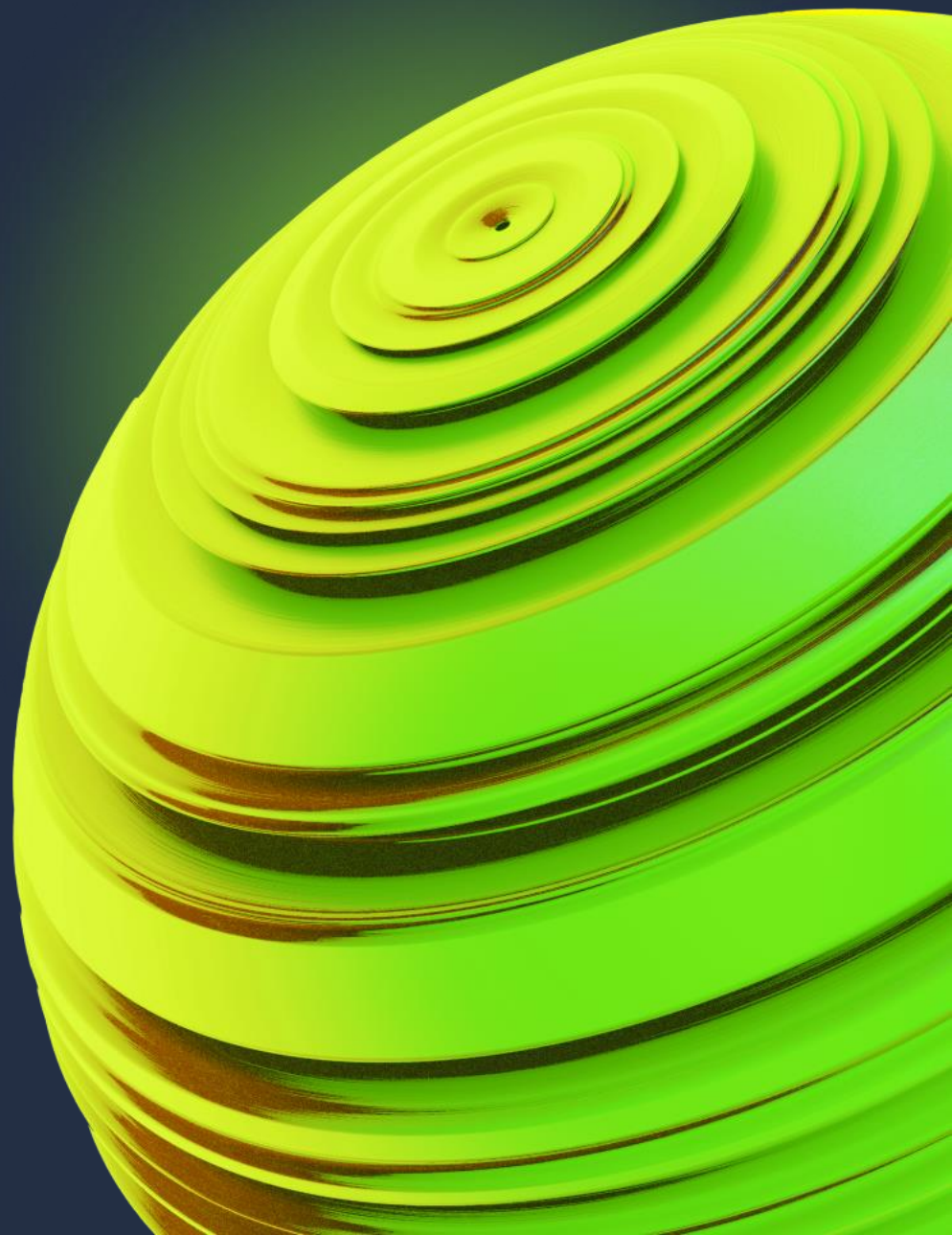
ИНСТИТУТ ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ
УНИВЕРСИТЕТА ИННОПОЛИС

УНИВЕРСИТЕТ ИННОПОЛИС

Предобработка текста nltk, pymorphy3, natasha

Корнеева Елена
e.korneeva@innopolis.ru, https://t.me/Allyonzy

2024



План занятия (лекция + семинар)

