электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

 # 10, октябрь 2015

УДК 004.047

Латентно-семантический анализ теста с применением алгоритма Портера

***Величкевич А.Г.**, студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Компьютерные системы и сети»

***Черепахин А.А.**, студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Компьютерные системы и сети»

Научный руководитель: Андреев А.М., доцент
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Компьютерные системы и сети»
k_iu6@bmstu.ru

Введение

Латентно-семантический анализ — это метод обработки информации на естественном языке, анализирующий взаимосвязь между множеством документов и терминами в них встречающимися, сопоставляющий некоторые факторы (тематики) всем документам и терминам. (В основе метода лежат принципы факторного анализа, в частности выявление латентных связей изучаемых явлений или объектов) [1].

Основные применение данного анализа:

• сравнение двух терминов между собой;

• сравнение двух документов между собой;

• сравнение термина и документа.

В настоящие время многие поисковые службы в интернете активно используют данную методику. Он позволяет находить тексты похожие по смыслу. Латентно-семантический анализ отображает документы и отдельные слова в так называемое «семантическое пространство», в котором и производятся все дальнейшие сравнения. При этом делаются следующие утверждения:

1) Документы - это просто набор слов. Порядок слов в документах игнорируется. Важно только то, сколько раз то или иное слово встречается в документе.

2) Семантическое значение документа определяется набором слов, которые, как правило, идут вместе. Например, в биржевых сводках, часто встречаются слова: «фонд», «акция», «доллар»

3) Каждое слово имеет единственное значение. Это, безусловно, сильное упрощение, но именно оно делает проблему разрешимой.

1. Выбор библиотеки для реализации алгоритма Портера

Перед тем как перейти к примеру анализа, рассмотрим реализацию алгоритма Портера для стемминга. Стемминг — это процесс нахождения основы слова для заданного исходного слова. Основа слова необязательно совпадает с морфологическим корнем слова. Данные алгоритм написан на многих языках программирования, но не реализован на C++. Его реализация была произведена, взяв за основу код на JAVA [2].

Для реализации данного алгоритма понадобились регулярные выражения для работы со строками. В связи с тем, что в стандартом сборке C++ нету регулярных выражений, пришлось выбрать между двумя библиотеками (BOOST и PCRE), в которых они есть.

В ходе анализа библиотек использовались исследования, проведенные компанией boost, которая сравнила время поиска регулярных выражений в английском тексте [3].

Для начала был взят текст объемом 19Мб, в котором производился поиск регулярных выражений, и измерялось время нахождения в различных библиотеках (см. таблицу 1).

Таблица 1

Время нахождения регулярных выражений в тексте, объема 19Мб

Регулярное выражение	Библиотека BOOST, с	Библиотека PCRE, с
Twain	$T_{b1B} = 3{,}49$	$T_{p1B} = 1$
beman|john|dave	$T_{b2B} = 1{,}03$	$T_{p2B} = 1{,}16$
^[^]*?Twain	$T_{b3B} = 1$	$T_{p3B} = 3{,}32$
Tom|Sawyer|Huckleberry|Finn	$T_{b4B} = 1{,}02$	$T_{p4B} = 1{,}08$

Затем был взят текст объемом 50Кб, в котором был произведен то же подсчет (см. таблицу 2).

Таблица 2

Время нахождения регулярных выражений в тексте, объема 50Кб

Регулярное выражение	Библиотека BOOST, с.	Библиотека PCRE, с.
Twain	$T_{b1S} = 1{,}8$	$T_{p1S} = 1$
beman|john|dave	$T_{b2S} = 0{,}5$	$T_{p2S} = 0{,}8$
^[^]*?Twain	$T_{b3S} = 1$	$T_{p3S} = 2{,}82$
Tom|Sawyer|Huckleberry|Finn	$T_{b4S} = 1$	$T_{p4S} = 1{,}28$

В ходе измерений был подсчитал средний результат:

$$T_{\text{ср } BOOST} = \left(\sum_{i=1}^{4} T_{biB} + \sum_{i=1}^{4} T_{biS}\right)/2.$$

Откуда $T_{\text{ср } BOOST} = 1{,}36\ c.$

$$T_{\text{ср } PCRE} = \left(\sum_{i=1}^{4} T_{piB} + \sum_{i=1}^{4} T_{piS}\right)/2.$$

Откуда $T_{\text{ср } PCRE} = 1{,}56\ c.$

В результате исследования была выбрана библиотека BOOST, так как она более быстрая в поиске регулярных выражений в тексте и имеет более высокий функционал.

