119270, Москва, Лужнецкая наб., д. 6,
стр.1, офис 214, ООО «ЭР СИ О»
Тел./факс: (495) 287-98-87
E-mail: info@rco.ru
http://www.rco.ru

Руководство разработчика RCO Top Extractor SDK

RCO Top Extractor – средство построения информационного портрета текста

Версия 1.0

(Microsoft Windows)

Москва, 2010

Содержание

Введение

Ядром приложения **RCO Top Extractor** является класс **CTopExtractor**, реализованный в файлах **top_extractor.cpp**, **top_extractor.h**. **CTopExtractor** посредством программного обеспечения (ПО) **RCO Fact Extractor** выделяет из анализируемого текста необходимые для построения его информационного портрета сведения. Программист-пользователь при написании собственного приложения может отфильтровать нужные ему данные и, применяя **CTopExtractor**, сформировать на их базе общий реферат документа, реферат по каждой теме или xml-файл с данными, необходимыми для создания карты ассоциативной сети между темами. Пример такого рода представлен в тестовом приложении (**TopExtractorXML.cpp**), поставляющемся вместе с исходным кодом **RCO Top Extractor**.

В роли тем выступают слова и словосочетания, обозначающие предметы и события. В информационный портрет текста не включаются признаки, описываемые прилагательными, наречиями или адъективными существительными. Также не учитываются элементы смысла, характеризующие позицию автора по отношению к предметам и событиям, и выражающиеся разными языковыми средствами как лексическими (слова служебных частей речи, строевые глаголы), так и грамматическими (вид, время и залог глагола).

Значимость темы оценивают по следующим факторам:

- самостоятельность термина, используемого в качестве темы, – употребляется ли он в составе словосочетаний или независимо;
- роль термина в предложении (например, позиция подлежащего соответствует основному фокусу внимания автора);
- близость к началу документа;
- частота встречаемости в тексте.

Вес темы (от 1 до 100) отражает ее значимость для автора текста по сравнению с другими темами. При оценке значимости темы используется комплексный лингвистический критерий, который учитывает особенности построения текста автором (тема-рематическое членение, синтаксические позиции). Программист может задать в своем приложении минимальный порог по весу, ниже которого темы не включаются в семантическую сеть.

Рефераты состоят из наиболее репрезентативных предложений текста. При отборе предложений учитываются количество и значимость входящих в них терминов. При этом обеспечиваются связность и читабельность текста.

Карта ассоциативных связей между терминами строится на основе анализа собственных частот значимых терминов, а также частот их совместной встречаемости в предложениях текста. Ассоциативная семантическая сеть – это ориентированный граф, чьими вершинами служат значимые темы, выделенные в анализируемом тексте, а дугами – связи между ними. С каждой вершиной связаны вес (значимость) и частота упоминания темы, а с каждой дугой – вес (сила) связи и частота подкрепления связи в тексте.

Вес связи отражает условную вероятность упоминания первой темы совместно со второй. Вес связи от темы *A* к теме *B*, равный 100, означает, что при упоминании темы *A* упоминалась и тема *B*. Например, вес прямых связи от темы «*финансовая деятельность предприятия*» к темам «*финансы*», «*деятельность предприятия*» и «*предприятие*» будет равен 100, а вес обратных связей будут зависеть от собственной частоты встречаемости трех последних тем в тексте и может принимать значения не более 99.

Интерфейс класса CTopExtractor

ProcResult – функция выделения тем и построения реферата

Функция **ProcResult** извлекает темы из текста и формирует реферат документа с учетом ограничений, задаваемых параметрами:

- *nMaxTopCount* – максимальное количество тем, которое можно выделить в тексте;
- *nMaxSentenceCount* – максимальное количество предложений в реферате не должно превосходить значения этого параметра и *КоличестваПредложенийТекста*/*nPressingRate* + 1.

Результаты работы функции заносятся в структуры **STopicInfo**, **SSentenceInfo**.

```
int ProcResult( FX_HANDLE hResult, int nMaxTopCount = 10, int
   nMaxSentenceCount = 3, int nPressingRate = 1 );
```

GetTopicRelations – функция расчета веса связей

Функция **GetTopicRelations** вычисляет вес связи между двумя темами (от первой ко второй и от второй к первой). Вес равен нулю, если число предложений, в которых присутствуют обе темы, меньше *nMinTopicCooccurrence*.

```
static void GetTopicRelations(
const STopicInfo* pTopic1, // указатель на функции структур,
   содержащих данные о выделенных темах
const STopicInfo* pTopic2, // указатель на функции структур,
   содержащих данные о выделенных темах
double& fRelation12, // ссылка на переменную, содержащую вес связи
   между 1 и 2 темами
double& fRelation21, // ссылка на переменную, содержащую вес связи
   между 2 и 1 темами
int nMinTopicCooccurrence // ограничение на минимальный выводимый вес
   связи
);
```

Выделение тем текста из структуры STopicInfo

Функции **GetFirstTopic и GetNextTopic** возвращают из структуры **STopicInfo** указатели на первую и следующие темы текста. Если тем больше нет – значение *NULL*.

