Московский Государственный Университет им. М.В.Ломоносова
Факультет Вычислительной Математики и Кибернетики

Конспект лекций по курсу
Введение в информационный поиск

Составил:
студент 520 группы
Нокель Михаил Алексеевич

Москва, 2010 г.

Содержание

1 Лекция 3

27 октября 2010 г.

Приведём вначале несколько терминов.

Для каждого термина *координатный индекс* хранит перечень всех позиций, где он встретился в документе.

Лексема – экземпляр последовательности символов в документе, объединённой в синтаксическую единицу для разбора.

Тип – класс всех одинаковых лексем.

Термин – тип, включённый в словарь информационного поиска.

Лексикон – множество всех терминов, включённых в словарь системы информационного поиска.

Общая задача у нас такова: на вход подаётся последовательность символов, которую надо разбить на отдельные лексемы (в простейшем случае – на слова). Слова затем выделяются в словоформы. Также очень важно хранить информацию о том, в каком падеже и в каком месте встретилось слово.

Основные алгоритмы[1]:

- Построение индекса сортировкой и группировкой. Это простейший алгоритм. Входные данные (получаются при разборе ддокументов): термины и соответствующие им идентификаторы документов, в котором они встретились. Основные стадии таковы:

  1. Алфавитное упорядочивание или лексико-графическая сортировка
  2. Группировка[2]

  Надо бы ещё учесть и то, как эти алгоритмы работают на конкретном аппаратном обеспечении (особенно ввиду большого объёма данных). Особенности аппаратного обеспечения таковы:

  1. Доступ к данным в памяти гораздо быстрее, чем к данным на диске. Часто используемые данные должны храниться в кеше, памяти, дисковом кеше. К сведению, у крупных поисковых систем все индексы хранятся в оперативной памяти. При чём индексы распределённые – на каждой машине хранится лишь часть индекса.
  2. Специфичные свойства диска: данные хранятся блоками. Если надо прочитать информацию с диска, то читаются все данные из блока. До этого ещё надо ещё спозиционировать магнитную головку диска. Это всё время, много времени.
  3. При работе с жёстким диском тоже есть кеш, особенности которого могут повлиять на производительность.
  4. Чем быстрее память, тем её меньше, и наоборот. Кеша меньше оперативной памяти, оперативной памяти меньше, чем дисковой. Но есть и исключение – кеш дисковой памяти меньше оперативной памяти.

[1] Вместо термина используется понятие идентификатора

[2] Хранится также ещё и куча дополнительных параметров терминов(указание контекста, падежа)

5. На процессоре что-то считается, а память обменивается данными с диском → нет простоя системы.

- Блочное индексирование, основанное на сортировке (BSBI). Если все данные не помещаются в памяти, то их надо разделить на блоки, которые помещаются, обработать блоки и потом объединить. У этого алгоритма много общего с сортировкой слиянием. Основные стадии работы алгоритма таковы:

  1. Разделение на части или сегментация коллекции
  2. Сортировка пары термин-идентификатор документа (аналогично простейшему алгоритму)
  3. Запись отсортированных блоков на диск
  4. Объединение всех получившихся блоков, их слияние. Тем самым получаем общий инвертированный индекс

  Блоки, как правило, выбираются одинакового размера. Для каждого блока строится свой инвертированный индекс, который потом записывается в отдельный файл. А в конце работы алгоритма эти индексы объединяются и на диск записывается итоговый файл. При этом обработка может идти параллельно для каждого блока. В качестве дополнительной информации для терминов могут записываться различные характеристики (в том числе и языковые).

- Однопроходное индексирование в оперативной памяти (SPIMI). Здесь используются термины, а не идентификаторы. Также не используется общий словарь для всех терминов, а для каждого блока динамически строится свой словарь. Если термин есть в словаре, то информация о нём добавляется к существующей. Если же термин встречается в первый раз, то он добавляется в словарь, и новые вхождения данного термина будут записываться именно сюда[3] . Все эти особенности позволяют экономить память по сравнению с предыдущими вариантами. От сортировки, присущей прошлому алгоритму, можно избавиться, поскольку каждый раз, когда встречается слово, уже известно, куда его записывать. Так как количество встреч конкретного слова не превосходит числа слов в документе, то из-за отсутствия сортировки достигается приличная экономия.

  Ещё одна особенность: если память заканчивается, то заканчиваем обрабатывать текущий блок, берём следующий и начинаем строить новый инвертированный индекс для него. Объединение блоков происходит аналогично предыдущему алгоритму, за исключением того, что сливаются не только слова, но и словари.

Что делать в случае, если документы меняются (как и сами документы, так и их количество)? Предыдущие алгоритмы рассматривают лишь статику,

[3]Надо учесть, что словарь реализуется в виде хеша → возникают проблемы с упорядоченностью.

но не динамику. В последнем случае надо как-то обновлять индекс. Самый простеший способ, который приходит в голову, - это не правка индекса (ибо это очень дорогая операция, поскольку надо двигать тесно укомплектованные блоки), а перестроение индекса (этим методом пользуются и крупные поисковые системы).

