Задание «Синтаксический анализ текстов»

Горелов А.А.

ВМК МГУ им. Ломоносова, кафедра ММП, 417 группа

28 января 2013

Введение

В этой работе мы решаем задачу построения системы, которая автоматически находит связи между словами в тексте. Эта задача называется задачей синтаксического анализа.

Необходимо разработать синтаксический анализатор, состоящий из двух основных компонентов: морфологический анализатор, использующий сокращённую модель Маркова, и синтаксический анализатор, принимающий морфологически размеченный текст и размечающий его уже синтаксически. Первая компонента была разработана в рамках данного исследования, в качестве второй компоненты использовался TurboParser [1]. Далее требовалось протестировать построенный комплекс программ и проанализировать результаты тестирования.

Морфологический анализатор

Сокращённая модель Маркова

На первой стадии нам необходимо определить морфологические свойства слова - например его часть речи, род, падеж, единственное или множественное число. Совокупность морфологических свойств слова мы называем тегом. При этом одно слово может иметь несколько тегов. Например, слову «кофе» могут соответствовать наборы тегов <sing, nomn>, <sing, gent>, <plur, accs>. Другим примером является слово «три», которое может быть глаголом с нормальной формой «тереть», а может быть числительным с нормальной формой «три».

Более формально, пусть w — некоторое слово. $D(w)$ — множество пар <нормальная форма, набор тегов>, соответствующее этому слову. Задача снятия морфологической омонимии состоит в выборе одного элемента из множества возможных тегов.

Традиционная скрытая марковская модель рассматривает наборы тегов как состояния, между которыми при последовательном рассмотрении слов в предложении совершаются переходы из предыдущего слова в следующее слово. При этом на множестве переходов вводится вероятностная мера $\mathrm{P}(t_n|t_p)$, обозначающая вероятность перехода между тегом предыдущего слова t_p и тегом следующего слова t_n. Эту вероятность оценивается по обучающей выборке и равна числу переходов в обучающей выборке, делённому на общее число переходов из тега t_p к другим тегам.

Ещё необходимо учесть частоту встречаемости слов: пока что мы работаем только с тегами, не учитывая нормальные формы. Это некорректно, так как интуитивно ясно, что иногда не очень вероятная связь (относительно тегов) с часто встречающимся словом предпочтительнее очень вероятной связи с редким словом. Поэтому введём также вероятность появления нормальной формы, которую можно оценить частотой встречаемости данной нормальной формы относительно других нормальных форм.

Теперь необходимо немного остановиться на отличиях оригинальной марковской модели от сокращённой: в оригинальной мы выбираем наиболее вероятный путь (последовательность переходов по состояниям) среди возможных путей (ограничения на пути ставят входные данные: если предложение состоит из слов $w_1\ w_2\ \ldots\ w_n$, то возможные пути будут иметь в качестве начальной вершины вершину из $D(w_1)$, в качестве второй — из $D(w_2)$ и так далее. Сокращённая же модель действует жадно: она итерационно идёт по предложению и на каждой итерации переходит в наилучшую возможную вершину-состояние. Таким образом, у нас есть некоторое начальное состояние, на первой итерации мы жадно выбираем тег и нормальную форму первого слова, на второй итерации — второго и так до конца предложения. Осталось определить понятие «наилучшего возможного состояния».

Итак, на i-й итерации мы сопоставили i-му слову тег t_i. Предположим независимость распределений вероятностей на нормальных формах и на связях. Будем учитывать тег только предыдущего слова. Интуитивно ясно, что нам надо выбрать максимально вероятный тег слова $i+1$ при условии, что тег слова i — t_i. Из независимости распределений следует, что

$\mathrm{P}(t_{i+1}, NormalForm|t_i) = \mathrm{P}(t_{i+1})\mathrm{P}(NormalForm)$. Обе вероятности в правой части мы можем оценить по обучающей выборке. Соответсвенно, задача сводится к максимизации правой части равенства по t_{i+1} и $NormalForm$. Заметим, что максимизация идёт по обычно не очень большому множеству $D(w_{i+1})$ (несколько десятков пар-состояний).

Мы получили жадный алгоритм, позволяющий восстанавливать морфологические признаки слов в предложениях. Но есть одна проблема — откуда взять множества $D(w)$?

Словарь OpenCorpora. Предсказание тегов слов, отсутствующих в словаре

Для слов, присутствующих в словаре, обозначенная проблема решается просто — сопоставляем словоформе множество нужных пар нормальных форм и тегов и просто используем его при необходимости.

Проблема возникает, когда нужное слово в словаре отсутствует. В этом случае надо попытаться как-то рационально составить множество $D(w)$. В описываемой программе это делается на основе окончаний слов — при формировании словаря считаются вероятности $\mathrm{P}(t|)$ и выбираются те теги, у которых эта вероятность не равна 0. Эта вероятность становится дополнительным множителем в приведённой выше формуле. Как показало практическое применение, даже в таком случае множество $D(w)$ получается сравнительно маленьким, поэтому перебор не растёт.

