# МАТЕМАТИЧЕСКОЕ ПРОГРАММИРОВАНИЕ В TEXT MINING

Р.М.Алгулиев Р.М.Алыгулиев
Баку, Институт Информационных Технологий НАН Азербайджана
rasim@science.az, a.ramiz@science.az

**Аннотация.**

С ростом объема текстовой информации, доступной на WWW, стало все более и более необходимым для пользователей использовать автоматизированные инструментальные средства, чтобы найти, извлечь, фильтровать и оценить желаемую информацию. Одним из таких средств является автоматическое резюмирование текстовых документов. С этой целью в настоящей статье предлагается метод резюмирования текстовых документов, основанный на кластеризации предложений. В отличие от традиционных методов, предлагается новый метод кластеризации, математическая реализация которого опирается на задачу целочисленного программирования. В статье во избежание трудоемких вычислительных процедур предлагается нейросетевая реализация задачи целочисленного программирования.

## Введение.

Взрывной рост WWW резко увеличил скорость и масштаб распространения информации. Из всех видов информации, собранной на WWW, как правило, наибольший интерес представляют текстовые данные. Несмотря на свою простоту, тексты на сегодняшний день - это важнейшие носители информации и, очевидно, еще долго будут оставаться таковыми. Подавляющая часть научных статей, документации, on-line новостей и т.п. имеют текстовую форму, что обусловливает интерес к проблеме обработки и поиска текста. Кроме того, большие объемы информации, не имеющие изначально текстовой формы, часто приводятся к ней с тем, чтобы использовать для поиска и обработки этой информации. В настоящее время механизмы поиска, на основе ключевого слова на Интернете, возвращают сотни и даже тысячи документов, что ошеломляет пользователя. С ростом количества текстовых документов, доступных на Интернете, обычные информационные поисковые технологии стали неудовлетворительными для нахождения релевантной информации. Поэтому, возникает увеличивающаяся потребность в новой технологии, которая может помочь пользователю фильтровать огромное количество информации, и быстро идентифицировать самые релевантные документы.

С ростом объема текстовых документов, представление пользователю резюме каждого документа очень облегчает задачу обнаружения ожидаемых документов. Текстовый поиск и резюмирование – это две технологии, которые дополняют друг друга.

Цель задачи автоматического резюмирования состоит в извлечении из документа значимых пассажей (контекст, предложение, параграф), отражающих его контента [1]. За последние годы в научной литературе опубликовано большое количество работ по проблеме резюмирования текстовых документов и предложены многочисленные методы резюмирования. Например, в работах [2,3] для извлечения значимых параграфов предложен метод TRM (Text Relationship Map). Идея метода заключается в представлении текста в виде графа, вершинами, которых являются параграфы. Каждый параграф идентифицируется взвешенным вектором слов, и вычисляется мера подобия между параграфами, определенная скалярным произведением. Если мера подобия больше заданного порога, то эти вершины соединяются. Критерий извлечения параграфа в резюме определяется количествами ребер, связывающих его с другими. В работе [3] предложены четыре типа критериев выбора параграфа: bushy path, depth-first path, segmented bushy path, augmented segmented bushy path.

Большая часть работ посвящена к определению счета релевантности предложения [4,5,6,7,8,9,10] с целью включения его в резюме. Счет релевантности предложения в работе [4] определяется взвешенной комбинацией его локальной и глобальной характеристик. Локальная характеристика предложения определяется методом Luhn [11], где вес слова определяется не по формуле TF*IDF (Term Frequency*Inverse Document Frequency), а по формуле TL*TF (Term Length*Term Frequency). Идея метода TL*TF [12] базируется на том, что

*Труды 7ой Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» - RCDL'2005, Ярославль, Россия, 2005.*

