УДК 519.767.6

А. М. Андреев, Д. В. Березкин,
К. В. Симаков

МЕТОД ОБУЧЕНИЯ МОДЕЛИ ИЗВЛЕЧЕНИЯ ЗНАНИЙ ИЗ ЕСТЕСТВЕННО-ЯЗЫКОВЫХ ТЕКСТОВ

Приведен метод обучения модели извлечения знаний из естественно-языковых текстов. Возможность обучения обеспечивается простотой правил извлечения и решеткой лексических ограничений, являющихся ключевыми элементами модели. Метод обучения формирует набор правил на основе обучающих примеров, подготовленных человеком-экспертом. Проведен ряд экспериментов, дана оценка зависимости основных показателей качества обученной модели от свойств исходной обучающей выборки.

Основное назначение технологий извлечения знаний из естественно-языковых текстов заключается в сборе представляющих интерес фактов по массиву текстов некоторой предметной области. Извлекаемые факты представляют собой структурированное описание событий и явлений, излагаемых в анализируемых текстах. Например, структурными элементами фактов могут быть имена/названия, участники события, их цели и средства, место события, его причины и последствия.

Одно из популярных применений технологий извлечения — это составление досье на представляющий интерес объект, информация о котором доступна из открытых источников, таких как тексты новостей электронных СМИ. Например, интересующим объектом может выступать некоторый политический деятель, досье на которого может включать такую информацию, как фамилия, имя, отчество, возраст, происхождение, образование и т.д. Аналогичным образом выполняется разведка в коммерческих целях, когда некоторая компания интересуется активностью конкурента, действия которого освещаются в СМИ. В данном случае извлечению подвергаются данные об анонсируемых продуктах конкурента, сделках с другими участниками рынка, изменениях, происходящих на руководящих должностях, а также о поглощениях и слияниях других компаний. Вместе с тем, компания может интересоваться собственным информационным портретом, отражаемым СМИ. Этот портрет, кроме досье, может содержать элементы, учитывающие отношения потребителей к продвигаемым компанией брендам.

Основной проблемой при построении системы извлечения является обеспечение должной полноты и точности модели. В большинстве

случаев модель извлечения представляется правилами извлечения, описывающими условия, которым должны удовлетворять фрагменты текста, чтобы из них было выполнено извлечение. В идеальной системе правила извлечения должны охватывать все возможные фрагменты текстов, подлежащие извлечению. Составление правил человеком-экспертом вручную в большинстве случаев требует больших трудозатрат, кроме того, приводит к появлению правил, противоречащих друг другу. Такие проблемы связаны с тем, что эксперт не в состоянии запомнить все правила, которые он уже составил, и все фрагменты текстов, которые эти правила охватывают. Зачастую правила, составленные таким образом, оказываются недостаточно полными и охватывают только фрагменты текстов, которые известны эксперту, но не охватывают аналогичные фрагменты, с которыми ранее эксперт не сталкивался. Возможна и противоположная ситуация, когда эксперт составляет слишком обобщенные правила, так что на практике они ошибочно покрывают фрагменты, существование которых изначально не было учтено экспертом. В таком случае имеет место низкая точность правил. Для решения указанных проблем целесообразно использовать методы машинного обучения, позволяющие автоматически формировать правила извлечения по обучающим примерам, подготовленным экспертом.

В настоящей работе описан метод обучения для разработанной ранее модели извлечения, кроме того, приведены результаты экспериментов, проведенных над текстами из разных предметных областей, дающие оценку точности и полноты обученных моделей и позволяющие судить о предложенном методе обучения.

Обзор методов обучения для задач извлечения. Методы обучения зависят от типа анализируемых текстов. Тексты можно разделить на структурированные, слабоструктурированные и неструктурированные [1]. Особенностью структурированных текстов является наличие специальных символов, не принадлежащих алфавиту естественного языка. Такие символы используются для явного определения структурных элементов в текстах, например с помощью HTML- или XML-тэгов. К слабоструктурированным относятся тексты, где некоторые извлекаемые знания явно выражены символами или цепочками символов, принадлежащими алфавиту языка. В работе [2] в качестве слабоструктурированных рассматривались тексты, описывающие свободные вакансии программистов в ИТ-компаниях. Описание каждой вакансии имело несколько полей: название компании, язык программирования, платформа, опыт работы и т.д. К неструктурированным относятся методы, извлекающие знания из текстов, авторы которых явно не выделяли знания при их написании. В статье изложен метод обучения, формирующий, в первую очередь, правила извлечения для

неструктурированных текстов, тем не менее, его можно использовать и для первых двух типов текстов.

Методы обучения разделяются по стратегии обучения на методы, действующие "сверху вниз" и "снизу вверх". Первые выполняют итеративную конкретизацию [3], формируя из общих правил более конкретные правила. Сложность таких методов применительно к естественному языку заключается в том, что при большой обучающей выборке примеров на первых шагах обучения приходится перебирать чрезвычайно большое число вариантов "расщепления" общего правила на более конкретные. Методы, действующие по принципу "снизу вверх" [4] формируют из конкретных правил более общие правила. Для обработки естественно-языковых текстов такие методы подходят лучше, поскольку число возможных вариантов обобщения текущих правил ограничено. Основным недостатком такой стратегии является "недоученность" модели. Это проявляется в том, что результат обучения представлен недостаточно общими правилами, что в итоге снижает полноту извлечения обученной модели.

По стратегии использования обучающих примеров методы разделяются на "сжимающие" и "покрывающие". Для первых [2] характерно использование всех обучающих примеров на каждом этапе обучения. Покрывающая стратегия предписывает отбрасывать обучающие примеры, для покрытия которых уже сформированы правила извлечения [5].

По способу представления обучающих примеров методы разделяются на следующие группы. Методы, использующие примеры, представленные в виде логики нулевого порядка (атрибутивная логика) [6]. Такие примеры ограничиваются описанием признаков текстовых элементов извлекаемых фрагментов, при этом не учитывается взаимосвязь между этими элементами. В таких методах полагается, что синтаксические шаблоны текстовых элементов извлекаемых фрагментов предопределены, поэтому синтаксические роли элементов могут быть представлены в виде соответствующих признаков (атрибутов). Методы, использующие примеры в виде логики первого порядка [5], учитывают не только признаки текстовых элементов, но и взаимосвязи между ними. Предопределенных синтаксических шаблонов не существует, они выводятся в процессе обучения и являются частью полученных правил извлечения.

