URCSN | University of Regina Computer Science Navigator

Sameer Singh

August 2022

Contents

1 Abstract

URCSN (University of Regina Computer Science Navigator) is an unofficial chatbot for the University of Regina that provides information about the University's Computer Science department. Students often get confused while choosing the right program for them. This is where URCSN comes in. Prospective future students can use URCSN to browse through the current CS programs offered at the University of Regina and decide which program to apply for. Current students will be able to use the chatbot to select their courses for registration in the upcoming terms.

2 Introduction

2.1 NLTK

NLTK (Natural Language ToolKit) is a collection of libraries and modules for NLP (Natural Language Processing) written in the Python language. The punkt module in the NLTK library is used to handle the tokenization of the user input (section 5.1).

2.2 Pytorch

The machine learning framework used in URCSN is Pytorch. It uses the "nn" module to create the neural network for the application. Although TensorFlow is a more popular choice, Pytorch was preferred due to some bugs/incompatibility issues with TensorFlow.

2.3 Feed Forward Neural Network

A Feed Forward Neural Network is used by URCSN for the classification the user inputs. It consists of an input layer which takes in the user queries and passes it on to the subsequent layers. Then, linear transformation is applied to the user input via 2 hidden layers using the "nn.Linear" function and the processed data is passed to the output layer. The output layer uses "nn.ReLu" activation function to remove all the negative elements and replace it with zeroes. The non-negative elements are not affected. At the end, the Feed Forward Neural Network calculates the probability of the the user input belonging to each tag, and classifies it accordingly. Once the query is classified, the response is generated from the json file.

2.4 Tkinter

The GUI of the chatbot is built using the package Tkinter. It is a standard package that is available on most of the operating systems based on Unix. It handles the various GUI elements and components of the app such as the "send" button. It is analogous to CSS for web applications.

3 Related work

The goal of this project is to apply my knowledge gained from my NLP class as well as the independent research I did for web scraping, chatbots, etc. Since I am new to AI development, time and lack of in-depth experience are the most limiting factors. The model used is based on George Kassabgi's TensorFlow framework (1).

3.1 installation

```
To run URCSN, follow the steps listed below:
1. Install NLTK via pip using "pip install nltk"

2. Download the "punkt" tokenizer using
nltk.download('punkt')

3. Install Pytorch by using the command
pip3 install torch torchvision torchaudio

4. Finally, install Tkinter using pip install tk
```

3.2 Usage

For the first run, the model needs to be trained on the json data provided in intents.json file. Run the command "python train.py" to create a data.pth file which will be used by Pytorch library. URCSN can then be used as a GUI by using the command "python app.py". If an error is encountered, please make sure the latest versions of the dependencies listed in section 3.1 are installed.

4 Novelty

The chatbot uses the JSON data provided in intents.json to classify the user inputs (patterns) into one of the configured tags. In the beta release, each response is hard-coded but in the future releases, it will be replaced by web scraping (see section 7.1). URCSN is based on a publicly available model that uses a feed forward neural network consisting of 2 hidden layers. Initially, the chatbot only ran in the terminal window. Using Tkinter, it was implemented to run as an interactive GUI application.

URCSN is based on some core NLP concepts like Tokenization, Stemming, and Bag Of Words (BOW) document. It uses the Natural Language ToolKit (NLTK) library, Pytorch and Tkinter to create the user friendly GUI for the chatbot, as described in the last section. The model is implemented with a feed-forward Neural Net with 2 hidden layers

5 Method

5.1 Tokenization

Tokenization in NLP refers to the process of breaking down the user input, such as a string, into discrete units. These units may be individual words, letters, or n-gram characters. In URCSN's model, each individual word is chosen as the discrete unit. Using tokenization, the words are fed into an array called "all-words".

5.2 Stemming

The root form of the words in this array is generated by stemming, a process where the ending of each word is truncated. For example, if the same stemming function was to be applied to the 3 words "organize", "organizes", and "organizing", the stemmed word would be "organ". After stemming is done, the words are turned into lowercase and punctuation marks are ignored.

5.3 Bag of Words

Each word in the "all-words" array is cross referenced with the patterns defined in intents.json file. If the word exists in a defined pattern, the corresponding element in the "bag-of-words" array is initialized as 1, and 0 otherwise. This is done for each word from the user input, giving us the "bag-of-words" array with 1s and 0s.

6 Future Updates

6.1 Web Scraping

Web scraping is the process of extracting data from a given website. Currently, URCSN uses the hard-coded data that I collected from the University website. In the next update, URCSN will use regular expressions (regex) and web scraping from the University website to generate responses, instead of having a preconfigured json file.

6.2 Discord integration

In recent times especially during the pandemic, many VoIP (Voice over IP) apps have become popular among students for communication. The CSSS (Computer Science Students' Society) has a discord server which has been really helpful for the students who started their University program during the lockdown. URCSN is currently a simple GUI application, but in the next release, it will also be integrated with discord as a bot that can be accessed with certain commands in text channels.

7 References

(1) Kassabgi, G. (2017, September 9). Contextual Chatbots with tensorflow. Medium. Retrieved from https://chatbotsmagazine.com/contextual-chat-bots-with-tensorflow-4391749d0077

(2) Python-Engineer. (2020, September 4). Python-engineer/pytorch-chatbot: Simple chatbot implementation with pytorch. GitHub. Retrieved from https://github.com/python-engineer/pytorch-chatbot