Предварителна обработка на текста
Булев модел на търсене

1. Предварителна обработка на текста

Обикновено преди да се пристъпи към даден алгебричен или вероятностен модел за анализ на текст, е необходимо да се извърши предварителна обработка на текста, за да се премахнат определени думи и символи, които могат да внесат много шум в анализа и да повлияят силно върху достоверността на резултатите. В списъка долу са посочени петте основни операции за предварителна обработка на текста. Желателно е поне първите четири да бъдат реализирани.

1. ***Премахване на пунктуационните символи***. В документите, които подлежат на анализ и търсене, най-вероятно има множество пунктуационни символи – *точки, запетаи, тирета, кавички, апострофи* и др. Те са важни за конструирането на изреченията и техния синтаксис, но от семантична гледна точка не носят никаква допълнителна информация. Не могат по никакъв начин да кажат за какво се отнася даден документ. Това, че не помагат само по себе си не е причина да се премахнат. Проблемът е, че пречат. Особено когато се докосват до думи. Тогава пречат за разпознаване на думите. Например една дума, непосредствено след която има тире или точка се разпознава като различна спрямо *същата* дума, след която няма такъв символ. Ако пък тези символи са разделени от думите с интервали, тогава се разпознават като уникални думи, с което безсмислено се увеличава размерността на векторите. Затова е най-добре да бъдат премахнати. Това става най-лесно с регулярен израз (например следния perl-съвместим /[\p{P}]/u) или с функция за заместване в символни низове.
2. ***Преобразуване на всички букви в малки***. Дали една дума ще бъде написана с малки букви, с големи букви или със смесени малки и големи букви, това не променя нейното значение по никакъв начин. Но при разпознаването на думите, буквите имат значение. Имат освен, ако не се преобразуват всички в малки (за предпочитане) или в големи букви. Желателно е да се направи такова преобразуване както за всички документи, така и за заявката. Това ще гарантира, че едни и същи думи винаги ще се откриват и разпознават, независимо с какви букви са написани.
3. ***Токанизация на документа***. Т.е. преобразуване на текста в масив/списък от уникални думи.
4. ***Премахване на семантично-незначимите думи*** (stop words). Това основно са предлози, съюзи, местоимения, наречия и др. Но съвсем не само такива части на речта. Те също имат важно синтактично значение, но по никакъв начин не представят смисъла и предметната област на документите. Освен това се срещат изключително често, което е проблем – ще имат несъразмерно голяма стойност на *tf* характеристиката си спрямо семантично-значимите думи, които наистина носят информация за смисъла и съдържанието на документите. Т.е. семантично-незначимите думи ще обезличат

семантично-значимите. Това донякъде може да се реши с използването на *idf*, но само до някъде и не в случаите на множество документи в една предметна област. Затова е най-добре тези думи просто да се премахнат. Обикновено те се дефинират предварително като списък, и разбира се са зависими от езика. За английски са едни, за български други, за гръцки – трети и т.н. Списъкът се зарежда като масив. В последствие, когато се обработва даден документ, просто се проверява дали текущата дума фигурира в този масив, и ако да, се игнорира.

5. ***Стемиране (stemming)***. Това е операция по отделяне на окончанията от корена на думите. Целта е в анализа да се използват само корените. Например ако в различни документи се съдържат думите *хубав*, *хубаво*, *хубава*, *хубави*, *хубавица*, и не са стемирани, всички те ще се разпознаят като различни думи. Но е ясно, че това е една и съща дума, просто в различни родове и числа, или производни на думата. Следователно има смисъл да се разпознаят като една дума, не като различни. За целта трябва да се отдели коренът на думата или да се премахнат окончанията. Това разбира се, е операция, която е силно езиково-зависима, защото правилата за формиране на окончанията в различните естествени езици са различни. Съществуват различни стемиращи алгоритми. За *английския език*, най-използваният стемър е на Martin Porter [3]. В Интернет могат да се намерят готови реализации на този алгоритъм на множество различни програмни езици. Разбира се има и други стемиращи алгоритми за английски. Например този на Chris Paice, наречен Lancaster Stemming Algorithm [2]. И за *български език* има подобен алгоритъм – BulStem [1], предложен от Преслав Наков.

2. Булев модел на търсене

Най-елементарният метод за търсене по съдържание е посредством т.нар. булев модел (Boolean retrieval). Той позволява търсене по думи, които са взаимосвързани чрез *логически изрази* „И“ и „ИЛИ“, като може и да се ползва инверсия „НЕ“.

Например, ако потребителят търси самолети, произведени от Boeing, трябва да напише следната заявка:

самолет И *Boeing*

Ако търси всички документи, в които се споменава Boeing или какъвто и да е друг самолет, заявката би била:

самолет ИЛИ *Boeing*

Ако се търсят всички самолети, които не са Boeing или Airbus, тогава заявката ще бъде:

самолет И (НЕ(*Boeing* ИЛИ *Airbus*))

Булевият модел за търсене е изключително използван. На практика всички системи за управление на бази от данни (СУБД), особено тези, управляващи релационни бази от данни (СУРБД) разчитат на него. Независимо дали търсенето е с *пълно съвпадение* (т.е. *атрибут=”стойност”*) или с *частично съвпадение* (търсене на подниз в низ, т.е. *атрибут LIKE “%стойност%”*), СУБД реализира именно този модел за търсене.

Популярността му е мотивирана от следните негови предимства:

- Лесен за разбиране.
- Лесен за реализация.
- Изчислително ефективен.

Но администраторите на бази от данни сигурно знаят и колко ограничения притежава, които обуславят и неговите недостатъци:

- Заявките трябва да се представят във вид на булеви изрази, което не винаги е лесно. Всъщност, при голям брой думи е изключително трудно, поради което, обикновено се използва *само за прости заявки, с малко думи*.
- В случай на много думи, като резултат се връщат *или огромно количество намерени* документи, ако думите са свързани с ИЛИ (ниска прецизност), *или изключително малко или дори никакви* документи, ако думите са свързани с И (ниска откриваемост).
- Поведението му е наистина булево – документът е или намерен, или не. *Не се изчислява коефициент на подобие*, следователно *резултатите не се ранкират, т.е. подреждат по степен на подобие.* Липсата на подредба означава, че е възможно някой много подобен/адекватен документ да се върне много назад в списъка с резултати, а други, по-неадекватни и не толкова свързани със заявката, да се изведат по-напред.

В резюме, булевият модел е подходящ за *търсене само по единични думи*, но не и по цели изречения, параграфи и документи. Липсата на изчислено подобие със заявката и подредба на резултатите не гарантира, че по-малко релевантни документи няма да бъдат изведени по-напред в резултатите от други значително по-адекватни документи.

При подготовката на този файл са използвани материали от:

Калмуков, Й. Методи и алгоритми за търсене и извличане на документи. Издателство Primax Русе, 2022 г., ISBN 978-619-7242-93-5

Литература

1. Nakov, Preslav. "BulStem: Design and evaluation of inflectional stemmer for Bulgarian." In Workshop on Balkan Language Resources and Tools (Balkan Conference in Informatics). 2003.
2. Paice, Chris D. "Another Stemmer." ACM SIGIR Forum 24.3 (1990): 56-61
3. Porter, Matrin F. An algorithm for suffix stripping. In J. S. Karen and P. Willet, editors, Readings in information retrieval, pages 313-316. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997