Извлечение информации из больших коллекций русскоязычных текстовых документов в среде Hadoop

© Д.О. Брюхов © Н.А. Скворцов
ИПИ РАН
Москва
brd@ipi.ac.ru nskv@ipi.ac.ru

Аннотация

В статье рассматриваются вопросы извлечения информации из больших коллекций русскоязычных текстовых документов в среде Hadoop. Предложена архитектура средств извлечения и анализа сущностей на основе технологических решений компании IBM, обеспечивающая масштабируемость относительно больших данных. Рассматриваются вопросы поддержки анализа русскоязычных текстовых документов в языке AQL.

1 Введение

Рост объемов данных практически во всех научных областях, экономике и бизнесе, государственных организациях достигает пропорций, характерных для эпидемии, оцениваемой, примерно, как ежегодное удвоение объема данных. Появляется существенный разрыв между генерируемым количеством данных и нашей способностью понимать, анализировать эти данные. В связи с этим активно развиваются технологии распределенного хранения и обработки «больших данных», в частности Hadoop.

В то же время большинство информации представлено в виде неструктурированных текстовых документов, таких, как новостные ленты, Вэб страницы, записи в социальных сетях, твиты, E-mails, отчеты. Все они содержат полезную информацию, такую, как сущности, взаимосвязи между сущностями, факты, тональность текста (sentiments). Извлечение полезной информации из неструктурированных текстов становится ключевым фактором во многих областях.

Извлечение информации (Information Extraction) – это процесс получения структурированной фактической информации из текстовых документов. Необходимость структурированной информации истекает из того, что существующие средства анализа данных (OLAP, Data mining) работают только со структурированной информацией. Чтобы использовать неструктурированные данные в средствах бизнес-аналитики необходимо, чтобы структурированные данные сперва извлекались из неструктурированных и слабоструктурированных данных.

В данной статье рассматриваются существующие подходы к трансформации контента больших неструктурированных (текстовых) коллекций данных в структурированную, агрегированную информацию. В статье предложен поход к анализу русскоязычных текстов в проекте с обеспечением масштабируемости относительно больших данных. Данная статья подготовлена в рамках проекта РФФИ 14-07-00548 «Трансформация средствами Hadoop-образных инфраструктур контента больших разно-структурированных коллекций данных в структурированную, агрегированную информацию».

Статья организована следующим образом. В разделе 2 показаны методы извлечения информации из текстовых документов. В разделе 3 рассмотрена архитектура решения от компании IBM для анализа больших объемов текстовых документов, на основе технологии Hadoop. Рассмотрена алгебра спанов и основанный на ней язык AQL (Annotation Query Language) для извлечения информации из текстовых документов. В разделе 4 представлены способы поддержки анализа русскоязычных текстовых документов в языке AQL. В разделе 5 представлен пример решения задачи тонального анализа над большими текстовыми данными в среде Hadoop.

2 Методы извлечения информации из текстов

Извлечение информации можно разделить на следующие этапы:

- Распознавание сущностей (Entity Recognition).
- Аннотация сущностей (Named Entity Annotation).
- Устранение неоднозначности сущностей (Entity Disambiguation).

Труды 16-й Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» — RCDL-2014, Дубна, Россия, 13–16 октября 2014 г.

Распознавание сущностей является задачей извлечения имен сущностей из текста. Возможны два варианта распознавания сущностей: с помощью словарей и с помощью регулярных выражений:

• Словари содержат список имен и фраз определенного класса (например, список стран или компаний), которые будут извлекаться из текста. Если словарь будет не полным, то пропущенные имена просто не будут распознаны в тексте.

• Словари не всегда подходят для извлечения сущностей, поскольку зачастую просто невозможно составить полный словарь, например для списка фильмов, книг, имен людей. Но существуют классы сущностей, которые определяются конкретными правилами, например, адреса электронной почты, даты. В этом случае для извлечения сущностей из текста можно использовать регулярные выражения.

