ПОЛНОТЕКСТОВЫЙ ПОИСК В ПОРТАЛАХ ТЕХНИЧЕСКОЙ ПОДДЕРЖКИ

М.Ю. Богатырев, А.П. Колосов
Тульский государственный университет,
SmartBear Software Inc.
Тула

ВВЕДЕНИЕ

Электронные ресурсы, посвященные ответам на вопросы (например, системы служб техподдержки или форумы) по мере своего развития накапливают все больше информации, поиск по которой способен, с одной стороны, помочь пользователям таких систем получить ответ немедленно, не ожидая ответа специалистов, а с другой – снизить количество дублирующихся вопросов и нагрузку людей, отвечающих на вопросы.

Однако практика показывает, что большинство пользователей предпочитает сразу прибегать к помощи других людей, не пытаясь найти ответ среди имеющейся информации. А значит, уменьшение количества дубликатов и снижение нагрузки специалистов может достигаться путем автоматического поиска по тексту вопроса до того, как этот вопрос (состоящий в общем случае из 1-10 предложений произвольного стиля на естественном языке) попадет в базу данных.

Имеющиеся в настоящее время системы позволяют производить автоматический поиск по ключевым словам из темы сообщения, но известная статистика показывает, что такой подход недостаточно эффективен. Только 1% пользователей находит нужную информацию среди автоматически подобранных результатов. Еще 4,5% после просмотра предложенных результатов изменяют свой вопрос. Подавляющее большинство все же отправляет свои вопросы в техподдержку без изменений. Объясняются такие результаты достаточно просто: лишь очень немногие пользователи могут достаточно кратко и в нужных терминах сформулировать суть проблемы, с которой они столкнулись. Большинство просто указывает абстрактные темы вроде «Support query» или «Usage problem». Следовательно, для повышения эффективности поиска необходимо анализировать весь текст запроса.

В данной статье рассматриваются задача и алгоритм поиска по полным текстам запросов. Решения иллюстрируются на примере портала техподдержки компании, занимающейся производством программного обеспечения.

ПОСТАНОВКА ЗАДАЧИ

Информационный ресурс и данные. Структура рассматриваемого информационного ресурса достаточно типична: имеется база данных опубликованной в Интернете справочной документации (Help, FAQ и т.п.) по продуктам, а также база данных техподдержки, в которую собираются вопросы пользователей, поступающие с форумов и по электронной почте. В той же базе данных хранятся ответы на эти вопросы.

Входными данными для поиска являются запросы пользователей на естественном языке, выходными – список документов, которые могут содержать нужную пользователю информацию. Запросы состоят из

темы (короткого описания) и полного текста с подробным (несколько предложений) описанием проблемы на русском или английском языке.

Отличия от стандартгного поиска в сети Интернет. Поставленная задача поиска имеет ряд важных отличий от классического поиска в сети Интернет.

1. Текст запроса может быть достаточно длинным (10-15 предложений) и содержать не влияющие на смысл слова и предложения (приветствия, подписи, и т.п.).

2. Все индексируемые страницы принадлежат одной и той же компании, заинтересованной в адекватности результатов поиска. Поэтому нет смысла выполнять действия, направленные на борьбу с некорректным продвижением сайтов.

3. Количество индексируемых документов в данной задаче составляет десятки тысяч, т.е. относительно невелико по сравнению с количеством страниц в Интернете, документы посвящены одной тематике, или ряду близких тематик.

ОПИСАНИЕ АЛГОРИТМА

Алгоритм, предлагаемый для решения поставленной задачи, можно разбить на следующие этапы.

1. Обработка запроса (разбиение на слова, выделение словосочетаний).
2. Поиск (вычисление релевантности по ключевым словам и словосочетаниям).
3. Обучение (корректировка веса ключевых слов).

Последняя операция не является обязательной, но позволяет улучшить качество поиска.

Обработка запросов. На этапе обработки запросов производится разбор текста на слова и предложения, фильтрация шумовых слов, раскрытие форм слов и выявление словосочетаний, которое рассмотрим подробнее.

