УДК 681.3:51

Саввина Г.В.
Институт проблем искусственного интеллекта

Распознавание ключевых слов в потоке слитной речи

Данная работа предлагает принципиальную схему системы автоматического распознавания слитной речи и описывает некоторые из ее компонентов.

This article proposes schematic diagram of the automatic continuous speech recognition system and describes some of its components.

Введение

Распознавание речи является лишь на 30% проблемой собственно распознавания. На 70% – это проблема семантического анализа текста. Допустим, вы изучали шведский язык и прекрасно знаете алфавит. Вы прочитали уже половину учебника. Но если вас попросят записать по буквам то, что произнес диктор шведского радио, результат будет печальный.

По мнению Давида Яна [1], президента компании ABBYY (BIT Software), человеку – идеально распознающей системе – в слитной речи удается распознать лишь 30% фонем, когда он не знает смысла произносимого текста. Остальную часть информации слушатель восстанавливает, исходя из понимания темы разговора.

Однако семантический анализ является последней, завершающей стадией обработки высказывания. Все должно начинаться с пофонемного распознавания, результатом которого является цепочка фонем, и нахождения всех возможных последовательностей слов заданного словаря, соответствующих полученной цепочке.

В системе автоматического распознавания фирмы IBM переход от цепочки фонем к словам осуществляется акустическим процессором, который отбирает в среднем до 30 вариантов слов словаря, чьё звучание может быть отмечено как звучание соответствующего отрезка. Далее этот список акустически похожих кандидатов слова уменьшается лингвистической моделью, которая отбрасывает некоторые априорно наименее вероятные варианты. Лингвистическая модель определяет вероятность следующего слова в предложении, если известны гипотезы предыдущих слов. Распознаватель использует стандартную модель триграмм языка, которая базируется на интерполяции относительных частот триграмм (последовательность, состоящая из 3-х элементов, в данном случае слов), биграмм и монограмм, полученных из большой выборки слов. Фирмой IBM проводились эксперименты по адаптации разработанной ими системы автоматического распознавания к русскому языку [2].

Для лингвистической модели русского языка основы и окончания слов рассматриваются отдельно. В такой модели последовательность из трех слов будет представлена кортежем из 6 элементов, состоящим из основ и окончаний. Как утверждают представители фирмы IBM, такая схема лингвистической модели ведет к очень медленному декодированию и не может быть использована на практике.

Данная статья предлагает иной подход к задаче, которую решает в вышеописанной системе лингвистическая модель языка. Лингвистическая модель описанной системы опирается на априорную информацию – относительную частоту последовательностей слов. При отсутствии ограничения на предметную область текста трудно, если вообще возможно, оценить минимально необходимый размер обучающей выборки. Вместо этого мы предлагаем опираться на словарь и правила русского языка. Словарь в данном случае определяет, словоформы каких слов могут встречаться в тексте. На данном этапе (в качестве ограничивающих правил) используются правила русского словоизменения [4].

Системы диктовки русского текста в настоящее время являются «белым пятном» для отечественных разработчиков. Единственной широко известной системой диктовки для русского языка является система распознавания «Горыныч» – надстройка фирмы White Computers над английской системой Dragon Dictate. Система требует многочасовой настройки и обучения, имеет небольшой словарь.

Отечественные системы распознавания речи в основном используют эталонный подход и ориентированы на такие задачи:

- распознавание слов (небольшие словари);
- выделение ключевых слов из слитной речи;
- распознавание слитно произнесенных фраз (из заданного набора).

Отличие предлагаемого подхода состоит в отказе от сравнения с эталонами слов или фраз. Вместо этого предлагается определять цепочку фонем произнесенной фразы (слова), приводить эту цепочку (транскрипцию) к письменной форме и далее находить все возможные последовательности слов, составляющих полученную строку графем.

Данная работа ставит перед собой цель создание системы для диктовки произвольного текста, основанной на пофонемном распознавании. В работе предложена схема функционирования системы автоматического распознавания русской слитной речи и описаны некоторые ее компоненты.

Схема функционирования системы распознавания слитной речи

Для распознавания слитной речи предлагается следующая схема действий (рис. 1):

1. Сегментация речевого сигнала (разбиение его на участки, отвечающие отдельным аллофонам), распознавание аллофонов и получение транскрипции.