2. Реализация алгоритма Портера

Рассматриваемый алгоритм не использует баз основ слов, в отличие от других алгоритмов стемминга. Он применяет последовательно ряд правил, отсекая окончания и суффиксы, основываясь на особенностях языка, в связи с чем работает быстро, но не всегда безошибочно. Данный алгоритм работает по следующим правилам [2]:

1) При поиске окончания из всех возможных выбирается наиболее длинное. Например, в слове «чтение» выбираем окончание «ие» а не «е».

2) **Rv**-область слова после 1 гласной, **r1**- область слова после первой гласной-согласной, **r2**-область **r1** после первой гласной-согласной. Все проверки производятся над областью **rv**. Например, в слове «необъятный» области будут такие: **rv**: «объятный»,

r1: «ъятный», **r2**: «ный». Ниже приведена схема-алгоритма Портера (см. рис. 1).

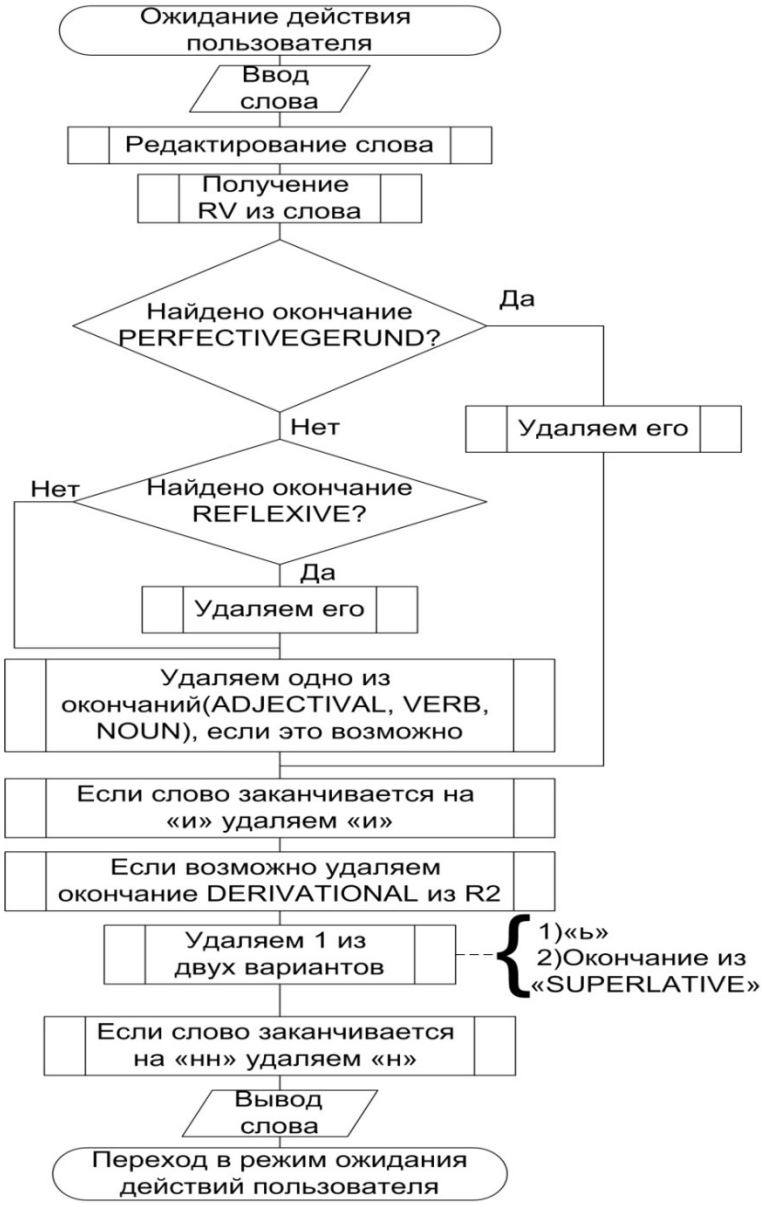

Рис. 1. Схема-алгоритма Портера

Регулярные выражения, которые применяются в алгоритме, представлены ниже. В них описаны наборы букв, на которые может заканчиваться слово, относящиеся к соответствующей части речи:

PERFECTIVEGERUND (деепричастие совершенного вида)

((ив|ивши|ившись|ыв|ывши|ывшись)|((?<=[ая]*)(в|вши|вшись)))$**;

REFLEXIVE (возвратный глагол) (ся|сь)$;

