



Грацианова Татьяна Юрьевна

Кандидат физико-математических наук, преподаватель подготовительных курсов факультета Вычислительной математики и кибернетики МГУ им. М.В. Ломоносова

Что может статистика?

Статистические методы компьютерной лингвистики — часть 2

В первой части этой статьи (см. предыдущий номер) мы научились определять частоту встречаемости различных символов в тексте, а также узнали, для решения каких задач такие программы можно использовать. Мы получили не очень много информации, но и подсчетов произвели немного. Что еще можно посчитать в тексте, чтобы информации было больше?

Биграммы

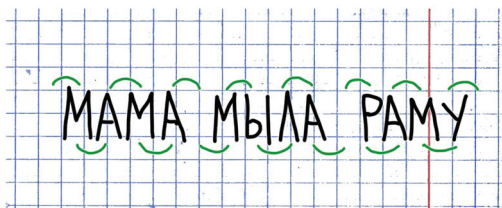
Давайте анализировать буквенные N-граммы. N-грамма — это последовательность из N-элементов. Обычно исследуют биграммы (последовательность из двух элементов) и триграммы — из трех элементов. Лингвистика уделяет

много внимания анализу буквенных N-грамм. В школе вы проходили правила, касающиеся биграмм «ши», «жи», триграмм «оло», «оро». Вспомните, что это были за правила. Какие еще N-граммы упоминались в школе?

о**з**о**р**о**д** д**о**р**о**з**а**
 со**л**о**м**а
 со**л**о**в**е**й** со**р**о**к**а
 мо**л**о**к**о

Давайте будем анализировать последовательность из двух рядом стоящих букв, посмотрим, какие они бывают, как часто встречаются. Пробелы тоже будем учитывать — они нам будут указывать, что буква является первой или последней в слове. Например, в предложении «мама мыла раму с мылом» последовательно выделяются следующие биграммы (вместо пробела напишем знак нижнего подчеркивания, чтобы он был заметен): «_м», «ма», «ам», опять «ма», «а_», еще раз «_м», «мы», «ыл» — и так далее. Даже на приме-

ре такого маленького текста видно, что частота биграмм разная: в нашем предложении «_м» встречается 3 раза, «ма», «мы», «ыл» и еще некоторые — по два, многие — под одному разу. Впрочем, что это мы «вручную» считаем — так и ошибиться недолго. Давайте писать программу.



Алгоритм

Есть ли разница — считать одиночные буквы или биграммы? Давайте посмотрим, будет ли отличаться алгоритм на Паскале. Сразу видим отличие: когда мы считали буквы, массив для подсчета состоял из N компонент (где N — количество букв в алфавите). Здесь компонент будет N^2 (пробел тоже посчитаем буквой, то есть для русского языка биграмм будет $34 \times 34 = 1156$). Как же их нумеровать? Давайте сделаем двумерный массив (матрицу). Нарисуем таблицу 34×34 , по горизонтали и вертикали поставим пробел и буквы от «а» до «я» — похоже на игру «Морской бой», только вместо чисел буквы. Левая верхняя клетка будет иметь индекс [пробел, пробел], правая нижняя — [я,я].

Для нашего примера после обработки текста про маму и раму матрица будет выглядеть так (на всех пустых местах стоят нули, расцветка цветом сделана для удобства просмотра) (рис. 1).

Ну а больше никаких особенных отличий в алгоритме нет: все элемен-

ты матрицы надо обнулить, а потом аккуратненько брать пару символов, смотреть, входят ли они в исследуемый диапазон (могут и не входить — если они цифры или знаки препинания), составлять из них пару индексов и добавлять единичку к соответствующей компоненте массива.

Будьте внимательны, считывая буквы из файла: каждая буква должна быть использована по два раза — сначала она будет в биграмме второй, а потом первой. Для единообразия, чтобы первая буква тоже использовалась два раза, надо перед ней поставить пробел (в файле или программно).

Мы уже выяснили, что получившийся массив-результат будет очень большим. А вся ли информация в нем будет одинаково значимой? Легко догадаться, что в нем будет очень много нулей: не все биграммы используются в языке. Например, трудно себе представить сочетания двух мягких или твердых знаков подряд. Может быть, заранее исключить такие биграммы из счета? Не получит-

	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	-
А													2																				2	
Б																																		
В																																		
Г																																		
Д																																		
Е																																		
Ё																																		
Ж																																		
З																																		
И																																		
Й																																		
К																																		
Л	1																																	
М	2																				1							1						
Н																																		
О																																		
П																																		
Р	1																																	
С																																		
Т																																		
У																																	1	
Ф																																		
Х																																		
Ц																																		
Ч																																		
Ш																																		
Щ																																		
Ъ																																		
Ы													1																					
Ь																																		
Э																																		
Ю																																		
Я																																		
-													2					1																

Рис. 1

ся! Во-первых, мы не знаем, сколько их, какие они — мы ведь в том числе для того и пишем программу, чтобы это узнать. А во-вторых, даже самая невероятная биграмма может встретиться в тексте. Причины могут быть самые разные: ошибка, опечатка, желание автора текста.

