САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

КАФЕДРА ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

Халиуллина Лия Рауфовна

Выпускная квалификационная работа бакалавра

Поиск дубликатов среди документов

Направление 01.03.02

Прикладная математика и информатика

Научный руководитель,

ст. преподаватель,

Малинина М. А.

Санкт-Петербург

2018

Содержание

Введение

Современное развитие информационных технологий и сети Интернет предоставило широким кругам пользователей легкий и быстрый доступ к огромным массивам самой разнообразной информации. Это вызвало бурный рост количества дублированного и заимствованного материала, ведь теперь для создания нового совсем не обязательно действительно что-то придумывать. В связи с этим крайне остро встал вопрос нахождения различного рода заимствований в научных статьях и прочих работах.

Проблема выявления оригинальности текста является одной из наиболее сложных и трудоемких задач анализа данных. Чаще всего она возникает в процессе установления нарушений авторских прав (проблема плагиата), но существуют и другие области, для которых данный вопрос актуален: кластеризация документов по содержанию, удаление избыточной информации в архивах поисковых систем для улучшения их качества, фильтрация поискового спама, и ряд других.

Однако колоссальный объем обрабатываемой информации является основной загвоздкой решения задачи поиска дубликатов. Попарное прямое сравнение текстов документов займет слишком много времени и потребует значительных вычислений, поэтому постепенно создавались различные методы, позволяющие снизить алгоритмическую сложность.

Различают понятия четких и нечетких дубликатов документов.

Четкий дубликат – точная копия текста. Также четким дубликатом будет считаться документ, полностью содержащий в себе другой документ.

Нечеткий дубликат – частично измененный текст, содержание которого отличается незначительно, т. е. может содержать в себе отрывки из другого документа, или не отличается вовсе, т. е. отличается от источника лишь заменой некоторых слов на их синонимы.

Постановка задачи

Целью данной работы является исследование существующих методов поиска дубликатов среди документов, а также представление некоторой модификации для улучшения показателей эффективности. Для достижения этой цели были поставлены следующие задачи:

1. Изложить наиболее популярные и эффективные с вычислительной точки зрения алгоритмы решения проблемы поиска дубликатов.
2. Предоставить оценку эффективности рассмотренных методов.
3. Описать модифицированный вариант метода и проанализировать полученные результаты.

Глава 1. Обзор методов поиска дубликатов документов

В решении задачи поиска дубликатов документа различают следующие основные шаги:

1. Представить документ в виде множества признаков.
2. Составить образ документа, выбрав некое подмножество признаков.
3. Определить степень сходства на образах документов.

Затем, в зависимости от поставленной задачи, могут выполняться следующие шаги (возможны комбинации):

4. Вычислить кластеры похожих документов.
5. Соединить кластеры похожих документов из разных коллекций.
6. Удалить найденные дубликаты документа, и другие.

Первый шаг осуществляется после снятия разметки (например, HTML). Представленные в виде линейных последовательностей символов документы преобразовываются во множества. Выделяют два основных подхода: синтаксический (выбираются последовательности символов, слов или предложений) и лексический (выбираются представительные языковые единицы).

Во втором шаге из получившегося множества синтаксических или лексических признаков выбирается некое подмножество, дающее краткое описание документа, называемое образом. В синтаксических подходах обычно используют схемы рандомизации [1-3], а в лексических – различные методы выбора существенных слов [4-6].

На третьем шаге вычисляется степень сходства документов, которая выражается некоторой числовой мерой. Данная мера принимает значения в интервале [0, 1] и определяет отношение сходства в паре документов. При

превышении мерой установленного порога документы будут считаться (не)четкими дубликатами.

Дальнейшие шаги зависят от поставленной задачи.

Рассмотрим перечисленные шаги в наиболее популярных методах обоих подходов, указанных в первом шаге.

1.1. Синтаксические методы

1.1.1. Алгоритмы sif и Koala

Первыми исследователями решений задачи поиска дубликатов являются Udi Manber [7] и Nevin Heintze [8].

U. Manber в своей работе 1994 года [7] представил утилиту sif для поиска похожих файлов в крупных файловых системах, позволяющую обнаружить даже 25% совпадение содержаний. Однако метод, используемый в sif, не учитывает смысл текста файла. Таким образом файлы, содержащие похожую информацию, записанную разными словами, не будут считаться дубликатами.