По типу формируемых правил выделяют методы, формирующие правила, которые извлекают значения только одного слота целевой структуры [7], и методы, формирующие правила, способные извлекать значения всех слотов целевой структуры одновременно [8].

Анализ существующих подходов к извлечению знаний из текстов выполнен преимущественно на основе зарубежных разработок, что

связано с повышенным интересом именно зарубежных исследователей к данной теме. Наиболее популярной является серия конференций MUC (Message Understanding Conference), проводимая при поддержке DARPA (Defense Advanced Research Projects Agency) в целях совершенствования методов компьютерной разведки. В связи с этим большинство существующих моделей извлечения и методов их обучения ориентированы на языки западной Европы (в первую очередь, на английский), а также на некоторые восточные.

Именно поэтому, в настоящей работе была поставлена цель — разработать метод обучения предложенной ранее модели извлечения [9–11], обеспечивающей извлечение знаний из неструктурированных текстов и учитывающей особенности русского языка. Для обеспечения практической применимости при разработке метода обучения было отдано предпочтение обобщающей стратегии "снизу вверх", но для повышения полноты обученной модели предложена ее модификация.

Представление знаний и текста. В задачах извлечения знаний текст рассматривается как последовательность сегментов. Минимальными элементами сегмента являются слова, представляющие собой последовательности символов алфавита естественного языка, а также знаки препинания. Данная модель текста представима в виде алгебраической системы вида

$$TM =< T, W, t_{\varnothing}, \bullet >, \tag{1}$$

где T — множество текстовых сегментов; W — множество слов; $t_{\varnothing}$ — пустой текстовый сегмент; $\bullet$ — операция сцепления на T. В модели текста определены следующие свойства:

1. $\forall w \in W \Rightarrow w \in T$ — каждое слово является текстовым сегментом;

2. $\forall t_1 \in T \wedge \forall t_2 \in T \exists! t = t_1 \bullet t_2 \wedge t \in T$ — операция сцепления позволяет из произвольной пары текстовых сегментов сформировать новый текстовый сегмент;

3. $t_{\varnothing} \in T \wedge \forall t \in T \Rightarrow t = t_{\varnothing} \bullet t \wedge t = t \bullet t_{\varnothing}$ — пустой текстовый сегмент является нейтральным элементом по отношению к операции сцепления;

4. $t_1, t_2 \in T \wedge t_1 \neq t_{\varnothing} \wedge t_2 \neq t_{\varnothing} \Rightarrow t_1 \bullet t_2 \neq t_2 \bullet t_1$ — некоммутативность операции сцепления.

На основе приведенных свойств модели можно сделать следующие выводы:

1. $\forall t \in T \Rightarrow t = w_1 \bullet \ldots \bullet w_n \colon w_i \in W \wedge w_i \in t$ — любой текстовый сегмент может быть представлен в виде сцепления слов;

2. Поскольку слова являются неделимыми сегментами, то удобно измерять длину сегментов в словах, далее длину сегмента t будем обозначать N_t;

3. Длина пустого сегмента $t_\varnothing$ равна нулю, т.е. $N_{t\varnothing} = 0$.

В качестве модели представления знаний используются фреймы [12, 13]. Фрейм рассматривается как структура с поименованными элементами — слотами. Для дальнейшего изложения ограничимся описанием аксиоматической части фреймовой модели

$$FA =< F, S, T, R_{FS}, R_{ST} >, \tag{2}$$

где F — множество фреймов; S — множество фреймовых слотов; T — множество значений слотов; $R_{FS} \subseteq F \times S$ — отношение, задающее связи между слотами и фреймами; $R_{ST} \subseteq S \times 2^T$ — отношение, задающее для каждого слота допустимую область значений.

Предположим, что FA задается человеком-экспертом, который определяет все возможные фреймы и составляющие их слоты. Также полагается, что все возможные значения слотов T представимы в виде текстовых сегментов модели TM.

Модель извлечения. Детальное описание предлагаемой модели извлечения приведено в работе [11]. Приведем описание некоторых компонентов этой модели и проиллюстрируем их на примерах.

Компоненты модели. Ключевыми компонентами модели являются множество правил извлечения V, множество образцов P и множество элементов образцов R. Правила конструируются из образцов, а образцы — из элементов при помощи операции сцепления. Любой образец можно представить в виде сцепления n элементов — $\forall p \in P \Rightarrow p = r_1 \circ \ldots \circ r_n$. Любое правило извлечения представляется в виде сцепления трех образцов: префиксного, извлекающего и постфиксного — $\forall v \in V \Rightarrow v = p_b \circ p_c \circ p_a$. Префиксный и постфиксный образцы могут быть пустыми (т.е. нейтральными по отношению к операции сцепления). В модели извлечения введена функция покрытия $a\colon T \times V \to \{\text{истина, ложь}\}$. Данная функция для любого правила извлечения и любого текстового сегмента позволяет ответить на вопрос, покрывает ли данное правило данный текстовый сегмент. Функция покрытия также применима для образцов и их элементов. Правило $v = p_b \circ p_c \circ p_a$ покрывает текстовый сегмент, если этот сегмент представим в виде тройки $t_b \bullet t_c \bullet t_a$, и каждый из этих сегментов покрывается соответствующим образцом из тройки $p_b \circ p_c \circ p_a$. Образец $p = r_1 \circ r_2 \circ \ldots \circ r_n$ покрывает текстовый сегмент, если этот сегмент представим в виде $t_1 \bullet t_2 \bullet \ldots \bullet t_n$, и каждый t_i покрывается соответствующим r_i. Функция покрытия для элемента образца определяется внутренней структурой элемента. Если правило покрывает текстовый сегмент, то извлечению подлежит та часть текстового сегмента, которая покрывается извлекающим образцом правила. Отсюда следует связь между моделью извлечения и моделью фреймов.

1. $\forall s \in FA \exists V_s \subset V : \forall v \in V_s \wedge \forall t = t_b \bullet t_c \bullet t_a \in T \wedge a(t, v) = $ $= $ истина $\Rightarrow t_c \in T_i : s\, R_{ST}\, T_i$ — с каждым слотом s связан набор правил V_s такой, что любой текстовый сегмент, извлекаемый одним из правил, принадлежит области значений данного слота.

2. $\forall s_1, s_2 \in FA \exists V_{s1}, V_{s2} \subset V : V_{s1} \cap V_{s2} = \varnothing$ — множества правил извлечения для каждого слота уникальны и не пересекаются между собой.

