УДК 004.853

Л.Э. Чалая, Ю.Ю. Харитонова

Харьковский национальный университет радиоэлектроники, Харьков

МЕТОД ВЕКТОРНО-ГРАФОВОЙ КЛАСТЕРИЗАЦИИ ДОКУМЕНТОВ В СИСТЕМАХ ОБРАБОТКИ ТЕКСТОВОЙ ИНФОРМАЦИИ

Статья посвящена разработке метода кластеризации текстовых документов на основе векторного представления корпуса текстов с использованием графовой модели. Предложенный метод позволяет осуществлять кластеризацию текстовых документов с учетом весовых коэффициентов отдельных слов, встречающихся в корпусе. Предложенный метод использует косинусную меру в качестве расстояния между документами и может быть использован для структурирования корпусов текстовой информации большой размерности.

Ключевые слова: *текстовый документ, векторное представление, графовая модель, косинусное расстояние.*

Введение

Эффективность современных информационно-поисковых систем (ИПС) определяется, прежде всего, уровнем релевантности и пертинентности в части совершенствования организации запросов пользователей, поиска по параметрам, за счет кластеризации, поиска по подобию, ранжирования отзывов, использованием «сюжетных подходов», всестороннего использования семантических методов (в том числе с применением автоматической группировки документов по классификатору, автоматическим определением ранее незаданных или слабо структурированных документов, ранжированием документов по содержательной релевантности, автоматическим анализом и содержательным преобразованием запросов, выявлением семантически подобных документов по сравнению с эталоном) [1, 2].

Современные вычислительные средства, используемых для обработки электронных текстов, позволяют задавать различные ограничения на искомые комбинации слов в тексте, определяя обязательность или необязательность, допустимое расстояние между словами и порядок их нахождения в тексте. Это дает возможность проводить анализ слова во всех грамматических формах, точно и полно описывая возможные способы представления необходимого содержания в тексте. Для повышения точности анализа текстов разрабатываются методы предварительной лингвистической обработки, что требует во-первых, значительных вычислительных затрат для лингвистического анализа индексированной коллекции текстов, во-вторых, разработки специализированной поисковой машины. Автоматизированное извлечение знаний из текста является одной из основных задач искусственного интеллекта и непосредственно связано с пониманием текстов на естественном языке. Задачи автоматизированной аналитической обработки текстовой информации с использованием различных моделей представления данных рассмотрены, в частности, в работах [2 – 4]. Для извлечения знаний из текстовой информации используются различные методы автоматического анализа Data Mining. Такие методы используют алгоритмы и средства искусственного интеллекта для исследования и изъятия из больших объемов информации знаний, которые будут практически полезны и доступны для интерпретации человеком. К основным методам Data Mining относятся классификация, кластеризация, регрессия, поиск ассоциативных правил, аннотирование и автоматическое реферирование [2].

Особый интерес при автоматическом анализе текстовых документов представляют задачи кластеризации. Кластеризация – это разбиение множества документов на кластеры (группы документов с общими признакам), которые представляют собой подмножества, смысловые параметры которых заранее неизвестны. Численные методы кластеризации базируются на определении кластера как множества документов. Кластеризация может применяться в произвольной области, где необходимо исследование экспериментальных и статистических данных. Для задачи кластеризации характерен поиск групп наиболее похожих объектов. После определения кластеров используются другие методы Data Mining. Кластерный анализ позволяет рассматривать большой объем информации и сокращать, сжимать большие массивы информации. Результат кластеризации зависит от природы данных объектов и от представления кластеров. Кластеризация отличается от классификации тем, что для проведения анализа не нужно иметь выделенную зависимую переменную. Задача кластеризации решается на начальных этапах исследования, а ее решение помогает лучше понять данные. Все описанные выше методы авто-

матического анализа Data Mining обеспечивают определенную структуризацию текстовой информации, ее обобщение или аннотирования. Однако для извлечения знаний из электронных текстов, в частности, сравнения текстов и выявления в них совпадений, необходимы средства автоматического лингвистического анализа [2 – 4].

