УДК 004.912 + 004.738.52

И.С. Мисуно, Д.А. Рачковский, С.В. Слипченко, А.М. Соколов

ПОИСК ТЕКСТОВОЙ ИНФОРМАЦИИ С ПОМОЩЬЮ ВЕКТОРНЫХ ПРЕДСТАВЛЕНИЙ

Рассматриваются векторные представления текстовой информации на основе частот встречаемости слов и их последовательности, а также распределенные векторные представления, учитывающие семантическую близость слов. Представлены архитектура программной системы, разработанной для экспериментального исследования алгоритмов поиска с помощью векторных представлений, и прототип системы поиска на базе веб-серверного подхода. Приводятся результаты сравнительного тестирования.

Введение

Интеллектуализация систем поиска текстовой информации требует учета ее смыслового содержания. Классические проблемы поиска документов – это синонимия (одно и то же понятие может быть выражено с использованием разных терминов – синонимов) и полисемия (один и тот же термин может иметь различные значения в различных контекстах). Традиционно эти проблемы решают путем расширения запроса семантически близкими словами из тезаурусов или из документов, возвращенных системой в ответ на запрос и помеченных пользователем как релевантные [1].

Ручное конструирование лингвистических ресурсов типа тезаурусов и онтологий (например, WordNet [2]) очень трудоемко. Поэтому привлекательны автоматические методы получения и представления семантической информации. Ряд таких методов основан на использовании векторных моделей [3 – 6], где информация о совместной встречаемости слов извлекается из больших коллекций (корпусов) текстов и фиксируется в так называемых семантических или контекстных векторах. Сходство контекстных векторов, вычисляемое как скалярное произведение или расстояние, принимают за меру семантической близости слов. Из контекстных векторов формируются представления документов и запросов, которые отражают не только набор составляющих их слов, но и их семантику (смысл). Сходство таких представлений позволяет системе найти документы, которые могут и не содержать слов запроса, но соответствуют запрашиваемой теме.

Векторные подходы близки к методам распределенного представления информации, развиваемым нами в рамках парадигмы ассоциативно-проективных нейронных сетей (АПНС) [7]. Получение распределенных семантических представлений текстовой информации, разработка систем и сравнительное исследование таких представлений в задаче поиска текстовых документов позволят расширить спектр применений АПНС и создать более эффективные поисковые системы вследствие учета семантики.

1. Векторные и распределенные представления текстовой информации

1.1. Векторные представления. Современные методы векторного представления текстовой информации являются развитием моделей векторных пространств VSM (Vector Space Models), предложенных в [8]. В этих моделях компоненты текстов, такие как слова, словосочетания, фрагменты текстов, целые документы, представлены многомерными векторами. Элементы векторов есть значения некоторой функции от частоты совместной встречаемости компонентов текстов и их контекстов. Степень сходства между компонентами текстов q и d определяется величиной сходства между их векторами $\mathbf{q}$ и $\mathbf{d}$. Обычно используется некоторая монотонная функция угла между векторами, например косинус $\cos(\mathbf{q},\mathbf{d})$, который для нормированных векторов совпадает со скалярным произведением $(\mathbf{q},\mathbf{d})$.

Рассмотрим в качестве компонентов текста слова, а в качестве контекстов – документы. Введем следующие обозначения:
w – количество уникальных слов в коллекции документов;

t – общее количество документов в коллекции;
tf_{ij} – частота встречаемости слова w_i в документе d_j;
df_i – количество документов, содержащих слово w_i;
$idf_i = \log(t/df_i)$ – обратная частота документов, содержащих слово w_i.

Построим матрицу **A (**размерностью $w \times t$) встречаемости слов в документах коллекции. Элементы матрицы a_{ij} – "веса" слов в документах. Часто полагают $a_{ij}=tf_{ij} \cdot idf_i$.

Ряд векторных представлений компонентов текстов (документов и запросов) можно рассматривать как проекции (трансформацию) векторов документов $\boldsymbol{d}$ или векторов запросов $\boldsymbol{q}$ в другое пространство. Трансформация осуществляется с помощью некоторой матрицы **P**, а сходство векторов в новом пространстве также определяется как скалярное произведение

$$\mathrm{sim}_P(\mathbf{d},\mathbf{q}) = (\mathbf{P}^T\mathbf{d})(\mathbf{P}^T\mathbf{q}) = \mathbf{d}^T\mathbf{P}\mathbf{P}^T\mathbf{q}. \quad (1)$$

Оригинальная модель VSM [1, 8] основана на предположении, что смысл всего документа выражается смыслом используемых в нем слов. Документы представлены векторами – столбцами матрицы **A**, запросы – векторами, составленными из слов запроса. Таким образом, для VSM модели **P**=**I**, а сходство вычисляется как

$$\mathrm{sim}_{VSM}(\mathbf{d},\mathbf{q}) = (\mathbf{I}^T\mathbf{d})(\mathbf{I}^T\mathbf{q}) = \mathbf{d}^T\mathbf{I}\mathbf{I}^T\mathbf{q} = \mathbf{d}^T\mathbf{q}. \quad (2)$$

