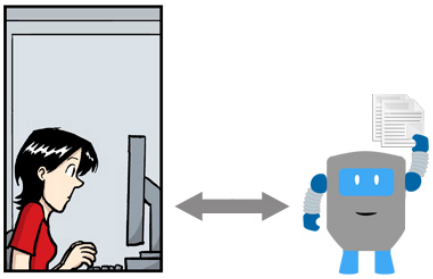
COMP 474/6741 Intelligent Systems (Winter 2022)

Project Assignment #2

Due date (Moodle Submission): Wednesday, April 13th
Counts for 19% of the total course marks

In this second part of our *Intelligent Systems* project, your tasks are:

- Fixing any issues that surfaced during the demo of Part #1;
- Populating the knowledge base with the course *content* from the two courses where you collected their material; and
- Adding a natural language interface to make it a real 'chatbot.'



A1 Updates. In case there were issues with your first project submission that were pointed out in the demo, please make sure you address them before working on the new parts.

Knowledge Base Population. We'll continue the automatic knowledge base construction for your bot's brain by adding triples representing the *content* of the two courses you captured in Part #1:

Pre-processing: Using a suitable library,[1] convert the material you collected (slides, web pages, lecture notes, lab documents, etc.) from their source format (e.g., PDF, DOCX) into plain text files.

Entity linking: Run *DBpedia Spotlight* over the converted documents in order to link entities that appear in them to DBpedia. For example, the topic *"Chatbot"* appearing on a slide would be linked to http://dbpedia.org/resource/Chatbot.[2] Develop a suitable filtering strategy to post-process the results from Spotlight, so that only named entities are matched (e.g., using POS tags or other linguistic features computed with spaCy).

Triplification: Using your RDF Schema from Part #1 of the project, encode the information you extracted in form of triples that link them to the corresponding URI of the course event (lab, lecture, tutorial, etc.) and material URIs[3] and add them your KB served by Fuseki.

Make sure all your URIs are properly connected: you have multiple courses, each course has multiple events (labs, lectures, ...), each event has multiple associated resources (slides, web pages, ...) and each resource has (typically) multiple topic mentions.

Write SPARQL queries that can answer the following questions:

1. For a course c, list all covered topics t, printing out their English labels and their DBpedia URI, together with the course event URI (e.g., 'lab3') and resource URI (e.g., 'slides10') where they appeared.
2. For a given topic t (DBpedia URI), list all courses where they appear, together with a count, sorted by frequency.
3. For a given topic t, list the precise course URI, course event URI and corresponding resource URI where the topic is covered (e.g., "NLP" is covered in COMP474 → Lecture 10→ Lab 10 → Lab Notes)

[1] For example, *Apache Tika*: https://tika.apache.org/
[2] Note that you will need to install a local copy of *Spotlight*, as frequent requests to the public Web API will get you blocked.
[3] Generally the `file://` URIs from Part #1, unless you set up a document server.

University Chatbot. For users to be able to interact with your bot, you need to develop a natural language interface to your knowledge base (i.e., a *grounding*-based bot). It has to be able to answer (at least) the following questions:

1. All of your competency questions from Part #1.
2. *"What is the <course> about?"*
   E.g, "What is COMP 474 about?": provides the course description as answer.
3. *"Which topics are covered in <course_event>?"*
   E.g, "Which topics are covered in Lab #2 of COMP 474?": provides the topics (in English), together with their resource URI where they can be found.
4. *"Which courses cover <Topic>?"*
   E.g., "Which courses cover Expert Systems?": lists all courses that include this topic, sorted by frequency of the topic.

You have to implement your natural language interface using the *Rasa* chatbot framework.[4] Implement suitable questions that you can map to SPARQL queries and answer using your Fuseki server, translating the response triples into natural language answers.

Report. Update your report from Part #1, i.e., submit a single, combined & revised report for your complete project, with the following changes:

Report updates: Fix any mistakes (spelling, structure, etc.) from Part #1.

KB population: Add a section (ca. one page) describing your process of creating the topic triples for your knowledge base (processing, linking, filtering), together with a table providing statistics for the generated triples (total number of triples, number of distinct topics, number of topic instances/course).

Chatbot design: Describe your method of translating the input questions into SPARQL queries, based on Rasa. Provide at least one example input & output for each of the questions above.

Deliverables. Your submission must include the following deliverables within a single `.zip` archive:

Deliverables from A1: All the parts of your project as listed for Part #1 (with any modifications you might have made for Part #2).

New queries & results: Make sure you also add the new topic SPARQL queries described above together with their output.

Chatbot: Your new chatbot Python code, including any associated resources (configuration files, training data, etc.).

Report: The project report, as detailed above, as PDF.

Submission. You must submit your code electronically on Moodle by the due date (late submission will incur a penalty, see Moodle for details). *Only if your group members have changed from Part #1:* Include a new, signed by all team members, *Expectation of originality* form (see https://www.concordia.ca/encs/students/sas/expectation-originality.html) with your submission.

Demo. We will schedule demo sessions for your project using the Moodle scheduler. Demos will take place on-campus, in-person. All team members must be present for the demo.

[4] See https://rasa.com/docs/rasa/installation/