Информационные процессы, Том 12, № 3, стр. 320–329

Использование открытых баз данных как семантических словарей для автоматической обработки текстов: система Texterra

П.Е. Велихов

Поступила в редколлегию 22.08.2012

Аннотация—Описана система автоматической обработки текстов Texterra, разработанная в ИСП РАН, использующая меру семантической близости между статьями открытых баз данных, таких как Википедия. Детально описаны модули системы, которые отвечают за усвоение данных Википедии, предварительную обработку текстов, обработку графа ссылок Википедии и классификацию и информационный поиск в коллекциях произвольных документов с использованием Википедии, как семантического словаря.

1. ВВЕДЕНИЕ

Идея использования открытых баз данных большого объема для автоматической обработки текстов, задач классификации и информационного поиска относительно не нова и довольно подробно исследована в работах автора настоящей статьи [12],[13], а также в работах Мильне [14]. Однако практическая реализация этой идеи сталкивается с большим количеством сложностей, преодолению которых и посвящена настоящая статья.

Начнем с того, что напомним, что в основе рассматриваемого подхода лежит понятие меры семантической близости между статьями открытых баз данных. В работе автора настоящей статьи [13] проведено сравнение различных мер семантической близости по критерию вычислительной эффективности для двух задач, на которых основаны методы Texterra: вычисление близости двух статей, и ранжирования статей относительно заданной. Рассматривались как контентные меры, так и ссылочные: меры случайного блуждания и нерекурсивные ссылочные меры.

К контентным мерам относятся меры, основанные на векторной модели документа TF-IDF [15]. Но из-за того, что такого рода меры используют только ключевые слова без фразеологизмов и не используют семантику, содержащуюся в графе ссылкок, эти меры страдают от недостатка точности, как показано Менцером в [16]. Также, такие меры плохо подходят для задачи ранжирования, так как отсутсвуют вычислительно эффективные алгоритмы ранжирования по этим мерам.

В семейство ссылочных мер случайного блуждания попадают популярные рекурсивные функции семантической близости: SimRank [17] и мера близости вершин Ньюмана [18], а также PageRank [7] и функция Грина [19]. Все меры основаны на интуиции, что похожие вершины графа ссылок ссылаются на вершины, которые в свою очередь похожи. Алгоритмы рекурсивны, и в начальном шаге рекурсии все вершины похожи только на себя с коэфицентом 1. Далее, в рекурсии вершины графа, связанные с похожими вершинами становятся более похожими друг на друга со значением, зависящим от степеней вершин и коэфицента затухания рекурсии. Но, не смотря на высокое качество результатов, вычислительная сложность задачи ранжирования по этим мерам так же слишком высока, чтобы эффективно использовать их в практических системах.

Наконец, нерекурсивные меры близости, такие как косинус или мера Дайса, по качеству результатов не сильно уступают рекурсивным мерам, но при этом, основываясь на статистических свойствах графа ссылок Википедии, были разработаны эффективные алгоритмы ранжирования по этим мерам [13].

Вычислительная эффективность тех или ссылочных иных мер близости, конечно, различается в зависимости от типа графа, в виде которого представляется ссылочная структура используемой базы данных. Наибольший интерес с практической точки зрения представляют так называемые, безмасштабные графы [20], т.е. графы в которых степени вершин распределены по степенному закону (из чего следует наличие “тяжелых хвостов” – значительного числа вершин с очень большими степенями). В отличие от модели Эрдоша–Реньи, где узлы соединяются друг с другом с равной вероятностью, в модели Барабаши новые узлы присоединяются к существующим с вероятностью, пропорциональной степени последних узлов. На примере Википедии это означает, что скорее всего в новой статье появятся ссылки на известные статьи с большим количеством входящих ссылок, чем на малоизвестные. Более формально, в модели Барабаши в каждую единицу времени t в граф добавляется узел t с m новыми ребрами, смежными с существующими вершинами, где вероятность присоединения к вершине i равна:

$$\Pr(i,t) = \frac{k_i}{mt}$$

В работе [13] проведены измерения параметров одной из наиболее интересных, с точки зрения использования в автоматической обработке текстов, открытой базы данных Wikipedia [1]. Измерения показывают хорошее соответствие ссылочной структуры Википедии известной модели безмасштабных графов, так называемой модели Менцера [21]. Принимая в качестве исходной модели вышеназванную модель Мецнера, в работе [13] были предложены эффективные вычислительные алгоритмы для вычисления и ранжирования по мере Дайса. Эти алгоритмы применены в системе Texterra, описание которой приводится ниже.