ADJECTIVE (прилагательное)

(ее|ие|ые|ое|ими|ыми|ей|ий|ый|ой|ем|им|ым|ом|его|ого|ему|ому|их|ых|ую|юю|ая|яя|ою|ею)$

PARTICIPLE (причастие)

((ивш|ывш|ующ)|((?<=[ая])(ем|нн|вш|ющ|щ)))$;

VERB (глагол)

((ила|ыла|ена|ейте|уйте|ите|или|ыли|ей|уй|ил|ыл|им|ым|ен|ило|ыло|ено|ят|ует|уют|ит|ыт|ены|ить|ыть|ишь|ую|ю)|((?<=[ая])(ла|на|ете|йте|ли|й|л|ем|н|ло|но|ет|ют|ны|ть|ешь|нно)))$;

NOUN (существительное)

(а|ев|ов|ие|ье|е|иями|ями|ами|еи|ии|и|ией|ей|ой|ий|й|иям|ям|ием|ем|ам|ом|о|у|ах|иях|ях|ы|ь|ию|ью|ю|ия|ья|я)$;

DERIVATIONAL (Слова с двумя корнями)

(ост|ость)$;

SUPERLATIVE (прилагательное превосходной степен)

(ейше|ейш)$;

* - наборам букв, указанным справа от (?<=[ая]), должна предшествовать буква «а» или «я».

** - символ, указывающие на проверку нахождения набора букв в конце слова.

3. Пример латентно-семантического анализа

Далее рассмотрим пример анализа. Было выбрано девять заголовков с различных новостей. Были взяты те заголовки, в которых есть повторяющиеся слова. Далее из заголовков были исключены все стоп-слова (союзы, частицы, предлоги и другие слова, не имеющие смысла). Были убраны слова, встречающиеся только в одной статье. Этот этап упрощает математическое вычисление. Далее была произведена операция стемминга для остальных слов с использованием алгоритма Портера. Если ее не производить, вычисления будут намного сложнее. В итоге у нас остались, так называемые, индексируемые слова, они выделены жирным шрифтом:

1. Британская **полици**я знает о местонахождении **основател**я **WikiLeaks**.
2. В **суд**е **США** начинается процесс **прот**ив россиянина, рассылавшего спам.
3. **Церемони**ю **вручени**я **Нобелевск**ой **премии** мира бойкотируют 19 **стран**.
4. В **Великобритани**и **арестова**н **основател**ь сайта **Wikileaks** Джулиан Ассандж.
5. Украина игнорирует **церемони**ю **вручени**я **Нобелевск**ой **премии**.
6. Шведский **суд** отказался рассматривать апелляцию **основател**я **Wikileaks**.
7. НАТО и **США** разработали планы обороны **стран** Балтии **прот**ив России.

8. **Полиц**ия **Великобритан**ии нашла **основателя WikiLeaks**, но, не **арестова**ла.

9. В Стокгольме и Осло сегодня состоится **вручени**е **Нобелевск**их **прем**ий.

Далее составляется матрица термин документ, в которой строки это индексируемые слова, а столбцы это номера заголовков. В каждой ячейке пишем, сколько раз встретилось слово в той или иной статье. Матрица-термин документ представлена в таблице 3.

Таблица 3

Матрица термин-документ

	T1	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**
wikileaks	1	0	0	1	0	1	0	1	0
арестова	0	0	0	1	0	0	0	1	0
великобритан	0	0	0	1	0	0	0	1	0
вручен	0	0	1	0	1	0	0	0	1
нобелевск	0	0	1	0	1	0	0	0	1
основател	1	0	0	1	0	1	0	1	0
полиц	1	0	0	0	0	0	0	1	0
прем	0	0	1	0	1	0	0	0	1
прот	0	1	0	0	0	0	1	0	0
стран	0	0	1	0	0	0	1	0	0
суд	0	1	0	0	0	1	0	0	0
сша	0	1	0	0	0	0	1	0	0
церемон	0	0	1	0	1	0	0	0	0