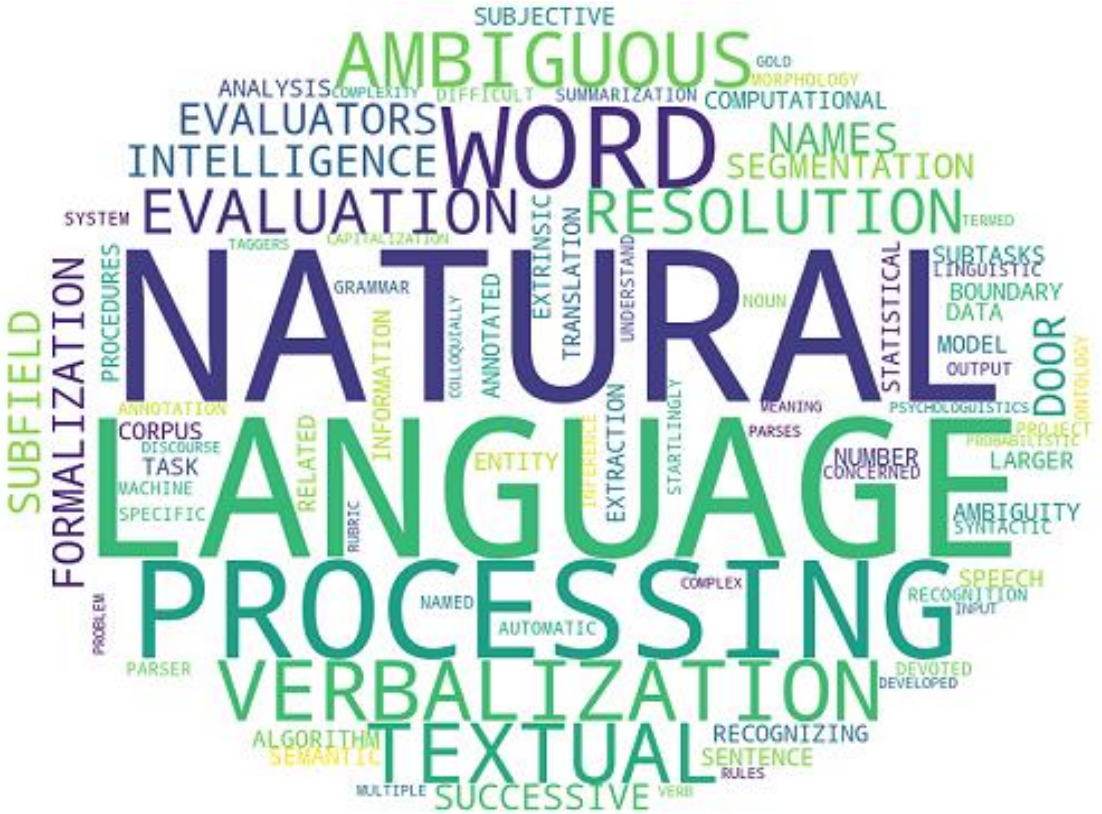
1. NLP vs Компьютерная лингвистика
2. Задача и этапы работы с текстом
3. Natural Language Toolkit: токенизация, удаление стоп-слов, приведение к нижнему регистру, лемматизация
4. pyMorphy3
5. natasha

Что такое Natural Language Processing (NLP)

NLP – одно из направлений *искусственного интеллекта*, которое работает

- с анализом,
- пониманием и
- генерацией живых языков,

для того, чтобы **взаимодействовать** с компьютерами и устно, и письменно, *используя естественные языки* вместо компьютерных.



NLP vs Компьютерная лингвистика

Компьютерная лингвистика тесно связана с NLP. Рассмотрим различия:

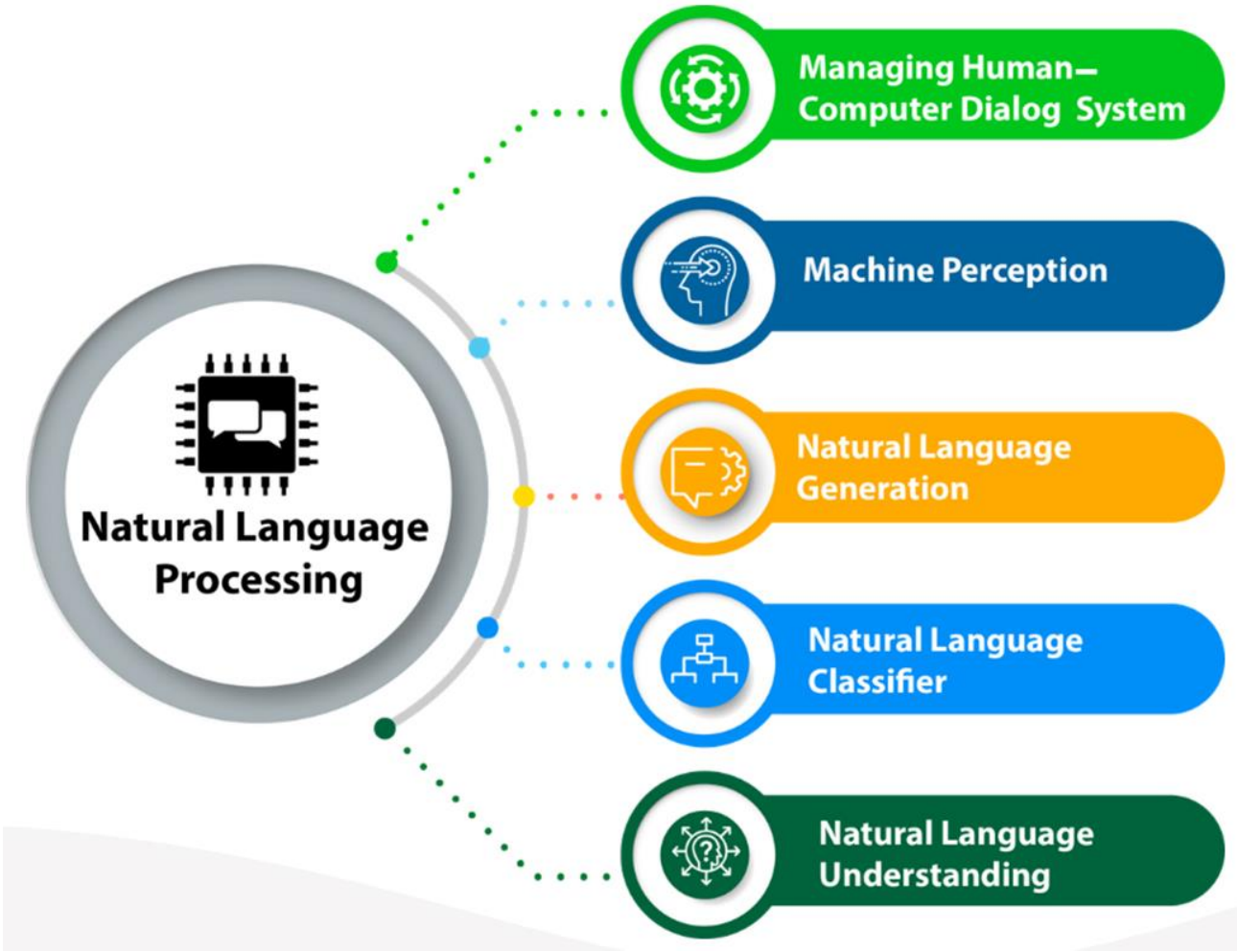
Компьютерная лингвистика

область знаний, которая занимается компьютерным моделированием владения естественным языком и решением прикладных задач автоматической обработки текстов и речи (*ответ на вопросы лингвистов*)

Обработка естественного языка (NLP)

пересечение машинного обучения и математической лингвистики, которое предназначено для решения инженерных проблем *(ответ на вопросы людей нелингвистов)*

Применение NLP



- Управление человеческими ресурсами (диалоговые системы)
- Машинное восприятие — способность компьютерной системы интерпретировать данные таким же образом, как люди используют свои чувства для связи с окружающим миром.
- Генерация естественного языка (автореферат)
- Классификация текста (поисковые системы, определение фэйков)
- Понимание естественного языка (переводчики)

Основные задачи работы с текстом NLP

- ***Токенизация (Tokenization)***
  Разделение текста на токены (слова, предложения или другие), чтобы сделать текст более структурированным для последующей обработки.