слова, которые появляются часто, стремятся быть краткими. Такие слова не описывают основную тему документа, т.е. являются стоп словами. Наоборот, слова, которые появляются редко, стремятся быть длинными. Выгода от использования TL*TF, для взвешивания слов, является то, что этот метод не требует никаких внешних ресурсов, и использует только информацию в пределах документа. Глобальная характеристика определяется методом TRM. Работа [5] предлагает два подхода, адресованные к этой проблеме: MCBA (Modified Corpus-Based Approach) и LSA+TRM (Latent Semantic Analysis+TRM). Первый подход является обучаемым, который учитывает некоторые особенности, включая позицию предложения в параграфе, позитивные ключевые слова, негативные ключевые слова, центральность предложения в документе и сходство предложения к заглавию. Второй подход получает семантическую матрицу документа с помощью LSA. Потом, используя семантическое представление, конструирует семантический TRM. Для определения счета релевантности предложения газетных статей в работе [6] предлагается комбинирование статистической и лингвистической особенностей. Статистическая особенность определяется стандартными методами информационного поиска, а лингвистическая особенность - из анализа резюме газетных статей. Текстовые резюме могут быть запросо-релевантными и универсальными. Запросо-релевантное резюме представляет содержание документа, которое близко связано с запросом поиска. Создание запросо-релевантного резюме по существу является процесс восстановления запросо-релевантных предложений (пассажей) из документа, которое имеет сильную аналогию с процессом поиска текстов. Поэтому, запросо-релевантное резюме часто достигается, расширением обычной технологии информационного поиска, и до настоящего времени, в литературе большое количество резюмирования текстов относятся к этой категории. С другой стороны, универсальное резюмирование позволяет обеспечить резюме с широким охватом содержания документа. Запросо-релевантное резюме полезно для того, чтобы ответить на такие вопросы, как, релевантен ли данный документ запросу пользователя, а если релевантен, какая часть(и) документа является релевантной. Запросо-релевантное резюме не охватывает полный смысл содержания документа, и, следовательно, не является подходящим для краткого обзора контента документа. Для того чтобы ответить на такие вопросы, как, к какой категории принадлежит документ, и каковыми являются ключевыми точками в документе, должно быть создано универсальное резюме и представлено к пользователю. Исходя из этого соображения в работе [7] предлагается два метода универсального резюмирования. Первый метод использует стандартные методы информационного поиска для ранжирования предложений относительно их счетов релевантностей. Счет релевантности каждого предложения определяется скалярным произведением взвешенных векторов документа и предложения. В этом методе основное усилие направлено на минимизацию избыточности в резюме, забывая при этом широкого охвата контента документа. Это следует из того, что после выбора предложения, с наибольшим значением счета релевантности, оно удаляется из документа. После удаления предложения вектор взвешенных слов документа вычисляется заново, где слова, содержащиеся в удаленном предложении, не присутствуют в определении этого вектора. Второй метод, используя LSA, идентифицирует семантически значимые предложения для создания резюме. Работа [9] посвящена резюмированию web-страниц, где по известным четырем методом вычисляется счет релевантности каждого предложения, и окончательный счет релевантности равняется сумме этих четырех счетов. В работах [8] и [10] прежде чем определить представительные предложения сначала кластеризируются параграфы и предложения. Метод, предложенный в работе [8], в основном состоит из трех фаз. На первой фазе создается взвешенный вектор параграфов, на второй фазе методом $k$-средних осуществляется кластеризация параграфов (разбиение в тематические разделы) и предлагается новый алгоритм определения количества кластеров, основанный на минимизации некоторой целевой функции. Наконец, на третьей фазе из каждого тематического раздела извлекаются предложения, с целью включения в резюме. Анализ вида целевой функции показывает, что такой подход определения количества кластеров и количество выбираемых представительных параграфов (из каждого кластера один параграф) не может охватывать главного содержания документа. Все это связано с тем, что нет четкого доказательства определения количества кластеров. А в работе [10] кластеризация предложений реализуется алгоритмом иерархической кластеризации, который с вычислительной точки зрения является более сложным, чем алгоритм $k$-средних.

В настоящей работе с целью обеспечения минимальной избыточности в резюме и максимально возможной степени охвата контента документа предлагается новый метод кластеризации предложений, основанный на решении задачи целочисленного квадратичного программирования с булевыми переменными. Поскольку решение задачи целочисленного программирования связано большими вычислительными трудностями, то предлагается использовать нейронные сети, с обратными связями. Определение количество кластеров является одним из сложных задач при кластеризации. Поэтому, в данной работе также предлагается алгоритм пошагового определения количества кластеров. После кластеризации, во избежание от избыточности в резюме, на каждом

кластере, т.е. на каждом тематическом разделе определяются представительные предложения и их количества.

## 1. Задача кластеризации предложений.

Для обнаружения естественных групп в наборе данных кластеризация является одной из самых полезных подходов в процессе интеллектуального анализа данных (data mining). Для решения задачи кластеризации обычно используются традиционные алгоритмы, такие как алгоритм $k$-средних, иерархическая кластеризация, алгоритм GEM (Gaussian expectation-maximization) и т.д. [13,14,15,16]. Среди них широко распространенным является алгоритм $k$-средних. Это обусловлено тем, что этот алгоритм математически хорошо формулируется. Формулировка алгоритма $k$-средних как задаче математического программирования была предложена в работах [17,18]. В работе [19] алгоритм $k$-средних сформулирован в терминах негладкой и невыпуклой оптимизации. Рассматривая в этом разделе задачи кластеризации, мы не применяем традиционные методы, а используем методику, предложенную в работе [20].