Применение кластерного анализа в общем виде сводится к следующим этапам:

– отбор выборки объектов для кластеризации;

– определение множества переменных, по которым будут оцениваться объекты в выборке и нормализация значений переменных;

– вычисление значений меры сходства между объектами;

– применение кластерного анализа для создания групп сходных объектов (кластеров).

Следует отметить, что существующие методы кластеризации не всегда позволяют сформировать кластеры при автоматической обработке электронных текстов, поскольку выходные матрицы текстовых документов, как правило, плохо обусловлены и имеют большую размерность.

Целью данной статьи является разработка метода кластеризации текстовых документов на основе их векторного представления с использованием графовой модели, который был бы работоспособным при обработке электронных текстов независимо от их объема и специфики.

Векторная модель представления документов в системах автоматической обработки текстов

Сегодня информационный поиск применяется в разных прикладных отраслях – от систем баз данных до веб-информационных поисковых систем. Основная идея состоит в нахождении документов, которые содержат термины, заданные пользователями в запросах. Отсутствие общих терминов в двух документах не обязательно означает, что документы не похожи между собой. Информационный поиск, согласно концепции традиционных подходов (например, векторная модель, вероятностная, булева) основаны на лексикографическом согласовывании терминов. Однако два термина могут быть семантически схожими (например, могут быть синонимами или иметь похожее значение), не смотря на лексикографическую разность. Таким образом, информационный поиск можно проводить лишь для документов с лексикографически похожими словами [3 – 5].

Для улучшения показателя релевантности выборки зачастую используют методы кластеризации, для выявления групп схожих по смыслу документов. Кластеризация документов – это одна из задач информационного поиска. Целью кластеризации документов является автоматическое выявление групп семантически похожих документов среди заданного фиксированного множества документов. Следует отметить, что группы формируются только на основе попарной схожести описаний документов, и никакие характеристики этих групп не задаются заранее, в отличие от классификации документов, где категории задаются заранее.

Рассмотрим наиболее распространенные меры оценки важности слов и текстовых фрагментах в обрабатываемых текстовых документах.

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

Мера TF-IDF часто используется в задачах анализа текстов и информационного поиска, например, как один из критериев релевантности документа поисковому запросу, при расчёте меры близости документов при кластеризации.

TF (term frequency – частота слова) – отношение числа вхождения некоторого слова к общему количеству слов документа. Значимость слова w_i в пределах отдельного документа может быть определена следующей TF-характеристикой:

$$\mathrm{tf}(t,d) = \frac{n_i}{\sum_k n_k}, \qquad (1)$$

где n_i – число вхождений слова t_i в документ d; $\sum_k n_k$ – общее число слов в данном документе.

IDF (inverse document frequency – обратная частота документа) – инверсия частоты, с которой некоторое слово встречается в документах коллекции. Учёт IDF уменьшает вес широкоупотребительных слов. Для каждого уникального слова в пределах конкретной коллекции документов существует только одно значение IDF.

IDF-характеристика определяется следующим отношением:

$$\mathrm{idf}(t,D) = \log \frac{|D|}{|(d_i \supset t_i|}, \qquad (2)$$

где $|D|$ – количество документов в корпусе; $|(d_i \supset t_i|$ – количество документов, в которых встречается t_i.

Выбор основания логарифма в формуле (2) не имеет значения, поскольку изменение основания приводит к изменению веса каждого слова на постоянный множитель, что не влияет на соотношение весов.

Таким образом, мера TF-IDF является произведением двух сомножителей:

$$\mathrm{tf} \cdot \mathrm{idf}(t,d,D) = \mathrm{tf}(t,d) \times \mathrm{idf}(t,D).$$

Большой вес в TF-IDF получат слова с высокой частотой в пределах конкретного документа и с низкой частотой употреблений в других документах.