В данном методе для построения выборки используются последовательности соседних букв, подстроки. В первом варианте метода подстроки определяются «якорями» - ограниченным набором ключевых слов или цепочек букв. Файл сканируется на наличие «якорей», затем рассматриваются следующие после «якоря» фиксированное количество символов, образующих подстроки, и считаются их дактилограммы (с помощью метода Рабина-Карпа [9]). Аналогичное происходит и с другими файлами, среди которых производится поиск дубликатов. Одинаковые значения дактилограмм будут получены со всех файлов, содержащих какую-либо определенную подстроку. Следовательно, при подсчете общих подстрок и происходит сравнение двух конкретных файлов. Однако «якори», оптимальные для какой-то определенной тематики файлов, могут совершенно не подходить другой тематике, поэтому список «якорей» нужно определить

заранее и с учетом темы сканируемого файла. Так что для простоты и облегчения процесса предлагается второй вариант метода, где «якори», по сути, являются универсальными и выбираются абсолютно случайно. В этой вариации считаются дактилограммы всех возможных подстрок фиксированной длины, и все вместе они составляют дактилограмму файла. Затем производится сравнение данного файла со множеством других, чьи дактилограммы также посчитаны. Мерой сходства двух файлов в обоих вариантах является отношение общих подстрок к размеру файла.

N. Heintze в своей работе 1996 года [8] представил систему Koala для поиска дубликатов документов, использующую схожий принцип, что и sif. Вначале предлагается считать дактилограмму документа по примеру дактилограммы файла U. Manber. Однако такой подход не практичен из-за больших объемов обрабатываемого текста (получаемые дактилограммы будут слишком длинными), хотя и является полезной мерой сходства документов. Поэтому рекомендуется выбирать фиксированное количество подстрок вне зависимости от размера рассматриваемого документа. Подстроки отбираются не случайным образом, а основываясь на частотной мере первых пяти букв, то есть выделяются те, чьи первые пять букв встречаются в тексте нечасто. Таким образом Koala фокусируется на выделении даже 3-5% совпадений текста. Помимо этого, отмечается неизбежное возникновение ошибок при получении текста документа. Ошибки могут заключаться в пунктуации, пробелах и не входящих в алфавит знаках, также есть вероятность возникновения сложностей с гласными и выделением регистра букв. Для решения этой проблемы автором предлагается избавиться от всех вышеперечисленных символов, оставив лишь согласные, и перевести все буквы в строчные. Таким образом применяемый системой подход основан скорее на последовательности символов исходного документа, чем на его подстроках.

Данные методы по сути считаются устаревшими и приведены в этой работе как примеры одних из самых первых методов поиска дубликатов.

1.1.2. Метод «шинглов»

В 1997 году Andrei Broder et al. [1, 10] усовершенствовали идею, заложенную в sif и Koala. Так появился метод «шинглов» – один из наиболее популярных современных синтаксических методов.

Документ, очищенный от разметки, знаков препинания и пробелов, представляется в виде всевозможных последовательностей фиксированной длины, состоящей из соседних слов (символов). Такие последовательности называются «шинглами» (от англ. shingle – чешуйка). Два документа считаются похожими, если множества их шинглов существенно пересекаются.

Поскольку число шинглов для каждого документа было примерно равно длине этого документа в словах, изначально методом отбирались только те шинглы, чьи дактилограммы (вычисленные по методу Рабина-Карпа [9]) делились без остатка на некоторое число m. Однако выборка для небольших документов могла оказаться короткой или даже пустой. Также использовался подход отбора фиксированного количества s шинглов с наименьшими значениями дактилограмм (если шинглов было меньше s, оставлялись все).

В дальнейшем A. Broder et al. [2] и Dennis Fetterly et al. [11] усовершенствовали этот метод. Для каждого шингла вычисляются 84 дактилограммы по алгоритму Рабина-Карпа [9] с помощью взаимно-однозначных и независимых («min-wise independent») функций. В результате документ оказывается представлен в виде 84 шинглов, имеющих минимальное значение вычисленных функций. Эти 84 шингла разбиваются на 6 групп по 14 шинглов в каждой, группы эти называются супершинглами. Исследования показали, что два документа оказывались нечеткими дубликатами при совпадении минимум 2 супершинглов. Таким образом каждый документ лучше представить различными попарными сочетаниями из 6 супершинглов – мегашинглами, которых всего 15 штук. Два документа сходны по содержанию, если у них совпадает хотя бы один мегашингл.