Другой подход заключается в построении вспомогательного индекса (дополнительного или индекса обновления). Для обработки запроса используются и основной, и вспомогательный индесы, а результаты объединяются. Когда вспомогательный индекс разрастается, надо проводить слияние двух индексов. Но эту операцию можно трактовать и как перестроение индекса.

Как быть, если документ устарел или был удалён? Из-за того, что есть дополнительная информация, то можно поставить флаг об этом. Но, к примеру, у Яндекса есть сервис для RSS-потоков, в архиве которого удалённая информация остаётся.

Есть также проблема безопасности индекса. Кому-то информация доступна, кому-то нет. Пример: поиск в корпоративной среде. Предположим, что у нас есть разные документы разного уровня доступа, к которым есть доступ у разных пользователей. Как обеспечить релевантную выдачу результатов? Если есть доступ у пользователя, то включаем в результаты выдачи, иначе - нет (в этом случае пользователь даже не знает о существовании документа). Реальные примеры: закрытые сети Одноклассники, Контакт. Для реализации этого механизма используются списки контроля доступа (ACL).

Поговорим теперь про *сжатие*. Хотелось бы больше данных в памяти размещать $\rightarrow$ для этого надо сжимать их. Есть простейшие методы сжатия, идея которых – подсчитать сколько раз слово встретилось, записать это число и 1 раз написать, что это за слово. Этот метод сжатия нетрудоёмкий и даёт приличный выигрыш.

2 Лекция 5. Лингвистические технологии в информационном поиске

24 ноября 2010 г.

Этапы лингвистического анализа (от простого к сложному)[4]:

1. Графематический анализ (токенизация)
2. Морфологический анализ
3. Синтаксический анализ
4. Семантический анализ
5. Прагматический анализ

[4]В разной литературе разные названия этапов, а некоторые (например, графематический и морфологический - допрагматический анализ) объединяются. Для информационного поиска актуальные первые два этапа

Почти всё в web передаётся по протоколу гипертекста, поэтому в тексте содержатся, в частности, типы MIME. Существует несколько разных классов типов, но нас интересуют: text и application (плоский текст, тексты разметки, pdf и прочие .doc файоы). На этапе извлечения информации получается плоский текст (возможно, с сохранением форматирования). Что касается текстов разметки, там содержатся специальные теги: для определения языка и т.п. Помимо формата надо определить язык и кодировку - может быть явно сказано, а может быть и нет $\rightarrow$ надо определять самому тогда.

Основные способы их определения:

1. Словарные методы (редко используются на практике)

2. Статистические методы (сопоставление кодов символов). Основаны на совместной встречаемости букв. Основываясь на этом и на знании таких распределений, можно определять язык. В основном считается на основе триграмм (троек символов). На их основе осуществляется определение языка во всех современных браузерах. Недостатки: если мало данных, то не работает. Пример: если браузер на коротких текстах ошибается, то он выдаёт турецкий язык (а не русский).

Теперь уже можно переходить к лингвистическим вопросам. На входе: поток символов, для которых известны язык и кодировка (от форматов уже избавились).

Напомним определения, введённые раньше.

Лексема (токен) - экземпляр последовательности символов, представляющих семантическую единицу для обработки текста

Тип - набор лексем, обладающих определёнными свойствами.

Например, *Ty-154* - это либо два, либо три токена (зависит от системы).

Необходимо этот поток символов побить на лексемы (токены). Введём два неэквивалентных определения:

Лексема (токен) - это последовательность символов между двумя разделителями.

Разделитель - пробел, перевод строки, возврат каретки, табуляция, знаки препинания (но иногда учитываются - в зависимости от того, как формируются словари) и служебные символы (например, C++). Точки могут быть разделителями, а могут и не быть (пример: А.С.Пушкин).

Далее следует *морфологический анализ*. Его задача - определить основную форму слова и морфологические характеристики данного слова. Иногда достаточно определить лишь основную форму слова (она попадает в словарь и далее индексируется).

Методы морфологического анализа гораздо более интересны. Они различаются для разных языков. Они делятся на:

1. Словарные

   (a) Словари основ

   (b) Словари окончаний (изменяемых частей)

2. Бессловарные

Языки отличаются изменением слов. В некоторых языках это происходит путём изменения самого слова (например, в русском), а в других - с помощью предлогов (например, в английском).

Для английского языка достаточно хорошо работает *стемминг*. Самый известный - стеммер Портера.

Стемминг - отсечение изменяющейся части слова. Например, для слова "крыша"отсекается окончание.

Стемминг может приводить лишь к некоторой начальной основе. Для определения того, что отсекать используется как статистика, так и словари изменяемых частей слов. Для простых запросов стемминг повышает полноту поиска.

Проблема: неправильные глаголы (в английском языке), глагол (идти). В немецком языке сложные термины строятся из объединения простых.

Что касается *словарного* способа, то используется словарь основ и словарь окончаний. Составить словарь для всего невозможно. Классический пример: "Верхневартовск". Используется *модель аналогии*. Если известно изменение аналогичного слова, то логично применить к нему правила измения того известного слова.