Пусть документ $d$ состоит из $m$ предложений, и представим его в виде набора предложений $d = (s_1, s_2, ..., s_m)$. Идея задачи кластеризации состоит в разбиении множества $d = (s_1, s_2, ..., s_m)$ на непересекающиеся кластеры $C_1, C_2, ..., C_q$, $q \geq 2$, с целью обеспечения максимальной близости между предложениями одного кластера, соответствующими определенной смысловой тематике, и максимального различия между кластерами. Определение понятия близость определяется метрикой Евклида, которой будет введена ниже.

Прежде чем перейти к формулировке метода, с использованием векторной модели каждое предложение идентифицируется взвешенным вектором $s_i = (w_{i1}, w_{i2}, ..., w_{in})$ слов, которые появляются в документе, где $n$-количество слов в документе $d$.

Вес $w_{ij}$ слова $j$ зависит от частоты его появления в конкретном предложении $i$ и во всем наборе предложений (в документе), который определяется формулой TF*IDF:

$$w_{ij} = f_{ij} \log_2\left(\frac{m}{m_j}\right), \; i = 1,...,m; \quad j = 1,...,n.$$

где $m_j$- количество предложений, в которых присутствует слова $j$.

Функция $f_{ij}$ частоты появления слова $j$ в предложении $i$, вычисляется следующим образом:

$$f_{ij} = \frac{n_{ij}}{n \cdot len(s_i)},$$

где $n_{ij}$- количество появления слова $j$ в предложении $i$. Здесь во избежание смещения, вызванного длиной (количество слов) предложения, функция $f_{ij}$ нормализована относительно длину предложения, $len(s_i)$-длина предложения $s_i$.

Для определения близости $d_{ip}$ между предложениями $s_i$ и $s_p$ наиболее часто используется евклидово расстояние:

$$d_{ip} = \sqrt{\sum_{r=1}^{n} (w_{ir} - w_{kr})^2}, \; i, p = 1,...,m.$$

## 2. Сведение задачи кластеризации к целочисленному квадратичному программированию.

Близость предложений в кластерах и отдаленность предложений, отнесенных к разным кластерам, означает, что общая сумма расстояний между предложениями в пределах кластерах должна быть минимальной, а общая сумма расстояний между предложениями, отнесенными к разным кластерам, должна быть максимальной.

Следуя вышесказанному, определим сумму $S_k$ расстояний $d_{ip}$ между предложениями $s_i$ и $s_p$ в кластере $C_k$:

$$S_k = \frac{1}{2} \sum_{s_i \in C_k} \sum_{s_p \in C_k} d_{ip}, \; k = 1,...,q; \quad i, p = 1,...,m.$$

Суммируя по $k$, получим общую сумму расстояний между предложениями во всех кластерах $C_k$, $k = 1,...,q$:

$$\sum_{k=1}^{q} S_k = \frac{1}{2} \sum_{k=1}^{q} \sum_{s_i \in C_k} \sum_{s_p \in C_k} d_{ip}, \qquad (2.1)$$

где $i, p = 1,...,m$.

Теперь определим сумму $S_{kl}$ расстояний $d_{ip}$ между предложениями $s_i$ и $s_p$, отнесенными к разным кластерам $C_k$ и $C_l$, соответственно:

$$S_{kl} = \frac{1}{2} \sum_{s_i \in C_k} \sum_{s_p \in C_l} d_{ip}$$

где $k, l = 1,...,q, i, p = 1,...,m$.

Суммируя по $k$ и $l$ $(l \neq k)$, получим общую сумму расстояний между предложениями, отнесенными к разным кластерам:

$$\sum_{k=1}^{q}\sum_{\substack{l=1\\ l\neq k}}^{q} S_{kl} = \frac{1}{2}\sum_{k=1}^{q}\sum_{\substack{l=1\\ l\neq k}}^{q}\sum_{s_i\in C_k}\sum_{s_p\in C_l} d_{ip}, \quad (2.2)$$

Таким образом, задача кластеризации предложений сводится к минимизации суммы (2.1) и одновременно, к максимизации суммы (2.2). Иными словами задача сводится к следующей задаче минимизации:

$$\sum_{k=1}^{q}\sum_{s_i\in C_k}\sum_{s_p\in C_k} d_{ip} - \sum_{k=1}^{q}\sum_{\substack{l=1\\ l\neq k}}^{q}\sum_{s_i\in C_k}\sum_{s_p\in C_l} d_{ip} \to \min \quad (2.3)$$

