О повышении качества синтаксического анализа текста за счет обучения с учителем[1]

On the quality improvement of text parsing using supervised learning

Вячеслав Вадимович ДРОЗДОВ, Эдуард Станиславович КЛЫШИНСКИЙ,
Москва, Россия
Московский Государственный Институт Электроники и Математики

Аннотация. В статье обосновывается актуальность создания метода автоматической генерации правил для систем, использующих грамматический синтаксический анализ, и дается описание такого метода.

Ключевые слова: синтаксический анализ, автоматическая генерация правил.

Annotation. The article describes the importance of automatic rules generation for the system using grammar parse and the such method realization for a machine translation system.

Keywords: syntax analysis, automatic generation of rules.

Введение

Начиная с 1950-х годов, в нашей стране и за рубежом интенсивно развиваются методы автоматической обработки текста (АОТ). До сих пор они остаются в центре внимания многих исследователей в области искусственного интеллекта.

Необходимость систем АОТ обусловливается большим ростом объема электронной текстовой информации, из которой нужно извлекать знания по возможности полностью автономными процедурами. Деловой мир заинтересован в качественных и надежных поисковых системах интернета, системах машинного перевода, системах автоматического реферирования текстов, автоматического извлечения данных из текста.

Одним из этапов работы системы АОТ является работа с синтаксисом. В синтаксисе решаются следующие основные вопросы:

- соединение слов в словосочетания и предложения;
- рассмотрение видов синтаксической связи;
- определение типов словосочетаний и предложений;
- определение значения словосочетаний и предложений;
- соединение простых предложений в сложные.

Этап работы с синтаксисом, синтаксический анализ, играет важную роль в задаче моделирования и понимания естественных языков. Заинтересованность делового мира в качественном синтаксическом анализе демонстрирует и

[1] Работы выполнена при поддержке гранта РФФИ № 10-01-00800.

недавняя покупка компанией «Yandex» программного комплекса синтаксического анализа текстов, созданного компанией Cognitive Technologies, для улучшения работы поисковой системы.

Современные системы, проводящие синтаксический анализ, можно разделить на два класса: статистические и грамматические. Обе технологии имеют свои плюсы и минусы.

Статистический анализ

В основу статистических систем положены вероятностные характеристики встречаемости и сочетаемости различных частей речи, собранные в специальных хранилищах [1].

Для формального описания статистической модели языка используются скрытые Марковские цепи. Наиболее широко распространенная версия этой модели основывается на этом формализме, и называется моделью n-грамм. Термин «языковые модели», широко используемый в литературе, синонимичен термину «модель n-грамм». Статистическая модель n-грамм соответствует скрытой Марковской цепи n-1 порядка. Вероятность P(w) какого-то символа последовательности ω=ω1,ω2,….ωt длины T - это первое разложение в соответствии с теоремой Байеса.

$$P(\boldsymbol{w}) = P(w_1)P(w_2|w_1)\dots P(w_T|w_1,\dots,w_{T-1}) = \prod_{t=1}^{T} P(w_t|w_1,\dots,w_{t-1})$$

В соответствии с этой формулой с увеличением длины T символьной последовательности можно получить вероятности для последовательности произвольной по длине. Тем не менее, для практического применения максимальная длина контекста ограничивается n-1 предшественником. Этот контекст зачастую называется историей. Определенная таким образом модель n-грамм позволяет предсказать появление символов в некотором ограниченном наборе, основываясь на контексте из n-1 предшествующего элемента. С увеличением длины контекста возникают серьезные трудности при вычислениях такой модели, и применение на практике предъявляет очень большие требования к памяти [2]. В самом деле, если словарь содержит N слов, то число возможных пар слов будет N^2. Даже если только 0,1% от них реально встречаются в языке, то минимально необходимый объём корпуса для

получения статистически достоверных оценок будет иметь порядок 125 млрд. слов или около 1 терабайта при специально подобранном корпусе. Для триграмм минимальные корпуса будут достигать размеров в сотни и тысячи терабайт. Для преодоления этого недостатка используется развитый аппарат техник сглаживания, которые позволяют производить оценку параметров модели в условиях недостаточных или вовсе отсутствующих данных. Другим подходом к решению той же проблемы является кластеризация словаря, позволяющая сократить модель [3]. Уже 4-граммные модели тяжело использовать в статистических системах. Несмотря на свои успехи триграммная модель ничего не знает о богатых синтаксических и семантических связях, которые содержат естественные языки, позволяя им быть легко распознаваемыми и понимаемыми людьми. Во многих реальных предложениях зависимые слова находятся на довольно большом расстоянии в 5-7 слов, и триграммная модель никак не может учесть эти связи. Использование n-граммных моделей с n=5,6,7 требует гигантских ресурсов и сталкивается с проблемой «редких данных» [4]. Одним из подходов к решению этой проблемы является «Модель дальнодействующих триграмм».

