Содержание

[Первая часть 1](#_Toc43238293)

[База данных 1](#_Toc43238294)

[Парсер 1](#_Toc43238295)

[Web-интерфейс 4](#_Toc43238296)

[Вторая часть 4](#_Toc43238297)

[Томита-парсер 4](#_Toc43238298)

[Word2Vec 5](#_Toc43238299)

[Третья часть 7](#_Toc43238300)

[Анализ тональности 7](#_Toc43238301)

[Руководство системного администратора 8](#_Toc43238302)

[Установка библиотек для Python 8](#_Toc43238303)

[Установка и настройка MongoDB 9](#_Toc43238304)

[Установка томита-парсера 10](#_Toc43238305)

[Руководство пользователя 10](#_Toc43238306)

[Работа с web-интерфейсом: 10](#_Toc43238307)

[Использование парсера: 11](#_Toc43238308)

[Использование модуля томита-парсера: 11](#_Toc43238309)

[Использование word2vec: 11](#_Toc43238310)

[Использование анализатора тональности: 11](#_Toc43238311)

# Первая часть

## База данных

Используется СУБД MongoDB.

Описание коллекций базы данных:

data – все найденные и скаченные новости

analysis - все найденные предложения с упоминанием персон и достопримечательностей.

tonality – тональность найденных предложений с у поминанием.

synonyms – найденные синонимы.

person – фамилии и имена персон.

attractions – название досопримечательностей.

## Парсер

Парсер новостного сайта (parser/scrapper.py) — проходит по новостям сайта novostivolgograda.ru и собирает ссылки на статьи. Затем заходит по всем собранным ссылкам и получает дату новости, её текст и название. Все данные записываются в коллекцию data.

Скриншот заполненной коллекции data парсером:



Рисунок 1. Коллекция data



Рисунок 2. Коллекция person



Рисунок 3. Таблица attractions

## Web-интерфейс

Web-интерфейс главной страницы выглядит следующим образом, на нем представлена ссылки на статьи с сайта novostivolgograda.ru, их названия и дата:

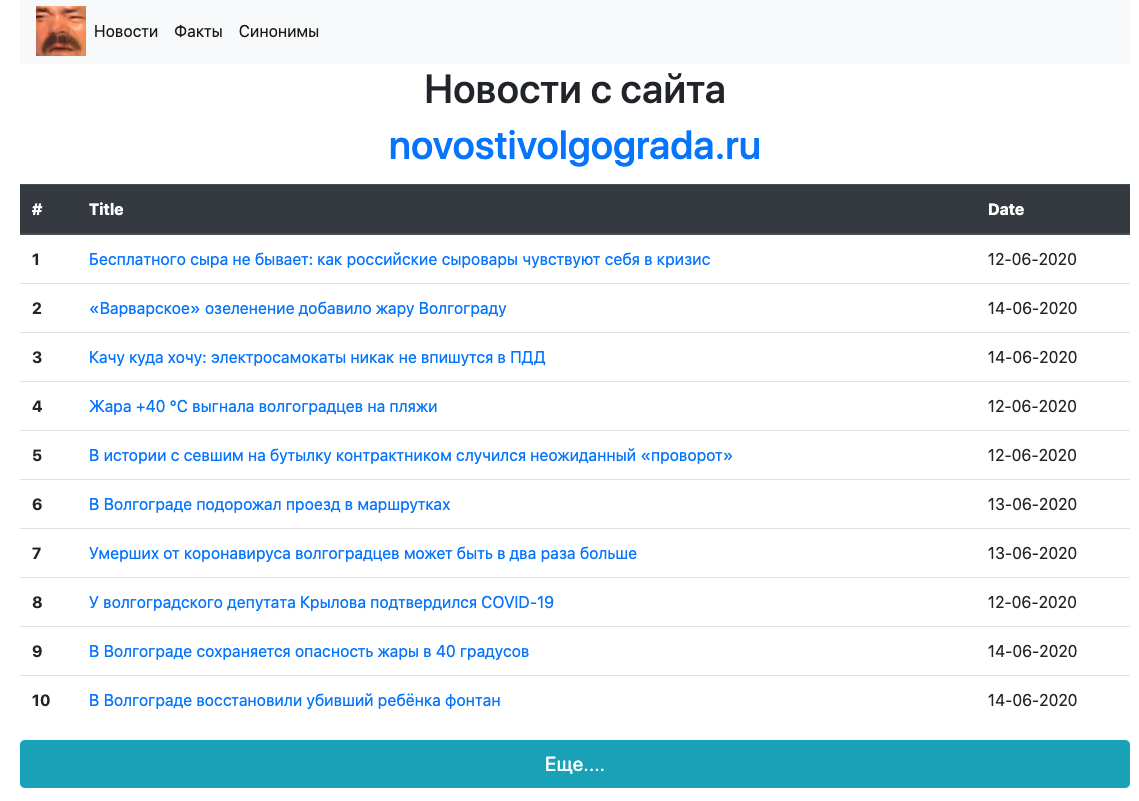


Рисунок 4. Интерфейс главной страницы

# Вторая часть

## Томита-парсер

Выделение персон и достопримечательностей и помещение их в таблицу analysis выполняет скрипт tomita.py.

Перед запуском необходимо его поместить в папку с файлами persons.cxx, attractions.cxx, mydic.gzt и facttypes.proto.

Пример выделяемых скриптом предложений:

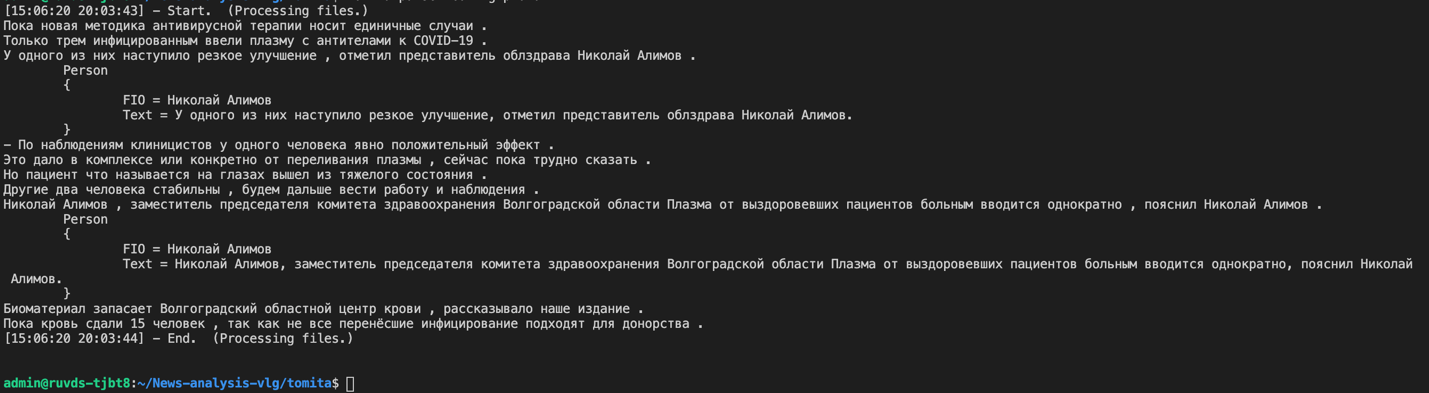


Рисунок 5. Пример выделяемых предложений

Рисунок 6. Интерфейс выделяемых предложений

## Word2Vec

Модель word2vec была обучена на новостных статьях из базы данных, объем которой составлял примерно 45000 статей. Модель находится в папке ./word2vec/model/kurs\_model/. Запуск модуля осуществляется запуском программы ./word2vec/main.py. Модуль записывает контекстные синонимы в БД и осуществляет следующий вывод:

'Контекстные синонимы слов, полученные из модели, обученной на статьях:'

'--------------------'

'алимов'

Row(word='зампредседателя', similarity=0.816402792930603)

Row(word='облздрава', similarity=0.781923234462738)

Row(word='себелев', similarity=0.7760810852050781)

Row(word='лукьяненко', similarity=0.7573933601379395)

Row(word='семисотов', similarity=0.7363854050636292)

'--------------------'

'бочаров'

Row(word='губернатор', similarity=0.9528191685676575)

Row(word='андрей', similarity=0.9124425649642944)

Row(word='бочаровглава', similarity=0.910007655620575)

