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 пользователя

RCO Deduplicator – программа для выявления дубликатов документов

Версия 1.1 (Microsoft Windows)

Москва, 2017

В содержание данного документа могут быть внесены изменения без предварительного уведомления. Названия организаций, имена и даты, используемые в качестве примеров, являются вымышленными, если не оговорено обратное.

© ООО «ЭР СИ О», 2013-2017. Все права защищены.

ЭР СИ О, Russian Context Optimizer, RCO являются охраняемыми товарными знаками.

ООО «ЭР СИ О» может являться правообладателем патентов и заявок, поданных на получение патента, товарных знаков и объектов авторского права, которые имеют отношение к содержанию данного документа.

Предоставление вам данного документа не означает передачи какой-либо лицензии на использование данных патентов, товарных знаков и объектов авторского права, за исключением использования, явно оговоренного в лицензионном соглашении ООО «ЭР СИ О».

Все другие названия юридических лиц и изделий являются охраняемыми товарными знаками или товарными знаками, принадлежащими их владельцам.

Содержание

Copyright © ЭР СИ О, 2017

Введение

Выявление дублей загружаемого документа среди имеющихся в базе данных (БД) необходимо для очистки результатов поиска от лишней информации и, следовательно, упрощения аналитической работы с базой.

Процедура избавления от дубликатов двухэтапная. Первый этап – выявление важных для обнаружения дубликатов характеристик поступившего в систему документа. Второй – поиск дубликатов.

Внимание! Использование библиотеки **RCO Deduplicator** возможно лишь при наличии работающей версии программы **RCO Fact Extractor**.

Характеристики документа и структуры для их хранения

Библиотека выделяет три набора характеристик текста документа, сохраняемых в структуру **`t_portrait`**, элемент структуры **`t_document`**.

Список характеристик:

1. Контрольные суммы *vSentenceCRC32*, рассчитанные для ***p1*** самых длинных предложений;

2. Число слов в тексте *nWordCount*, исключая короткие предложения (длиной менее ***p2*** слов);

3. Контрольные суммы *vFrequentWordCRC32*, рассчитанные для ***p3*** самых частых слов, кроме стоп-слов и слов из коротких предложений (длиной менее ***p2*** слов). Под словами здесь понимаются одиночные слова и устойчивые словосочетания длиной до 3 слов.

Значения ***p1***, ***p2***, ***p3*** определяются следующими параметрами:

p1 – ***nMaxSentenceCount*;**

p2 – ***nMinSentenceSize*;**

p3 – ***nMaxFrequentWordCount*.**

Замечание. В тестовом приложении **DeDuplicatorDllTest.cpp**, входящем в комплект поставки для проверки и демонстрации работы библиотеки, ***p1***, ***p2***, ***p3*** определяются через параметры ***nMaxSentenceCount***, ***nMinSentenceSize***, ***nMaxFrequentWordCount***. При написании иных приложений, задействующих программный интерфейс библиотеки, необходимо использовать те же параметры (функция **`InitializeFeatureExtractionInstance`**).

Структуру **`t_document`**, помимо структуры **`t_portrait`**, составляют такие элементы, как:

- `sDocId` – идентификатор документа;
- `nDuplicateId` – служебный параметр, используемый при поиске дубликатов и всегда равный −1.

Алгоритм поиска дубликатов документа

Дубликаты выявляются с использованием следующих условий:

1. Хотя бы одна контрольная сумма предложений совпадает (необходимое условие);

Copyright © ЭР СИ О, 2017

2. Разница в числе слов документов не превышает заданного значения или отношение длин документов не превосходит определенного значения (необходимое условие);
3. Все контрольные суммы предложений совпадают (достаточное условие);
4. Контрольные суммы частых слов совпадают (достаточное условие).

Руководствуясь приведенными критериями, можно создать приложение для отбора дубликатов, описание работы которого приведено в разделе «Использование поставляемого приложения».

Чтобы не перебирать последовательно все документы базы, строится отображение ключевых признаков документов на их индексы. Наиболее существенные признаки документа – контрольные суммы самых длинных предложений. Из всего массива отбираются документы с хотя бы одним общим ключевым признаком (т.е. одно общее с новым документом предложение, тождественное по crc32). Из этого набора исключаются документы, явно отличающиеся по размеру от нового. Отобранные документы последовательно сравниваются. Два документа считаются дубликатами, если все контрольные суммы их предложений совпадают или контрольные суммы k из n слов одинаковы (например, 2 из 5 самых частых слов).