Выделение реферата текста из структуры SSentenceInfo

Функции **GetFirstSummarySentence** и **GetNextSummarySentence** возвращают из структуры **SSentenceInfo** указатели на первое и следующие предложения реферата. В отсутствие предложений – значение *NULL*.

Структура SSentenceInfo

Структура содержит функции, извлекающие (посредством обращения к результатам разбора ПО **RCO Fact Extractor**) сведения о предложениях исходного документа.

GetTextOffset

Функция возвращает количество символов от начала текста до начала предложения.

```
GetTextOffset(
   hResult // дескриптор результата анализа текста
)
```

GetTextLength

Функция возвращает количество символов от начала предложения до его конца.

```
GetTextLength(
   hResult // дескриптор результата анализа текста
)
```

GetText

Функция возвращает текст предложения, очищенный от элементов форматирования текста (html-тегов, переносов строки и т.п.).

```
GetText(
   hResult // дескриптор результата анализа текста
)
```

GetWeight

Функция возвращает вес предложения в реферате.

```
GetWeight()
```

GetIndex

Функция возвращает номер предложения в тексте.

```
GetIndex()
```

Структура STopicInfo

Структура содержит функции, извлекающие (посредством обращения к результатам разбора ПО **RCO Fact Extractor**) сведения о темах исходного документа.

GetF

Функция возвращает частоту встречаемости темы в тексте.

```
GetF()
```

GetWeight

Функция возвращает вес темы в тексте.

```
GetWeight()
```

IsDerivated

Возвращаемое значение сигнализирует о вложенности темы в другую: *true* – вложена. Подробнее – в разделе «***Атрибуты сущности***» документа «***Руководство разработчика RCO Fact eXtractor SDK***» (значение *FX_ENTITY_NAMETYPE_DERIVATED* атрибута `FX_ATTR_ENTITY_NAMETYPE`).

```
IsDerivated()
```

GetName

Функция возвращает имя темы в канонической форме.

```
GetName()
```

GetNameTemplate

Функция возвращает инвариант имени темы, в котором каждое слово стоит в канонической форме.

```
GetNameTemplate()
```

GetSemanticType

Функция возвращает семантический тип темы.

```
GetSemanticType()
```

Порядок работы RCO Top Extractor

Поскольку в работе **RCO Top Extractor** задействовано ПО **RCO Fact Extractor**, необходимым условием функционирования являются корректная настройка и инициализация последнего. Информация на эту тему содержится в документе «***Руководство разработчика RCO Fact eXtractor SDK***».

Если настройка выполнена, сценарий работы с классом **RCO Top Extractor** будет следующим:

1. Задание через набор типов `m_mTopicTypes` сущностей, которые могут рассматриваться в качества тем, и весовых коэффициентов. Кроме этих сущностей, будут выделены имена существительные, не отфильтрованные через тезаурус как общеупотребительные. В конструкторе класса задается базовый набор сущностей.

2. Вызов функции **`ProcResult`** для извлечения тем из текста и формирования реферата документа с учетом заданных ограничений.

3. Последовательный вызов функций **`GetFirstSummarySentence`**, **`GetNextSummarySentence`**, – для получения указателей на первое и следующие предложения реферата, **`GetFirstTopic`**, **`GetNextTopic`** – для получения указателей на первую и следующие темы документа.

4. При необходимости расчета веса связей между темами – вызов функции **`GetTopicRelations`**.

Пример приложения, использующего RCO Top Extractor

RCO Top Extractor поставляется вместе с исходным кодом тестового приложения **TopExtractorXML**. Программа при помощи **CTopExtractor** извлекает сведения для построения информационного портрета документа:

- реферат текста,
- отобранные темы,
- связи.

Отобранные данные представляются в виде xml-файла формата, совместимого с входным форматом ПО **ORA** (**Organizational Risk Analyzer**), осуществляющего визуализацию карты ассоциативной сети.

В качестве аргументов приложению **TopExtractorXML** при запуске необходимо сообщить путь к конфигурационному файлу библиотеки **RCO FX Ru**, путь к обрабатываемому тексту, имя выходного файла XML.