Мера TF-IDF часто используется для представления документов коллекции в виде числовых векторов, отражающих важность использования каждого слова из некоторого набора слов (количество слов набора определяет размерность вектора) в каждом документе. Подобная модель называется векторной моделью и даёт возможность сравнивать тексты, сравнивая представляющие их вектора в какой либо метрике (евклидово расстояние, косинусная мера, манхэттенское расстояние, расстояние Чебышева и др.), то есть, производя кластерный анализ.

Для дальнейшего анализа необходимо построить векторную модель документа. Векторная модель – в информационном поиске представление коллекции документов векторами из одного общего для всей коллекции векторного пространства.

Векторная модель является основой для решения многих задач информационного поиска, как то: поиск документа по запросу, классификация документов, кластеризация документов.

Документ в векторной модели рассматривается как неупорядоченное множество термов. Термами в информационном поиске называют слова, из которых состоит текст, а также такие элементы текста, как, например, имена собственные или даты.

Различными способами можно определить вес терма в документе – «важность» слова для идентификации данного текста. Например, можно просто подсчитать количество употреблений терма в документе, так называемую частоту терма, – чем чаще слово встречается в документе, тем больший у него будет вес. Если терм не встречается в документе, то его вес в этом документе равен нулю.

Все термы, которые встречаются в документах обрабатываемой коллекции, можно упорядочить. Если теперь для некоторого документа выписать по порядку веса всех термов в этом документе, получится вектор, который и будет представлением данного документа в векторном пространстве. Размерность этого вектора, как и размерность пространства, равна количеству различных термов во всей коллекции, и является одинаковой для всех документов.

Формально вектор документа d_j можно представить следующим образом:

$$d_j = (w_{1j}, w_{2j}, ..., w_{nj}), \quad (3)$$

где d_j – векторное представление j-го документа, w_{ij} – вес i-го терма в j-м документе, n – общее количество различных термов во всех документах коллекции.

Располагая таким представлением для всех документов, можно, например, находить расстояние между точками пространства и тем самым решать задачу подобия документов – чем ближе расположены точки, тем больше похожи соответствующие документы. В случае поиска документа по запросу, запрос тоже представляется как вектор того же пространства – и можно вычислять соответствие документов запросу.

Для полного определения векторной модели необходимо указать, каким именно образом будет отыскиваться вес терма в документе. В нашем случае вес терма будет исчисляться на основе tf-idf индекса для документа.

Для дальнейшего анализа использовалась мера косинусного сходства векторов. Косинусное сходство – это мера сходства между двумя векторами предгильбертового пространства, которая используется для измерения косинуса угла между ними.

Если даны два вектора признаков, A и B, то косинусное сходство, cos(θ), может быть представлено с использованием скалярного произведения и нормы:

$$\cos(\Theta) = \frac{AgB}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}. \quad (4)$$

В случае информационного поиска, косинусное сходство двух документов изменяется в диапазоне от 0 до 1, поскольку частота терма (веса tf-idf) не может быть отрицательной. Угол между двумя векторами частоты терма не может быть больше, чем 90°.

Одна из причин популярности косинусного сходства состоит в том, что оно эффективно в качестве оценочной меры, особенно для разреженных векторов, так как необходимо учитывать только ненулевые измерения. Матрицу косинусных расстояний можно использовать в качестве весовой матрицы смежности графа, вершины которого соответствуют документам [6, 7].

Реализация метода кластеризации текстов с использованием векторного представления документов

Для получения векторных моделей документов с целью их последующей кластеризации был разработан программный модуль на языке Java, который производит обработку входных данных, а на выходе получаем матрицу косинусных расстояний между векторами документов.

Алгоритм работы модуля предполагает последовательную реализацию следующих этапов:

– парсинг исходных данных из формата csv;

– разделение предложений на слова;

– удаление стоп-слов;

– стемминг исходных данных;

– получение tf-idf индексов для слов документа;

– отсечение индексов ниже границы;
– получение матрицы векторов для документов;
– генерация матрицы косинусных расстояний;
– запись матрицы в csv файл.