В связи с этим основным недостатком n-грамм можно считать заведомо неверное предположение о независимости вероятности очередного слова от более длинной истории, что затрудняет работу и не позволяет моделировать более глубокие связи в языке и колоссальные, но всё-таки недостаточные для получения достоверных оценок объёмы обучающих данных. Основным достоинством данного класса моделей оказывается возможность построения модели по обучающему корпусу достаточно большого размера и высокая скорость работы [3].

Грамматический анализ

Грамматическая технология основана на применении правил (алгоритмов), когда программа анализирует текст и на основе проведенного анализа синтезирует вариант перевода. Работа такой системы сходна с процессом мышления человека: система анализирует текст, используя множество алгоритмов. Грамматические модели, в отличие от вероятностных, требуют большего внимания при построении, и их автоматическое обучение представляется затруднительным. Однако, даже экспериментальные модели,

состоящие из заведомо неполного свода правил, показывают результаты, сравнимые с самыми современными стохастическими подходами [1]. При этом одной из основных причин, сдерживающих развитие систем, использующих грамматический синтаксический анализ, является сложность в сопровождении грамматики, в частности, добавление новых правил.

Одним из наиболее известных подходов к решению данной проблемы является метод генетических алгоритмов. Грамматики естественных языков могут изучаться с помощью генетических алгоритмов, которые воспроизводят и изменяют грамматические правила и тэги частей речи, улучшая качество последующих поколений грамматических компонентов. Грамматические правила генерируются случайным образом, а затем развиваются. Существующие правила и правила, полученные случайным образом, фильтруются по производительности и допускаются к дальнейшему комбинированию. В отличие от традиционных грамматик, основанных на правилах, метод генетических алгоритмов развивает грамматическую структуру и тэги без априорной информации о грамматических предположениях (таких как часть речи), эмпирически создавая и улучшая грамматики для данного конкретного применения.

Язык содержит набор терминов и правил, позволяющих манипулировать терминами, создавая грамматические конструкции, допустимые языком. Используя генетический алгоритм LUST (Linguistic Using Sexual Techniques) были созданы и оценены грамматики не по их сходству с грамматиками, созданными экспертами-лингвистами, а по тому, насколько хороша производительность системы, использующей генетически созданные грамматики. Постепенное улучшение производительности грамматики, полученной этим генетическим алгоритмом, будет наблюдаться на примере той задачи, на которой происходит ее обучение. В грамматиках, используемых для синтаксического анализа естественного языка текст представлен в качестве «гена», который развивается посредством процесса, аналогично тому, как это происходит в живой природе при естественном отборе, где существа имеющие преимущества перед другими имеют больше шансов выжить и оставить потомство. Алгоритм обеспечивает для правил их создание, размножение и «отмирание» в зависимости от того насколько хорошо они справляются со своей задачей [5].

Приближенно эту идею можно представить так. Пусть, к примеру, перед выполнением очередного шага анализа имеются правила:

A → BCD (1),
A → EF (2),
M → NP (3).

Пусть при анализе некоторого фрагмента правила 1 и 2 оказались применимыми, а правило 3 — нет. В этом случает анализатор может сгенерировать новое правило, в правой части которого будут комбинироваться термы из правой части всех подошедших правил, а левая останется неизменной. Например, может быть создано правило A → ECD (4). Новое правило называется “потомком”, а старые (термы из которых участвовали в формировании нового правила) — его предками. Комбинирование термов происходит случайным образом, и даже, возможно, с добавлением некоторых случайных термов, не входивших в правила-предки.

После добавления нового правила весь набор правил снова применяется к фрагменту текста, затем по некоторому критерию отбираются правила с “наилучшей” применимостью, от них порождаются правила-потомки и итерация повторяется. Время от времени выполняется контроль качества анализа, достигаемого очередным поколением правил-”генов”. Численный критерий, используемый для оценки “применимости” правил и “качества” анализа также является существенной частью представленной модели.