Аннотация сущностей – это задача извлечения имен сущностей из текста и их аннотация классом из заданного набора классов (например, классы Person, Organization, Location). Существует два подхода к аннотированию сущностей [1]: на основе правил и на основе статистических моделей.

• Аннотация сущностей на основе правил [2], [3], [4], [5] использует правила вида PATTERN => CLASS. Это правило аннотирует фрагмент текста классом CLASS, если он удовлетворяет паттерну PATTERN. Паттерн представляет собой последовательность свойств (features), каждое из которых может быть регулярным выражением, словарем или другим паттерном. Обычно правила задаются вручную, но могут и автоматически выявляться из текста.

• Аннотация сущностей на основе статистических моделей [6], [7], [8], [9] основана на разделении исходного текста на вектор слов (токенов) X и аннотировании каждого из этих слов классом из заданного вектора классов Y. Аннотирование происходит на основе вычисления вектора классов Y, максимизирующего функцию $\Sigma i \Sigma j\ wj\ f\ j(X,\ i,\ yi)$, где свойство $f\ j$ – это функция, отображающая вектор слов X, позицию i в X и класс y в значении типа real, а $w\ j$ – это вес соответствующего свойства.

Многие имена имеют неоднозначное толкование, например, Форд – марка машины или имя человека. Примерами устранения неоднозначности сущностей могут служить следующие методы:

• Разрешение неоднозначностей по контексту (context disambiguation) осуществляется на основе сравнения контекста имени в тексте (контекст формируется как множество всех имен, встречающихся в одном предложении с рассматриваемым именем) и контекста имени в базе знаний (контекст определяется как множество всех имен, связанных с рассматриваемым именем, в базе знаний).

• Разрешение неоднозначностей по важности (prior disambiguation) осуществляется по наиболее часто встречающемуся значению этого имени в заданной коллекции документов.

• Разрешение неоднозначностей по критерию согласованности (coherence disambiguation). Критерий согласованности утверждает, что имена, встречаемые в одном документе, должны быть связаны в базе знаний.

3 Текстовая аналитика над большими данными

3.1 Обработки больших текстов

Технология Hadoop [10], [11] предлагают эффективную программную инфраструктуру для обработки больших объемов данных. Технологии IBM дополняют эту инфраструктуру аналитическим ПО для анализа текстовых документов и основанные на Eclipse инструментальные средства для разработки приложений.

На рисунке 1 показана архитектура решения IBM для анализа текста при помощи InfoSphere BigInsights [12] и Streams [13]. Разработчики используют декларативный язык (AQL) для создания экстракторов текстовых данных. Исполняющий механизм оптимизирует декларативные инструкции на языке AQL. На этом шаге оптимизации выводится скомпилированный план, определяющий, как BigInsights будет обрабатывать коллекцию входных документов, хранящихся в распределенной файловой системе HDFS. При развертывании на BigInsights-кластере экстракторы текстов можно вызвать посредством языка запросов и сценариев Jaql [14], [15], Java API или средства анализа данных BigSheets.

3.2 Алгебра спанов

Алгебра спанов [16], [17] основана на упрощенной реляционной алгебре с минимальными расширениями для обработки текстов. В алгебре спанов существует 3 типа данных: спан (span), кортеж и отношение.

• Спан – это упорядоченная пара <begin, end>, которая указывает на сегмент в текстовом документе;

• Кортеж – это конечная последовательность w спанов <s1, ..., sw>;

• Отношение – это мультимножество кортежей одинаковой длины.

Каждый оператор в алгебре спанов имеет ноль или больше входных отношений и одно выходное отношение. Существует 3 вида операторов: реляционные операторы, операторы извлечения спанов и операторы агрегирования спанов.

Реляционные операторы являются аналогами операторов реляционной алгебры, таких как: Select, Project, Join и др. Основным дополнением является поддержка ряда предикатов для работы со спанами (например, предикат FollowsSpan).