Критика VSM (например, [4]) основывается на том, что в качестве ортогонального базиса векторного пространства используются слова, которые на самом деле часто семантически зависимы. Для преодоления этого недостатка была предложена альтернатива в виде метода GVSM (Generalized Vector Space Model – обобщенной модели векторного пространства [9]. Она основана на том, что **A** рассматривается не только как представление документов с помощью слов, как в VSM, но и как представление слов с помощью документов. Каждая строка **A** отражает распределение слов по документам. Два слова считают семантически связанными друг с другом, если похожи их распределения по документам. Сходство запроса и документа в методе GVSM можно представить как (**P** = **A**)

$$\mathrm{sim}_{GVSM}(\mathbf{d},\mathbf{q}) = (\mathbf{A}^T\mathbf{d})(\mathbf{A}^T\mathbf{q}) = \mathbf{d}^T\mathbf{A}\mathbf{A}^T\mathbf{q}. \quad (3)$$

В схеме LSI (Latent Semantic Indexing – латентное семантическое индексирование [3]) полагается, что пространство с уменьшенной размерностью, содержащее наиболее значимые линейные комбинации измерений первоначального пространства, дает лучший базис для представления содержания документов. В основе LSI лежит процедура SVD (singular value decomposition – декомпозиция по сингулярным значениям), которая позволяет представить **A** как

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T, \quad (4)$$

где $\mathbf{D}=\mathrm{diag}(\sigma_1,...,\sigma_r)$ – диагональная матрица сингулярных значений, $\mathbf{U}=(\boldsymbol{u}_1,...,\boldsymbol{u}_r)$ и $\mathbf{V}=(\boldsymbol{v}_1,...,\boldsymbol{v}_r)$ – матрицы $w \times r$ и $t \times r$ с ортонормированными столбцами левых и правых сингулярных векторов соответственно.

В LSI в качестве нового базиса для редуцированного векторного пространства выбираются k столбцов $\mathbf{U}_k$ (ортогональных сингулярных векторов), соответствующих k наибольшим сингулярным значениям. Столбцы $\mathbf{U}_k$ являются линейными комбинациями документов. В [3] представлениями слов предлагается считать соответствующие строки матрицы $\mathbf{U}\mathbf{\Sigma}_k$. Сходство компонентов текста находится после их проецирования в k-мерное пространство векторов-столбцов матрицы $\mathbf{U}_k \equiv \mathbf{I}_k\mathbf{U}^T$ ($\mathbf{P} = \mathbf{U}\mathbf{I}_k$,) как

$$\mathrm{sim}_{LSI}(\mathbf{d}, \mathbf{q}) = (\mathbf{I}_k\mathbf{U}^T\mathbf{d})\,(\mathbf{I}_k\mathbf{U}^T\mathbf{q}) = \mathbf{d}^T\mathbf{U}\mathbf{I}_k{}^2\mathbf{U}^T\mathbf{q} = \mathbf{d}^T\mathbf{U}\mathbf{I}_k\mathbf{U}^T\mathbf{q}. \quad (5)$$

1.2. Распределенные представления. Для больших коллекций документов полноразмерная матрица **A** оказывается слишком большой для обработки и даже хранения в оперативной памяти компьютера. Один из подходов к решению этой проблемы основан на применении распределенных представлений. В отличие от локальных представлений, где каждый информационный компонент данных представлен элементом кодвектора, т.е. одним из ортогональных измерений входного пространства, в распределенных представлениях каждому компоненту данных соответствует N-мерный вектор со случайно сгенерированными элементами (см. [7]). В многомерном пространстве «почти ортогональных» измерений гораздо больше, чем ортогональных. Для ряда распределений элементов случайные векторы близки к ортогональным и достаточно хорошо аппрокси-

мируют исходный базис. Если в качестве текстовых компонентов рассматривать документы и $N<t$, то размерность $w \times N$ новой матрицы **S** меньше исходной $w \times t$:

$$\mathbf{S} = \mathbf{AR}, \quad (6)$$

где **R** – матрица случайных векторов $t \times N$. Попарные углы между строками **S** (представления слов) аппроксимируют углы между строками **A** (см. также [10 – 13].

Матрицу **S** можно сформировать без **A** в процессе последовательного пословного прохода корпуса текстов (метод случайного индексирования RI – Random Indexing, [5]). При этом к строкам $\boldsymbol{s}_i$ матрицы **S** прибавляются индексные векторы документов $\boldsymbol{r}_j$ (строки матрицы **R**), когда слово i встречается в документе j:

$$\boldsymbol{s}_i = \sum_{j:w(i)\in t(j)} \boldsymbol{r}_j \, a_{ij}, \quad (7)$$

где a_{ij} – частота встречаемости слова i в документе j. Вычислительная сложность формирования **S** сравнима со сложностью формирования **A** и равна $O(LM)$ или $O(L)$, где L – количество слов в корпусе; M – количество ненулевых элементов в $\boldsymbol{r}$ (обычно M постоянно и составляет малую долю N). Аналогично можно сформировать контекстные векторы, в которых в качестве контекстов используются не документы, а слова в некоторой окрестности целевого слова [6, 14, 15].

