Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа № 2

По дисциплине ЕЯИИС

Тема: «Разработка системы автоматического реферирования документов»

Выполнил:

Студент 4 курса

Группы ИИ-16

Журавлёв В.А.

Проверил:

Слинко Е.В.

Брест, 2021

**Цель работы:** освоить на практике основные принципы автоматического реферирования документов.

**Код программы:**

package com.university.service;

import com.university.entity.ClassicEssay;

import com.university.entity.Essay;

import com.university.entity.KeyWordsEssay;

import com.university.entity.Sentence;

import org.springframework.stereotype.Service;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

import java.nio.file.Files;

import java.nio.file.Path;

import java.nio.file.Paths;

import java.util.\*;

import java.util.stream.Collectors;

@Service

public class EssayService {

private static final String REGEX = "[– \n,.;:\"'»«()—!0-9\\[\\]\\-]";

public Essay getEssay(final String fileName) throws Exception{

File file = Files.walk(Paths.get("src/main/resources"))

.filter(Files::isRegularFile)

.map(Path::toFile)

.filter(f -> f.toString().contains(fileName))

.findFirst()

.orElseThrow(() -> new RuntimeException(String.format("File %s not found", fileName)));

//считываем все слова

BufferedReader br = new BufferedReader(new FileReader(file));

StringBuilder resultStringBuilder = new StringBuilder();

String line;

while ((line = br.readLine()) != null) {

resultStringBuilder.append(line).append(" ");

}

List<String> sentences = Arrays.stream(resultStringBuilder.toString().split("[.]"))

.filter(sentence -> !sentence.isEmpty())

.collect(Collectors.toList());

List<String> words = Arrays.stream(resultStringBuilder.toString().split(REGEX))

.filter(word -> !word.isEmpty() && isNeeded(word))

.map(String::toLowerCase)

.collect(Collectors.toList());

Map<String, Integer> allWordsWithCount = new HashMap<>();

String wordWithMaxCount = "";

int maxCount = Integer.MIN\_VALUE;

for (String newWord : words) {

if (!allWordsWithCount.containsKey(newWord)) {

int count = 0;

for (String word : words) {

if (word.equals(newWord))

count++;

}

if(count > maxCount){

maxCount = count;

wordWithMaxCount = newWord;

}

allWordsWithCount.put(newWord, count);

}

}

//считываем по параграфам

Map<Integer, String> paragraphs = new HashMap<>();

Integer paragraphCounter = 0;

br = new BufferedReader(new FileReader(file));

while ((line = br.readLine()) != null) {

if(line.startsWith(" ")){

paragraphs.put(paragraphCounter, resultStringBuilder.toString());

resultStringBuilder = new StringBuilder();

paragraphCounter++;

}

resultStringBuilder.append(line).append(" ");

}

if(!resultStringBuilder.toString().isEmpty()){

paragraphs.put(paragraphCounter, resultStringBuilder.toString());

}

List<String> mainWordsInParagraph = new ArrayList<>();

String finalWordWithMaxCount = wordWithMaxCount;

paragraphs.forEach((number, paragraph) -> {

List<String> wordsInParagraph = Arrays.stream(paragraph.split(REGEX))

.filter(word -> !word.isEmpty() && isNeeded(word))

.map(String::toLowerCase)

.collect(Collectors.toList());

Map<String, Integer> allWordsWithCountInParagraph = new HashMap<>();

String wordWithMaxCountInParagraph = "";

int maxCountInParagraph = Integer.MIN\_VALUE;

for (String newWord : wordsInParagraph) {

if (!allWordsWithCountInParagraph.containsKey(newWord)) {

int count = 0;

for (String word : wordsInParagraph) {

if (word.equals(newWord))

count++;

}

if(count > maxCountInParagraph

&& !newWord.contains(finalWordWithMaxCount)

&& !finalWordWithMaxCount.contains(newWord)

&& !mainWordsInParagraph.contains(newWord)){

if(newWord.length() > 4 && finalWordWithMaxCount.length() > 4){

if(!newWord.substring(0, newWord.length() -3)

.contains(finalWordWithMaxCount.substring(0, finalWordWithMaxCount.length() - 3))){

maxCountInParagraph = count;

wordWithMaxCountInParagraph = newWord;

}

}

else {

maxCountInParagraph = count;

wordWithMaxCountInParagraph = newWord;

}

}

allWordsWithCountInParagraph.put(newWord, count);

}

}

mainWordsInParagraph.add(wordWithMaxCountInParagraph);

});

Map<String, List<String>> keyWordsEssayData = new HashMap<>();

keyWordsEssayData.put(wordWithMaxCount, mainWordsInParagraph);

List<Sentence> sentenceList = new ArrayList<>();

//классический реферат

for (String s : sentences) {

int score = 0;

List<String> wordsInSentence = Arrays.stream(s.split(REGEX))

.filter(word -> !word.isEmpty() && isNeeded(word))

.map(String::toLowerCase)