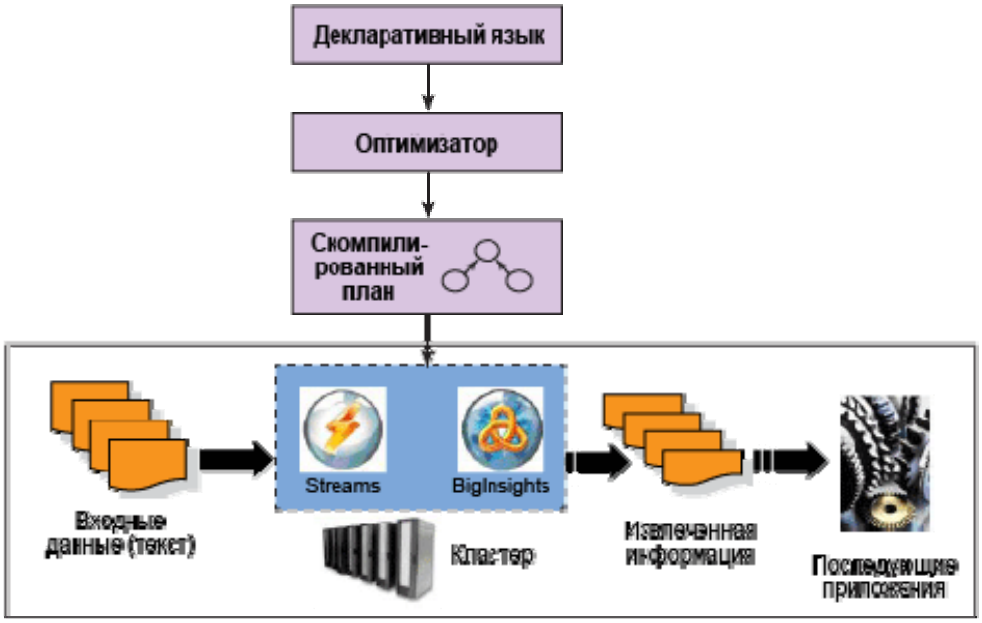

Рис. 1. Архитектура решения IBM для анализа текста

Операторы извлечения спанов идентифицируют сегмент текстового документа, который удовлетворяет заданному условию, и создают спаны, соответствующие каждому такому сегменту. Существует 2 оператора извлечения спанов:

• Матчер регулярных выражений (Regular expression matcher (Ere)). По заданному регулярному выражению r, матчер регулярных выражений Ere(r) идентифицирует все непересекающиеся сопоставления, когда r вычисляется над текстом s слева направо. Результатом Ere(r) является спанов, соответствующих этим сопоставлениям.

• Матчер словарей (Dictionary matcher (Ed)). По заданному словарю dict, состоящему из слов и фраз, матчер словарей Ed(dict) создает множество спанов, каждый из которых соответствует одной из записей в словаре dict.

Операторы агрегирования спанов по заданному входному множеству спанов создают множество выходных спанов, выполняя определенные агрегирующие операции. Операторы агрегирования спанов включают:

• Оператор Block. Оператор Block используется для идентификации сегмента текста, состоящего из близких фрагментов (спанов) в исходном тексте. Оператор Block содержит два параметра: count – определяет сколько фрагментов (спанов) могут формировать блок, и separation – определяет расстояние, в котором фрагменты текста считаются близкими;

• Оператор Consolidate. В случае когда несколько паттернов извлечения информации используются для идентификации одного и того же понятия, разные паттерны зачастую продуцируют одинаковые или пересекающиеся сопоставления (matches). Для разрешения таких «дублированных» сопоставлений определены несколько функций консолидирования (consolidation functions), которые определяют, как поступать в таких случаях. Например, Containment consolidation определяет, что спаны, которые включаются (containment) в другие спаны, отбрасываются.

3.3 Язык AQL

Annotation Query Language (AQL) [18] – это язык для разработки экстракторов текстовой аналитики в среде IBM InfoSphere BigInsights. Экстрактор – это программа, написанная на языке AQL, которая извлекает структурированную информацию из неструктурированных и слабоструктурированных текстовых документов. AQL основан на алгебре спанов [16], [17]. AQL является развитием проекта SystemT [22], [17].