Целью разработки системы Texterra является создание ситемы автоматического анализа текстов, используя семантику, извелеченную из открытых баз данных, в основном из Википедии. Система Texterra разрабатывалась как модульная клиент-серверная система в среде Java/J2SE. Основные модули системы – это: модуль усвоения Википедии; серверный модуль Texterra (отвечающий за хранение и использование словарей и графа ссылок Википедии); модуль анализа текстов; и модуль классификации и полнотекстового поиска. Общая архитектура системы приведена на диаграмме 1.

2. МОДУЛЬ УСВОЕНИЯ ВИКИПЕДИИ

Первая задача в работе Texterra – это извлечь все необходимые данные из корпуса Википедии. Википедия – бесплатная энциклопедия, и все ее данные свободно доступны в интернете. В дополнение к этому, так как Википедия привлекает большое количество исследователей, которым в работе требуется содержание статей Википедии, проект предоставляет периодические срезы всей Википедии для загрузки. Срезы доступны в форматах XML и SQL, но, к сожалению SQL формат содержит только ссылки меджу статьями, а XML формат не использует возможности языка XML для разметки текстов статей. Вместо этого, в XML срезе Википедии все статьи размечены плохо стандартизированным языком разметки MediaWiki[2], и лишь обернуты в разметку XML на уровне статей. Для MediaWiki, используемого в Википедии, не существует надежных синтаксических анализаторов, способных без ошибок обработать все статьи Википедии. Чтобы использовать данные Википедии, в рамках проекты Texterra мы реализовали свой синтаксический анализатор языка MediaWiki, Wiki2XML Parser[3], который преобразует формат MediaWiki в полностью размеченный формат XML. Wiki2XML не обрабатывает весь синтаксис MediaWiki, а только самые часто встречающиеся структуры языка.

Далее, размеченный корпус Википедии поступает в модуль усвоения Википедии системы Texterra. Здесь создаются следующие базы данных:

Таблица названия статей.