Для реализации программного парсинга текстовых данных, представленных в CSV формате, была разработана специальная процедура. Каждая строка CSV файла представляется одной строкой формируемой текстовой таблицы, а значения отдельных колонок разделяются разделительным символом delimiter (запятой). Кроме того, стандарт CSV допускает использование иных символов в качестве разделителя. В частности в локалях, где десятичным разделителем является запятая, в качестве табличного разделителя, как правило, используется точка с запятой. Значения, содержащие зарезервированные символы (двойная кавычка, запятая, точка с запятой, новая строка) обрамляются двойными кавычками; если в значении встречаются кавычки, то они представляются в файле в виде двух кавычек подряд. Строки разделяются парой символов CR LF (0x0D 0x0A).

При представлении текстов в CSV формате возможно использование набора значений, характеризующихся разными типами разделителями, с различной кодировкой и с различными окончаниями строк. Это значительно затрудняет перенос данных из одних программ в другие в рамках проектируемого модуля векторного представления текстов. В программе предусмотрена возможность выбора используемого разделителя как при записи, так и при чтении. CSV парсер при этом представляет собой отдельный класс, входные данные которого должны содержать путь к обрабатываемому файлу, тип используемого разделителя и тип кодировки файла. Первоначальная диаграмма класса CSV парсера для решаемой задачи приведена на рис. 1.

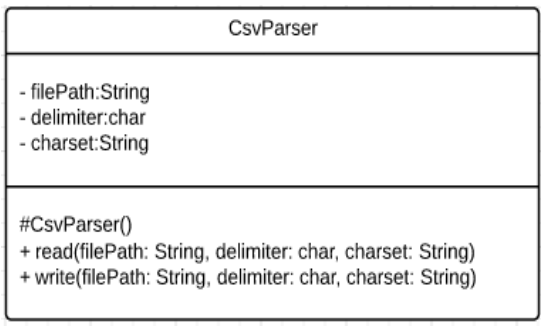


Рис. 1. Диаграмма класса CsvParser

Для чтения и записи текстовых файлов в Java используются наследники абстрактных классов Reader и Writer, а именно FileReader и FileWriter. Данные классы принимают на вход объект File, который указывает на физический файл в файловой системе, и производят операции чтения и записи соответственно.

Для того чтобы операции чтения и записи были эффективными с точки зрения проектирования программной архитектуры, необходимо предусмотреть классы модели, которые будут эффективно работать с входящими данными. Было принято решение создать класс DynamicMatrix.java, для хранения двумерного массива объектов, которые буду представлять документы в виде строк.

Исходя из целей, класс должен удовлетворять следующим требованиям дизайна:

– класс должен служить контейнером для объектов;

– контейнер должен быть динамически расширяемым;

– контейнер должен быть типизируемый;

– контейнер должен предоставлять методы для добавления объектов в строку, а также для добавления новых строк;

– контейнер должен предоставлять методы для получения объектов.

В применения предложенной процедуры форируется класс DynamicMatrix.java, диаграмма которого представлена на рис. 2.

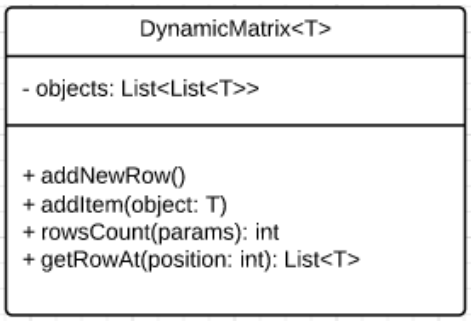
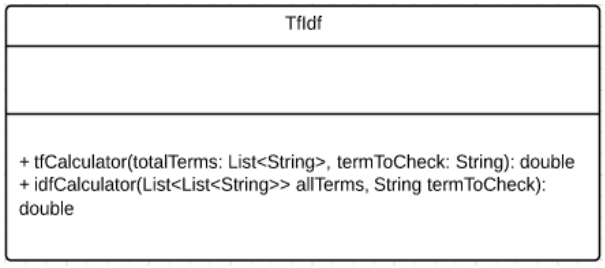
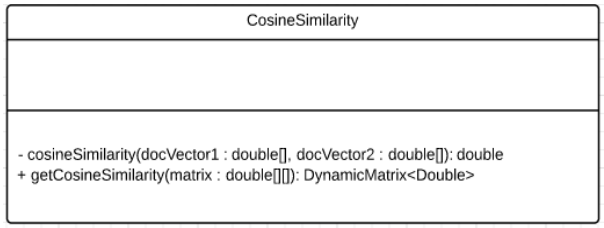