Замечание. В тестовом приложении максимальная разница в числе слов документов определяется значением параметра ***nDocLengthDifference***, максимальное отношение длин документов – ***fDocLengthRatio***, минимальное количество одинаковых наиболее частых слов в документах – ***nMatchWordCount***. При написании иных приложений, задействующих программный интерфейс библиотеки, необходимо использовать те же параметры (функция `InitializeDocumentComparingInstance`).

Copyright © ЭР СИ О, 2017

Применение библиотеки

Тестовое приложение **DeDuplicatorDllTest.cpp**, с одной стороны, показывает порядок работы с библиотекой, с другой стороны, реализацию алгоритма поиска дубликатов, приведенного в разделе «Алгоритм поиска дубликатов документа».

Использование библиотеки в приложениях

Взаимодействие с библиотекой осуществляется через описываемый в данном документе программный интерфейс по следующему алгоритму:

1. Инициализация библиотеки функцией **`GetInstance`**;
2. Инициализация определения характеристик документа (подсчета контрольных сумм и количества слов) **`InitializeFeatureExtractionInstance`**;
3. Извлечение характеристик документа функцией **`ExtractFeatures`** и сохранение их в структуру **`t_document`**;
4. Освобождение ресурсов функцией **`FreeInstance`**;
5. Повторная инициализация библиотеки функцией **`GetInstance`**;
6. Инициализация сравнения характеристик документов (контрольных сумм и количества слов, извлекаемых из структуры **`t_document`**) при помощи функции **`InitializeDocumentComparingInstance`**;
7. Поиск дубликатов документов функцией **`DuplicateSearch`**;
8. Доступ к результатам работы **`DuplicateSearch`** посредством функций **`GetDuplicateInfo`** (получение информации о выявленных дубликатах) и **`GetDocumentPortrait`** (формирование портретов дублирующихся документов).
9. Освобождение ресурсов функцией **`FreeInstance`**.

Использование поставляемого приложения

Приложение получает на вход пути к каталогу с документами (**Text**) и к директории (**DuplicatesResult**), в которой сохраняются файлы с информацией о найденных в ходе анализа множествах дубликатов и файл (**index_duplicates.htm**) со сводными данными о результатах выявления дубликатов (количество и порядковые номера множеств дубликатов, пути к составляющим эти множества документам).

```
DeDuplicatorTest.exe .\Text\ .\DuplicatesResult\ index_duplicates.htm
```

Далее **DeDuplicatorDllTest.cpp** инициализирует библиотеку **FX** (составная часть пакета **RCO Fact Extractor**, специальным образом настроенная на ускоренную обработку текста) конфигурационным файлом **config.xml**, получает контекст **FX** и передает его функции **`GetInstance`**. Дальнейшая обработка происходит в соответствии с последовательностью, описанной в разделе «Использование библиотеки в приложениях». Те же операции проводятся в отношении всех документов из входного каталога. В результате анализа дубликаты формируют множества, описание которых сохраняется в HTML-файлах, содержащих в названии номер множества (0, 1, 2 и т.п.), в формате:

```
[id документа]\n[crc32 предложений]… \n[число слов документа]\n[число
   ключевых слов] (\n[crc32 слов]…)
```

Общие сведения об обнаруженных дубликатах записываются в отдельный файл (в данном случае **index_duplicates.htm**) в следующем в формате:

Copyright © ЭР СИ О, 2017

```
[количество проверенных документов]\n[количество обнаруженных
   множеств дубликатов] (\n[номер множества [количество дубликатов во
   множестве] ссылка на портрет документа: идентификаторы
   документов]…)
```

Copyright © ЭР СИ О, 2017

Интерфейс библиотеки

Функции для работы с ресурсами

GetInstance

Функция создает заданный тип дескриптора анализатора текста.

```
void GetInstance(
   void **phHandle, // дескриптор заданного типа
   handle_type type // тип дескриптора
);
```

Параметры

phHandle
[вых] Указатель на дескриптор анализатора текста заданного типа.

type
[вх] Тип дескриптора – элемент перечисления **handle_type**. Предусмотрены два вида дескрипторов:

- *FEATURE_EXTRACTION_HANDLE* – дескриптор извлечения характеристик текста;
- *DOCUMENT_COMPARING_HANDLE* – дескриптор сравнения документов.