Введем булево переменное $x_{ik}$, равное $1$, если предложение $s_i$ относится кластеру $C_k$, или равное $0$, в противном случае:

$$x_{ik} = \begin{cases} 1, & \text{если } s_i \in C_k \\ 0, & \text{если } s_i \notin C_k \end{cases}, \; i = 1,...,m; \quad k = 1,...,q.$$

После такого обозначения формулу (2.3) записываем в следующем виде:

$$\sum_{k=1}^{q}\sum_{i=1}^{m}\sum_{p=1}^{m} d_{ip}x_{ik}x_{pk} - \sum_{i=1}^{m}\sum_{k=1}^{q}\sum_{\substack{p=1\\ l\neq k}}^{m}\sum_{l=1}^{q} d_{ip}x_{ik}x_{pl} \to \min \quad (2.4)$$

Введением следующего обозначения

$$d_{ip}e_{kl} = a_{ikpl},$$

где

$$e_{kl} = \begin{cases} 1, & \text{если } k = l \\ -1, & \text{если } k \neq l \end{cases},$$

задачу (2.4) перепишем в компактном виде:

$$\sum_{i=1}^{m}\sum_{k=1}^{q}\sum_{p=1}^{m}\sum_{l=1}^{q} a_{ikpl}x_{ik}x_{pl} \to \min. \quad (2.5)$$

Так как кластеры не пересекаются, т.е. каждое из $m$ предложений отнесено только к одному из $q$ кластеров, то должно выполняться следующее условие:

$$\sum_{k=1}^{q} x_{ik} = 1, \quad i = 1,...,m. \quad (2.6)$$

С другой стороны, предполагается, что каждый кластер содержит хотя бы одно предложение:

$$\sum_{m=1}^{m} x_{ik} \geq 1, \quad k = 1,...,q, \quad (2.7)$$

где

$$x_{ik} \in \{0,1\} \text{ для любого } i,k. \quad (2.8)$$

Итак, задача кластеризации предложений сведена к задаче целочисленного квадратичного программирования с булевыми переменными (2.5)-(2.8).

## 3. Нейросетевая реализация задачи целочисленного квадратичного программирования.

Задача (2.5)-(2.8) относится к задачам комбинаторной оптимизации. Многие из таких задач являются $NP$-полными, решение которых связано с недоступными временными затратами. Для решения таких задач разрабатываются специальные методы и алгоритмы, характеризующиеся полиномиальной сложностью. Однако, существующие в настоящее время алгоритмы позволяют найти приемлемые по качеству и временным затратам решения только для задач небольшой размерности. Поскольку нет убедительных доводов в пользу существования алгоритмов их решения за приемлемое время, то эти задачи относятся к задачам $NP$-полных.

Поэтому для решения данного типа задач представляется целесообразным использовать нейронные сети, которые нашли эффективное приложение в задачах комбинаторной оптимизации [20,21]. Использование нейронных сетей с обратными связями позволяет существенно сократить время решения таких задач (и еще в большей степени задач $NP$-полных). Ясно, что нейронные сети в общем случае не гарантируют глобальности оптимального решения задачи (2.5)-(2.8). Однако на практике зачастую требуется за определенное время найти одно или несколько локальных минимумов. В таком случае использование нейронных сетей очень эффективно. Исходя их этого соображения, с целью обеспечения практичности оптимизационного подхода к задаче кластеризации, предлагается нейросетевая реализация задачи (2.5)-(2.8).

С целью синтеза нейронной сети для решения задачи оптимизации синтезируем тройку вида $\{\boldsymbol{N},\boldsymbol{W},\boldsymbol{B}\}$, где $\boldsymbol{N}$ -множество нейронов сети, $\boldsymbol{W}$ - матрица синаптических связей и $\boldsymbol{B}$ -вектор внешних смещений. Задача синтеза сети в общем случае заключается в определении всех компонентов данной тройки, т.е. вида и количества нейронов, структуры матрицы связей и значение ее элементов, значение внешних смещений. Считаем, что тип и модель динамики нейроподобных элементов определены. Поэтому задача синтеза сети сводится к определению структуры сети, матрицы связей $\boldsymbol{W}$ и

векторы смещений $\boldsymbol{B}$, удовлетворяющих целевому использованию синтезируемой сети.