Рис. 2. Диаграмма класса DynamicMatrix

Операция разделения на слова выполняется на этапе чтения файла. В результате парсинга CSV файла получается матрица, строки которой представляют документы, а значение в каждом столбце – это конкретный терм в документе.

Одним из наиболее ответственных этапов подготовки текстовых документов к кластеризации является этап предварительной обработки данных (ПОД). ПОД состоит из удаления разметки, служебных символов и стоп-слов (часто встречающиеся слова, которые не несут смысловой информации – местоимения, предлоги, союзы и т.д.), выбора модели представления документов, выявления информативных признаков (терминов) и присвоения им веса. В модуле также использованы методы удаления стоп-слов, а также стемминга. Удаление стоп-слов подразумевает собой сравнение текущего списка термов с заранее подготовленным списком стоп-слов для конкретного языка и удаление совпадений (библиотеки стоп-слов для различных языков доступны). Из списка термов убираются те слова, кото-





рые имеют высокий показатель tf-idf, но не несут смысловой нагрузки. Из существующих методов стемминга был выбран стеммер Портера, достоинство которого заключается в том, что он не использует никаких словарей, и выделение основы осуществляется путем преобразования слова согласно определенным правилам. Реализации этого алгоритма можно найти в свободном доступе в сети интернет, в том числе и для Java. Для генерации матрицы косинусных расстояний необходимо предварительно построить векторные модели документов. Для этого были созданы вспомогательные классы TfIdf.java и CosineSimilarity.java. Для начала для каждого терма в документе был высчитан tf-idf индекс, совокупность которых представляла собой вектор документа. Далее из векторов была составлена матрица, в которой каждая строка представляла собой отдельный документ. Перед тем как высчитать косинусные расстояния между векторами, необходимо убедиться, что каждый вектор содержит показатели tf-idf для каждого терма, встречаемого в корпусе документов. Таким образом, матрица была обработана еще раз, с целью добавить в каждый вектор недостающие веса термов.

Полученная матрица была обработана при помощи класса CosineSimilarity.java, который принимает на вход матрицу векторов, и возвращает матрицу косинусных расстояний. Диаграммы сформированных вспомогательных классов TfIdf.java и CosineSimilarity.java приведены на рис. 3 и 4 соответственно.

Рис. 3. Диаграмма класса TfIdf.java

Рис. 4. Диаграмма класса CosineSimilarity.java

На этапе тестирования модуля в качестве входных данных была выбрана выборка аннотаций авторефератов диссертационных работ, которые содержат краткое описание сути работы, а также цифровой шифр научной специальности.

На рис. 5 и рис. 6 приведены соответственно фрагменты расширенной матрицы векторов и матрицы косинусных расстояний, полученных в процессе тестирования ограниченного набора документов.

Полученная матрица косинусных расстояний используется для дальнейшей кластеризации текстовых документов.

```
агенто            агротехническ      адаптац             адаптивн            адаптирова          адекватн            администрирован      а
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.1188582088741771 0.04459236274160565 0.0                 0.0                 0.0                 0.0                  0
0.079421383809934 0.0                0.0                 0.08930380470871524 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.04218834912101375 0.06554339798468019 0.0                 0.0                  0
0.0               0.0                0.0                 0.07082715545863622 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.06158883083359672 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.13214067124347362 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.03572152188348609 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.1202766481287345  0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.05512200359606906 0.0                 0.07322965991511385 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.06035705421692478 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.22533533175755013  0
0.0               0.0                0.0                 0.0                 0.0                 0.0                 0.0                  0
```

