СТАТЬИ

УДК 519.763:002.513.5

СТРАТЕГИЯ «УМНОГО» ПОИСКА ДЛЯ ПЕРЕЧНЯ НЕСТРУКТУРИРОВАННЫХ ДАННЫХ

Галкин Р.В., Горбачев Д.В., Соловьев Н.А.

ФГБОУ ВО «Оренбургский государственный университет», Оренбург, e-mail: gordi47@mail.ru

Основной задачей проводимого исследования является разработка поисковой системы, учитывающей ошибки пользователя при вводе слов и словосочетаний. Такой поиск, например, может быть связан с поиском лекарственных препаратов. По своей сути, данные, участвующие в поисковом запросе, имеют текстовый тип. В предлагаемой работе в качестве метода, обеспечивающего необходимую релевантность результата поискового запроса, предлагается использовать нечеткий поиск. Задача нечеткого поиска заключается в последовательном переборе индекса, включающего все неструктурированные наименования предметной области, с целью выбора вариантов, содержащих определённое число совпадающих с исходной строкой символов. Известные в настоящее время методы на основе нечетких правил приближенно имеют близкую к линейной зависимость скорости работы от размера словаря. В то же время применяемые методы оптимизации алгоритмов поиска позволяют получать приемлемое время работы даже при значительных объемах словарей. Однако известной проблемой поисковых систем по-прежнему остаются вопросы достижения требуемой релевантности результатов поисковым запросам. Поэтому актуальность проводимого исследования обуславливается тем, что в поисковых запросах часто встречаются грамматически неправильные написания терминов предметной области. Одним из рассматриваемых в работе подходов практического применения нечетких правил поиска являются фонетические алгоритмы и алгоритмы лексического стемминга. В данной работе алгоритм полнотекстового нечёткого поиска строится на основе расстояния Дамерау – Левенштейна, поскольку в этом случае учитываются все возможные ошибки при вводе символов слова в запросе, но у него есть недостаток – выводит дистанцию одинаково для двух слов, которые имеют смещение, различные части или вхождения одного в другое. Для его компенсации предлагается использовать алгоритм схожести строк Джаро – Винклера, который позволяет минимизировать преобразования символов, обеспечивающего правильное написание слова.

Ключевые слова: поисковые системы, релевантный результат, нечеткий поиск, семантика, поисковый индекс, строка символов

SMART SEARCH STRATEGY FOR A LIST OF UNSTRUCTURED DATA

Galkin R.V., Gorbachev D.V., Solovev N.A.

Orenburg State University, Orenburg, e-mail: gordi47@mail.ru

The main objective of the ongoing study is to develop a search engine that takes into account user errors when entering words and phrases. Such a search, for example, may be related to the search for pharmaceuticals. In essence, the data involved in a search query is of the text type. In the proposed work, fuzzy search is proposed as a method to provide the necessary relevance of the search query result. The task of fuzzy search consists in successive search of the index including all unstructured names of a subject domain, for the purpose of a choice of variants containing the certain number of conterminous with initial string of symbols. Currently known methods based on fuzzy rules approximate close to linear dependence of the speed of operation on the size of the dictionary. At the same time, the applied methods of search algorithm optimization allow to get the acceptable running time even for large dictionary sizes. However, a known problem of search engines is still the problem of achieving the required relevance of the results to the search queries. Therefore, the relevance of the study is due to the fact that in search queries often occur grammatically incorrect spelling of the terms of the subject area. One of the approaches of practical application of fuzzy search rules considered in work are phonetic algorithms and algorithms of lexical stemming. In this paper, the full-text fuzzy search algorithm is based on the Dahmerau-Levenstein distance, because in this case takes into account all possible errors in the characters of the word in the query, but it has a drawback – it outputs the distance the same for two words that have an offset, different parts or occurrences of one in the other. To compensate for it, it is proposed to use the Jaro-Winkler string similarity algorithm, which allows you to minimize the character conversion that provides the correct spelling of the word.

Keywords: search engines, relevant results, fuzzy search, semantics, search index, character string

Наряду с появлением интернет-магазинов появились и первые поисковые системы товаров [1]. Несмотря на большие возможности поисковых систем в электронной коммерции, они функционально ограничены. Так, например, в проектах с малым бюджетом компаний, которые не способны обеспечить себе штат программистов, используются зарубежные системы, не способные корректно разбирать строки, содержащие нелатинские символы, а также учитывать семантику и фонетику слов. В то время как учёт семантики (смысла) слов в поисковых алгоритмах торговых сервисов не имеет практической ценности из-за низкой производительности алгоритмов и малой эффективности, получить фонетику (звучание) слова можно для любого языка, используя стандартизированный процесс транслитерации [2]. Зачастую по-

иск на сайтах ограничен точным линейным поиском, предполагающим разбиение строк с помощью разделителя (обычно пробел) [3].