Row(word='поручил', similarity=0.8725244402885437)

Row(word='проинспектировал', similarity=0.8666603565216064)

Модуль имеет следующий web-интерфейс, на нем представлено поле для ввода слово, к которому нужно найти контекстный синоним, после нажатия кнопки «Поиск» происходит вывод контекстных синонимов:

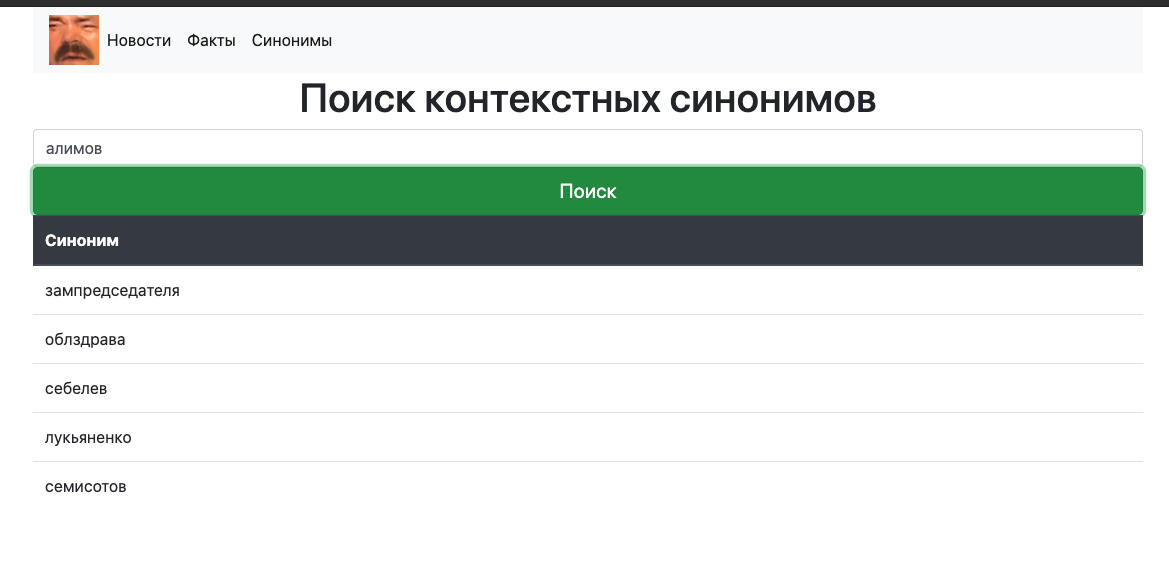


Рисунок 7. Интерфейс поиска контекстных синонимов

Скриншоты данных в таблицах:

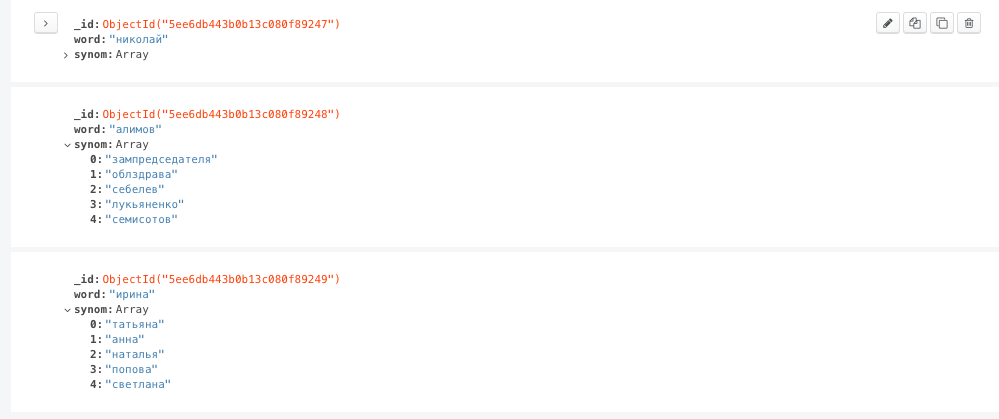


Рисунок 8. Таблица synonyms

# Третья часть

## Анализ тональности

Тональность определяется у предложений, выявленных томита-парсером на втором этапе работы. Для анализа была использована библиотека Dostoevsky. Запуск модуля осуществляется запуском программы tonality/dos.py. В результате работы модуль записывает в БД id предложений и их тональность, также. В web-интерфейсе представлены предложения, выявленные на втором этапе и их тональность, которая указана в скобках:

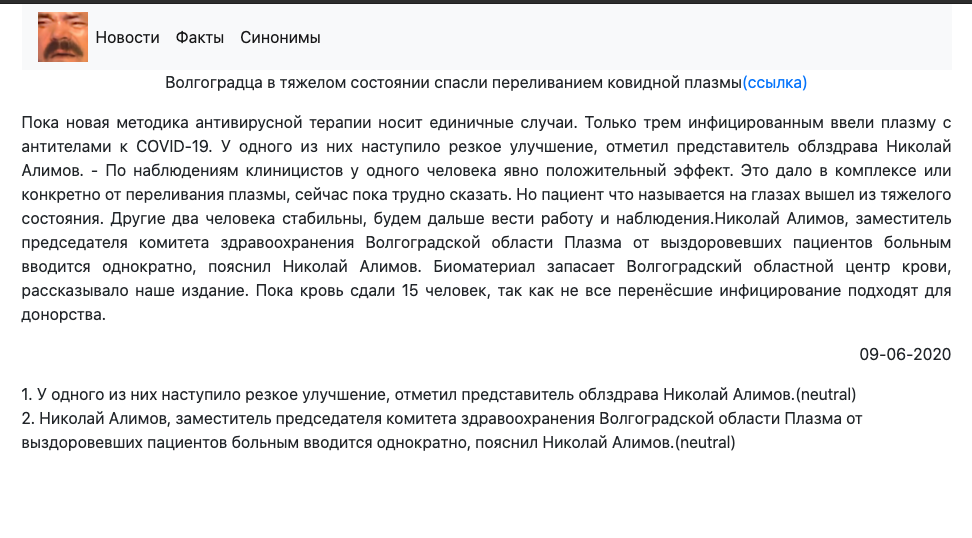


Рисунок 9. Интерфейс предложений и их тональности

Таблица tonality:

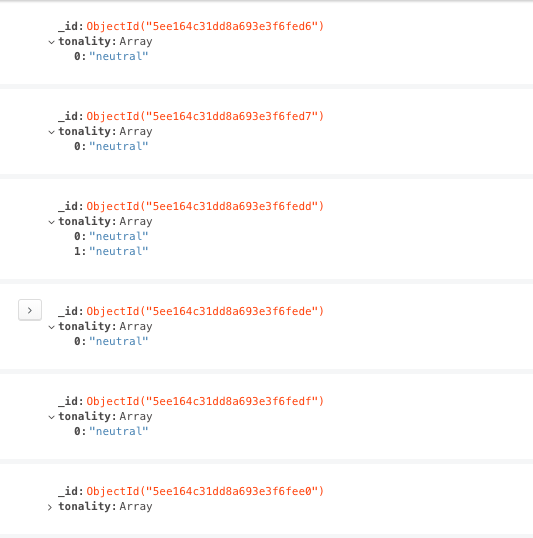


Рисунок 10. tonality

# Руководство системного администратора

## Установка библиотек для Python

Установить python 3.6 и pip (если их нет)

sudo apt install python3.6

Установить библиотеки

pip3 install pyspark flask dostoevsky pyppeteer pymongo schedule

Для корректной работы парсера установите зависимости

sudo apt install -y gconf-service libasound2 libatk1.0-0 libc6 libcairo2 libcups2 libdbus-1-3 libexpat1 libfontconfig1 libgcc1 libgconf-2-4 libgdk-pixbuf2.0-0 libglib2.0-0 libgtk-3-0 libnspr4 libpango-1.0-0 libpangocairo-1.0-0 libstdc++6 libx11-6 libx11-xcb1 libxcb1 libxcomposite1 libxcursor1 libxdamage1 libxext6 libxfixes3 libxi6 libxrandr2 libxrender1 libxss1 libxtst6 ca-certificates fonts-liberation libappindicator1 libnss3 lsb-release xdg-utils wget