Синтез нейронной сети для решения задачи оптимизации состоит из следующих этапов:

**Этап 1.** Нейросетевая интерпретация задачи.

Для нейросетевой интерпретации задачи оптимизации рассмотрим сеть бинарных нейронов, представляющую собой матрицу $\boldsymbol{Y} = \|y_{ik}\|$ размерностью $m \times q$. Каждой булевой переменной $x_{ik}$ ставится соответствие выходной сигнал $y_{ik}$ $ik$-го нейрона. Возбужденное состояние нейрона $y_{ik} = 1$ в такой матрице соответствует тому факту, что предложение $i$ отнесено кластеру $k$.

**Этап 2.** Конструирование энергетической функции сети.

Второй этап процесса построения оптимизируемой сети заключается в конструировании энергетической функции сети. Конструируемую энергетическую функцию сети построим в виде суммы, отдельные слагаемые которой представляют собой выпуклые функции, принимающие минимальные значения на состояниях сети, удовлетворяющих рассмотренным ограничениям на состояниях сети и минимизирующих целевую функцию.

Исходя из вышесказанного, слагаемое, обеспечивающее задачу минимизации (2.5), можно конструировать в следующем виде:

$$\boldsymbol{E_0} = -\frac{\lambda_0}{2}\sum_{i=1}^{m}\sum_{k=1}^{q}\sum_{p=1}^{m}\sum_{l=1}^{q} a_{ikpl}\, y_{ik}\, y_{pl}\ , \quad (3.1)$$

слагаемые, обеспечивающие выполнения ограничений (2.6)-(2.8), можно конструировать в таком виде:

$$\boldsymbol{E_1} = \frac{\lambda_1}{2}\sum_{i=1}^{m}\sum_{k=1}^{q} y_{ik}(1-y_{ik}) + \frac{\lambda_2}{2}\sum_{i=1}^{m}\left(\sum_{k=1}^{q} y_{ik} - 1\right)^2 + \quad (3.2)$$

$$+\frac{\lambda_3}{2}\left(\sum_{i=1}^{m}\sum_{k=1}^{q} y_{ik} - m\right)^2 + \frac{\lambda_4}{2}\sum_{k=1}^{q}\varphi^2\left(\sum_{i=1}^{m} y_{ik} - 1\right)$$

где $\lambda_0, \lambda_1, \lambda_2, \lambda_3, \lambda_4$ - положительные константы, а $\varphi(z) = z - |z|$ функция, обладающая свойством $\varphi^2(z) = 2z\varphi(z)$.

Первое слагаемое в (3.2) соответствует бинарности переменных (2.8), второе слагаемое соответствует ограничению (2.6), что каждая строка матрицы $\boldsymbol{Y}$ содержит не более одной единицы, третье слагаемое соответствует тому, что в матрице $\boldsymbol{Y}$ содержится ровно $m$ единиц, наконец, последнее слагаемое соответствует ограничению (2.7). Отсюда вытекает, что при выполнении ограничений (2.6)-(2.8), выражение (3.2) принимает свое минимальное, равное нулю, значение.

После суммирования выражений (3.1) и (3.2), и несложных преобразований, получим следующий вид энергетической функции нейронной сети:

$$\begin{aligned}\boldsymbol{E} = \boldsymbol{E_0} + \boldsymbol{E_1} = &-\frac{\lambda_0}{2}\sum_{i=1}^{m}\sum_{k=1}^{q}\sum_{p=1}^{m}\sum_{l=1}^{q} a_{ikpl} y_{ik} y_{pl} - \\ &-\frac{\lambda_1}{2}\sum_{i=1}^{m}\sum_{k=1}^{q}\sum_{p=1}^{m}\sum_{l=1}^{q}\delta_{ip}\delta_{kl} y_{ik} y_{pl} + \frac{\lambda_1}{2}\sum_{i=1}^{m}\sum_{k=1}^{q} y_{ik} + \\ &+\frac{\lambda_2}{2}\sum_{i=1}^{m}\sum_{k=1}^{q}\sum_{p=1}^{m}\sum_{l=1}^{q}\delta_{ip} y_{ik} y_{pl} - \lambda_2\sum_{i=1}^{m}\sum_{k=1}^{q} y_{ik} + \\ &+\frac{\lambda_2}{2}m + \frac{\lambda_3}{2}\sum_{i=1}^{m}\sum_{k=1}^{q}\sum_{p=1}^{m}\sum_{l=1}^{q} y_{ik} y_{pl} - \\ &-m\lambda_3\sum_{i=1}^{m}\sum_{k=1}^{q} y_{ik} + \frac{\lambda_3}{2}m^2 + \\ &+\lambda_4\sum_{k=1}^{q}\left(\sum_{i=1}^{m} y_{ik} - 1\right)\varphi\left(\sum_{i=1}^{m} y_{ik} - 1\right)\end{aligned} \quad (3.3)$$

где $\delta_{ip}$ - символ Кронекера.