Было бы разумнее формировать массив по ходу исследования текста: встретилась биграмма — смотрим,



если она раньше не встречалась, записываем ее в массив с единицей, если раньше встречалась — увеличиваем значение на единицу. Давайте этот алгоритм реализуем на Питоне.

В прошлый раз при подсчете частоты встречаемости букв на Пи-

тоне мы использовали словарь. Там его использование не было до конца оправдано, а вот в данном случае мы сможем воспользоваться преимуществами этой структуры данных: в Питоне есть встроенные средства для проверки, входит ли ключ в словарь.

Получаем такую программу:

```
# Подсчет количества буквенных биграмм, запись их в словарь,
# сортировка
slovbe={} # создание пустого словаря
f =open('text2.txt','r', encoding="UTF-8") # открыли файл для
# чтения
b1=' ' # пробел
b2=f.read(1) # считываем первый служебный символ
b2=f.read(1) # считываем очередной символ
while b2 != '': # пока из файла что-то можно прочитать
    if (b2>='А') and (b2<='Я'):# если большая русская буква,
# заменяем на маленькую
        b2=chr(ord(b2)+ord('a')-ord('А'))
    elif (b2<'а') or (b2>'я'):
        b2=' '
    s=b1+b2 # соединяем символы в пару
    if s !=' ': # два пробела подряд не рассматриваем
        if s in slovbe: # если биграмма есть в словаре
            slovbe[s]+=1 # увеличим значение на 1
        else:
            slovbe[s]=1 # иначе добавим бигramму в словарь
# со значением 1
    b1=b2 # готовимся к работе с очередной парой
    b2=f.read(1) # читаем очередной символ
print(slovbe) # так для отладки можно распечатать полученный
# словарь
print('результат после сортировки')
sorted_be = sorted(slovbe.items(), key=lambda x: x[1],
reverse=True)
print(sorted_be)
```

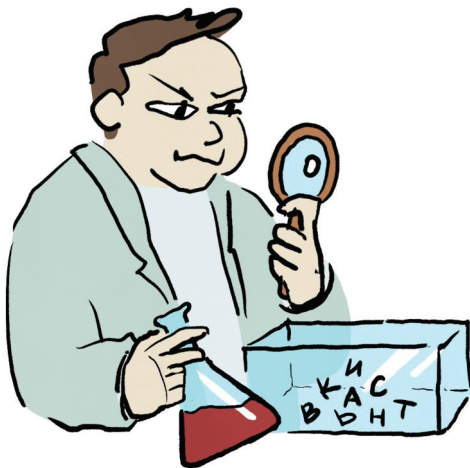
Можно заметить, что мы здесь произвели некоторую предобработку входных символов:

- заменили большие буквы на маленькие;
- все символы, не являющиеся русскими буквами, заменили на пробелы.

Таким образом, мы учитываем только буквы, знаки препинания и цифры не рассматриваем. Возможно, для решения каких-то задач потребуется другая предобработка, другой учет кодировки.

Алгоритм

Что же можно делать с нашими результатами? Напомним, что для получения правильных результатов текст, который подается на вход программы, должен быть большим.



Во-первых, можно посмотреть, с каких букв чаще всего начинаются слова (это биграммы, у которых на первом месте пробел). На первый взгляд, эту информацию можно получить из словаря — просто посмотреть, сколько слов на каждую букву в нем содержится. Однако в словаре могут быть очень редкие слова, которые в тексте не содержатся. Словарь показывает, сколько слов содержится в языке, а с помощью нашей программы мы можем подсчитать, какие слова часто употребляются. Вы знаете, что больше всего слов в русском языке начинаются на букву «п»?

Можно посмотреть, на какие буквы чаще всего оканчиваются слова: это сильно зависит от направленности текста. Если в нем много прилагательных (художественный текст, поэзия), часто последними буквами будут «я», «й».

