Минобрнауки России

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Волгоградский государственный технический университет»

Факультет Электроники и вычислительной техники

Кафедра Системы автоматизированного проектирования и поискового конструирования

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовой работе (проекту)

по дисциплине Компьютерная Лингвистика

на тему Программный модуль для анализа новостей из БД

_____

_____

_____

Студент Онищенко Владислав Андреевич
(фамилия, имя, отчество)

Группа ИВТ-364

Руководитель работы (проекта) _____ Коробкин Д.М.
(подпись и дата подписания) (инициалы и фамилия)

Члены комиссии:

_____ _____
(подпись и дата подписания) (инициалы и фамилия)

_____ _____
(подпись и дата подписания) (инициалы и фамилия)

_____ _____
(подпись и дата подписания) (инициалы и фамилия)

Нормоконтролер _____ _____
(подпись, дата подписания) (инициалы и фамилия)

Волгоград 2020 г.

Содержание

1. Цель проекта

Необходимо разработать программный модуль для анализа новостей из БД с помощью Томита-парсера и программный модуль для проведения с помощью Spark MlLib анализа модели word2vec на всем объеме новостных статей из БД.

2. Задача проекта

Написать программу в интегрированной среде разработки PyCharm 2019 Professional на языке python с помощью Томита-парсера, а также встроенных библиотек, которая будет выполнять выделение упоминаний в тексте значимых персон Волгоградской области и достопримечательностей и заносить их в базу данных, а также программный модуль, анализирующий модели word2vec на всех новостях из базы данных Запуск программы осуществить с помощью операционной среды Linux.

3. Описание программного модуля

В проекте создается программный модуль для анализа новостей из БД. Выделение с помощью Томита-парсера упоминаний в тексте значимых персон Волгоградской области и достопримечательностей. Происходит фиксирование в БД предложения с их упоминанием для дальнейшего анализа тональности.

Также создается программный модуль для проведения с помощью Spark MlLib анализ модели word2vec на всем объеме новостных статей из БД. Для персон Волгоградской области и достопримечательностей определить контекстные синонимы и слова, с которыми они упоминались в тексте.

Для реализации программы, информация для программного модуля будет браться из базы данных MongoDB в формате json. Для этого необходимо конвертировать хранимую информацию в коллекциях базы данных в файл json.

Команда конвертирования данных: mongoexport -d parserCL -c parser-o C:/test/news.json

Создадим файлы для Томита-парсера, которые будут отвечать за нахождение Персон и Достопримечательностей Волгоградской области.

В них будут хранится словари, с помощью которых будут определяться ФИО персон и имена собственные достопримечательностей. Полученные предложения с упоминанием фиксируются в базе данных для дальнейшей работы третьим студентом.

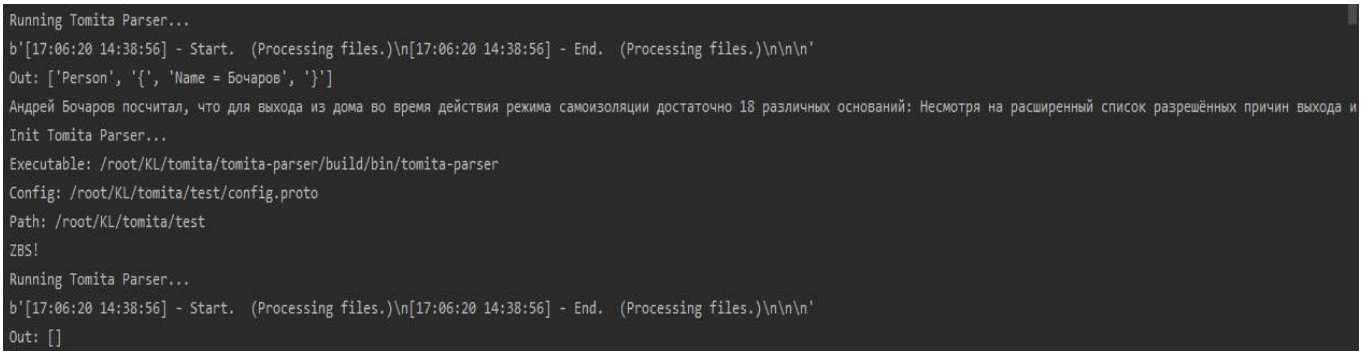

Рисунок 1. Пример выполнения томита-парсера

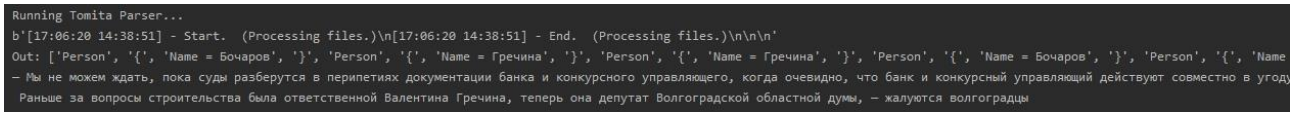

Рисунок 2. Пример выделяемых предложений

Word2Vec

Модель word2vec была обучена на новостных статьях из базы данных, объем которой составлял примерно 1600 статей. Модель находится в папке ./news_analyser/w2v/model/.