1.1.3. Long Sent

Этот метод разбивает документ на предложения, которые после сортируются по убыванию количества слов (при равном количестве сортируются в алфавитном порядке). Далее выделяются 2 самых длинных предложения и сцепляются в строку. Контрольная сумма CRC32 этой строки принимается за сигнатуру документа. Два документа считаются похожими, если у них совпадают сигнатуры.

1.2. Лексические методы

1.2.1. Методы на основе меры TF-IDF

TF-IDF – статистическая мера, используемая для оценки важности термина в контексте документа, являющегося частью коллекции документов. Вес некоторого слова пропорционален количеству употребления этого слова в документе и обратно пропорционален частоте употребления слова в других документах коллекции [12].

В качестве терминов используют конкретные слова в тексте документа длиной не менее 4 символов и не содержащие никаких цифр. Большинство слов длиной менее 4 символов являются стоп-словами (например, различные предлоги), поэтому их отфильтровывание улучшает качество работы метода.

TF (*term frequency* – частота слова) измеряет частоту появления слова в тексте документа. Изначально TF вычислялся как количество вхождений слова в документ. Однако, так как документы могут различаться объемами, и число появления слова в большом документе может оказаться гораздо больше числа появления этого же слова в меньшем по размеру документе, TF часто делят на длину документа (то есть общее число слов в документе), нормализуя таким образом величину. Так оценивается важность каждого слова в пределах отдельного документа.

$$TF_{t,d} = \frac{n_t}{\sum_k n_k}$$

где n_t – число появлений слова t в документе d, $\sum_k n_k$ – общее число слов в документе d.

Идея метода состоит в построении частотного словаря документа, где слова упорядочены по убыванию TF. После этого k слов с наибольшим значением частот сортируются в алфавитном порядке и сцепляются в строку. В качестве сигнатуры документа вычисляется контрольная сумма CRC32 полученной строки. Два документа считаются похожими, если у них совпадают сигнатуры.

Эта идея сохраняется при построении словаря коллекции, где каждому слову t ставится в соответствие его «вес» W_t [13]. Далее также выбирают k слов с наибольшим значением веса и сцепляют их в алфавитном порядке. Контрольная сумма CRC32 этой строки берется за сигнатуру документа. При совпадении сигнатур документы считаются похожими.

$$W_t = TF_{t,d} * IDF_t \quad (1)$$

IDF (*inverse document frequency* — обратная частота документа) — инверсия частоты, с которой некоторое слово встречается в документах коллекции.

$$IDF_t = \log \frac{N}{df_t}$$

где N – общее число документов в коллекции, df_t – количество документов, содержащих слово t.

В результате большой вес получат слова с высокой частотой в пределах конкретного документа и с низкой частотой употреблений в других документах.

Однако длинные документы при описанных выше формулах недооцениваются, и для решения этой проблемы используют дополнительную нормализацию:

$$norm\, TF_{t,d} = 0.5 + 0.5\frac{tf_{t,d}}{tf_{max}(d)}$$

где $tf_{t,d}$ – частота слова t в документе d, а $tf_{max}(d) = \max\limits_{\tau \in d} tf_{\tau,d}$ – максимальная частота среди всех слов в документе d.

Используются и другие варианты вычисления веса слова. Одной из наиболее популярных является формула Okapi BM25 [14], в которой частота слова и обратная частота документа имеют следующий вид:

$$TF_{t,d} = \frac{tf_{t,d}}{k_1\left((1-b)+b\frac{dl}{avg\_dl}\right)+tf_{t,d}} \quad (2)$$

где $tf_{t,d}$ – частота слова t в документе d, dl – длина документа (общее число слов в нем), avg_dl – средняя длина документа в коллекции, k_1 и b – свободные коэффициенты, обычно вычисления производят при $k_1 = 2$ и $b = 0.75$.

$$IDF_t = \log\frac{N - df_t + 0.5}{df_t + 0.5} \quad (3)$$

где N – общее число документов в коллекции, df_t – количество документов, содержащих слово t.

У такого способа вычисления IDF_t есть своя особенность: если некоторое слово появляется в более половины документах коллекции, его IDF_t принимает отрицательно значение. То есть часто встречающиеся слова портят оценку документа, поэтому обычно при вычислениях либо все отрицательные слагаемые отбрасываются, либо устанавливают некоторую нижнюю границу ε для IDF_t: если $IDF_t < \varepsilon$, считаем $IDF_t = \varepsilon$.