Рис. 5. Фрагмент расширенной матрицы векторов

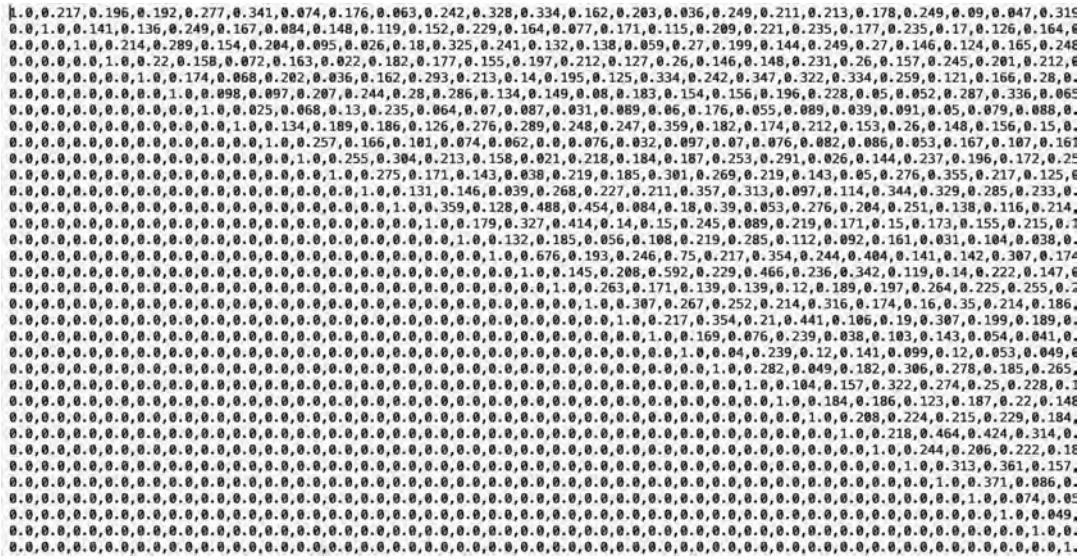


Рис. 6. Фрагмент матрицы косинусных расстояний

Эта матрица является матрицей смежности, которая визуализируется при помощи графа, каждое ребро которого имеет вес, равный косинусному расстоянию между документами.

На рис. 7 приведен пример такого графа, построенного на основе матрицы косинусных расстояний для 20 вершин. Из рис. 7 видно, что на графе можно выделить компоненты связности, позволяющие сформировать кластеры документов. Под компонентом связности графа будем понимать такое множество его вершин, что для любых двух вершин из этого множества существует путь из одной в другую, и не существует пути из вершины этого множества в вершину другого множества. В предложенном методе кластеризации задается пороговое значение расстояния, при которой можно удалить ребра с низким весом. Например, в графе, представленном на рис. 7, присутствуют три соединительных ребра, с малым весом, убрав которые можно получить четко выраженные компоненты связности (рис. 8).