Получаемые таким образом распределенные контекстные векторы имеют действительные элементы, что усложняет их обработку. Обработка векторов с дискретными (бинарными или тернарными) элементами намного более эффективна, а их формат соответствует формату АПНС [7]. Поэтому предлагается использовать дискретные контекстные векторы, которые могут быть получены применением пороговых операций к элементам действительных векторов – как "исходной локальной" матрицы **A**, так и матрицы распределенных контекстных векторов **S**. Пороговая операция $\tau(x, \theta^+, \theta^-)$ над скаляром x дает скаляр y:

$$\begin{aligned} &y = 1, \text{ если } x > \theta^+ \geq 0; \\ &y = -1, \text{ если } x < \theta^- \leq 0, \\ &\text{иначе } y = 0. \end{aligned} \quad (8)$$

Такая операция выполняется над всеми элементами контекстных векторов, образующих матрицы **A** или **S**. Значения элементов вектора порогов $\boldsymbol{\theta}$ могут быть одинаковыми для всех элементов вектора (например, 0 при тернаризации с нулевым порогом или подобранное значение порога для обеспечения заданного числа ненулевых элементов), или индивидуальными (например, зависящими от среднего значения $E(x_i)$). Регулирование плотности ненулевых элементов может также осуществляться с использованием процедур контекстно-зависимого прореживания CDT [16].

Дискретные контекстные векторы могут конструироваться и непосредственно из частей индексных векторов:

$$\boldsymbol{c}_i = \sum_{j:w(i)\in t(j)} \boldsymbol{q}_j \, (\boldsymbol{r}_j , f(\mathrm{j})), \quad (9)$$

где $t(j)$ – контексты, в которых встречается слово; $\boldsymbol{r}_j$ – дискретные индексные векторы, имеющие по M ненулевых элементов; $\boldsymbol{q}_j(\boldsymbol{r}_j,n)$ – вектор с n ненулевыми элементами, полученный из $\mathbf{r}_\mathrm{j}$ с помощью процедуры CDT [16]; $f(j)$ – функция, регулирующая представительство $\boldsymbol{r}_\mathrm{j}$ в $\boldsymbol{c}_i$. Например, её можно определить как

$$\begin{aligned} &f(j) = \mathrm{ent}(\varphi(j)) + 1 \text{ при } y < \varphi(j) - \mathrm{ent}(\varphi(j)), \\ &f(j) = \mathrm{ent}(\varphi(j)) \text{ иначе.} \end{aligned} \quad (10)$$

Здесь $\varphi(j)$ – действительная функция "важности" контекста; y – псевдослучайное число из диапазона (0,1), фиксированное для данного j. Например, $\varphi(j) = M/L$ (где L – общее количество контекстов) или с нелинейным учетом частот встречаемости слов

$$\varphi(j) = M(1+\log a_{ij}) \, idf_j / \Sigma_\mathrm{i} \, \varphi(i). \quad (11)$$

На заключительном этапе к $\boldsymbol{c}$ применяется пороговая операция $\boldsymbol{c}_i = \tau(\boldsymbol{c}_i,\boldsymbol{\theta})$. При $\boldsymbol{\theta} = 0$ суммирование в (9) эквивалентно дизъюнкции.

Векторы документов и запроса для поиска формируются суммированием контекстных векторов слов. Возможно взвешивание и нормирование, а также формирование дискретных векторов документов с помощью операций, аналогичных (8) и CDT [16]. Сходство документа и запроса находится как (нормированное) скалярное произведение их контекстных векторов.

2. Прототип поисковой системы

Для разработки и тестирования алгоритмов поиска на основе векторных представлений создана экспериментальная поисковая система. При ее разработке особое внимание уделялось обеспечению гибкости архитектуры, позволяющей легко добав-

лять, модифицировать и сравнивать различные варианты алгоритмов. На основе объектно-ориентированного подхода разработаны библиотека классов C++ и набор модулей, выполняющих последовательность обработки текстовой информации.

Система состоит из модулей формирования контекстных векторов и собственно модуля поиска. Обобщенная архитектура приведена на рис. 1.

Процесс формирования контекстных векторов можно разбить на 4 этапа: (I) индексирование корпуса, (II) формирование матриц *слова×документы* и/или *слова×слова*, (III) формирование контекстных векторов слов и матрицы скалярных произведений контекстных векторов слов. По сформированной матрице производится поиск либо в режиме тестирования (IV), либо в режиме он-лайн поиска.

Индексирование корпуса включает формирование словаря и индексированного файла корпуса, в котором слова заменены на их индексы. Словарь может быть подвергнут фильтрации: удалению малоинформативных слов или слишком редких/частых слов (*and*, *or*, …); обрезке слов по длине (основой слова считаются первые 4,6,8… букв) или выделению основ (например, библиотекой *morph* А. Курсина).

На основе словаря и индексированного корпуса формируются матрицы *слова×документы* и/или *слова×слова*, которые в последующем используются для формирования контекстных векторов слов (см. разд. 1.2). Из контекстных векторов слов, с использованием матрицы *слова×документы* взвешенным суммированием и нормированием строятся векторы документов.

Несмотря на то что векторы документов в большинстве случаев разреженные, нахождение сходства вектора запроса с векторами всех документов является трудоемкой задачей ($t>>w$). Для ускорения предварительно вычисляется матрица скалярных произведений каждого слова с каждым документом. Тогда сходство запроса с каждым из документов можно получить как сумму ее строк, соответствующих словам запроса (с выбранным взвешиванием и нормированием). Матрица не является разреженной и требует большого объема дискового пространства для хранения. Однако создание такой матрицы и организация эффективного доступа к ее строкам позволяют значительно ускорить последующий процесс поиска.