1.2.2. Heavy Sent

Идея метода похожа на Long Sent: документ делится на предложения и 2 из них сцепляют в строку в алфавитном порядке. Однако выбор осуществляется несколько иначе: считают вес каждого предложения, являющий собой сумму весов входящих в него слов, и выбор падает на два самых «тяжелых» предложения. Контрольная сумма CRC32 полученной после сцепления строки принимается за сигнатуру документа. Два документа считаются похожими, если у них совпадают сигнатуры.

1.2.3. I-Match

В 2002 году Abdur Chowdhury et al. предложили сигнатурный подход, основанный на лексических принципах [4].

В данном подходе для представления содержания документа вычисляется дактилограмма I-Match. Для этого сначала для исходной коллекции документов строится словарь L, который состоит из слов со средними значениями IDF, так как такие слова обычно позволяют достигнуть более точных результатов при поиске нечетких дубликатов. То есть слова с большими и маленькими значениями IDF не берутся в расчет.

Далее для каждого документа создается множество U различных слов, входящих в него, и рассматривается пересечение U и словаря L. Если размер этого пересечения больше некоторого определяемого экспериментально минимального порога, то список слов в пересечении упорядочивается. Затем вычисляется I-Match сигнатура (hash-функция SHA1). Два документа считаются похожими, если у них совпадают I-Match сигнатуры (имеет место коллизия hash-кодов).

Однако ее основным недостатком является неустойчивость к небольшим изменениям текста. Для решения этой проблемы в 2004 году алгоритм был модифицирован авторами [5], добавив возможность различных перемешиваний изначального словаря.

На основе словаря L строятся дополнительные словари L_1-L_k путем случайного удаления фиксированного количества p слов, составляющих примерно 30% от объема L. Экспериментальным путем были выяснены оптимальные значения параметров: k = 10 и p = 0.33. Таким образом для каждого документа считаются (k+1) сигнатуры I-Match тем же способом, и два документа считаются похожими, если у них совпадет хотя бы одна координата векторов, составленных из I-Match сигнатур. Такой подход похож на метод супершинглов (1.1.2), в котором ищут совпадения хотя бы одного супершингла.

1.2.4. Метод «описательных» слов

Еще один лексический метод, предложенный в 2002 году Сергеем Ильинским, Максимом Кузьминым, Александром Мелковым и Ильей Сегаловичем [6].

Сначала определенным способом выделяется набор из N слов, называемый «описательным набором» (число N определяется экспериментальным путем). Для каждого слова фиксируется пороговая частота b_i, и для каждого документа вычисляется вектор, в котором i-ый компонент равен 1, если значение относительной частоты i-го слова из «описательного набора» больше пороговой частоты, и равен 0 в ином случае. Этот двоичный вектор становится сигнатурой документа. Два документа считаются похожими, если их сигнатуры одинаковы.

Слова, входящие в «описательный набор», выбираются исходя из следующих критериев:

1. Набор слов должен покрывать максимально возможное количество документов.
2. Количество слов в наборе должно быть минимальным.
3. «Качество» слова должно быть наивысшим.

Под «качеством» слова подразумевается относительная стабильность соответствующего компонента вектора к малым изменениям документа. То есть для «хорошего» слова вероятность перехода через пороговое значение минимально для малых изменений текста.

Пороговое значение b_i выбирается как относительная частота с наилучшей точностью. Точность же тем выше, чем меньше встречаемость слова в дельта-окрестности данного значения частоты. Подобный подход обеспечивает почти всем документам ненулевые векторы сигнатур.

Сам же «описательный набор» определяется несколькими повторениями 2 этапов:

1. Максимизируется покрытие документов в индексе при фиксированной (ограниченной снизу) точности.
2. Максимизируется точность при фиксированном покрытии.

После каждой итерации самые «плохие» слова отбрасываются. В результате метод позволяет сократить сотни тысяч слов до 3-5 тысяч.

Глава 2. Оценка эффективности методов поиска дубликатов документов

Работу любого алгоритма имеет смысл проверить на качество. Стандартный подход оценки эффективности текстового поиска заключается в вычислении метрик, которые в большинстве своем используют понятие релевантности. Документ считается релевантным, если он соответствует сформулированной информационной потребности [13]. Документ релевантен не просто потому, что в нем содержатся слова из запроса, а также потому, что его смысловое содержание соответствует запросу.