В качестве базового словаря использовался словарь проекта OpenCorpora [2]. Словарь хранится в файле в виде списка тегов, к которым относится список пар лемма-нормальная форм. Все эти списки переведены в бинарный вид и запакованы с помощью библиотеки zlib [3].

Синтаксический анализатор

В качестве синтактического анализатора, как уже было сказано, использовался TurboParser[1]. Собственно, на плечи созданной в рамках исследования программы ложилось только преобразование формата ввода данных в виде XML в формат TurboParser и применение на данных описанного выше морфологического анализатора.

```
1   В            В            PREP   PREP   _                                 0   punct   _   _
2   связи        СВЯЗЬ        NOUN   NOUN   inan|femn|sing|gent               0   punct   _   _
3   с            С            PREP   PREP   _                                 0   punct   _   _
4   последними   ПОСЛЕДНИЙ    ADJF   ADJF   plur|ablt                         0   punct   _   _
5   событиями    СОБЫТИЕ      NOUN   NOUN   inan|neut|plur|ablt               0   punct   _   _
6   на           НА           PREP   PREP   _                                 0   punct   _   _
7   Украине      УКРАИНА      NOUN   NOUN   inan|femn|Sgtm|Geox|sing|loct     0   punct   _   _
8   российские   РОССИЙСКИЙ   ADJF   ADJF   inan|plur|accs                    0   punct   _   _
9   казаки       КАЗАК        NOUN   NOUN   anim|masc|plur|nomn               0   punct   _   _
10  решили       РЕШИЛ        VERB   VERB   perf|tran|plur|past|indc          0   punct   _   _
11  помочь       ПОМОЧЬ       NOUN   NOUN   inan|femn|Arch|sing|accs          0   punct   _   _
12  украинским   УКРАИНСКИЙ   ADJF   ADJF   Geox|plur|datv                    0   punct   _   _
13  казачьим     КАЗАЧИЙ      ADJF   ADJF   Poss|plur|datv                    0   punct   _   _
14  организациям ОРГАНИЗАЦИЯ  NOUN   NOUN   inan|femn|plur|datv               0   punct   _   _
15  .            .            PNCT   PNCT   _                                 0   punct   _   _
```

Тестирование

Морфологический анализатор тестировался на размеченных текстах проекта OpenCorpora. , но доля правильно проставленных тегов среди известных словарю слов на тестовой выборке равнялась 96.86%. Тестовая выборка составляла $\frac{1}{6}$ от всей выборки (вся выборка состояла из 5396 предложений. Заметим, что если слова предложения не входят в словарь корпуса OpenCorpora, то при предсказании его тегов возможны дополнительные ошибки, .

Тестирование синтаксического анализатора проводилось на корпусе RusTreeBank v2, состоящем из 1983 предложений. С помощью метода кросс-валидации он 8 раз делился на

№	1	2	3	4	5	6	7	8	Усреднение
Без учёта связей	0.698	0.708	0.679	0.662	0.68	0.688	0.695	0.696	0.688
С учётом связей	0.645	0.669	0.635	0.630	0.634	0.654	0.643	0.649	0.645

Таблица 1: Результаты на корпусе RusTreebank v2

обучающую и тестирующую выборку, размеры тестирующей к обучающей выборке относились как 1 к 7. Результаты тестирования можно увидеть в таблице .

Старт программы, состоящий из загрузки словаря и морфологической модели, занимает 7 секунд. Обучение на всём корпусе занимает около 5 минут на Intel i5 с 4Gb RAM. После прохождения обучения, разметку всего корпуса текстов программа выполняет за 21 секунду. Более половины этого времени занимает работа TurboParser'a.

На тестовых выборках, которые в 8 раз меньше выборки для обучения, морфологический анализатор отрабатывает за 2-3 секунды, а TurboParser работает 7-8 секунд.

Выводы

Из проведённых экспериментов следует, что необходимо улучшение качества работы синтаксического анализатора. При этом различных наборов тегов достаточно много, число возможных вариантов синтаксических связей велико. Поэтому одна из вероятных причин состоит в недостаточном объёме и качестве разметки синтаксического корпуса текстов RusTreeBank v2. Вероятно, двух тысяч предложений мало для качественного обучения: либо надо расширять корпус, либо уменьшать разнообразие тегов.

Другой возможной причиной ошибок является неоднородность корпуса: (а) в нём есть предложения, состоящие почти целиком из английских слов; (б) много предложений с пропущенными словами; и (в) в некоторых предложениях знаки препинания «сливаются» со словами, а в некоторых — отделены от них.

Третьей возможной причиной ошибок являются ошибки в морфологическом словаре OpenCorpora. Например, слова "великий" и "баскетбольный" могут иметь винительный падеж.

Четвёртой возможной причиной ошибок является слишком сильное ограничение в 3 буквы слова на тег, по которому определяется возможный тег для незнакомых слов. Таким образом, для незнакомого слова "двукратная" определяется гипотеза с родительным падежом.

Литература

[1] http://www.ark.cs.cmu.edu/TurboParser/

[2] http://opencorpora.org/

[3] http://www.zlib.net/