Пример: слово "кровать". Предположим, что оно неизвестно. Аналогия: "дескать "печать "пировать".

Как видно, однозначностью не пахнет. Надо либо выбрать лучше, либо использовать все. Чем больше используем, тем лучше полнота, но хуже точность.

Итак, вернёмся к определению начальной формы слова:

- "стеки"→ стек, стечь
- "стекла"→ стекло, стечь
- "стеками"→ стек

Пример: баг Яндекса: "Сочи это глагол, не существительное.

Вдогонку скажем, что к начальной форме слова можно поставить часть речи (существительное, глагол) и его признаки (падеж, склонение, род).

Переходим плавно к *синтаксическому анализу*. Задача: определить синтаксическую структуру некоторой единицы текста (будем считать, предложения) и построить в том или ином виде дерево разбора. Аналогия: разбор предложения в школе.

Рассмотрим пример: "Пакетик чая залить кипящей водой". Залить "что?"пакетик, пакетик "чего?"чая, залить "чем?"водой, водой "какой?"кипящей. И получается дерево разбора: Залить → пакетик → чая Залить → водой → кипящей

Методы, решающие данную задачу:

1. Работа на основе шаблонов. Шаблоны - информация о том, какие характеристики слов встречаются подряд (глагол, существительное, падеж, род).
2. Выделение формальной грамматики. Естественный язык очень сложно описать формальной грамматикой. Естественно появляются примеры, на которых всё ломается.

Построение такой структуры позволяет часть проблем убрать с неоднозначностью. Пример остающихся неоднозначностей: "поднять упавшего на рельсы". Как минимум 2 интерпретации. С точки зрения здравого смысла, здесь всё однозначно.

Остался последний уровень - *прагматический*. Это самый глубокий уровень понимания совсем странных и неодинаковых вещей. Определяется, что такие словосочетания, как "Олег Лупанов "декан мехмата"надо рассматривать вместе. Для этого нужны определённые дополнительные знания из предметной области.

Теория закончилась. Движемся к практике. Существуется такое понятие, как "тест Тьюринга тест на предмет того, можем ли мы отличить машину от человека. Нюанс: разные люди разговаривают по-разному.

На практике применяются самые разные методы искусственного интеллекта. ИИ (по Игорю Ашманову) - ещё не решённые интеллектуальные задачи. Как только задача решилась, то она оттуда убирается. На практике используются колдунчики, вопросно-ответный поиск, сервисы автоматического реферирования (Яндекс.Новости).

Ещё есть частная задача - выделение имён собственных. При выделении по шаблонам возникают забавные вещи: "Ясная Любовь это человек; "МГУ Анатолий "Город Энгельс Саратовской области".

Лингвистические задачи: автоматический или автоматизированный перевод. Есть текст на одном языке, надо перевести его на другой. Можно разобрать текст, построить его модель и перевести на другой. Это классический текст. В последнее время начали применяться статистические методы (google translate: "кустарник поздравил сезон рыбной ловли "я эстонец"(I am russian)).

Ещё есть лингвистическая задача: spell-checker (проверка на грамотность и поиск опечаток). Словарные слова надо писать, как они есть в словаре. С точки зрения лингвистики, словарное слово - это слово, что есть в словаре. С точки зрения информационного поиска, словарное слово - это любое часто встречаемое слово. Например, "афтар"$\rightarrow$ "аффтар". Алгоритмы поиска опечаток: словарные методы и N-граммы ("шч "аь"и др.). Word предлагал поделить "мультик анальный"и "культ урология". Есть ещё слова (абсценные), которые есть в словаре, но предлагать их в качестве замены нельзя. Ещё есть проверка на знаки препинания.

Всё это может использоваться в рамках информационного поиска для борьбы и генерации спама.

3 Лекция 6. Модели информационного поиска

01 декабря 2010 года

Модели:

1. Булевская. Самая старшая (50-е годы прошлого века) и самая простая. Модель релевантности основана на том, содержится ли слово запроса в слове документа и термы запросов можно группировать с помощью логи-

ческих операторов (NOT, AND, OR). Пример: npol and not судья. До 90х годов прошлого века недалеко от этого ушли.

2. Векторная. Идея векторной модели в том, что каждый документ и запрос можно представить в виде вектора в некотором векторном пространстве термов. В каждый момент времени словарь конечен. Все термы можно упорядочить в лексикографическом порядке и перенумеровать. Каждая координата означает, присутствует ли данный термин в документе или нет (1 или 0). Таким же образом можно представить и запрос. При таком представлении документов, какая-то часть информации теряется - в каком месте документа данное слово было расположено. Пример: "А лучше Б"и "Б лучше А"одно и то же. Эта модель "bag-of-words". TF (частота терма) - сколько раз данный термин встретился в данном документе. Но не все термины одинаково полезны. Для решения данной проблемы вычисляется IDF (обратная частота документа): $IDF = \log \frac{N}{DF}$, где N - общее число документов в коллекции, DF - число документов, в которых данный термин встретился. Посылка к данному способу определения весов: если число встречается слишком часто, то оно не самое важное. Пример: коллекция про автомобили. Слово "дизель"будет иметь более высокий общий вес, поскольку оно более специфическое. Самый часто используемый вес - это $TF \times IDF$. Если термин встречается в некоторой небольшой группе документов, то его вес растёт и это соответствует большей специфике данного документа.