2. Преобразование цепочки транскрипционных символов в строку слитного текста (строку букв) – модуль «Детранскриптор».

3. Преобразование слитного текста в последовательность слов – программа «Строка».

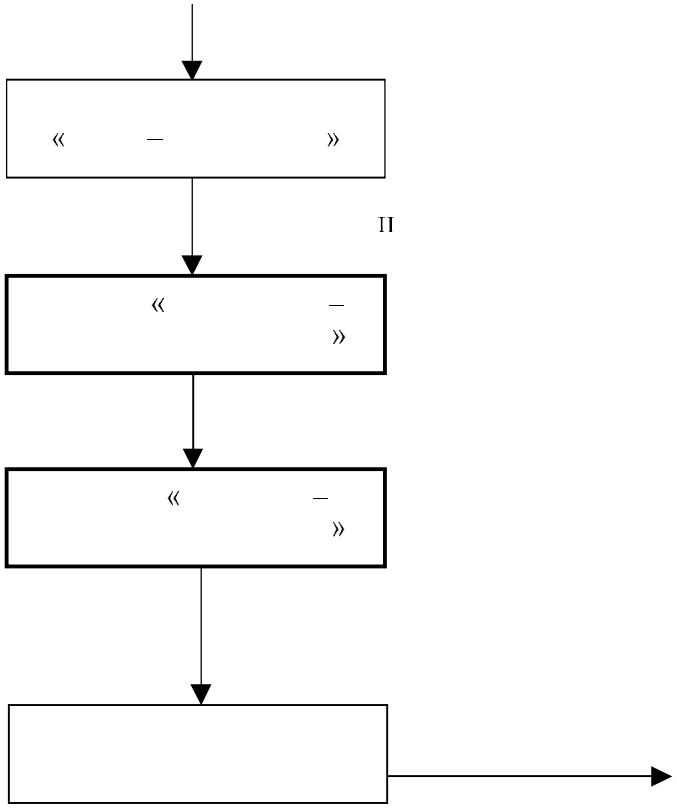

Рис. 1. Схема функционирования системы распознавания

В настоящей работе мы не затрагиваем проблемы сегментации сигнала [5] и распознавания аллофонов. Данная работа посвящена описанию модулей детранскриптора и деления слитной строки на слова. Начнем с первого, ориентируясь на систему транскрипции [6].

Модуль «Детранскриптор»

Основная идея, которая используется для построения модуля, состоит в получении всех возможных вариантов строки букв из исходной транскрипции. Исходная транскрипция представляет собой фонетическую транскрип-

цию (московская фонетическая школа), адаптированную к возможностям системы распознавания, применяемой в отделе фундаментальных проблем распознавания речевых образов Института проблем искусственного интеллекта (Донецк). Адаптация заключалась в отождествлении ряда аллофонов. Соотношение знаков русской транскрипции и принятых в программе приведено в табл. 1.

Таблица 1

Соотношение знаков русской транскрипции изнаков, принятых в программе

Знаки транскрипции			
русск. яз.	**программа**	**русск. яз.**	**Программа**
у	у	и′	и
ы	ы	э′	э
Q	q	иᵊ	i
ыᵊ	ъ	у	u
.а,.а.	я.	о,.о.	ё
.э,.э	.е	.у,.у.	ю
а	.а	о	.о
э	.э	у	.у
с	с	с′	s
з	з	з′	z
п,т,к	-	п′, к	-
б,д,г	д	б′,д′, г′	d
в	в	в	v
ф	ф	ф′	f
ш	ш	ш:′	щ
х	х	х	h
-ц	-ц	ч′	-ч
ж	ж	т′	-t
р	р	р	r
л	л	л	′l
м	м	н′	n
н	н	м′	m
j	j		

Для ускорения построения и обработки строки букв используются динамически формируемые массивы Grapheme и Info. Grapheme – массив вариантов букв строки, Info – массив целых, хранящий характеристики звука (твердость / мягкость, звонкость / глухость, ударность / безударность, гласная / согласная).

Преобразование транскрипции в строку букв осуществляется так:

1. Для всех символов транскрипции получаем все варианты букв, звучание которых может обозначаться данным транскрипционным символом (рис. 2).

2. Анализируем характеристики звука, а также соседние звуки и уменьшаем число вариантов (рис. 3).