Следующим шагом мы проводим сингулярное разложение полученной матрицы. Сингулярное разложение - это математическая операция, раскладывающая матрицу на три составляющих. Т.е. исходную матрицу M мы представляем в виде [4]:

$$M = U \cdot W \cdot V^T.$$

Где W – диагональная матрица из корней собственных чисел матрицы $M \cdot M^T$, расположенных в порядке убывания. Диагональные элементы матрицы W называются сингулярными числами. Матрица U составляется из собственных векторов для $M \cdot M^T$. Матрица V составляется из собственных векторов для $M^T \cdot M$, где U и V^T – ортогональные матрицы. В итоге получаем (см. рис. 2):

wikileaks	0,57	-0,01	0,01	-0,2	0,13	0,16	-0,16	-0,25	-0,64
арестова	0,34	0	0,07	0,41	-0,42	-0,02	0,1	0,17	0,01
великобритан	0,34	0	0,07	0,41	-0,42	-0,02	0,1	0,17	-0,01
вручен	0	0,52	0,07	-0,06	-0,08	-0,15	-0,17	0,02	-0,07
нобелевск	0	0,52	0,07	-0,06	-0,08	-0,15	-0,17	0,02	0,32
основател	0,57	-0,01	0,01	-0,2	0,13	0,16	-0,16	-0,25	0,64
полиц	0,31	0	0,05	0,07	0,57	-0,6	0,29	0,37	0
прем	0	0,52	0,07	-0,06	-0,08	0,15	-0,17	0,02	-0,25
прот	0,02	0,03	-0,61	0,13	-0,05	-0,22	0	-0,25	0
стран	0,01	0,22	-0,31	0,39	0,41	0,56	-0,22	0,4	0
суд	0,12	0,01	-0,38	-0,62	-0,3	0,12	0,21	0,55	0
сша	0,02	0,03	-0,61	0,13	-0,05	-0,22	0	-0,25	0
церемон	0	0,38	0,03	0,02	0,08	0,31	0,82	-0,29	0

•

3,41	0	0	0	0	0	0	0	0
0	3,3	0	0	0	0	0	0	0
0	0	2,27	0	0	0	0	0	0
0	0	0	1,49	0	0	0	0	0
0	0	0	0	1,19	0	0	0	0
0	0	0	0	0	0,98	0	0	0
0	0	0	0	0	0	0,71	0	0
0	0	0	0	0	0	0	0,43	0
0	0	0	0	0	0	0	0	0

•

T1	T2	T3	T4	T5	T6	T7	T7	T9
0,43	0,05	0,01	0,54	0	0,37	0,01	0,63	0
0	0,02	0,65	-0,01	0,59	0	0,09	-0,01	0,47
0,03	-0,07	-0,04	0,06	0,1	-0,16	-0,67	0,09	0,09
-0,22	-0,24	0,15	0,28	-0,11	-0,68	0,44	0,33	-0,13
0,69	-0,32	0,22	-0,49	-0,12	-0,03	0,27	-0,02	-0,19
-0,27	-0,34	0,44	0,29	-0,13	0,45	0,12	-0,31	-0,45
-0,03	0,3	0,14	-0,17	0,44	-0,15	-0,3	0,24	-0,71
-0,3	0,12	0,4	-0,39	-0,53	0,12	-0,23	0,46	0,13
0,35	0,35	0,35	0,35	-0,35	-0,35	-0,35	-0,35	0

Рис. 2. Сингулярное разложение матрицы термин-документ

Особенностью сингулярного разложения является то, что оно выделяет ключевые составляющие матрицы, позволяя игнорировать шумы [4]. Согласно простым правилам произведения матриц, видно, что столбцы и строки соответствующие меньшим сингулярным значениям дают наименьший вклад в итоговое произведение. Например, можно отбросить последние столбцы матрицы U и последние строки матрицы V^T, оставив только первые две. При этом гарантируется оптимальность полученного произведения. Разложение такого вида называют двумерным сингулярным разложением (см. рис. 3).

wikiliaks	0,57	-0,01
арестова	0,34	0
великобритан	0,34	0
вручен	0	0,52
нобелевск	0	0,52
основател	0,57	-0,01
полиц	0,31	0
прем	0	0,52
прот	0,02	0,03
стран	0,01	0,22
суд	0,12	0,01
сша	0,02	0,03
церемон	0	0,38

•

3,41	0
0	3,3

•

T1	T2	T3	T4	T5	T6	T7	T8	T9
0,43	0,05	0,01	0,54	0	0,37	0,01	0,63	0
0	0,02	0,65	-0,01	0,59	0	0,09	-0,01	0,47

Рис. 3. Двумерное сингулярное разложение

Теперь отметим на графике точки соответствующие отдельным текстам и словам (см. рис. 4).