- ***Предобработка текста (Text Preprocessing)***
  Очистка и преобразование текста перед анализом. Это может включать удаление стоп-слов, пунктуации, приведение к нижнему регистру, лемматизацию и стемминг.

- ***Парсинг (Parsing)***
  Анализ синтаксической структуры предложений для определения связей между словами и дерева разбора.

- ***Частеречная разметка (Part-of-Speech Tagging):***
  Присвоение частям речи (существительное, глагол, прилагательное и т.д.) для каждого слова в предложении.

- ***Извлечение информации (Information Extraction):***
  Извлечение структурированных сущностей из текста, таких как имена, даты, местоположения и т.д.

- ***Извлечение фактов и связей (Relation Extraction):***
  Определение связей и отношений между сущностями в тексте.

Основные задачи работы с текстом NLP

- ***Разрешение семантической неоднозначности (Word Sense Disambiguation):***
  Определение правильного значения слова в контексте, когда у слова есть несколько возможных смыслов.

- ***Обработка естественного языка с помощью глубокого обучения (Deep Learning NLP):***
  Применение глубоких нейронных сетей для различных задач NLP, таких как машинный перевод, анализ тональности, вопросно-ответные системы и т.д.

- ***Анализ эмоциональной окраски (Sentiment Analysis):***
  Определение эмоциональной окраски текста (негативной, позитивной, нейтральной).

- ***Генерация текста (Text Generation):***
  Создание текстовых данных с помощью алгоритмов, которые могут автоматически составлять предложения и тексты.

Этапы работы с текстом NLP

Этап 1. Обработка текста (очистка и синтаксис)
- Токенизация (разделение исходного текста на токены).
- Поиск частей речи
- Лемматизация (приведение слов к нормальной словарной форме)
- Удаление «стоп слов»
- Тематическое моделирование
- Поиск устойчивых словосочетаний (n-gramm)

Этап 2. Векторизация текста (статистический анализ, глубокое обучение, семантический анализ)
- Кодирование данных помощью методологии TF-IDF
- Поиск близких по смыслу слов с помощью векторной модели word2vec, bert, обработка rnn сетями

Этап 3. Выбор модели классификации и обучение (выбор алгоритма для работы)

Natural Language Toolkit

Пакет библиотек и программ для символьной и статистической обработки естественного языка. Содержит графические представления и примеры данных. Поддерживает работу с множеством языков, в том числе, русским

- Наиболее известная и многофункциональная библиотека для NLP;
- Большое количество сторонних расширений;
- Быстрая токенизация предложений;
- Поддерживается множество языков

- Медленная;
- Сложная в изучении и использовании;
- Работает со строками;
- Не использует нейронные сети;
- Нет встроенных векторов слов.

Токенизация (Tokenization) NLTK

```
import nltk
nltk.download('punkt')

from nltk.tokenize import word_tokenize, sent_tokenize

text = "Пример текста для токенизации. Токенизация
разделит его на предложения и слова."
sentences = sent_tokenize(text)
words = word_tokenize(text)

print("Предложения:", sentences)
print("Слова:", words)
```

Токенизация (иногда – сегментация) по предложениям или словам *– это процесс разделения письменного языка на предложения-компоненты или слова-компоненты. Идея выглядит довольно простой.* В английском и некоторых других языках мы можем вычленять предложение каждый раз, когда находим определенный знак пунктуации – точку. В английском и многих других языках, использующих ту или иную версию латинского алфавита, пробел – это неплохой разделитель слов.

Удаление стоп-слов (Stop words removal) NLTK

```
nltk.download('stopwords')

from nltk.corpus import stopwords

stop_words = set(stopwords.words('russian'))
filtered_words = [word for word in words if
word.lower() not in stop_words]

print("Слова после удаления стоп-слов:",
```

Стоп-слова *– это слова, которые выкидываются из текста до/после обработки текста Когда мы применяем машинное обучение к текстам, такие слова могут добавить много шума, поэтому необходимо избавляться от нерелевантных слов.*

Приведение к нижнему регистру (Lowercasing) NLTK

```
lowercase_words = [word.lower() for word in filtered_words]

print("Слова после приведения к нижнему регистру:",
lowercase_words)
```

Приведение к нижнему регистру *– это процесс преобразования текста таким образом, чтобы все символы верхнего регистра были заменены символами нижнего регистра.*

Такой процесс полезен для обработки и анализа текста, так как он упрощает сравнение и поиск информации.

