Университет ИТМО

Практическая работа №4

По дисциплине “Компьютерная лингвистика”

Автор: Лайок Олег Владимирович

Группа: К3242

Факультет: ИКТ

Преподаватель: Чернышева А.В.

Санкт-Петербург, 2021 г.

После создания словаря токенов, для частеречной разметки и более удобного анализа в дальнейшем, я решил их пролемматезировать, используя бибилиотеку pymorphy2:

import json

import pymorphy2

with open("tokens.json", "r", encoding ='utf-8') as f:

    clear\_tokens = json.load(f)

analyzer = pymorphy2.MorphAnalyzer()

normalized = []

for tokens in clear\_tokens:#лемматизация токенов

    tokens = [analyzer.parse(word)[0].normal\_form for word in tokens]

    normalized.append(tokens)

Далее, для поиска нетепичных документом путем анализа значений средних долей для каждого части, в каждом документе было посчитано количество вхождений каждого токена:

word\_count = []

for tokens in normalized:#ранжирование по частоте

    tokens = [(word, tokens.count(word)) for word in set(tokens)]

    word\_count.append(tokens)

for words in word\_count:

    words.sort(key=lambda pair:pair[1], reverse=True)

Полученые данные:

[[('pussycat', 1), ('робин', 1), ('радиолентач', 1), ('хореограф', 1),…], […],…]

После этого уже была произведена частеречная разметка. Создаем список словарей каждого доккумента, где каждое слово в документе имеет поля: слово, часть речи и частота появления в доккументе:

final\_dict=[]

for i in range(len(word\_count)):

    final\_dict.append([])

    for j in range(len(word\_count[i])):

        final\_dict[i].append({'word': word\_count[i][j][0], 'p\_of\_s': analyzer.parse(word\_count[i][j][0])[0].tag.POS, 'freq': word\_count[i][j][1]})

Полученные данные:

[[{'word': 'pussycat', 'p\_of\_s': None, 'freq': 1}, {'word': 'робин', 'p\_of\_s': 'NOUN', 'freq': 1}, {'word': 'радиолентач', 'p\_of\_s': 'NOUN', 'freq': 1}, {'word': 'хореограф', 'p\_of\_s': 'NOUN', 'freq': 1},…],…[…]]

Далее ищем количество каждой части речи в каждом документе(тэги для частей речи взяты из документации бибилиотеки pymorphy2). Не учитываем неопознаные части речи, обозначенные как null, т.к. это английские слова и цифры, не влияющие на анализ документа с точки зрения частот вхождений частей речи:

list\_of\_quant = []

for i in range(len(final\_dict)):# создаем список словарей каждого доккумента, где каждое слово в документе имеет поля: слово, часть речи и частота появления в доккументе

    a = {'NOUN': 0, 'ADJF': 0, 'ADJS': 0, 'COMP': 0, 'VERB': 0, 'INFN': 0, 'PRTF': 0, 'PRTS': 0, 'GRND': 0, 'NUMR': 0, 'ADVB': 0, 'NPRO': 0, 'PRED': 0, 'PREP': 0, 'CONJ': 0, 'PRCL': 0, 'INTJ': 0, 'number\_of\_words': 0}

    for j in range(len(final\_dict[i])):

        if final\_dict[i][j]['p\_of\_s'] != None:

            a[final\_dict[i][j]['p\_of\_s']] = a[final\_dict[i][j]['p\_of\_s']] + final\_dict[i][j]['freq']

            a['number\_of\_words'] = a['number\_of\_words'] + final\_dict[i][j]['freq']

    list\_of\_quant.append(a)

Далее находим частоту вхождения части речи в документах:

string = ('NOUN', 'ADJF', 'ADJS', 'COMP', 'VERB', 'INFN', 'PRTF', 'PRTS', 'GRND', 'NUMR', 'ADVB', 'NPRO', 'PRED', 'PREP', 'CONJ', 'PRCL', 'INTJ')

for docs in list\_of\_quant:

    for obj in string:

        docs[obj] =round(docs[obj] / docs['number\_of\_words'],2)

print(list\_of\_quant)

Полученный список:

[{'NOUN': 0.67, 'ADJF': 0.25, 'ADJS': 0.0, 'COMP': 0.0,…},…],[..]]

Для нахождения среднеквадратичного отклонения, группируем частоты для каждой части речи во всех документах, находим среднюю и по ней, с помощью модуля numpy находим среднеквадратичное отклонение:

std\_a ={'NOUN': [], 'ADJF': [], 'ADJS': [], 'COMP': [], 'VERB': [], 'INFN': [], 'PRTF': [], 'PRTS': [], 'GRND': [], 'NUMR': [], 'ADVB': [], 'NPRO': [], 'PRED': [], 'PREP': [], 'CONJ': [], 'PRCL': [], 'INTJ': [], 'number\_of\_words': []}

for docs in list\_of\_quant:

    for obj in string:

        std\_a[obj].append(docs[obj])

medium\_a = dict(std\_a)

for obj in string:

        medium\_a[obj] = round(np.average(std\_a[obj]), 2)

print("средние значение", medium\_a)

for obj in string:

    std\_a[obj] = round(np.std(std\_a[obj]), 2)

print('стандартные отклонение', std\_a)

средние значение {'NOUN': 0.58, 'ADJF': 0.16, 'ADJS': 0.0, …}

стандартные отклонение {'NOUN': 0.11, 'ADJF': 0.09, 'ADJS': 0.0, …}

По полученным данным, можно сказать, что документы в основном состоят из существительных, прилагательных и глаголов (в форме инфинитива, т.к. была произведена лемматизация). Я считаю данный результат удовлетворительным, т.к. данная выборка, а именно посты с заголовками или кратким содержанием новостей содержат в основном именно такие части речи.

Используя найденные стандартные отклонения задаем для поиска нетипичных документов: документ является нетипичным если частота встречаемости части речи в документе отклоняется более чем на 2 стандартных отклонения:

for i in range(len(list\_of\_quant)):

    for j in string:

        if list\_of\_quant[i][j] > (medium\_a[j] + std\_a[j]\*2):

            print(f'документ номер {i} нетипичный из-за того что {j} превышает среднеквадратичное отклонение в {round((list\_of\_quant[i][j] - medium\_a[j])/std\_a[j], 2)} раз(а)')

Пример найденного документа:

документ номер 1 нетипичный из-за того что NUMR превышает среднеквадратичное отклонение в 4.0 раз(а)

В основном, нетипичные документы, это те новости в которых много числовых данных(«пятеро человек…», «двенадцать регионов…»), а так же документы с не часто используемыми частями речи(наречиями, причастиями, местоимениями и тд)

Ссылка на код и размеченный словарь: https://github.com/OlegLaiok/Comp\_Lingv/tree/homework4