AQL является декларативным языком, синтаксис которого похож на синтаксис языка SQL. Модель данных языка AQL подобна стандартной реляционной модели. Данные хранятся в кортежах, коллекция кортежей формируют отношение. Поддерживаются следующие типы: Integer, Float, Text, Span, List.

AQL экстрактор состоит из коллекции представлений (views), каждое представление определяет отношение. Ниже приведен пример представления на языке AQL для извлечения номеров телефонов:

```
create view PhoneNumbers as
   extract regex /(\d{3})(-)(\d{3})(-)(\d{4})/ on d.text
return
      group 1 as areacode and
      group 3 as exchange and
      group 5 as extension
   from Document d;
```

Язык AQL содержит следующие основные компоненты:

• Create view – Создает представление и определяет его структуру;

• Extract – Предоставляет средства для извлечения информации из текста с помощью регулярных выражений;

• Create dictionary – Определяет словарь слов и фраз для использования в операторе extract, который может быть создан как внутри AQL файла, так и во внешнем файле;

• Select – Механизм для конструирования сложных паттернов из простых блоков;

• Pattern – Используется для нахождения соответствий на основе паттернов;

• Part-of-speech – Оператор идентификации различных частей речи в тексте (к сожалению, не поддерживает русский язык);

• Built-in functions и User-defined functions – Встроенные и пользовательские функции, используемые в правилах извлечения информации.

Для поддержки разработки экстракторов существует плагин InfoSphere BigInsights Tools для среды разработки Eclipse.

4 Поддержка русского языка

Поддержка многоязычности в IBM InfoSphere BigInsights Text Analytics (а также в Streams) реализована на уровне выделения базовых элементов текста (токенов) и анализа частей речи. Анализ частей речи используется для определения местоположения в тексте разных частей речи в определённой форме и извлечения слов по найденным местам текста. Информация о частях речи формируется многоязычным токенайзером в виде строки со списком тегов, соответствующих частям речи в определённой форме слова. Части речи и их формы обозначаются по-разному в соответствии с морфологией определенного языка.

Существующий многоязычный токенайзер с поддержкой анализа частей речи для определённого набора языков реализован средствами IBM на основе каркаса Apache UIMA, являющегося стандартом средств обработки неструктурированной информации.

К сожалению, поддержка русского языка в IBM InfoSphere BigInsights Text Analytics (Streams) ограничивается токенайзером, не включая анализ частей речи. Выделение базовых элементов текста для русского языка не содержит каких-либо особенностей относительно других алфавитных языков и основано на выделении пробельных символов и символов пунктуации. Несмотря на то, что в IBM LanguageWare присутствуют некоторые русские словари, в IBM BigInsights Text Analytics не реализована возможностью работы с частями речи на русском языке.

Однако и это полностью не устраняет проблему. В языке AQL нет средств определения основной формы слова. Для английского языка, в частности, это не принципиально, и для разбора форм используются регулярные выражения. Однако сложность морфологии русского языка, заключающаяся в разнообразии словоформ и правил их образования, не позволяет пользоваться этими средствами, и определение основной формы слова и атрибутов для каждой анализируемой словоформы в тексте становится принципиально важным для работы с русскоязычными текстами. Поэтому поддержка русского языка в существующих средствах анализа частей речи в AQL была бы также недостаточна без добавления в язык AQL средств генерации основной формы слова.

Для решения проблемы поддержки русского языка в IBM BigInsights Text Analytics проведены эксперименты в двух направлениях:

• применение имеющихся словарей русского языка IBM LanguageWare для аннотирования текста информацией об основных формах слов, морфологических атрибутах исходных форм слов, в том, числе, частях речи, и дальнейшего использования этой информации в BigInsights Text Analytics;

• применение библиотеки морфологического разбора (например, AOT [19], RCO Morphology SDK [20], ABBYY Retrieval & Morphology [21].

