CS 486/686 Introduction to Artificial Intelligence

Fall 2016

Assignment 4 A Natural Language Generation "Hackathon"

Due Date: Monday December 5, 2016 by 11:59pm to LEARN Assignment 4 Dropbox

Purpose: This assignment is intended to further your understanding of Knowledge Representation and Natural Language Processing (NLP). You will experiment with generating natural language text from OWL ontologies using the "NaturalOWL" Natural Language Generation (NLG) engine.

This assignment may be done individually or in teams of 2-4 people.

You are encouraged to share knowledge about Protégé/OWL, Knowledge Representation (ontologies), Natural Language Generation (NLG) concepts and resources, and the NaturalOWL NLG engine. However, you should be careful to plan your own experiments, and to document your own ideas, methodology, and results.

Where to get help:

If you encounter problems at any step you can post to the Assignment 4 LEARN Discussion Forum, see the Instructor or TAs during their office hours, arrange an individual appointment by emailing the Instructor or TAs, or bring the problem to the Assignment 4 in-class Labs.

Details are given below about how to install and use NaturalOWL. If you should encounter any problems with the installation or having question about using NaturalOwl please contact the Instructor (cdimarco@uwaterloo.ca).

Where to find the Assignment materials:

All materials for this Assignment—the assignment itself, a basic "harrypotter.owl" ontology, NaturalOWL articles, and NaturalOWL compressed tar file containing NaturalOWL plugin for Protégé and sample OWL ontologies (and other items you will not need) can be found in the "ASSIGNMENT4-MATERIALS" folder in "Assignments" on our course LEARN site.

INTRODUCTION/OVERVIEW OF THE ASSIGNMENT

In this assignment, you will gain experience with an important Natural Language Processing (NLP) task: **Automated Natural Language Generation (NLG).** More precisely, you will experiment with using a Natural Language Generation engine, "NaturalOWL", which uses a formal knowledge representation—an OWL ontology—as a resource for the language generation. You will also learn how the use of "linked" linguistic resources (e.g., lexicon, syntactic/semantic templates = "sentence plans", and more) can enhance quality and expressiveness of the output text.

What is "NaturalOWL"?

NaturalOWL is an open-source natural language generation engine written in Java. It produces English (and Greek—which we won't be using) descriptions of individuals (e.g., products or museum exhibits) and classes (e.g., types of products) from OWL 2.0 ontologies. The ontologies can be optionally associated with linguistic and user modeling resources, also represented in OWL.

For our assignment you will be installing NaturalOWL version 2.0 as a plugin within the Protégé editor. Using Protégé you will then be able to load an ontology and run NaturalOWL to generate natural language descriptions based on the contents of the ontology.

The following NaturalOWL 2.0 references are available in "ASSIGNMENT4-MATERIALS":

- ("jair-2013.pdf) I. Androutsopoulos, G. Lampouras and D. Galanis, "Generating Natural Language Descriptions from OWL Ontologies: the NaturalOWL System." *Journal of Artificial Intelligence Research (JAIR)*, 48:671-715, 2013.

A more detailed description of the system can be found in the following report:

- ("techreport-2012.pdf") I. Androutsopoulos, G. Lampouras and D. Galanis, "Generating Natural Language Descriptions from OWL Ontologies: A Detailed Presentation of the NaturalOWL System." Technical Report, Natural Language Processing Group, Department of Informatics, Athens University of Economics and Business, 2012.

NaturalOWL was initially developed in the Department of Informatics of the Athens University of Economics and Business (AUEB) during the Greek project XENIOS and was extended during the European project INDIGO. Version 2.0 of NaturalOWL was developed at AUEB in the PhD thesis of Gerasimos Lampouras.

*****In the rest of this document, it will be helpful if you have reviewed the articles above.*****

Cautionary note: NaturalOWL 2.0 is still a research prototype. If you encounter any problems using the software the Instructor and TA will provide support to the best of their ability, and in consultation with the developers of NaturalOWL.