   Также помимо наивной информации в каждой координате можно хранить число вхождений терма в документ (его можно каким-либо образом нормировать). В рамках данной векторной модели можно определить понятие близости между документами и его формализовать. Из линейной алгебры известно достаточно много мер близости, но в задаче информационного поиска наибольшее распространение получила косинусная близость. Пусть имеются 2 вектора A и B, соответсвующие двумя документам. Можно посчитать сходство между этими документами как $sim(A, B) = \frac{(A,B)}{||A|| \times ||B||}$, что соответствует косинусу между векторами в соответствующем пространстве. По св-ву: $(A, B) = \sum_{i=1}^{n} (a_i \times b_i)$.

3. Вероятностная. Идея: у некоторого документа есть своя вероятность соответствия запросу. Выдаются наиболее вероятные документы.

4. Ссылочная. Стала возможной в результате появления веба (в нём куча различных ссылок). Сюда относится PageRank и некоторые другие методы, основанные на гиперссылках. Все они появились в конце 90-х годов прошлого века. В качестве какой-то предтечи для них явились индексы научного цитирования. Идея в том там, что чем лучше научная статья, тем больше у неё ссылок. Кроме того, есть журналы, в которых эти хорошие статьи группируются.

   Рассмотрим PageRank ("модель миллиона обезьян"). Идея: в вебе все до-

кументы достаточно сильно связаны ссылками и тот документ, на который ссылаются, лучше, чем тот, на который не ссылаются. Формула расчёта PageRank: $PageRank(p) = (1 - \alpha) \times \sum_{p,q \to \inf} \frac{PageRank(q)}{links(q)}$. Считается, что каждый конкретный пользователь с некоторой вероятностью перейдёт на другую, на которую есть ссылка из данной. Если нет ссылок из данной страницы, то пользователю придётся набрать некоторый адрес в строке адресной. Поэтому и возникает коэффициент.

Марковская цепь - случайное блуждание. Допустим, есть человек, который может пройти за единицу времени либо единицу вперёд, либо единицу назад. То, куда он может попасть, зависит от его текущего положения. Если человек идёт в обе стороны с равной вероятностью, то в бесконечности он останется на месте. Иначе - может в соответствующую сторону сдвинуться. Положение зависит только от текущего состояния и не зависит от пути. Возможны различные расширения за счёт углубления по времени.

Число страниц в каждый конкретный момент времени конечно. Считается, что в начальный момент времени человек находится в начальной точке, а далее то, куда он перейдёт, зависит лишь от данной точки, где находится, и не зависит от того, как он туда пришёл. Это наивный подход, на практике всё сложнее - имеет место субъективный фактор.

Рассмотрим другие модели, например метод HITS (порождаемый гиперссылками тематический поиск - Hyperlink Induced Topic Search), который был предложен Клейнбергом. Он обнаружил, что не все ссылки одинаково полезны - есть некоторая закономерность в расстановке ссылок. В частности, хорошие страницы ссылаются на хорошие гораздо чаще, а также то, что все страницы делятся на хабы и источники. Источники - это те страницы, которые содержат непосредственно интересующую нас информацию, а хабы - это те страницы, которые содержат достаточно много ссылок на хорошие интересующие нас страницы. На практике таким образом можно найти хороших источников и хороших последователей. Исчточники хорошие, если на них ссылаются хорошие посредники, и наоборот.

В рамках данной модели все страницы поделены на хабы и источники. Ценность страницы рассчитывается как $h(V) = \sum_{v \to y} (a(y))$ и $a(V) = \sum_{y \to v} (h(y))$.

Любым из методов данной группы прежде всего мы получаем данные о ссылочной структуре, а не о качестве. Забавные случаи: по запросу "жалкий неудачник"выдавался первым результатом сайт Джорджа Буша, аналогично "тупой ублюдок". Это было сделано расставлением ссылок на официальные сайты некоторой группой лиц. Ещё один пример: по запросу "грёбаный враг"выдавался сайт майкрософта. Официальный сайт Лукашенко выдавался по запросу "жопа "жопа Минска "жопа Беларуссии".

Ссылки бывают неестественные (спам) и естественные. Естественные говорят о том, что они хорошие. По поводу спама: один магазин ничего не поставлял клиентам, в результате чего сильно поднялся в рейтинге по товарам (так как на него писали кучу негативных отзывов).

5. Лингвистическая. Она основана на порождающих грамматиках. Выбираются грамматики, которые могут породить определённое число документов.

4 Лекция 7. Вопросно-ответные системы

8 декабря 2010 г.

Вопросно-ответный поиск - поиск на естественном языке.

Вопросно-ответная система - система, основанная на поиске на естественном языке.