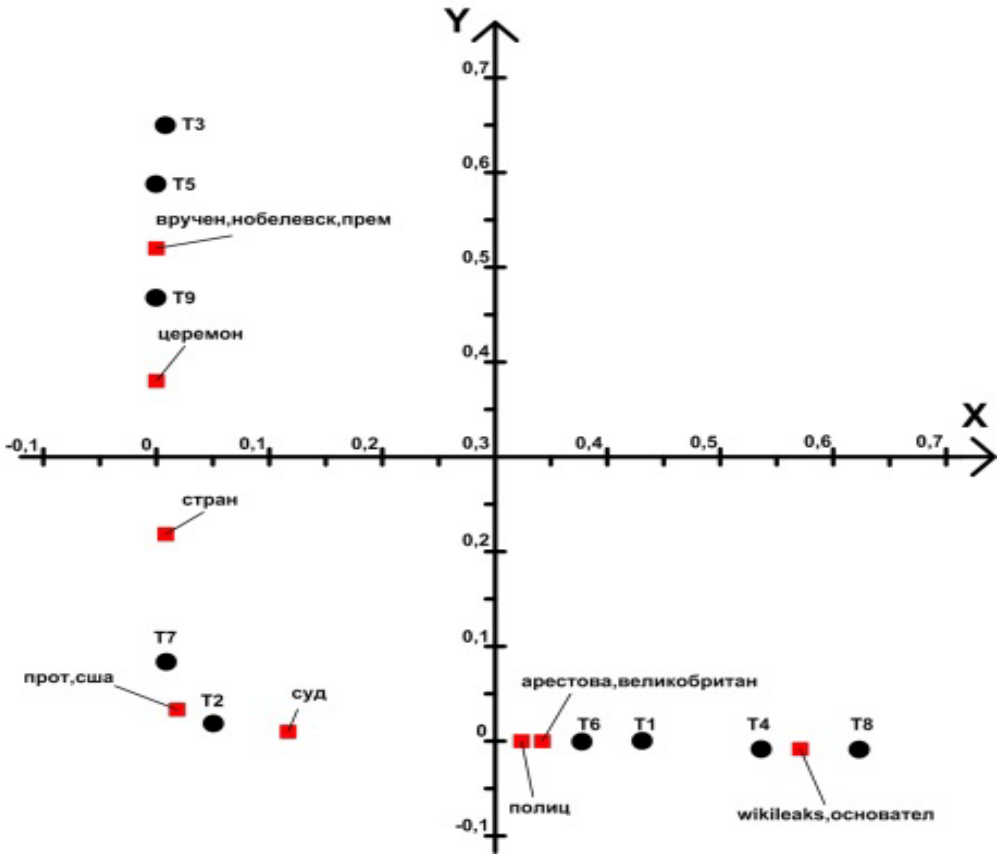

Рис. 4. Семантическое пространство

Из данного графика видно, что статьи образуют три независимые группы. Первая группа статей располагается рядом со словом «wikileaks» и «основател», и действительно, если посмотреть названия этих статей становится понятно, что они имеют отношение к wikileaks и к основателю. Другая группа статей образуется вокруг слова «США», и действительно, в них идет обсуждение США. В связи с этим можно сказать, что произведен кластерный анализ текста.

Заключение

На практике, конечно, количество групп будет намного больше, пространство будет не двумерным а многомерным, но принцип остается тот же. Можно определять местоположения слов и статей в пространстве и использовать эту информацию для определения тематики статьи.

Этот подход используется в поисковой машине Google. Например, запрос «apple computer» будет интерпретирован как относящийся к компьютерной компании Apple, поэтому среди результатов будут документы, описывающие технологии этой компании (даже если эти документы не содержат слов «apple» или «computer»).

В ходе исследования был реализован алгоритм Портера для стемминга на языке C++ и освоена математическая модель, лежащая в основе метода латентного семантического анализа, а также изучен используемый в нём математический аппарат. Были проанализированы статьи с использованием рассматриваемого алгоритма.

Список литературы

1. Латентно-семантический анализ. Режим доступа: https://ru.wikipedia.org/wiki/Латентно-семантический-анализ (дата обращения 10.05.2015).
2. Стеммер Портера для русского языка. Режим доступа: http://www.algorithmist.ru/2010/12/porter-stemmer-russian.html (дата обращения 15.05.2015).
3. Сравнение библиотек. Режим доступа: http://www.boost.org/doc/libs/1_41_0/libs/regex/doc/gcc-performance.html (дата обращения 12.05.2015).
4. Кристофер Д. Маннинг, Прабхакар Рагхаван, Хайнрих Шютце. Введение в информационный поиск. М.:Издательство Вильямс, 2011. 506 с.