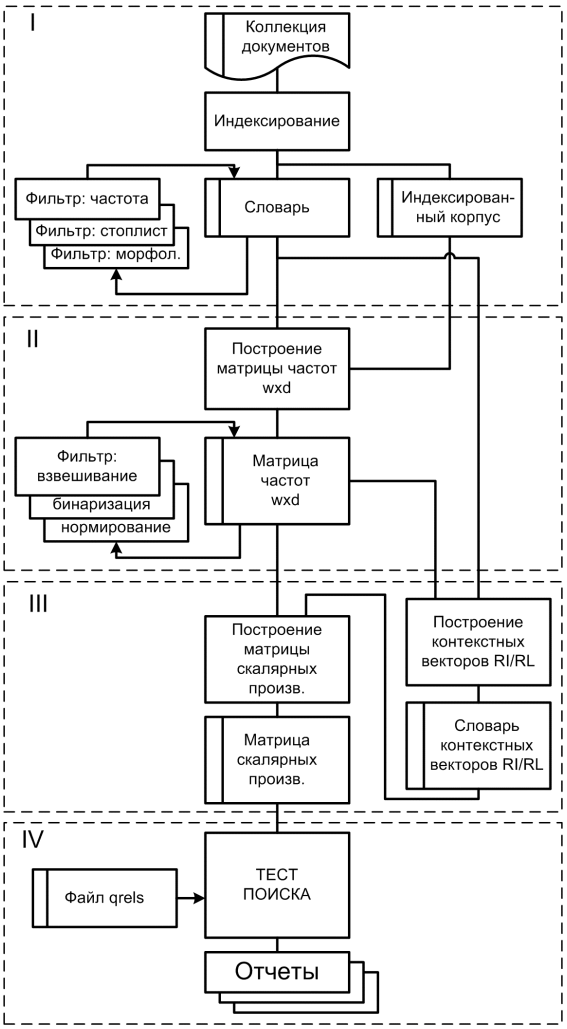

Рис. 1. Архитектура поисковой системы

Реализован также прототип системы поиска документов в режиме взаимодействия с пользователем (режим он-лайн). Эта система выполняет поиск по предварительно проиндексированным коллекциям документов. Система работает под управлением веб-сервера MS IIS (Internet Information Services), интерфейс пользователя реализован на Dynamic HTML с отображением в веб-браузере.

Работа с контекстными векторами на серверной стороне осуществляется с помощью технологии программирования ASP (Active Server Pages) и JScript, а также COM-объектов, реализованных на C++. COM-объекты осуществляют чтение словаря, индексированный поиск вектора по

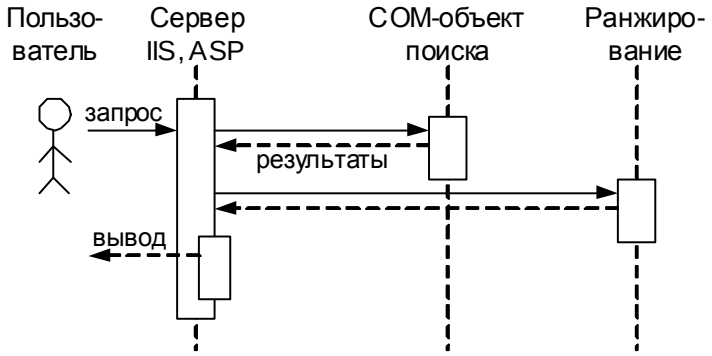

Рис. 2. Диаграмма последовательности вызовов при работе системы поиска

слову, вычисление контекстного вектора нескольких слов и коэффициента сходства между двумя текстовыми фрагментами, поиск наиболее близких запросу документов, произвольный доступ к документам в коллекции. Диаграмма последовательности вызовов, осуществляемых при взаимодействии пользователя с системой поиска, приведена на рис. 2.

Пользователь вводит запрос в окно веб-браузера, который обращается к веб-серверу MS IIS. Сервер выполняет программу на языке ASP/JScript, обрабатывающую запрос с помощью вызовов COM-объектов. Результаты поиска в виде HTML передаются пользователю, который просматривает их в браузере. Внешний вид окна веб-браузера при работе с системой поиска представлен на рис. 3.

3. Экспериментальное исследование

3.1. Коллекции документов. В экспериментах по поиску использовались коллекции MEDLARS, Cranfield, Time Magazine [17], которые предназначены для исследования поисковых систем и имеют сформированные экспертами списки запросов и документов, соответствующих каждому запросу. Это позволяет численно оценивать качество поиска по стандартным мерам полнота-точность (см. разд. 3.2).

MEDLARS (*Medical Literature Analysis and Retrieval System*) – коллекция аннотаций к статьям биомедицинской направленности. Содержит 1033 документа средней длиной порядка 50 слов. Объем словаря составляет порядка 12 тыс. слов. Заданы 30 поисковых запросов, содержащих в среднем

