ЭФФЕКТИВНЫЙ АЛГОРИТМ ФОРМИРОВАНИЯ КОНТЕКСТНО-ЗАВИСИМЫХ АННОТАЦИЙ.

Губин Максим Вадимович

max@gubin.spb.ru,

Меркулов Андрей Игоревич

u_andrewshi@pisem.net.

ИК «Кодекс», Санкт-Петербург

Большинство современных систем информационного поиска выводят в списке результатов контекстно-зависимые аннотации найденных документов. В статье предлагается алгоритм формирования подобных аннотаций и описывается методика оценки качества его работы. Авторами проведен ряд экспериментов по исследованию предлагаемого и его сравнению с другими известными алгоритмами. В статье приводятся результаты этих экспериментов и их анализ.

Введение

Современные поисковые системы являются незаменимым средством для работы с большими объемами текстовой информации. Большинство таких систем формируют для каждого документа в списке результатов поиска аннотации – короткие выдержки из его текста, помогающие пользователю быстрее принять решение о полезности документа [2,4]. В англоязычной литературе их обычно называют snippet. В настоящее время имеется большое количество работ посвященных различным аспектам работы поисковых систем, но, по мнению авторов данной статьи, работ, исследующих подходы к формированию этих аннотаций и их экспериментальной оценке, существует относительно мало. Известные научные инициативы по исследованию автоматизированного формирования аннотаций, такие как SUMMAC[1], ориентированы на относительно длинные аннотации – обычно проценты от длины документов, в то время как нас интересуют очень короткие выдержки, содержащие, как правило, всего несколько слов.

Большинство современных поисковых машин используют достаточно простые алгоритмы формирования контекстно-зависимых аннотаций. Например, в большинстве доступных в исходных текстах систем в качестве snippet используется первый фрагмент текста, в котором встречаются слова запроса [3].

Столкнувшись на практике с необходимостью реализовать механизм формирования аннотаций, авторы провели ряд исследований, которые и описываются в данной статье.

Требование к контекстно-зависимым аннотациям.

Формируемые системой аннотации должны удовлетворять ряду требований. Их можно разделить на требования, которые выдвигает пользователь, и требования, которые накладываются реализацией системы. Можно назвать следующие требования, выдвигаемые пользователем:

1) Аннотация должна давать представление о том, какую информацию содержит документ о предмете запроса.
2) Быть достаточно короткой, чтобы ее анализ пользователем не занимал много времени. Человек должен иметь возможность охватить сформированный текст «одним взглядом».
3) В случае большого документа, она должна давать представление о том, какие части документа несут релевантную информацию.

С точки зрения реализации к алгоритму формирования аннотации предъявляются следующие требования:

1) Конечная вычислительная трудоемкость. Обработка для создания аннотаций является, с точки зрения поисковой системы, вспомогательной задачей и не должна приводить к заметному замедлению системы;
2) Максимальное использование информации полученной при поиске на этапах отбора и взвешивания документов;
3) Приемлемый требуемый объем оперативной памяти.

В связи с вышесказанным, можно выделить критерии эффективности алгоритма. В данной работе такими критериями мы считали:

1) Субъективную оценку эксперта о «полезности» аннотации для отбора документа;
2) Время формирования аннотации.

Описание алгоритмов.

В настоящее время известно много алгоритмов автоматического аннотирования или формирования краткого содержания документов [5,6]. В связи с требованием, чтобы аннотация была достаточно короткой, практически не длиннее одного предложения, мы решили использовать стандартный для поисковых систем подход, когда аннотация содержит один непрерывный фрагмент текста документа, и основное внимание уделить выбору оптимального фрагмента.

В данной работе нами были рассмотрены три алгоритма формирования контекстно-зависимых аннотаций:

1) Алгоритм, который анализирует в документе только слова запроса и отбирает фрагмент с их наибольшей плотностью. В дальнейшем этот алгоритм мы будем называть базовым.
2) Алгоритм, где кроме слов запроса при отборе фрагмента учитывались и другие слова, имеющие высокую частоту встречаемости в тексте в некотором окружении вокруг фрагмента. Этот алгоритм в статье будет называться Freq.
3) Алгоритм, где кроме слов запроса при отборе фрагмента использовались слова, отобранные с помощью алгоритма LRU-K, учитывающего повторяемость слов в тексте. Этот подход к формированию аннотаций мы будем обозначать LRU-K.