В первом варианте IBM LanguageWare в результате обработки текста генерирует документы в формате XMI, в которых среди прочего присутствует информация об основных формах слов и морфологических атрибутах присутствующих в тексте словоформ. Однако формат XMI оказывается неудобным для дальнейшего анализа текста, и необходим предварительный разбор формата XMI программными средствами для получения текста, аннотированного тегами. К тому же, не ясно насколько влияет на качество результата отсутствие в LanguageWare специализированного русского словаря частей речи. Второй вариант включает вызов из языка AQL пользовательской функции (UDF), которая обращается к библиотеке соответствующего морфологического анализатора. Возможны два варианта вызова средств анализа текста: пословно или весь текст целиком.

5 Пример решения задачи тонального анализа над большими коллекциями текстовых данных

В этом разделе мы рассмотрим предлагаемый нами подход на задаче извлечения сущностей (на примере организаций) и определения их тональной окраски из сообщений в социальных сетях (ВКонтакте, твиттере). На рисунке 2 представлена архитектура средств решения этой задачи в среде Hadoop. Hadoop автоматически распараллелит выполнение этой задачи по различным узлам кластера.

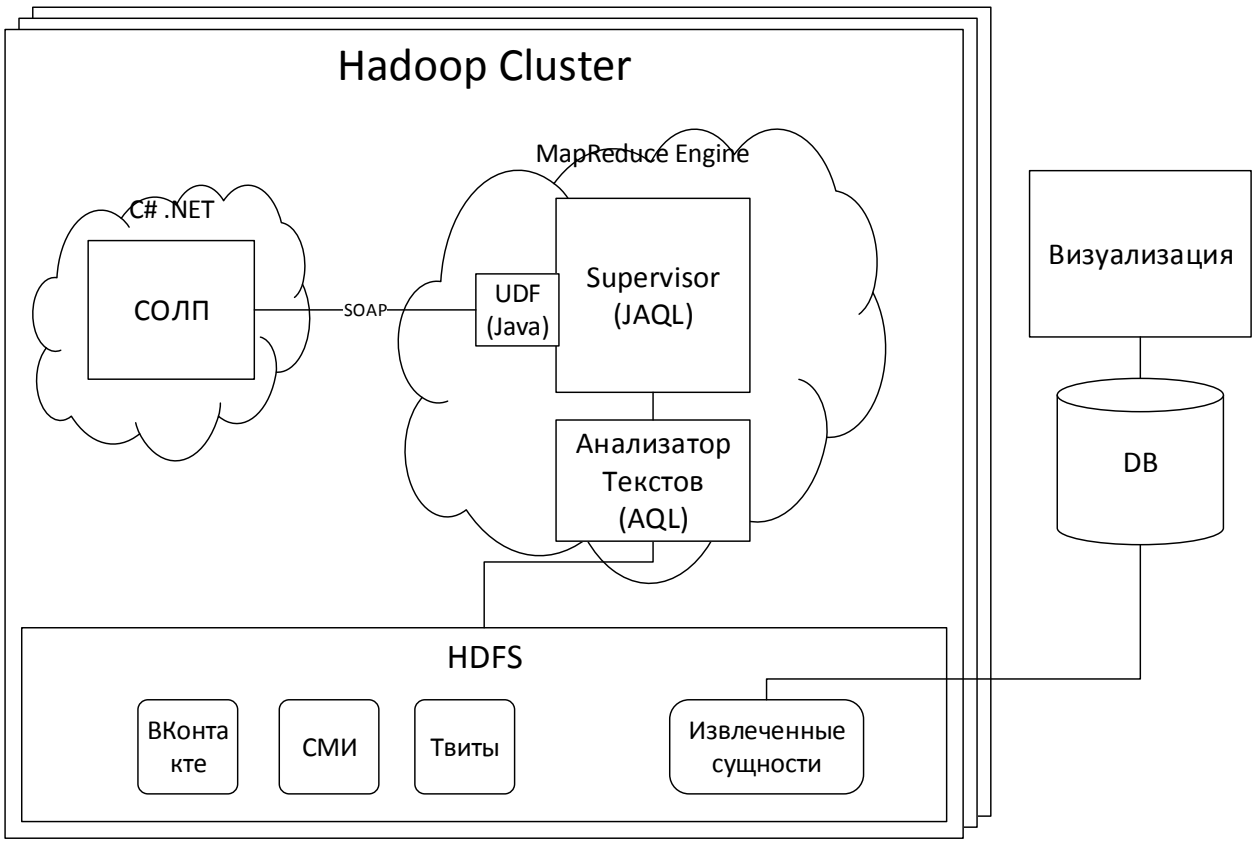

Рис. 2. Архитектура средств извлечения и анализа сущностей в среде Hadoop

5.1 Сценарий работы

Сценарий работы задаче извлечения сущностей (на примере организаций) и определения их тональной окраски выглядит следующим образом:

0. Сбор исходных коллекций текстов из социальных сетей (твиты, ВКонтакте) и загрузка их в HDFS, позволяющую хранить большие объемы данных. Чтобы не было множества небольших файлов для каждого сообщения, сообщения объединяются в файлы размером около 1 Гб. Тексты представляются в формате CSV, состоящего из 2 столбцов: идентификатор сообщения и текст сообщения;

1. На вход компонента Supervisor поступает список собранных исходных текстовых документов. Средствами Hadoop обработка этих документов распараллеливается по файлам и по блокам, в которых хранятся эти файлы;

2. Каждый такой фрагмент текста передается компоненту морфологического анализа (СОЛП) [23] для преобразования всех слов в текстовом документе к нормальной форме. Для этого компонент СОЛП должен быть установлен на каждый компьютер Hadoop кластера;