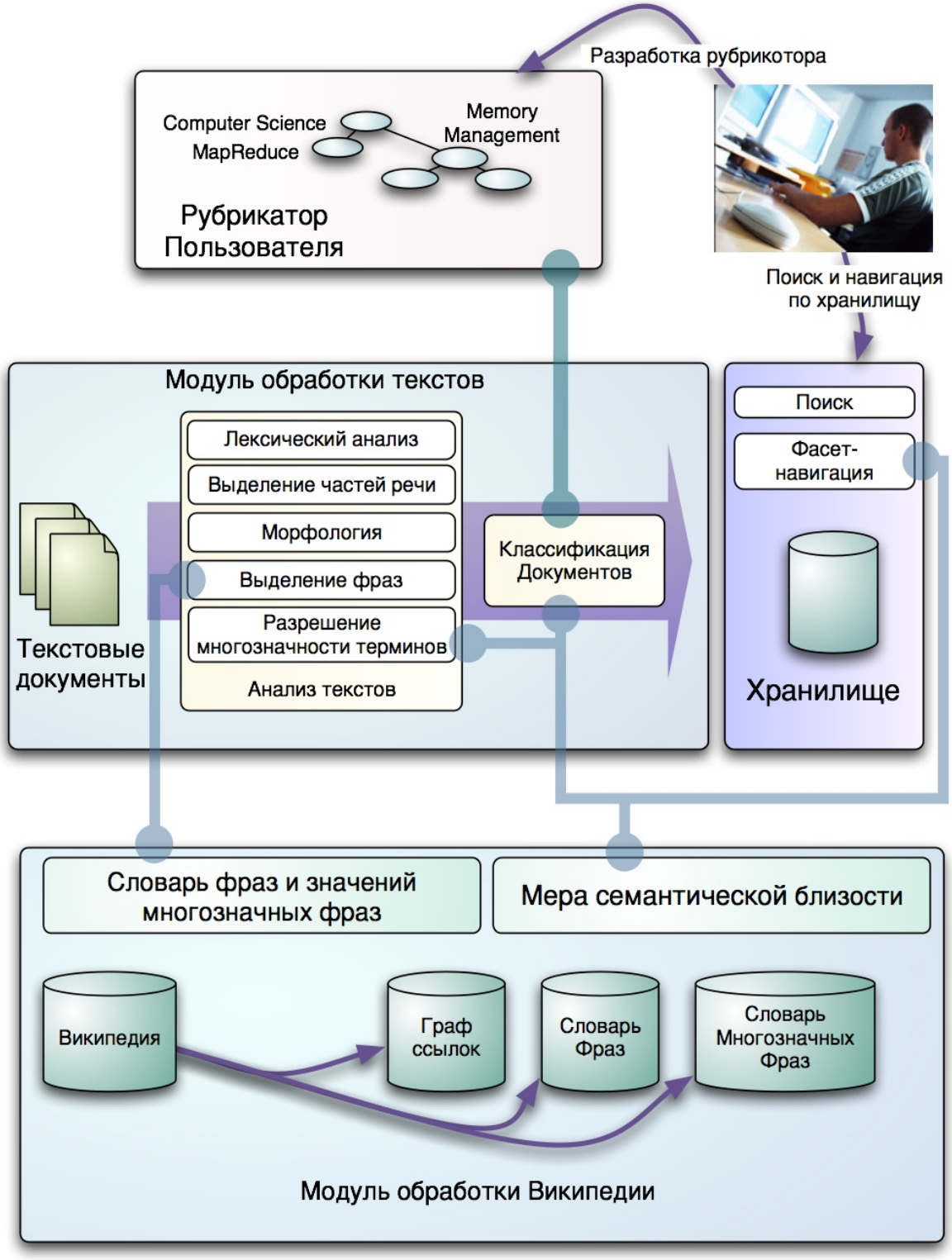

Диаграмма 1: Архитектура системы Texterra

Каждая статья в Википедии имеет название, состоящее из нескольких терминов, и уникальный идентификатор статьи. При усвоении данных из Википедии мы используем уникальные идентификаторы статей как идентификаторы узлов графа ссылок. Таблица названия статей используется системой в дальнейшем, как словарь фразеологизмов, при обработке произвольного текста каждому термину текста ставится в соответствие идентификатор соответвующей статьи. При этом, для эффективного поиска в таблице, названия статей обрабатывается таким же алгоритмом стеммизации, который будет использован при обработке произвольных текстов. Так как мы используем англоязычную Википедию, в системе Texterra мы используем очень простой стеммер, который просто убирает множественные окончания с концов слов. При этом, все равно возникают ситуации, когда два разных названия статей становятся одинаковыми, после применения стеммера. В таком случае, эти две статьи заносятся в словарь многозначных терминов.

Также, в англоязычной Википедии порядка 7 миллионов статей перенаправления. Например на статью “Artificial Intelligence” перенаправлено 48 страниц, например, такие как: “AI”, “Artifical Conciousness”, “Cognitive System”, “Machine Intelligence”. В системе Texterra, все статьи перенаправления отождествляются со статьями, на которые они перенаправляются, то есть в таблицу названия статей заносятся идентификаторы статей, на которых осуществляется перенаправление. Так же, может возникнуть ситуация, когда обработанная стеммеров статья перенаправления совпадает с другой статьей Википедии, причем отличной от той, на которую происходит перенаправление. В этом случае, так же как и с обычными статьями, статья перенаправления заносится в словарь многозначных терминов.

Таблица многозначных фразеологизмов.