Исследования проводились вплоть до 50-го поколения правил. При этом отмечается достаточно стабильный рост качества анализа, однако форма графиков зависимости качества от номера поколения позволяет предположить замедление темпов роста для поколений старше 50-го.

Описанный метод заслуживает весьма серьезного внимания, но к практическому применению он пока не готов, о чем свидетельствуют показатели качества работы генетического анализатора [6].

Широко известных методов автоматической генерации правил для грамматических моделей крайне мало. Метод генетических алгоритмов является, пожалуй, самым известным. И, как видно из его обзора, к практическому применению он пока не готов. В этих условиях задача создания

системы автоматической генерации правил для грамматических моделей приобретает особую актуальность. При успешном решении этой задачи правила будут генерироваться без участия лингвистов или при их незначительном привлечении, для контроля правильности формируемых правил, что не только ускорит процесс создания правил для синтаксического анализа и синтеза, но и сделает его экономически более выгодным.

В этой статье будет описан метод автоматической генерации правил для системы машинного перевода «Кросслейтор».

Автоматическая генерация правил для системы «Кросслейтор»

Система машинного перевода «Кросслейтор» является представителем грамматической технологии синтаксического анализа. В этой системе правила синтаксического анализа и синтеза записываются в модифицированных Бэкусовских нормальных формах. На вход алгоритма разбора системы поступает цепочка структур. Каждая структура хранит набор морфологических омонимов слов предложения. Разбор осуществляется слева направо по правилам рекурсивного спуска.

Правила разбора хранятся в отдельной базе. Для базы определяется набор правил, являющихся входными (те правила, с которых начинается рекурсивный спуск). Правила записываются в виде модифицированной Бэкусовской нормальной формы.

Результатом разбора правила является успех или неуспех. Неуспех означает, что применить данное правило не удалось. В случае успеха также возвращается набор деревьев и наборы выходных параметров, соответствующих каждому дереву.

Если предложение разобрать удалось, то и создавать новые правила необходимости нет. Нас будет интересовать случай, когда предложение разобрать, используя имеющиеся в базе правила, не удалось.

Деревья подчинения и деревья разбора, используемые в методе генерации правил [2]

Введем некоторые формальные определения. Пусть V - непустое

[2] Данный материал излагается на основе [7].

конечное множество, которое мы будем называть словарем, а его элементы - элементарными символами. Будем называть цепочкой в словаре V конечную последовательность элементарных символов, т.е. отображение в V некоторого непустого начального отрезка натурального ряда {1,…,n}(n>=1). В математической лингвистике цепочки используются в качестве простейших моделей языковых объектов разных уровней: морф, слов или предложений. Элементарные символы интерпретируются в первом случае как фонемы, во втором - либо как фонемы, либо как морфы, в третьем - чаще всего как слова. Нас будет интересовать только третья интерпретация, так что словарь мы будем понимать как множество всевозможных слов некоторого языка (например, русского). Итак, пусть П (от слова предложение) - непустое конечное множество, на котором определено отношение линейного порядка < (отношение предшествования одного слова другому). Элементы множества П будут называться точками. Если П - конечное линейное упорядоченное множество, то всякое дерево D, для которого П служит множеством узлов, называется деревом синтаксического подчинения (или просто деревом подчинения) на П. Если П - множество точек некоторой цепочки X, говорят, что D - дерево (синтаксического) подчинения для X. Рассматривая предложение как цепочку слов, мы можем представить информацию о его синтаксическом строении как набор сведений о «главенствовании» одних точек цепочки над другими. Задать такой набор - значит задать некоторый граф на множестве точек цепочки. То, что этот граф является деревом, есть эмпирический факт: практически всегда, когда изображение структуры предложения в виде набора сведений о главенствовании одних вхождений слова над другими достаточно адекватно, соответствующий граф оказывается деревом [7].

Приведем пример дерева подчинения.

Построение деревьев подчинения для конкретных предложений

естественного языка опирается на языковую интуицию, которую, впрочем, трудно ограничить от навыков, усвоенных при изучении грамматики. В сложившейся практике стрелки проводят в соответствии с традиционным, «школьным» представлением о подчинении одних слов другими, когда это представление достаточно определенно [7].