Таблица 1. Виды документов при текстовом поиске.

		Релевантность	
		Положительная	Отрицательная
Оценка, данная методом	Положительная	TP	FP
	Отрицательная	FN	TN

Таким образом, любой метод делит документы рассматриваемой коллекции на 4 группы, принимая следующие решения:

- TP – истинно-положительное (true positive), то есть документы, которые заявлены методом как релевантные и являющиеся таковыми;
- FP – ложно-положительное (false positive), то есть документы, которые заявлены методом как релевантные, но не являющиеся таковыми;
- FN – ложно-отрицательное (false negative), то есть документы, которые являются релевантными, но не были найдены методом;

- TN – истинно отрицательное (true negative), то есть документы, которые заявлены методом как нерелевантные и являющиеся таковыми.

2.1. Основные метрики

Наиболее популярными метриками оценки эффективности текстового поиска являются точность (precision) и полнота (recall). [13]

Точность – способность метода находить документы, действительно являющиеся дубликатами. Вычисляется как отношение количества найденных релевантных документов ко всем найденным документам:

$$Precision = P = \frac{TP}{TP + FP}$$

Полнота – способность метода находить дубликаты документов без учета количества ошибочно определенных дубликатов. Вычисляется как отношение количества найденных релевантных документов ко всем релевантным документам:

$$Recall = R = \frac{TP}{TP + FN}$$

Естественно стремление повысить значения и точности, и полноты, но на практике максимальные точность и полнота недостижимы одновременно. Поэтому часто используют другую метрику, F-меру (F measure), объединяющую в себе значения полноты и точности.

$$F\ measure = F = \frac{1}{\alpha\frac{1}{P} + (1 - \alpha)\frac{1}{R}} = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

где $\beta^2 = \frac{1-\alpha}{\alpha}$, $\alpha \in [0, 1]$, и, соответственно, $\beta^2 \in [0, \infty)$.

Используя эту формулу, мы можем придать различный вес точности и полноте. Значения $0 < \beta < 1$ отдают приоритет точности, а значения $\beta > 1$

ставят в приоритет полноту. Выделяют стандартную F-меру, также обозначаемую как F_1, которая подразумевает одинаковый вес для точности и полноты и достигается при $\alpha = 1/2$ или $\beta = 1$:

$$F_1 = \frac{2PR}{P+R} \quad (4)$$

Точность, полнота и F-мера принимают значения [0, 1], их также можно обозначить через проценты [0, 100].

2.2. Оценка эффективности рассмотренных методов

В таблице 2 приводится сравнение эффективности методов из Главы 1 при обработке коллекции веб-сайтов РОМИП (~ 500000 документов), данные представлены в соответствии с [15].

Эталоном 100%-й полноты для этой коллекции взято объединение дубликатов, найденных всеми алгоритмами. Эталоном 100%-й точности считались результаты, полученные с помощью функции Perl String::Similarity с порогом сходства 80%.

Перед обработкой над каждым документом был проведен необходимый препроцессинг — очистка от html-тегов, удаление стоп-слов (служебной лексики и коротких слов длиной 3 и менее букв), символов новой строки, лишних пробелов, специальных знаков и т.п.

Таблица 2. Эффективность рассмотренных в Главе 1 методов (результаты упорядочены по убыванию F-меры)

Название метода	Полнота	Точность	F-мера
Long Sent	0.84	0.80	0.82
TF (k = 6)	0.60	0.94	0.73
Heavy Sent	0.62	0.86	0.72
TF-IDF (k = 6)	0.54	0.96	0.69
Модификация	0.50	0.97	0.66

I-Match (k = 10)			
Метод «описательных» слов	0.44	0.77	0.56
Мегашинглы	0.36	0.91	0.51
MD5	0.23	1.00	0.38

Дополнительно для сравнения в таблице 2 указаны результаты работы метода простого хеширования всего документа функцией MD5. Подобный подход ориентирован на поиск четких дубликатов, что подтверждается 100% точностью, которую показал данный метод. Его полнота же, очевидно, довольно низкая.

Метод Long Sent лидирует по всем метрикам оценки поиска дубликатов, однако его легко обмануть: достаточно внести изменения в 2 самых длинных предложения текста.

Так как мера TF-IDF устойчива к малым изменениям текста документа, методы, использующие её, показали приемлемую точность, лишь немного не дотянув до «чемпиона» по точности модифицированного метода I-Match. Последний, в свою очередь, имеет высокую вычислительную эффективность, но обладает не лучшей полнотой, возможно потому, что требует точного совпадения слишком большого подмножества словарей документа. Тем не менее, его показатели превосходят результаты алгоритма шинглов.