Чтобы дать интерпретацию функции покрытия для элементов образцов, рассмотрим структуру элемента

$$r_i =< c, e,\ \ l_1, l_2 >, \tag{3}$$

где $c \subseteq W$ — лексическое ограничение; $e \subset W$ — исключение лексического ограничения; l_1 и l_2 — минимальная и максимальная длины покрытия элемента. Лексическое ограничение c и его исключение e определяют множество слов $c \backslash e = \{w\}$, которые могут встречаться в текстовых сегментах $T_{ri} = \{t\}$, покрываемых элементом r_i. Слова $\{w\}$ берутся из множества W модели текста (1). Минимальная и максимальная длины покрытия l_1 и l_2 определяют допустимый диапазон длин текстовых сегментов T_{ri}. Таким образом, чтобы элемент r покрывал текстовый сегмент t, необходимо, чтобы все слова, сцепление которых образует t, принадлежали множеству слов, разрешенных лексическим ограничением элемента, не попадали в исключения, а длина текстового сегмента должна находиться в диапазоне $[l_1, l_2]$.

Поясняющие примеры реализации модели. В программной реализации модели используется XML-нотация для описания правил извлечения. Правило описывается XML-элементом <rule. . . >, содержащим пустые дочерние элементы с тэгами <ct/> и <ex/>. XML-элементы <ct/> описывают элементы префиксного и постфиксного образцов, XML-элементы <ex/> описывают элементы извлекающего образца. Данные элементы имеют атрибуты set и len. Синтаксис записи значения атрибута len следующий: len="[l_1;l_2]", где l_1 и l_2 — числа, обозначающие верхнюю и нижнюю границы задаваемого диапазона. Атрибут set имеет следующий синтаксис: set="A\B", где A и B — записи, задающие соответственно множества лексических ограничений c элемента образца и e — исключений из c. В случае, когда $e = \varnothing$, вторая часть в записи set="A\B" отсутствует. Записи A и B имеют одинаковый синтаксис, допускающий комбинации из следующих вариантов.

1. Непосредственное перечисление допустимых к употреблению слов. Запись такого множества имеет вид "(word$_1$|word$_2$|. . . |word$_n$)", где word$_i$ — i-е слово множества.

2. Перечисление концевых буквосочетаний слов, допустимых к употреблению. Запись такого множества имеет вид "(*end$_1$|*end$_2$|. . .

$\ldots|$*end$_n$)", где end$_i$ — i-е концевое буквосочетание слов множества. Концевое буквосочетание end$_i$ определяет множество всех слов, концевые буквы которых совпадают с end$_i$.

3. Перечисление морфологических признаков слов, допустимых к употреблению. К морфологическим признакам относится часть речи и принятые в естественном языке значения грамматических категорий. Для русского языка такими категориями являются падеж, число, род, лицо и др. Запись такого множества имеет вид "$\{z_1|z_2|\ldots|z_n\}$", где z_i — i-й морфологический признак. Морфологические признаки связываются логической функцией "И". Таким образом, итоговое множество слов является пересечением множеств, соответствующих указанным в записи морфологическим признакам.

Кроме указанных способов, существуют и другие способы задания множеств лексических ограничений, например использование классификации слов, задаваемой тезаурусами [14, 15] или толковыми словарями [16, 17], но в данной работе они не применялись.

Возьмем в качестве примера текстовые сегменты: "**Компания** nVidia **официально отложила день** выпуска видеокарты..." и "**Фирма** Apple **опровергла слухи** о том...". Оба примера представимы в виде $t_b \bullet t_c \bullet t_a$, где подчеркиванием выделены сегменты t_c, состоящие из одного слова и подлежащие извлечению. Значениями целевого слота являются названия компаний. Для первого примера $t_b = компания$, $t_c = nVidia$, $t_a = официально\ отложила\ день$. Для второго примера $t_b = фирма$, $t_c = Apple$, $t_a = опровергла\ слухи$. XML-запись правила, покрывающего данные примеры, имеет следующий вид.

```
<rule name = "company_1">
  ct len = "[1;1]" set = "{И|ЕД}"/>
 <ex len = "[1;1]" set = "{eng}"/>
 <ct len = "[0;1]" set ="{нрч}"/>
 <ct len ="[1;1]"set ="{сов|пхд|глг|ЕД}/>
 <ct len ="[1;1]"set ="{В}"/>
</rule>
```

Приведенное правило состоит из пяти элементов. Оно представимо в виде $p_b \circ p_c \circ p_a$, так что образец p_b состоит из одного элемента **<ct len="[1;1]" set="{И|ЕД}"/>** и покрывает все текстовые сегменты, состоящие из одного слова, которое должно быть отнесено к категории единственного числа именительного падежа. Для данного примера такими сегментами являются $t_b = компания$ и $t_b = фирма$. Извлекающий образец p_c состоит из одного элемента, выделенного подчеркиванием, <ex len="[1;1]" set="{eng}"/>. Этот элемент покрывает все текстовые сегменты, состоящие из одного слова и записанные символами английского алфавита. В данном случае p_c покрывает сегменты

$t_c = nVidia$ и $t_c = Apple$. Постфиксный образец правила состоит из трех элементов, выделенных жирным шрифтом. Первый элемент образца **<ct len="[0;1]" set="{нрч}"/>** покрывает текстовые сегменты длиной от 0 до одного, слова которых должны относиться только к категории наречий. Второй элемент **<ct len="[1;1]" set="{сов|пхд|глг|ЕД}"/>** покрывает текстовые сегменты, состоящие только из одного слова, которое должно относиться к категории переходных глаголов совершенного вида единственного числа. Последний элемент образца **<ct len="[1;1]" set="{В}"/>** покрывает текстовые сегменты длиной в одно любое слово, у которого допустимо выделить винительный падеж. Поскольку минимальная длина покрытия первого элемента равна 0, в тексте могут не встречаться сегменты, покрываемые этим элементом. Так, если положить $t_a = t_1 \bullet t_2 \bullet t_3$, то для первого примера t_1="официально", t_2="отложила", t_3="день", тогда как для второго примера $t_1 = t_\varnothing$, t_2="опровергла", t_3="слухи".

