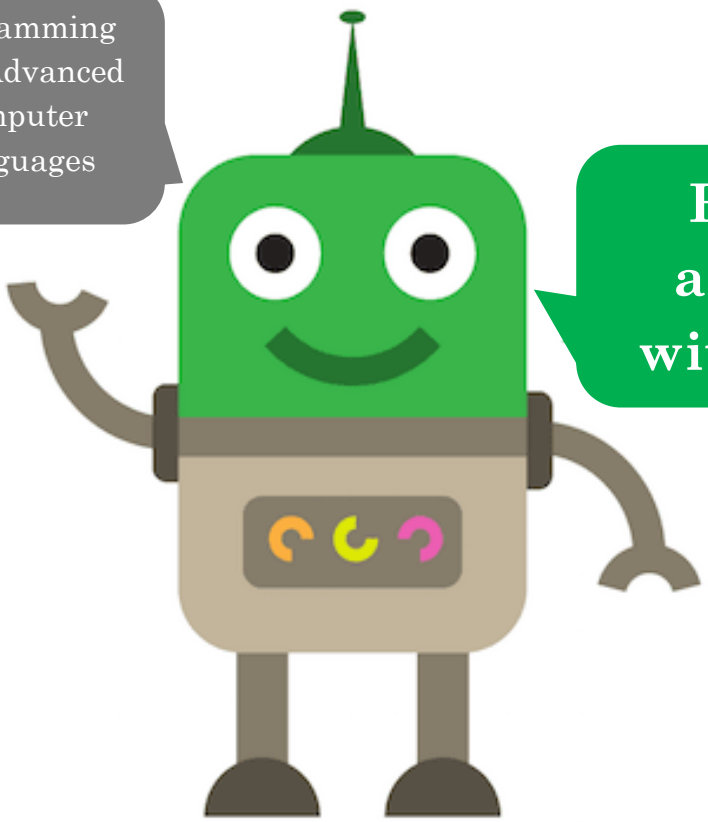
Programming with Advanced Computer Languages



Building a Chatbot with Python

Final Group Project

Human names: Daniel Domingues & Henry Duret

Coder names: CoderD & GandalfComeBack

21.12.2018

University of St.Gallen

Preliminary Remarks

This program was coded by *CoderD* and *GandalfComeBack*. Both coders are beginners and have developed this application with the help of books and the world wide web.

All documents are available on GitHub by clicking here.

- Anaconda Navigator (see here) is used, and especially Jupyter Notebook. On the GitHub link, both Python and Jupyter files are available.
- NLTK (see explanations below) is used and necessary to chat with the bot. Make sure to run the first lines to download it.
- In the code, the path to the text file "lordofthering.txt" is relative. It must be adapted to the user path so that the code runs properly. See line 6:

```
f=open('/Users/Henry/lordofthering.txt','r',errors = 'ignore')
```

- The sample conversations included have been run through Jupyter Notebook.

Introduction about Chatbot

A chatbot is an artificial intelligence-powered software used on a network able to interact with a user. It can be developed to achieve a variety of goals: answer questions about a specific topic, understand expectations or needs, provide recommendations and receive and process given information.

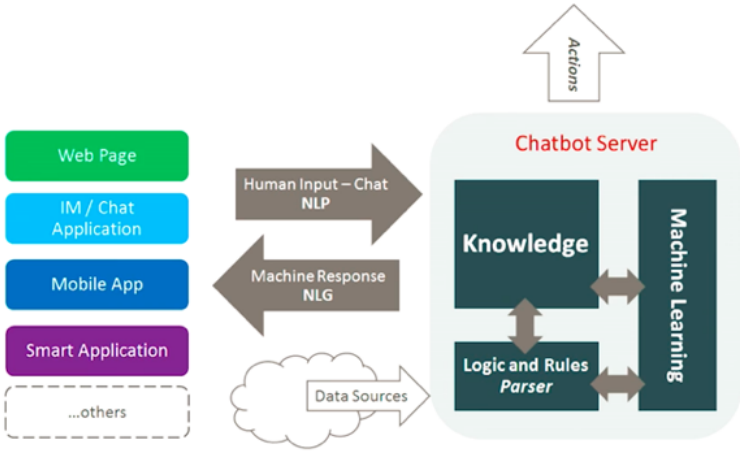
The very first chatbot was called Eliza, created in 1966 by the MIT and simulated a psychotherapist. Today this is much more developed. Chatbots try to measure consumer's needs and then help them to perform a particular task like a commercial transaction, hotel booking, form submission, etc.

