

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ
им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра «Систем обработки информации и управления»

ОТЧЕТ

Домашнее задание
по дисциплине «Методы машинного обучения»

Тема: «Предобработка текста»

ИСПОЛНИТЕЛЬ: Кожуро Б.Е. ФИО
группа ИУ5-21М
подпись
" _ " _ 2024 г.

ПРЕПОДАВАТЕЛЬ: Гапанюк Ю.Е. ФИО
подпись
" _ " _ 2024 г.

Москва - 2024

Задание

Для произвольного предложения или текста решите следующие задачи:

1. Токенизация.
2. Частеречная разметка.
3. Лемматизация.
4. Выделение (распознавание) именованных сущностей.
5. Разбор предложения.

Выполнение

ДЗ


Кожуро Б.Е.

Для произвольного предложения или текста решите следующие задачи:

1. Токенизация.
2. Частеречная разметка.
3. Лемматизация.
4. Выделение (распознавание) именованных сущностей.
5. Разбор предложения.

В качестве текста выберем банальный пример - случайный фрагмент из "Преступления и наказания" Достоевского.

токенизация

Ввод [1]: 

Ввод [2]:

```
import nltk
nltk.download('punkt')
from nltk import tokenize
dir(tokenize)[:18]
```

```
[nltk_data] Downloading package punkt to
[nltk_data]   C:\Users\ksarb\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[2]: ['BlanklineTokenizer',
'LegalitySyllableTokenizer',
'LineTokenizer',
'MWETokenizer',
'NLTKWordTokenizer',
'PunktSentenceTokenizer',
'RegexTokenizer',
'ReppTokenizer',
'SEExprTokenizer',
'SpaceTokenizer',
'StanfordSegmenter',
'SyllableTokenizer',
'TabTokenizer',
'TextTilingTokenizer',
'ToktokTokenizer',
'TreebankWordDetokenizer',
'TreebankWordTokenizer',
'TweetTokenizer']

Ввод [16]:

```
nltk_tk_1 = nltk.WordPunctTokenizer()
nltk_tk_1.tokenize(text1)[0:20]
```

Out[16]: ['он',
'не',
'мог',
'выразить',
'ни',
'словами',
',',
'и',
'ни',
'восклицаниями',
'своего',
'волнения',
'.',
'Чувство',
'бесконечного',
'отвращения',
',',
'начинавшее',
'давить',
'и']

```
Ввод [17]: nltk.tk_sents = nltk.tokenize.sent_tokenize(text1)
print(len(nltk.tk_sents))
nltk.tk_sents
```

15

```
Out[17]: ['Но он не мог выразить ни словами, ни восклицаниями своего волнения.',
'Чувство бесконечного отвращения, начинавшее давить и мутить его сердце еще в то время, как он только шел к старухе, дости-
гло теперь такого размера и так ярко выяснилось, что он не знал, куда деться от тоски своей.',
'Он шел по тротуару как пьяный, не замечая прохожих и сталкиваясь с ними, и опомнился уже в следующей улице.',
'Оглядевшись, он заметил, что стоит подле распивочной, в которую вход был с тротуара по лестнице вниз, в подвальный эта-
ж.',
'Из дверей, как раз в эту минуту, выходили двое пьяных и, друг друга поддерживая и ругая, взбирались на улицу.',
'Долго не думая, Раскольников тотчас же спустился вниз.',
'Никогда до сих пор не входил он в распивочные, но теперь голова его кружилась, и к тому же палящая жажда томила его.',
'Ему захотелось выпить холодного пива, тем более что внезапную слабость свою он относил и к тому, что был голоден.',
'Он уселся в темном и грязном углу, за липким столиком, спросил пива и с жадностью выпил первый стакан.',
'Тотчас же всё отлегло, и мысли его прояснили.',
'«Всё это вздор, — сказал он с надеждой, — и нечем тут было смущаться!»,
'Просто физическое расстройство!',
'Один какой-нибудь стакан пива, кусок сухаря, — и вот, в один миг, крепнет ум, яснееет мысль, твердеют намерения!',
'Тьфу, какое всё это ничтожество!..» Но, несмотря на этот презрительный плевок, он глядел уже весело, как будто внезапно о-
свободясь от какого-то ужасного бремени, и дружелюбно окинул глазами присутствующих.',
'Но даже и в эту минуту он отдаленно предчувствовал, что вся эта восприимчивость к лучшему была тоже болезненная.']
```

В Spacy

```
Ввод [19]: from spacy.lang.ru import Russian
import spacy
nlp = spacy.load('ru_core_news_sm')
spacy_text1 = nlp(text1)
for t in spacy_text1[:20]:
    print(t)
```

```
Но
он
не
мог
выразить
ни
словами
,
ни
восклицаниями
своего
волнения
.
Чувство
бесконечного
отвращения
,
начинавшее
давить
и
```

В результате токенизации текст разбивается на токены - атомарные единицы. В основном используют либо слова, либо предложения, либо абзацы.

Частеречная разметка (Part-Of-Speech tagging, POS-tagging)