Решетка лексических ограничений. Для того чтобы представленная модель извлечения была обучаемой единым требованием для всех способов задания лексических ограничений и их исключений является возможность представить все их множество C в виде алгебраической решетки (4). Для этого множество C должно быть частично упорядоченным и на нем должны быть определены операции наименьшей верхней и наибольшей нижней границы:

$$CL =< C, \leq, \underline{\vee}, \bar{\wedge} >, \tag{4}$$

где $C \subseteq 2^W$ — множество лексических ограничений и их исключений; $\leq$ — отношение частичного нестрогого порядка на C; $\underline{\vee}$ — операция наименьшей верхней границы; $\bar{\wedge}$ — операция наибольшей нижней границы. Наименьшая верхняя граница $c_1 \underline{\vee} c_2$ для двух элементов c_1 и c_2 определяется как $(c_u = c_1 \underline{\vee} c_2) \wedge \forall c \in C\colon c \leq c_u \Rightarrow (c \leq c_1 \vee c \leq c_2)$. Наибольшая нижняя граница $c_1 \bar{\wedge} c_2$ для двух элементов c_1 и c_2 определяется как $(c_l = c_1 \bar{\wedge} c_2) \wedge \forall c \in C : (c \leq c_1 \wedge c \leq c_2) \Rightarrow c \leq c_l$. Требование к представлению множества C лексических ограничений и исключений в виде решетки CL гарантирует существование метода обучения. Этот факт сформулирован и доказан авторами в виде теоремы "О поиске модели извлечения".

Метод обучения модели извлечения. Задача обучения заключается в генерации множества правил V модели извлечения EM. Разработанный метод обучения относится к методам, основанным на примерах, идея которых заключается в формировании правил извлечения на основе обучающих примеров, подготовленных человеком-экспертом.

Представление обучающих примеров. В задачах обучения [2, 5, 18, 19] принято использовать позитивные и негативные примеры. Дадим формальное определение обучающего примера. Предположим,

что имеется текстовый сегмент вида

$$t_e = t_b^e \bullet t_c^e \bullet t_a^e. \tag{5}$$

Предположим, что наверняка известно следующее:

$$\exists\,(f,s) \in R_{FS} \wedge \exists\,(s,T_i) \in R_{ST}\colon t_c^e \in T_i,$$

т.е. у некоторого фрейма имеется слот, области значений которого принадлежит сегмент t_c^e, являющийся частью t_e. Тогда t_e можно объявить позитивным примером проявления слота s в тексте. По аналогии с данными работ [2] и [37], в качестве негативных примеров принимается любой текстовый сегмент, не входящий в T_e.

Описание метода. Метод обучения использует обучающую выборку $T_e = \{t_e\}$ примеров вида (5). Задача обучения — получить на основе T_e множество правил V модели извлечения EM. Основополагающим критерием генерации правил извлечения является максимизация числа покрываемых правилом позитивных примеров и минимизация числа покрываемых правилом негативных примеров. Поэтому в процессе обучения на каждом шаге выполняется оценка качества полученной к данному шагу модели извлечения. Решения по модификации множества правил извлечения на каждом шаге принимаются только, если это приводит к возрастанию функции $F\,(V,T_e) = \dfrac{1}{N_v}\sum\limits_{v\in V} f\,(v,T_e)$, где N_v — количество правил извлечения множества V; $f(v,T_e)$ — функция качества отдельно взятого правила v. Для оценки качества отдельного правила в настоящей работе используется F-мера [20]:

$$f\,(v,T_e) = \frac{(1+\beta^2)\,P\,(v,T_e)\,R\,(v,T_e)}{P\,(v,T_e) + \beta^2 R\,(v,T_e)}, \tag{6}$$

где $P(v,T_e)$ — точность извлечения правила v; $R(v,T_e)$ — полнота извлечения правила; β — вес, определяющий значимость полноты по отношению к точности, в данной работе использовался $\beta = 1$. Полнота и точность правила v оцениваются как $R\,(v,T_e) = \dfrac{a\,(v,T_e)}{d\,(v,T_e)}$ и $P\,(v,T_e) = \dfrac{a\,(v,T_e)}{b\,(v,T_e)}$ соответственно, где $a(v,T_e)$ — число корректно извлеченных сегментов; $b(v,T_e)$ — общее число извлеченных сегментов; $d(v,T_e)$ — требуемое число извлеченных сегментов, которые должно покрыть в идеале правило. Поскольку в идеале каждое правило должно стремиться покрыть всю обучающую выборку, примем $d(v,T_e) = N_e$, где N_e — число обучающих примеров. Тогда функция качества модели извлечения $F(V,T_e)$ запишется как

$$F\,(V,T_e) = \frac{1}{N_v}\sum_{v\in V}\frac{2a\,(v,T_e)}{b\,(v,T_e) + N_e}. \tag{7}$$

Разработанный метод можно разделить на следующие этапы: формирование предельно конкретных правил, итеративное обобщение, деградация незадействованных примеров, генерация исключений. Рассмотрим первые два этапа метода обучения подробнее.

Формирование предельно конкретных правил. Формирование предельно конкретных правил выполняется на основе позитивных примеров вида (5), каждый такой пример объявляется правилом вида $v = p_b \circ p_c \circ p_a$. Элементы каждого образца формируются на основе слов соответствующей части t_b^e, t_c^e и t_a^e примера. Каждый элемент образца имеет вид $r_i =< \{w_i\}, \{\}, 1, 1 >$, где $\{w_i\}$ — множество из одного слова w_i, соответствующего $(r_i - 1)$-му элементу образца; $\{\}$ — пустое множество исключений. Каждое полученное таким образом правило покрывает ровно один позитивный пример, на основе которого оно было получено.

Итеративное обобщение подразумевает создание новых, более общих правил на основе существующих. Процедура итеративна, поскольку на каждом шаге заменяет существующее множество правил новым множеством сформированных обобщенных правил так, что на следующем шаге предпринимаются попытки обобщения новых правил без участия старых. Данный подход к обобщению отличается от принятых стратегий "сжатия" и "покрытия", поскольку замене подлежит все текущее множество правил извлечения, а не отдельно взятые правила. Алгоритм итеративного обобщения представлен следующими выражениями:

$$
\begin{array}{l}
V = \varnothing \\
V_m = \{v_e\} \ - \textit{предельно конкретные правила} \\
\textit{пока } V_m \neq \varnothing \Rightarrow \\
\quad V_c = \varnothing \ - \textit{правила, полученные на данной итерации} \\
\quad G = (V_m, V_g, R_{mg}) \ - \textit{граф обобщений}\colon V_m - \textit{вершины},\ V_g - \textit{ребра} \\
\quad \forall v_i, v_j \in V_m \ v_{ij} = \textit{Generalize}\,(v_i, v_j) \vee v_{ij} = \varnothing \\
\qquad G\,[v_i]\,[v_j] = G\,[v_j]\,[v_i] = v_{ij} \Leftrightarrow v_i R_{mg} v_{ij} \wedge v_j R_{mg} v_{ij} \\
\quad \forall v_i \in V_m\colon \exists G\,[v_i]\,[v_j] \neq \varnothing \Rightarrow \exists C_i{=}v_i \ldots v_k \ldots v_i - \textit{контур} \\
\qquad \forall v_k, v_l{\in}C_i \ : l{=}k+1 \Rightarrow \exists v_{kl} \in V_g \wedge f\,(v_{kl}, T_e) = \max\limits_{v_{ks} \in V_g} f\,(v_{ks}, T_e) \\
\qquad \forall v_k, v_{k+1} \in C_i \Rightarrow V_c = V_c \cup G\,[v_k]\ [v_{k+1}] \wedge V_m = V_m \backslash \{v_k, v_{k+1}\} \\
\quad V_m = V_c \\
\quad V = V \cup V_c \\
\textit{повторить для нового } V_m.
\end{array}
\qquad (8)
$$

Итеративное обобщение оперирует с множеством V правил, полученных к текущему шагу, и множеством V_m правил, обобщаемых на текущем шаге. Изначально множество V не содержит ни одного правила, множество V_m содержит предельно конкретные правила, полученные

на первом этапе обучения. Итерации выполняются до тех пор, пока удается пополнить множество V, которое и является результатом работы алгоритма. На каждой итерации формируется множество V_c обобщенных правил. Для текущего набора правил V_m формируется граф обобщений G, так что вершинам этого графа соответствуют правила текущего набора V_m, а с каждым ребром связано правило $v_{ij} \in V_g$, полученное в результате парного обобщения правил v_i и v_j, вершины которых соединяет данное ребро. Для дальнейшего изложения удобно принять, что данный граф является ориентированным мультиграфом, у которого кратность каждого ребра равна 2, так что любую пару вершин v_i и v_j в действительности соединяют два противоположно направленных ребра, с каждым из которых связано обобщенное правило v_{ij}. Пример такого графа с учетом указанных замечаний приведен на рис. 1.

Далее, согласно алгоритму (8) полученный таким образом граф анализируется на предмет наличия контуров. Для каждой вершины графа находят оптимальный контур C такой, чтобы для каждой вершины контура из всех ребер графа, инцидентных ей, контуру принадлежало бы ребро v_{ij} с максимальным значением качества $f(v_{ij}, T_e)$. На рис. 1 приведен пример такого контура, составленный из вершин правил v_1 и v_2, соединенных ребром v_{12}. При этом согласно определению контура выполняется условие: $f(v_{12}, T_e) > f(v_{23}, T_e) \wedge f(v_{12}, T_e) > > f(v_{25}, T_e)$. Обобщенные правила v_{kl}, соответствующие ребрам контура C, заносятся в результирующее множество правил текущей итерации V_c, правила v_k и v_l, на основе которых образовано v_{kl}, помечаются как обработанные, чтобы исключить их из дальнейшего анализа графа. Оценка качества обобщенных правил $f(v_{ij}, T_e)$ выполняется согласно уравнению (6).

Для сокращения вычислительных затрат при расчете каждой оценки $f(v_{ij}, T_e)$ используется порог точности θ_p, задающий минимально допустимую точность обобщенных правил, так что $f(v_{ij}, T_e) = 0$, если

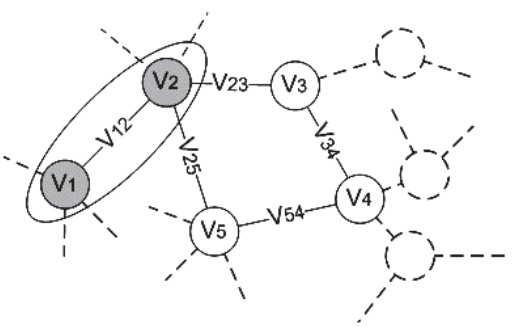

Рис. 1. Пример графа обобщения

$P(v_{ij}, T_e) < \theta_p$. Такой подход позволяет существенно ограничить число проверок покрытий правилом v_{ij}. Так, если при проверке число покрытий правилом превысило значение

$$b(v_{ij}, T_e) > \frac{a(v_{ij}, T_e)}{\theta_p}, \tag{9}$$

то правило можно дальше не проверять и принять его качество $f(v_{ij}, T_e) = 0$. Выигрыш от такого подхода возможен, так как для расчета $a(v_{ij}, T_e)$ достаточно использовать только часть всей обучающей выборки T_e, состоящую из позитивных примеров для текущего слота, тогда как для расчета $b(v_{ij}, T_e)$ в общем случае требуется определять покрытия по всей T_e.

Алгоритм обобщения пары правил Generalize (v_i, v_j) используется при итеративном обобщении в выражениях (8). Пусть правила v_i и v_j представлены в виде троек образцов: $v_i = p_{bi} \circ p_{ci} \circ p_{ai}$ и $v_j = p_{bj} \circ p_{cj} \circ p_{aj}$. Обобщение выполняется независимо для каждой пары образцов (p_{bi}, p_{bj}), (p_{ci}, p_{cj}) и (p_{ai}, p_{aj}). Результатом обобщения каждой такой пары являются множества префиксных (P_b), извлекающих (P_c) и постфиксных (P_a) обобщенных образцов. Для каждой тройки $(p_b, p_c, p_a) \in P_b \times P_c \times P_a$ формируется правило $v = p_b \circ p_c \circ p_a$, если v удовлетворяет критерию (9), то выполняется расчет его качества (6). Из всех возможных троек $v = p_b \circ p_c \circ p_a$ выбирается единственное правило v_{ij} с максимальным качеством $f(v_{ij}, T_e)$.

При обобщении пары образцов (p_i, p_j) независимо от их типа (префиксный, постфиксный или извлекающий) выполняется построение матрицы соответствий A (рис. 2), в которой со строками связаны элементы образца $p_i = q_1 \circ q_2 \circ \ldots \circ q_m$, а со столбцами — элементы образца $p_j = r_1 \circ r_2 \circ \ldots \circ r_n$. Таким образом, размер матрицы составляет $m \times n$.