Different Types of Chatbots

- Rule-based: the bot answers questions according to some rules upon which it is designed. As a consequence, it can only respond to a specific command. The bot is only as smart as it is programmed to be.
- Self-learning: the bot is based on machine learning. It understands language and not just commands.
    i. Retrieval-based: responses that are predefined or come from a library
    ii. Generative: responses that are generated by the bot thanks to a word-by-word approach

Anatomy of a Chatbot

The following figure highlights the inner-workings of a chatbot. It showcases how human input through the chat leads the bot to use a trio of platforms (knowledge, machine learning and the parser) to analyse the given information. It can then perform further actions outside this ecosystem (such as ordering at a restaurant) and give a machine response to the user.



Platforms

Many messaging apps offer to develop/host chatbots. Among them, the most famous ones are: Facebook Messenger, Slack, Telegram, etc.

But it isn't only limited to messaging apps. Today almost every company has a chatbot deployed to engage with the users, available on their website and/or app to fulfil a specific request. For example:

+ Uber chatbot enables to book a drive
+ Pizza Hut chatbot helps you order a pizza
+ Sephora chatbot provides tips and formulate recommendations
+ KLM runs its own chatbot via WhatsApp to help with customer relations
+ Duolingo helps users to learn a new language in an interactive way
+ …and much more!

The potential is huge as customers are able to access information much more directly by communicating with intelligent chatbots rather than looking through all the website or struggling to find what they are looking for.

Language Library

Introduction

To build a chatbot, an existing library is necessary to avoid typing manually all the answers of the bot and trying to make it cleverer and more comprehensive. To this end, we have used the *Natural Language Toolkit (NLTK).*

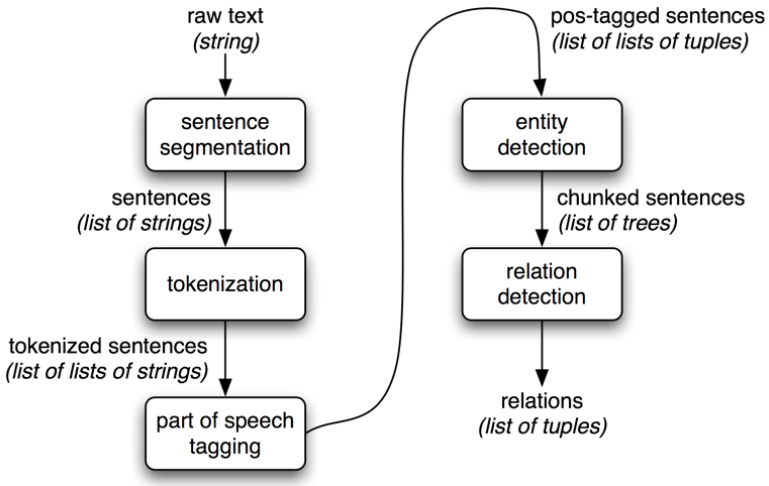
Natural Language Toolkit (NLTK)

NLTK is a platform to build Python programs that work with human language data developed at the University of Pennsylvania. It is suitable for research support, including linguists, cognitive science, artificial intelligence and machine learning. The ease with which one can use the libraries and tools are what set NLTK apart. NLTK's library is described as follows:

> It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries (NLTK, 2018).

Extracting Information from Text

The figure below shows the building design for how NLTK extracts information out of a system. It starts off by looking at a document splitting the text into sentences using a sentence segment. Each of these sentences is then further divided thanks to a tokenizer. Through parsing, the sentences are tagged with "part-of-speech" tags. During the entity detection step, it looks out for mentions of potential entities in each sentence. Finally, it uses relation detection to search for likely relations between different entities in the text.