Фактографический поиск - поиск какой-то конкретной информации. Т.е. на вопросительные запросы непосредственно предоставляются ответы

Классификация вопросно-ответных систем:

1. Поиск осуществляется по закрытому домену
2. Поиск осуществляется по открытому домену

Классификация вопросов:

1. Фактографические - "В каком году произошла Октябрьская революция?". "Простой вопрос "простой ответ".
2. Простые доказательные - "Как погиб Тьюринг?"Ответ - "отравился".
3. Синтезирующие. Идея: для ответа необходим синтез нескольких документов (для ответа). Пример: "Какая поисковая система занимает первое место по числу посещения пользователей?"или "Какие школы есть в Москве?"
4. С интерактивным контекстом. Здесь с системой идёт диалог: задаются подсказки, наводящие вопросы. Пример: "Какова единица измерения массы?А в чём измеряется масса?А не в кг ли?". Или: "Какие есть школы в Москве?Какие есть специализированные школы в Москве?"
5. Умозрительные (speculative). Важна ещё есть и прагматика: помимо текста есть ещё и подтекст. "Придёт ли Аня на следующую лекцию?"

Поговорим теперь об архитектуре систем:

1. Обработка вопроса. На вход получили вопрос. Надо понять, о чём идёт речь. Надо определить тип вопроса по вопросу. Могут помочь вопросительные слова: "Кто? "Что? "Когда?"...
2. Переформулировка вопроса или разбиение на несколько маленьких
3. Расширение запроса. Информация в документе может быть сформулирована другими словами, а не словами запроса. Пример: "Bill Klinton и Уильям Клинтон один и тот же человек; синоним "машина"и "автомобиль".

4. Выделение возможных ответов.

   (a) Поиск релевантных документов. В документах как правило нет вопросительных слов

   (b) Поиск или выделение фрагментов (passage)

   (c) Выделение предложений

   (d) Выделение потенциальных ответов

5. Выбор ответов. Есть кандидаты, надо среди них всех выбрать наиболее подходящий. Простейший критерий - частотность.

6. Повышение достоверности. Идея: есть ответ, надо улучшить. Иногда бывает, что здесь выясняется, что ответа нет. Способы решения: ослабить требования (релаксация) и возвратиться на предыдущий результат (попробовать проделать всё то же самое, но с меньшими требованиями).

В индекс документов можно помещать именные сущности. Также можно оптимизировать в сторону концпетуального поиска.

Поговорим о выделении ответов:

1. На основе статистики

2. На основе некоторых шаблонов и проверки на соответствие текстов шаблонам. Пример: если пробуем выдать дату рождения и получаем "родился, вылупился то, очевидно, здесь получаем требуемую информацию. Шаблонов очень много и составляются они вручную, что требует гигантской работы. Были попытки оптимизации с автоматической генерацией кандидатов в шаблоны и потом выдавать экспертам для разметки: "шаблон"/"не шаблон". Если генерируем простые шаблоны, то можно генерировать по нескольку штук за час. Если же речь идёт о каких-то глобальных шаблонах, то может уходить и месяц, а то и несколько.

3. На основе машинного обучения. Некоторый набор текстов размечается вручную экспертами: как-то "данное слово имеет такие-то морфологические характеристики "здесь такие-то факты и для всего этого создаётся разметка. Есть и плюсы (уменьшение ручного труда), есть и минусы (трудно исправлять абсурды).

4. Глубокая обработка естественного языка (понимание). Это классические методы искусственного интеллекта.

Большинство систем часто выдают странные результаты, но некоторые успешно применяются на практике.

Поговорим более подробно про фактографические вопросы. *Извлечение информации* - извлечение структурированной информации из неструктурированных текстов. Пример: есть некий текст без семантической разметки. Из этого текста выделяются какие-то даты, временные сущности и строится база. По ней

уже в дальнейшем может осуществляться поиск. Методы извлечения информации более-менее похожи на вопросно-ответный поиск.

На практике самая известная система - Яндекс.Новости. Есть такой сервис - выделение портрета человека: для данного человека извлекаются все факты (дата рождения, род деятельности). Теперь несколько примеров выделения недостоверной информации. Пример: "Александра Солженицына "По словам Марии Ивановой овцы новой породы "Подозреваемого Ивана Козлова в расстреле дикого козла "Иван дурак царь"и др.

Несколько слов теперь о существующих вопросно-поисковых системах. Есть несколько систем, говорящих о том, что они интеллектуальны, но это не совсем так. Достаточно задать такие вопросы: "Что курил автор? "литературные новеллы и многое другое".

Пример: интеллектуальная поисковая система - "Нигма". Есть и другие вопросно-ответные поисковые системы: Ask.Yandex.ru, ответы на Mail.ru, ответы на Google. Они устроены на более простом способе. Суть: вопросы задают люди и отвечают тоже они. Из всего этого можно найти и правильные ответы. Ask.Yangex.ru - был, но сейчас фактически нет.

5 Лекция 8. Поиск по изображениям

15 декабря 2010 г.

Классификация:

1. По содержанию. Поиск на основе информации, содержащейся в графическом файле.

2. По описанию. Поиск по сопровождающей информации.

