Информационный поиск

Константин Мосиенко

Yandex School of Data Analysis *konstantin.mosienko@gmail.com*

2017

Содержание

Литература

- C.D. Manning, P. Raghavan, H. Schutze. Introduction to Information Retrieval [2008]
- B. Croft, D. Metzler, T. Strohman. Search Engines: Information Retrieval in Practice [2009]
- S. Buttcher, C. L. A. Clarke and G. V. Cormack. Information Retrieval: Implementing and Evaluating Search Engines [2010]
- https://en.wikipedia.org/wiki/Information_retrieval

Конференции

- TREC (Text Retrieval Conference)
- CLEF (Cross Language Evaluation Forum)
- WWW (World Wide Web Conference)
- ESSIR (European Summer School in Information Retrieval)
- SIGIR (Special Interest Group on Information Retrieval)
- WSDM (Web Search and Data Mining)
- CIKM (Conference on Information and Knowledge Management)

Определение

Определение

Информационный поиск – это область научных исследований, ориентированная на изучение структуры, организации, хранения, поиска и извлечения информации. [G. Salton, 1968]

Определение

Информационный поиск — процесс поиска неструктурированной документальной информации, удовлетворяющей информационные потребности, и наука об этом поиске. [C. Manning, 2011]

Мы будем рассматривать вопросы, касающиеся поиска по интернет сайтам.

Запрос - документ

Определение

Документ - это информационная сущность, которая хранится в базе поисковой системы (индексе). Процесс занесения документа в индекс - индексация. Документом могут быть: локальные файлы различных форматов, html-страницы, видео, аудио, картинки.

Определение

Запрос - способ выражения информационных потребностей. Обычно запрос задаётся с помощью языка запросов соответствующей поисковой системы.

Что умеет поисковая система

- Находить и скачивать документы.
- Детектировать язык и кодировку. Извлекать информацию из документов различных форматов.
- Оценивать частоту обновления документа.
- Находить в своей базе похожие документы и спам.
- Быстро отвечать на запросы к своему индексу.
- Ранжировать результаты поиска по релевантности.

Определение

Релевантность - семантическое соответствие поискового запроса и найденного документа.

Два поиска

Слово «поиск» может употребляться в контексте разных задач:

- Поиск в имеющейся базе. Например, поиск релевантных запросу документов в индексе поисковой системы. Базовая операция - перечисление документов, содержащих определённое слово(словосочетание).
- Обнаружение кандидатов на занесение в индекс. Например, поиск в интернете отсутствующих в индексе(новых) документов. Базовая операция - перечисление документов, на которые есть ссылки с имеющегося документа.

Некоторые особенности и сложности

- Информация доступна в неструктурированном с точки зрения индексирования виде: например, как понять, где на странице важный текст, а где рекламный блок?
- Пользователь не всегда ищет текст, он может искать и видео.
- Актуальность. Необходимо иметь как можно более точный «слепок» интернета. Быстро находить новую информацию и не забывать удалять не актуальную.
- Региональность. Один запрос, заданный из разных мест, иногда должен приводить к разным результатам. Например, если вы заказываете пиццу.

Базовые компоненты

- Поисковый робот. Скачивает документы из интернета, обнаруживает новые документы, планирует очередь скачки(так как обычно нет возможности скачать все известные документы, необходимо сделать выбор, какие обойти сейчас, а какие, может быть, никогда).
- Индексатор. Обрабатывает скаченные документы, строит поисковый индекс.
- Поиск. Отвечает на запросы пользователей, генерирует статистику.

Масштабы трагедии

Абсолютные показатели различных экспериментов не совпадают, поэтому необходимо смотреть на отношения.

- Согласно косвенным показателям, количество страниц, доступных для индексирования, в 2005 году составляло 11.5 миллиарда, в 2009 году - 25 миллиардов.
- В соответствии с исследованиями 2001-го года, большая часть документов интернета - 550 миллиардов - не обнаружена поисковыми системами, эту часть называют DeepWeb.
- В 2008 году Google знал 1 триллион уникальных URL-ов.

Так как нет возможности положить в индекс такое количество документов, современная поисковая система производит поиск по десяткам-сотням миллиардов документов.

Uniform Resource Locator

scheme:[//[user:password@]host[:port]][/]path[?query][#fragment]

Один URL можно записать разными способами:

- Схема и имя хоста не чувствительны к регистру.
- Можно не писать стандартный порт.
- Вместо символа можно написать его код через %

HTTP

GET /index.html HTTP/1.1
Host: www.example.com

HTTP/1.1 200 OK
Date: Mon, 23 May 2005 22:38:34
Content-Type: text/html;
charset=UTF-8
Content-Encoding: UTF-8
Content-Length: 138
Last-Modified: Wed, 08 Jan 2003
23:11:55 GMT
...

HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
    <title>HTML Document</title>
  </head>
  <body>
    <p>
      <b>
        Этот текст будет полужирным,
        <i>а этот — ещё и курсивным</i>
      </b>
    </p>
  </body>
</html>
```

Инвертированный индекс

Определение

Инвертированный индекс - это структура данных, хранящая для каждого слова список документов, в которых это слово встречается.
Постинг лист - вышеупомянутый список документов.

В инвертированном индексе можно ещё хранить и удобно получать доступ к таким данным:

- Свойства самого слова. Например, число его вхождений в корпус.
- Свойства слова и документа. Например, число вхождений слова в документ.

Постинг листы обычно хранят отсортированными по идентификатору документа для ускорения поиска.

Булев поиск

Определение

Булев поиск - первая и самая простая модель информационного поиска. Основывается на выполнении теоретико-множественных операций над списками документов в соответствии с запросом.

- Пусть дан запрос вида $q = (t_1|t_2)\&t_3...$
- На первом шаге необходимо для каждого терма запроса t_i с помощью инвертированного индекса получить список документов, содержащих этот терм.
- На втором шаге необходимо выполнить указанные в запросе операции с полученными множествами документов.

Недостатки булева поиска

- Находит только документы, точно соответствующие запросу. Например, для запроса $q = t_1 \& t_2$ если какой-то документ содержит только терм t_1, он не найдётся даже если остальные документы не содержат ни одного слова из запроса.
- Не ранжирует результаты поиска.
- Все слова для поиска имеют одинаковую важность, что не соответствует действительности.

Расширенный булев поиск

Недостатки простого булева поиска можно устранить введя в рассмотрение веса термов и модифицировав процедуру поиска:

- $q = \{(t_1, w_{q1}), ..., (t_n, w_{qn})\}$ - термы запроса со своими весами.
- $d = \{..., (t_1, w_{d1}), ..., (t_n, w_{dn}), ...\}$ - вхождения термов запроса в документ, веса соответствующих термов относительно документа.

Замечания:

- Схема выставления весов не является частью модели.
- $w_{\{q|d\}i} \in [0, 1]$
- Документ может и не содержать определённые термы запроса, для таких термов $w_{di} = 0$.

Расширенный булев поиск

Предлагается от простого отношения «слово запроса входит в документ» перейти к учёту весов термов для построения метрики «близости» запроса и документа:

- $sim(d, q = t_1 \& ... \& t_n) = 1 - \left(\sum_{i=1}^{n} w_{qi}^p (1 - w_{di})^p\right)^{\frac{1}{p}} \left(\sum_{i=1}^{n} w_{qi}^p\right)^{-\frac{1}{p}}$
- $sim(d, q = t_1 | ... | t_n) = \left(\sum_{i=1}^{n} w_{qi}^p w_{di}^p\right)^{\frac{1}{p}} \left(\sum_{i=1}^{n} w_{qi}^p\right)^{-\frac{1}{p}}$

Замечания:

- p - параметр модели и требует подбора.
- Указанные формулы необходимо рекурсивно применять в соответствии с разбором запроса на элементарные булевы функции, используя в качестве веса для некоторой подформулы значение функции *sim* на ней.

Векторная модель документа

- Документы и запросы представляются в виде разреженных векторов размерности $|T|$ (размер словаря, количество термов):

$$d_i = \{w_{d_i 1}, w_{d_i 2}, \ldots, w_{d_i |T|}\}$$

$$q = \{w_{q1}, w_{q2}, \ldots, w_{q|T|}\}$$

- Каждая компонента вектора соответствует появлению определённого терма: если $w_{d_i j}$ отличен от нуля, значит терм t_j встретился в документе d_i.
- Метрика схожести документа и запроса(или другого документа) - косинус угла между их векторными представлениями.

Векторная модель документа

Замечания

- На веса никакие ограничения не накладываются.

Плюсы (относится и к расширенному булевому поиску)

- Учитывает веса слов.
- Допускает отсутствия слов запроса в документе.
- Позволяет ранжировать результаты.

Недостатки (относится и к расширенному булевому поиску)

- Модель подразумевает, что слова появляются в тексте независимо.
- Не учитывается порядок слов.
- Не учитывается смысл документа - если важное слово заменить на синоним, документ может перестать быть релевантным с точки зрения модели.

Расчёт весов термов