Лемматизация (Lemmatization) NLTK

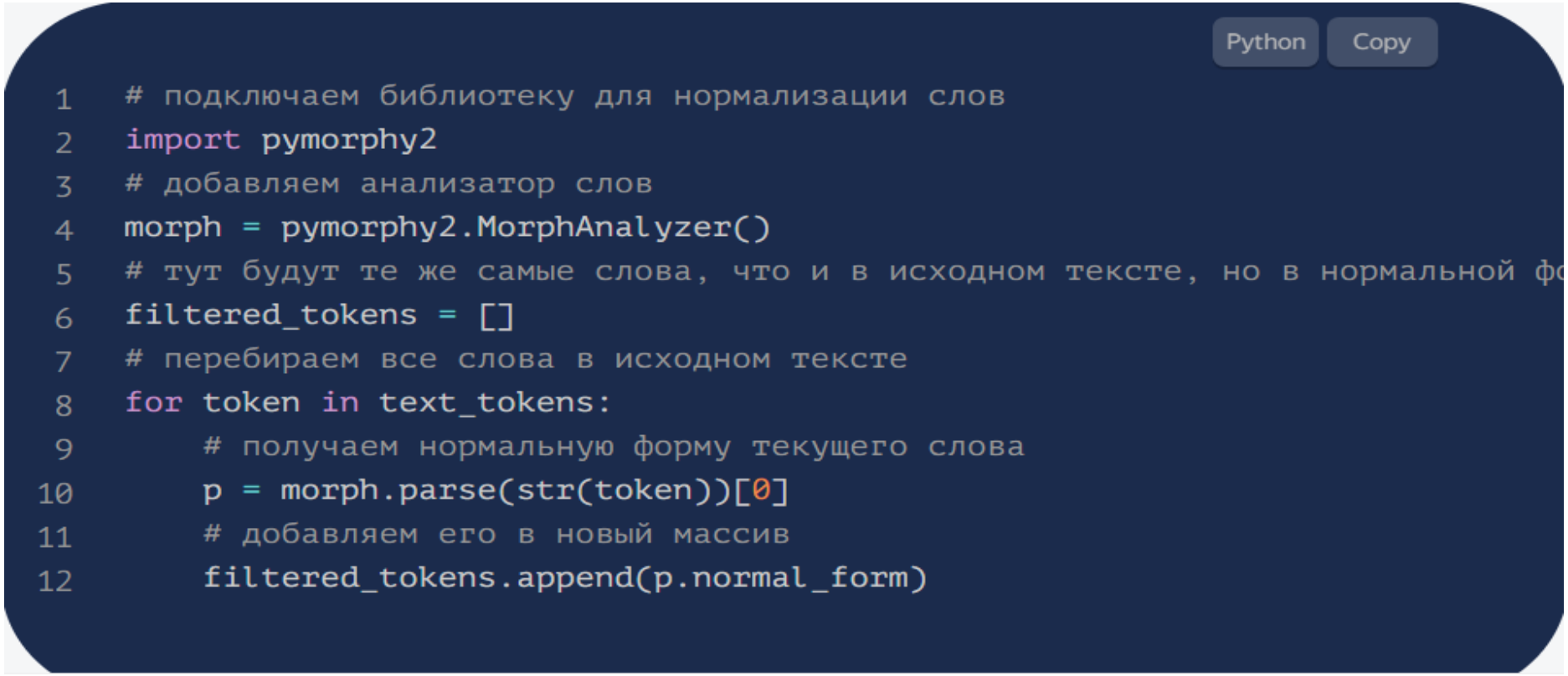
```
nltk.download('wordnet')
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()
lemmatized_words = [lemmatizer.lemmatize(word) for word in
lowercase_words]

print("Слова после лемматизации:", lemmatized_words)
```