Виды информации в изображении[5]:

1. Семантика. Любые данные о изображении, если они есть. Большая часть поисковых движков по изображениям использует как раз текстовую информацию. Учитывается информация о теге, задающем изображение. Затем учитывается текст из окружения картинки. Если в самой картинке нет ничего из того, что её окружает, то картинка всё равно будет находиться по соответствующему запросу. Проблема - семантический разрыв. Содержательная часть изображения с точки зрения человека отличается от формального машинного представления соответствующих изображений и между ними лежит огромный разрыв. К примеру, человеку очень легко понять, есть ли на картинке лицо или нет, а машине приходится использовать специальный алгоритм для этого. Или по цветовым характеристикам изображения корабля на океане и самолёта в небе почти не отличаются.

2. Форма объектов. Необходимо представлять изображение так, чтобы изменения одного и того же объекта были бы минимальными. Этот способ

[5]уровень информативности растёт снизу вверх

должен быть устойчив к различным преобразованиям (переносам и т.д.). Здесь используется метод "границы областей". Объект образует некоторую замкнутую область и эта область отличается от окружающего пространства. Один из способов - цепные коды. По изображению строится контур из отрезков, близких к форме соответствующего обьекта. И дальше для такой ломаной строится цепной код на основе направления изменения отрезков. Каждому направлению ставится соответствующая цифра. Таким образом, при обходе соответствующего контура получается некоторое формальное описание.

3. Текстура. Главные "грабли затруднение с формальным определением текстуры. Используется информация между различными областями текстуры.

4. Цвет и яркость. Самый простой и тупейший метод - создать контрольную сумму (fingerprint), т.е. посчитать количество пикселов на изображении. Какая-то информация получится, но цветов достаточно много. Частичное решение - нормирование (сводится к некоторому фиксированному количеству цветов). Близкие контрольные суммы дают и достаточно различные картинки. Близкие объединяются в кластеры и считается, что они не различаются. Кластеры могут группироваться как учитывая особенности восприятия цветов человеком, так и не учитывая. Очевидное улучшение алгоритма: изображение разбивается на области и распределение цветов считается для каждой области. Принцип: чем больше, тем лучше. Тем самым можно построить матрицу на основе цветовой информации.

   Помимо цвета каждого пиксела учитывается и его яркость. Для этого строится матрица смежности, показывающая степень перехода от одного пиксела к другому, от одной области к другой.

Сделаем важное замечание, которое достаточно тривиально. Прежде чем начинается работа над изображением, изображение приводится к некоторому стандартному виду (к стандартному размеру).

Некоторое время назад начали использовать контекстную информацию по изображению. Ляп: бомж на свалке и запросы "Отдых в Подмосковье". Почти во всех поисковых системах используется совместное применение методов. Ещё пример. Самые малые изменения в запросе приводят к существенным изменениям в поисковой выдаче.

5.1 Оценка качества информационного поиска

Вспомним несколько определений.

Релевантность - мера соответствия результатов обработки запроса самому запросу.

Формальная релевантность - это релевантность, вычисленная автоматически.

Информационные потребности пользователя - то, что пользователь хочет. Они могут быть классифицированы так:

1. Неосознанная - пользователь не знает, что он хочет.

2. Осознанная

3. Сформулированная

4. Запрос

Разница между запросом и осозанной потребностью может быть радикальна (пользователь не хочет сообщить системе о каких-то словах, которые ему нужны и пытается добраться иносказательно). По частоте здесь доминирует поиск по фактографическим материалам. Но иногда явная формулировка запрос может приводить к нежелательным последствиям. Одно из обвинений было задано логом в Google на тему "отравление".

Существуют оценки информационного поиска:

1. Академическая. Работает, когда нет чёткого понимания пользовтаельских задач. Не представляет интереса в качестве представления объективных данных. Мало применима на практике. Комментарий: физику дали построить модель для производства молока. Его работа начинается со слов "рассмотрим сферическую корову в вакууме".

2. Индустриальная. Пользовательские задачи определены строго в рамках интерфейса. Необходима оценка того, как работает система в зависимости от пользователей. Бюджет значительно богаче. Должна отображать объективную реальность. Здесь оценивается также и количество запросов. Весьма значимо то, что выдаёт система для среднего пользователя.

Поговорим теперь о проектах и направлениях в области информационного поиска. Поговорим о конференциях, котоыре существуют в мире:

1. TREC. Американская конференция (с 1992 года).

2. CLEF. Европейский проект (с 2000 года).

3. РОМИП. Российский проект по оценке методов инфомрационного поиска (с 2003 года).

На каждой такой конференции производится оценка результатов различных информационных систем для разных задач. Существуют различные дорожки и системы пытаются их решить на общем наборе и затем производися оценка. В плане коллекций следует упомянуть: текстовая коллекция 11го рейха (от 200000 документов). Эти конференции представляют как начную, так и практическую ценность. Результаты оцениваются вручную, эти люди называются ассесорами. Они приводят оценки релевантности для каждого документа.

Рассмотрим шкалы релевантности документа:

1. Витальный - документ, полностью удовлетворяющий потребности пользователя. Не для всех запросов существуют витальные документы. Например, сайт сбербанка, если мы искали информацию о нём.