Посмотрим на биграммы. Здесь разброс частоты встречаемости гораздо больше, чем с буквами. Очень много биграмм, которые в текстах не встречаются. И это не только что-то невероятное типа двух твердых знаков подряд или мягкого знака после гласной. Оказывается, многие согласные буквы в русском языке плохо сочетаются между собой. Скажем, уже упомянутая буква «п» любит быть первой, однако после нее не бывает большинства согласных букв, нет слов с сочетаниями «пб», «пв», «пг» и многих-многих других. Самые частые биграммы в русском языке «ст», «но», «ен», «то», «на» — не будем здесь все перечислять, посмотрите сами, воспользовавшись нашей программой. Популярные русские триграммы «сто», «ено», «нов», «тов». В английском самые частые биграммы «th», «he», «in», «er», а самой популярной триграммой является «the» (что весьма ожидаемо, правда?).

Посмотрите на разных текстах, получаются ли у вас похожие результаты. Посмотрите, зависит ли частота биграмм и триграмм от:

- направленности текста (проверьте художественный, научный текст, стихотворение);

- предназначения текста (проверьте, изменится ли частота биграмм, если на вход программе подать стишки и песенки, предназначенные для малышей, или текст из смс-сообщений или форумов Вашего общения с одноклассниками);

- времени написания текста (попробуйте, например, «Слово о полку Игореве»).

Использование

Поиграли, посмотрели, пора теперь узнать, зачем это может быть нужно, как используется.

Появление в тексте не характерной для языка биграммы или триграммы — сигнал об ошибке или о наличии нового сокращения. Например, до появления аббревиатуры «ЕГЭ» вероятность встретить в тексте биграмму «гэ» была близкой к нулю. Соответственно можно автоматически, без словаря распознавать возможные ошибки (или новые аббревиатуры). Причем ошибки могут возникнуть как при наборе текста на клавиатуре (опечатки), так и при распознавании рукописного или печатного текста, или при распознавании речи.

Существуют алгоритмы автоматического определения неправильной раскладки клавиатуры: когда Вы набираете текст, например, на английском языке, но при этом раскладка включена русская. Вместо слова «begin» получится «иупшт». Практически все биграммы в этом получившемся по ошибке слове в русском языке не встречаются. Обнаружив это, программа может подать сигнал о неправильном переключении раскладки клавиатуры.

 gJntYwbFk

Вы искали: ПОТЕНЦИАЛ

Поможет частотный анализ и при выборе алгоритма кодировки текста. Текст в памяти компьюте-

ра хранится и передается в виде двоичных чисел. Иногда случаются ошибки декодирования — текст раскодируется неправильно, с помощью другого алгоритма, получается бесполезный набор символов. На этот случай существуют алгоритмы автоматического определения кодировки. Как компьютер может «догадаться», какой алгоритм правильный? Путем статистического анализа текста, исследования его биграмм, триграмм, частот букв — соответствуют ли получившиеся частоты тем, которые определяются в обычных текста на данном языке.

Частотный анализ помогает при автоматическом определении языка текста (например, это делают программы-переводчики) — программа вычисляет, для какого языка наиболее характерны часто встречающиеся в тексте биграммы, могут ли встретиться в этом языке все остальные сочетания букв из текста.

Мы уже говорили, что частотный анализ встречаемости букв можно использовать для дешифровки текста: когда разные буквы заменены разными значками, а одинаковые — одинаковыми, но соответствие значков буквам неизвестно. Определяем самые частые значки — они должны соответствовать самым частым буквам языка, далее смотрим, какие значки стоят рядом с «разгаданными» буквами, «догадываемся» о том, какие это могут быть буквы, анализируя возможные в языке биграммы.

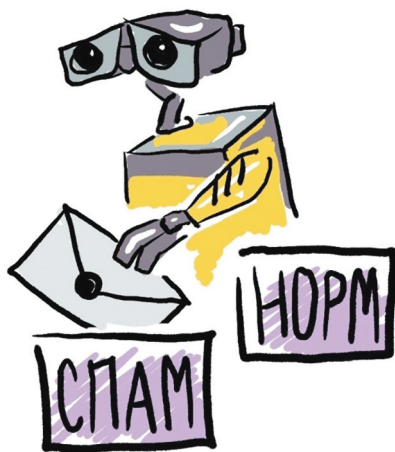
А если посчитать слова?

Конечно, можно посчитать не только биграммы и триграммы, но и группы из 4 или 5 букв. Это делают редко: долгие вычисления, требующие много памяти. Уж лучше посчитать слова. Ведь информация о том, какие слова часто встречаются в тексте, может нам дать ответы на многие вопросы. Правда, с подсчетом слов все непросто. Во-первых, слов очень много. Во-вторых, в русском языке в тексте могут быть использованы разные формы одного слова: например, кот — коты — котов — котами. То есть сочетания букв разные, а слово одно и то же. Возможна и обратная ситуация, когда один и тот же набор букв в разных случаях имеет разные значения: например, слово «мой» в словосочетаниях «где мой учебник» и «мой руки». Поэтому для статистического анализа слов нужна сложная предобработка текста. Такие программы есть, они активно используются.