3. Полученные нормализованные текстовые документы передаются компоненту Анализатор Текстов;

4. Компонент Анализатор Текстов извлекает найденные в тексте организации и определяет их тональную окраску. Результат сохраняется в HDFS;

5. Извлеченные сущности (организации) импортируются из HDFS в реляционную базу данных для дальнейшего анализа существующими средствами визуализации и анализа данных.

5.2 Компоненты

Компонент Supervisor

Компонент Supervisor, написанный на JAQL, ответственен за взаимодействие всех компонентов системы. На входе он берет исходные текстовые документы. Затем передает их компоненту морфологического анализа (СОЛП) для преобразования всех слов в текстовом документе к нормальной форме. Полученные текстовые документы в нормальной форме передаются компоненту Анализатор Текстов.

Компонент СОЛП

Модуль СОЛП служит для морфологического анализа текстов. СОЛП реализован на языке .NET, под Unix он запускается с помощью программного продукта Mono. Модуль может работать как в пакетном режиме, так и в режиме сервера, обрабатывающего SOAP-запросы.

Входными данными являются тексты, оформленные в формате CSV в кодировке UTF-8, где каждая строка представляет конкретное сообщение, собранное из социальных сетей, в следующем виде: идентификатор сообщения, текст сообщения.

Пример входного текста:

```
"33650_559","Родители отдали Ивана в музыкальную школу по классу фортепиано. А могли бы отдать на баян или контрабас."
"36147_2936","МТС подкинул проблем ) ни один номер не работает %) сеть легла похоже у них"
```

Результирующие данные представляются в том же формате.

```
"33650_559","РОДИТЕЛЬ ОТДАТЬ ИВАН В МУЗЫКАЛЬНЫЙ ШКОЛА ПО КЛАСС ФОРТЕПИАНО. А МОЧЬ БЫ ОТДАТЬ НА БАЯН ИЛИ КОНТРАБАС."
"36147_2936","МТС ПОДКИНУТЬ ПРОБЛЕМА ) НИ 1 НОМЕР НЕ РАБОТАТЬ %) СЕТЬ ЛЕЧЬ ПОХОЖИЙ У ОНИ"
```

Компонент Анализатор Текстов

Компонент Анализатор Текстов служит для извлечения найденные в тексте сообщений организаций и определения их тональной окраски. Компонент написан на языке AQL.

Входными данными являются тексты, оформленные в формате CSV в кодировке UTF-8 в следующем виде: сначала идентификатор сообщения, затем текст сообщения в нормализованном виде. Также на входе компонент получает словари: словарь позитивных слов, словарь негативных слов и словарь организаций. Словари представляются в текстовом виде, где каждая строка содержит соответствующий термин.

