Natural Language Processing – Ex5

Please submit a zip file with your code and a README txt file,
and a single pdf file for the theoretical questions

Due: 2.2.2025 23:55

In this Python exercise we will implement a simple open information extraction system that takes text from Wikipedia and extracts triplets of (Subject, Relation, Object), where each of them is a span of text. In this exercise, the Subject and Object slot fillers are names (proper nouns), and the Relation slot filler is a verb or a verb along with a preposition.

For instance, given the sentence "Brad Pitt married Angelina Jolie" we will aim to extract the triplet ("Brad Pitt","married","Angelina Jolie").

Required Packages: In order to carry out the exercise, you will need to install the *spacy* package for NLP and the *wikipedia* package that provides a Python interface for Wikipedia.

wikipedia can be installed through pip or by directly loading the files:
`https://pypi.python.org/pypi/wikipedia/`
See how to install *spacy* here:
`https://spacy.io/usage/`
You will also need the default English model of SpaCy. See instructions here:
`https://spacy.io/usage/models`.

Once installed, processing a Wikipedia page through spacy is straightforward. Example code:

```
import wikipedia, spacy

nlp = spacy.load("en_core_web_sm")
page = wikipedia.page('Brad Pitt').content
analyzed_page = nlp(page)
```

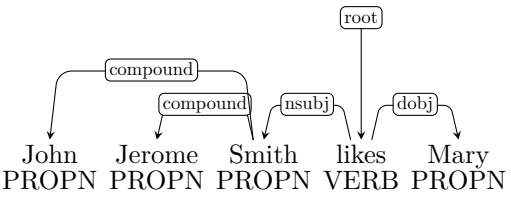
You can find more details on the *spacy* website, where the two most important data structures are *doc* (for a document) and *token*:
`https://spacy.io/api/doc`
`https://spacy.io/api/token`

1. Implement an extractor based only on the POS tags of the tokens in the document. The extractor should follow the following steps:

   - Find all proper nouns in the corpus by locating consecutive sequences of tokens with the POS PROPN.
   - Find all consecutive pairs of proper nouns such that all the tokens between them are non-punctuation (do not have the POS tag PUNCT) and at least one of the tokens between them is a verb (has the POS VERB).

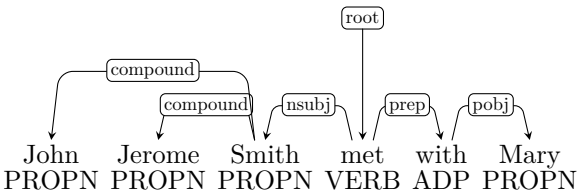
- Upon detecting a pair of proper nouns as above, output a (Subject, Relation, Object) triplet where the first proper noun is the Subject, the second proper noun is the Object, and the Relation is the sequence of tokens between the proper nouns, excluding all tokens which are not verbs or prepositions (i.e., only tokens with POS tags VERB or ADP should be included in Relation).

2. Implement an extractor based on the dependency trees of the sentences in the document [1]. The extractor should follow the following steps.



- Find all tokens that serve as heads of proper nouns in the corpus (henceforth, *proper noun heads*). Do so by locating all tokens with the POS PROPN that do not have the dependency label *compound* (i.e., the edge from them to their headword is not labeled *compound*). For instance, in the tree above, "Smith" and "Mary" are such tokens.
- For each proper noun head t, define the corresponding proper noun as a set including t along with all its children tokens that have a dependency label *compound* (i.e., the edge from them to t is labeled *compound*). For instance, in the tree above, {"John","Jerome","Smith"} and {"Mary"} are such sets.
- For each pair of proper noun heads h_1 and h_2, extract a (Subject, Relation, Object) triplet if one of the following conditions hold:
  (a) **Condition #1**: h_1 and h_2 have the same head token h, the edge (h, h_1) is labeled *nsubj* (nominal subject), and the edge (h, h_2) is labeled *dobj* (direct object).
  (b) **Condition #2**: h_1's parent in the dependency tree (denoted with h) is the same as h_2's grandparent (denote h_2's parent with h'), the edge (h, h_1) is labeled *nsubj* (nominal subject), the edge (h, h') is labeled *prep* (preposition), and the edge (h', h_2) is labeled *pobj* (prepositional object).

  An example of proper nouns that meet condition #1 are {"John","Jerome","Smith"} and {"Mary"} in the example above. An example for condition #2 is {"John","Jerome","Smith"} and {"Mary"} in the following tree:



  In both cases, the proper noun corresponding to h_1 is the Subject, the proper noun corresponding to h_2 is the Object. In the case condition #1 holds, the Relation is defined to be h, while if condition #2 holds, the Relation is defined to be the concatenation of h and h'. If none of the condition holds, no triplet is outputted.

[1] *spacy* outputs ClearNLP dependencies trees, not universal dependencies trees, so the labels and annotation may differ a little from those seen in class.

3. **Evaluation:** in order to evaluate the extractors, apply the two extractors on the Wikipedia pages corresponding to:

   - Donald Trump
   - Ruth Bader Ginsburg
   - J. K. Rowling

   For each page, report the total number of triplets outputted by the extractors. In addition, for each of the two extractors take a random sample of 15 triplets (5 from each Wikipedia page), and manually verify whether the triplet is indeed valid or not.

   For instance, ("Neil Gorsuch","appointed to","Court Supreme") is a valid triplet from the Wikipedia page of Donald Trump, as the page indeed mentions that Neil Gorsuch was appointed to the Supreme Court. The triplet ("Republican White House","combined with","Congress") is an invalid triplet because the text does not say that a Republican White House was combined with Congress or anything similar.

   You can use your own judgment when deciding on valid/invalid triples.

4. **Large Language Model:** we want to compare our previous extractors to an out-of-the-box LLM.

   For this, you will write a script with api calls to a commercial LLM. This can be either Gemini, ChatGPT, or Claude.

   Notice: Gemini offers a 'free tier' program without charge. Make sure to check the documentation.

   Report (in the answer document) the exact prompts you used as input for the model.

   Perform the same evaluation step as in the previous section.

5. **Questions:**

   (a) Compare the performance of the models. Explain why this happens.

   (b) Suppose we want to automate the evaluation. Someone suggested to use an LLM as a verifier, such that given each triplet, the model is simply asked whether the relation is true.
   Describe two limitations of this approach.

   (c) Suppose we have a list of human-extracted triplets from a document. Now we test, for each extracted triplet, whether it appears in the list of human triplets.
   Describe two limitations of this approach.

Submit all your code (including api calls) and an answer PDF. The PDF should be a separate file and the code should be in a zip file.