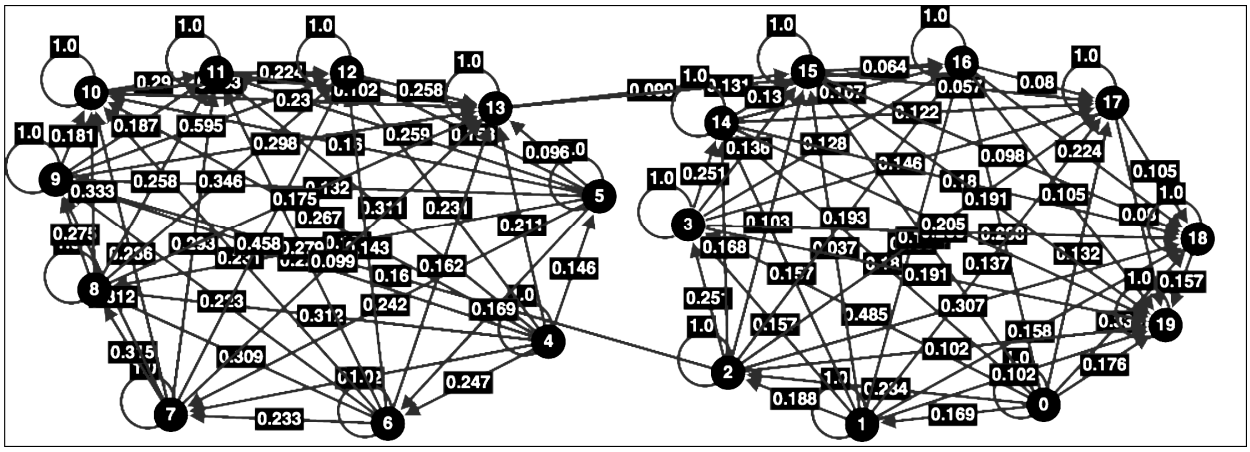


Рис. 7. Пример графа на основе матрицы косинусных расстояний

Можно предположить, что все вершины, которые входят в тот или иной компонент связности, принадлежат одному и тому же кластеру. Таким образом, так как вершины соответствуют документам анализируемого корпуса, а ребра – косинусным расстояниям между документами, то документы также принадлежат сформированным кластерам.

Отметим, что при проведении тестового эксперимента была также предпринята попытка обработать векторную модель анализируемого корпуса документов при помощи пакета Fuzzy Clustering в программной среде Matlab с использованием алгоритмов k-means и fuzzy c-means.

Однако ни один из методов кластеризации не позволил сформировать кластеры, поскольку выходные матрицы текстовых документов, как правило, плохо обусловлены и имеют большую размерность.

В то же время тестирование предложенного метода подтвердило его работоспособность.

Выводы

Анализ существующих методов информационного поиска показал, что существует ряд подходов, которые применимы для решения данной задачи кластеризации электронных текстовых документов.

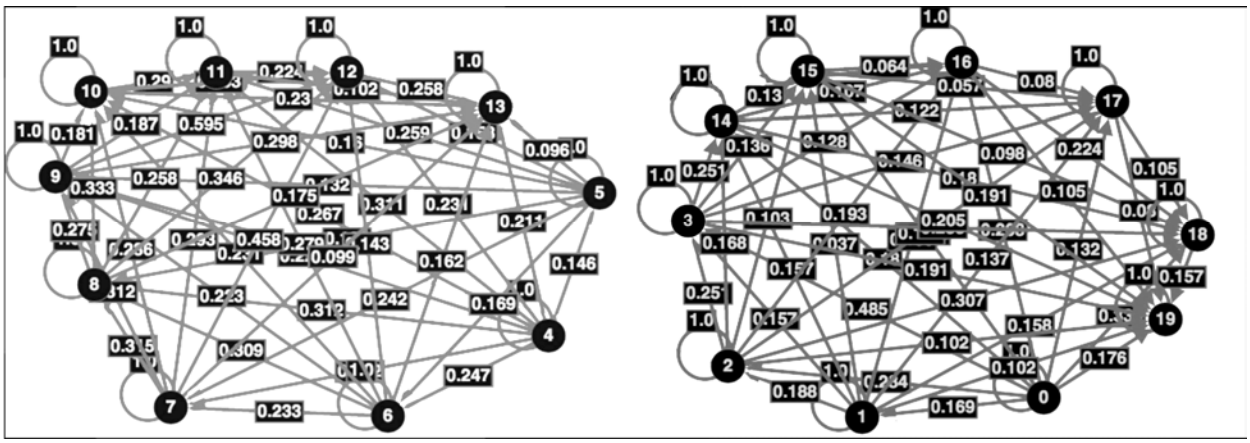


Рис. 8. Компоненты связности модифицированного графа

Предложенный в данной работе метод кластеризации основан на применении векторной модели документа. Для получения таких моделей и последующей кластеризации документов был разработан программный модуль на языке Java, который производит поэтапную обработку входных данных и формирование матрицы косинусных расстояний между векторами документов. Метод использует такие вспомогательные процедуры как удаление стоп-слов из документов, стемминг, определение важности терма в корпусе документов по tf-idf характеристикам терма. Выходные данные могут быть обработаны для получения визуализированного представления графов, на основе которых производится дальнейший поиск кластеров. В результате тестирования были получены кластеры документов, которые объединяют в себе документы на основе их лексического содержания.