Базовый алгоритм.

При данном алгоритме производилось сканирование текста документа, при этом отбирался самый «тяжелый» фрагмент фиксированной длины не пересекающий границу абзаца. Требуемая длина выбиралась из того требования, что аннотация легко могла быть просмотрена и оценена пользователем. Нами была выбрана длина около 25 слов, но не длиннее 300 символов.

Вес фрагмента документа определялся следующим образом:

$$Wb = Sum(Wi)+K*n/ L$$

, где

Sum(Wi) – сумма весов слов запроса, вошедших в фрагмент. Каждое слово учитывалось только один раз. Вес каждого слова зависел от распределения слова в коллекции и был тем выше, чем более редкое это слово. Он рассчитывался по формуле:

$Wi = \log2(Ni)/\log2(N)$, где Ni – число документов коллекции, где встретилось данное слово, N – общее число документов в коллекции;

K – некоторая константа

n – число слов из запроса, которые встретились в фрагменте;

L – расстояние между первым и последним словом запроса, встретившимся во фрагменте.

Таким образом, указанная формула выше оценивала фрагменты, в которых слова запроса располагались более «кучно», встречалось больше слов запроса, и выше оценивались те слова запроса, которые реже встречались во всей коллекции.

В список результатов поиска для документа помещался фрагмент текста, который получил наибольший вес. Если несколько фрагментов получали одинаковые веса, то выдавался тот, который находился ближе к началу текста.

Отобранный фрагмент «выравнивался в тексте», то есть вырезалось некоторое количество слов до первого слова запроса, и несколько после, добавляя длину фрагмента до выбранной длины.

Алгоритм Freq.

Описанный базовый алгоритм, как мы считаем, не может показать высокого качества формирования аннотаций, если запрос содержит всего одно слово или часто встречающееся словосочетание. В этом случае он просто возвращает первый фрагмент текста, в котором встретились слова запроса. Нами было принято решение использовать кроме слов запроса другие слова документа, которые имеют высокую частоту встречаемости.

Однако анализ частоты слова во всем документе оказался для нас не приемлемым по следующим причинам:

1) Эту информацию (частоты слов в документе) нельзя было быстро получить из используемого нами для поиска инвертированного файла, а вычисление ее сканированием по тексту документа потребовало бы двухпроходного алгоритма для формирования аннотаций, что было бы не приемлемо с точки зрения быстродействия;
2) Для больших документов характерна внутренняя смысловая неоднородность и лучшие результаты должна давать частота слов в некоторой окрестности отбираемого фрагмента.

Нами был реализован алгоритм, который учитывал частоту слов в окне длиной в 1000 слов вокруг анализируемого фрагмента (то есть фрагмент находился в середине окна). Отбирались 10 слов, которые встречались в данном фрагменте наиболее часто. Использовалась следующая формула для вычисления веса:

$$Wfreq=Wb+Sum(\log2(Fk)),$$

где Wb – вес, вычисленный по базовому алгоритму,

Fk сколько раз слово встретилась в окне в 1000 слов, включающий фрагмент.

Таким образом, наибольший вес получали те фрагменты, которые кроме того, что содержали наибольшее число слов запроса, но и большее количество слов часто встречающихся в документе.

Алгоритм LRU-K.

Со времен работ Luhna[10] при статистической обработке документа в качестве метрики значимости термина обычно используется частота его встречаемости. По нашему мнению, данной метрике свойственен ряд недостатков. Во-первых, частота не несет информации о распределении слова в документе – распределено ли оно равномерно по всему документу или только в некоторых фрагментах, во-вторых, вычисление частот для фрагментов документов может требовать относительно много ресурсов.

В последнее время определенную популярность приобрели более сложные статистические модели, обычно основанные на Марковских цепях[11,12]. Однако эти модели достаточно сложны и имеют еще более высокую вычислительную сложность. Мы решили воспользоваться алгоритмом LRU-K, который является вариантом алгоритма «последний недавно использованный». Данный алгоритм хорошие результаты как алгоритм замещения буферов в различных индексных структурах [**Error! Reference source not found.**], при этом при его исследовании использовались аналогичные вероятностные модели[13], что навело нас на мысль о возможности его использования в приложении к определению значимости термина в тексте. Кроме этого, мы исходили из того известного из психологии предположения, что человек в быстрой памяти сохраняет только относительно малое количество объектов, поэтому алгоритмы класса «последний используемый» должны показывать хорошие результаты.

