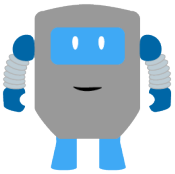
COMP 474/6741 Intelligent Systems (Winter 2024)

Project Assignment #1

Due date (Moodle Submission): Tuesday, March 19th
Counts for 50% of the course project

Meet Roboprof. *Staring into the abyss of academic challenges, you might find yourself pondering deep, existential questions: "Where can I unravel the mysteries of NLP? Will DEEP 490 illuminate the dark recesses of my neural network knowledge? And why does the cafeteria coffee taste like a weak regression model?" Fear not, for* Roboprof *is here to light the way! This isn't your average, run-of-the-mill digital assistant. Oh no, Roboprof is a marvel of modern technology, wielding the mighty tools of Knowledge Graphs, Natural Language Processing, and Deep Learning to navigate the academic seas. With a byte of wisdom and a bit of wit, this intelligent agent is your steadfast companion on this scholarly adventure, ready to answer your burning questions faster than you can say "eigenvalues"!*

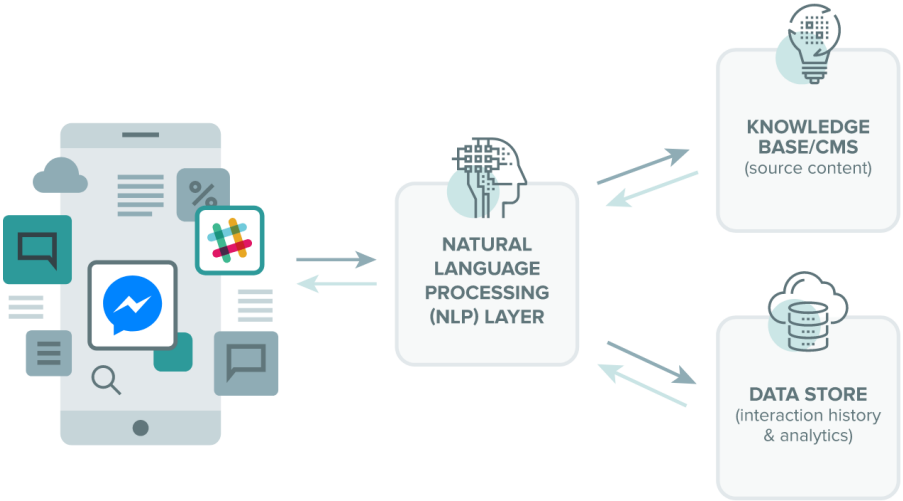
This is the first of two project assignments in our course. Note that the assignments are not 'stand-alone', but rather build on top of each other to form a bigger project.



The overall goal of this project is to build **Roboprof**, an *intelligent agent* that can answer university course- and student-related questions, using a knowledge graph and natural language processing.

For example, Roboprof will be able to answer questions such as, *"What is course COMP 474 about?"*, *"Which topics is Jane competent in?"* or *"Which courses at Concordia teach deep learning?"*

Project structure. The initial assignment is dedicated to the construction of the knowledge base. The subsequent phase will focus on refining this base with additional enhancements, developing the natural language processing interface, and integrating these components into a cohesive system.



This structured approach ensures a foundational understanding of both the theoretical aspects and practical applications of AI technologies, laying the groundwork for advanced exploration in the subsequent assignment.

Roboprof's Knowledge Graph Vocabulary. To competently respond to queries, *Roboprof* requires a wealth of knowledge on courses, lectures, and their respective content. Consequently, the initial phase of this project involves constructing a knowledge graph utilizing standard W3C technologies, specifically RDF (Resource Description Framework) and RDFS (RDF Schema). You are tasked with modeling your graph to encapsulate at least the following information (you may include additional details you deem beneficial):

Universities. Information about universities:

1. The name of the university
2. Link to the university's entry in *DBpedia* and/or *Wikidata*

Courses. Information about *courses* offered at a university, including:

1. Course name (e.g., "Intelligent Systems")
2. Course subject (e.g., "COMP", "SOEN")
3. Course number (e.g., "474")
4. Course credits
5. Course description (e.g., "Knowledge representation and reasoning. Uncertainty and conflict resolution . . . ")
6. A link (`rdfs:seeAlso`) to a webpage with the course information, if available.
7. Course outline, if available.