HOW TO INSTALL AND USE NATURALOWL 2.0 IN PROTÉGÉ

This information is based on the NaturalOWL 2.0 README file by:
Gerasimos Lampouras and Ion Androutsopoulos
Natural Language Processing Group, Department of Informatics,
Athens University of Economics and Business, Greece
November 2013

Follow the steps below in order:

Step 1: Installing a Java Virtual Machine

You will need **Java 1.7u79** (Windows 10) or **Java 1.8** (Mac OSX) installed on your computer to use NaturalOWL 2.0. These can be downloaded from:

http://www.oracle.com/technetwork/java/

Mac OS X:
For Mac OSX you will need to install the **JRE** version 1.8.111 and also the **legacy Java 6 runtime** (needed for the version 4.3 of Protégé we will be using.)

a. Go to: http://www.oracle.com/technetwork/java/
b. Under "Software Downloads" click on "Java Downloads".

c. Click on “JRE Download”.
d. Accept the license agreement.
e. Download and install Java SE Runtime Environment 8u111.
f. Go to: https://support.apple.com/kb/DL1572?locale=en_US
g. Download Java for OS X 2015-001 and install.

Windows 10:
For Windows 10 you will need to download the **older version 1.7u79**. (This is due to issues with Java 1.8 with Windows 10 and the version 4.3 of Protégé we will be using.)

a. Go to: http://www.oracle.com/technetwork/java/
b. Scroll to the bottom of the screen to “Java Archive” and click on “Download”.
c. Click on “Jave SE 7”.
d. Download Java SE Runtime Environment 7u79—Windows x64 version (what we tested).
e. You will need to accept the license agreement.
h. You may need to create a new account in order to download.
i. We used 7-ZIP to unzip the file (other utilities may also work).

To check the correct version of JRE is now installed on your computer, type the following in a command-line shell:

```
java -version
```

Step 2: Installing Protégé 4.3 on your own machine

NaturalOWL 2.0 runs with Protégé version 4.3. **You should not use a later version of Protégé—the current Protégé 5.0 is not compatible with NaturalOWL 2.0. You will need to re-install Protégé using version 4.3 on your machine.** The 4.3 version of Protégé can be downloaded from:

http://protege.stanford.edu/download/protege/4.3/

a. Click on “installanywhere”.
b. Click on “Web Installers”.
    i. For Mac OSX click to download without Java VM.
    ii. For Windows 64bit click to download without Java VM

Do not select an installer that includes a “Java VM” as this may install an older version of the Java Virtual Machine (VM) than the version required by NaturalOWL 2.0.

c. After downloading Protégé, follow the installation instructions on the same web page. You will be asked to specify an installation directory. Write down the installation directory you specified. (For Mac OSX we installed Protégé 4.3 in the “Applications” folder.)

Step 3: Installing NaturalOWL 2.0 as a Protégé plug-in

a. Copy the “NaturalOWL2.0.tar.gz” file from “ASSIGNMENT4-MATERIALS” into a directory on your own machine. Unzip and expand this file. This will create a directory “NaturalOWL2.0”.

b. Go to the Protégé 4.3 installation directory of Step 2 and find the "plugins" sub-directory. Copy "gr.aueb.cs.nlg.NLOwlPlugin.jar" from the NaturalOWL2.0 directory to "plugins".

Step 4: Testing NaturalOWL 2.0 with the sample MPIRO ontology

a. Start Protégé 4.3. (You may be asked to install some plugins; if so restart Protégé 4.3.)
   i. For Mac OSX we doubleclicked on the Protégé icon in the Protégé_4.3 installation directory of Step 2.
   ii. For Windows we doubleclicked on "protege.exe" in the installation directory of of Step 2.

   You should now see "NaturalOWL" in Protégé's topmost menu, along with "File", "Edit", "View", "Reasoner", "Tools", "Refactor", "Window", "Help".)

b. You will find the domain ontology file "mpiro.owl" in the NaturalOWL2.0/ Ontologies/ MPIRO sub-directory.
c. Load this file into Protégé and run the Reasoner. (Don't worry if you get a small error.)