Необходимая стратегия поиска должна предполагать работу с неструктурированной базой наименований продукции для осуществления поисковой задачи. Такие базы актуальны на сегодняшний день из-за появления большого числа источников информации, таких как открытые веб-сайты, устаревшие базы данных (не имеющие структуры), перечни товаров в электронных таблицах. В структурированных системах поиск осуществляется поэтапно, используя эталонные записи и/или фильтры. В этом же заключается недостаток таких систем, так как они требуют наличия персонала, который сможет классифицировать товар и задать ему первоначальные признаки. Преимущества неструктурированного поиска [4] заключаются в автоматическом выборе системой эталона, в независимости поиска от семантики данных. Однако в противовес такой поиск требует хорошо продуманного алгоритма, учитывающего варианты написания поискового запроса. Это, в свою очередь, добавляет трудностей на этапе внедрения, так как появляется необходимость провести тестирование и определить граничные значения фильтрации, а также значимость каждого из факторов сортировки. Стоит отметить, что такой тип поиска имеет низкую результативность при запросах, содержащих неожиданную для системы терминологию, например различные единицы измерения или сокращения слов, если они не присутствуют в исходной таблице данных.

Целью исследования является разработка системы, осуществляющей автоматический сбор информации с открытых интернет-сервисов для дальнейшей обработки, сохранения, с последующей выдачи в качестве результата на поисковые запросы пользователя. Основу разрабатываемой системы составляют алгоритмы нечёткого поиска.

Материалы и методы исследования

При разработке стратегии поиска был проведён анализ существующих поисковых систем, чтобы определить количество и качество результатов их работы в совокупности со стоимостью технологии [5]. По результатам было определено, что ценность нелинейных (умных) корпоративных поисковых систем обусловлена низким быстродействием или, как правило, высокой ценой. При анализе поисковой системы с точки зрения экономии ресурсов поиск должен быть или быстрым, при сохранении прежнего уровня эффективности, или выгодным для реализации. В любом случае, несмотря на открытость поисковых технологий, чаще всего при разработке используются линейные поисковые алгоритмы, так как этого функционала зачастую достаточно непосредственно заказчику. Однако клиента набор функций поиска обычно не удовлетворяет, что уменьшает привлекательность программного обеспечения.

Большинство алгоритмов поиска имеют зависимость скорости выполнения от количества символов в тексте. Соответственно, использование их для полнотекстового поиска нецелесообразно. Поэтому для увеличения скорости работы алгоритма поиска, предлагается применить прямую индексацию таблиц баз данных, другими словами, необходимо преобразовать оригинальные строки в строки, содержащие только ключевые символы. Для решения этой задачи можно использовать преобразование строк с помощью «метафонов».

Metaphone – это фонетический алгоритм для индексирования слов по их звучанию с учётом основных правил английского произношения, из схожих по звучанию слов получаются одинаковые ключи [6].

Результаты исследования и их обсуждение

Сущность предлагаемого подхода заключается в следующем.

Задачу поиска по неструктурированной информации можно разделить на три этапа и распределить ее по соответствующим подсистемам: 1) обработка входных данных, 2) поиск по индексированным наименованиям, 3) фильтрация по параметрам выходного результата. Подсистема обработки предполагает операции, используемые при индексировании данных: удаление специальных и повторяющихся символов из наименований; формирование слов из чисел и наоборот; разбиение наименований на слова; транслитерация слов с помощью английского алфавита (как международного языка); применение фонетических алгоритмов (метафонов) на строках для получения интерпретации слов; подсчёт количества вхождений слов в наименования; формирование для каждого слова индекса в базе данных системы, который указывает на вхождение в наименование.

Подсистема поиска имеет более интересную функциональную реализацию, чем стандартные линейные методы поиска за счёт использования методов нечёткого поиска (мер схожести).

Практическое использование алгоритмов нечеткого поиска в реальных поисковых системах тесно связано с фонетически-

ми алгоритмами, алгоритмами лексического стемминга – выделения базовой части у различных словоформ одного и того же слова, а также с ранжированием на основе статистической информации, либо же с использованием сложных изощренных метрик [7].