Метод шинглов, в частности, с довольно высокой точностью выявляет четкие и нечеткие дубликаты. Однако его основным недостатком является высокая чувствительность к перестановке фраз или предложений, т.е. небольшому изменению структуры текста.

Метод «описательных» слов набрал удивительно низкие результаты, тогда как в таблице 3 [6] приводится его сравнение с методом «шинглов», где

очевидно гораздо более быстрое выполнение поиска дубликатов именно методом «описательных» слов при небольшой разнице результатов.

Для выполнения сравнения было взято 60 миллионов документов объемами от 1600 до 3000 слов. Результаты методов сравнивались с экспертным мнением.

Таблица 3. Сравнение методов, описанных в параграфах 1.1.2. и 1.2.4.

Название метода	Кол-во предложенных копий	Разница <6%	Разница <15%	Разница <30%	Время обработки данных
Метод «описательных» слов	22 миллиона	61%	76%	91%	~1,5 часа
Метод «шинглов»	19 миллионов	66%	80%	94%	~3,5 часа

Алгоритм, используемый в утилите sif и системе Koala, почти не используется в силу существования гораздо менее ресурсозатратных и экономных по времени методов. Действительно, существенная проблема этого способа поиска дубликатов – сравнительно большое время, затрачиваемое на подсчет дактилограмм подстрок файлов, так как вычислительная сложность алгоритма достаточно высока. Однако, в своей работе [1], U. Manber предлагает некоторые идеи для решения выявленной проблемы. A N. Heintze в [8] предлагает несколько вариантов уменьшения количества ложно-положительных решений системы.

Также стоит отметить, что из всех рассмотренных методов поиска дубликатов документов только метод шинглов способен показать, какие именно фрагменты текста оказались заимствованы.

Глава 3. Модифицированный метод

Как уже было замечено, процесс выявления заимствований осуществляется при решении разнообразных задач. Однако наиболее востребованной была и остается проблема плагиата, особенно среди различных научных публикаций, студенческих курсовых или дипломных работ.

В настоящее время существуют различные сервисы, с помощью которых любой желающий может проверить какой-либо документ на наличие плагиата. Однако подобные сервисы обычно предоставляют свои услуги на платной основе и достаточно сильно ограничивают возможности в бесплатном режиме (например, количество обрабатываемых документов за фиксированное время или поиск лишь в ограниченных коллекциях). К тому же, принципы реализации работы сервисов предпочитают не разглашать, ведь иначе обман системы был бы лишь вопросом времени.

Рассмотренные в Главе 1 методы не учитывают специфику анализируемого документа. Тогда как при сосредоточении на документах определенного стиля можно выставить ряд условий, подходящих для анализа данного конкретного стиля и позволяющих улучшить показатели эффективности методов поиска дубликатов среди документов одного стиля.

Научный стиль, как и любой другой, имеет свои особенности, среди которых можно выделить следующие:

1. Наличие четкой структуры текста, то есть документы обычно поделены на разделы, нередко со схожими или даже одинаковыми названиями.
2. Частое использование шаблонных фраз вроде «таким образом», «из этого следует» и других.

3. Применение различных символов или слов на другом языке (обычно английском или латинском) для записи формул или обозначения терминов.

При проверке документов на схожесть в большинстве случаев хочется знать, какие именно части документа оказались выделены как заимствование, поэтому имеет смысл рассмотреть более подробно метод шинглов и модифицировать его с учетом перечисленных выше особенностей.

3.1. Описание модифицированного метода

Модификация основывается на указанной особенности документов научного стиля о разбиении на разделы. Таким образом идея этого метода сводится к сопоставлению каждого раздела одного документа к каждому разделу второго документа.

Процесс работы алгоритма в целом можно разбить на несколько этапов:

1. Оба рассматриваемых документа проходят через предварительную подготовку.
2. В каждом документе происходит выделение разделов.
3. Берется раздел и делится на блоки.
4. Берется блок текущего раздела, вычисляется его степень плагиатности и решается, является ли он плагиатом.
5. Возврат к шагу 4, если еще остались нерассмотренные блоки текущего раздела.
6. Решается, является ли плагиатом текущий раздел.
7. Возврат к шагу 3, если еще остались нерассмотренные разделы.
8. Решается, является ли плагиатом рассматриваемый документ.