**Этап 3.** Определение параметров сети.

Третий этап заключается в непосредственном определении параметров нейронной сети – матрицы синаптических связей $\boldsymbol{W}$ и вектора внешних смещений $\boldsymbol{B}$ - путем сопоставления сконструированной энергетической функции $\boldsymbol{E}$ с ее канонической формой $\boldsymbol{E_c}$, которая конструируется в следующем виде:

$$\begin{aligned}\boldsymbol{E_c} = &-\frac{1}{2}\sum_{i=1}^{m}\sum_{k=1}^{q}\sum_{p=1}^{m}\sum_{l=1}^{q} w_{ikpl} y_{ik} y_{pl} + \\ &+\sum_{i=1}^{m}\sum_{k=1}^{q} b_{ik} y_{ik}\end{aligned} \quad . \quad (3.4)$$

Сопоставляя выражения (3.4) и (3.5) и приравняв их линейных и квадратичных составляющих, находим параметры нейронной сети:

$$\begin{cases} w_{ikpl} = \lambda_0 a_{ikpl} + \lambda_1\delta_{ip}\delta_{kl} - \lambda_2\delta_{ip} - \lambda_3 \\ b_{ik} = \dfrac{\lambda_1}{2} - \lambda_2 - m\lambda_3 + \lambda_4 \end{cases}, \quad (3.5)$$

где $i, p = 1,..., m;\ k, l = 1,..., q$.

Отметим, что при определении параметров (3.5), в (3.3) слагаемые, не зависящие от состояния $y_{ik}$ нейронной сети, не были учтены.

Таким образом, построена нейронная сеть, параметры которой определены с точностью до постоянных коэффициентов. Вопрос определения

коэффициентов $\lambda_0, \lambda_1, \lambda_2, \lambda_3, \lambda_4$ требует отдельного исследования.

## 4. Алгоритм определения количества кластеров.

Отметим, что выбор подходящего количества кластеров является важным этапом кластерного анализа [19,22,23,24]. Априорно трудно определить, сколько кластеров представляет рассматриваемое множество. Здесь предлагается следующая стратегия: начиная с достаточно малого количества кластеров $q$, лицо, принимающее решение, должно поэтапно увеличивать количество кластеров до тех пор, пока некоторый критерий завершения не удовлетворен. С точки зрения перспективы оптимизации это означает, что, если решение соответствующей задачи оптимизации (2.5)-(2.8) не удовлетворительно, то лицо, принимающее решение, должно рассмотреть задачу (2.5)-(2.8) с $q+1$ кластерами и так далее. Это подразумевает, что лицо, принимающее решение, должно неоднократно решать задачу целочисленного программирования типа (2.5)-(2.8) с различными значениями $q$. Ниже проводится алгоритм пошагового вычисления кластеров.

Введем функцию $F(x)$, которая определяется соотношением:

$$F(x) = \frac{\sum_{k=1}^{q}\sum_{i=1}^{m}\sum_{p=1}^{m} d_{ip}x_{ik}x_{pk}}{\sum_{i=1}^{m}\sum_{k=1}^{q}\sum_{p=1}^{m}\sum_{\substack{l=1 \\ l\neq k}}^{q} d_{ip}x_{ik}x_{pl}},$$

где числитель соответствует первому слагаемому, а знаменатель второму слагаемому в (2.4).

*Шаг 1*. Задается допуск $\varepsilon > 0$. Положим $k = 2$ и решаем задачу (2.5)-(2.8). Пусть $F_2$ значение функций $F(x)$, соответствующее решению задачи (2.5)-(2.8).

*Шаг 2*. Положим $k = k+1$ и решаем задачи (2.5)-(2.8). Пусть $F_{k+1}$ значение функций $F(x)$, соответствующее решению задачи (2.5)-(2.8).

*Шаг 3*. Если

$$\frac{F_k - F_{k+1}}{F_2} < \varepsilon,\ k \geq 2,$$

то следует останавливать алгоритм, в противном случае полагать $k = k+1$ и перейти к *Шагу 2.*

Легко показать, что для всех $k$ выполняется условие $F_k \geq F_{k+1} > 0$. Таким образом, в результате получается убывающая последовательность $\{F_k\}$ и $F_k > 0$, для всех $k$, следовательно, после $k^*$ итерации критерий останова в *Шаге 3* будет удовлетворен.