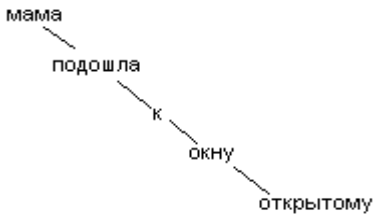
Но структура дерева подчинения, используемого для генерации новых правил, несколько отличается от структуры, предложенной А.В.Гладким. Более детально отличия и причины их появления будут рассмотрены ниже.

Узел D нашего дерева подчинения - это пятерка вида <w,p,r,n,m>, где w - слово в предложении, p - часть речи слова, r - роль слова в предложении, n - порядковый номер слова, от которого зависит текущее (если зависит от вышестоящего по дереву подчинения, то параметр пуст), m - порядковый номер предлога, который относится к текущему слову (используется при согласовании). Информация, хранящаяся в узле дерева D, будет использована нами для построения правила.

После выполнения синтаксического разбора системой «Кросслейтор» нам доступен полный лог разбора, проводившегося системой по правилам для предложения. Лог представляет собою лес, содержащий деревья разбора для каждой из начальных вершин, при этом для каждого начального символа разбора есть дерево с учетом согласования параметров и без учета. Лист дерева разбора L - это пара вида <t,s>, где t - номер терминала в П, с которым проходило сравнение, а s - результат этого сравнения. Нелистовые узлы дерева разбора могут быть двух типов. Первый тип H - это четверка <r,p,n,s>, где r - правило, по которому проходил разбор, p - позиция, с которой проходил разбор, n - количество продукций, s - успешность разбора по этому правилу. Второй тип нелистового узла P - это тройка вида <n1,p,s>, где n1-номер продукции в правиле, по которому идет разбор, p - позиция, с которой проходил разбор, s - успешность разбора по этой продукции.

Однако при построении дерева подчинения по предложению не всегда возможно указать главное и зависимое слово. Например, для случая предлога и относящегося к нему слова. Как и в книге А.В. Гладкого «Синтаксические структуры естественного языка» в этом случае мы будем считать, что в дереве подчинения предлог должен быть родителем того слова, к которому он относится. Например, построим дерево подчинения для предложения «Мама

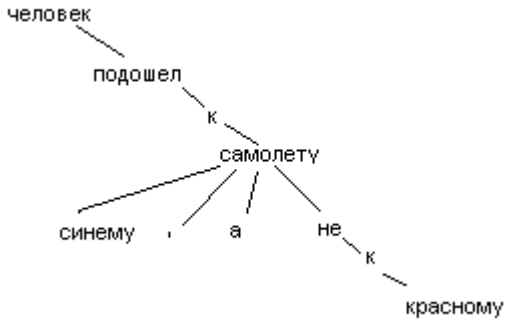
подошла к открытому окну».



Введем еще ряд важных особенностей используемого нами дерева.

- Братьями в дереве подчинения могут на данный момент быть только однородные члены предложения, а также союзы и знаки препинания, разделяющие их.
- Если к слову относится частица, то она в дереве подчинения должна быть родителем этого слова.
- Порядок следования потомков одной вершины соответствует порядку слов в предложении.

Например, дерево подчинения для предложения «Человек подошел к синему, а не к красному самолету».



Генерация новых правил

Итак, после неудачного разбора предложения у нас есть лес деревьев разбора и дерево подчинения, построенное пользователем для неразобранного предложения П. Алгоритм генерации нового правила состоит в следующем.

Шаг 1. *Требуется найти такое предложение П' , все точки которого принадлежат П, которое мы можем разобрать имеющимися у нас в базе правилами.*

Для этого воспользуемся рекурсивным проходом по дереву подчинения. Проход

по дереву и составление потенциального предложения П' начинается с корня дерева и идет в глубину по крайним левым потомкам. Разбор в глубину прекращается при нахождении такого потомка, добавление которого к предложению П' не позволяет разобрать его имеющимися в базе правилами. При проходе в глубину по ветви устанавливаются признаки, что по этим вершинам был осуществлен проход вглубь. Далее осуществляется подъем по дереву до самой верхней вершины, которая еще не участвовала в разборе в ширину (в первом случае - это корень дерева), и отмечаем ее как участвовавшую в проходе в ширину. И начиная с нее, пытаемся собрать предложение с братьями, то есть однородными членами. При этом если вершина не отмечена, как участвовавшая в проходе вглубь, то по ней идет проход в глубину. После прохода в глубину опять возвращаемся наверх, к самой верхней вершине, которая участвовала только в проходе в глубину, и повторяем процедуру до тех пор, пока таким образом не проанализировали все дерево.