Рассмотрим эти этапы более подробно.

Под предварительной обработкой документа подразумевается удаление так называемой «лишней информации»: знаки препинания, лишние пробелы,

цифры, отступы, стоп-слова (предлоги, союзы, междометия), шаблонные фразы и символы другого языка (см особенности 2 и 3 документов научного стиля выше). Также все слова переводятся в нижний регистр. Дополнительно следует выполнить стемминг – поиск основы слова путем удаления окончаний и суффиксов.

После всех манипуляций над текстом можно приступить к разбиению на шинглы, и на этом этапе следует внести соответствующую модификацию с учетом особенности 1 научного стиля: текст делится на разделы, а каждый раздел разбивается на непересекающиеся блоки одинаковой длины. Длину блоков можно изменять, например, в зависимости от длины абзацев. Далее каждый блок в соответствие с методом, описанным в параграфе 1.1.2., разбивается на шинглы, длину которых можно настраивать. Шинглы, в отличие от блоков, будут пересекаться, то есть их количество = количество слов в блоке - длина шингла k + 1. Затем шинглы текущего блока объединяют с шинглами блока, с которым сравнивают, и все они сортируются по возрастанию посчитанных контрольных сумм (хэш-функций). В качестве хэш-функции берется, например, MD5. Остается лишь последовательно сравнить стоящие рядом значения. Так как хэш-функции для разных наборов символов получаются разные, достаточно посчитать количество одинаковых значений и затем вычислить процент совпадений. Если этот процент превышает, скажем, 75%, можно сделать вывод, что рассматриваемый блок содержит в себе слишком много заимствований, то есть он плагиатный.

Каждый раздел первого документа сравнивается с каждым разделом второго документа. Каждый блок текущего раздела сопоставляется с каждым блоком раздела, с которым сравнивают. Процедура сравнивания блоков текущего раздела повторяется, либо пока не закончатся блоки, либо пока количество блоков, определенных как плагиатные, не превысит в процентном соотношении все те же 75%. Если имеет место последний вариант, можно сделать вывод, что рассматриваемый раздел содержит в себе слишком много

заимствований, то есть он плагиатный. Аналогично делается вывод о плагиатности документа: документ будет считаться плагиатным, если процентное соотношение разделов, определенных как плагиатные, превысит 75%.

3.2. Анализ результатов

Описанный в параграфе 3.1. подход позволяет сделать общий вывод об использовании плагиата в рассматриваемом тексте. Тем не менее метод шинглов в целом допускает уточнение частей текста, которые были засчитаны как скопированные.

Естественно, указанный порог в 75% можно менять. Также возможны удаление этого условия и простой подсчет процента совпадений.

Однако работа алгоритма именно в том виде, как описано в параграфе 3.1., позволяет сократить время определения плагиатности документа: попарное сравнение значений хэш-функций шинглов прекращается после достижения 75% совпадений или же, наоборот, 25% уникальности. Также подход с делением на блоки дает возможность проверить лишь отдельные фрагменты текста.

Для тестирования этого метода было выбрано 15 различных курсовых и дипломных работ, для каждой работы были созданы копии с разными процентами заимствования (от ~10% до ~90%). Затем документы (оригинал и его копия) сравнивались попарно описанным в параграфе 3.1. методом. За порог плагиата принимается 75% совпадение текста. Длина шингла k = 5. Дополнительно сравнения производились простым методом шинглов (параграф 1.1.2), LongSent (параграф 1.1.3) и TF-IDF (1) по формулам Okapi BM25 (2), (3). Результаты работы методов записаны в виде стандартной F-меры (4) в таблице 4.

Таблица 4. Результаты работы алгоритмов

Номер работы	Модифиц. метод	Метод шинглов	LongSent	TF-IDF
1	0.71	0.65	0.83	0.68
2	0.75	0.74	0.66	0.71
3	0.71	0.68	0.74	0.59
4	0.70	0.73	0.78	0.65
5	0.67	0.59	0.75	0.64
6	0.73	0.66	0.81	0.73
7	0.72	0.72	0.77	0.66
8	0.71	0.68	0.72	0.63
9	0.76	0.75	0.67	0.78
10	0.69	0.71	0.73	0.67
11	0.69	0.67	0.63	0.74
12	0.72	0.69	0.79	0.62
13	0.70	0.70	0.76	0.69
14	0.74	0.73	0.81	0.73
15	0.73	0.71	0.78	0.72