```
Ввод [20]: for token in spacy_text1[0:20]:
print('{} - {} - {}'.format(token.text, token.pos_, token.dep_))
```

```
Но - CONJ - cc
он - PRON - nsubj
не - PART - advmod
мог - VERB - ROOT
выразить - VERB - xcomp
ни - PART - cc
словами - NOUN - obl
, - PUNCT - punct
ни - CONJ - cc
восклицаниями - NOUN - conj
своего - DET - det
волнения - NOUN - nmod
. - PUNCT - punct
Чувство - NOUN - nsubj
бесконечного - ADJ - amod
отвращения - NOUN - nmod
, - PUNCT - punct
начинавшее - VERB - acl
давить - VERB - xcomp
и - CONJ - cc
```

```

Ввод [27]: from razdel import tokenize, sentenize
            from navec import Navec
            from slovnet import Morph

            def n_sentenize(text):
                n_sen_chunk = []
                for sent in sentenize(text):
                    tokens = [_.text for _ in tokenize(sent.text)]
                    n_sen_chunk.append(tokens)
                return n_sen_chunk

            n_sen_chunk_1 = n_sentenize(text1)

            navec = Navec.load('navec/navec_news_v1_1B_250K_300d_100q.tar')
            n_morph = Morph.load('slovnet/slovnet_morph_news_v1.tar', batch_size=4)
            morph_res = n_morph.navec(navec)

            def print_pos(markup):
                for token in markup.tokens:
                    print('{} - {}'.format(token.text, token.tag))

            n_text1_markup = list(_ for _ in n_morph.map(n_sen_chunk_1))
            for x in n_text1_markup[0:1]:
                print_pos(x)

```

```

Ho - CCONJ
он - PRON|Case=Nom|Gender=Masc|Number=Sing|Person=3
не - PART|Polarity=Neg
мог - VERB|Aspect=Imp|Gender=Masc|Mood=Ind|Number=Sing|Tense=Past|VerbForm=Fin|Voice=Act
выразить - VERB|Aspect=Perf|VerbForm=Inf|Voice=Act
ни - CCONJ|Polarity=Neg
словами - NOUN|Animacy=Inan|Case=Ins|Gender=Neut|Number=Plur
, - PUNCT
ни - CCONJ|Polarity=Neg
восклицаниями - VERB|Aspect=Imp|VerbForm=Inf|Voice=Act
своего - DET|Case=Gen|Gender=Neut|Number=Sing
волнения - NOUN|Animacy=Inan|Case=Gen|Gender=Neut|Number=Sing
. - PUNCT

```

Частеречная разметка позволяет исследовать все токены-слова отдельно. В отличие от spacey, razdel не только категоризирует часть речи, но и предоставляет информацию о склонении, падеже, завершённости формы глагола, лице и поле. Разметка не идеальная: слово "подле" - предлог, было определено как подле - VERB|Aspect=Perf|VerbForm=Inf|Voice=Act.

Лемматизация

```

Ввод [28]: for token in spacy_text1[0:20]:
            print(token, token.lemma, token.lemma_)

```

```

Ho 14653780147686393572 но
он 7004339974413567607 он
не 5319710824202933802 не
мог 14329395112709808155 мочь
выразить 1949575889363403232 выразить
ни 10089292569908228859 ни
словами 1386213856741127517 слово
, 2593208677638477497 ,
ни 10089292569908228859 ни
восклицаниями 13127012205942603523 восклицание
своего 12292881551881158589 свой
волнения 6026467723303534222 волнение
. 12646065887601541794 .
Чувство 3912985599449811266 чувство
бесконечного 2408103873523094030 бесконечный
отвращения 4523946167221646868 отвращение
, 2593208677638477497 ,
начинавшее 382883286054309220 начинать
давить 12609501716882608852 давить
и 15015917632809974589 и

```

```

Ввод [9]: from natasha import Doc, Segmenter, NewsEmbedding, NewsMorphTagger, MorphVocab

            def n_lemmatize(text):
                emb = NewsEmbedding()
                morph_tagger = NewsMorphTagger(emb)
                segmenter = Segmenter()
                morph_vocab = MorphVocab()
                doc = Doc(text)
                doc.segment(segmenter)
                doc.tag_morph(morph_tagger)
                for token in doc.tokens:
                    token.lemmatize(morph_vocab)
                return doc

```

```
Ввод [29]: n_doc1 = n_lemmatize(text1)
{_.text: _.lemma for _ in n_doc1.tokens[0:20]}
```

```
Out[29]: {'Но': 'но',
'он': 'он',
'не': 'не',
'мог': 'мочь',
'выразить': 'выразить',
'ни': 'ни',
'словами': 'слово',
',': ',',
'восклицаниями': 'восклицаниями',
'своего': 'свой',
'волнения': 'волнение',
'.': '.',
'Чувство': 'чувство',
'бесконечного': 'бесконечный',
'отвращения': 'отвращение',
'начинавшее': 'начинать',
'давить': 'давить',
'и': 'и'}
```

Лемматизация приведет слова-токены к их нормальной форме. Теперь возможно проводить частотный и другие анализы композиции текста - поскольку все слова приведены в одинаковое состояние.

Выделение (распознавание) именованных сущностей.

