EDAN20 - Lab 5

André Frisk - an8218fr-s

October 2021

Extraction of subject-verb-object triples

This lab focuses on extracting the subject-verb-object triples and the lab surrounds this function to extract this (at least for the submissions). The function is further down. In this lab we use Talbankens corpus for the languages French, English, Russian and Swedish. What this function does is that it goes through every sentence in the corpus for the designated language, where every word in the sentence has the format:

```
'0': {'ID': '0',
  'FORM': 'ROOT',
  'LEMMA': 'ROOT',
  'UPOS': 'ROOT',
  'XPOS': 'ROOT',
  'FEATS': 'ROOT',
  'HEAD': '0',
  'DEPREL': 'ROOT',
  'DEPS': '0',
  'MISC': 'ROOT'}
```

This format gives information about the word for example which index this word occurs in the sentence and what form it has (word), what kind of part of speech it is and what word (index) this word points to (if they are some sort of subject-verb-object pair). If HEAD is 0 that means that the word does not point to another word and is the end destination. The variables FORM, HEAD and DEPREL (part of speech).

What this function does is that it goes through every sentence and every word in the sentence. First we look for the subject-verb pairs which is done by looking if the word contains the following: `'DEPREL': 'nsubj'` which is a subject. If this exist we extract the verbs HEAD id that is connected to the subject, we save the subject and verbs FORM for later use in the function. Now we iterate through the whole sentence again to look for an object that could be connected to this subject-verb pair. We extract the DEPREL from the current word in this second iteration on the sentence and checks if its of the type `'obj'`. If it is of that type we compare the HEAD id of the verb and the object, if they are the same they point at the same subject hence giving a subject-verb-object triple. Then we insert this triple into a dictionary and count the frequency of these triples that occurs in the corpus.

```
def extract_triples(formatted_corpus_dict):
    triples_sv = {}
    for i,sentence in enumerate(formatted_corpus_dict):
        for word in sentence:
            if formatted_corpus_dict[i][word]['DEPREL'].startswith('nsubj'):
                verb = formatted_corpus_dict[i][word]['HEAD'] #verb connected with nsubj
                first = formatted_corpus_dict[i][word]['FORM'].lower()
                second = formatted_corpus_dict[i][verb]['FORM'].lower()
                for word2 in sentence: #Iterate again and look for object
                    obj = formatted_corpus_dict[i][word2]['DEPREL']
                    if obj.startswith('obj'):
                        third = formatted_corpus_dict[i][word2]['FORM'].lower()
                        if verb == formatted_corpus_dict[i][word2]['HEAD']:
                            pair = (first, second, third)
                            if pair in triples_sv:
                                triples_sv[pair] += 1
                            else:
                                triples_sv[pair] = 1
    return triples_sv
```

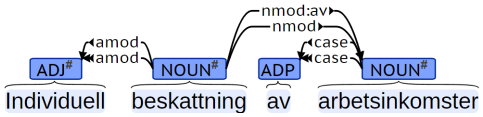
For the submission we extract the three most frequent triples in the different language corpuses. However, one more submission about entities exists also. It works the same as the function above but we must check if both the subject and the object are proper nouns and not set the word to lowercase. The checking is done in the first and second if-statement with the following line

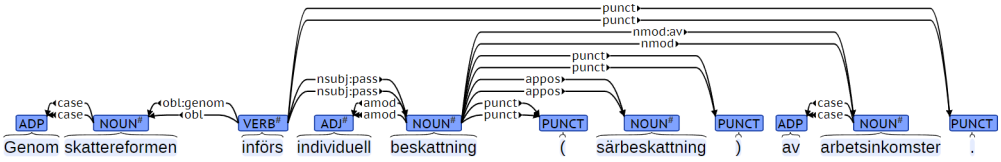
```
if formatted_corpus_dict[i][word]['DEPREL'].startswith('nsubj') and
formatted_corpus_dict[i][word]['UPOS'].startswith('PROPN'):
```

Questions from the lab

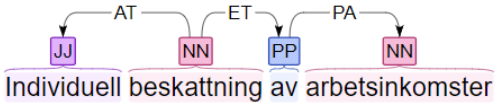
There are a few questions and definitions that should be answered in the report. They are provided here in a list:

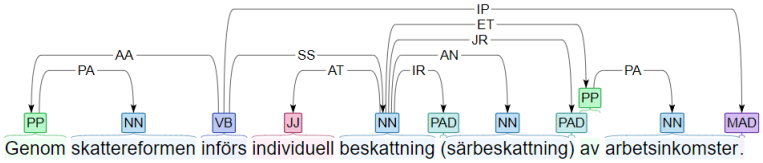
- **Examining the annotation**: The two first sentences in the Swedish corpus are *Individuell beskattning av arbetsinkomster* and *Genom skattereformen införs individuell beskattning (särbeskattning) av arbetsinkomster.*. With the CoNLL-U format we model these two sentences to:





And with a dependency parser we get:





These graphs show basically the same output, however the relational arrows shows different things.

- **Check three languages for nbest**: In this part we were supposed to test three other languages and get the nbest (3) frequent triple words for that language. I have selected Italian, Danish and Norwegian. See the result and the code.

```
[(('individuo', 'ha', 'diritto'), 22), (('sigla', 'significa', 'cosa'), 14),
(('proprietario', 'ha', 'diritto'), 11)]
[(('det', 'drejer', 'sig'), 8), (('de', 'taler', 'tyrkisk'), 3), (('han', 'gjorde', 'det'), 3)]
[(('jeg', 'føler', 'meg'), 7), (('jeg', 'har', 'lyst'), 7), (('jeg', 'har', 'spørsmål'), 6)]
```

```
path_it = ud_path + 'UD_Italian-ISDT/it_isdt-ud-train.conllu'
path_dk = ud_path + 'UD_Danish-DDT/da_ddt-ud-train.conllu'
path_no = ud_path + 'UD_Norwegian-Bokmaal/no_bokmaal-ud-train.conllu'
paths = [path_it, path_dk, path_no]
for language in paths:
    sentences = read_sentences(language)
    formatted_corpus = split_rows(sentences, column_names_u)
    formatted_corpus_dict = convert_to_dict(formatted_corpus)
    triples = extract_triples(formatted_corpus_dict)
    sorted_triples = sorted(triples, key=lambda x: (-triples[x], x))
    frequent_triples = [(triple, triples[triple]) for triple in sorted_triples][:nbest]
    print(frequent_triples)
```

- **Extracting chunks**: The sentence to be focused with is ('Baba', 'remember', 'George'). The sentence of these two proper nouns and the verb is the sentence *Goodwyn dutifully notes that Baba Groom didn't remember George telling drunk stories.* (Complete value of the subject, Complete value of the verb, Complete value of the object) = (Baba Groom, didn't remember ... telling drunk stories, George).

Inducing Knowledge from a Large Scale Lexicalized Relation Resource

In section 6 "Frame Cut" they talk about the "Frame Cut Extraction" where it seems that they extract more frame cuts such as: S-V-O, S-V-O-IO, S-V-P-O etc. (where S - subject, V - verb, O - object, IO - indirect object). This is similar to our assignment to look for pairs where the subject, verb and object (could be indirect object too) is connected to each other.