Например, для дерева подчинения (см. выше) выбор предложения П' будет проходить так. Предположим, что это предложение разобрать не удалось. Сначала будет выбран корень дерева «человек», разбор этого предложения прошел успешно. Берем потомка для разобранной вершины и получившееся предложение пытаемся разобрать. «Человек подошел» разбирается правилами из нашей базы. Снова берем потомка для последней разобранной вершины. Это предлог «к». Так как предлог относится к какому-то слову всегда, то необходимо взять и потомка для предлога - слово «самолету». При формировании предложения для разбора стараемся учитывать особенности русского языка. Пытаемся разобрать предложение «человек подошел к самолету». Разбор успешен. Берем следующего потомка «синему». Формируем предложение «человек подошел к синему самолету». Порядок слов в формируемом предложении не должен нарушать порядка слов в исходном предложении. То есть, если $a,b \in П$ и $a<b$, то в $П' \subseteq П$ соотношение $a<b$ должно соблюдаться. Разбор в глубину по первой ветви закончен. Мы должны вернуться наверх к вершине и, найдя первого брата сверху, выполнить рекурсивно для него операции по формированию предложения. Таким образом, следующим сформированным предложением по этим принципам будет

«человек подошел к синему, а не к красному самолету». Это предложение имеющимися у нас в базе правилами разобрать не удастся. В итоге мы нашли искомое П' «Человек подошел к синему самолету».

Шаг 2. *Выбор неразобранного поддерева для построения нового правила.*

Помимо того, что у нас есть предложение, которое удается разобрать имеющимися у нас правилами, мы также нашли неразобранные части дерева. Выполнив синтаксический анализ для П', деревом подчинения которого является D', можно получить успешный лес деревьев разбора F‘. Теперь надо выбрать из неразобранных частей дерева ту, для которой мы будем строить правило в первую очередь. Выбор осуществляется на этапе рекурсивного формировании предложения. Приоритет для построения нового правила отдается первой встреченной при рекурсивном проходе в глубину или последней встреченной при рекурсивном проходе в ширину неразобранной ветви, с учетом специфики используемого способа рекурсивного прохода.

Пусть D" $\supset$ D - найденное неразобранное поддерево. В дереве подчинения D у нас есть R" - вершина поддерева D’’ и R’ - успешно разобранная вершина, потомком или братом которой является R’’. Для случая однородных может понадобиться создание фиктивной вершины. Например,

В этом случае для узлов «,» , «а», «не» создадим фиктивную вершину, которая и будет R’’. В случае на рисунке R’’ - брат R’.

Шаг 3. *Поиск в лесе деревьев разбора предложения П’ листьев, по которым был успешно разобран узел R’.*

Следующим шагом для создания правила будет поиск в лесе деревьев разбора предложения П’ листьев, по которым был успешно разобран узел R’, т.е.

L=(t,s)-лист ∈ F’, такой что t = порядковому номеру R’ в X’ и в s’ установлен флаг успешности разбора.

Найдя эти вершины, мы получаем непустое конечное множество.

M={{L1,P1,H1},….,{Li,Pi,Hi},…., {Ln,Pn,Hn}}, где n>=1,

Li - лист дерева разбора ∈ F’

Pi - вершина второго типа дерева разбора ∈ F’

Hi - вершина первого типа дерева разбора ∈ F’

Из этого множества мы можем получить информацию: по какому правилу, по какому номеру продукции, и с какой позиции был успешно разобран узел R’.

