# COMP 474/6741 Intelligent Systems (Winter 2024)

## Worksheet #6: Intelligent Agents

**Task 1.** What kind of question would you expect a *Concordia Chatbot* to be able to handle?

1. For *new students* (not currently registered at Concordia):
   - How do I enroll in a class?
   - .....
   - .....
2. For *current students* (already enrolled at Concordia):
   - .....
   - .....
   - .....

**Task 2.** Write a *regular expression* that matches different variations of naming https://www.wikidata.org/wiki/Q326342: *"Concordia"*, *"Concordia U."*, *"CU"*, *"Concordia University"*, *"Université Concordia"*, . . . :

https:\/\/www\.wikidata\.org\/wiki\/Q326342|Concordia(\sU\.?|(\s)?University)?|CU|Universit(é|e)\sConcordia

Test it at https://regex101.com/

\s = whitespace  \.?: This means that the dot . is optional.

**Task 3.** Suppose we want to use an existing set of 1000 questions for training a ML classifier. If we use tf-idf vectors to represent each question, how many dimensions will the vectors have (make a rough estimate)?

10 unique words on average

1000 * 10 = 10,0000 unique word set

**Task 4.** Ok, here is an (extremely simplified) idea of creating 2D feature vectors out of a natural language question: The first dimension $a$ encodes the first occurrence of a question word (see table below) and the second dimension $b$ the number of Capital Letters in the sentence:

| Contains? | Value |
|---|---|
| Who | 1 |
| What | 2 |
| Where | 3 |
| *(none)* | 0 |

| # | Question | a | b | Class |
|---|---|---|---|---|
| 1 | Where is Concordia? | 3 | 1 | Location |
| 2 | Who was Steve Jobs? | 1 | 2 | Definition |
| 3 | What city is McGill in? | 2 | 2 | Location |
| 4 | What is NLP? | 2 | 1 | Definition |

**Task 5.** Using the online parser at https://corenlp.run/, create a *parse tree* for the sentence *What is McGill?*. Note that you can now extract the *subject* of the sentence, e.g., to plug it into a SPARQL query.

Results: SUBJECT = McGill

**Task 6.** Now apply the kNN classification algorithm on the new question below to classify its type, according to the training data from Task 4. Use $k = 3$ and the Euclidian distance $d(\vec{p}, \vec{q}) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$:

| # | Question | a | b | d-Q1 | d-Q2 | d-Q3 | d-Q4 | Class? |
|---|---|---|---|---|---|---|---|---|
| 5 | What is McGill? | 2 | 2 | 1 | 1 | 1 | 0 | Definition |

You can now match the new question with a corresponding SPARQL template to obtain a query for your knowledge graph, filling in variables with the values extracted from the question.

$dQ1 = \sqrt{(2-3)^2 + (1-1)^2} = 1$

The nearest neighbors for k = 3 are:

(2, 1) - Definition
(3, 1) - Location
(2, 2) - Definition

Majority voting: The majority class among the nearest neighbors is "Definition".

**Task 7.** Now define a SPARQL template that can obtain information about a *person* from DBpedia. To keep it simple, for now we assume that the name extracted via NLP from the question is identical to the full name stored in the (English) `label` field for the subject (e.g., "Steve Jobs").

```
PREFIX dbr: <http://dbpedia.org/resource/>
SELECT .    .    .
WHERE {
        .    .    .

}
```

```
PREFIX dbr: <http://dbpedia.org/resource/>
SELECT ?property ?value
WHERE {
  dbr:Steve_Jobs ?property ?value .
}
```

**Task 8.** Create a *competency question* and a corresponding SPARQL query for our FOCU university example:

"What are the courses offered by FOCU University?"

Testing query for this question:

```
SELECT   .      .      .
WHERE {
            .      .      .

}
```

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?course
WHERE {
  ?university rdf:type dbo:University ;
        rdfs:label "FOCU University"@en ;
        dbo:offers ?course_resource .
  ?course_resource rdfs:label ?course .
}
```

**Task 9.** An early, well-known commercial service for semantic annotation of textual (mostly news) documents was Thompson Reuter's *OpenCalais*, which has since been spun out and re-branded as *LSEG Data & Analytics PermID* (formerly Refinitif Intelligent Tagging): Try out the online demo at https://permid.org/tagging on a document, for example the first part of the Wikipedia article on Concordia. Look at the entities that were detected and go to the "RDF view": what ID is given to Concordia in this knowledge graph?

?

*Hint:* There is another tool at the top of the page, *Entity Search*, where you can cross-check your entities.

**Task 10.** Go to the DBpedia *Spotlight* online demo at https://demo.dbpedia-spotlight.org. Try analyzing a test document with some ambiguities, e.g, *"Paris Hilton went to the Hilton in Paris."* Inspect the entities that were linked to DBpedia. Are they correct?

no, hovering over it gives:
http://ca.dbpedia.org/resource/Paris_Hilton

**Task 11.** For the questions in Task 1 above, which of the chatbot techniques covered so far would be able to answer them?