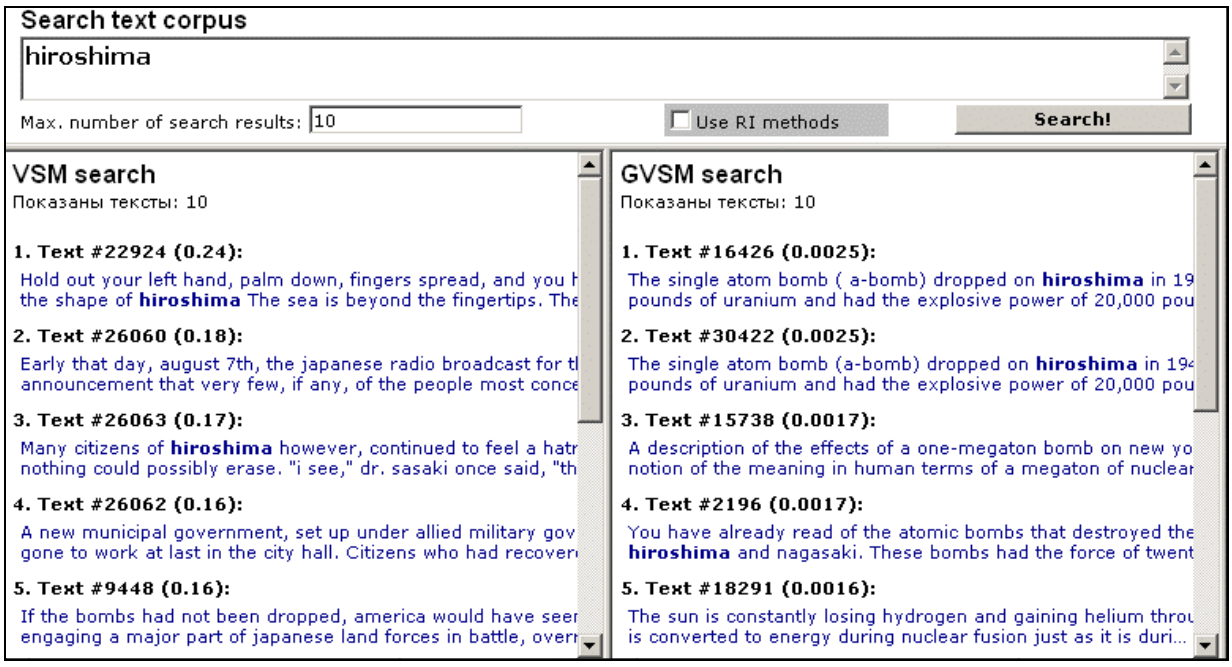

Рис. 3. Пример работы макета программы поиска текстов

по 10 слов, каждому запросу в среднем соответствуют около 20 документов.

Cranfield (Cranfield University UK) – коллекция аннотаций к статьям по аэродинамике. Содержит порядка 1400 документов и 225 запросов.

Time Magazine – коллекция статей из «Time Magazine». Содержит 425 документов и 83 запроса.

Кроме того, использовалась коллекция *TASA* (*Touchstone Applied Sciences Association*) [18] – тексты по литературе, биологии, экономике, промышленности, науке, искусству, социологии и бизнесу, рекомендованные для чтения в колледжах США. Более чем 37 600 текстов, каждый из которых состоит примерно из 166 слов из словаря в 94 000 слов, в сумме содержат более чем 10 млн слов. Для этой коллекции нет списка запросов с ответами.

3.2. Методика оценки результатов поиска. Для численной оценки качества работы систем поиска применялась средняя (интерполированная) точность (11-pt average precision) [8]. Определим полноту $recall=d/d_1$, точность $precision=d/d_2$, где d – количество найденных релевантных запросу документов; d_1 – количество релевантных запросу документов; d_2 – общее количество возвращенных по запросу документов. Методика вычисления 11-pt average precision включает в себя следующие этапы:

(i) для каждого запроса получить ранжированный список документов и для каждого J=1,..., d_2 из этого списка вычислить значение точности и полноты;

(ii) в каждом из 10 интервалов полноты между 0%,10%,…,100% выбрать наибольшее значение точности в качестве *репрезентативного* значения левой границы интервала. Для полноты 100% репрезентативной точностью будет либо точное значение, если оно существует, либо ближайшее имеющееся. Если интервал пуст, «репрезентативная» точность равна нулю;

(iii) выполнить интерполяцию – в каждом интервале в качестве репрезентативной точки выбрать максимальное значение точности в данном интервале или в интервале с большим значением полноты;

(iv) построить (в соответствии с пп. i,ii,iii) индивидуальные характеристики для всех запросов, после чего усреднить репрезентативные значения в точках с одинаковой полнотой. Это и есть результирующая 11-точечная характеристика системы поиска;

(v) усреднить все 11 значений и получить число "11-pt average precision".

3.3. Поиск текстовой информации. Ниже приведены примеры поиска с учетом семантики слов, выполненного на коллекции TASA.

Пример 1. Запрос: «hiroshima». Возвращенные в результате VSM-поиска документы содержат искомое слово, в то время как тексты, возвращенные другими методами, не содержат слова “hiroshima”, хотя в них речь идет об атомной бомбардировке Хиросимы и Нагасаки 1945 года и их последствиях.

Пример 2. Запрос: «vessel» (судно, посудина, кровеносный сосуд). Возвращенные в результате поиска по методам GVSM, RI и RL тексты используют это слово в разных значениях. Некоторые возвращенные тексты не содержат «vessel», но в них говорится о кровеносной системе и давлении крови.

Пример 3. Запрос: "america". Результаты поиска содержат несколько текстов, в которых это слово не встречается, но речь идет о гражданской войне в Америке 1830 г., упоминается Thomas Jefferson и т.д.

Приведенные примеры показывают, что поиск при учете семантической связи слов посредством использования контекстных векторов позволяет найти тексты на тему запроса, не содержащие слов запроса. В ряде случаев это также улучшает качество поиска.