Lectures. Information about *lectures* in a course, including:

1. Lecture number (sequential count)
2. Lecture name, if available (e.g., "Knowledge Graphs")
3. Lecture content, such as:
   - Slides
   - Worksheets
   - Readings (book chapters, web pages, etc.)
   - Other material (videos, images, etc.)

   (create a subclass hierarchy for different lecture content types)
4. A link (`rdfs:seeAlso`) to a web page with the lecture information, if available

Topics. Information about the *topics* that are covered in a course:

1. The topic's name as mentioned in the source material (e.g., "DL" might be mentioned in an outline, referring to "deep learning")
2. Topics must have *provenance* information, that is, you must record where the topic was identified as being covered in the course (e.g., the (graduate) calendar, the course outline or a specific lecture)
3. A topic must be linked to DBpedia and/or Wikidata. For example, the topic "Knowledge graph" in a course description could be linked to https://dbpedia.org/resource/Knowledge_graph.

Students. Information about students:

1. Name (first, last)
2. ID Number
3. Email
4. Completed courses with their grades. Note: you do not have to model courses-in-progress, but you do have to model the possibility that a student re-takes a course (e.g., to obtain a passing grade).
5. Explicitly model the grade a student achieved in a course, taking into account that a student might re-take a course to achieved a different grade (both must be contained in your graph)
6. Competencies, defined as a set of topics, based on the courses a student has successfully completed (e.g., if a student passes COMP474, which includes the topic "Knowledge Graphs", then the student is considered competent in this topic).

Your first task is to create a **Vocabulary** using RDFS (to be submitted in Turtle format) to capture these information. Guidelines include:

- Re-use existing vocabularies where appropriate (e.g., FOAF,[1] VIVO[2]).
- If you need to define new concepts and/or properties:
  - Aim to extend existing vocabularies where feasible.
  - Ensure each of your new classes and properties is accompanied by a *label* and *comment* in English (additional languages are welcome).

Knowledge Graph Queries. As part of ensuring that Roboprof can effectively utilize the constructed knowledge graph, the following queries are designed to assess the bot's ability to accurately extract and present information based on the established knowledge graph structure:

1. List all courses offered by [university]
2. In which courses is [topic] discussed?
3. Which [topics] are covered in [course] during [lecture number]?
4. List all [courses] offered by [university] within the [subject] (e.g., "COMP", "SOEN").
5. What [materials] (slides, readings) are recommended for [topic] in [course] [number]?
6. How many credits is [course] [number] worth?
7. For [course] [number], what additional resources (links to web pages) are available?
8. Detail the content (slides, worksheets, readings) available for [lecture number] in [course] [number].
9. What reading materials are recommended for studying [topic] in [course]?
10. What competencies [topics] does a student gain after completing [course] [number]?
11. What grades did [student] achieve in [course] [number]?
12. Which [students] have completed [course] [number]?
13. Print a *transcript* for a [student], listing all the course taken with their grades.

[1] See http://xmlns.com/foaf/spec/
[2] See https://wiki.lyrasis.org/display/VIVODOC114x/VIVO+1.14.x+Documentation

These queries are foundational for Roboprof to function as an educational assistant, facilitating access to a wide array of academic information. Note that competencies are dynamically established based on a student's (successfully) completed courses.

Create at least one SPARQL query for each of these question that validates your vocabulary design. Your queries must show the *label* for each queried entity, in addition to its URI (e.g., you might have a topic URI `http://.../T5421` that has the English label "Neural Network").

Automated Knowledge Base Construction. To fill our agent's brain, we need to create a *knowledge base* using the vocabulary defined above. Towards this end, in this first part you have to populate the knowledge base with information about Concordia's courses from the open datasets available at https://opendata.concordia.ca/datasets/.