Что мы можем выяснить, посчитав, какие слова в тексте часто встречаются?

Можно определять направленность, тематику текста: если это научный текст, в нем будет много слов, относящихся к какой-либо научной области. Например, в математическом тексте могут часто встречаться слова «интеграл», «производная», а в медицинском — «заболеваемость», «кровь», «температура».

Можно классифицировать почту, отличать спам от обычных писем. Для этого надо составить список слов, которые наиболее часто встречаются в спам-рассылках, и при этом нечасто — в других.



Посчитав количество «положительных» и «отрицательных» слов, можно догадаться о тональности текста: человек пишет о чем-то хорошем, о том, что ему понравилось, или же, наоборот, он чем-то недоволен. Это направление называется «Анализ мнений», «Оценка тональности текста», используется для автоматического определения оценки по отзывам от зрителей фильмов, покупателей в магазинах и т.п.

Лингвисты используют информацию о словах, входящих в тот или иной текст, для анализа творчества писателя. Исследуют и тексты «обычных» людей — смс-сообщения, блоги, посты из социальных сетей. Путем подсчетов можно установить, каков объем словаря современного человека, как он зависит от возраста, профессии, места жительства. Например, компания Яндекс подготовила исследование про слова, которые используются только в некоторых частях страны, а в других местах будут людям непонятны. («Как го-

ворится: местные слова в разных регионах России» <https://yandex.ru/company/researches/2021/local-words>). Даже по словам из поисковых запросов можно сделать выводы о жизни, интересах людей в той или иной местности: например, если часто спрашивают про кондиционеры, помощь при тепловом ударе — понятно, какая там погода.

Мы не можем здесь привести соответствующие программы, как это сделали для букв. Программы для работы со словами достаточно сложные, требуют много памяти, машинного времени, для предобработки текста часто требуется использование правил лингвистики. Для анализа обычно недостаточно просто распечатать результаты подсчета — слишком много там разных чисел, поэтому для иссле-

дования результатов подсчетов существуют специальные алгоритмы, соответствующие программы, в том числе используются нейронные сети.

Вы можете использовать более простые алгоритмы: подсчитывать частоту встречаемости слов из заранее известного массива. Чтобы учитывать вхождение в текст разных форм слова, можно все формы хранить в этом массиве, а можно подсчитывать вхождения не только слов, но и частей слов. Например, учитывать не только слово «компьютер», но и все слова, в которые это сочетание букв входит, таким образом будут учтены формы «компьютеры», «компьютеров», а заодно и прилагательные типа «компьютерный».

Подумайте, что еще можно подсчитать в тексте? Ответ — в следующем номере.

Новости

Новости

Новости

Новости

Студенты и молодые ученые изучили изменение климата и проанализируют распространение пластикового загрязнения в Арктике в рамках программы «Плавающие университеты»

32 студента и аспиранта разных специальностей из МФТИ, МГУ имени М. В. Ломоносова, СПбГУ, РГГМУ, МИРЭА, РНИМУ им. Н. И. Пирогова, СевГУ, а также ИФА РАН отправились изучать акваторию Карского моря на борту судна «Академик Иоффе». МФТИ и Институт океанологии РАН впервые провели научно-образовательную экспедицию в Арктике по программе «Плавающие университеты». Проект прошел в рамках Года науки и технологий при поддержке Министерства науки и высшего образования России и Фонда целевого капитала МФТИ.

Ключевые исследования прошли в Карском море, при этом были затронуты зоны нескольких рек — Оби, Енисея и Пясины, — проливы Карские Ворота и Вилькицкого, а также глубоководные желоба Святой Анны и Воронина. По словам ученых, данная местность пока мало изучена и активно осваивается. Экспедиция направлена на исследование поступления теплых атлантических вод в северной части Карского моря, предположительно влияющих на потепление климата в Арктике, анализ распространения пластикового загрязнения в Арктике, а также изучение распространения и трансформации стока Великих Сибирских рек Оби и Енисея и их влияния на карбонатную систему моря. Экспедиция прошла с 6 августа по 5 сентября, став площадкой для знакомства и обмена опытом между студентами вузов и учеными. После экспедиции студентов ожидает защита научных проектов в Музее Мирового океана в Калининграде.

Продолжение статьи: <https://mipt.ru/news>