2. Релевантный (сильно релевантный). Отличается от слабо релевантных степенью отношения. Пример: если интересует страна в какой-то период, то сильно релевантный - сайт о климате в стране, слабо релевантный - сайт, повествующий о чём-то другом, но в котором упоминается нужная инфомация.

3. Слабо релевантный

4. Нерелевантный

5. Невозможно оценить. Ассесор не смог это сделать.

6. Вредный (спам и т.д.)

Самая первая коллекция, на которой производилась оценка - коллекция, содержащая от 1000 до 1500 документов. Современные коллекции отличаются размерами в гигабайты и сотни гигабайт.

Здесь становится важным такое понятие, как согласие ассесоров. Существуют разные мнения на данный предмет. При оценке иногда используется такая терминология: сильно релевантный - документ, который признали все ассесоры, а слабо релевантный - тот, который признал хотя бы один ассесор. Подобную терминологию необходимо уточнять в начале соответствующей статьи.

На практике применяется часто метод "общего котла для каждого запроса собираются ответы (некоторое количество среди первых выдачей систем). Затем берутся документы, которые выдала хотя бы одна поисковая система, убираются дубликаты, оцениваются. А оставшиеся документы выкидываются, поскольку их не выдала ни одна поисковая система.

К примеру в том же Яндексе есть отдельный запрос ассесору "является ли данный документ порнографическим, и в какую категорию попадает спам".

5.2 Численные методы оценки информационного поиска

Полнота и точность - 2 основных понятия.

Все документы делятся на релевантные и нерелевантные, найденные и ненайденные. Введём обозначения:

a - релевантные найденные

b - нерелевантные найденные

c - релевантные ненайденные

d - нерелвантные ненайденные

Полнота - $R = \frac{a}{a+c}$

Точность - $P = \frac{a}{a+b}$

Можно ввести ещё 2 термина:

1. Аккуратность - число правильных решений к общему числу решений: $A = \frac{a+d}{a+b+c+d}$

2. Ошибка - число неправильных решений к общему числу решений: $E = \frac{b+c}{a+b+c+d}$

Отсюда понятно, что одновременно система достигать хороших результатов по полноте и точности не может: один увеличиваем, другой - уменьшается. Полнота - 100% (выдали пользователю все документы).

На практике используется F-мера: $F = \frac{1}{\frac{1}{P}+\frac{1}{R}}$. Свойства F-меры:

1. $0 \leq F \leq 1$
2. $min(P, R) \leq F \leq \frac{R+p}{2}$

Также используется F-мера с параметрами (в предыдущем случае считаем, что и R и P имеют одинаковый вес, но это не всегда так):

$$F_\beta = \frac{(\beta * \beta + 1) \times P \times Q}{\beta * \beta \times P + R}$$

и

$$F_\alpha = \frac{1}{\frac{\alpha}{P} + \frac{(1-\alpha)}{R}}$$

и

$$\beta * \beta = \frac{1-\alpha}{\alpha}$$

6 Лекция 9

22 декабря 2010 г.

Информация о курсе на сайте leozub.ru/courses/ir2010/

Усреднение:

1. Микроусреднение - найти общее количество документов, удовлетворяющее всем сразу, и посчитать метрику
2. Макросуреднение - для каждого в отдельности посчитать, а потом усреднить.

Точность на уровне n документов - количество релевантных документов среди первых n документов выдачи, делённое на n. n для разных задач - разное.

Следующая метрика - R-точность. Допустим, что для данного запроса существует n релевантных документов. R-точность - точность на уровне n документов. Для разных запросов точность может радикально отличаться, поэтмоу имеет смысл делать такое усреднение.

Средняя точность $AvgPrec = \frac{1}{K} \sum\limits_{i=1}^{k} (prec_r el(i))$, где $prec_r el(i)$ - точность на уровне i-го релевантного документа, равная $p(pos(i))$, если i-й релевантный документ есть в выдаче; и 0, иначе. Средняя точность - нормированная сумма точностей на уровне i-го релевантного документа.

Свойства средней точности:

1. Средняя точность $\leq$ полнота

2. Если релевантные документы лишь в начале списка выданных результатов, то средняя точность равна полноте

3. Если релевантные документы равномерно распределены, то средняя точность равна произведению точности и полноты.

4. Документы, ранжированные ниже последнего учитываемого документа, не влияют. Они просто отсекаются.

Предыдущие документы рассчитываются исходя из априорных знаний о количестве документов. Но это не всегда так (см. прошлую лекцию). Была введена метрика *Bpred*. Пусть R - число известных релевантных документов, а r - какой-то из релевантных документов. $Bpred = \frac{1}{R}\sum_r(1 - \frac{NonRelBefore(r)}{R})$, где NonRelBefore(r) - количество документов, ранжированных не выше r.

Ещё используется метрика $Bpred\_10 = \frac{1}{R}\sum_r(1 - \frac{NonRelBefore(r)}{10+R})$. Позволяет более осмысленно оценивать качество соответствующей системы, если число релевантных документов невелико.