Возможность использовать метрики при обработке данных позволяет получить разницу между начальным запросом и конечным выводом системы. Каждый из существующих алгоритмов поиска задействует собственную метрику, в основном базирующуюся на схожести букв или частей строк с индексированным наименованием. Некоторые алгоритмы и впрямь позволяют назвать численную метрику принадлежностью одной строки к другой (отсюда и название «нечеткий»).

Подсистема поиска первым этапом производит обработку поискового запроса для приведения к общему виду, достаточного для индекса, составляя, таким образом, совокупность из слов. Вторым этапом производится лексический анализ каждого из совокупности слова по базе индексированных наименований посредством меры схожести лексем, формируя нечеткие множества A_j, состоящие из степеней принадлежности индексов к слову.

Нечетким множеством A в некотором (непустом) пространстве X, что обозначается как $A \in X$, называется множество пар $A = \{(x, \mu_A(X))\}$, где $\mu_A(X)$ – функция принадлежности нечеткого множества A. Нечеткое множество полностью определяется заданием функции принадлежности $\mu_A(X)$: ее область определения – $x \in X$, область значений – отрезок [0,1] [8].

Чем выше значение $\mu_A(X)$, тем выше оценивается степень принадлежности элемента $x \in X$ нечеткому множеству A.

Тогда $A_j = \{(I_1 \mid \mu_{j,1}), \ldots, (I_i \mid \mu_{j,i})\}$, где j – размер совокупности слов, i – количество индексов, $\mu_{j,i}$ – значение функции принадлежности j-го слова к i-му индексу. Из множества отсекаются варианты, значения принадлежностей которых не входят в граничные значения алгоритма.

Тем не менее нельзя полагать, что использование одной метрики позволит добиться максимальных результатов. К тому же некоторые метрики можно вычислить быстрее, но они менее надёжны (например, сходство Джаро – Винклера [9]), и, наоборот, некоторые метрики медленнее, но более информативны (расстояние Дамерау – Левенштейна (Задача о расстоянии Дамерау-Левенштейна)). Поэтому для улучшения поисковой релевантности целесообразно использовать несколько мер схожести строк. Это позволяет увеличить быстродействие на этапе поиска данных, отсекая лишние значения выборки, сокращая время на перебор для каждого последующего вычисления метрики.

По завершению анализа всей совокупности формируется итоговое множество, состоящее из пересечений промежуточных множеств, до тех пор пока последующее пересечение не даст пустое множество или не закончится их перебор. Конечным этапом определяются исходные наименования, на которые указывают найденные индексы, составляется нечёткое множество:

$$T = \{ (I_1 | \frac{\sum_1^m D_{m,1}}{m}), \ldots, (I_k | \frac{\sum_1^m D_{m,k}}{m}) \}, \quad (1)$$

где k – количество индексов, из которого состоит наименование, I_k – k-й индекс; m – число используемых метрик схожести; $D_{m,k}$ – нормализированное значение m-й метрики для k-го индекса (относительно граничных значений функции).

Таким образом, определяется, насколько найденные результаты соответствуют поисковому запросу, позволяя гибко распределять их на группы для выполнения последующих операций над ними.

Подсистема фильтрации часто является конечной подсистемой любого поискового алгоритма. Именно на этом этапе сортируются значения в указанном заранее порядке по необходимому критерию (наименованию, цене) и фильтруются при указанном диапазоне входных значений, которые заранее структурированы, например дата, или номер склада (наименование). В случае поиска неструктурированной информации подсистема фильтрации позволяет привести итоговое множество к табличной форме, добавляя один или несколько структурированных критериев (релевантность по метрике) при сортировке, позволяя более гибко получать результаты поиска: вывод в начале релевантных результатов, а затем похожих по написанию или звучанию.

Стратегия поиска в предлагаемой поисковой системе заключается в последовательном переборе определённого количества символов до 20. При этом предполагается, что задача обработки и индексирования данных решена.

Пусть даны две символьные строки: s_1 – одна из частей обработанной поисковой строки, s_2 – одна из индексированных наименований.

Шаг 1. Вычислять $d_j(s_1, s_2)$ – функцию Джаро – Винклера. Значения функции лежат в интервале [0; 1].

Расстояние Джаро d_j между двумя заданными строками s_1 и s_2 это [9]:

$$d_j = \begin{cases} 0 & ,когда\, m = 0 \\ \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right) & ,во\, всех\, остальных\, случаях \end{cases}, \quad (2)$$

где $|s_1|$ – длина строки s_i; m – число совпадающих символов; t – половина числа транспозиций.