For the *content* of a course (lectures, labs, tutorials, etc.), you will have to add two courses to your knowledge base:

1. Our course, COMP 474/6741
2. A second course that you took in the past (you can pick one as a team, but at least one team member must have taken the course so you have access to the material)

In this first part, collect and organize the material for the two courses into suitable datasets (e.g., folders with lectures, slides, etc.).[3] Give each item an automatically generated URI that you can use to identify the item. You can use `file://` URIs that resolve locally,[4] for example, to refer to a set of slides in our course, you could generate a URI like `file:///home/roboprof/COMP474_6741/lectures/slides01.pdf`

In the second part of the project, we will automatically *extract* information about topics using NLP techniques from these datasets. For this first part of the project, **simply create a few triples manually** that demonstrate how the knowledge base will look like (e.g., modeling a few topics for our course) and that enable you to run the above queries.

For students with their grades, create a simple spreadsheet with example data that you convert automatically into triples with a Python program using RDFLib.

Make sure that all these data triples are *in a different namespace* from your vocabulary.

Triplestore and SPARQL Endpoint. Configure a query server utilizing Apache Fuseki[5] to load and interact with your generated triples. Execute the queries designed for the graph questions, and document sample outputs for each. Additionally, formulate queries to derive statistics about your knowledge base (KB), including the total number of triples and the count of course URIs present.

Report. Compile a report on your design and implementation, incorporating the following sections:

Title page: The title page of your report must have:

- Your group details such as the Moodle team name, team members, student ID numbers, each team member's specialization, and a link to your project repository (e.g., on Github).
- Add the following statement and signatures to the title page: *"We certify that this submission is the original work of members of the group and meets the Faculty's Expectations of Originality"*, with the signatures and ID numbers of all the team members and the date.

[3] You do not need to add video files, since we are not processing these further within this project.
[4] You could also setup a document server in order to create HTTP-resolvable URIs for the course content, using a server like Couchbase or CouchDB, but this is not required for this project.
[5] See https://jena.apache.org/documentation/fuseki2/

Vocabulary: Description how you modeled the schema for your knowledge base, including the vocabularies you reused, any vocabulary extensions you developed, etc. Give brief justifications where appropriate (e.g., choice of existing vocabulary).

Knowledge Base Construction: Describe (a) your dataset and (b) your process and developed tools for populating the knowledge base from the dataset. Describe how to run the tools to create the knowledge base.

Graph Queries: Describe your translation of the knowledge graph questions above into SPARQL queries. Provide (brief) example output for each query.

Triplestore and SPARQL Endpoint Setup: Outline the steps taken to set up the Triplestore and SPARQL Endpoint, highlighting any challenges encountered and how they were addressed.

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

RDF Schema: Your RDFS file in Turtle format.

Dataset: Your dataset used to construct the knowledge base (e.g., course descriptions, academic calendars, etc.), in their original and any converted formats. Provide the source information for all files (filename, source URL).

KB Construction: Your Python program(s) to automatically construct the knowledge base from the dataset above. In other words, it must be possible to re-construct your knowledge base using the submitted Python tools and datasets.

Knowledge Base: Your complete, constructed knowledge base (in both N-Triples and Turtle format).

Queries and their result: Text files with your queries for the questions (`q1.txt`, `q2.txt`, ...) and the full output of your queries when run on your KB (`q1-out.csv`, `q2-out.csv`, ...).

README. A comprehensive `readme.txt` or `readme.md` file:

- It must enumerate the contents and describe the purpose of each file in your submission.
- Clearly outline the steps to execute your code for building the knowledge base. If your instructions are incomplete and your code cannot be run you might not receive marks for your work.

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

Originality Form. Include an *Expectation of Originality* form[6] for each team member:

- This form, attesting to the originality of your work, must be electronically signed and we need a form from each team member.
- If a form is missing, your project will not be marked!

Submission Procedure: You must submit your code electronically on Moodle by the due date (late submission will incur a penalty, see Moodle for details).

Project Demo. We will schedule demo sessions for your project using the Moodle scheduler. The demos will be on campus and all team members must be present for the demo. Guidelines for preparing for the demo will also be posted on Moodle.