Анлийская Википедия содержит около 900,000 многозначных терминов. Многозначные термины обычно организованы в отдельные страницы дизамбигуации, которые содержат все различные значения терминов. Напрмер страница дизамбигуации “Platform” содержит 22 отдельных термина, такие как: “Diving Platform”, “Oil Platform”, “Railroad Platform”, “Software Platform” и “Platform shoe”. Texterra отдельно обрабатывает такие страницы и составляет по ним словарь многозначных терминов. Также, добавляются многозначные термины, появившиеся в следствие работы стеммера. При этом, страницы дизамбигуации Википедии не всегда хорошо структурированы, и приходится применять набор эвристик, чтобы качественно выделить все многозначные термины. Обычно, различные значения терминов перечислены в первом параграфе страницы дизамбигуации, или на строках, начинающихся с символа “*”. Но часто, такие строки ссылаются на статьи, не являющиеся одним из значений термина. Например статья дизамбигуации термина “War” содержит следущую запись: “War, a song by Joe Satriani off The Extremist”, где ссылка поставлена на альбом “The Extermist”. Таким образом, если просто собирать все ссылки со страниц дизамбигуации, мы получим большое количество неверных значений многозначных терминов. Чтобы избежать таких ошибок, в системе Texterra выбираются термины, для которых в тексте перед ссылкой не встречается сам многозначный термин. Еще один источник многозначных терминов появляется из-за регистра символов Википедии. Википедия чувствтельна к регистру символов, например статьи перенаправления “SUX” и “Sux” указывают на два разных термина: “Sioux Gateway Airport” и “Suxamethonium chloride” соответсвенно. Но в системе Texterra нет чувствительности к регистру, так как произвольные тексты, для которых предназначена система, не всегда соблюдают регистры в сокращениях. Поэтому создаются отдельные значения терминов для таких случаев.

Таблица статей-категорий

В Википедии реализована простая онтология, где каждой каждая статья Википедии относится к одной или нескольким категориям. При этом категории представляются в Википедии как отдельные статьи, состоящие из списка всех обычных статей, входящих в данную категорию. Также, категории могут являтся под-категориями других категорий, таким образом существует граф категорий Википедии, который классифицирует все статьи энциклопедии. Система Texterra извлекает граф категорий Википедии в отдельную таблицу.

Граф ссылок Википедии

В процессе усвоения Википедии строится граф ссылок статей Википедии, который потом используется системой для вычисления семантической близости между терминами. Так как требования к вычислительной эффективности подсчета близости между терминами и ранжирования терминов по близости чрезвычайно высоки, Texterra хранит граф ссылок в оперативной памяти на серверной части приложения, и все алгоритмы затрагивающие граф ссылок также реализованы на серверной части.

3. СЕРВЕРНЫЙ МОДУЛЬ TEXTERRA

Серверный модуль реализован как серверный объект, используя технологию Java RMI. Клиентский модуль Texterra, который может находится на другом физическом устройстве, подключаются к серверу по RMI. Все словари и граф ссылок доступны только через обращение к серверному модулю системы. Серверный модуль реализует следущие функции через Java RMI:

`public Term lookupTerm( String name )`
Поиск термина в Википедии, возвращается термин, который может быть конечным термином, однозначно соответсвующим какой-то статье, или содержать ссылку на многозначный термин.

`public List<Term> homonyms(Term t)`
Вернуть весь список различных значений многозначного термина

`public List<CategoryTerm> getSubcategories(int id)`
Получить список всех категорий, к которым принадлежит данный термин

`public double computeSimilarity(Term t1, Term t2)`
Вычислить функцию семантического подобия для двух терминов Википедии

`public List<WeightedTerm> getSimilar(Term t, int n)`
Отранжировать термины Википедии относительно заданного, используя алгоритм точного ранжирования. Параметр k определяет, сколько наиболее похожих статей вернуть пользователю

`public List<WeightedTerm> getSimilarOL(Term t, int k, int n )`
Отранжировать термины Википедии относительно заданного, используя эвристический алгоритм ИС+ первые k.

`public List<WeightedTerm> getSimilarAL(Term t, int k, int n )`
Отранжировать термины Википедии относительно заданного, используя эвристический алгоритм АС+ первые k.

`public List<WeightedTerm> allPairsSimilarity(List<Term> l1, List<Term> l2)`
Вычисление всех пар семантических расстояний от списка терминов l1 к списку терминов l2. Эта процедура используется при разрешении многозначности терминов, а также при классификации документов.