Лемматизация *– это более тонкий процесс, который использует словарь и морфологический анализ, чтобы в итоге привести слово к его канонической форме – лемме.*

pyMorphy2 и pyMorphy3



Python Copy

```python
# подключаем библиотеку для нормализации слов
import pymorphy2
# добавляем анализатор слов
morph = pymorphy2.MorphAnalyzer()
# тут будут те же самые слова, что и в исходном тексте, но в нормальной фо
filtered_tokens = []
# перебираем все слова в исходном тексте
for token in text_tokens:
    # получаем нормальную форму текущего слова
    p = morph.parse(str(token))[0]
    # добавляем его в новый массив
    filtered_tokens.append(p.normal_form)
```

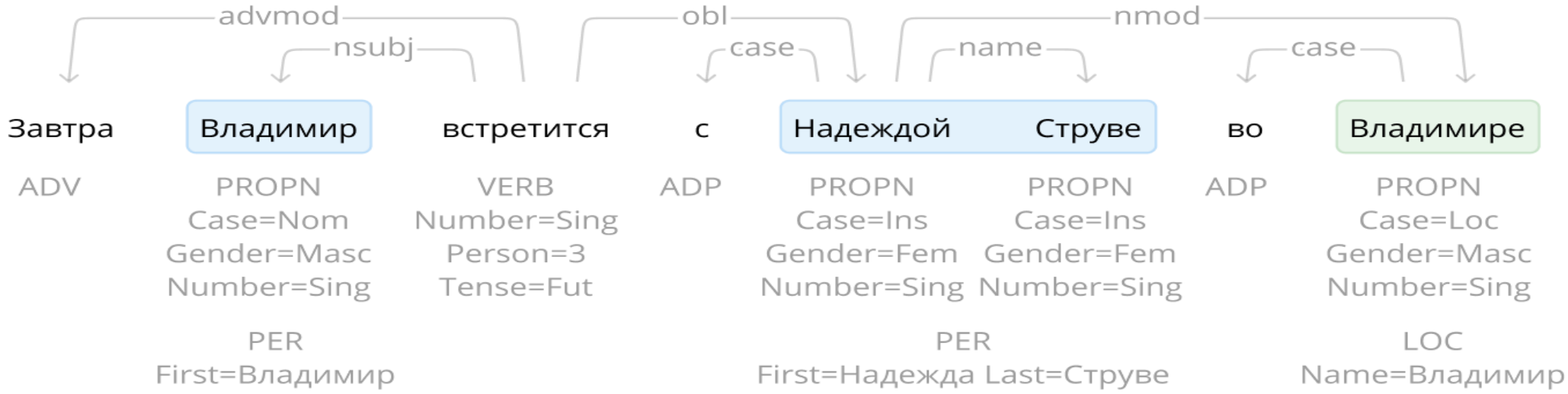
pymorphy2** - морфологический анализатор для русского языка на языке Python (использует OpenCorpora). Более новая версия **pymorphy3

pyMorphy3. Склонения по падежам

```
word = "стол"
p = morph.parse(word)[0]
print(p.inflect({'nomn'}).word)  # именительный падеж
print(p.inflect({'datv'}).word)  # дательный падеж
print(p.inflect({'accs'}).word)  # винительный падеж
print(p.inflect({'ablt'}).word)  # творительный падеж
print(p.inflect({'loct'}).word)  # предложный падеж
```

pymorphy2** - морфологический анализатор для русского языка на языке Python (использует OpenCorpora). Более новая версия **pymorphy3

