Siddhartha Edara and Bharat Venkat

Computer Systems Period 4

Research Paper Draft 1

Feb 2, 2017

Natural Language Processing + Syntax = Efficient?

With over 2000 lines of code and nearly 6 months of effort, the project was near completion for its first phase. The program was intended to assist students trying to learn how programming works without the need of learning an actual language. We were designing an interpreter which would interpret a user's program written in the English language and convert it into Java. The user would simply type the logic for a given problem in English, run a very simple command line prompt, and view the results in the command prompt screen. The problem with having so much code condensed in such a small screen was that it made it hard to connect functions with other similar methods. This problem became evident through one of our preliminary tests. Using the methods we had already created, we typed up some pseudo-code in a text file to test the program. Everything was going fine, till the interpreter spit out this error: "[UNKOWN] ERROR FOUND IN LINE [54]." The only problem was that the interpreter's error referred to a line number in my pseudo-code and not my actual program, thus making it extremely difficult to figure out what went wrong.

The premise of this project revolves around Natural Language Processing. Natural Language Processing is a programming concept in which a given program is capable of understanding and comprehending an unknown language for the system (W. Phillips, n.d.). For example, image a boy who can only understand Spanish. Now as the

programmer you add Natural Language Processing to help the boy understand English. My project uses this concept to convert the user's English text into its appropriate Java commands.

The project also requires a basic detail of the rules and syntax for the user to use in order to make the task of the interpreter easier and more reliable. Syntax is the design and structure of any programming language. It provides the rules and layouts for how the language can be written much like how grammar works in English (Qui et al., 2016). With the correct combination of flexible syntax and natural language processing an effective and usable product can be designed.

The first question which arises for this project is whether or not it is feasible. Natural Language Processing is a very time-consuming process and creating an actual language using this concept would be even more time-consuming. According to Good et al., "...natural language provides support for understanding computational concepts, [but] it introduces additional difficulties when used for coding" (2016). Dr. Good et al. conducted an experiment to see whether or not applications, specifically game designing applications which use natural language processing, effectively work for its intended person by a layperson who doesn't know about the application (Good et al., 2016).

Using students from the English departments from various universities, researchers asked for them to create a story line for a game and use the three different applications to program the game (Good et al., 2016). According to their results, the researchers concluded that while Natural Language Processing does have its positives, it cannot be declared as a new generation programming language as it has no structure, compared to other generation languages, and is thus unreliable (Good et al., 2016). Thus,

it is proven that Natural Language Processing by itself cannot be treated as a Programming Language as it doesn't have the structure or layout of a proper programming language. In order for it to be a language one would need to add structure, or in the programming world: syntax.

Since Natural Language Processing is too flexible to follow the rules of a programming language, the only way for one to create a language using this concept would be to add syntax. In a study done by Qiu et al., they conducted an experiment to "understand how programming language syntax is employed in actual development and explore their potential applications based on the results of syntax usage analysis" (Qiu et al., 2016). Their research describes the various processes used in writing syntax and how syntax is determined usable/structured vs unusable/flexible. In addition, their research is based on the language Java, the programming language I am basing my project off of as well thus simplifying my research as the rules of Java are clearly laid out making my syntax writing much easier for the programming language I am creating.

This may seem to be a simple solution, but what adding syntax actually does is remove the entire reason for adding a natural language processor. A Natural Language Processor allows the users to input any language he/she would like to use whereas syntax limits the user to one word/phrase. The major problems which arise are thus whether or not a flexible and structured syntax can be created and also whether or not the syntax will be efficient. After emailing Dr. Barr, a researcher who worked on this paper (Qiu et al., 2016), he replied stating that it is very plausible for a flexible and structured programming language. An easy way to accomplish this would be to match synonyms to a base word, thus allowing the user to make a selection from a wide variety of choices.

About efficiency, he replied stating that efficiency is something yet to be discovered as there are no means of describing or calculating the efficiency of the syntax.

More recently, “researchers at MIT's Computer Science and Artificial Intelligence Laboratory have demonstrated that, for a few specific tasks, it's possible to write computer programs using ordinary language rather than special-purpose programming languages” (Carleen, 2013). This is practically what I am trying to accomplish but to a different audience and on a different scale. This recent news article was published on July 11, 2013, and describes the progress made by MIT researchers in developing this program.

While the process of figuring out where the interpreter code is failing is rather tedious, the task is very much needed. Currently our program is able to convert basic syntax commands into their respective Java commands, and we are still working on making the program more flexible with various syntax. Our goal for this project is simply that: help people learn the skills needed to become a better programmer without the stress of learning a frustrating language.

References

Carleen, A. (2013, July 11). Researchers develop systems that convert ordinary language to code [Newsgroup post]. Retrieved from EurekaAlert! website: https://www.eurekalert.org/pub_releases/2013-07/miot-rds071113.php

Good, J., & Howland, K. (2016, October 28). Programming language, natural language? Supporting the diverse computational activities of novice programmers. Retrieved from Science Direct database. (Accession No.

http://dx.doi.org/10.1016/j.jvlc.2016.10.008)

Li, W., Zhang, X., Niu, C., Jiang, Y., & Srihari, R. (2013, July 7). An expert lexicon approach to identifying English phrasal verbs. Retrieved from ACM Digital Library database. (Accession No. 10.3115/1075096.1075161)

Paul, N. P., Revathy, P., Sini, G. M., & Binu, R. (2016). Automatic AMR Generation for Simple Sentences Using Dependency Parser. Retrieved from Science Direct database. (Accession No. http://dx.doi.org/10.1016/j.protcy.2016.05.119)

Perovšek, M., Kranjc, J., Erjavec, T., Cestnik, B., & Lavra?, N. (2016, June 1). http://www.sciencedirect.com/science/article/pii/S0167642316000113. Retrieved from Science Direct database. (Accession No. http://dx.doi.org/10.1016/j.scico.2016.01.001)

Phillips, W. (n.d.). Introduction to Natural Language Processing. Retrieved January 29, 2017, from Consortium on Cognitive Science Instruction website: http://www.mind.ilstu.edu/curriculum/protothinker/natural_language_processing.php

Qiu, D., Li, B., Barr, E. T., & Su, Z. (2016, October 20). Understanding the syntactic rule usage in java. Retrieved from Science Direct database. (Accession No. http://dx.doi.org/10.1016/j.jss.2016.10.017)

Sheena, N., Jasmine, S. M., & Joseph, S. (2016, September). Automatic Extraction of Hypernym & Meronym Relations in English Sentences Using Dependency Parser. Retrieved from Science Direct database. (Accession No. http://dx.doi.org/10.1016/j.procs.2016.07.269)