Матрица заполняется следующим образом. Для любой пары элементов $r_i = < c_i, \varnothing, l_1^i, l_2^i >$ и $q_j = < c_j, \varnothing, l_1^j, l_2^j >$, при использовании операции наименьшей верхней границы решетки лексических ограничений, формируется наименьшее общее лексическое ограничение $c = c_i \underline{\vee} c_j$. Для него определяется величина $s_{ij} = 1 - \sum_{w \in c} p(w)$, значение которой записывается в соответствующую ячейку матрицы, где

	r_1		r_j		r_n
q_1	s_{11}	…	s_{1j}	…	s_{1n}
q_i	s_{i1}	…	s_{ij}	…	s_{in}
q_m	s_{m1}	…	s_{mj}	…	s_{mn}

Рис. 2. Матрица соответствий образцов p_i и p_j

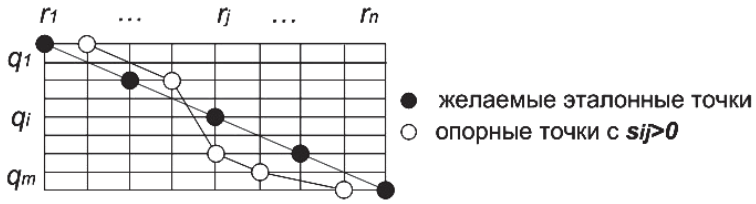

Рис. 3. Поиск маршрута, близкого к основной диагонали

$p(w)$ — вероятность встретить слово w в тексте. Значения в ячейках тем больше, чем больше общих слов содержат исходные лексические ограничения c_i и c_j, а также чем меньше вероятности этих слов. Для полностью идентичных образцов p_i и p_j длиной n элементов матрица A будет квадратной, при этом первые n ячеек с максимальными значениями s_{ij} будут расположены на основной диагонали матрицы. Поэтому поиск общего образца для анализируемой пары p_i и p_j сводится к поиску среди ячеек матрицы A, для которых $s_{ij} > 0$, маршрута опорных точек, в геометрическом смысле близкого к эталонным точкам, равномерно размещенным на основной диагонали матрицы (рис. 3).

Для любой пары соседних ячеек $s_{ij},\ s_{kl}$ этого маршрута должно выполняться следующее условие: $i \le k \wedge j \le l$. Это необходимо, чтобы сохранить порядок следования элементов нового образца таким же, как в исходных образцах. Для оценки геометрической близости маршрута к диагонали используется следующий критерий:

$$W(p) = \sqrt{\sum_{l=1}^{L} A\left[i_l^s; j_l^s\right]^2 H\left(i_l^s, i_l^e\right) H\left(j_l^s, j_l^e\right)}, \qquad (10)$$

где L — число ячеек в маршруте; $A\left[i_l^s; j_l^s\right]$ — значение в соответствующей ячейке матрицы; i_l^s, j_l^s — номера строки и столбца опорной точки; i_l^e, j_l^e — номера строки и столбца для эталонной точки; $H(i_1, i_2) = -\frac{i_1}{i_1+i_2}\log_2\left(\frac{i_1}{i_1+i_2}\right) - \frac{i_2}{i_1+i_2}\log_2\left(\frac{i_2}{i_1+i_2}\right)$ — функция, подобная энтропии, позволяющая оценить близость для произвольной пары ячеек матрицы. Для одинаковых значений $(i_1 = i_2)$ индексов $H(i_1, i_2) = 1$.

Ячейки найденного маршрута являются опорными для построения обобщенного образца. В итоговом образце элементы формируются поочередным применением следующих правил:

$$r =< c, \varnothing, l_1, l_2 >: (c = c_k \underline{\vee} c_l) \wedge l_1 =$$

$$= \min\left(l_1^k, l_1^l\right) \wedge l_2 = \max\left(l_2^k, l_2^l\right); \quad (11)$$

$$r =< c, \varnothing, l_1, l_2 >: c =$$

$$= \underset{s=n+1}{\overset{k-1}{\underline{\vee}}} c_s \underline{\vee} \underset{q=m+1}{\overset{l-1}{\underline{\vee}}} c_q l_1 = \min\left(\sum_{s=n+1}^{k-1} l_1^s, \sum_{q=m+1}^{l-1} l_1^q\right) \wedge l_2 =$$

$$= \max\left(\sum_{s=n+1}^{k-1} l_2^s, \sum_{q=m+1}^{l-1} l_2^q\right). \quad (12)$$

Здесь c_k, c_l — лексические ограничения элементов q_k и r_l исходных образцов, которым соответствует опорная точка матрицы; l_1^k, l_1^l — минимальная длина покрытий элементов q_k и r_l, l_2^k, l_2^l — максимальная длина покрытий элементов q_k и r_l. Первый и последний элементы итогового образца формируются правилом (12). Правило (12) применяется к элементам, которые соответствуют ячейкам подматриц матрицы A, заключенных между соседними опорными точками. На рис. 4 эти подматрицы выделены темным цветом для примера, представленного на рис. 3.

В правой части рис. 4 приведена первая подматрица для данной матрицы. Если положить, что строка и столбец верхней левой опорной точки имеют номер m и n, а строка и столбец нижней правой опорной точки имеют номера l и k, то получим правило (12) формирования элементов итоговых образцов на основе подматриц. Назначение правила (11) — создать новый элемент на основе пары лексически близких (похожих) элементов, образующих опорную точку маршрута. Назначение правила (12) — сформировать новый элемент на основе "непохожих" элементов исходных образцов, заключенных в промежутке между двумя парами "похожих" элементов. Если опорные ячейки располагаются так, что $k = n+1$ и $l = m+1$, то правило (12) создает пустые элементы с пустым лексическим ограничением $c = \varnothing$ и числом повторений $l_1 = 0$ и $l_2 = 0$. Поскольку пустой элемент является нейтральным по отношению к операции сцепления, то такие элементы можно не добавлять к текущему образцу.