Основные стадии работы TopExtractorXML.cpp

Описание используемых объектов

- Структура **`top_param`** – для фильтрации тем, связей и построения рефератов:
  - Ограничения на характеристики реферата документа:
    - `m_nSentenceMaxCount` – максимальное число предложений;
    - `m_nSentenceMaxLength` – максимальная суммарная длина предложений, выраженная в количестве символов;
    - `m_fSentenceMaxRatio` – максимально допустимое соотношение объемов реферата и текста.
  - Ограничения на количество возвращаемых тем:
    - `m_nTopicMaxCount` – максимальное количество слов и словосочетаний;
    - `m_fTopicMinWeight` – минимальный относительный вес темы;
    - `m_nTopicMinFrequency` – минимальная частота встречаемости темы.
  - Ограничения на типы тем:
    - `m_nTopicMaxLength` – максимальная длина ключевой темы, выраженная в количестве слов;
    - `m_setTopicTypesCommon` – перечень типов извлекаемых ключевых тем (персона, организация, географическое наименование, термин и т.д.). Если не задан, извлекаются все типы;
    - `m_setTopicTypesAlways` – перечень типов ключевых тем, извлекаемых вне зависимости от прочих тематических ограничений;
    - `m_bTopicExtractDerivated` – определяет, выдавать или нет производные темы;
    - `m_nTopicExtractHomonyms` – флаг генерации всех вариантов имени сущности для слов с неразрешимой омонимией. Подробнее – см. описание идентификатора параметра `FX_PARAM_ENTITYNAME_HOMONYMS` в документе «***Руководство разработчика RCO Fact eXtractor SDK***».
  - Дополнительные ограничения:
    - `m_nTopicSentenceMaxCount` – максимальное количество предложений в реферате по теме;
    - `m_nMinTopicCooccurrence` – ограничение на минимальный выводимый вес связи.
- Структура **`t_sentence`** – для извлечения сведений о предложениях, содержащих термины:
  - `nIndex` – индекс предложения в тексте;
  - `sText` – текст предложения, очищенный от элементов форматирования;
  - `nOffset` – позиция предложения в исходном реферате;

  - `fWeight` – вес предложения в реферате.
- Структура **`t_topic`** – для отбора тем:
  - `sName` – имя темы в канонической форме;
  - `sNameTemplate` – инвариант имени темы, где каждое слово стоит в канонической форме;
  - `sSemanticType` – семантический тип темы;
  - `nFrequency` – частота встречаемости;
  - `fWeight` – вес темы в тексте;
  - `bDerivated` – значение *true* свидетельствует о вложенности в другую тему;
  - `vTopicSentences` – информация о предложениях, в которых появлялась данная тема (может отсутствовать);
  - `vInSentences` – индексы предложений, в которых появлялась данная тема (всегда заполняется).
- Структура **`t_relation`** – для отбора связей:
  - `nTopic1` – индекс первой темы;
  - `nTopic2` – индекс второй темы;
  - `fWeight12` – вес связи между первой и второй темами;
  - `fWeight21` – вес связи между второй и первой темами.

Описание используемых функций

В число основных функций приложения входят:

- **`CreateContext`** – инициализирует библиотеку **RCO FX Ru**;

```
int CreateContext(
const char* pszConfigFilePath, // указатель на конфигурационный файл
FX_HANDLE& hContext // ссылка на переменную, получающую дескриптор
   анализатора текста
);
```

- **`DestroyContext`** – освобождает ресурсы библиотеки **RCO FX Ru**;

```
void DestroyContext(
FX_HANDLE& hContext // ссылка на переменную, получающую дескриптор
   анализатора текста
);
```

- **`ExtractTopics`** – извлекает темы, формирует рефераты, рассчитывает силу связей между темами;

```
int ExtractTopics(
const top_param* pParam, // указатель на параметры структуры
   top_param
FX_HANDLE hResult, // дескриптор результата анализа текста
int nTextLength, // длина текста в символах
vector<t_topic>& vTopics, // ссылка на параметры структуры t_topic
vector<t_sentence>& vSentences, // ссылка на параметры структуры
   t_sentence
vector<t_relation>& vRelations // ссылка на параметры структуры
   t_relation
);
```

- **`MakeXMLResult`** – представляет результаты в формате XML.

```
int MakeXMLResult(
const vector<t_topic>& vTopics, // ссылка на параметры структуры
   t_topic
const vector<t_sentence>& vSentences, // ссылка на параметры
   структуры t_sentence
const vector<t_relation>& vRelations, // ссылка на параметры
   структуры t_relation
string& sXmlResult // ссылка на строку с результатами анализа
);
```

Об инициализации библиотеки и освобождении ее ресурсов рассказывает документ «***Руководство разработчика RCO Fact eXtractor SDK***».

Кроме того, предусмотрены вспомогательные функции:

- **`CheckSemanticType`** – фильтрация по признаку соответствия темы заданному типу;
- преобразования кодировок и замены специальных символов XML:
  - **`ConvertToUnicode`** – из ANSI в Unicode;
  - **`ConvertToUtf8`** – из ANSI в UTF-8;
  - **`ConvertFromUnicode`** – из Unicode в ANSI;
  - **`XMLEncode`** – замена специальных символов.

Последовательность операций в приложении TopExtractorXML.cpp

1. Открытие файла и считывание текста из него.
2. Инициализация библиотеки **RCO FX Ru**, обработка текста с ее помощью.
3. Выделение тем и атрибутов, составление реферата текста функцией **`ProcResult`**.
4. Разбор результатов функциями **`GetFirstSummarySentence`**, **`GetNextSummarySentence`**, **`GetFirstTopic`**, **`GetNextTopic`**.
5. Вычисление веса связей между темами посредством функции **`GetTopicRelations`**.
6. Вывод результатов в xml-файл.
7. Освобождение ресурсов библиотеки.