Нам не известно упоминаний в литературе использования подобных алгоритмов для анализа текста. Наша реализация выполняла следующее:

1) При инициализации создавалось 3 структуры данных: массивы слов и 2 массива с указателями на слова (array1 и array2) и длинами k.
2) Для каждого слова при обработке документа производились следующие действия:
    a) Поиск в массиве слов.
        1. Если слово не найдено, то ссылка на него помещалась в массив array1 в первую позицию, остальные позиции в массиве сдвигались, самое последнее слово удалялось из array1 и из массива слов.
        2. Если слово найдено и встречалось в array1, то оно из него удалялось и переносилось на первую позицию в array2. При этом, если array2 полностью заполнен, то последнее слово из него так же удалялось, как и в первом случае.
        3. Если слово найдено и уже было в array2, то оно просто перемещалось на первую позицию.

Легко можно показать, что если бы слова в тексте имели равную вероятность появления, то после обработки фрагмента текста, содержащего слов намного больше k, содержимое массива array2 совпадало бы с k наиболее часто встретившихся слов. То есть данный алгоритм можно рассматривать как один из вариантов оценки локальной частоты терминов, при предположении равномерного распределения слов. Однако, предлагаемый алгоритм, кроме этого, должен выделять слова, которые имеют не только высокую частоту, но и равномерно распределенные вблизи выбираемого фрагмента.

Вычисление веса фрагмента производилось также как и для алгоритма Freq, но вместо суммы по наиболее часто встречающимся словам, к весу, вычисленному по базовому алгоритму, прибавлялось количество слов из массива array2, встретившихся в анализируемом фрагменте.

Формирование аннотаций для больших документов.

В случае большого документа для пользователя становится важным навигация внутри документа, то есть не только сам факт релевантности документа, но и какая часть документа содержит релевантную информацию. Поэтому мы несколько изменили наш подход, формируя для таких документов несколько аннотаций. При этом использовалась разметка документа – выделение в тексте заголовков частей, статей и т.д.

При сканировании документа, формировались аннотации с помощью описанных алгоритмов, но не для всего текста, а для каждого фрагмента ограниченного заголовками частей. Таким образом, после обработки документа получали массив заголовков частей и сформированной для каждой части аннотаций. Далее из них отбиралось не более 5 элементов документа, имеющих наибольший вес. Пользователю в результате поиска выводилось название документа и названия частей, содержащих наиболее «тяжелые» аннотации, с гипертекстовыми ссылками на элементы документа и сами аннотации.

Описание экспериментов

Эффективность описываемых алгоритмов была проверена с помощью серии экспериментов. К сожалению, нам не известна общепринятая методика оценки качества подобных аннотаций. Широко

известные методики оценки качества информационного поиска[8] и автоматического аннотирования[1,9] нам не подходили по постановке задачи. Поэтому нами была разработана собственная методика, основанная на использовании оценок экспертов.

Постановка эксперимента.

Эксперимент проводился с привлечением группы экспертов, специалистов в области, по которой отбирались документы.

Каждому из них было поставлено задание в следующей формулировке:

Основной вопрос: «Помогла ли аннотация для получения информации, отвечающей на запрос?».

Разъяснение: «Необходимо определить, что вывод аннотаций помогает определить полезный или бесполезный данный документ с точки зрения поисковой задачи, т.е. ускорило ли это поиск нужной информации».

Таким образом, постановка задачи очень похожа на ad hoc задача конференции SUMMAC[1], но в несколько упрощенном варианте и более ориентированном на практическое приложение.

Эксперимент проводился следующим образом:

1) Эксперт выполнял поиск, получал список из 20 документов, в котором каждого документа представлено три варианта аннотации, сформированных описанными алгоритмами. Алгоритм, которым формировалась данная аннотация, никак не отмечался. Эксперт не мог определить, каким алгоритмом сформирована конкретная аннотация.
2) Эксперт для каждого документа отбирал варианты аннотирования, которые ему показались наиболее удовлетворяющими поставленной задачи. Можно было отметить ни одной, одну или все три аннотации.

В связи с ограниченными ресурсами каждый эксперт обрабатывал свой набор запросов, кросс-проверка не производилась.