Note: The M-PIRO and INDIGO domain ontologies were originally based on information provided by the Foundation of the Hellenic World (FHW, http://www.fhw.gr/). See the JAIR article cited above for information on the origin of the Consumer Electronics ontology and its sample instances.

d. To initiate the NaturalOWL natural language generation engine, after you have loaded the .owl file of a domain ontology (in our test example, "mpiro.owl"), go to Protégé's "Window" menu, select the "Tabs" sub-menu and tick "Lexicon", "NL Names", "Sentence Plans", "Sections and Order", "User Modelling", and "Text Generation" to add them to the visible tabs.

Note: You can use these six tabs to author linguistic and user modeling resources ("NL resources") as explained in the JAIR article and the technical report mentioned in the Introduction. We will give examples in class describing how to author these resources.)

Step 5: Generating text with NaturalOWL WITHOUT Natural Language (NL) resources

You can generate texts from an OWL domain ontology with or without NL resources. The quality of the texts will be much better if appropriate NL resources are provided.

To initially generate a text using "mpiro.owl" WITHOUT any additional NL resources:
Note: This demo only works for Mac OSX. See explanation below.

a. Go to the "Text Generation" tab.
b. You select a class and/or an individual (instance of a class) of the domain ontology by using Protégé's Class Browser (i.e., pane) and Individual Browser on the left. For example, since you have loaded M-PIRO's domain ontology, select Class "exhibit", then "vessel", and then "kylix" in the Classes Browser; then "exhibit22" in the Individuals Browser.
c. Make sure "English" is set as the target language in the "Text Generation" tab. (NaturalOWL will also generate Greek text, but we will not be using this feature.)

d. Note: If you tick "Generate comparisons", the generated texts will occasionally contain comparisons to individuals described in previous texts and/or other individuals of the ontology.
e. Specify the "Maximum Graph Distance in Content Selection"; see the articles mentioned in the introduction for an explanation of this parameter. For M-PIRO's ontology, we recommend setting this value to 2.
f. Press "Generate text". A textual description of the selected class or individual should appear.

It should be apparent from this initial test that the quality of text generated using the ontology alone, with no linguistic resources, will not be very fluent!

Step 5: Generating text with NaturalOWL WITH Natural Language (NL) resources

Go to the "NaturalOWL" menu and select "Open NL resources". Browse through the directories until you find the NaturalOWL2.0/Ontologies/MPIRO sub-directory. "NLResources.owl" will be in this directory. Click on its name then on "Open". Run the Reasoner.

"Generate text" as on steps (a) through (f) from Step 4. Do you notice anything different???

Note:

i. For Windows, the "NLResources.owl" file will be loaded at the same time as the domain ontology file (e.g., mpiro.owl) if they are in the same directory.
ii. To see the comparison in text generated without/with the additional linguistic resources temporarily move "NLResources.owl" to a different directory than the domain ontology file, then open it manually yourself.

iii. If you generate texts for several individuals (e.g., try generating previews for the three "kylikes" of M-PIRO's ontology, then generate a preview for the lekythos "exhibit15"), you will notice that NaturalOWL 2.0 avoids repeating information that has already been conveyed; if you have ticked "Generate comparisons", it will also generate comparisons to previous individuals.

Other generation options:

i. Clicking on "Reset interaction history" will force NaturalOWL 2.0 to "forget" the previous descriptions it has produced.

ii. NaturalOWL 2.0 will generate text that is appropriate to the reader (i.e., adult, child, expert). In M-PIRO's domain ontology, the texts for experts do not convey information that the experts would already know (e.g., what a hydria was used for), nor information they would be able to infer from the image of the exhibit (e.g., that a particular amphora was painted with the red-figure technique).

iii. "Show Syntactic and Semantic Annotations" will show the generated texts with syntactic and semantic markup in XML.

iv. "Shape text paragraphs" will shape the generated text into paragraphs corresponding to the authored sections.

v. "Markup use of default resources" will colour the sentence plans and NL names in the text that are generated via built-in NL resources.

