一种用未分析语料训练文法的方法*

王 挺 史晓东 陈火旺 杨 谊

(国防科技大学计算机系 长沙 410073)

摘要 本文提出了文法规则的推导概率和归约概率的概念,通过修改 Inside-Outside 算法,将其应用于从未分析语料中获取一般形式的上下文无关文法规则的概率参数.

关键词 Inside-Outside 算法,推导概率,归约概率,语料.

中图法分类号 TP301

运用概率统计技术直接从未加工的语料中获得词法和句法概率知识的方法,因其不依赖于对语料的预先加工而日益受到自然语言处理 NLP(natural language processing)界的重视. 如 Merialdo 将隐马尔可夫模型引入语料的词法标注中,通过使用 Baum-Welch 重估计方法[1],用未标注的语料来训练模型的参数,获得了很好的结果.[2]作为 Baum-Welch 重估计方法在上下文无关文法 CFG(context-free-grammar)上的扩展,Baker 在 1979 年提出了 Inside-Outside 算法(简称 I/O 算法). 该算法可以根据给出的观察序列,估算出产生该观察序列的上下文无关文法.[3]然而,该算法在 NLP 研究中却未得到较好的应用.

我们认为制约 I/O 算法应用的因素主要有 3 个:①该算法复杂度较大,其复杂度是 $O(N^3T^3)$,这里 N 是文法的非终结符的数目,T 是用来训练的观察序列的长度;②该算法只能用来训练乔姆斯基范式文法 CNF(Chomsky normal form),如何训练一般的上下文无关文法颇值得研究;③由于该算法生成文法的非终结符的语言学意义难以确定. 为了克服这些缺陷,Shih 等人[4]使用初步分析过的语料作为训练材料,以降低算法的复杂度和提高估算的准确率. 此外,Kupiec 和 Black 等人也对 I/O 算法进行了一些改进.[5,6]

本文首先介绍了 I/O 算法的基本思想,指出若将该算法用于估算一个给定的 CFG 文法规则的概率比生成一个文法更适合自然语言的研究. 然后,针对现有的上下文无关文法的分析方法,提出了文法规则的推导概率和归约概率的概念,并设计了估算一般的 CFG 文法的概率参数的方法. 文章最后说明了实验的结果,并讨论了相关的问题.

1 Inside-Outside 算法简介

I/O 算法是 Baker 在 1979 年提出的. 为了方便理解,我们将在本节中首先简单介绍一下该算法,更详细的描述和有关证明可参见文献[3,7].

设 $O=O(1)O(2)\ldots O(T)$ 是由随机上下文无关文法(SCFG)G 产生的观察序列(在自然语言问题中,它可理解为语言的一个句子). 文法 G 具有 CNF 形式,即其规则形为:

$$i \to jk \text{ 或 } i \to m$$

其中 i,j,k 是文法的非终结符,m 是文法的终结符(下同). 另外,我们用矩阵 A 和 B 来描述文法 G 的各规则的概率参数:

$$a[i,j,k]=P(i \to jk|G), b[i,m]=P(i \to m|G)$$

其中 $P(i \to jk|G)$ 是非终结符 i 产生非终结符 jk 的概率,$P(i \to m|G)$ 是非终结符 i 产生终结符 m 的概率,我们称这种概率为规则的推导概率.

定义内(Inner)概率 e 如下:

$$e(s,t,i)=P(i \stackrel{*}{\Rightarrow} O(s)\ldots O(t)|G)$$

---

* 本文研究得到国家 863 高科技项目基金资助. 作者**王挺**,1970 年生,博士生,主要研究领域为计算语言学,机器翻译. **史晓东**,1966 年生,博士,讲师,主要研究领域为计算语言学,机器翻译. **陈火旺**,1936 年生,教授,博士导师,主要研究领域为人工智能,计算机软件. **杨谊**,女,1973 年生,硕士生,主要研究领域为计算语言学.

本文通讯联系人:王挺,长沙 410073,国防科技大学计算机系

本文 1996-02-05 收到原稿,1997-02-25 收到修改稿

$e(s,s,i)$表示非终结符i经过若干步推导产生出观察序列$O(s)\ldots O(t)$的概率，它可以按下面的方法计算：

$$e(s,s,i) = P(i \Rightarrow O(s)|G) = b[i,O(s)] \tag{1}$$

$$e(s,t,i) = \sum_{j,k}\sum_{r_0=s}^{t-1} a[i,j,k]e(s,r_0,j)e(r_0+1,t,k), \text{for all } i \text{ and } s \neq t \tag{2}$$

定义外(Outer)概率f如下：

$$f(s,t,i) = P(S \overset{*}{\Rightarrow} O(1)\ldots O(s-1)\ i\ O(t-1)\ldots O(T)|G)$$

$f(s,t,i)$表示文法G的开始符号S经过若干步推导产生出序列$O(1)\ldots O(s-1)\ i\ O(t+1)\ldots O(T)$的概率，它可以按下面的方法计算：

$$f(1,T,i) = \begin{cases} 1 & \text{if } i = s \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

$$f(s,t,i) = \sum_{j,k}\left[\sum_{r_0=1}^{s-1} f(r_0,t,j)a[j,k,i]e(r_0,s-1,k) + \sum_{r_0=t+1}^{T} f(s,r_0,j)a[j,i,k]e(t+1,r_0,k)\right] \tag{4}$$

