2017 Fall CSIT9000F Optional Project
The Winograd Schema Challenge

Due Date: 23:59 on Novemver 20, 2017

This is an individual project on writing a program to compete in the Winograd Schema Challenge.

1 Project Objectives

This is an open ended project aimed at motivating you to work on a difficult AI problem. Specifically, to do this project, you'll need to:

- Understand natural language processing (NLP) and make use of an NLP tool.
- Understand Pronoun Disambiguation Problems (PDPs).
- Understand and make use of AI techniques such as heuristic search, knowledge representation, reasoning and machine learning.

2 Description

The Winograd Schema Challenge is about pronoun disambiguation. An example question is as follows:

> The city councilmen refused the demonstrators a permit because they feared violence.
>
> Snippet: they feared violence
>
> – The city councilmen
> – The demonstrators

Questions like this are often easy for human, but so far hard for computers. Details about this challenge can be found on the following web page, which has a collection of Winograd schemas as well as sample input and output.

http://www.cs.nyu.edu/faculty/davise/papers/WinogradSchemas/WS.html

3 Requirements and Marking Scheme

Your program should meet the following requirements:

- **Runnable.** You can use any language but you need to include a makefile to compile your program into a runnable code on a typical unix system like Mac OS or Linux. You need to provide detailed instructions on how to run your code, incuding instructions on how to install the required libraries and packages. Do not use any commercial package that requires a license to use it.
- **Reproducible.** The outputs of separate executions must be the same. Random algorithms will not be accepted.
- **Reasonable.** Theoretical basis is necessary for a convincing algorithm. Merely searching key words by search engines followed by comparing the number of results will **NOT** be considered as an effective method.
- **Readable.** Make what you proposed understandable. For example, if you use machine learning, explain how it works in terms of training set (if there is any), target functions (if using neuron networks, what types do you use), and the leraning algorithm.

We will mark your project according to the following scheme:

- **NLP Basics(15%)**. Be able to use an NLP tool and extract key components from a natural sentence.
- **Method and Implementation(50%)**. Put forward a feasible algorithm which meets the above requirements and implement it with any programming language.
- **Testcases(20%)**. Give two pairs of questions that your system can handle(5% each). Be able to parse the .xml file of the Collection of Winograd Schemas and return your answer to each question(10%). Accuracy is **NOT** considered when we give marks.
- **Project Report(15%)**. A detailed report which describes all above.

4 Tutorial

We briefly introduce several popular tools. All the techniques in this section are optional, which merely serve as a reference.

4.1 Stanford CoreNLP

Stanford CoreNLP provides a set of human language technology tools. It can give the base forms of words, their parts of speech, mark up the structure of sentences in terms of phrases and syntactic dependencies, indicate which noun

phrases refer to the same entities, indicate sentiment, extract particular or open-class relations between entity mentions, get the quotes people said, etc.

You can get Stanford CoreNLP at:

https://stanfordnlp.github.io/CoreNLP/

4.2 Prover9 and Mace4

Prover9 is an automated theorem prover for first-order and equational logic, and Mace4 searches for finite models and counterexamples. Out of several logic reasoning solvers we introduce Prover9 as it's very easy to get started for novices. For more information, visit its website:

https://www.cs.unm.edu/~mccune/mace4/

4.3 word2vec

This model is used for learning vector representations of words, called "word embeddings". Natural language processing systems traditionally treat words as discrete atomic symbols, which makes it hard to use machine learning. Representing words as unique, discrete ids furthermore leads to data sparsity, and usually means that we may need more data in order to successfully train statistical models. Using vector representations can overcome some of these obstacles. You can find multiple implementations of word2vec from its wiki:

https://en.wikipedia.org/wiki/Word2vec#Implementations

4.4 ConceptNet

ConceptNet is a freely-available semantic network, designed to help computers understand the meanings of words that people use. ConceptNet is used to create word embeddings – representations of word meanings as vectors, similar to word2vec, GloVe, or fastText, but better. For more information about ConceptNet, you can refer to:

http://conceptnet.io/

4.5 More

Again, you are encouraged to use any AI technique. Find more interesting tools by yourself and have a try.

5 Submission

Pack everything into a zip file and submit it on Canvas.