Project specification

Data-driven learning of the meaning of route descriptions

Kristoffer Hallqvist (khallq@kth.se)
Dmitrij Lioubartsev (dmitrijl@kth.se)

Background and problem description

We often use computers to aid us when it comes to finding out appropriate routes to a given destination. However, when considering the human-computer interaction aspect, we run into a few problems. A route description in a human-spoken language can be expressed in many different ways, and also be ambiguous, thus making it hard for a computer to understand. In this project we want to explore ways of make the interaction easier, by developing a computer program that, by analyzing route descriptions, can learn to recognize their nature.

Goal

We want to see what the possibilities are for a computer to learn to understand route descriptions in a natural language. We will limit our world we work with, and have our program being able to learn words with certain meanings from any natural language, that we take only a few assumptions about.

Project plan and assumptions

We have a certain amount of collected data that make up different route descriptions. These are recordings from different people describing a route to the computer while manually tracing the path themselves with the mouse pointer on a picture of a map. One route description contains information in text form about when which words are said, and where the mouse pointer is positioned at any given time during the recording.

We will start by pre-defining a set of events (mouse movement patterns). These events will be given identifiers, which can be thought of as the "internal language" of the program. A very simple example would be an internal language of {"south", "north", "west", "east"} with corresponding events. For example, we could define that moving the mouse pointer predominantly in an upwards direction, regardless of location, is one common event that would correspond to "north" in this simple 4-word language. Note that the what we call the events is not important, because internally the program would think of it as a specific mouse movement, and not be bound to the word "north" from the English language (this is an important distinction).

Even in the simplest of worlds, knowing the directional words is not enough to describe a route, you need to know how far to go in that direction as well. For this, we would use the landmarks from the map. So we need to expand our internal language to allow for learning words for landmarks. So the simplest language in which you could describe a route would be {south, north, west, east, {landmarks} }. Example route description from landmark "Train Station" to "Church":
"*Train Station north Crossroad west McDonald's north Church***"**

Other relevant language words to consider are prepositional words, because "left of the Crossroad" is not the same as "right of the Crossroad", but for simplicity reasons we will define that being at a specific landmark is the same as being within a set distance from it (we will have to define this distance).

So now we have defined a very simple internal language in which we can describe simple routes. Now, it is important to distinguish the terms "*internal language***"** and "*spoken language***"**. The

internal language is what was just described in the above paragraphs. The spoken language is the language a person is communicating with the program in. We do the following assumptions about the spoken language:

- It does have at least one word that correspond to each of the words/events in the internal language.
- The words in the spoken language that correspond to the words/events in the internal language are all distinguished from each other and have no other definition in the spoken language.
- The words in the spoken language that correspond to the words/events in the internal language are not written as multiple words, but in fact are written as one word.

The next step is therefore to find mappings between the words in the internal language and the spoken language, by processing the route description data. Because of the assumptions above, the mappings “internal language”:“spoken language” can be 1:n mappings, but not n:1 mappings. How will we achieve this? In short, the program will try to identify the events from the internal language in the mouse movement sequence, and associate these with words said at approximately the same time.

There are multiple ways do these associations, or mappings, but our current plan is to use the *tf-idf* technique. It is used in information retrieval to find words that appear in some parts of the text, but not in others. We are not sure how well this will work, but we have good hopes. We also have an idea of having the program try different words. For example: we want to go from landmark “Train Station” to “Church” (see above example). We have used the tf-idf algorithm to map “left” in the spoken language to “west” in our internal language, but possible matches for “north” are “up” and “coin”, according to tf-idf. So we can try to map “coin” to “north”, and follow the directions from the spoken language, and see if we actually end up in “Church”. If we don’t, we can scrap the word “coin” from possible matches. However, we have to be careful, because there are some possible problems with this: other mappings can be wrong, or multiple mappings can be “uncertain” at the same time.

If we have time, there are multiple ways to expand the project:

1. We create a small program that, after creating the mappings of “internal language”:”spoken language”, will be able to take a route description in the spoken language and draw out the route on the map, or perhaps generate a route description given the route.
2. Expand our internal language to also handle prepositional words.
3. We could move away from the pre-defined events, and have the computer recognize patterns instead. This is most likely very hard though.

Preliminary time plan

This is a preliminary time plan. While we have the general idea of how to achieve our goal, practical details are still unclear. The first deadline, about program design, will allow us to make a more detailed plan. It is also very important that the planning is done properly, so we might need more time on that deadline as well.

18-2-2013 Basing program plan, design and code skeleton
25-2-2013 First version of code for predefined events
5-3-2013 Half-way meeting
25-3-2013 A first working program that suits our goal
4-4-2013 Testing and further improvements done
12-4-2013 Essay deadline

Relevant literature

Learning to Follow Navigational Route Instructions - Nobuyuki Shimizu & Andrew Haas:
http://ijcai.org/papers09/Papers/IJCAI09-249.pdf
Quote from the abstract: **"***We have developed a simulation model that accepts instructions in unconstrained natural language, and then guides a robot to the correct destination."*. This seems relevant to our project.

A Testbed for Examining the Timing of Feedback using a Map Task - Gabriel Skantze:
http://www.speech.kth.se/prod/publications/files/3761.pdf
This is the experiment that gave us the test data to work with, which might come in handy when learning about the maps and gives some background to our project.

Learning to Follow Navigational Directions - Adam Vogel & Dan Jurafsky:
http://nlp.stanford.edu/pubs/spatial-acl2010.pdf
This seems very similar to our project. It also has a section of "related work", which might be useful.

C. D. Manning, P. Raghavan and H. Schütze, Introduction to Information Retrieval, Cambridge University Press, 2008 (Course literature for the KTH course *DD2476: Search Engines and Information Retrieval Systems*):
http://nlp.stanford.edu/IR-book/information-retrieval-book.html
This book contains information about information retrieval in general, but the parts about ranked retrieval is what particularly interests us. One of the techniques described here is tf-idf - something we are likely to utilize in this project.