Homework #4

Deadline: Wednesday, 13 March 2019, 23:59

Introduction

In this homework, we will build a simple chatbot. The chatbot will serve as a simple public transport information system and can aid the user in travel planning with public transport. For example, the user can ask questions on travel options for specific departure/arrival times or different destinations. The chatbot should be able to answer travel-related questions, remember information and ask for more information from the user when needed. For some examples or inspiration, you can take a look at the paper on the OVIS by [**?**].

First, write a knowledge base (KB) with travel information which the bot can use to advise on transport in python. The KB should include at least some examples for trains traveling between different locations at different times. You are permitted to work together with other groups to create larger KBs or find other creative ways to create a larger KB.

Next, we will write a chatbot in AIML (using the python package `python-aiml`). AIML is an XML dialect for processing natural language. The A.L.I.C.E (Artificial Linguistic Internet Computer Entity) chatbot was the first to use the AIML language and interpreter. In AIML, you can match language patterns to response templates. An example is shown below:

```
<category>
<pattern>HELLO</pattern>
<template>Hi there!</template>
<category>
```

With this code, when a user says '*hello*', the chatbot will respond with '*Hi there!*'. For some tutorials and inspiration on writing a chatbot in AIML, you can look here.

The KB and AIML can be loaded into the Jupyter notebook `lab4.ipynb`. This notebook contains a simple loop to keep the chatbot running. Currently, example files `basic_chat.aiml` and `complex_processes.aiml` are used by loading `simple-start.xml`. Make sure these files are in the same folder as your notebook. Adjust and/or add files to load your own chatbot.

Assignment

Write a chatbot which aids the user in travel planning. The chatbot should be able to answer the following questions/statements (at least):

- *'What trains are leaving at* `<TIME>`*?'*
- *'What trains are arriving at* `<TIME>`*?'*
- *'I want to go to* `<LOCATION>`*'.*
- *'I want to go to* `<LOCATION>` *at* `<TIME>`*'.*
- *'I want to arrive at* `<LOCATION>`*'.*
- *'I want to arrive at* `<LOCATION>` *at* `TIME`*'.*
- *'What is the last train to* `<LOCATION>`*?'.*
- *'What is the first train to* `<LOCATION>`*?'.*

The chatbot should be able to handle these questions/statements for different variables and ask for more information when needed. For example, for questions such as *'What trains are leaving at 10:30 AM?'*, the chatbot should know what the (departure) location of the user is. The chatbot should also be able to handle completely new requests for different travel information in the same session (e.g., when the user wants to find out the returning route).

Create a KB of travel information. Create an AIML file which handles the language patterns. Adjust the code in the Jupyter notebook `lab4.ipynb` so that your correct files are loaded.

Give conversation examples with your bot in a README file which illustrates it can answer the questions/statements mentioned above.[1]

Extension to two weeks

If you do this exercise as a two week (extended) exercise, we expect a larger set of questions that can be answered by the system. In addition,**you have to add some prosodic information usage**. However, you do not have to actually extract prosodic information from speech. You can make the chatbot handle examples in which adding prosodic information can lead to different answers and then hardcode this information into the text input (for example, by adding '—' for boundaries or 'F' for focus) as if it was extracted from speech input.[2] Make sure to add examples in the README file in which the different outcomes are illustrated.

Grading

You can do this exercise as a one week exercise (and spend this week working on the previous exercise), or a two week exercise (in which case you started earlier and the grade counts twice). The deadline stays the same, but if you spend one week on this exercise and two weeks on the previous, the deadline for the assignment of session 3 is one week later and the grade for that session counts twice.

[1] We will also test out scenarios ourselves

[2] The guest lectures, e.g. by Julian Schlöder, might give you some inspiration/ideas.

Grading will consider:

- Does the notebook compile without any errors? (Grade 1 will be given when one of the requested files are missing or if there is no README file.)
- Is the chatbot able to handle the questions given in this homework? (Essential for a passing grade.)
- A higher grade can be achieved by creating different scenarios which the chatbot can handle, adding more information or more complex travel planning which the chatbot can return, as well as adding usage of prosodic information. These steps should be documented in the README file.

The modified Jupyter notebook, together with your AIML file(s), KB file and README file, should be submitted in a compressed folder in Canvas. The deadline is Wednesday, 13 March 2019 at 23:59.