Проиллюстрируем сказанное на примере транскрипции слова *абитуриент.*

Буквы ‘я’, ‘ё’, ‘е’ в начале слова, а также перед гласными дают два звука. В данном случае было бы [jq]. Следовательно, эти варианты нужно исключить

из рассмотрения. Звук [i] следует за мягкой согласной (3-й и 7-й звуки). Следовательно, не может быть звучанием букв ‘ы’, ‘э’. Звуки [je] дают букву ‘е’. В результате количество вариантов уменьшается:

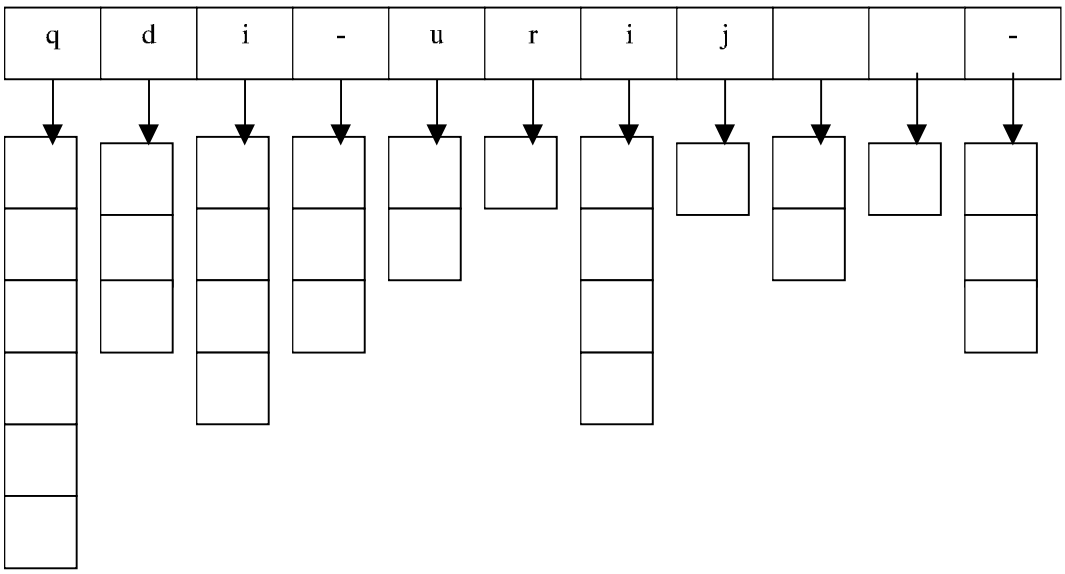

Рис.2

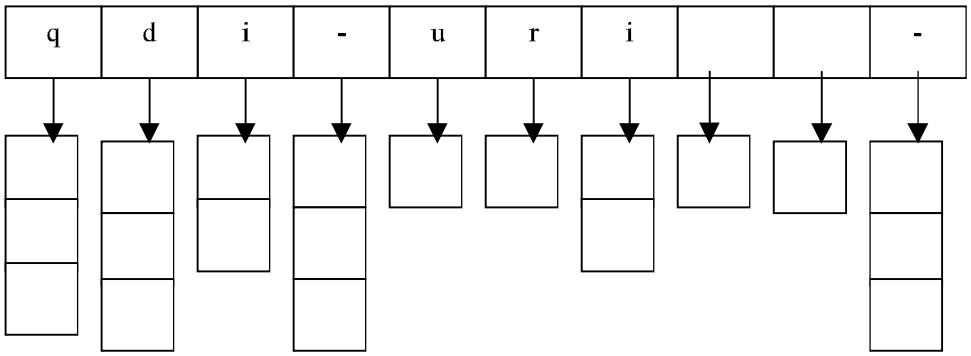

Рис. 3

Независимо от описанного выше модуля проводились эксперименты по членению слитного текста на слова, которые дали неплохие результаты. Его описание приведем ниже.

Модуль «Строка» – программа членения слитного текста

Под *слитным текстом*, по аналогии со слитной речью, будем понимать текст, не разделенный на слова пробелами и знаками препинания.