Модифицированный метод показал неплохие результаты, в большинстве случаев работая эффективнее простого метода шинглов. Однако он не дотянул до результатов LongSent, что, впрочем, не характеризует метод с негативной стороны: LongSent дает плохие результаты при замене слов в двух самых длинных предложениях, тогда как для снижения эффективности модифицированного метода требуется больше перестановок. С другой стороны, как и метод простых шинглов, модифицированный метод слишком чувствителен к перестановке слов, но в целом показатели обоих держатся на

приемлемом уровне. Метод TF-IDF в целом имеет средние результаты, к тому же он как раз устойчив к небольшим изменениям текста, что наталкивает на еще один вариант сопоставления блоков в модифицированном методе.

Так, можно построить частотный словарь для каждого раздела, затем внутри каждого блока выбираются и сцепляются в алфавитном порядке фиксированное количество слов с наибольшими весами, и контрольная сумма этой строки принимается за контрольную сумму блока. Соответственно, при определении плагиатности блока достаточно проверить значения контрольных сумм сравниваемых блоков на равенство. Однако такой подход может оказаться достаточно трудоемким в вычислении, но должен быть более устойчив к малым изменениям текста.

Таким образом, модифицированный метод учитывает стиль документов и их длину. Но он не выявляет цитирование и сноски на литературу, так как символы кавычек и обозначения сносок окажутся удалены при предварительной подготовке.

Заключение

В рамках данной курсовой работы были получены следующие результаты:

1. Изложены наиболее популярные и эффективные с вычислительной точки зрения алгоритмы решения проблемы поиска дубликатов.
2. Предоставлены оценки эффективности рассмотренных методов.
3. Описан модифицированный вариант метода и проанализированы полученные результаты.

Список литературы

[1] A. Broder. On the resemblance and containment of documents // In Proceedings of Compression and Complexity of Sequences 1997, pages 21–29. IEEE Computer Society, 1997.

[2] A. Broder, M. Charikar, A.M. Frieze, M. Mitzenmacher. Min-Wise Independent Permutations // Proceedings of the thirtieth annual ACM symposium on Theory of computing, 1998

[3] A. Broder, Identifying and Filtering Near-Duplicate Documents // in Proc. Annual Symposium on Combinatorial Pattern Matching, 2000.

[4] A. Chowdhury, O. Frieder, D. Grossman, M. McCabe. Collection statistics for fast duplicate document detection // ACM Transactions on Information Systems (TOIS), Vol. 20, Issue 2, April 2002.

[5] A. Kolcz, A. Chowdhury, J. Alspector. Improved Robustness of Signature-Based Near-Replica Detection via Lexicon Randomization // Knowledge Discovery and Data mining, 2004.

[6] S. Ilyinsky, M. Kuzmin, A. Melkov, I. Segalovich. An efficient method to detect duplicates of Web documents with the use of inverted index // World Wide Web Conference 2002.

[7] U. Manber. Finding Similar Files in a Large File System // Winter USENIX Technical Conference, 1994.

[8] N. Heintze. Scalable document fingerprinting // In Proc. of the 2nd USENIX Workshop on Electronic Commerce, Nov. 1996.

[9] M. Rabin. Fingerprinting by random polynomials. Report TR-15-81 // Center for Research in Computing Technology, Harvard University, 1981

[10] A. Broder, S. Glassman, M. Manasse, G. Zweig. Syntactic clustering of the

Web // Proc. Of the 6th International World Wide Web Conference, April 1997.

[11] D. Fetterly, M. Manasse, M. Najork, J. Wiener. A Large-Scale Study of the Evolution of Web Pages, World Wide Web Conference, May 2003.

[12] TF-IDF. http://www.tfidf.com

[13] C. Manning, P. Raghavan, H. Schütze. Introduction to Information Retrieval. Cambridge University Press. 2008. 117-120 с. 155-156 с.

[14] S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, M. Gatford. Okapi at trec-3 // The Third Text REtrieval Conference (TREC-3), 1995.

[15] Зеленков Ю.Г., Сегалович И.В. Сравнительный анализ методов определения нечетких дубликатов для WEB-документов // Труды 9-ой Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» RCDL’2007: Сб. работ участников конкурса, том 1. Переславль-Залесский, Россия: «Университет города Переславля», 2007. 166-174 с.