Перспективной является дальнейшая модернизация алгоритма для увеличения быстродействия, точности обработки и улучшения работы с большими массивами данных.

Список литературы

1. Рассел С. Искусственный интеллект. Современный подход [Текст] / С. Рассел, П. Норвиг, 2-е изд.: Пер. с англ. – М.: Издательский дом «Вильямс», 2006. – 1408 с.

2. Feldman R. The text mining handbook: advanced approaches in analyzing unstructured data [Текст] / R. Feldman, J. Sanger. – Cambridge University Press, 2007. – 410 p.

3. Moyotl-Hernandez E. An Analysis on Frequency of Terms for Text Categorization [Текст] / E. Moyotl-Hernandez, H. Jimenez-Salazar // Procesamiento del lenguaje natural. – 2004. – Vol. 33. – P. 141-146.

4. Moyotl-Hernandez E. Some Tests in Text Categorization using Term Selection by DTP [Текст] / E. Moyotl-Hernandez, H. Jimenez-Salazar // Proceedings of the Fifth Mexican International Conference on Computer Science ENC'04. – Colima. – 2004. – P. 161-167.

5. Using Bigrams in Text Categorization [Электронный ресурс]. – Электрон. текст. дан. – 2003. – Режим доступа: www/ URL: www.cs.umass.edu/~ronb/ papers /bigrams.pdf. – 05.02.2014 г. – Загл. с экрана.

6. Boutin F. Cluster Validity Indices for Graph Partitioning [Текст] / F. Boutin, M. Hascoet // Proceedings of the Eight International Conference on Information Visualization (IV'04). – 2004. – P. 232-240.

7. Чалая Л.Э. Матричное синонимическое представление корпусов электронных текстов в информационных поисковых системах [Текст] / Л.Э. Чалая, Ю.Ю. Шевякова // Міжн. НТК «Інформаційні системи та технології – 2013», матеріали першої міжнар. наук.-техн. конф., 16–22 вересня 2013 р. Євпаторія, 2013. – С. 71-72.

Поступила в редколлегию 14.08.2015

Рецензент: д-р техн. наук, проф. С.Г. Удовенко, Харьковский национальный университет радиоэлектроники, Харьков.

МЕТОД ВЕКТОРНО-ГРАФОВОЇ КЛАСТЕРИЗАЦІЇ ДОКУМЕНТІВ В СИСТЕМАХ ОБРОБКИ ТЕКСТОВОЇ ІНФОРМАЦІЇ

Л.Е. Чала, Ю.Ю. Харитонова

Статтю присвячено розробці метода кластеризації текстових документів на основі векторного представлення корпуса текстів з використанням графової моделі. Запропонований метод дозволяє здійснювати кластеризацію текстових документів з урахуванням вагових коефіцієнтів окремих слів, які зустрічаються в корпусі. Запропонований метод використовує косинусну міру як відстань між документами та може бути використаний для структурування корпусів текстової інформації великого розміру.

Ключові слова: *текстовий документ, векторне уявлення, графова модель, косинусна відстань.*

THE VECTOR-GRAPH'S CLUSTERING METHOD OF DOCUMENTS IN TEXT PROCESSING SYSTEMS

L.E. Chala, Yu.Yu. Kharytonova

The article is devoted to the development of a method of clustering of text documents based on vector representation of text corpus using the graph model. The proposed method allows the clustering of text documents, taking into account the weighting factors of individual words in the corpus. The proposed method uses a cosine measure as the distance between the documents and can be used for the structuring text information corps of large dimension.

Keywords: *text document, vectorial presentation, count model, cosine distance.*