Выявление словосочетаний производится путем обработки знаков препинания в индексируемых документах, а также путем построения *концептуальных графов* [1], соответствующих предложениям запросов. Концептуальный граф как семантическая модель текста предложения позволяет найти в нем словосочетания в виде пар концептов, связанных определенными отношениями, например, отношением «*атрибут*». В данной технологии применяется программное обеспечение для автоматического построения концептуальных графов для текстов [3].

В предлагаемом алгоритме знаки препинания, разделяющие слова, учитываются при вычислении позиции слова относительно начала текста. В индексах используется векторная модель текстов – каждому слову ставится в соответствие его вес, вычисленный по известной формуле tf (term frequency) [5]. Также для каждого слова хранятся порядковые номера, характеризующие расстояние относительно начала текста. Наличие знаков препинания приводит к искусственному увеличению (или уменьшению) расстояния между словами, давая базовое представление о том, какие слова могут быть связаны, а какие, напротив, не должны иметь семантических связей друг с другом. Благодаря этому информация о знаках препинания неявно сохраняется в индексах, хранящих только вес слов и их позицию.

Знаки препинания и расположение слов и позволяют судить о том, какие слова в индексируемых документах могут быть связаны, в том числе и в виде словосочетаний. Для боле полного выявления словосочетаний этого недостаточно. Для более детального анализа предлагается использовать концептуальные графы, учитывающие информацию о формах слов, частях речи и т.п. Здесь используется одно важное наблюдение – для описания технической сути проблемы пользователи стараются использовать грамматически корректные сочетания слов и термины, которые они видят в используемых продуктах – например, в текстах ошибок, в интерфейсе и т.п. Незначащие предложения (например, приветствия, подписи, и т.п.) часто не являются грамматически корректными, и их можно отфильтровать простым способом, построив для них концептуальные графы. Такие графы окажутся некорректными, т. е. в них окажутся концепты, не связанные никакими отношениями. Обнаружение таких концептов выполняется также при помощи системы построения концептуальных графов [3].

Для построения концептуальных графов использовались два внешних ресурса: словарь, хранящий информацию о частях речи, формах слов и т.п. и набор шаблонов, описывающих различные семантические роли, т.е. отношения между словами.

Результатом разбора запроса является ряд словосочетаний, состоящих из семантически связанных (а не просто идущих подряд) слов. Если какое-то словосочетание повторяется в тексте несколько раз, ему присваивается больший вес. Далее эти словосочетания объединяются оператором OR (поскольку текст может содержать несколько не связанных друг с другом вопросов) и подаются на вход поисковой системы. В результате задача поиска по тексту сводится к поиску по набору словосочетаний, а эта задача решается с помощью предлагаемого алгоритма.

Поиск по словам и словосочетаниям. Алгоритм поиска, использует итоговую релевантность документа запросу, которую определяют следующие факторы:

1. операторы поиска (AND, AND NOT, OR, “exact phrase”, и т.п.),
2. скорректированный вес ключевых слов,
3. позиции слов в документе и знаки препинания,
4. результаты морфологического разбора запроса.

Сначала производится булевский поиск, т.е. отбор документов, соответствующих имеющимся в запросе операторам поиска (если операторы не заданы явно, все слова считаются объединенными оператором AND). Затем для каждого отобранного документа вычисляется релевантность по отдельным словам. После этого вычисляется релевантность по словосочетаниям. Поскольку проиндексированные документы считаются состощими из нескольких полей (заголовок и текст, заголовок имеет больший вес), релевантность вычисляется отдельно по каждому полю. После этого, с учетом веса поля, считается итоговая релевантность.

Для вычисления релевантности по ключевым словам предлагается использовать косинус угла между векторами документов. Также рассматривалась и другая популярная формула, Okapi BM25, но эксперименты, проводившиеся на тестовом множестве из 10 000 документов, показали, что оптимальные результаты как по качеству, так и по производительности дает именно косинус угла, часто использующийся для определения смысловой близости при векторной модели текстов:

$$\cos(d_j, q) = \frac{d_j \cdot q}{\|d_j\| \|q\|} = \frac{\sum_{i=1}^{N} w_{i,j} \cdot w_{i,q}}{\sqrt{\sum_{i=1}^{N} w^2_{i,j}} \sqrt{\sum_{i=1}^{N} w^2_{i,q}}}$$

где $w_{i,j}$ – вес соответствующих слов, вычисленный по формуле tf.