`public List<WeightedTerm> pageRankCentrality( List<Term> l, nIterations n)`
Вычисление центральности терминов документа по алгоритму PageRank

Эффективность вычисления меры семантической близости и ранжирования терминов

Мы измерили скорость вычисления меры семантической близости между парами статей Википедии. Мы выбрали статью «Beer», чья степень составляет 2000, и считали семантическую близость со статьями различной степени. Результаты приведены на рисунке 1.

Рис. 1. Время вычисления меры близости для пар статей.

Для задачи ранжирования мы вычисляли первые 20 результатов ражнирования по евристике ИС + первые 40 для статей различных степеней. Эксперимент был поставлен на компьютере MacPro с 8-ми ядерным процессором с частотой 2 гигагерц и 16-ю гигабайтами оперативной памяти. Результаты приведены на рисунке 2.

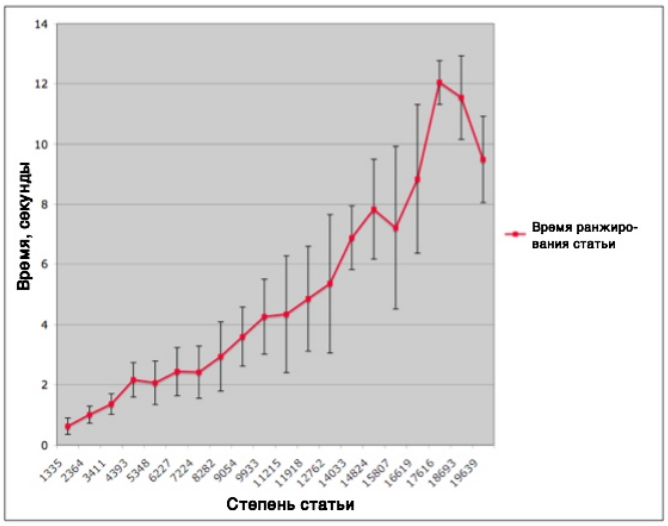

Рис. 2. Вычислительная эффективность ранжирования.

4. МОДУЛЬ АНАЛИЗА ТЕКСТОВ

Анализ произвольных текстов с использованием Википедии осуществляется клиентской частью системы, с обращениями к серверной части, когда требуется доступ к словарям или графу ссылок. Предварительная стадия обработки текстов перед тем, как они становятся доступными для классификации или поиска, представляет из себя следующую последовательность:

1. Лексический анализ
2. Выделение частей речи
3. Морфологический анализ
4. Выделение фразеологизмов Википедии
5. Разрешение многозначности фраз

6. Выделение доминантных фраз

В качестве лексического анализатора системы используется пакет Apache OpenNLP[4]. Лексический анализ, производимый через этот пакет, включает в себя детектор границ предложений и токенизатор. Полученный размеченный текст представляется в системе Texterra объектом TextModel, который содержит в себе оригинальный текст документа, аннотации, проставленные OpenNLP, фразы, выделенные из текста, а также доминантные термины документа.

Далее следует выделение частей речи из отдельных предложений, также с ипользованием пакета OpenNLP, на основании вероятностной модели. В системе Texterra мы использовали готовую обученную модель, которая была натренирована на корпусе Penn Treebank[5]. Полученные аннотации добавляются в TextModel. Морфологический анализ английского языка состоит из самых простых правил, нормализующих существительные единственного и множественного числа. Далее следует стадия выделения фраз Википедии по методу, предложенному Juneston[6]. В этой стадии выделяется маскимально длинная последовательность лексим, являющаяся названием статьи Википедии, с несколькими ограничениями: термин должен представлять из себя следующую стандартную последовательность частей речи: *((Adj|Noun)+|(Adj|Noun)*(NounPrep)?(Adj|Noun)*)Noun.* Полученные фразы сверяются с серверой частью Texterra, и если в словаре присутсвует данный фразеологизм, то в TextModel добавляется соответсвующий термин, который может быть определенным или многозначным.