Веса термов влияют на разные метрики близости документов.

- Чем больше вес - тем больше вклад в метрику.
- Поэтому хочется давать большой вес «важным» словам.

Как понять, что слово важное?

- Если слово запроса часто встречается в документе, стоит считать его важным.
- Если только это слово не встречается часто во всех документах.

Учтя вышесказанное, возьмём в рассмотрение следующие характеристики:

- Частота терма в документе (tf).
- Доля документов с данным термом (df).

Term Frequency

$f_{t,d}$ - количество вхождений терма t в документ d. $|d|$ - общее количество термов в документе. $tf(t,d)$ - способ придать терму вес относительно данного документа. Возможны варианты:

- $tf(t,d) = \{0,1\}$ (входит / не входит).
- $tf(t,d) = f_{t,d}$.
- $tf(t,d) = f_{t,d}/|d|$, $tf(t,d) = f_{t,d}/max_{t' \in d} f_{t',d}$.
- $tf(t,d) = 1 + log(f_{t,d})$.
- $tf(t,d) = K + (1-K) f_{t,d}/max_{t' \in d} f_{t',d}$, $K \in [0,1)$.

Выбор конкретной схемы зависит от задачи, например, для расширенного булева поиска необходимы веса из $[0,1]$.

Inverse Document Frequency

$|D|$ - общее количество документов в коллекции D (корпусе). n_t - количество документов, в которых встретился терм t. $idf(t)$ - способ придать вес терму относительно всей коллекции документов, указывающий на количество информации, которое несёт появление терма:

- $idf(t) = \log \frac{|D|}{n_t}$.
- $idf(t) = \log \frac{|D|}{n_t+1}$.
- $idf(t) = \log \frac{max_{t' \in d} n_{t'}}{n_t+1}$.
- $idf(t) = idf(t) / max_{t' \in d} idf(t')$.

TF-IDF

$$tfidf(t, d) = tf(t, d)idf(t)$$

- С одной стороны, компонент $tf(t, d)$ увеличивает вес с увеличением количества вхождений терма в документ.
- С другой стороны, компонент $idf(t)$ стремится к нулю при увеличении доли документов, в которых встретился терм.
- $tfidf(t, d)$ максимален для самого частотного терма t, который встречается только в документе d. Можно считать, что такой терм идеально характеризует свой документ.

Вероятностная модель

В основе модели лежит попытка вероятностно-статистически обосновать понятие релевантности документа запросу и вычислить вероятность того, что пользователь оценит данный документ как релевантный. Воспользовавшись Байесовскими правилами, можно сделать следующие выводы:

- Если $P(R = 1|D) > P(R = 0|D)$, можно считать документ D релевантным.
- $P(R|D) = P(D|R)P(R)/P(D)$.

Все вычисления произведены в условиях наличия некоторого запроса. Проблемой является вычисление вероятностей $P(D|R)$.

Бинарная модель независимости

- С целью сделать $P(D|R)$ вычислимой на практике, предполагается независимость появления термов в документе: $P(D = d|R) = \prod_{i=1}^{|d|} P(t_i|R)$.
- Изначально этот результат использовался для выставления весов термов для векторной модели:

$$w_i = \log \frac{p_i(1 - u_i)}{u_i(1 - p_i)}$$

Где p_i - вероятность встретить терм t_i в релевантном документе, а u_i - в нерелевантном.

Бинарная модель независимости

Теперь проблемой является вычисление $p_i = P(t_i|R = 1)$ и $u_i = P(t_i|R = 0)$. Подход к решению данной проблемы зависит от доступных данных:

- Если на этапе настройки для каждого запроса есть список релевантных документов, можно явно оценить вероятности через частоты с учётом независимости термов.
- Если имеется информация о том, какие документы релевантны некоторым запросам (не известно каким), то можно считать, что распределения термов различаются только между релевантными/нерелевантными документами.
- Если ничего не известно, то можно самим попытаться восстановить множество релевантных документов, например, более строгой моделью поиска.
- ...

Okapi BM25

$$sim(d, q) = \sum_{i=0}^{|q|} idf(t_i) \frac{tf(t_i, d)(k+1)}{tf(t_i, d) + k((1-b) + b\frac{|d|}{average|d_j|, d_j \in D})}$$

- Okapi - поисковая система, созданная в Лондонском городском университете. BM - best match.
- k и b - подбираемые параметры.
- По сей день может использоваться как фактор для более сложных функций ранжирования.

Что мы не рассмотрели?

- Latent Semantic Indexing.
- Обобщённая векторная модель.
- Вариации на тему BM25.