Пример словаря позитивных слов:

```
АВТОРИТЕТНЫЙ
БЕЗОПАСНЫЙ
БЛАГОПОЛУЧИЕ
БЛАГОПОЛУЧНЫЙ
```

Извлечение организация осуществляется на основе соответствующего словаря. На языке AQL это выглядит следующим образом:

```
create view Organizations as
extract dictionary 'OrganizationDict'
  on D.text as name
from Document D;
```

Исходя из того, что сообщения в социальных сетях имеют небольшую длину, для данного примера мы используем простую формулу определения коэффициента тональной окраски извлекаемых сущностей (организаций). Коэффициент тональной окраски равен разнице в количестве позитивных и негативных слов в рамках одного и того же сообщения.

Сперва с помощью регулярных выражений мы разбиваем исходный документ на конкретные сообщения:

```
create view Lines as
extract
    split using B.boundary
    retain right split point
    on B.text
    as text
  from (
    extract
      D.text as text,
      regex /(\n)/ on D.text as boundary
      from Document D
  ) B
;

create view Messages as
extract regex /\"(\w+)\",\"(.*)\"/ on L.text
  return group 1 as name
    and group 2 as text
from Lines L;
```

Затем выявляем отдельно сообщения, где встречаются организации и позитивные/негативные слова:

```
create view MessagesOrganizationPositive as
select
  m.name
  , o.name as organization_name
  , pw.word as word
  , 1 as cnt
from Messages m, Organizations o, PositiveWords pw
where
  Contains(m.text, o.name)
  and Contains(m.text, pw.word)
;

create view MessagesOrganizationNegative as
select
  m.name
  , o.name as organization_name
  , nw.word as word
  , -1 as cnt
from Messages m, Organizations o, NegativeWords nw
where
  Contains(m.text, o.name)
  and Contains(m.text, nw.word)
;
```

В заключении, для каждой найденной организации мы подсчитываем коэффициент тональной окраски этой организации:

```
create view MessagesOrganizationSentiment as
(select
  mop.name
  , mop.organization_name
  , mop.cnt
from MessagesOrganizationPositive mop)
union all
(select
  mon.name
  , mon.organization_name
  , mon.cnt
from MessagesOrganizationNegative mon)
;
```

```
create view OrganizationSentiment as
select
  GetText(mos.name) as message_id
  , GetText(mos.organization_name) as
organization_name
  , Sum(mos.cnt) as sentiment
from MessagesOrganizationSentiment mos
group by
  GetText(mos.name)
  , GetText(mos.organization_name)
;
```

Результатом выполнения компонента является таблица, содержащая идентификатор сообщения, извлеченная из этого сообщения организация, и коэффициент тональной окраски этой организации. Пример результирующей таблицы показан ниже:

message_id	organization_name	sentiment
33650_553	ОКСФОРДСКИЙ УНИВЕРСИТЕТ	0
36147_2826	СОЮЗ	1
36147_2936	МТС	-1
43079_4740	СОЮЗ	1
61454_2016	СОЮЗ	-15

Полученная таблица используется для дальнейшего анализа существующими средствами визуализации и анализа данных.

6 Заключение

В статье рассмотрен обзор методов извлечения информации из текстов, включающих в себя методы распознавание сущностей, аннотации сущностей и устранения неоднозначностей.

Рассмотрена технология, предложенная компанией IBM, для обработки больших объемов текстовых документов в среде Hadoop. Она включает в себя:

• Декларативный язык для идентификации и извлечения информации из текстовых документов AQL (Annotation Query Language), основанный на алгебре спанов;

• Создаваемые пользователями или предметно ориентированные словари;

• Программные средства на основе Eclipse, позволяющие разработчикам создавать правила извлечения информации – инструменты обнаружения шаблонов и создания регулярных выражений (regex).

Также рассмотрены варианты поддержки анализа русскоязычных текстов в технологии IBM InfoSphere BigInsights.