Шаг 4. *Формирование нового правила.*

Правило, по которому был разбор вершины R’, хранится в параметре r вершины типа первого H. Номер продукции в правиле находится в параметре n1 вершины второго типа P. С помощью этих данных мы однозначно можем найти во множестве правил разбора нашей базы то правило, по которому прошел успешный разбор вершины R’. Создаваемое нами новое правило будет иметь своим родительским правилом именно его. Имея номер продукции в правиле, возьмем саму продукцию. Она послужит частью нового создаваемого правила, так как вершина, разобранная с помощью этой продукции, является родителем или братом для неразобранной вершины. Если неразобранная вершина R’’ является потомком для вершины R’, то будут сгенерированы два новых правила. Первое правило будет содержать продукцию, по которой был проведен успешный разбор вершины. Второе правило будет содержать новую сгенерированную продукцию. Так как мы имеем дерево D, в котором заданы такие параметры как часть речи, роль слова, то для генерации большей части продукции информация у нас есть. Но для формирования части продукции, содержащей информацию о параметрах согласования в большинстве случаев, на данный момент, понадобится помощь пользователя. При формировании параметров согласования главного и зависимого слова ему необходимо будет отметить, какие из параметров согласуются, а какие создаются заново. По информации из базы правил пользователю можно предоставить предполагаемый выбор, но окончательное решение этого вопроса пока оставлено за пользователем. Также при создании нового правила следует учесть

наличие между главным и зависимым словом предлогов, частиц, знаков препинания, которые не имеют согласования. А, значит, их надо вносить в создаваемое правило, но согласовывать слова по дереву подчинения придется не просто, беря первого потомка, а выбирать потомка для согласования, учитывая его роль в предложении и часть речи. После создания двух новых правил, надо создать и новую продукцию в правиле, по которому была разобрана вершина R'. Возможно возникновение ситуации, когда созданное нами правило уже есть в базе правил разбора, просто не была описана продукция, делающая разбор по нему доступным, потому перед включением новых правил обязательно проверяем эту ситуацию. Если же она возникла, то создаем только новую продукцию и правила, которых в базе еще не было. При создании новых правил для случая, когда неразобранная вершина R'' является братом вершины R', мы уже имеем правило, описывающее один из однородных. Это связано с тем, что проход при составлении предложения П' осуществляется сначала в глубину. Как и в случае создания правила для потомка, необходимо создать правило для неразобранного однородного. Также необходимо создать правило, описывающее саму структуру однородных членов, включив в него знаки препинания и союзы, если они есть в структуре предложения. Именно название этого правила будет использоваться в сформированной продукции для правила, по которому прошел разбор вершины R'.

Шаг 5. *Проверка, есть ли в базе правил правило, описывающее оставшуюся неразобранной часть поддерева.*

После создания нового правила, нужно проверить, не описывается ли оставшаяся неразобранная часть D'' с помощью правил, которые уже имеются в системе. Для этого из оставшейся неразобранной части поддерева строим фразу, которую пытаемся разобрать правилами, уже имеющимися в базе правил. И при нахождении таких правил используем их для построения новых.

Пример генерации нового правила

Приведем пример формирования правила для предложения «человек подошел к синему, а не к красному самолету». П' в данном случае - «человек подошел к синему самолету», R' - «синему», R'' - созданная фиктивная вершина, являющаяся родителем для вершин «,» , «а» , «не» и братом для вершины R'. Номер вершины R' в П' равен 3 (подсчет идет с 0). Сформируем

множество M, состоящее из элементов {Li,Pi,Hi}, где у узла Li параметр t равен 3. В нашем случае, на имеющейся базе правил разбора, оно будет состоять из одного элемента, состоящего из 3-х узлов. В узле H параметр r хранит название правила, по которому был разбор вершины R', а в параметре n1 узла H хранится номер продукции в этом правиле. В итоге из базы данных правил можно однозначно извлечь искомое правило и продукцию. В нашем случае это:

ADJ11|gender 1 ^, number 1 ^, case 1 ^|

[1:;adj;gender1+,number1+,case1 +,comp 1 pos]+property+|

[1:;participle;gender 1 +,number 1 +,case 1 +]+property+|

[1:;poss_pron;gender 1 +,number 1 +,case 1 +]+property+

Номер продукции, по которой прошел разбор - 0. Сформируем первое правило, содержащее продукцию, по которой была разобрана вершина R'.