При вычислении релевантности по словосочетаниям учитываются следующие предположения:

1. словосочетания, повторяющиеся в тексте запроса несколько раз, имеют больший вес (зависящий от количество повторений),
2. чем больше слов в словосочетании, тем более узкий смысл оно выражает, а значит, тем больше смысловая близость между документами, содержащими это словосочетание,
3. чем дальше слова расположены друг от друга, тем меньше вероятность того, что они связаны.

Правило подсчета количества искомых словосочетаний в документе основывается на том, что слова, расположенные далеко друг от друга и/или в разных предложениях, скорее всего, не связаны между собой. Соответственно, контекстное окно должно учитывать оба фактора, что легко достигается за счет искусственного увеличения позиции слов при обработке знаков препинания.

Считается, что искомое словосочетание присутствует в документе, если расстояние между каждой парой составляющих его слов меньше некой величины, равной искусственному приращению позиции после конца предложения. Данное условие является более гибким, чем то, что используется в недавно появившемся алгоритме PATeR [2], в котором учитывается только разбиение текста на предложения, но не обрабатываются ситуации, когда два слова находятся на разных концах длинного предложения. Также предлагаемый алгоритм учитывает большее количество факторов (например, количество слов в словосочетании).

Релевантность по словосочетаниям предлагается вычислять по следующей формуле:

$$R_{phrase}(q, d_j) = \frac{\sum_{i=1}^{N} R_p(p_i, d_j) w_{p_i}}{N},$$

то есть релевантность документа d_j запросу q вычисляется как среднее арифметическое релевантностей по каждому из рассматриваемых словосочетаний p_i, выделенных из запроса q, с учетом веса каждого словосочетания, вычисляющегося как частота возникновения словосочетания в запросе (аналогично формуле tf). Здесь R_p – релевантность документа словосочетанию p_i, вычисляемая по следующей формуле:

$$R_p(p_i, d_j) = \sum_{k=1}^{m} \frac{2^{2n_{t_i}}}{\Delta_{p_i,k}},$$

где n_{t_i} – количество слов в словосочетании p_i (в общем случае оно может состоять из двух и более слов), $\Delta_{p_i,k}$ – суммарное расстояние между каждым из этих слов в рассматриваемом документе d_j, вычисленное для каждого вхождения в документ словосочетания p_i, m – общее количество вхождений словосочетания p_i в документ d_j.

Формула итоговой релевантности учитывает следующие факторы:

1. релевантность по словосочетаниям должна иметь больший вес, чем релевантность по отдельным словам, но ее отсутствие не должно приводить к нулевому итоговому значению,

2. релевантность по словам и словосочетаниям вычисляется для каждого проиндексированного поля отдельно, после чего умножается на вес поля.

Для того, чтобы сделать формулу итоговой релевантности более универсальной, логично не связывать ее с конкретными алгоритмами, использующимися для вычисления каждой составляющей релевантности.

Вычислять итоговую релевантность предлагается по следующей формуле:

$$R(q,d_j)=\sum_{k=1}^{Nf} w_f \cos(q,d_j)(R_{phrase}(q,d_j)+1),$$

где Nf – общее количество проиндексированных полей (в текущее реализации равно 2), w_f – вес проиндексированного поля, cos(q, d_j) – косинус угла между векторами запроса q и документа dj, характеризующий их близость по ключевым словам, R_{phrase} – релевантность по словосочетаниям, вычисленная ранее.

ЭКСПЕРИМЕНТАЛЬНЫЕ РЕЗУЛЬТАТЫ

Рассматриваемый поисковый алгоритм реализован и опробован в рамках эксперимента на реальных данных. Использовалось тестовое множество из 10 000 статей, являющихся документацией по 6 программным продуктам. Поиск проводился по 30 наиболее популярным словосочетаниям, выявленным из статистики поиска по сайту компании SmartBear Software [6] за 2010 год. В качестве асессоров выступали сотрудники компании, проставлявшие оценки качества первых 10 выданных системой результатов от 0 (не релевантный) до 2 (максимальная релевантность).