Запуск модуля осуществляется запуском программы ./news_analyzer/w2v/main.py.

Модуль записывает контекстные синонимы в БД и осуществляет следующий вывод:

'Бочаров Андрей Иванович'

бочаров

андрей

губернатор

попков

куприков

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

'Мержоева Зина Османовна'

зина

мержоева

вицегубернатор

эксвицегубернатор

выстропов

Заключение

В результате проделанной работы был разработан программный модуль, осуществляющий выделение упоминаний в тексте значимых персон Волгоградской области и достопримечательностей и заносить их в базу данных со страницы новостей и занесение информации в базу данных MongoDB. Разработка осуществлялась с помощью операционной системы Linux, что позволило приобрести ряд навыков для написания проекта.

Список использованной литературы

1. Установка и использование оболочки Linux – Режим доступа: https://www.comss.ru/page.php?id=4897 (дата обращения 08.06.2020).
2. Основы работы в операционной системе Linux С.А.Немнюгин – Режим доступа: http://www.cph.phys.spbu.ru/documents/Second/unix_SNv2.pdf (дата обращения 09.06.2020)
3. Наташа — библиотека для извлечения структурированной информации из текстов на русском языке – Режим доступа: https://habr.com/ru/post/349864/ (дата обращения 12.06.2020)
4. Natasha - Документация https://natasha.readthedocs.io/ru/latest/ (дата обращения 12.06.2020)
5. Ознакомления с работой с MongoDB и PyMongo – Режим доступа: https://api.mongodb.com/python/current/tutorial.html (дата обращения 11.06.2020)

Приложение А

Программный код модуля

```
import os
import subprocess
import  re
import xml.etree.ElementTree as ElementTree
from pymongo import MongoClient

client = MongoClient()
db = client.parserCLV1ru
parser = db.parser
tomita = db.tomita



class TomitaParser(object):
    def __init__(self, executable, config, debug=True, validate=False):
        self.debug_mode = debug
        self.debug("Init Tomita Parser...")

        self.executable = os.path.expanduser(executable)
        if not os.path.exists(self.executable):
            raise Exception("Tomita executable not found at: %s" % self.executable)

        self.debug("Executable: %s" % self.executable)

        self.config = os.path.expanduser(config)
        if not os.path.exists(self.config):
            raise Exception("Config file not found at: %s" % self.config)

        self.debug("Config: %s" % self.config)

        self.path = self.config[:self.config.rfind("/")]

        self.debug("Path: %s" % self.path)
        self.debug("ZBS!")

        if validate:
            self.validate_config()

    def validate_config(self):
        is_xml = False
        with open(self.config, "r") as f:
            for line in f.readlines():
                line = line.strip()
```

```
                if line:
                    if line.startswith("File"):
                        raise Exception("This library uses STDIN and STDOT for
communicating with Tomita Parser, "
                                        "please remove all File = \"...\" from
Input and Output sections in config.")
                    if line.startswith("Format") and "xml" in line:
                        is_xml = True
        if not is_xml:
            raise Exception("This library working only with XML output, "
                            "please add \"Format = xml;\" to the Output section.")

    def run(self, text = "", with_facts=True, with_leads=True):
        self.debug("Running Tomita Parser...")
        with open('/root/KL/tomita/example/input.txt', 'w', encoding='utf-8') as
inputFile:
            inputFile.writelines(text)
        pipe = subprocess.Popen(
            [self.executable, self.config],
            stdout=subprocess.PIPE,
            stdin=subprocess.PIPE,
            stderr=subprocess.PIPE,
            cwd=self.path
        )
        pipe.wait()
        out, err = pipe.communicate(input=text.encode("utf-8"))

        self.debug(err)

        output = []
        with open("/root/KL/tomita/example/output.txt", 'r', encoding='utf-8') as
outputFile:
            readedNews = outputFile.readlines()
            appending = False
            for news in readedNews:

                if "Person" in news or "Place" in news:
                    appending = True
                if appending:
                    output.append(news.strip())
                if "}" in news:
                    appending = False
        return output

    def debug(self, text):
```

```
        if self.debug_mode:
            print(text)


def getPropWihtCoincidence(match, text, start = 0):
    out = None
    indexMatch = text.find(match, start)
    split_regex = re.compile(r'[.|!|?|…]')
    if(indexMatch is not -1):
        tt = text[start:]
        for prop in split_regex.split(tt):
            ind = prop.find(match)
            if ind is not -1:
                print(prop)
                out = prop
                break
    return out


if __name__ == "__main__":
    config = "/root/KL/tomita/test/config.proto"
    for i in parser.find():
        tomita = TomitaParser("/root/KL/tomita/tomita-parser/build/bin/tomita-
parser", config, debug=True)
        output = tomita.run(i["text"])
        print("Out:", output)
        srt = 0
        for s in output:
            ind = s.find("Name = ")
            if ind is not -1:
                prop = getPropWihtCoincidence(s[ind+7:], i['text'], start=srt)
                toBD = {
                    'match': s[ind+7:],
                    'prop': prop,
                }
                out = parser.insert_one(toBD)
                srt = ind+7
```