Задача состоит в получении последовательности слов из строки слитного текста при условии, что строка состоит из слов заданного словаря. Например, из строки «дождьлилцелуюнеделю» должно получиться «дождь лил целую неделю». Следует отметить, что «слово русского языка» – это не только начальная форма слова, хранимая в словаре, но и его парадигма (множество словоформ). Ожидается, что словарь содержит порядка 100.000 слов и включает в себя лишь начальные формы, снабженные разметкой, которая содержит всю информацию о построении парадигмы. Любая словоформа слова, представленного в словаре, должна быть классифицирована как слово русского языка. Необходимо также учесть возможность существования нескольких вариантов членения строки на слова. В этом случае требуется найти все такие варианты.

Решение «в лоб» – поиск слов полным перебором подстрок исходной строки слишком громоздко и медленно. Во избежание полного перебора предлагается следующая оптимизация. Принято считать, что в русской речи столько ударений, сколько в ней самостоятельных слов, так как каждое самостоятельное слово имеет ударение, притом обычно одно. Это позволяет считать ударение одним из основных признаков самостоятельного слова. Служебные слова и частицы не имеют на себе ударения и примыкают к самостоятельным словам. Самостоятельное слово и употребленное с ним служебное слово или частица обычно имеют одно ударение, составляя одно фонетическое слово [3]. Таким образом, получив из транскрипции строку букв с отмеченными ударными гласными, мы можем определить количество самостоятельных слов в строке и крайне грубо «разграничить» слова, тем самым ограничив пространство поиска.

Затем нужно найти все слова в области, ограниченной ударениями соседних слов. Для этой цели используется модуль лемматизации (построение начальной формы слова – леммы – по его косвенным формам). Он описан ниже.

Поиск допустимого разбиения строки на слова осуществляется следующим образом:

а) Производим поиск слов в строке по следующему алгоритму:

1. Запоминаем номера букв, которые являются ударными гласными.
2. Считаем, что перед началом строки и после последней из букв есть ударения (фиктивные).Вводим переменную N и полагаем $N = 0$.
3. Используя функцию лемматизации, находим все возможные слова между ударениями N и $N+2$. Запоминаем их длину и позицию в строке.
4. Увеличиваем N на 1.
5. Если N меньше числа ударений переходим к п. 3.
6. Вводим переменные i (номер символа строки), $B[i]$ – список длин найденных слов, начало которых приходится на i-ый символ строки, j – номер варианта в списке $B[i]$, $B[i]\ \ j]$ – длина j-го варианта в i-ом списке. Полагаем $i =$

б) Формируем предложение (допустимое разбиение строки) из слов, начиная с i-го символа строки:

1. Полагаем $j = 0$, очищаем список вариантов предложений.
2. Если число элементов списка $B\,i]$ равно 0,

то результат – «неудача», конец.

3. Если $i + B\,i\left[\left[j\right]\right]$ равно длине строки,

то

предложение = слово с началом в i-ом символе и длиной $B\,i\left[\left[j\right]\right]$;

перейти к п. 9.

4. Формируем предложение, начиная с $i + B\,i\left[\left[j\right]\right]$-го символа строки (полагаем $i = i + B\,i\left[\left[j\right]\right]$.

5. Если результат – «успешно»,

то

добавляем в начало всех вариантов предложения начинающегося с $\left(i + B\left[i\left[j\right]\right]\right)$-го символа, слово, которое начинается с символа i и имеет длину $B\,i\left[\left[j\right]\right]$.

6. Наращиваем i.

7. Если $j <$ числа элементов списка $B\left[i\right]$,

то перейти к п. 3.

8. Если нет вариантов разбиения строки,

то результат – «неудача», конец.

9. Результат – «успешно», конец.

Описание модуля лемматизации

Особенностью предлагаемого подхода является учет закономерностей и моделирование морфемной структуры русского языка, без ориентации на конкретную предметную область.

К каждому исходному слову применяется набор правил для перехода от словоформы к лемме. Будем называть этот этап этапом генерации кандидатов на роль леммы. Очевидно, не все кандидаты – леммы данного слова. Отбор лемм осуществляется путем поиска кандидата в словаре исходных форм и сопоставления информации, содержащейся в разметке, с правилом, применением которого получен конкретный кандидат.

Основные принципы, положенные в основу системы.

1. Система должна находить все леммы исходного слова.

2. Система не должна находить «лишних» лемм (чья парадигма не содержит исходного слова). В таблице 2 приведены примеры, характеризующие суть проблемы.

