Лабораторна робота №1
Грищенко Юрій, ПЗС-2
Комп’ютерний морфологічний аналіз, POS-tagging, генерація парадигми слів

1. **POS-tagging.**

Використаємо бібліотеку SpaCy.

Встановлення:

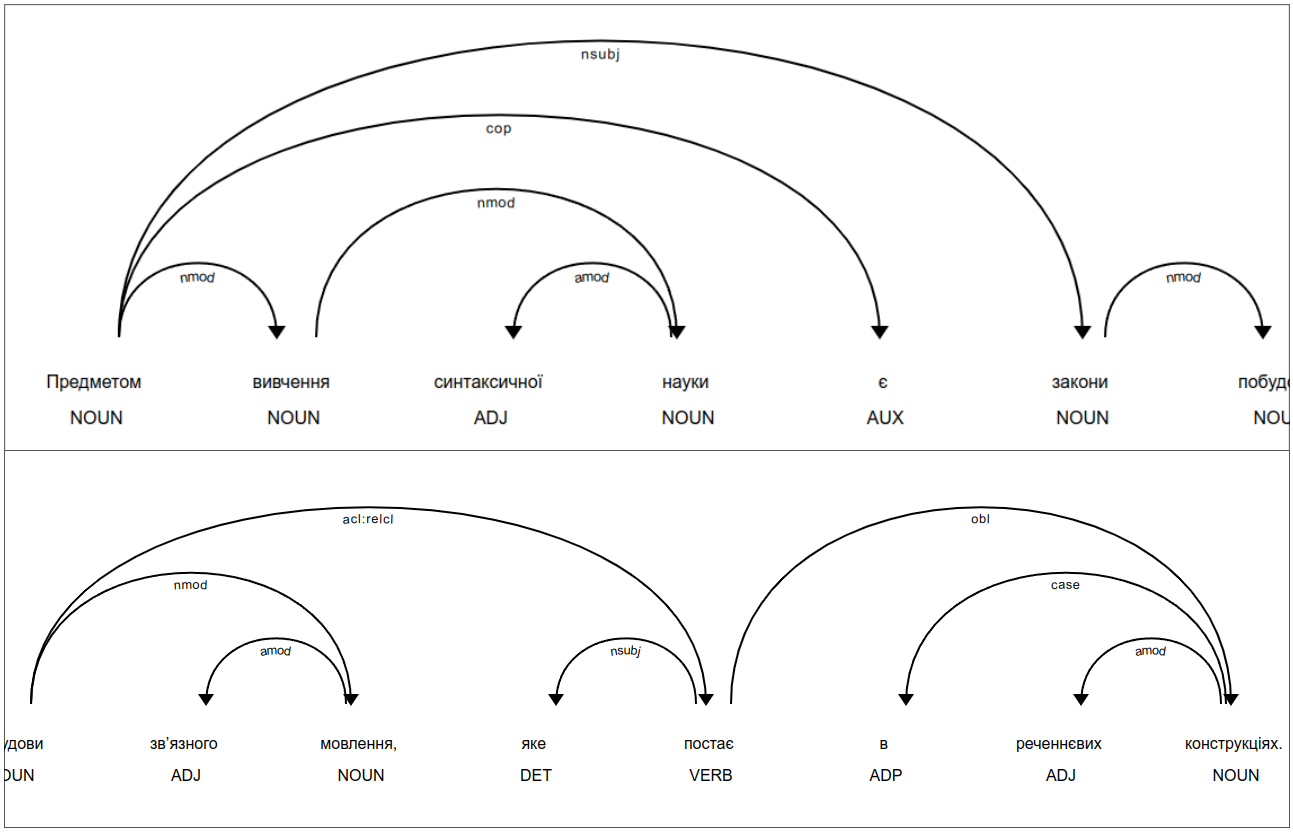
```
python -m venv .env
source .env/bin/activate
pip install -U pip setuptools wheel
pip install -U spacy
python -m spacy download uk_core_news_sm
```

