote some of my own corpora, as well as training him using some of the prewritten ones. The corpora I wrote allowed me to give him information about himself, meaning he could answer when asked what his name was and who made him, as well as some personality unique to him.

Besides this corpus he also learned the greetings and emotions corpora which are fairly self-explanatory.

I tried many different corpora to determine which I preferred, beginning by training Barold in all of the English language corpora. This resulted in many answers being related to sports or Alice in Wonderland, so I decided to be more specific and write some of my own in order to shape how I wanted his responses.

Along with corpora training, I also trained him through conversations. Chatterbot is set up so that training can be done by chatting with your bot. When training is being used, the bot will log each input and response in order to learn more responses and become more accurate.

Results

|  |
| --- |
| 100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149 |

Using these methods, I was able to successfully create a chatbot named Barold. The chatbot runs successfully, although slower than I would prefer. Barold does only respond to immediate input; he is a very basic chatbot and does not retain context to questions. However, overall, I see my chatbot as a huge success. He can answer basic questions and provides entertaining conversation, which was my goal with this project.

I also learned a lot about the programming side of tokenization and how artificial intelligence

works with logic and machine learning by using a hands-on approach. I learned the steps involved to create a functional chatbot, from input to preprocessing to logic and training, and finally to a chatbot that outputs responses that make sense in lieu of the input (mostly).

|  |
| --- |
| 250  251  252  253  254  255  256  257  258  259  260  261  262  263  264  265  266  267  268  269  270  271  272  273  274  275  276  277  278  279  280  281  282  283  284  285  286  287  288  289  290  291  292  293  294  295  296  297  298  299 |

References

About ChatterBot. (n.d.). Retrieved June 3, 2019, from <https://chatterbot.readthedocs.io/en/stable/index.html>

How to Clean Text for Machine Learning with Python. (2018, September 10). Retrieved June 3, 2019, from <https://machinelearningmastery.com/clean-text-machine-learning-python/>

Natural Language Toolkit. (n.d.). Retrieved June 3, 2019, from <https://www.nltk.org/>

|  |
| --- |
| 200  201  202  203  204  205  206  207  208  209  210  211  212  213  214  215  216  217  218  219  220  221  222  223  224  225  226  227  228  229  230  231  232  233  234  235  236  237  238  239  240  241  242  243  244  245  246  247  248  249 |