Предложенная архитектура средств извлечения и информации продемонстрирована на задаче извлечения сущностей (на примере организаций) и определения их тональной окраски из сообщений в социальных сетях (ВКонтакте, твиттере).

Литература

[1] Sunita Sarawagi: Information Extraction. Foundations and Trends in Databases 1(3): 261–377 (2008).

[2] D. E. Appelt, J. R. Hobbs, J. Bear, D. J. Israel, and M. Tyson, “Fastus: A finite-state processor for information extraction from real-world text,” in IJCAI, pp. 1172–1178, 1993.

[3] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan, “Gate: A framework and graphical development environment for robust nlp tools and applications,” in Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics, 2002.

[4] M. E. Califf and R. J. Mooney, “Relational learning of pattern-match rules for information extraction,” in Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99), pp. 328–334, July 1999.

[5] S. Soderland, “Learning information extraction rules for semi-structured and free text,” Machine Learning, vol. 34, 1999.

[6] C. D. Manning and H. Schutze, Foundations of Statistical Natural Language Processing. Cambridge, MA: The MIT Press, 1999.

[7] P. Domingos, D. Lowd: Markov Logic: An Interface Layer for Artificial Intelligence. Morgan & Claypool 2009.

[8] L. Getoor, B. Taskar (Eds.): Introduction to Statistical Relational Learning. MIT Press 2007.

[9] D. Koller, N. Friedman: Probabilistic Graphical Models: Principles and Techniques. MIT Press, 2009.

[10] Hadoop. http://hadoop.apache.org/

[11] Tom White. Hadoop: The Definitive Guide. O'Reilly Media; Third Edition edition (May 26, 2012).

[12] IBM InfoSphere BigInsights Information Center. http://pic.dhe.ibm.com/infocenter/bigins/v2r1/index.jsp

[13] Sherif Sakr. An introduction to InfoSphere Streams. IBM developerWorks. http://www.ibm.com/developerworks/library/bd-streamsintro/

[14] Jaql Documentation. https://code.google.com/p/jaql/w/list

[15] Kevin S. Beyer, Vuk Ercegovac, Rainer Gemulla, Andrey Balmin, Mohamed Eltabakh, Carl-Christian Kanne, Fatma Ozcan, Eugene J. Shekita. Jaql: A Scripting Language for Large Scale Semistructured Data Analysis. VLDB 2011.

[16] Spanners: A formal framework for information extraction, with Benny Kimelfeld, Frederick Reiss, and Stijn Vansummeren. Proc. 2013 ACM Symposium on Principles of Database Systems (PODS'13), pp. 37–48.

[17] SystemT: a system for declarative information extraction, Rajasekar Krishnamurthy, Yunyao Li,

Sriram Raghavan, Frederick Reiss, Shivakumar Vaithyanathan, Huaiyu Zhu, ACM SIGMOD Record 37(4), 7–13, ACM, 2009.

[18] Annotation Query Language. http://www-01.ibm.com/support/knowledgecenter/SSPT3X_2.1.1/com.ibm.swg.im.infosphere.biginsights.aqlref.doc/doc/aql-overview.html?lang=en

[19] Автоматическая Обработка Текста (АОТ), http://www.aot.ru/

[20] RCO Morphology SDK, http://rco.ru/

[21] ABBYY Retrieval & Morphology (ARM) Engine, http://www.softfor.ru/products/abbyy/557.html

[22] System-T Project. http://researcher.ibm.com/researcher/view_project.php?id=1264

[23] И.П. Кузнецов, А.Г. Мацкевич. Семантико-ориентированные системы на основе баз знаний: монография. – М.: Связьиздат, 2007. – 173 с.

Information Extraction from Large Collection of Russian Text Documents in Hadoop Environment

Dmitry O. Briukhov, Nikolay A. Skvortsov

The paper describes issues of information extraction from large collections of the Russian text documents in Hadoop environment. The architecture for entity extraction and analysis tools basing on IBM technologies is introduced. The methods for supporting analysis of documents in Russian language are described.