Каждый символ строки s_1 сравнивается со всеми соответствующими ему символами в s_2. Количество совпадающих (но отличающихся порядковыми номерами) символов, которое делится на 2, определяет число транспозиций [9].

Расстояние Джаро–Винклера использует коэффициент масштабирования p, что дает более благоприятные рейтинги строкам, которые совпадают друг с другом от начала до определённой длины l, называемой префиксом.

Для двух строк s_1 и s_2 расстояние Джаро – Винклера d_w – это:

$$d_w = d_j + (lp(1 - d_j)), \quad (3)$$

где l – длина общего префикса от начала строки до максимума четырех символов; p – постоянный коэффициент масштабирования, использующийся для того, чтобы скорректировать оценку в сторону повышения для выявления наличия общих префиксов. Коэффициент p не должен превышать 0,25, поскольку в противном случае расстояние может стать больше, чем 1. Стандартное значение этой константы в работе Винклера: $p = 0{,}1$ [9].

Шаг 2. Выбрать $J_{ср}$ – некоторое пороговое значение функций Джаро – Винклера для отсеивания лишних результатов, например, 0,75.

Шаг 3. Проверить $d_w(s_1, s_2) > J_{ср}$ для последующей обработки только похожих вариантов.

Последовательный перебор между поисковым запросом и индексами таблицы работает как линейный поиск, за исключением увеличения сравнивающих операций. Как показали эксперименты, такой метод является достаточно быстрым за счёт небольшого количества операторов функции.

Шаг 4. Получить значение

$$DL(s_1, s_2) = D(s_1, s_2)\left[|s_1|, |s_2|\right]$$

матрицы рекуррентного соотношения Дамерау – Левенштейна. Значения функции элементов лежат в интервале $\left[0; |s_i|\right]$. Упрощённо функцию элементов матрицы можно представить следующим образом [10]:

$$D_{i,j} = \min\left(X_в + 1, X_у + 1, X_з + C_{замены}, X_т + C_{транспозиции}\right), \quad (4)$$

$$C_{замены} = \begin{cases} 1, если\, S_1[i] \neq S_2[j] \\ 0, иначе \end{cases} \quad (5)$$

$$C_{транспозиции} = \begin{cases} 1, если\, S_1[i] = S_2[j-1] и\, S_1[i-1] = S_2[j] \\ \infty, иначе \end{cases}, \quad (6)$$

где i, j – строка и столбец матрицы, используемые для проверки символа первой строки символом второй строки. $i, j \in \left[0; \max\left(|s_1|, |s_2|\right)\right]$; X_B – необходимое количество вставок символов; $X_у$ – необходимое количество удалений символов; $X_З$ – необходимое количество замен символов; $X_т$ – необходимое количество транспозиций; $X_В$, $X_у$, $X_З$, $X_т$, – число элементов, необходимых для получения исходной строки.

Шаг 5. Выбрать $DL_{ср}$ – некоторое пороговое значение функций Дамерау – Левенштейна для отсеивания лишних результатов, например, 2 (символа).

Шаг 6. Проверять $DL(s_1, s_2) < DL_{ср}$ для последующей обработки только вариантов с семантическим совпадением.

Последовательный перебор производится между уже отфильтрованными результатами и индексами таблицы, тем самым снижая требования к вычислительной

мощности и увеличивая общее быстродействие системы.

Шаг 7. Пусть k – количество структурированных элементов, используемых при сортировке результата (факторы поиска), причем $k \geq 0$, а C_k – значение k-го элемента, например, стоимость лекарственного препарата, геопозиция покупателя относительно пункта продаж (аптеки).

Шаг 8. Выполнить нормализацию значений $d_w(s_1,s_2)$, $DL_{i,j}(s_1,s_2)$, $C_1, \ldots, C_k$, соответственно получим $d_w'(s_1,s_2)$, $DL'(s_1, s_2)$, $C_1', \ldots, C_k'$.

Шаг 9. Определить количество групп, на которые можно разбить результат, например, «точное совпадение», «возможные варианты» и «альтернативные варианты».

Шаг 10. Сформировать нечёткое множество из нормализованных данных.

В таком случае, функцию принадлежности можно сформировать следующим образом:

$$m(s_1,s_2) = 1 - \frac{d_w'(s_1,s_2) + DL'(s_1, s_2)}{2} \in [0;1], \quad (7)$$

где $d_w'(s_1,s_2)$, $DL'(s_1, s_2)$ – нормализированные значения самих функций или данных.