Средняя точность (см. разд. 3.2) поиска в разных коллекциях приведена в табл. 1, а зависимость средней точности от полноты для разных методов на коллекции Medlars – на рис. 4. Применялись нормировки матрицы частот **A** в позиционной нотации SMART [8]: *l* означает логарифмирование частоты (1+log *tf*); *t* – умножение на *idf*; *c* – нормирование векторов документов (строк матрицы **A**) к L2; *n* – отсутствие соответствующего взвешивания или нормировки. Например, nnn означает, что никаких действий ни с матрицей, ни с векторами не осуществлялось; ltc – использованы все три процедуры. В экспериментах применялся

Таблица 1. **Средняя точность поиска в коллекциях MED, TIME, CRAN**

	MED					TIME					CRAN				
	VSM	GVSM	CONTEXT			VSM	GVSM	CONTEXT			VSM	GVSM	CONTEXT		
tr=8	dotp	dotp	dotp	L1	L2	dotp	dotp	dotp	L1	L2	dotp	dotp	dotp	L1	L2
nnn	0.45	0.54	0.23	0.55	0.54	0.42	0.39	0.14	0.36	0.48	0.26	0.21	0.03	0.27	0.28
ntn	0.51	0.68	0.36	0.66	0.67	0.58	0.62	0.20	0.55	0.64	0.35	—	0.08	—	0.42
lnn	0.50	0.53	0.20	0.56	0.56	0.51	0.46	0.10	0.49	0.53	0.34	0.25	0.03	0.28	0.29
ltn	0.54	**0.69**	0.32	0.66	**0.68**	**0.62**	**0.67**	0.16	0.67	**0.69**	0.40	**0.42**	0.06	0.44	**0.46**
nnc	0.50	0.55	0.26	0.57	0.55	0.57	0.43	0.14	0.47	0.52	0.37	0.23	0.04	0.27	0.28
ntc	0.54	0.69	0.42	0.67	0.67	0.68	0.66	0.25	0.63	0.67	0.41	0.41	0.08	0.41	0.43
lnc	0.53	0.54	0.23	0.56	0.56	0.64	0.45	0.11	0.52	0.55	0.40	0.26	0.03	0.29	0.29
ltc	**0.55**	**0.70**	0.37	0.67	**0.69**	**0.68**	**0.66**	0.18	0.65	**0.69**	**0.42**	**0.42**	0.06	0.44	**0.45**

единственный способ морфологической обработки – выделение первых 8 букв слова в качестве базовой словоформы (*tr*=8).

Контекстные векторы RI формировались по (7) при *N*=2000, количестве ненулевых элементов в индексных векторах *n*=*p*=5 (*p* и *n*, соответственно плюс и минус единиц), а векторы документов – суммированием полученных контекстных векторов. Результаты по VSM/GVSM получены нахождением сходства запроса и документа скалярным произведением их векторов (dotp), а по контекстным векторам (CONTEXT) – скалярным произведением (dotp) с возможным нормированием контекстных векторов по L1, L2. Нотация lt2, lt3 означает, что к контекстным векторам слов применялась, соответственно, бинаризация и тернаризация (8). Результаты с дискретизацией контекстных векторов приведены в табл 2.

Поиск по GVSM превосходит VSM (улучшение до 20%) для коллекции MED, показывая сравнимые результаты на TIME и CRAN. Применение представлений, учитывающих семантику посредством контекстных векторов, дает результаты на уровне GVSM (или на несколько процентов выше) на всех трех коллекциях. Применение контекстных представлений с дискретными элементами дает результаты, сравнимые с контекстными представлениями с действительными элементами, но обеспечивает большие возможности повышения эффективности обработки вследствие сокращения объема памяти и упрощения применяемых операций.

Кривые средней точности поиска по методам GVSM-ltc и RI-ltc с действительными векторами находятся рядом и выше кривых других методов. Кривая средней точности поиска по контекстным векторам RI с дискретизацией проходит немного ниже, но также над кривыми других методов при полноте больше 15%.

Исследовался также поиск с использованием q-граммного представления тек-

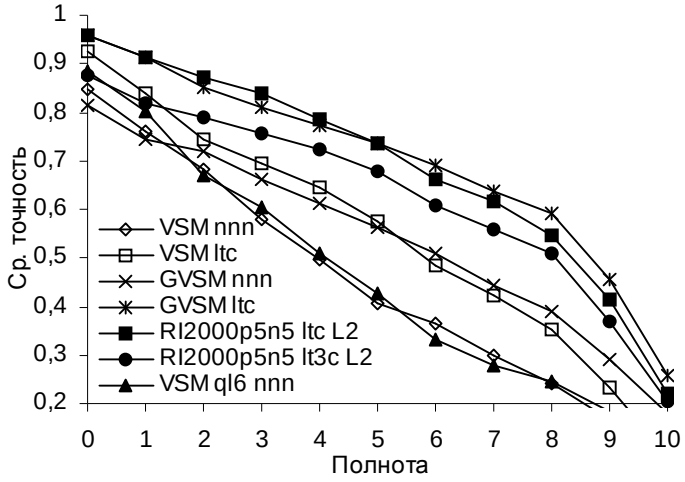

Рис. 4. Зависимость средней точности поиска от полноты для Medlars

Таблица 2. **Средняя точность поиска с дискретными контекстными векторами**

	MED			TIME			CRAN		
	CONTEXT			CONTEXT			CONTEXT		
tr=8	dotp	L1	L2	dotp	L1	L2	dotp	L1	L2
lt2n	0.05	0.60	0.61	0.07	0.61	**0.70**	0.02	0.25	**0.43**
lt2c	0.08	0.52	0.61	0.08	0.63	0.64	0.02	0.31	0.41
lt3n	0.21	0.61	**0.63**	0.09	0.65	**0.70**	0.03	0.38	0.40
lt3c	0.32	0.61	**0.63**	0.13	0.66	0.66	0.07	0.44	**0.45**