Parsing

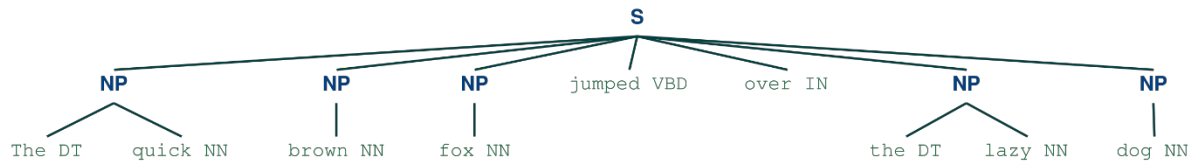
Parsing is used to decode NLP problems such as conversational dialogues and text summarization. There are three different types of parsing: Deep parsing, Constituency parsing and Dependency parsing (Common Lounge, 2018).

Deep parsing looks at the entire syntactic structure of a sentence. It considers which words are the subject and which could be the object. Probabilistic parsing, which is a parsing method within deep parsing, uses grammatical rules to better deduct its conclusion.

Constituency parsing, like the tree above, is used to analyse and specify the components a sentence is composed of. The non-terminals nodes in the tree are the class of phrases and the terminal nodes are the words which are constituents of the phrase (the parse tree below clarifies this graphically).

Dependency parsing assumes that linguistic units, e.g. words, are connected to each other by directed links. The verb is taken to be the centre of the structure. All other words are either directly or indirectly linked to the verb in terms of the directed links, which are called dependencies.

Parse Tree in NLTK



A parse tree looks at the syntax (arrangement of words and phrases to create well-formed sentences in a language) of a string within a context-free grammar. This term is primarily used in computational linguistics. NLTK uses constituency-based and dependency-based parse trees. The former distinguishes between terminal and non-terminal nodes and the latter sees all nodes as terminal.

The above tree-structure reflects the syntactic structure of the following English sentence: *The quick brown fox jumped over the lazy dog.*

+ S: Sentence, the top-level structure
+ NP: Noun Phrase,
+ DT: Determiner, in this case “the”
+ NN: Noun
+ VBD: Verb Phrase, serves as the predicate

Defining our Chatbot

Two critical questions were considered:

+ What is the aim of this chatbot? In other words, what should it be able to answer?
+ Which personality do we want it to have? Which tone, vocabulary, etc.?

The aim of our chatbot

To answer the first question, the current chatbot is able to give different information about the "Lord of the Rings" movies. This information was organised in a .txt file and comes from Wikipedia and fan-blogs.

The goal of our chatbot was to learn about the inner-workings of an "artificial intelligence" program and to better understand how these bots could be used on a wider basis in day-to-day life. At the beginning we set out basic tasks that our bot should be able to complete. These involved:

+ Engaging interactively with the user
+ Being able to answer questions and fun facts regarding the "Lord of the Rings"
+ Print out text and inform the user when it does not understand what is being asked
+ Ending the bot by saying goodbye to the user

Overall, all these functions were implemented. Our bot can answer questions regarding predefined topics which include:

+ Introduction
+ Prologue
+ The Fellowship
+ The Two Towers
+ The Return of the King
+ Information on all protagonists
+ Influences on pop culture
+ Legacy
+ 20 Fun Facts

For every section of code, we have added comments that further explain the role of the coding within the entire program. This should allow future programmers to understand our work and try to replicate it, should they wish to do so.

Personality of the Chatbot

The personality of the chatbot should reflect the author's sense of humour and passion for the "Lord of the Rings" saga.

The general character and attitude of our bot should reflect a fun and interactive atmosphere. The tone with which the chatbot's language was written should be taken with a pinch of salt: we even went so far as to use "Gollum" as the chatbots primary personality. The bot does give a "cheekier" output when the user fails to input the correct data or speech-format. While the vocabulary used throughout the conversations with the bot is ordinary English, we use WordNet, an English dictionary included in NLTK, to help us out.