Рассмотрим пример запроса в техподдержку:

“Hi there! I have a ***script test*** *with a* ***bunch of checkpoints****, but when it hits a checkpoint cannot be verified, the* ***execution of the script stops*** *and* ***any tests*** *after the* ***failed checkpoint*** *do not get executed. Thank's in advance.*

Жирным выделены словосочетания, имеющие технический смысл и выявленные в результате разбора. Отметим, что предложения *Hi there* и *Thank’s in advance* были отфильтрованы на этапе построения концептуальных графов, поскольку между словами не обнаружилась семантическая связь. Фрагмент концептуального графа, построенного автоматически из приведенного запроса, приведен на рис. 1:

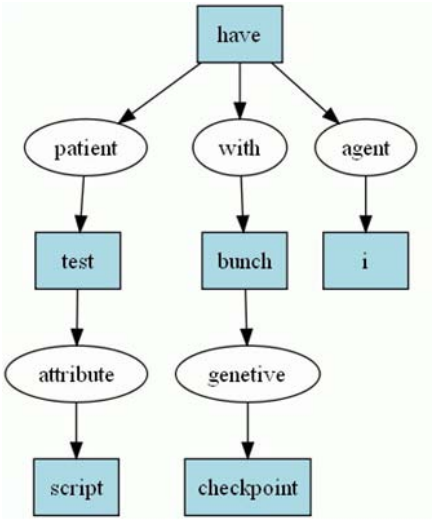

Рис. 1. Фрагмент концептуального графа

Как видно из рисунка, построенный граф позволил выявить взаимосвязь между фразами *I have*, *test script* и *bunch of checkpoints*, хотя в исходном тексте эти словосочетания разделены артиклями и предлогами. Подсчет встречающихся рядом слов, как, например, при латентно-семантическом анализе [4], не позволил бы выявить эту связь.

Для оценки качества предлагаемого поискового алгоритма использовалась формула discounted cumulative gain:

$$DCG_p = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2 i}$$

где rel_i – проставленная асессором релевантность (от 0 до 2), i – порядковый номер результата поиска в выдаче, p – количество оцениваемых результатов (в данном эксперименте бралось равное 10). Отметим, что в отличие от популярной метрики p@K (precision at top K), данная формула учитывает как релевантность каждого результата, так и порядковый номер результата поиска в выдаче. График, показывающий усредненные значения результатов, представлен на рис. 2:

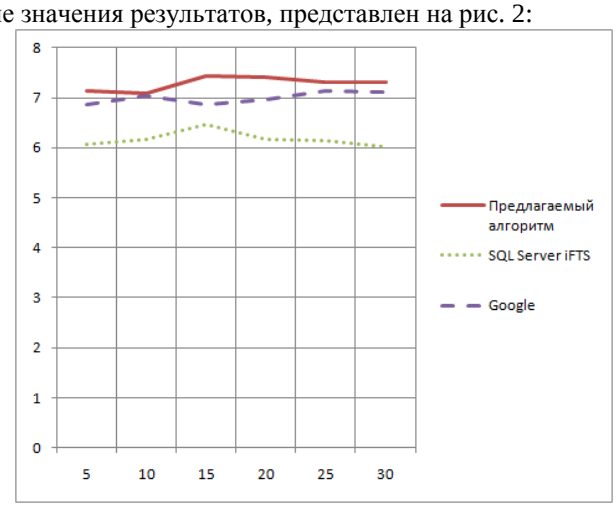

Рис. 2. Качество результатов поиска

Вертикальная ось соответствует качеству результатов поиска (максимально возможное значение равно 10,51). Горизонтальная ось – количеству поисковых запросов, по которым считалась средняя величина DCG_p.

Видно, что по мере увеличения количества запросов предлагаемый алгоритм дает более качественные результаты, чем два других (SQL Server iFTS относится к встроенному поисковому алгоритму SQL Server, используемому в данный момент для поиска по сайту). Низкие результаты этой системы можно объяснить тем, что она использует формулу Okapi BM25, а она, как уже упоминалось, дает на рассматриваемом множестве статей менее качественные результаты. Некоторое улучшение относительно Google можно объяснить тем, что формула релевантности этого поисковика дает очень большой вес «внестраничным» факторам, таким как ссылки на документ, а в случае поиска по документации такой подход, как уже упоминалось ранее, не оправдан.