стовой информации. При таком представлении в качестве признаков используются последовательности из q элементов алфавита. В отличие от обсуждавшихся выше методов поиска VSM, GVSM, RI это позволяет учесть не только наличие, но и последовательность признаков (например, слов в текстах).

При использовании в качестве элементов алфавита слов, при q=1 q-граммное представление является аналогом представления bag-of-words (традиционный VSM). Использовались комбинированные q-граммные представления текстов, где вектор признаков содержал q-граммы длиной от 1 до q_{max} с разными весами. Для схемы 1 взвешивание не применялось, для схемы 2 q-граммы длиной $q \in \{1, q_{max}\}$ входили с весом $w_q = q_{max} - q + 1$, а для схемы 3 – с весом $w_q = q$. В табл 3 приведены результаты VSM-поиска в трех коллекциях по ненормированным (nnn) q-граммным векторам, полученным для *tr*=8.

Результаты показывают, что добавление в представления текстов последовательностей слов позволяет несколько улучшить результаты поиска (на 6–15% при q_{max}=2,3) по сравнению с VSM-nnn без учета последовательности. Наилучший результат наблюдается, когда последовательности большей длины имеют больший вес в векторных представлениях (столбцы qwc3 для всех коллекций).

Заключение

Проведено исследование векторных и распределенных представлений слов в задаче поиска текстовых документов. Использовались как "традиционные" представления (векторы, составленные из слов текстового фрагмента – запроса или документа), так и представления, отражающие семантическую близость слов и текстов (основанные на контекстных векторах).

Исследования показали, что применение распределенных представлений позволяет сократить размерность контекстных векторов и обойти сложности конструирования и обработки полной матрицы слова-контексты, размер которой может быть очень велик для больших текстовых массивов. Использование бинарных и тернарных контекстных векторов позволяет значительно сократить затраты памяти и вычислительных ресурсов на их хранение и обработку на обычных компьютерах. Такие распределенные представления совместимы с форматом данных в ассоциативно-проективных нейронных сетях [7]. Это позволяет

Таблица 3. **Средняя точность поиска по q-грамным векторам**

	MED				TIME				CRAN			
q_{max}	словарь	qwc1	qwc2	qwc3	словарь	qwc1	qwc2	qwc3	словарь	qwc1	qwc2	qwc3
1	10144	0,449	0,449	0,449	18247	0,417	0,417	0,417	6715	0,261	0,261	0,261
2	77329	**0,466**	0,454	**0,477**	129186	0,435	0,421	**0,447**	81883	**0,296**	0,272	**0,306**
3	157895	0,465	0,460	0,460	252613	**0,437**	0,425	0,460	194665	0,300	0,278	0,280
4	241687	0,461	0,461	0,445	378283	0,436	0,432	0,459	320562	0,298	0,285	0,242
5	327049	0,461	**0,461**	0,430	504914	0,436	**0,433**	0,449	452472	0,295	**0,289**	0,210
6	413618	0,459	0,461	0,429	632157	0,436	0,433	0,437	588197	0,289	0,291	0,198
7	501279	0,456	0,459	0,429	759907	0,436	0,433	0,435	726833	0,285	0,291	0,184
10	770557	0,450	0,457	0,431	1145820	0,436	0,433	0,434	1156554	0,275	0,292	0,176
15	1240027	0,446	—	—	1797585	0,441	—	—	1908175	0,256	—	—

использовать методы, предложенные для обработки таких представлений в АПНС, пополнить арсенал АПНС семантическими представлениями, а также обеспечить эффективную аппаратную поддержку в виде специализированных вычислителей и нейрокомпьютеров [19].

Разреженные варианты представлений с малым количеством ненулевых элементов являются нейробиологически релевантными и дают возможность дальнейшего повышения эффективности обработки (например, путем использования процедур, построенных на обратном индексе). Рост эффективности сильно зависит от формы представления информации, применяемых алгоритмов и аппаратных средств. Для обычных компьютеров экономия памяти и повышение быстродействия достигается за счет использования 1–2 бит памяти для хранения содержимого элемента дискретного вектора вместо 64 бит для действительного. При эффективной реализации специализированными аппаратными средствами логических битовых операций, требующихся для дискретных векторов, возможно дополнительное повышение быстродействия по сравнению с арифметическими операциями с плавающей точкой, требующихся для действительных векторов.

Разработан и реализован ряд прототипов систем формирования векторных представлений текстовой информации, а также поисковых систем, основанных на них. Результаты, полученные с помощью распределенных представлений, находятся на уровне результатов локальных представлений, но превосходят их по эффективности и масштабируемости. Использование разработанных представлений текстовой информации показало улучшение результатов до 20% по сравнению с обычным VSM-поиском по словам запроса вследствие применения контекстных векторов слов и документов.