После этой стадии разрешаются многозначные фразеологизмы. Для каждого мнозначного фразеологизма система собирает несколько однозначных, обнаруженных на предыдущей фазе фраз, в дальнейшем имменуемыми как контекст многозначного фразеологизма, и вычисляет семантическую близость между контекстом и каждым значением многозначной фразы. Алгоритм работает следующим образом: в окружении многозначного термина t выделяется контекст из нескольких однозначных терминов $c_1 \ldots c_n$, мы выбрали контекст размером в 7 терминов. Далее, из значений термина $t_1 \ldots t_k$ выбирается одно значение, максимизирующее сумму близости значения и терминов контекста:

$$t_{result} = \arg\max_{t_i} \sum_{j=1}^{n} sim(t_i, c_j)$$

Для оценки точности этого алгоритма мы использовали примеры разрешения многозначности из самой Википедии. В статьях Википедии ссылки имеют вид: *[подпись|ссылка]*, где *подпись* – это текст, который виден читателю статьи, а *ссылка* – это полное название статьи Википедии. Мы выбрали из Википедии ссылки, у которых подпись совпадает с многозначным термином Википедии, а ссылка ведет на одно из значений. Также, мы немного модифицировали меру близости для этой задачи: для каждой статьи мы использовали вес, обратный степени статьи. Затем мы проверили алгоритм на описаном множестве, где он правильно определил значения многозначных терминов в 72% случаев. При этом, в 90% случаев правильное значение попадает в первые три самых близких к конексту значений. Результат в 72% дает достаточно хорошие результаты для задач классификации текстов, информационного поиска и фасетнавигации. Для подсчета близости используется серверная функция allPairsSimilarity, которая возвращает все расстояния между терминами контекста и различными значениями многозначного термина. Многозначный термин в TextModel замещается определенным термином.

После выделения фразеологизмов Википедии в тексте и разрешения многозначных терминов модель документа представляет из себя множество терминов Википедии в пространстве меры семантической близости. В этом простанстве есть группы сильно связанных терминов, которые близки по смыслу друг к другу, а также встречаются термины, удаленные от всех других терминов документа. Обычно, маргинальные термины представляют собой несущественные термины документа или даже ошибки алгоритмов выделения терминов и разрешения многозначности. Но при наличие меры семантической близости, мы можем посчитать так называемую центральность терминов текста и присвоить соответсвующие веса терминам документа. В теории графов используются несколько мер центральности, самые известные из них – это центральность по кратчайшим путям и центральность по собственному значению. Центральность по кратчайшим путям оценивает процент кратчайших путей, проходящих через каждый узел графа, относительно общему количеству кратчайших путей:

$$C_B(v) = \frac{1}{n^2} \sum_{s,t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

В приведенной формуле переменная σ_{st} обозначает количество кратчайших путей из вершины *s* в вершину *t*, а $\sigma_{st}(v)$ – количество кратчайших путей проходящих через вершину *v*, обе переменные принимают значение 0 или 1. Мера центральности по кратчайшим путям считается самой качественной мерой и широко применяется для кластеризации графов и разбиении графа на тесно связанные подмножества [11]. Но эта мера обладает высокой вычислительной сложностью в $O(n^3)$, поэтому она применима только на графах малого объема.

Часто используемая мера центральности в графах – это центральность по собственному значению, похожей на известную меру PageRank[7]. Граф документов преобразуется в стохастическую матрицу, в которой итерационным методом находится стационарный вектор. Так как метод итерации сходится достаточно быстро, сложность вычисления данной меры центральности составляет $O(n^2)$. Метод pageRankCentrality в серверном модуле Texterra реализует эту меру центральности и возвращает веса для всех терминов документа, относительно этой меры.

После этой стадии обработки размеченный документ сохраняется в хранилище и используется в задачах классификации текстов и информационного поиска.