FreeInstance

Функция освобождает ресурсы библиотеки.

```
void FreeInstance(
   void *hHandle, // дескриптор заданного типа
   handle_type type // тип дескриптора
);
```

Параметры

hDeDuplicator
[вх] Указатель на дескриптор анализатора текста заданного типа.

type
[вх] Тип дескриптора – элемент перечисления **handle_type** (характеристика дескрипторов дана в описании функции **GetInstance**).

Copyright © ЭР СИ О, 2017

Функции для анализа текста и запроса результата

InitializeFeatureExtractionInstance

Функция инициализирует определение необходимых для выявления дубликатов характеристик документа.

```
void InitializeFeatureExtractionInstance(
   FEATURE_EXTRACTION_HANDLE hExtractor, // дескриптор извлечения
   характеристик текста
   const SFeatureExtractionParameters &tParameters // параметры,
   определяющие условия выделения характеристик документа
);
```

Параметры

hExtractor

[вх] Дескриптор извлечения характеристик текста.

tParameters

[вх] Ссылка на значения параметров, определяющих условия выделения характеристик документа: ***nMaxSentenceCount***, ***nMinSentenceSize***, ***nMaxFrequentWordCount***.

ExtractFeatures

Функция извлекает характеристики текста, разобранного функцией **`FXProcessText`** (вызывается в приложении отдельно) библиотеки **RCO FX Ru**, и создает дескриптор, через который впоследствии можно получить доступ к характеристикам текста.

```
void ExtractFeatures(
   FEATURE_EXTRACTION_HANDLE hExtractor, // дескриптор извлечения
   характеристик текста
   FX_HANDLE hResult, // результат анализа текста
   const char *pszDocName // имя анализируемого документа
);
```

Параметры

hExtractor

[вх] Дескриптор извлечения характеристик текста.

hResult

[вх] Переменная, содержащая результат разбора текста функцией **`FXProcessText`** библиотеки **RCO FX Ru**.

pszDocName

[вх] Указатель на имя анализируемого документа.

Copyright © ЭР СИ О, 2017

Функции для сравнения характеристик документов и запроса результата

InitializeDocumentComparingInstance

Функция инициализирует сравнение характеристик документов и создает соответствующий дескриптор.

```
void InitializeDocumentComparingInstance(
   DOCUMENT_COMPARING_HANDLE hDocComparer, // дескриптор сравнения
   характеристик документов
   const SDocumentComparingParameters &tParameters // параметры,
   определяющие условия сравнения характеристик документа
);
```

Параметры

hDocComparer
[вх] Дескриптор сравнения характеристик документов.

tParameters
[вх] Ссылка на значения параметров, определяющих условия сравнения характеристик документа: ***nDocLengthDifference***, ***fDocLengthRatio***, ***nMatchWordCount***.

GetDocumentPortrait

Функция записывает в структуру **t_document** характеристики документа, формирует его портрет.

```
void GetDocumentPortrait(
   FEATURE_EXTRACTION_HANDLE hExtractor, // дескриптор извлечения
   характеристик текста
   t_document **pptDocument // массив с характеристиками документов
);
```

Параметры

hExtractor
[вх] Дескриптор извлечения характеристик текста.

pptDocument
[вых] Указатель на первый элемент массива с характеристиками документов.

DuplicateSearch

На основании характеристик обработанных документов функция выявляет дубликаты.

```
void DuplicateSearch(
   DOCUMENT_COMPARING_HANDLE hDocComparer, // дескриптор сравнения
   характеристик документов
   std::vector<t_document> *pvDocuments // характеристики
   сравниваемых документов
);
```

hDocComparer
[вх] Дескриптор сравнения характеристик документов.

Copyright © ЭР СИ О, 2017

pvDocuments
[вх] Указатель на вектор характеристик сравниваемых документов.

GetDuplicateInfo

Функция записывает в массив информацию об обнаруженных дубликатах.

```
void GetDuplicateInfo(
   DOCUMENT_COMPARING_HANDLE hDocComparer, // дескриптор сравнения
   характеристик документов
   DUPLICATE_INFO_CONTAINER **ppvDuplicates // массив с информацией о
   дубликатах
);
```

hDocComparer
[вх] Дескриптор сравнения характеристик документов.

ppvDuplicates
[вых] Указатель на первый элемент массива с информацией о дубликатах.

Copyright © ЭР СИ О, 2017