Hackathon Day 1: Lecture 21 Tuesday November 21

Authoring NL Resources/Testing NaturalOWL 2.0

We'll start with a "bare-bones" domain ontology to practice authoring various NL resources.

a. Load "harrypotter.owl" from "ASSIGNMENT4-MATERIALS".
b. Run the Reasoner.
c. Make sure all the NaturalOWL Tabs (see Step 4c above) are selected.
d. Go to the "Text Generation" tab.
e. Select Class "Student" in the Classes Browser.
f. Select "HarryPotter" in the Individuals Browser.
g. Make sure "English" is set as the target language in the "Text Generation" tab.
h. Specify the "Maximum Graph Distance in Content Selection to have value 1 for this demo.
i. Press "Generate text". A textual description of the individual "Harry Potter" should appear.

How can we make this text more natural-sounding and fluent?

a. First, let's create some lexical items to use English words in place of Class names.
b. Demo—Lexicon: Create adjective "good".
c. Demo—Lexicon: Create noun "wizard".
d. Save the current ontology. You will be asked to save to a new Natural Language (NL) resource ontology. It is suggested you use the name "NLResources.owl". Note: This file should be saved to the same directory as your domain ontology file (i.e., harrypotter.owl).
e. Demo—NL Names: Create a NL Name "GoodWizard". Connect to Class "GoodWizard".
f. Rerun the Text Generation.
    - Q: Any better? What more is needed?
g. Demo—Modifying the built-in sentence plan "isASPEN" (isA Sentence Plan English).
h. Demo—NL Names: Create a Name "Student". Connect to Class "Student".
i. Demo—Sentence Plans: Creating a sentence plan for "OwnerOf".
j. Demo—Lexicon, NL Names: Create a masculine entry for "Harry Potter".
k. Demos—Sentence Plans:
    - Creating a sentence plan for "HouseMemberOf".
    - Q: Do we need a new lexical entry for "house member"?
    - Creating a sentence plan for "StudentOf".
l. Demo—Controlling the amount of content (facts) per sentence.
m. Demo—Aggregating sentences.
n. Demo—User modelling.

Important: Saving NL Resources

You should save the linguistic resources you have authored by selecting Protégé's "NaturalOWL" menu, and then "Save NL resources" or "Save NL resources as…". In the same menu, you can also create a new NL resources ontology, open an existing one, or import the resources from an additional NL resources ontology to the currently loaded one. **Only one NL resources ontology can be loaded at a time.**
Note: Only the linguistic resources are saved in the NL resources ontology. The user modeling and ordering annotations are saved in the domain ontology; to save the domain ontology, select the "File" menu, and then "Save" or "Save as…".

(Windows only) If a NL resources ontology is saved in a file named "NLResources.owl" and placed in the same directory as the .owl file of the domain ontology, it will also automatically be loaded when the domain ontology is loaded.

Hackathon Day 2: Lecture 22 Thursday November 23

- **Before class read through the assignment description below.**
- **Think about which issues interest you.**
- **During class we will build teams.**
- **We will also work with each team on drafting an outline of their plan for the assignment.**

Hackathon Drop-in Labs will be scheduled—details will be posted on LEARN and to the class mailing list

Assignment: A Natural Language Generation Hackathon—Some Ideas

1. First decide on the team you want to work with (1–4 people).
2. Decide which of the issues (1)–(6) below you and your team will work on.
3. Note that you can work on one more of (1) through (6) but one issue is fine.
4. Draw up an experimental plan:
    a. What issues you plan to research;
    b. What your primary domain ontology will be about;
    c. What existing ontologies or other resources you will use;
    d. Planned extensions to the domain ontology or resources.
    e. What hypotheses you will test, e.g.,
        i. Merging ontologies on the same topic will lead to more expressive output text;
        ii. Creating additional subClasses and Properties for a given ontology will lead to more expressive text;
        iii. Adding more sentence plans or other natural language resources will lead to more fluent text;
        iv. And so forth.

The Instructor and TA will work with you to draw up your experimental plan.

Suggested Issues to Work On for the Assignment:

1. **Knowledge Representation issues:**

a. Get NaturalOWL working on a seed ontology: the Harry Potter example, your own Assignment 2 ontology, another ontology from the OWL libraries.
b. You will need to create some basic Lexicon entries, NL Names, and 4-5 Sentence Plans.
c. Experiment with adding more-specialized Classes and Properties.
d. Report on how building up the ontology affects the output text generated.
e. Also document any problems you encountered and how these were resolved.