Результаты экспериментов. Для оценки качества разработанного метода обучения были проведены эксперименты с тремя обучаю-

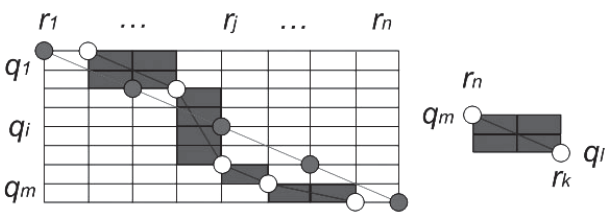

Рис. 4. Промежуточные подматрицы матрицы соответствий

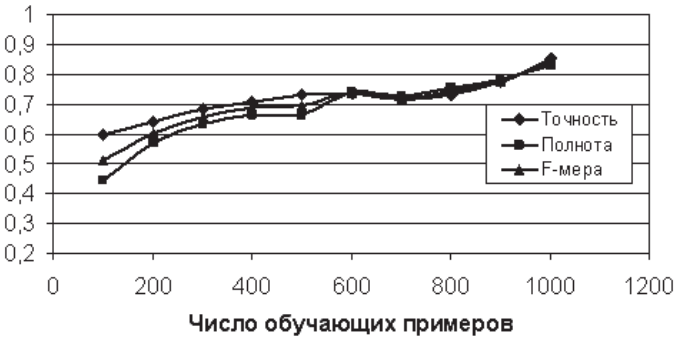

Рис. 5. Результат эксперимента для текстов новостей

щими выборками. Экспериментальной оценке подвергались: точность $P = a/b$, полнота $R = a/N_t$ и F-мера извлечения $F = 2a/b + N_t$, где a — количество корректных извлечений, выполненных обученной моделью; b — общее число извлечений, выполненных обученной моделью; N_t — эталонное число корректных извлечений, которые должна сделать модель.

Обучение на текстах новостей. Тестовая выборка сформирована на основе ленты новостей Интернет-портала, посвященного сфере информационных технологий. Проверке подвергались значения слота "Название компании". Тестовая выборка содержит 3044 названия компаний-производителей продуктов информационных технологий. В обучении использовались выборки от 100 до 1000 обучающих примеров с шагом 100. На рис. 5 отражены графики зависимости оцениваемых показателей качеств от размера обучающей выборки.

Графики демонстрируют общий рост всех трех показателей качества от размера обучающей выборки, при этом разница между точностью и полнотой не превышает 0,05. Для 30 % общего числа обучающих примеров F-мера обученной модели достигает значения 0,85.

Обучение на стенограммах заседаний. Выборка взята из базы данных стенограмм заседаний Совета Федерации Федерального Собрания РФ. Проверке подвергались значения слота "Фамилия члена Совета Федерации". Тестовая выборка содержит 1177 фамилий членов Совета Федерации. В обучении использовали выборки от 50 до 250 обучающих примеров с шагом 50. На рис. 6 приведены графики зависимости оцениваемых показателей качества от размера обучающей выборки, которые демонстрируют практически линейную зависимость показателей качества от размера обучающей выборки. Максимальная разница между полнотой и точностью достигает 0,1. В отличие от предыдущего теста, значение F-меры, равное 0,85, достигается для 25 % общего числа обучающих примеров.

Обучение на текстах почтовых адресов. Тексты взяты из базы почтовых адресов клиентов банка. Проверке подвергались значения

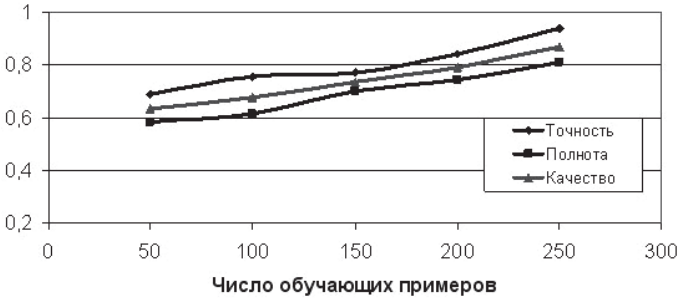

Рис. 6. Результат эксперимента для текстов стенограмм заседаний

слота “Название улицы”. Тестовая выборка содержит 200 адресов. В обучении использовались выборки от 5 до 40 обучающих примеров с шагом 5. На рис. 7 приведены графики зависимости оцениваемых показателей точности от размера обучающей выборки для данного теста, которые демонстрируют экспоненциальную зависимость показателя качества F от размера обучающей выборки. Разница между точностью и полнотой, начиная с размера выборки 30, не превышает 0,05. Особенность данного теста заключается в том, что для 20 % общего числа обучающих примеров, модель достигает значения F-меры, близкого к 1.

Сопоставление с аналогами. Наиболее близкой к данной разработке является система Rapier [3]. Эта систем тестировалась на сообщениях об ИТ-вакансиях в различных компаниях, извлечению подвергались названия компаний, языки программирования и др. По заявлениям авторов, точность системы извлечения, обученной на 200 примерах, составляет 0,85, а полнота извлечения — 0,6. При этом значение F-меры составляет 0,7. Как утверждают сами авторы, для Rapier характерна высокая точность, но низкая полнота. Разница между этими параметрами составляет 0,25. Если сопоставлять данные показатели с нашими тестами, проведенными с текстами новостей, то предложенный метод обучения на 200 примерах обеспечит значения F-меры (см. рис. 5)

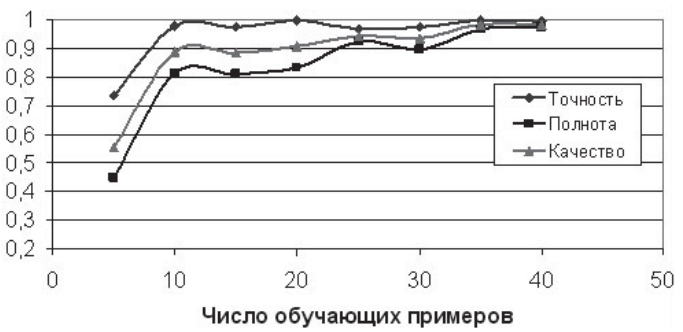

Рис. 7. Результат эксперимента для текстов почтовых адресов

только 0,6, хотя разница между точностью и полнотой в этой точке не превышает 0,1. Такое низкое качество объясняется тем, что для Rapier точка в 200 примеров является точкой насыщения, после которой графики точности и полноты практически не изменяются, тогда как в нашем случае такой точкой можно считать отметку в 600 примеров. В этом случае F-мера обученной модели извлечения достигает 0,75, при этом разница между полнотой и точностью составляет не более 0,05. Это является основным преимуществом разработанной модели по сравнению с рассматриваемыми аналогами — малая разница между полнотой и точностью обученной модели в точке насыщения. Такое свойство обученной модели, в первую очередь, связано с предложенной стратегией итеративного обобщения процесса обучения. Вместе с тем, в нашем случае насыщение достигается позже, т.е. для качественного обучения требуется большее число обучающих примеров. Но этот факт нельзя считать недостатком в сравнении с Rapier, поскольку на этот показатель сильно влияет содержимое обучающей выборки, предметная область текстов и естественный язык, на котором эти тексты написаны. Графики на рис. 5, 6 и 7 это наглядно демонстрируют.