Инверсия в функции необходима, так как используется расстояние между словами, а не схожесть. При этом усреднение расстояний позволяет добиться большего результата, за счёт того, что оба метода описывают разные метрики, соответственно их объединение добавляет признаки обоих.

Шаг 11. Путём проведения опытов над тестовой совокупностью определить критерии вхождения (пороговые значения) в группы данных. Например, результат с принадлежностью 0.8 может входить в две группы, при этом в группе с точным совпадением он будет иметь меньшую ценность, а в группе с меньшими критериями – большую. Таким образом, это позволит определить качество исходных данных и эффективность поиска. При этом для увеличения результативности будет достаточно поменять критерии вхождения в группу.

Шаг 12. Составить таблицу, состоящую из групп в порядке убывания критериев вхождения (например, сначала будут элементы множества «точные» результаты), добавив столбцы со структурированными данными $C_1', \ldots, C_k'$.

Шаг 13. Для решения задачи определения релевантности результата воспользуемся кластерным анализом.

Основная задача кластеризации формулируется следующим образом: разделить объекты на группы таким образом, чтобы сходство между объектами одной группы было велико, а сходство между объектами разных групп – мало [11].

В алгоритме уже были выделены характеристики, определены метрики и сделано разбиение на группы. Поэтому можно выделить несколько параметров, описывающих характер объекта:

– $\mu(s1, s2)$ – насколько результат соответствует запросу (релевантность);

– $C_1', \ldots, C_k'$ – нормализованные значения структурированных данных.

Применив меру близости и правило объединения объектов в кластер на основе имеющихся параметров получим искомые группы. Результат выбора зависит от конечной реализации, так как необходимо определить значимость каждого из факторов и подобрать необходимые группы. Например, за кластер можно принять склады, на которых необходимо найти товар(-ы), тогда структурированными данными будет цена и расстояние до склада.

Шаг 14. На основе значений меры близости отсортировать результат в соответствии с требованиями к алгоритму. При необходимости использовать группировку данных.

Заключение

Таким образом, предлагаемая стратегия поиска по неструктурированным данным предполагает использование фонетических и лингвистических методов обработки данных, а также методов нечёткого поиска. Это позволяет создать поисковую систему, которая учитывает ошибки пользователей, допущенные в результате нажатия неверной клавиши при вводе или в результате фонетической ошибки. Такая система может выдавать не только точные результаты, но и похожие, и, в зависимости от реализации, даже альтернативные варианты. В настоящее время предлагаемый умный алгоритм поиска проходит апробацию в геоинформационной системе поиска лекарственных препаратов с учетом местоположения покупателя. Кроме того, данная стратегия может применяться в глобальных поисковых системах, так как в них большая часть информации не структурирована.

Список литературы

1. Савельев А.И. Электронная коммерция в России и за рубежом: правовое регулирование. М.: Статут, 2014. 543 с.

2. Большая российская энциклопедия. Т. 32: Транслитерация / А.В. Суперанская. М., 2016. С. 340–341.

3. Колисниченко Д.Н. Поисковые системы и продвижение сайтов в Интернете. М.: Диалектика, 2007. 272 с.

4. Оганесян А. Неструктурированные данные 2.0 // Открытые системы. СУБД. 2012. № 04. С. 68.

5. Голубева Д.М., Бочкова А.А. Сравнительный анализ современных поисковых систем // Скиф. Вопросы студенческой науки. 2017. №16.

6. Binstock A., Rex J. Practical Algorithms for Programmers. Addison-Wesley, 1995. 577 с.

7. Нечёткий поиск в тексте и словаре // Habr. URL: https://habr.com/ru/post/114997/ (дата обращения: 17.09.2021).

8. Нейронные сети, генетические алгоритмы и нечеткие системы / пер. с польск. И.Д. Рудинского: учебное пособие / Д. Рутковская, М. Пилиньский, Л. Рутковский. 2-е изд. М.: Горячая линия – Телеком, 2013. 384 с.

9. Winkler W.E. String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage // Proceedings of the Section on Survey Research Methods: journal. American Statistical Association, 1990. С. 354–359.

10. Задача о расстоянии Дамерау – Левенштейна // Университет ИТМО. URL: https://neerc.ifmo.ru/wiki/index.php?title=Задача_о_расстоянии_Дамерау-Левенштейна (дата обращения: 12.4.2021).

11. Сивоголовко Е.В. Методы оценки качества четкой кластеризации // Компьютерные инструменты в образовании. 2011. № 4. С. 14–31.