ЗАКЛЮЧЕНИЕ

Для решения поставленной задачи поиска по длинному запросу, состоящему из нескольких предложений, предлагается алгоритм, выделяющий из текста запроса связанные словосочетания путем построения концептуальных графов. Таким образом, задача поиска по тексту сводится к задаче поиска по словосочетаниям. Для поиска по словосочетаниям предлагается алгоритм, учитывающий большее количество факторов, чем существующие аналогичные алгоритмы. Экспериментальные результаты доказывают эффективность предлагаемого алгоритма поиска по словосочетаниям.

В дальнейшем планируется улучшить качество поиска за счет применения обучения без учителя. Такое обучение на основе статистических данных о поведении пользователей поможет улучшить релевантность выдаваемых результатов. Также планируется ввести обучение распознаванию

словосочетаний, которое поможет улучшить качество построения концептуальных графов, и расширит число поддерживаемых языков.

ЛИТЕРАТУРА

1. A World of Conceptual Graphs, http://conceptualgraphs.org/
2. Bani-Ahmad S.G., Al-Dweik G. A new term-ranking approach that supports improved searching in literature digital libraries // Research Journal of Information Technology, 2011. Volume 3, Number 1, p. 44-52.
3. Bogatyrev, M.Y., Mitrofanova, O. A., Tuhtin, V.V. Building Conceptual Graphs for Articles Abstracts in Digital Libraries. - Proceedings of the Conceptual Structures Tool Interoperability Workshop (CS-TIW 2009) at 17th International Conference on Conceptual Structures (ICCS'09) - Moscow, Russia, July 2009, - p.p. 50-57.
4. Landauer, T.K., Foltz, P.W., Laham, D. An Introduction to Latent Semantic Analysis. Discource Processes, Issue 25, p. 259-284, 1998.
5. TF-IDF, Wikipedia, 10.01.2007 http://en.wikipedia.org/wiki/Tf%E2%80%93idf
6. Компания SmartBear Software http://smartbear.com/

МОДЕЛЬ ОНТОЛОГИЧЕСКОГО ПРЕДСТАВЛЕНИЯ ТЕХНОЛОГИИ

М.В. Воронов, Г.И. Письменский, Д.А. Андреев

Современная гуманитарная академия

Москва

Знания о технологиях, несомненно, относятся к наиболее ценной в современном мире информации (технологии здесь рассматриваются во всем спектре их применения: в промышленности, социальных процессах, культуре и т.д.). Однако фиксация сведений о технологиях, как правило, осуществляется в виде вербального описания, использование которого для обработки компьютерными средствами крайне затруднительно [1,2]. Это описание, в зависимости от решаемой задачи, может требовать большей или меньшей глубины (степени) детализации, причем для отдельных компонентов – различной. В этой связи актуальной представляется задача разработки инструментария для формализации описания технологий.

Решение такой задачи предлагается осуществлять в виде построения конструктивной модели онтологического представления технологии, основу которого образуют так называемые элементарные модели следующего вида [3].

$$\langle TP, T, X, Y \rangle, \qquad (1)$$

где

- TP – имя технологии (технологического процесса), которое, как правило, представлено в форме глагола (или глагольной фразы), описывающего соответствующее технологическое действие;
- T - время реализации технологии;
- $X = \langle X_0, SX, RX \rangle$ – множество всех исходных (входных) компонентов рассматриваемой технологии, где X_0 – объекты, участвующие в технологии, SX – их состояние на момент начала технологического процесса, RX – отношения между этими объектами;
- Y – множество всех результирующих (выходных) компонентов данной технологической системы.

Пару (X, Y) называем «внешней границей» технологического процесса.

Подчеркнем, что эта модель, представляющая своего рода модель «черного ящика», содержит полное описание внешней границы технологического процесса, собственно механизм которого описывается лишь его именем (в разработанных программных комплексах каждое имя сопровождается определенным описанием в виде фрейма).

Процесс конструирования онтологии состоит из двух основных этапов: построения исходного представления и многошагового процесса декомпозиции представления, полученного на предыдущих шагах.

За исходное принимается представление рассматриваемой технологии в виде элементарной обобщенной модели типа $TS_0 = \langle TP_0, T_0, X_0, Y_0 \rangle$.