Выбор допуска $\varepsilon > 0$ является очень существенным в описанном алгоритме. Большое значение $\varepsilon$ может привести к большим кластерам, объединяющие другие кластеры, и малое значение $\varepsilon$ может привести к появлению малых и искусственных кластеров. Выбор допуска $\varepsilon$ определяется экспериментальным путем.

## 5. Выбор представительных предложений и оценка резюмирования.

Следующим шагом после кластеризации является определение представительных предложений в каждом кластере. Представительность предложения определяется мерой близостью, вычисляемой между ними и соответствующим кластерным центроидом, т.е. чем меньше евклидово расстояние между предложением и соответствующим кластерным центроидом, тем это предложение считается более представительным. Прежде чем включить предложения в резюме, они ранжируются в порядке возрастания их мер близости к соответствующему кластерному центроиду. Текстовый документ обычно состоит из нескольких тем. Некоторые темы описываются многими предложениями и, следовательно, формируют главного содержания документа. Другие темы могут только быть кратко упомянуты, чтобы дополнить главную тематику. Следовательно, количества предложений в каждом кластере будут отличаться друг от друга. При этом количество выбираемых предложений из каждого кластера тоже будет разным. Такой подход позволяет в максимально возможной степени охватывать главного контента документа и избегать избыточности. В общем случае, количество предложений, включаемых в резюме, зависит от коэффициента сжатия. Коэффициент сжатия $\alpha_{comp}$ определяется отношением длин резюме и документа:

$$\alpha_{comp} = \frac{len(summ)}{len(doc)},$$

и является важным фактором, влияющим на качество резюме, где $len(summ), len(doc)$ - длина резюме и документа, соответственно. Поскольку при малом значении коэффициента сжатия, резюме будет более кратким, и основная часть информации будет потеряна. В то же время, при большом значении коэффициента сжатия, резюме будет обильным, однако оно будет содержать незначительных предложений. В работе [1] было показано, что если коэффициент сжатия находится в интервале $[0.05, 0.3]$, то результат резюмирования считается приемлемым.

Учитывая вышеизложенное, определим количество $N_k$ представительных предложений,

отобранных из каждого кластера $k$, вычисляемое следующей формулой:

$$N_k = \left| \frac{len(C_k) \cdot \alpha_{comp}}{len_{aver}} \right|, \quad k = 1,..., q,$$

где $len(C_k)$, $len_{aver} = \frac{len(doc)}{m}$ - длина кластера $C_k$ и средняя длина предложений в документе, соответственно, а $|\cdot|$ означает целую часть.

Для оценки результата резюмирования используем $F_1$ - критерии. Пусть $N_d^{rel}$ -количество релевантных предложений в документе, $N_s^{rel}$ - количество релевантных предложений в резюме, $N_s$ -количество предложений в резюме, $P$ - точность, $R$ -полнота. Тогда отсюда следует, что

$$P = \frac{N_s^{rel}}{N_s},$$

$$R = \frac{N_s^{rel}}{N_d^{rel}},$$

$$F_1 = \frac{2PR}{P + R}.$$

## Заключение.

В связи с ростом текстовых данных на WWW возникает потребность в автоматических методах резюмирования этих данных. Целью задачи автоматического резюмирования заключается в извлечении из текста нескольких обобщающих предложений, отражающих его главную тематику с сохранением минимальной избыточности.

С этой целью в настоящей работе приводится описание метода универсального резюмирования, основанного на кластеризации предложений. В отличие от традиционных методов кластеризации, в работе предлагается новый метод, математическая реализация которого опирается на задачу целочисленного программирования. Во избежание трудоемких вычислительных процедур предлагается нейросетевая реализация задачи целочисленного квадратичного программирования с булевыми переменными. Отметим, что выбор подходящего количества кластеров является одним из трудных задач в кластерном анализе. Априорно трудно определить, сколько кластеров представляет рассматриваемое множество. Для преодоления этих трудностей в работе предлагается алгоритм пошагового определения количества кластеров. Чтобы обеспечить широкий охват контента документа и во избежание избыточности в резюме, на каждом кластере, т.е. на каждом тематическом разделе определяются представительные предложения и их количества.