[6] Available at https://www.concordia.ca/ginacody/students/academic-services/expectation-of-originality.html

Project Contribution and Grading Policy. In the spirit of collaboration and team unity, by default, all team members will receive the same grade for the project. This default policy is based on the expectation that all team members actively contribute, collaborate, and collectively drive the project to completion.

However, we recognize that team dynamics can vary, and there might be instances where contributions are disproportionate. In the event of a dispute regarding contributions:

1. The team must first attempt to resolve the dispute internally. Open dialogue and clear communication are encouraged to ensure all members are aligned on expectations and deliverables.
2. If the internal resolution is unsuccessful and team members believe that the contributions have been significantly uneven, the team should approach their designated TA (acting as their "project manager") with their concerns. The TA will provide guidance and possibly mediate to help resolve the issue.
3. If after the TA's intervention the dispute remains unresolved, the team may ask the TA to escalate the matter to the course instructor.
4. For a dispute to be considered at this stage, the team must provide clear evidence that delineates individual contributions. This evidence should ideally be in the form of version control records, such as Github change logs, commit messages, pull requests, and code reviews. Merely having more commits doesn't necessarily indicate greater contribution; the quality, relevance, and impact of the changes will be evaluated.
5. Along with the evidence, a written statement detailing the nature of the dispute, reasons for the perceived uneven contribution, prior attempts to resolve the matter internally, and the involvement of the TA should be submitted.
6. The course instructor will review the submitted evidence and statements. After consideration, they may adjust individual grades to reflect the contributions more accurately. This decision is final.
7. Any claims towards the contribution made after the final (late) submission deadline for each project part will not be considered.

Teams are strongly encouraged to maintain regular communication with their TA throughout the project to preemptively address any potential conflicts and ensure smooth progression. The goal of this policy is not to encourage disputes but to provide a fair mechanism for resolution in the rare event it's needed.

Academic Integrity Guidelines for the Roboprof Project. Upholding the principles of academic integrity is paramount to the learning process. To ensure fairness, clarity, and ethical behavior throughout this project, the following guidelines have been set:

1. **Originality of Work:** All submissions must be the original work of the team members. Copying or adapting work from other teams is strictly prohibited. Using external sources without proper citation is also strictly prohibited (this includes ChatGTP and similar tools, see below). Such actions will be considered academic dishonesty and may lead to a failing grade for the project or other disciplinary actions as deemed appropriate by the university's academic integrity policies. Please make sure you review the academic code of conduct at https://www.concordia.ca/conduct/academic-integrity.html. Not knowing the code is not a valid defence for violating it!
2. **Citing External Sources:** Should you use external sources, such as datasets, code snippets, or any other resources, you must provide clear citations. Ensure that you give appropriate credit by adding the source into your report's reference section as defined above (using the IEEE format).

Remember, acknowledging sources not only maintains academic integrity but also highlights your diligence in research.

3. **Usage of Large Language Models (LLMs) like ChatGPT:** Recognizing the relevance of LLMs in modern AI, the use of tools like ChatGPT is permitted with specific restrictions:
   a. LLMs should complement your work, not serve as the main source of your solutions or content.
   b. When using LLMs, both the prompt you provided and the response from the model must be clearly displayed in your report. This ensures transparency and allows for a clear differentiation between student work and LLM-generated content.
   c. While LLMs can offer insights or clarify concepts, relying heavily on them diminishes the learning experience. Aim to understand and articulate in your words, using LLMs as a supportive tool, not a primary crutch.
4. **Collaboration vs. Copying:** While collaboration is encouraged for brainstorming and problem-solving, always ensure that what you submit is your team's authentic work. Sharing code, data, or report content between teams is considered a breach of these guidelines.
5. **Ensure Private Repositories:** You are strongly encouraged to make use of an online repository to coordinate and store your team's work, e.g., using Github or Gitlab. However, it is crucial that you make your repository *private* (make sure you give access to your team's TA). If your repository is public and another team uses your work, *both* your team (for sharing) and the copying team (for using) can be held accountable for academic misconduct.

Remember, the purpose of this project is to immerse yourself in the world of AI, develop skills, and gain a profound understanding of the challenges and responsibilities that come with the domain. Adhering to these academic integrity guidelines ensures a level playing field and a genuine learning experience for all.