Другой алгоритм, использующий скрытые марковские модели (HMM), предложенный в работе [8], использовался для распознавания адресных объектов в почтовых адресах, представленных сплошными строками. Система [8] достигала значения $F = 0{,}9$ на 50 примерах американских адресов, и на 300 примерах индийских адресов. Как видно из рис. 7, предложенная в данной работе модель обучается на 40 примерах российских адресов для достижения аналогичного качества.

Поскольку в работах [2] и [8] проводились эксперименты с англоязычными выборками, которыми мы не располагаем, говорить о преимуществах представленного в настоящей статье метода было бы несправедливо. Здесь мы всего лишь хотим показать сопоставимость нашего подхода с подходами зарубежных исследователей.

Выводы. В работе описан метод обучения разработанной ранее модели извлечения знаний из текстов на естественном языке. Метод сохраняет работоспособность в условиях “зашумленности” обучающих примеров, т.е. примеров, содержащих как ошибки эксперта-составителя, так и естественно-языковые исключения. Модифицированная стратегия итеративного обобщения позволяет получить в результате обучения малую разницу между значениями точности и полноты модели при том, что общее значение F-меры сохраняется высоким.

Разработанный метод может быть использован в различных задачах, связанных с обработкой неструктурированных и слабоструктурированных текстов. Обученные по данному методу модели извлечения

могут применяться при мониторинге потоков новостей для извлечения конкретных данных по интересующим нас событиям (место, где событие происходит, участники события и др.). Другой областью применения обученных моделей является наполнение тезаурусов и онтологий, когда в качестве источника знаний выступают репрезентативные естественно-языковые тексты предметной области.

В настоящий момент разработанный метод и обученные модели используются в системе семантического контроля текстов редактируемых документов для поиска несоответствий в текстах стенограмм заседаний Совета Федерации Федерального Собрания РФ. Кроме того, эта же технология используется в интеллектуальной системе выявления и исправления ошибок в почтовых адресах клиентов банка.

СПИСОК ЛИТЕРАТУРЫ

1. Jordi Turmo, Alicia Ageno, Neus Catala. Adaptive information extraction // ACM Computing Surveys. – Vol. 38, No 2.
2. Mary Califf, Raymond J. Moony. Bottom-up relational learning of matching rules for information extraction // Journal of Machine Learning Research 4, 2003.
3. Herve Dejean. Learning rules and their exceptions // Journal of Machine Learning Research 2, 2002.
4. Vincent Claveaue, Pascale Sebillot. Learning semantic lexicons from a part-of-speech and semantically tagged corpus using inductive logic programming // Journal of Machine Learning Research 4, 2003.
5. Scott B. Huffman. Learning to extract information from text based on user-provided examples, ACM, 1996.
6. Jun-Tae Kim, Dan I. Moldovan, PALKA: a system for lexical knowledge acquisition. ACM'1993.
7. Ted Pedersen. A simple approach to building ensembles of naive bayesian classifiers for word sense disambiguation. ACM.
8. Vinayak Borkar, Sunita Sarawahi. Automatic segmentation of text into structured records. ACM, 2001.
9. Андреев А. М., Березкин Д. В., Симаков К. В. Особенности проектирования модели и онтологии предметной области для поиска противоречий в правовых электронных библиотеках // VI Всерос. научн. конф. RCDL'2004.
10. Андреев А. М., Березкин Д. В., Рымарь В. С. Использование технологии Semantic Web в системе поиска несоответствий в текстах документов // VIII Всерос. научн. конф. RCDL'2006.
11. Андреев А. М., Березкин Д. В., Симаков К. В. Модель извлечения фактов из естественно-языковых текстов и метод ее обучения // VIII Всерос. научн. конф. RCDL'2006.
12. Udo Hahn, Kornel G. Marko. Joint knowledge capture for grammars and ontologies. ACM'2001.
13. Гаврилова Т. А., Червинская К. Р. Извлечение и структурирование знаний для экспертных систем. – М.: Радио и связь, 1992.
14. Udo Hahn. Knowledge mining from textual sources. ACM'1997.
15. George A. Miller. WordNet: A lexical database for English // Communications of the ACM, Vol. 38, No 11, 1995.

16. German Rigau, Jordi Atserias, Enekko Agirre. Combining unsupervised lexical knowledge methods for word sense disambiguation. ACM, 1996.
17. Yael Karov. Similarity-based word sense disambiguation. Computation Linguistics. – Vol. 24, No 1, 1998.
18. Shigeaki Sakurai, Akihiro Suyama. Rule discovery from textural data based on key phrase patterns. ACM, 2004.
19. Benjamin Rosenfeld, Ronen Feldman, Moshe Fresko TEG — A hybrid approach to information extraction. ACM, 2004.
20. C.J.van Rijsbergen. Information Retrieval. Buttersworth, 1979.

Статья поступила в редакцию 22.05.2007

УДК 004.942: 519.876.5

А. М. Андреев, Д. В. Березкин,
Р. С. Самарев, В. В. Сюзев

АНАЛИЗ ПРОИЗВОДИТЕЛЬНОСТИ РАЗРАБАТЫВАЕМЫХ СИСТЕМ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ И ИНФОРМАЦИОННЫХ СИСТЕМ НА ИХ ОСНОВЕ С ИСПОЛЬЗОВАНИЕМ АЛГЕБРАИЧЕСКИХ МОДЕЛЕЙ

Рассмотрены аспекты моделирования производительности систем управления базами данных и информационных систем, приведены метод оценки характеристик на основе алгебры процессов и пример моделирования.

Моделирование характеристик систем управления базами данных (СУБД) и информационных систем (ИС) актуально в следующих случаях: создание новой СУБД, проверка допустимых режимов работы уже существующей СУБД и оптимизация работы существующих приложений СУБД.

Архитектура СУБД наиболее существенно влияет на производительность будущих систем, построенных с применением данной СУБД. Целесообразно моделировать работу СУБД и возможные последствия от введения каких-либо элементов архитектуры СУБД на ранних стадиях разработки. Кроме того, существует возможность оценить области применения СУБД по предельно допустимым режимам работы.

В случае, когда СУБД уже разработана, необходимо знать допустимые режимы ее эксплуатации (типы приложений и наборы операций, которые не приведут к снижению производительности), а также иметь возможность оценивать требуемые параметры оборудования, такие как