James Hargreaves
Gonville and Caius
jh2045

Diploma in Computer Science Project Proposal

Exploration of Spoken Programming Languages

January 24, 2019

Project Originator: James Hargreaves

Project Supervisor: Paula Buttery

Signature:

Director of Studies: Graham Titmus

Signature:

Overseers: Robert Mullins and Pietro Lio

Signatures:

Introduction and Description of the Work

Currently dictation can be, and is, used by people to take notes and write essays. However, it does not work very well for writing code. The limited software that does exist for this task forces the user to speak in an unnatural way. For example, the user has to specify punctuation characters in the code, such as brackets and commas. When we are reading out regular text we do not directly specify the punctuation that is written in front of us, but instead it is inferred by the listener. Why should dictating a computer program be any different?

In this project I will create an application that takes a spoken language transcript and maps it to pseudocode. The content of the spoken language transcripts will be that of a programmer explaining a function in the most natural way to them. It will function as if one person was explaining to another person how to code up the function in a pair programming session.

Resources Required

I plan to use my own computer, (2.9 GHz CPU, 16 GB RAM, 1TB Flash Storage, macOS Sierra). I will regularly push my changes in the project folder to github - such that I can recover from failure or loss of my local machine. I accept full responsibility for this machine and I have made contingency plans to protect myself against hardware and/or software failure.

Starting Point

I have already undertaken a pilot study to explore methods for collecting an appropriate training corpus. I have briefly looked at current solutions such as https://voicecode.io/. I did this to determine firstly if they solve the same problem and secondly if I could learn anything from them.

Work to be Done

1. Collect Corpus - This will be done by asking experienced programmers to explain a common computer algorithm such as a quicksort function and recording what exactly they are saying. We will then transcribe this for ease of processing by the system (but we will retain speech recording of the subjects for use in a possible extension—see below). An Ethics approval form will be required which will be handed in asap.

2. Write code for baseline system

    - First, we will tokenise the transcript files to produce a stream of tokens, as would be done in a standard front end compiler. This will be required for us to be able to handle variable names etc. since from a functional point of you, a variable name has no semantic meaning other than as an identifier.
    - Secondly, we will re-order the tokens to the most probable order based on an n-gram model of the pseudocode (which will be trained on labelled examples).

3. Training a statistical machine translation system. This will involve training a phrase aligner to add into the baseline system. The tokeniser from the previous part will once again be used for handling variable names.

4. Creating mapping for syntax trees to abstract syntax trees(AST), this will involve parsing the transcripts with a natural language parser and writing a very simple chart parser for the pseudocode. We will then define a mapping from the Natural Language Syntax Tree to AST in the pseudocode language. Finally, we will walk down the tree to generate the pseudocode text.

5. Evaluate your systems using Minimum Edit Distance (i.e. write code to count the number of insertions deletions and substitutions in the system output compared to the desired pseudocode)

Success Criterion

A successful implementation of this project will produce:

- A parallel corpus containing spoken descriptions of algorithms and their associated pseudocode.
- Three systems for translating between the descriptions and the pseudocode (one using a baseline bag-of-words method, one using a statical machine translation method, and one using a rule-based translation method). Each system should produce valid pseudocode tokens only.
- An implementation of the minimal edit distance algorithm for evaluating the translation systems.

Possible Extensions

A possible extension is to include the metadata of the speech such as: pauses length between words and intonation. If time, we will include this in the input to the system to see if this increases the accuracy of the translation.

Timetable and Milestones

Weeks 1 to 2 - Starting 22/10/18

Evaluation of the currently available code dictation software especially any areas in which they could be improved.

Milestones: A summary of the good/bad feature of the current solutions.

Weeks 3 and 4 - Starting 05/11/18

Confirming exactly how the data collection will be carried out, including the prompts which will be presented to the respondents.

Milestones: Have a plan for exactly how the data collection will be undertaken including any prompts. Will also need to submit the Ethics form.

Weeks 5 to 6 - Starting 19/11/18

Running the surveys to gather data for the project and turning the spoken words into typed transcripts. As well as, finding and familiarising myself with the libraries that will be required to process the data as well as defining the pseudocode language that I will use for the output.

Milestones: Have the data which the system will be trained / evaluated on prepared for input into the system.

Weeks 7 to 8 - Starting 03/12/18

This project will likely include some NLP / stats libraries which I wont have come across before. Choosing and familiarising myself with the libraries that will be used int he writing of modules.

Milestones: Have a list of libraries to use and some working examples of relevant concepts using these libraries.

Weeks 9 to 10 - Starting 17/12/18

Write the Traditional Machine Translation based module.

Milestones: Have a working Traditional Machine Translation Module.

Weeks 11 to 12 - Starting 21/01/18

Write the Statistical Machine Translation based module written.

Milestones: Have a working the Statistical Machine Translation Module.

Weeks 13 to 14 - Starting 04/02/19

This period will be used to continue work on either of the 2 previous modules if they need more time. Alternatively, if these module have been completed, see extensions section.

Milestones: Have both the Statistical and Traditional machine translation complete and ready to test / evaluate.

Weeks 15 to 16 - Starting 18/02/19

Evaluating the system. This will include building the baseline module as specified in the success criteria section.

Milestones: Built baseline module, evaluated both modules against the baseline and each other.

Weeks 17 to 18 - Starting 04/03/19

Writing the dissertation.

Milestones: Have the initial chapters of the dissertation completed.

Weeks 19 to 20 - Starting 18/03/19

Continue writing the dissertation.

Milestone: have the dissertation first draft completed and handed in to supervisor.