5. МОДУЛЬ КЛАССИФИКАЦИИ ТЕСКТОВ И ИНФОРМАЦИОННОГО ПОИСКА

Классификация документов

После предварительной обработки текста, система Texterra позволяет проводить классификацию текстов относительно линейного или иерархического рубрикатора, где каждая рубрика задается названием статьи Википедии. Из-за широкого охвата Википедии, почти любая интересующая пользователя рубрика будет представленна какой-нибудь статьей. Так как почти любая рубрика и фразеологизм классифицируемого документа имеют ненулевую степень семантического подобия, классификация имплементирована сделующим образом: подможество классифицируемых документов извлекается из хранилища, и подсчитывается семантическая близость от каждого термина документа до каждой рубрики, используя функцию allPairsSimilarity. Текстовый документ, состоящий из выделенных терминов $t_1 \ldots t_n$, при заданных рубриках $c_1 \ldots c_k$ классифицируется в рубрику, которая максимизирует сумму близости терминов к рубрике:

$$c = \arg\max_{c_i} \sum_{j=1}^{n} sim(c_i, t_j)$$

При этом возможно классифицировать документ сразу в несколько рубрик, используя сумму мер, как степень принадлежности к классу. Чтобы оценить качество классификации, мы провели эксперимент с новостным сайтом Google News [8], на котором новости классифицируются в несколько широких рубрик, таких как: Политика, Бизнес, Наука, Спорт, и т.д. Мы выбрали соответсвующие статьи Википедии в качестве рубрик для нашего алгоритма и при рубрикации получили 80% совпадения с рубрикацией Google News. При этом ошибки рубрикации оказались спорными и одинаково хорошо подходили нескольким рубрикам. Например темы Медицина и Бизнес в Google News очень часто пересекались, так как статьи описывали разработку медикаментов фармацевтическими компаниями.

Информационный поиск

Информационный поиск осуществляется традиционным методом – полнотекстовым поиском, реализованом в системе Apache Lucene[9]. Все документы, обработанные на первой стадии индексируются в системе Lucene. При этом используется стандартый лексический и морфологический анализ слов системы Lucene. Поэтому фразы документа, состоящие из нескольких слов, проиндексируются отдельно. При поиске, полнотекстовый запрос проходит обработку, которой подвергаются обычные документы в системе Texterra. Но система Lucene получает при этом обычный поисковый запрос, состоящий только из ключевых слов.[1] Получив набор доку-

[1] В системе Lucene есть возможность использовать фразы из нескольких ключевых слов для формирования запроса.

ментов, удволетворяющих полнотекстовому запросу, Texterra позволяет сузить набор результатов методом фасет-навигации, или наоборот, расширить его, добавляя термины, семантически близкие на термины, использованные в запросе.

Когда пользователь вводит запрос к отдельной коллекции документов, он часто получает слишком много результатов, среди которых сложно найти нужные документы. Фасет-навигация позволяет пользователю получить обзор тем полученных результатов и сузить свой запрос до нужной ему подтемы. В поисковых системах часто используют алгоритмы кластеризации для группировки результатов поиска в фасеты, как например в [10], но при этом встает задача вывода правильного названия кластера, которая пока не решается качественно. С использованием меры семантической близости, фасеты для навигации по результатам можно выбирать из терминов, наиболее похожих на запрос, но которые также встречаются в документах.

В Texterra реализован простой алгоритм фасет-навигации, который работает слудующим образом: при запросе пользователя q и наборе документов $d_1 \ldots d_n$, найденных полнотесктовой поисковой системой, фасеты вычисляются как пересечение терминов наиболее похожих на запрос q, с терминами из $d_1 \ldots d_n$. Из списка полученных фасетов, выбираются первые k фасетов, наиболее близкие к запросу. Такой простой алгоритм не всегда производит качественные фасеты, так как не учитывается смысловая близость терминов документа самому документу и фасеты могут отражать второстепенные термины. Улучшить такой алгоритм можно предварительно вычисляя центральность каждого термина относительно других терминов документа и ранжируя фасеты по совокупности семантической близости к запросу и средней величины центральности термина в документах-результатах запроса. При таком подходе получаются более качественные фасеты, но возрастает нагрузка на предварительную обработку текстов, так как подсчет центральности требует вычисления семантической близости между каждой парой терминов документа.

Противоположная ситуация – это когда пользовательский запрос слишком узкий, и не возвращает достаточное количество релевантных документов. Релевантные документы могут содержать термины, близкие по смылсу к ключевым словам запроса, но не совпадающие с ними. В таком случае, Texterra позволяет расширить запрос терминами, наиболее близкими к терминами поискового запроса. Для этого необходимо проранжировать все статьи Википедии относительно данного термина из запроса и выбрать первые несколько вариантов. При этом, полное точное ранжирование всех терминов нас не интересует, поэтому Texterra использует эффективный эвристический алгоритм ИС + первые k, реализованом функцией getSimilarOL в серверном модуле системы.