## Установка и настройка MongoDB

sudo apt install -y mongodb

sudo systemctl status mongodb

Вы увидите следующий результат:

mongodb.service - An object/document-oriented database

Loaded: loaded (/lib/systemd/system/mongodb.service; enabled; vendor preset: enabled)

Active: active (running) since Sat 2018-05-26 07:48:04 UTC; 2min 17s ago

Docs: man:mongod(1)

Main PID: 2312 (mongod)

Tasks: 23 (limit: 1153)

CGroup: /system.slice/mongodb.service

└─2312 /usr/bin/mongod --unixSocketPrefix=/run/mongodb --config /etc/mongodb.conf

Создаём пользователя:

mongo

use admin

Создайте пользователя mongo-admin:

> db.createUser(

{

user: "**mongo-admin**",

pwd: "**passw0rd**",

roles: [ { role: "userAdminAnyDatabase", db: "admin" } ]

}

)

Создайте пользователя mongo-root:

> db.createUser(

{

user: "**mongo-root**",

pwd: "**passw0rd**",

roles: [ { role: "root", db: "admin" } ]

}

)

## Установка томита-парсера

cd ~

git clone <https://github.com/yandex/tomita-parser>

sudo apt-get install build-essential cmake lua5.2

cd tomita-parser && mkdir build && cd build

cmake ../src/ -DCMAKE\_BUILD\_TYPE=Release

make

Переместить файл libmystem\_c\_binding.so в ~/tomita-parser/build/bin/ :

wget https://github.com/yandex/tomita-parser/releases/download/v1.0/libmystem\_c\_binding.so.linux\_x64.zip

unzip libmystem\_c\_binding.so.linux\_x64.zip

rm libmystem\_c\_binding.so.linux\_x64.zip

Экспортируем томиту

export PATH="$HOME/tomita-parser/build/bin:$PATH"

source ~/.bashrc

Установка приложения

Загрузить с github:

git clone <https://github.com/Sergey1888888/News-analysis-vlg>

# Руководство пользователя

## Работа с web-интерфейсом:

Работа web-интерфейса осуществляется запуском программы web/app.py. После запуска откроется окно (Рисунок 4), на нем можно просмотреть список названий новостей и дату их публикации с сайта <https://novostivolgograda.ru/> , при нажатии на название новости осуществляется переход на страницу новости. Пользователь может перейти к поиску синонимов нажав кнопку «Синонимы», откроется окно (Рисунок 9), в котором пользователь может ввести персону или достопримечательность города Волгограда и получить 5 синонимов. Если пользователь перешел на страницу новости, то перед ним откроется окно (Рисунок 9) с текстом новости, ссылкой на её источник, датой и предложениями выделенными томита-парсером с их тональностью. Если пользователь с главной страницы нажал на кнопку «Факты», то откроется окно с предложениями, выделенными томита-парсером и ссылками на новости, в которых эти предложения встречались.

## Использование парсера:

Для сбора данных необходимо запустить файл parser/scrapper.py, который найдет новые новости, запишет их в БД.

Примеры команды запуска:

python3 parser/scrapper.py

Таблицы person и attractions берутся из репозитория. (файлы person.json и attractions.json)

## Использование модуля томита-парсера:

Запустить tomita.py, дождаться окончания. tomita.py следует запускать из папки, в которой он расположен. В папке со скриптом должны находиться файлы persons.cxx, attraction.cxx, mydic.gzt и facttypes.proto.

Пример использования:

cd tomita

python3 tomita.py

## Использование word2vec:

Если проект скачан из репозитория, то в нем уже имеется обученная на 45000 статей модель (word2vec/model/kurs\_model/), готовая к выявлению контекстных синонимов. Если же вы хотите получить свою модель на основе данных из БД, то удалите/переименуйте папку исходной модели и запустите программу word2vec/run.py, начнется процесс подготовки и обучения модели. Для теста и просмотра контекстных синонимов запустите программу word2vec/test.py, в консоле выводиться синонимы.

## Использование анализатора тональности:

Чтобы воспользоваться анализатором тональности установить на свой ПК библиотеку Dostoevsky (<https://github.com/bureaucratic-labs/dostoevsky>), после чего необходимо запустить код tonality/dos.py. Результатом работы программы будет являться тональность, выявленная у предложений, выделенных томита-парсером.