GEN130|gender 1 ^, number 1 ^, case 1 ^|

[1:;prep;]+prep+[2:;adj;gender 1 +,number 1 +,case 1 +,comp 1 pos]+property+

К продукции из правила ADJ11 при формировании нового правила была добавлена часть, описывающая наличие предлога перед прилагательным. Далее необходимо создать правило, описывающее неразобранный однородный член. Так как это однородные, то часть речи, роль в предложении, параметры согласования у него будут совпадать с однородным братом. Но необходимо учесть наличие перед ним частицы и предлога, поэтому сформированное правило будет иметь вид:

GEN982|gender1^,number1^,case1^|

{1:;'HE':particle;}+nadv+[2:;prep;]+prep+[3:;adj;gender 1 +,number 1 +,case 1 +,comp 1 pos]+property+

Проверим, что таких правил в нашей базе еще нет. В нашем случае их, действительно, нет. Теперь следует создать правило, описывающее структуру однородных членов. Правило будет состоять из названий двух новых сформированных правил, разделенных союзами и знаками препинания в соответствии с деревом подчинения предложения. Оно будет иметь вид:

GEN90|gender 1 ^, number 1 ^, case 1 ^|

<1:;GEN130;gender1+,number1+,case1+>{2:1;',':sign}+sign+{3:2;'A':interj;}+property+<4:3;GEN982;gender 1 +, number 1 +, case 1 +>

Осталось сформировать новую продукцию для правила ADJ11. Новая

продукция будет состоять из названия правила, описывающего структуру однородных. Так как описываются однородные, то параметры согласования будут идентичными тем, которые использовались для разбора вершины R'. В итоге, правило ADJ11 примет вид:

ADJ11|gender 1 ^, number 1 ^, case 1 ^|

[1:;adj;gender1+,number1+,case1 +,comp 1 pos]+property+|

[1:;participle;gender 1 +,number 1 +,case 1 +]+property+|

[1:;poss_pron;gender 1 +,number 1 +,case 1 +]+property+|

<1:;GEN90;gender 1 +,number 1 +,case 1 +>

Пополненная таким образом база правил разбора позволяет осуществить разбор для исходного предложения П «человек подошел к синему, а не к красному самолету».

Вывод

Созданная система генерации новых правил может пополнять существующую базу правил системы машинного перевода «Кросслейтор», позволяя разбирать предложения, которые с помощью исходной базы правил успешно разобрать не удавалось.

При формировании новых правил возникает ряд довольно сложных проблем, которые сейчас пока находятся на стадии разрешения. Основные из них:

1) Как уже было сказано выше, на данный момент, во многих случаях окончательный выбор параметров согласования остается за пользователем. Эту проблему можно будет решить, взяв статистику по многим деревьям.

2) Свободное построение предложений в русском языке, особенно в художественной литературе. На данный момент удается строить новые правила, в основном, для проективных и слабо проективных предложений. Для непроективных предложений попытка построения новых правил предпринимается, но не всегда успешно.

Список используемой литературы

1) В.И. Божич, А.В. Тарасенко, *Омонимические неоднозначности при анализе текстов*, Информационное противодействие

угрозам терроризма, 2005, 253-257.

2) G.A. Fink. *Markov Models for Pattern Recognition. From Theory to Applications*, Springer-Verlag Berlin Heidelberg 2008, 101.

3) А.Б. Холоденко, *О построении статистических языковых моделей для систем распознавания русской речи*, Интеллектуальные системы. Т.6, вып. 1-4, 2002, 384-385.

4) С.В. Протасов, *Вывод и оценка параметров дальнодействующей триграммной модели языка*, Компьютерная лингвистика и интеллектуальные технологии, Труды Международного семинара Диалог-2008.

5) R.M. Losee, *Learning Syntactic Rules and Tags with Genetic Algorithms for Information Retrieval and Filtering: An Empirical Basis for Grammatical Rules*, Information Processing & Management, 1996, 185-197.

6) А.М. Андреев, Д.В. Березкин, А.В. Брик, Ю.А. Кантонистов, *Вероятностный синтаксический анализатор для информационно-поисковой системы*, Компьютерная хроника N1, 1999.

7) А.А. Гладкий, *Синтаксические структуры естественного языка*, Изд. 2 - М.: ЛКИ, 2007, 12-15.

Дроздов Вячеслав Вадимович. Аспирант 3 года обучения, Московский государственный институт электроники и математики. Автор 6 статей и докладов. Область научных интересов: обработка текстов на естественном языке. slava985@yandex.ru , моб. тел. 8 (903) 544-22-83

Клышинский Эдуард Станиславович. Доцент Московского государственного института электроники и математики, доцент, канд. техн. наук. Автор и соавтор почти 50 статей, двух монографий. Область научных интересов: обработка текстов на естественном языке.
klyshinsky@itas.miem.edu.ru
8 (910) 464-92-78