Проект Natasha

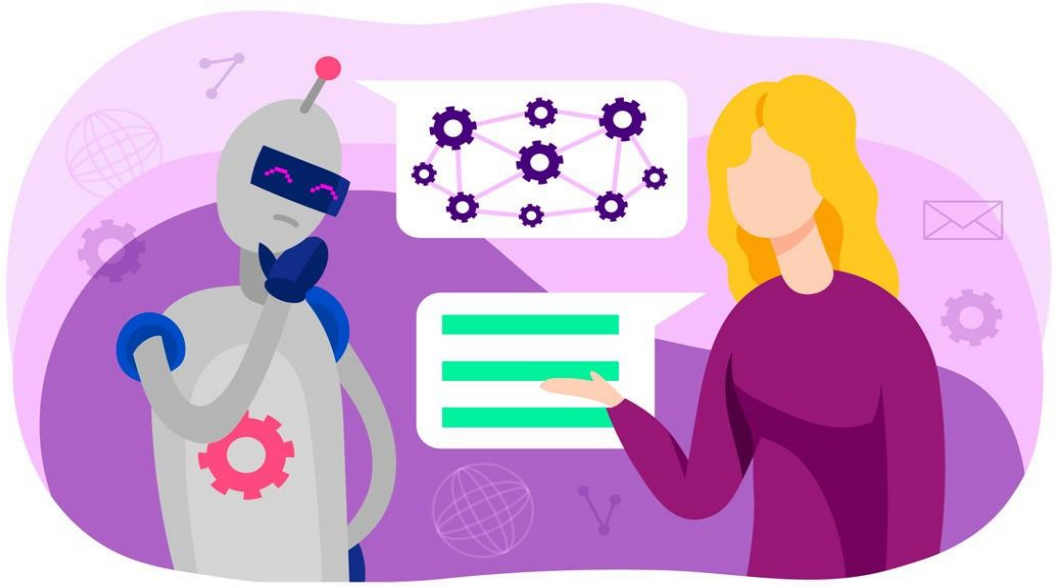


Библиотека Natasha *решает все базовые задачи обработки естественного русского языка: сегментация на токены и предложения, морфологический и синтаксический анализ, лемматизация, извлечение именованных сущностей.*

В проекте 9 репозиториев, библиотека Natasha объединяет их под одним интерфейсом: corus, navec, nerus, razdel, slovnet, yargy, natasha, naeval, ipymarkup. Обратить внимание на решение задачи NER для русского языка на рисунке

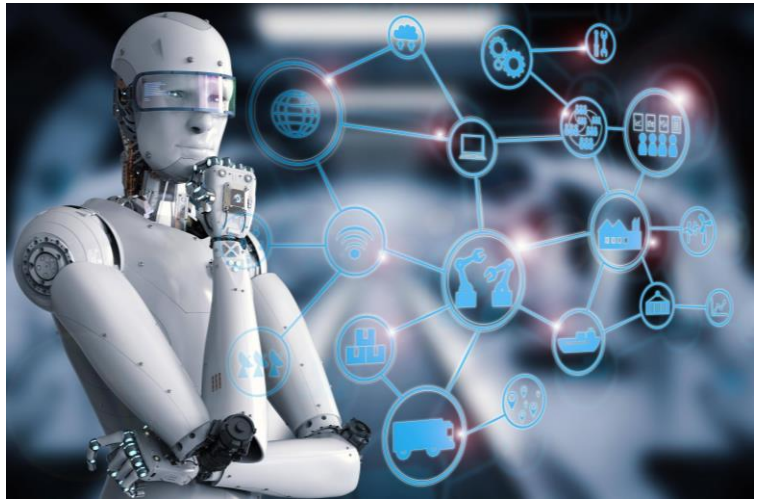
ДЕМОНСТРАЦИЯ

Пример работы с библиотеками
для предобработки текста



Полезные ссылки

1. Преобразование текстовых данных и работа с ними в Python. URL: https://clck.ru/3DNGPf
2. Основы Natural Language Processing для текста. URL: https://habr.com/ru/companies/Voximplant/articles/446738/
3. Проект Natasha. Набор качественных открытых инструментов. URL: https://habr.com/ru/articles/516098/
4. Библиотека Наташа. URL: https://natasha.github.io/
5. Репозиторий курса по анализу текстов и обработке естественного языка на ФИВТ МФТИ. URL: https://github.com/andybelov/nlp_mipt
6. Большакова Е.И. АВТОМАТИЧЕСКАЯ ОБРАБОТКА ТЕКСТОВ НА ЕСТЕСТВЕННОМ ЯЗЫКЕ И КОМПЬЮТЕРНАЯ ЛИНГВИСТИКА. URL: https://clck.ru/3DNGSF
7. Морфологический анализатор pymorphy2. URL: https://pymorphy2.readthedocs.io/en/stable/



ИНСТИТУТ ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ
УНИВЕРСИТЕТА ИННОПОЛИС

Спасибо за внимание!

Telegram

E-mail

Контакты

Корнеева Елена

https://t.me/Allyonzy

e.korneeva@innopolis.ru