## Литература


1. Mani I., Maybury M.T. Advances in automated text summarization. Cambridge. MIT Press. 1999.
2. Salton G., Singhal A., Mitra M., Buckley C. Automatic text structuring and summarization //Information Processing and Management. 1997. V. 33. № 2.
3. Mitra M., Singhal A., Buckley C. Automatic text summarization by paragraph extraction //Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization. Madrid. Spain. 1997.
4. Kruengkrai C., Jaruskulchai C. Generic text summarization using local and global properties of sentences //IEEE/WIC International Conference on Web Intelligence (WI'03). Halifax. Canada. 2003.
5. Yeh J-Y., Ke H-R., Yang W-P., Meng I-H. Text summarization using a trainable summarizer and latent semantic analysis //Information Processing and Management. 2005. V. 41. № 1.
6. Goldstein J., Kantrowitz M., Mittal V., Carbonell J. Summarization text documents: sentence selection and evaluation metrics //Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99). Berkeley. USA. 1999.
7. Gong Y., Liu X. Generic text summarization using relevance measure and latent semantic analysis //Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. New Orleans. USA. 2001.
8. Hu P., He T., Ji D., Wang M. A study of Chinese text summarization using adaptive clustering of paragraphs //Proceedings of the 4th International Conference on Computer and Information Technology (CIT'04). Wuhan. China. 2004.
9. Shen D., Chen Z., Yang Q., Zeng H.J., Zhang B., Lu Y., Ma W.Y. Web-page classification through summarization //Proceedings of the 27th Annual International Conference on Research and Development in Information Retrieval. Sheffield. UK. 2004.
10. Delort J.-Y., Bouchon-Meunier B., Rifqi M. Enhanced web document summarization using hyperlinks //Proceedings of the 14th ACM Conference on Hypertext and Hypermedia. Nottingham. UK. 2003.
11. Luhn H.P. The automatic creation of literature abstracts //IBM Journal of Research and Development. 1958. V. 2. №2.
12. Banko M., Mittal V., Kantrowitz M., Goldstein J. Generating extraction-based summaries from hand-written summaries by aligning text spans //Proceedings of the 4th Conference of the Pacific Association for Computational Linguistics (PACLING'99). Waterloo. Canada. 1999.

13. Grabmeier J., Rudolph A. Techniques of cluster algorithms in data mining //Data Mining and Knowledge Discovery. 2002. V. 6. № 4.
14. Halkidi M., Batistakis Y., Vazirgiannis M. On clustering validation techniques //Journal of Intelligent Systems. 2001. V. 17. № 2-3.
15. Jain A.K., Murty M.N., Flynn P.J. Data clustering: a review //ACM Computing Surveys. 1999. V. 31. № 3.
16. Jiang D., Tang C., Zhang A. Cluster analysis for gene expression data: a survey //IEEE Transactions on Knowledge and Data Engineering. 2004. V. 16. № 11.
17. Mangasarian O.L. Mathematical programming in data mining //Data Mining and Knowledge Discovery. 1997. V. 1. № 2.
18. Bradley P.S., Fayyad U.M., Mangasarian O.L. Mathematical programming for data mining: formulations and challenges //INFORMS Journal on Computing. 1999. V. 11. № 3.
19. Bagirov A.M., Ferguson B., Ivkovic S., Saunders G., Yearwood J. New algorithms for multi-class diagnosis using tumor gene expression signature //Bioinformatics. 2003. V. 19. № 14.
20. Алгулиев Р.М., Алыгулиев Р.М., Алекперов Р.К. Подход к оптимальному назначению заданий в распределенной системе //Проблемы управления и информатики. 2004. №5.
21. Нейроматематика. Книга 6. Учебное пособие для вузов. /Под общей редакцией А.И. Галушкина. М.: ИПРЖР. 2002.
22. Kim D.-W., Lee K.H., Lee D. On cluster validity index for estimation of the optimal number of fuzzy clusters //Pattern Recognition. 2004. V. 37. № 10.
23. Kothari R., Pitts D. On finding the number of clusters //Pattern Recognition Letters. 1999. V. 20. № 4.
24. Sun H., Wang S., Jiang Q. FCM-based model selection algorithms for determining the number of clusters //Pattern Recognition. 2004. V. 37. № 10.

**MATHEMATICAL PROGRAMMING IN TEXT MINING**

R.M.Alguliev, R.M.Aliguliyev
Institute of Information Technology of National Academy of Sciences of Azerbaijan, Baku
rasim@science.az , a.ramiz@science.az

With an increase in the volume of text information, accessible on WWW, became everything more and more necessary for the users to employ the automated instrument means, in order to find, to extract, to filter and to estimate the desired information. One of such means is automatic summarization of text documents. For this purpose in this article is proposed the summarization method of text documents, based on clustering of sentences. In contrast to the traditional methods the new method of clustering is proposed, mathematical realization of which rests on the integer programming problem. In the article to avoid labor-consuming computational procedures the neuronet implementation of the integer programming problem is suggested.