3. Система должна работать в реальном времени (от 0.01-0.08с).

4. Система должна строить морфологическую информацию (МИ) слова, которая будет применяться для последующих этапов анализа.

Программа имеет массив, содержащий всевозможные окончания всех частей речи и всех их словоформ, с полной информацией о том, к какому случаю относится данное окончание.

Поиск леммы осуществляется следующей последовательностью действий.

1. Сопоставление концовки слова с окончаниями из упомянутого массива.
2. Извлечение МИ, характерной для каждого из совпавших окончаний.
3. Построение от форм, соответствующих МИ, кандидатов на роль леммы.
4. Отбор лемм из числа кандидатов.

Таблица 2

Примеры построения лемм

Слово	Кандидат	Примененная процедура	Расшифровка разметки	Результат сопоставления
бубном	бубен	-‘ом’(Т. ед. м.р. тв.гр.), вставка ‘е’, + ‘’(И. ед. м.р.тв.гр.)	м.р., тв.гр. выпадает ‘о’| ‘е’	Лемма
	бубон	-‘ом’ (Т. ед. м.р. тв.гр.), вставка ‘о’, + ‘’(И. ед. м.р. тв.гр.)	м.р., тв.гр.	не лемма
баром	бар	-‘ом’ (Т. ед. м.р. тв.гр.), + ‘’ (И. ед. м.р. тв.гр.)	м.р., тв.гр.	Лемма
	барий	-‘ом’ (П. ед. м.р. адъект.скл.), + ‘ий’ (И. Ед. м.р. адъект.скл.)	м.р., тв.гр.	не лемма

Здесь
Т. – творительный падеж;
И. – именительный падеж;
П. – предложный падеж;
ед. – единственное число;
м.р. – мужской род;
тв.гр. – твердая группа;
адъект.скл. – адъективное склонение.

Поясним приведенное в таблице на примере слова *бубном*. Колонка «Кандидат» содержит те слова-кандидаты на роль леммы, которые были найдены в словаре. Для слова *бубном* это *бубон* и *бубен*.

Для слова *бубном* «примененная процедура» означает следующее. В массиве окончаний ищется такое, которое совпадает с концовкой слова, в данном случае «ом». Из этого массива получаем, что это окончание может иметь существительное твердой группы единственного числа мужского рода в творительном падеже. В программу заложена информация о возможных чередованиях, в данном случае это выпадение гласной ‘о’| ‘е’. Для построения кандидата на роль леммы делаются все возможные чередования и окончание «ом» заменяется нулевым окончанием (-‘ом’,+‘’). Получаем 3 кандидата: «бубен», «бубон», «бубн». Каждый из них ищется в словаре. Если мы нашли его, то информация, содержащаяся в разметке (см. колонку «Расшифровка разметки»), сопоставляется с вышеприведенной. В случае совпадения кандидат включа-

ется в число вариантов леммы, иначе – отбрасывается. Процедура проверки с помощью построения соответствующей словоформы не используется.

Аналогично лемматизируется слово «баром». Пометка об адъективном склонении делается, поскольку этот случай достаточно редкий. При отсутствии этой пометки у существительных – склонение субстантивное.

Описанные компоненты «Детранскриптор» и «Строка» в настоящий момент представляют собой отдельные реально работающие приложения.

Литература

1. Internet пресс-конференция/IT Infoart Stars http://www.ah.ru/congress/chat/yan/index.htm
2. D. Kanevsky, Monkowski M., Sedivy J. Large vocabulary speaker-independent continuous speech recognition in Russian language // SPECOM'96. International Workshop SPEECH AND COMPUTER, St.-Petersburg, Russia 28-31, October 1996.
3. Аванесов Р.И. Фонетика современного русского литературного языка. – М.: Изд-во Московского государственного университета, 1956. – 240 с.
4. Зализняк А.А. Грамматический словарь русского языка. Словоизменение. Около 100000 слов. – М.: «Русский язык»,1977. – 880 с.
5. Дорохин О.А., Старушко Д.Г., Федоров Е.Е., Шелепов В.Ю. О проблеме сегментации речевого сигнала.
6. Грабовая В.А., Федоров Е.Е., Шелепов В.Ю. О системе распознавания русской речи с автоматическим построением эталонов // Искусственный интеллект. — 2000 – № 1.