Reflections on our work

The motivation behind creating our chatbot is manifold. On one side it was the fact that chatbots are increasingly becoming omni-present in today's world. This would allow us to better understand what will be at the forefront of tomorrow's technology. We feel as though within the business student community there is a stigma attached to talking about coding. For some reason or other business students are afraid to engage with technological topics, and if they do, then on a non-technical level. While we are aware that we do not understand much in comparison of everything we could know about coding, we are confident that creating this chatbot has helped us in further gaining some understanding regarding chatbots and how artificial intelligence works.

Another motivation was the fact that both authors had no previous coding experience and we wanted to challenge ourselves. The program is obviously basic, nonetheless we feel as though we were able to implement a lot of learnings and went beyond the course program online.

The last piece of motivation is related to our passion for tech-start-ups. We both want to work in tech in the future and believe that while we do not need to know how to code perfectly, it would be good for us to understand different frameworks within the programming community. A chatbot combined with NLTK has allowed us to dig deeper into the topic and gain valuable knowledge. It helped us think through the design and challenge of creating a chatbot.

Limitations & Suggestions for further improvements

It is important to state that the basis of our work is at a beginner's stage. Advances in terms of capabilities and a further extension of the application of NLTK would be highly

welcomed. Our program can answer pre-defined questions with pre-defined answers. If a user fails to ask the right question, she will be kindly asked to try again ("YOU SHALL NOT PASS. Sméagol wants you to rephrase"). It would further be interesting to explore a program in which the chatbot analyses web searches for Lord of the Rings and would re-direct the user to a website if it did not have the correct answer.

We also see further improvements in combination with NLTK. NLTK does a simple *Named Entity Recognition* which is a part of natural language understanding (NLU). Additionally, it would need a trio of functional NLU, a bot framework SDK and a suitable channel to build a chatbot. Since NLTK is slower for detecting entities, it only detects prebuilt entities based on the model one feeds it. For custom entities and intents, one would need something else. This is where we see development potential.

Happy tinkering!

References

Bird, S., Edward L. and Ewan K., (2009). Natural Language Processing with Python. *O'Reilly Media Inc*.

Commonlounge. (2018). Parsing: Understanding Text Syntax and Structures, Part 3. *Commonlounge*. Retrieved December 2018, from https://www.commonlounge.com/discussion/1ef22e14f82444b090c8368f9444ba78

Daly, L., (2018). An interactive guide to writing bots in Python. *World Writable*. Retrieved December 2018, from https://apps.worldwritable.com/tutorials/chatbot/

Monchalin, E., (2017). An introduction to Chatbots. *Atos*. Retrieved December 2018, from https://atos.net/en/blog/an-introduction-to-chatbots

Natural Language Toolkit (2018). NLTK Documentation. *NLTK Project*. Retrieved December 2018, from https://www.nltk.org/

Panchal, J., (2018). Making a simple Chatbot. *GitHub*. Retrieved December 2018, from https://github.com/jigna-panchal/Chatbot/blob/37cb41622699b8ada1085b3284085abcfe3b40be/.ipynb_checkpoints/Chatbot-checkpoint.ipynb

Pandey P., (2017). Building a Simple Chatbot from Scratch in Python (using NLTK). *Medium*. Retrieved December 2018, from https://medium.com/analytics-vidhya/building-a-simple-chatbot-in-python-using-nltk-7c8c8215ac6e

Payal (2017). An Introduction to Chatbots. Medium. Retrieved December 2018, from https://medium.com/kontikilabs/an-introduction-to-chatbots-34a6a123796a

Schlicht, M. (2016). The Complete Beginner's Guide To Chatbots. *Medium*. Retrieved December 2018, from https://chatbotsmagazine.com/the-complete-beginner-s-guide-to-chatbots-8280b7b906ca

Withey, D., (2017). What is a Chatbot? An Introduction to the Latest Trend. *Ubisend*. Retrieved December 2018, from https://blog.ubisend.com/discover-chatbots/what-is-a-chatbot-introduction