Виконаємо Part-of-speech tagging, з візуалізацією структури речення за допомогою вбудованого візуалізатора displacy:

```
import spacy
from spacy import displacy

nlp = spacy.load("uk_core_news_sm")
doc = nlp("Предметом вивчення синтаксичної науки є закони побудови зв’язного мовлення, яке постає в реченнєвих конструкціях.")
displacy.serve(doc)
```

(рис. розділено на дві частини)



Синтаксична структура речення нас не цікавить в цьому завданні, але бачимо підписи з частинами мовами слів. Частини мови NOUN, ADJ, VERB зрозумілі. За допомогою

```
print(spacy.explain("ADP"))
print(spacy.explain("AUX"))
```

отримуємо «adposition» (прийменник) та «auxillary» (служебне слово; **цікаво, що SpaCy визначив слово «є» як служебне слово, а не як дієслово**)

2. **Морфолічний аналіз.**

Через ту ж бібліотеку SpaCy отримуємо подробиці морфолгічного аналізу:

```
for token in doc:
        print(token.text, token.lemma_, token.pos_, token.tag_,
token.dep_, token.shape_, token.is_alpha, token.is_stop, sep='\t')
```

Text	Lemma	POS	Tag	Dep.	Shape	Alpha	Stop
Предметом	предмет	NOUN	NOUN	ROOT	Xxxxx	True	False
вивчення	вивчення	NOUN	NOUN	nmod	xxxx	True	False
синтаксичної	синтаксичний	ADJ	ADJ	amod	xxxx	True	False
науки	наука	NOUN	NOUN	nmod	xxxx	True	False
є	бути	AUX	AUX	cop	x	True	True
закони	закон	NOUN	NOUN	nsubj	xxxx	True	False
побудови	побудова	NOUN	NOUN	nmod	xxxx	True	False
зв’язного	зв’язного	ADJ	ADJ	amod	xx’xxxx	False	False
мовлення	мовлення	NOUN	NOUN	nmod	xxxx	True	False
,	,	PUNCT	PUNCT	punct	,	False	False
яке	який	DET	DET	nsubj	xxx	True	False
постає	поставати	VERB	VERB	acl:relcl	xxxx	True	False
в	в	ADP	ADP	case	x	True	True
реченнєвих	реченнєвий	ADJ	ADJ	amod	xxxx	True	False
конструкціях	конструкція	NOUN	NOUN	obl	xxxx	True	False
.	.	PUNCT	PUNCT	punct	.	False	False

Де:

- lemma — базова форма форма слова
- tag — детальніший тег POS
- dep — синтаксична залежність
- shape - «форма» слова: великі, малі літери, цифри, пунктуація тощо
- alpha — чи токен складений з літер алфавіту?
- stop — чи є токен стоп-словом?

В документації бачимо, що для англ. мови tag та POS дійсно відрізняються, наприклад, слово «is» має pos = «AUX», tag = «VBZ» (verb, 3rd person singular present). На жаль, українська модель не розмежовує такі теги, слово «є» так і залишається лише «службовим словом».

3. **Генерація парадигми слів**

Використаємо іншу бібліотеку: pymorphy3 (форк pymorphy2)

```
pip install -U pymorphy3-dicts-uk
Requirement already satisfied: pymorphy3-dicts-uk in
./.env/lib/python3.11/site-packages (2.4.1.1.1663094765)
```

Цікаво, тобто SpaCy вже для себе встановив цю бібліотеку як dependency.

```
import pymorphy3
import pprint

pp = pprint.PrettyPrinter()

morph = pymorphy3.MorphAnalyzer(lang="uk")
pp.pprint(morph.parse("Мати"))
```