6. ЗАКЛЮЧЕНИЕ

Мы представили систему Texterra и подробно описали ее реализацию в клиент-серверной среде Java/SE. Используя семантические данные, извлеченные из Википедии, Texterra позволяет разрешать многозначные фразы в документах, определять доминантные фразы документа, производить рубрикацию текстов без использования обучающих выборок, помогать реформулировать запрос пользователя и вычислять фасеты при навигации по результатам полнотесктового запроса и по рубрикатору. Система основана на использовании меры семантической близости между статьями Википедии, которая вычисляется, используя граф ссылок Википедии. Так как граф ссылок достаточно небольшого размера, он помещается в операционную память современных компьютеров и все запросы к графу выполняются в приемлимое для практических систем время. Задача расширения запроса при полнотекстовом поиске решается эффективно в системе Texterra благодаря разработанным эвристическим методам ранжирования, описаными в [12] и [13].

СПИСОК ЛИТЕРАТУРЫ

1. Wikpedia: www.wikipedia.org
2. MediaWiki: www.mediawiki.org
3. Wiki2XML: modis.ispras.ru/texterra/download/wiki2xml.zip
4. Apache OpenNLP: opennlp.apache.org

5. Penn Treebank: www.cis.upenn.edu/~treebank
6. Juneston J.S. and Katz S.M., Technical Terminology: some linguistic properties and algorithms for identification in text. In Proceedings of ICCL-95.
7. L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web, 1998.
8. Apache Lucene: http://lucene.apache.org/core/
9. Google News: Google News: news.google.com
10. Wouter de Winter, Maarten de Rijke, Identifying Facets in Query-Biased Sets of Blog Posts, ICWSM 2007
11. M. E. J. Newman and M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69, 026113 (2004).
12. D. Turdakov, P. Velikhov. Semantic Relatedness Metric for Wikipedia Concepts Based on Link Analysis and its Application to Word Sense Disambiguation. In proceedings of SYRCoDIS, 2008.
13. П. Велихов «Меры семантической близости статей Википедии и их применение к обработке текстов», Журнал Информационные Технологии и Вычислительные Системы, 2009/01
14. Milne, D., Witten, I.H. and Nichols, D.M. (2007). A Knowledge-Based Search Engine Powered by Wikipedia. In Proceedings of the ACM Conference on Information and Knowledge Management (CIKM'2007)
15. Spärck Jones, Karen, A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation 1972.
16. Filippo Menczer, Combining Link and Content Analysis to Estimate Semantic Similarity. Proc. 13th Intl. WWW Conf. Alt. Track Papers and Posters, pp. 452-453, 2004.
17. Glen Jeh, Jennifer Widom. SimRank: a measure of structural-context similarity. Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, July 23-26, 2002, Edmonton, Alberta, Canada
18. Leicht, E. A. and Holme, Petter and Newman, M. E. J., Vertex similarity in networks, Phys. Rev. E, 73:026120, 2006.
19. Yann Ollivier, Pierre Senellart, Finding Related Pages Using Green Measures: An Illustration with Wikipedia, In AAAI (2007).
20. Albert R. and Barabási A.-L. Statistical mechanics of complex networks. Rev. Mod. Phys. 74, 47–97 (2002).
21. Menczer Filippo. Evolution of document networks. Proceedings of the National Academy of Sciences of the United States of America, 101: 5261-5265, 2004.

Using open databases as semantic dictionaries in automatic text processing tasks: Texterra system

Pavel Velikhov

We describe the Texterra system, developed at the Institute for System Programming, a text processing system that uses semantic relatedness measure between objects of open databases, such as Wikipedia. We give a detailed overview of all the modules of the system, including Wikipedia ingestion, text preprocessing, Wikipedia link analysis, classification and search modules.

KEYWORDS: Text processing, document classification, text search, semantic relatedness, word sense disambiguation