```
Ввод [11]: for ent in spacy_text1.ents:
           print(ent.text, ent.label_)
```

Раскольников PER

```
Ввод [12]: from spacy import displacy
displacy.render(spacy_text1, style='ent', jupyter=True)
```

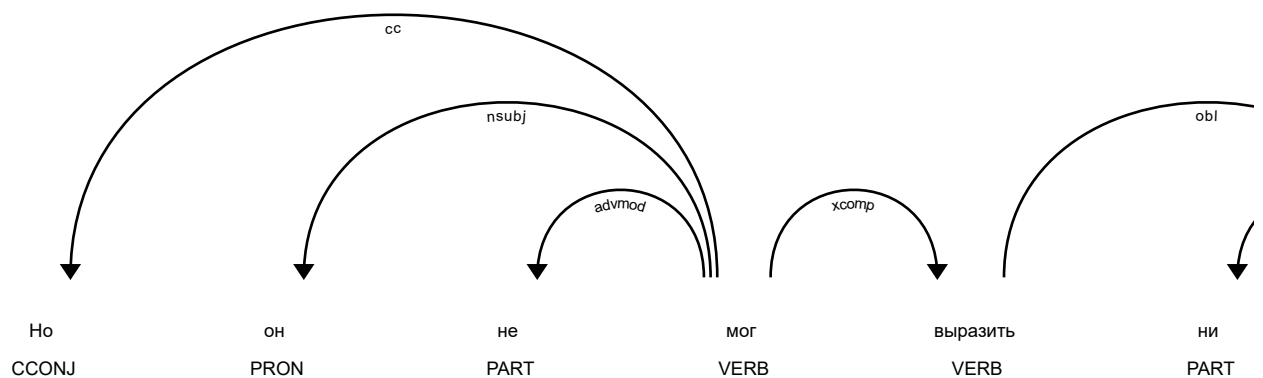
Но он не мог выразить ни словами, ни восклицаниями своего волнения. Чувство бесконечного отвращения, начинавшее давить и мутить его сердце еще в то время, как он только шел к старухе, достигло теперь такого размера и так ярко выяснилось, что он не знал, куда деться от тоски своей. Он шел по тротуару как пьяный, не замечая прохожих и сталкиваясь с ними, и опомнился уже в следующей улице. Оглядевшись, он заметил, что стоит подле распивочной, в которую вход был с тротуара по лестнице вниз, в подвальный этаж. Из дверей, как раз в эту минуту, выходили двое пьяных и, друг друга поддерживая и ругая, взбирались на улицу. Долго не думая, Раскольников PER тотчас же спустился вниз. Никогда до сих пор не входил он в распивочные, но теперь голова его кружилась, и к тому же палящая жажда томила его. Ему захотелось выпить холодного пива, тем более что внезапную слабость свою он относил и к тому, что был голоден. Он уселся в темном и грязном углу, за липким столиком, спросил пива и с жадностью выпил первый стакан. Тотчас же всё отлегло, и мысли его прояснели. «Всё это вздор, — сказал он с надеждой, — и нечем тут было смущаться! Просто физическое расстройство! Один какой-нибудь стакан пива, кусок сухаря, — и вот, в один миг, крепнет ум, яснее мысль, твердеют намерения! Тьфу, какое всё это ничтожество!..» Но, несмотря на этот презрительный плевок, он глядел уже весело, как будто внезапно освободясь от какого-то ужасного бремени, и дружелюбно окинул глазами присутствующих. Но даже и в эту минуту он отдаленно предчувствовал, что вся эта восприимчивость к лучшему была тоже болезненная.

Распознавание сущностей позволяет выделить в тексте значимых актёров, локации и другие атрибуты, которые могут характеризовать текст с точки зрения контекста/

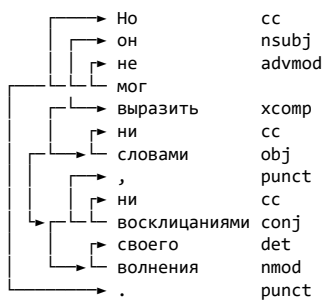
разбор предложения

```
Ввод [30]: from spacy import displacy
```

```
Ввод [31]: displacy.render(spacy_text1, style='dep', jupyter=True)
```



```
Ввод [32]: from natasha import NewsSyntaxParser
emb = NewsEmbedding()
syntax_parser = NewsSyntaxParser(emb)
n_doc1.parse_syntax(syntax_parser)
n_doc1.sents[0].syntax.print()
```



Для анализа токенов-предложений значимым будет создание графа или дерева со структурой предложения, поскольку свойства самих слов анализируем на прошлых этапах

Вывод:

В ходе выполнения домашнего задания была проведена предобработка текста, которая подготовила его к разнообразным типам анализа, основанным как на анализе отдельных слов, так и полных предложений.