2. **Advanced Knowledge Representation:**
(Should be attempted only after doing (1) above and consulting with the Instructor.)

a. Try merging ontologies.
b. Report on how the merging affects the output text generated.

3. **Natural Language Resource issues:**

a. Get NaturalOWL working on a seed ontology: the Harry Potter example, your own Assignment 2 ontology, another ontology from the OWL libraries.
b. You will need to create some basic Lexicon entries, NL Names, and 3-4 Sentence Plans.
c. Experiment with enriching the Lexicon, NL Names, and additional 5-10 Sentence Plans.
d. Report on how building up language resources affects the output text generated.
e. Also document any problems you encountered and how these were resolved.

4. **Advanced Natural Language issues**
(Should be attempted only after doing (3) above and consulting with the Instructor.)

a. ***(Document planning)*** Try controlling how components of the output text get ordered:
    i. Global coherence: Overall logical structure of a document.

b. ***(Microplanning)*** Try working on controlling various aspects of sentence planning:
    i. Local coherence: Logical transitions between adjacent sentences.
        - Note: You could test various semantic similarity metrics here.
    ii. Aggregation: Deciding when and how to combine sentences.
    iii. Generating referring expressions: Deciding when to replace a noun with a pronoun.

c. ***(Linguistic Resources)*** Try incorporating one or more Linguistic Linked Open Data (LLOD) resources, e.g., the WordNet lexicon. LLOD resources contain lexicons, among many other items, in RDF format (i.e., the format OWL uses).

5. **User Modelling issues:**
(Should be attempted only after doing (1) or (3) and consulting with the Instructor.)

a. Try creating different user types for your domain (e.g., Expert, Novice; Adult, Child) and creating Personal User Models to tailor the output text to their interests and/or language level and/or other characteristics.

6. **Multilingual Language Generation:**
   (Should be attempted only after doing (1) or (3) and consulting with the Instructor.)

a. NaturalOWL 2.0 can be used to generate text in multiple languages from the same underlying sentence plans. Different lexicons must be created though.

What to hand in:

a. Word or PDF file:
   - Your experimental plan;
   - Description of the parts of the plan actually implemented;
   - Note: Part (a) should be 2-3 pages. Clarity and good design will count.

b. OWL file: Your own NL Resources.

c. Word or PDF file:
   - Samples of the texts you generated;
   - Informal evaluation of the texts in terms of:
     - Fluency;
     - Interestingness.
   - Note: Part (d) should be 2-3 pages. Clarity and good design will count.

ALL FILES SHOULD BE SUBMITTED IN A SINGLE ZIPFILE TO THE ASSIGNMENT 4 LEARN DROPBOX

- **Nominate one member of your team to submit the zipfile**
- **Name the zipfile: <LastName.FirstName>.zip with the name of your team member who will be doing the submission**
- **You can submit multiple times up to the final submission due date/time**

SUPPLEMENT: Modifying NaturalOWL 2.0's source code and using it within Protégé
(This information is provided by the NaturalOWL developers. You are welcome to try experimenting with the NaturalOWL source code for your own interest on your own—the Instructor and Course Staff will not be able to provide assistance.)

The source code of NaturalOWL v.2.0 is included in the distribution (see directory "source"). Provided that you respect its GPL license, you may modify the source code to create a tailored version of NaturalOWL. To use the new version within Protégé, follow the following steps:

Step 1:
Extract all files of the distribution into the same directory.

Step 2:
Modify the Java files of the source code.

Step 3:
To create a new file "gr.aueb.cs.nlg.NLOwlPlugin.jar" it is recommended you also download the source code of Protégé and add the plugin there. You can find relevant instructions here: http://protegewiki.stanford.edu/wiki/CompileProtege4InEclipseFromSvn. Following them, you can build the new "gr.aueb.cs.nlg.NLOwlPlugin.jar" through Eclipse.

Step 4:
Copy "gr.aueb.cs.nlg.NLOwlPlugin.jar" to the "plugins" subdirectory of Protégé's installation directory.

==========

FINAL NOTE re: NaturalOWL 2.0

You can call NaturalOWL 2.0's generation engine directly from another application.

File "TestNLGEngine.java" provides an example of how to do this.