Использовались следующие экспериментальные данные:

1) Базы данных, содержащие документы Российского законодательства (около 75 000 документов) и Санкт-Петербургского законодательства (около 40 000 документов)
2) Было отобрано 30 запросов из протоколов реальной работы пользователей.

Экспериментальная система была реализована в виде Web-приложения, доступного в интранет сети предприятия. В протокол вносился выбор эксперта, время формирования аннотаций.

Результаты эксперимента.

Результаты выбора экспертов приведены в Таблица 1.

Таблица 1

Алгоритм	Отданных экспертами голосов «За»
Базовый	14
Freq	40
LRU-K	43

Из этих результатов видно, что, если учитываются только слова запроса (Базовый алгоритм), эксперты редко оценивают такие аннотации как помогающие отбору документов. В случае если алгоритм учитывает другие слова документа, эксперты значительно чаще оценивают аннотации как полезные. Алгоритм LRU-K показал немного лучшие результаты, чем Freq.

Быстродействие алгоритмов, как суммарное время формирования аннотаций, для 600 документов (по 20 документов, отобранных как релевантных, для каждого из 30 запросов) приведено в Таблица 2.

Таблица 2

Алгоритм	Время (сек)
Базовый	25
Freq	127
LRU-K	35

Базовый алгоритм показал наилучшие результаты, так как он обрабатывает меньше всего данных. Алгоритм Freq требует более чем в 5 раз больше времени. Это объясняется тем, что вычисление локальной частоты для слов является достаточно трудоемкой операцией. Алгоритм LRU-K показал очень не плохое быстродействие, лишь на треть большее, чем базовый алгоритм.

Выводы

Результаты экспериментов позволяют сделать следующие выводы:

1) учет при формировании аннотации кроме слов запроса других слов документа позволяет значительно увеличить качество формирования аннотаций с точки зрения пользователя;
2) алгоритм LRU-K позволяет оценивать важность слов не хуже, а в некоторых случаях и лучше, чем при использовании классического подхода, основанного на частоте слов, при этом имеет значительно более высокое быстродействие.

Список литературы:

1) The TIPSTER SUMMAC Text Summarization Evaluation. Final Report // MITRE Technical Report - MTR 98W0000138 – 1998.
2) Sally Kleinfeldt, Jaideep Baphna. A Commercial Perspective on Hypertext Search Results. In Conference Information Doors Where Information Search and Hypertext Link May 30, 2000 San Antonio, Texas
3) dtSearch Support, How to generate a synopsis to display in search results http://support.dtsearch.com/faq/dts0194.htm
4) Shao-Fen, Liang Siobhan, Devlin John Tait. Can Automatic Abstracting Improve on Current Extracting Techniques in Aiding Users to Judge the Relevance of Pages in Search Engine Results? In Proceeding CLUK 2004
5) C.G. Wolf, S.R. Alpert, J.G. Vergo, L. Kozakov, Y. Doganata. Summarizing technical support documents for search: expert and user studies. IBM Systems Journal, Sept, 2004.
6) Min-Yen Kan. Automatic text summarization as applied to information retrieval: Using indicative and informative summaries. Thesis. Columbia University 2003.
7) Elizabeth J. O'Neil, Patrick E. O'Neil, and Gerhard Weikum, The LRU-K Page Replacement Algorithm for Database Disk Buffering, Proceedings of the 1993 SIGMOD International Conference on Management of Data, ACM Press, June 1993.
8) Inderjeet MANI. Summarization Evaluation: An Overview. In Proceedings of the NTCIR Workshop 2 Meeting on Evaluation of Chinese and Japanese Text Retrieval and Text Summarization. Tokyo: National Institute of Informatics.
9) Некрестьянов И.С. Кураленок И.Е. Оценка систем текстового поиска. Программирование, 28:226-242, 2002.
10) LUHN, H.P., 'The automatic creation of literature abstracts', IBM Journal of Research and Development, 2, 159-165 (1958).
11) John M. Conroy, Dianne P. O'leary, Text summarization via hidden Markov models, Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, p.406-407, 2001
12) Pascale Fung; Grace Ngai; Chi-Shun Cheung, Combining Optimal Clustering and Hidden Markov Models for Extractive Summarization, Proceedings of the ACL 2003 Workshop on Multilingual Summarization and Question Answering, 2003
13) O'Neil, E.J., O'Neil, P.E., Weikum, G., An Optimality Proof of the LRU-K Page Replacement Algorithm, Journal of the ACM 46(1), 1999