Розібрали слово «мати»:

```
[Parse(word='мати', tag=OpencorporaTag('NOUN,inan femn,gent'),
normal_form='мата', score=1.0,
methods_stack=((DictionaryAnalyzer(), 'мати', 36, 1),)),
 Parse(word='мати', tag=OpencorporaTag('NOUN,inan plur,nomn'),
normal_form='мата', score=1.0,
methods_stack=((DictionaryAnalyzer(), 'мати', 36, 7),)),
 Parse(word='мати', tag=OpencorporaTag('NOUN,inan plur,accs'),
normal_form='мата', score=1.0,
methods_stack=((DictionaryAnalyzer(), 'мати', 36, 10),)),
 Parse(word='мати', tag=OpencorporaTag('NOUN,inan plur,voct'),
normal_form='мата', score=1.0,
methods_stack=((DictionaryAnalyzer(), 'мати', 36, 13),)),
 Parse(word='мати', tag=OpencorporaTag('VERB,impf infn'),
normal_form='мати', score=1.0,
methods_stack=((DictionaryAnalyzer(), 'мати', 194, 0),)),
 Parse(word='мати', tag=OpencorporaTag('NOUN,inan plur,nomn'),
normal_form='мат', score=1.0,
methods_stack=((DictionaryAnalyzer(), 'мати', 242, 10),)),
 Parse(word='мати', tag=OpencorporaTag('NOUN,inan plur,accs'),
normal_form='мат', score=1.0,
methods_stack=((DictionaryAnalyzer(), 'мати', 242, 13),)),
 Parse(word='мати', tag=OpencorporaTag('NOUN,inan plur,voct'),
normal_form='мат', score=1.0,
methods_stack=((DictionaryAnalyzer(), 'мати', 242, 16),)),
 Parse(word='мати', tag=OpencorporaTag('NOUN,anim femn,nomn'),
normal_form='мати', score=1.0,
methods_stack=((DictionaryAnalyzer(), 'мати', 3475, 0),)),
 Parse(word='мати', tag=OpencorporaTag('NOUN,anim
Arch,femn,accs'), normal_form='мати', score=1.0,
methods_stack=((DictionaryAnalyzer(), 'мати', 3475, 5),)),
 Parse(word='мати', tag=OpencorporaTag('NOUN,anim femn,voct'),
normal_form='мати', score=1.0,
methods_stack=((DictionaryAnalyzer(), 'мати', 3475, 8),))]
```

Дивлячись на normal_form та tag, бачимо що є кілька можливих значень цього слова, деякі з них менш відомі:

- МАТИ, тері, ж. 1. Жінка стосовно дитини, яку вона народила
- МАТИ2, маю, маєш, недок., перех. 1. також без додатка. Уживається на означення того, що комусь належить що-небудь, є його власністю; володіти чимось, посідати щось.
- МАТ1, а, ч . 1. Положення в шаховій партії, при якому король, що перебуває під ударом фігури супротивника, не може захиститися, і партія вважається програною
- МАТА, и, ж. 1. Сплетене із соломи або очерету покривало, підстилка тощо

Оберемо саме дієслово:

```
parse = morph.parse("Мати")
for p in parse:
    if "VERB" in p.tag:
        pp.pprint(p)
```

Маємо:

```
Parse(word='мати', tag=OpencorporaTag('VERB,impf infn'),
normal_form='мати', score=1.0,
methods_stack=((DictionaryAnalyzer(), 'мати', 194, 0),))
```

За допомогою атрибуту Parse.lexeme дістанемо парадигму слова:

```
for p in parse:
    if "VERB" in p.tag:
        lex = p.lexeme
        for l in lex:
            print(l.word, l.tag)
```

Отримуємо:

```
мати VERB,impf infn
мать VERB,impf infn
май VERB,impf sing,2per,impr
маймо VERB,impf plur,1per,impr
майте VERB,impf plur,2per,impr
маю VERB,impf sing,1per,pres
маєш VERB,impf sing,2per,pres
має VERB,impf sing,3per,pres
маємо VERB,impf plur,1per,pres
маєм VERB,impf plur,1per,pres
маєте VERB,impf plur,2per,pres
мають VERB,impf plur,3per,pres
матиму VERB,impf sing,1per,futr
матимеш VERB,impf sing,2per,futr
матиме VERB,impf sing,3per,futr
матимем VERB,impf plur,1per,futr
матимемо VERB,impf plur,1per,futr
```

```
матимете VERB,impf plur,2per,futr
матимуть VERB,impf plur,3per,futr
мав VERB,impf masc,past
мала VERB,impf femn,past
мало VERB,impf neut,past
мали VERB,impf plur,past
```