Следующее понятие: *Ценность ответа* - $ReciporalRank = rank(pos(rel))$. Эта метрика предназначена для определения количества усилий, требуемых пользователю, чтобы добраться до релевантного документа пользователю. Эта метрика часто применяется для вопросно-ответного поиска. Для разных конференций разные ранги:

- Для TREC – {1, 0.5, 0.33, 0.2, 0.1, 0, ...}
- Для РОМИП - {1, 0.9, ... , 0.1, 0, ...}

Рассмотрим график Полнота/Точность. Берутся фиксированные значения полноты и смотрятся значения точности для фиксированного значения полноты. Графики строятся разными способами, самые распространённые - 11точечные графики (и в РОМИП, и в TREC, но немного отличаются интерполяцией результатов). Откладываются интервалы от 0 до 1 с шагом $\frac{1}{10}$. График обычно кривой с перепадами (вид пилообразный). При построении графика для каждого фиксир. значения полноты ставится точка. Если добавили ещё 1 релевантный документ, то всё растет. А если добавили нерелвантный, то полнота та же самая, а точность падает. Обычно учитывается ещё и интерполяция.

6.1 Поисковая оптимизация и методы борьбы с ней

Ашманов - достаточно известная личность в области информационного поиска в России, был одним из руководителей Ramblera. Когда из Rambler разбежались все разработчики, основал свою компанию, которая была одной из первых в области поисковой оптимизации. Сейчас разрабатывают тематические поисковые системы, пытаются внедрить интеллект в поисковые системы. Основная книжка по оптимизации - книга Ашманова. Из любопытных фактов: Ашманов - сын профессора ВМК.

Теперь к оптимизации. Точного определения нет. Оптимизация - это задача поднять видимость какого-то сайта в выдаче поисковых систем по некоторым запросам.

Классификация оптимизации[6]:

1. Белая - нет никаких запрещенных технологий. Это разумное движение. Повышая качество самого сайта, повышаем место в выдаче. Можно почитать справку Яндекса.

2. Серая - середина. Выходим из чистой зоны в сторону спама. Явного запрета нет.

3. Чёрная - разные виды спама.

Перейдём к более конкретным вещам и технологиям.

1. Дорвей- страница, созданная для выдвижения страницы в выдачу, содержит редирект, генерируется автоматически. Борьба с помощью специальных скриптов. Основной способ генерации текста с помощью дорвей - автоматичсекий, с помощью марковских цепей, и результат существенно отличается от текста, составленного человеком. На основе марковских цепей получается текст, синтаксически более-менее хороший, но по семантике бардак. Как бороться? Здесь учитываются различные глобальные характеристики (в том числе и жанровые). И пытаются определить, насколько текст похож на данный стиль или нет. Считаются десятки или сотни таких характеристик. Можно применять различные методы машинного обучения. Признаки:

   (a) Средняя длина предложения
   (b) Читаемость предложений
   (c) Средняя встречаемость слов-маркеров (союзов, предлогов)
   (d) Средняя частота встречаемости частей речи
   (e) ...

   Найти текст, для которого все характеристики были бы учтены, сгенерированный марковскими цепями, просто невозможно. Таким образом много чего плохого отсекается.

2. Crime. Это метод, при котором пользователю и роботу сайт даёт разное содержимое одного и того же сайта (по idшникам роботом отдаётся специальная страница со смыслом, а пользователю - мусор, спам, вирус). Борьба: если по userAgent одно, а по запросу другое, то не пропустит поисковая система этот сайт. Если встречается вирус, то нельзя давать пользователю или давать со специальной пометкой. Несколько слов о Большом Брате. Информация собирается в ловушки, и браузер направляет запрос соответствующему сервису, для того, чтобы узнать, хороший ли этот сайт или нет (Chrome, Mozilla - на Google) и потом результаты выдаются пользователю.

[6] Чёткой границы нет

3. Ссылочный спам. На страницу ставятся искусственные ссылки. Поисковые системы с этим пытаются бороться. Способы борьбы отличаются от естественных. Могут использоваться глобальные методы типо анализа графа ссылок. Если есть аномалии, то это говорит о том, что здесь не всё чисто.

Поговорим немного об анализаторах качества и вспомним Ашманова. Анализатор Ашманова - попытка анализатора промышленного качества поиска. Существует некоторое количество наборов запросов и некоторые типы идеальных ответов, определённых заранее, и смотрится, что выдаётся по соответсвующим запросам поисковыми системами. Там (на сайте) есть и менее распространённые поисковые системы.

Анализаторы:

1. Анализ качества информационного поиска. Если пользователь спрашивает "Комсомольская правда то первой выдачей должен быть её сайт.

2. Анализатор качества подсказок

3. Анализатор устойчивости

4. Анализатор спама. Наличие порнухи в выдаче непорнушных запросов. В Яндексе на тему "рассказ"выдавалось 9 подсказок порнушных. Затем сделали градацию: порно/не порно. Мало что известно о технологиях, применяемых поисковыми системами здесь. Это делается для того, чтобы спамерам было неизвестно.