В описанных схемах учитывалась информация об относительном расположении слов. Это лишь малая часть структурной информации, доступной в больших коллекциях текстов. Для более полного выявления семантики текста важно использовать и другие структурные отношения, присущие естественному языку. Выбранный метод построения поисковой системы и гибкость разработанной архитектуры позволяют в дальнейшем перейти к более глубокому смысловому анализу в процессе поиска (см. [20]).

Таким образом, проведенные исследования и полученные результаты создают предпосылки и открывают перспективы для создания новых информационных технологий и прикладных систем поиска информации с элементами учета семантики. Построение такого рода систем предполагается осуществлять на основании компонентно-ориентированного подхода [21].

1. *Grossman D., Frieder O.* Information Retrieval: Algorithms and Heuristics. – Boston: Kluwer, 1998. – 255 p.
2. *Miller G.* Wordnet: a lexical database for english // Communications of the ACM. – 1995. – № 11. – P. 39-41.
3. *Indexing by latent* semantic analisys / S. Deerwester, S. Dumais, G. Furnas, T. Landauer, R. Harshman. – J. of American Society for Information Sciences. – 1990. – № 1(6). – P. 391-407.
4. *Translingual information* retrieval: learning from bilingual corpora / Y. Yang, J. Carbonell, R. Brown, R. Frederking. – Artificial Intelligence. – 1998. – № 103. – P. 323-345.
5. *Kanerva P., Kristoferson J., Holst A.* Random indexing of text samples for Latent Semantic Analysis // 22nd Ann. Conf. of the Cognitive Science Society. – U Pennsylvania: Erlbaum, 2000. – P. 1036.
6. *Karlgren J., Sahlgren, M.* From Words to Understanding // Foundations of Real-World Intelligence / Y. Uesaka, P. Kanerva, H. Asoh (eds.) – Stanford: CSLI Publications, 2001. – P. 294-308.
7. *Куссуль Э.М.* Ассоциативные нейроподобные структуры. – Киев: Наук. думка, 1991. – 144 с.
8. *Salton G.* Automatic Text Processing: The Transformation, Analisys, and Retrieval of Information by Computer. – Addison-Wesley, Reading, MA. – 1989. – 530 p.
9. *Wong S., Ziarko W., Raghavan V., Wong P.* On modeling of information retrieval concepts in vector space // ACM Transactions of Database Systems. – 1987. – № 2. – P. 299-321.
10. *Kaski S.* Dimensionality Reduction by Random Mapping: Fast Similarity Computation for Clustering // IJCNN'98, Intern. J. Conf. on Neural Networks. – Piscataway, NJ. – 1998. – **1**. – P. 413-418.
11. *Johnson W., Lindenstrauss J.* Extensions of Lipshitz mapping into Hilbert space // Contemporary Mathematics. – 1984. – № 26. – P. 189-206.
12. *Indyk P.* Algorithmic Applications of Geometric Embeddings // FOCS-01. IEEE. – 2001.– P.10-35.
13. *Charikar M.* Similarity estimation techniques from rounding algorithms // ACM Symp. on Theory of Computing. – 2002. – P. 380-388.
14. *Burgess C., Lund K.* The dynamics of meaning in

memory // Cognitive Dynamics / E. Dietrich, A.B. Markman (eds.). – Mahwah, NJ: Lawrence Erlbaum Associates. – 2000. – P. 117-156.
15. *Caid W., Dumais S., Gallant S.* Learned vector-space models for document retrieval // Information Processing and Management. – 1995. – **31**(3). – P. 419-429.
16. *Процедура связывания* для бинарного распределенного представления данных / Д.А. Рачковский, С.В. Слипченко, Э.М. Куссуль, Т.Н. Байдык. – Кибернетика и системный анализ. – 2005. – № 3 – С. 3-18.
17. *Cornell* University SMART text collections. – ftp://ftp.cs.cornell.edu/pub/smart/
18. *TASA* text collection. – http://lsa.colorado.edu/spaces.html
19. *Нейрокомпьютеры* и интеллектуальные роботы / Н. Амосов, Т. Байдык, А. Гольцев, А. Касаткин, Л. Касаткина, Э. Куссуль, Д. Рачковский. – Киев: Наук. думка, 1991. – 272 р.
20. *Гладун В., Величко В.* Конспектирование естественноязыковых текстов // 10 Intern. Conf, "Knowledge – Dialogue – Solution" KDS'05. – 2005. – № 2. – С.344-347.
21. *Грищенко В.Н., Лаврищева Е.М.* Компонентно-ориентированное программирование. Состояние, направления и перспективы развития // Пробл. программирования. – 2002. – № 1-2. – Р. 80-90.

Об авторах

Рачковский Дмитрий Андреевич,
канд. техн. наук, ст. научн. сотр.

Мисуно Иван Семенович,
вед. инж.-программист

Слипченко Сергей Витальевич,
вед. инж.-программист

Соколов Артем Михайлович,
мл. науч. сотр.

Место работы авторов:
Международный научно-учебный центр информационных технологий и систем НАН Украины и МОН Украины
Просп. Акад. Глушкова, 40, Киев, Украина
Тел. +38 (044) 502 6341
E-mail: dar@infrm.kiev.ua
i.misuno@longbow.kiev.ua
slipchenko_serg@ukr.net
sokolov@ukr.net