.collect(Collectors.toList());

for (String wordInSentence : wordsInSentence) {

score += allWordsWithCount.get(wordInSentence);

}

sentenceList.add(Sentence.builder()

.score(score)

.value(s)

.build());

}

List<String> resultSentences = sentenceList.stream()

.sorted((s1, s2) -> s2.getScore().compareTo(s1.getScore()))

.limit(5)

.map(Sentence::getValue)

.collect(Collectors.toList());

return Essay.builder()

.keyWordsEssay(KeyWordsEssay.builder()

.data(keyWordsEssayData)

.build())

.classicEssay(ClassicEssay.builder()

.data(resultSentences)

.build())

.build();

}

private boolean isNeeded(String word){

return !word.equalsIgnoreCase("в")

&& !word.equalsIgnoreCase("что")

&& !word.equalsIgnoreCase("то")

&& !word.equalsIgnoreCase("где")

&& !word.equalsIgnoreCase("я")

&& !word.equalsIgnoreCase("и")

&& !word.equalsIgnoreCase("он")

&& !word.equalsIgnoreCase("она")

&& !word.equalsIgnoreCase("вы")

&& !word.equalsIgnoreCase("ты")

&& !word.equalsIgnoreCase("они")

&& !word.equalsIgnoreCase("мы")

&& !word.equalsIgnoreCase("на")

&& !word.equalsIgnoreCase("под")

&& !word.equalsIgnoreCase("по")

&& !word.equalsIgnoreCase("от")

&& !word.equalsIgnoreCase("до")

&& !word.equalsIgnoreCase("после")

&& !word.equalsIgnoreCase("возле")

&& !word.equalsIgnoreCase("над")

&& !word.equalsIgnoreCase("из")

&& !word.equalsIgnoreCase("из-за")

&& !word.equalsIgnoreCase("тыс")

&& !word.equalsIgnoreCase("лет")

&& !word.equalsIgnoreCase("н")

&& !word.equalsIgnoreCase("для")

&& !word.equalsIgnoreCase("э")

&& !word.equalsIgnoreCase("считают")

&& !word.equalsIgnoreCase("как")

&& !word.equalsIgnoreCase("так")

&& !word.equalsIgnoreCase("как-то")

&& !word.equalsIgnoreCase("не")

&& !word.equalsIgnoreCase("к")

&& !word.equalsIgnoreCase("её")

&& !word.equalsIgnoreCase("ее")

&& !word.equalsIgnoreCase("их")

&& !word.equalsIgnoreCase("но")

&& !word.equalsIgnoreCase("то")

&& !word.equalsIgnoreCase("или")

&& !word.equalsIgnoreCase("все")

&& !word.equalsIgnoreCase("и")

&& !word.equalsIgnoreCase("при")

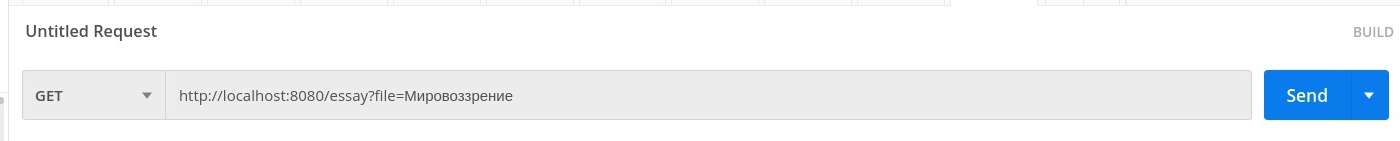
&& !word.equalsIgnoreCase("за");

}

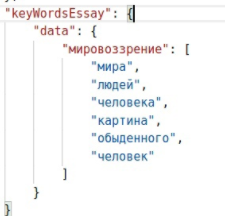
}

**Тестирование:**

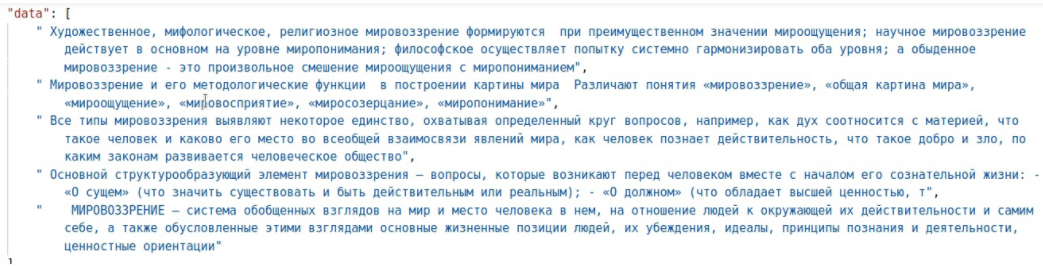
При запросе “Мировоззрение” (в системе лежит файл с таким именем)



Реферат в виде ключевых слов:



Классический реферат:



**Вывод:** освоил на практике основные принципы автоматического реферирования документов.