Для цього слова результат задовільний, співпадає з https://slovnyk.ua/index.php?swrd=мати, і при цьому навіть наводить альтернативні форми «мать» та «маєм».

```
lex = morph.parse("майно")[0].lexeme
for l in lex:
    print(l.word, l.tag)
```

Дає результат:

```
майно NOUN,neut,inan nomn
майна NOUN,neut,inan gent
майну NOUN,neut,inan datv
майно NOUN,neut,inan accs
майном NOUN,neut,inan ablt
майні NOUN,neut,inan loct
майну NOUN,neut,inan loct
майно NOUN,neut,inan voct
```

Також правильно, слід зазначити, що для місцевого відмінку не використовуються прийменники «в»/«на».

Для більш рідкісного слова також результат задовільний, навіть наводяться альтернативні форми «(на) двовалентному», «(на) двовалентнім».

```
двовалентний ADJF masc,nomn
двовалентного ADJF masc,gent
двовалентному ADJF masc,datv
двовалентного ADJF masc,accs
двовалентний ADJF masc,accs
двовалентним ADJF masc,ablt
двовалентнім ADJF masc,loct
двовалентному ADJF masc,loct
двовалентний ADJF masc,voct
двовалентна ADJF femn,nomn
двовалентної ADJF femn,gent
двовалентній ADJF femn,datv
двовалентну ADJF femn,accs
двовалентною ADJF femn,ablt
двовалентній ADJF femn,loct
двовалентна ADJF femn,voct
двовалентне ADJF neut,nomn
двовалентного ADJF neut,gent
двовалентному ADJF neut,datv
двовалентне ADJF neut,accs
```

```
двовалентним ADJF neut,ablt
двовалентнім ADJF neut,loct
двовалентному ADJF neut,loct
двовалентне ADJF neut,voct
двовалентні ADJF plur,nomn
двовалентних ADJF plur,gent
двовалентним ADJF plur,datv
двовалентних ADJF plur,accs
двовалентні ADJF plur,accs
двовалентними ADJF plur,ablt
двовалентних ADJF plur,loct
двовалентні ADJF plur,voct
```

Вигадали слово «квапотливкий», такого слова не існує в укр. мові (і Google видає 0 результатів). Цікаво, що таке слово теж провідміняли:

квапотливкий ADJF masc,nomn
квапотливкого ADJF masc,gent
квапотливкому ADJF masc,datv
квапотливкого ADJF masc,accs
квапотливкий ADJF masc,accs
квапотливким ADJF masc,ablt
квапотливкім ADJF masc,loct
квапотливкому ADJF masc,loct
квапотливкий ADJF masc,voct
квапотливка ADJF femn,nomn
квапотливкої ADJF femn,gent
квапотливкій ADJF femn,datv
квапотливку ADJF femn,accs
квапотливкою ADJF femn,ablt
квапотливкій ADJF femn,loct
квапотливка ADJF femn,voct
квапотливке ADJF neut,nomn
квапотливкого ADJF neut,gent
квапотливкому ADJF neut,datv
квапотливке ADJF neut,accs
квапотливким ADJF neut,ablt
квапотливкім ADJF neut,loct
квапотливкому ADJF neut,loct
квапотливке ADJF neut,voct
квапотливкі ADJF plur,nomn
квапотливких ADJF plur,gent
квапотливким ADJF plur,datv
квапотливких ADJF plur,accs
квапотливкі ADJF plur,accs
квапотливкими ADJF plur,ablt
квапотливких ADJF plur,loct
квапотливкі ADJF plur,voct

Загалом, pymorphy3 виконав своє завдання задовільно, але SpaCy видав помилку (незрозуміло, чому слово «є» визначили, як допоміжне слово, при тому, що SpaCy спирається саме на pymorphy3, яка визначає це слово як VERB).