在计算过程中，我们先自底向上地计算内概率，然后自顶向下地计算外概率．在此基础上，我们可以采用下面的方法重新估算矩阵A和B：

设

$$P = P(S \overset{*}{\Rightarrow} O/G) = e(1,T,s)$$

那么，重新估算$a[i,j,k]$和$b[i,m]$的公式如下：

$$\hat{a}[i,j,k] = P(i \to jk|i \text{ used}) = \frac{P(i \to jk, i \text{ used})}{P(i \text{ used})}$$

$$= \frac{\frac{1}{P}\sum_{s=1}^{T-1}\sum_{t=s+1}^{T}\sum_{r_0=s}^{t-1} a[i,j,k]e(s,r_0,j)e(r_0+1,t,k)f(s,t,i)}{\frac{1}{P}\sum_{s=1}^{T}\sum_{t=s}^{T} e(s,t,i)f(s,t,i)} \quad \text{for all } i,j,k \tag{5}$$

$$\hat{b}[i,m] = P(i \to m|i \text{ used}) = \frac{P(i \to m, i \text{ used})}{P(i \text{ used})} = \frac{\frac{1}{P}\sum_{t \in O(t)=m} e(t,t,i)f(t,t,i)}{\frac{1}{P}\sum_{s=1}^{T}\sum_{t=s}^{T} e(s,t,i)f(s,t,i)} \tag{6}$$

适合于多观察序列的公式可以类似地推出，详细推导见文献[7]．

运用这些公式，我们可以根据给出的观察序列迭代地计算矩阵A和B的值，直到P的变化小于一个阈值，这时我们将所有概率值大于0的规则找出来，就得到了推算的文法．

从上面的推导过程可见，该算法的复杂度为$O(N^3T^3)$．Lari使用该算法做了一个推算文法的实验，能够处理具有20～30个非终结符和60～100个终结符的文法．[7]但是在实际的NLP系统中，如果将文法写成CNF形式，非终结符数日将达到100的数量级．因此，该算法在自然语言领域很少得到应用．另外，该算法所生成的文法的可用性也非常值得怀疑．在通常的NLP系统中，文法的符号都具有特定的语言学意义，如在我们研制的MATRIX机器翻译系统的文法中，NP，PP和SUBJ分别代表名词短语、介词短语和句子的主语等，只有这样我们才能用文法来表示、分析和生成自然语言．而I/O算法所生成的文法中的符号的语言学意义我们无法确定，不知道哪个符号是NP，哪个是PP等等．这个缺陷使得该算法难以在NLP中得到较好的应用．

因此，我们认为，如果将I/O算法用于训练一个文法的概率参数而不是生成一个文法，将在计算和实际应用上更加可行和合理．即，如果给定一个设计好的文法，我们可以通过该算法，从未加工的语料中获取该文法的概率参数．为此，我们将公式(1)、(2)、(4)分别变为：

$$e(s,s,i) = \begin{cases} b[i,O(s)] & \text{if } i \to O(s) \in G \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

$$e(s,t,i) = \sum_{i \to jk}\sum_{r_0=s}^{t-1} a[i,j,k]e(s,r_0,j)e(r_0+1,t,k), \text{for all } i \text{ and } s \neq t \tag{8}$$

$$f(s,t,i) = \sum_{j \to ki}\sum_{r_0=1}^{s-1} f(r_0,t,j)a[j,k,i]e(r_0,s-1,k) + \sum_{j \to ik}\sum_{r_0=t+1}^{T} f(s,r_0,j)a[j,i,k]e(t+1,r_0,k) \tag{9}$$

可见，我们所做的修改使得最外层的求和从计算所有可能的非终结符的组合变为只计算文法中有的规则．类似地，公式(5)(6)也只为文法中的规则计算概率参数．这样我们可以迭代地训练文法，以获取文法规则的概率参数．通过

修改，算法复杂度从 $O(N^3I^3)$ 降为 $O(MT^3)$，其中 M 为文法的规则的数目. 在大多数 NLP 系统中，M 比 N^3 要小得多，如我们在实验中使用的 CNF 形式的 MATRIX 系统的文法有 142 个非终结符，311 条规则. 我们的修改使得计算复杂度减少了 4 个数量级. 更重要的是，我们避免了原方法生成的文法难以理解和使用的缺陷. 这样，I/O 算法就能够方便地用于自然语言的文法了. 第 4 节给出了实验结果.

2 归约概率及其估算方法

我们知道，CFG 的规则所描述的文法符号之间的关系既可以理解为规则左边的符号推导出其右边符号串，也可以理解为是规则右边的符号串归约为其左边的符号. 到目前为止，实际中使用的 SCFG 的概率参数都是规则的推导概率，即由规则左边的符号推出右边的符号串的概率. 它反映的是一种从文法的开始非终结符向终结符推导的条件概率. 然而，在自然语言的语法分析这一问题中，我们面对的问题是：根据一个句子(即观察序列)，通过分析得到该句的语法分析树，也就是试图把句子从文法终结符归约到文法的开始符. 目前许多语法分析器采用的就是这种自底而上的分析方法，如广义 LR 分析器等，其分析过程是一种归约的过程而不是推导的过程. 因此我们提出了与推导概率不同的归约概率的概念，即规则右边的符号串归约到左边的符号的概率. 显然，归约概率和推导概率是两种意义不同的条件概率. 在 I/O 算法的基础上我们进一步提出了归约概率的计算方法，试图针对自然语言分析的特点，通过估算归约概率，获得更准确的概率参数.

首先重新定义概率矩阵 A 和 B 的意义. 我们依然用下式来表示矩阵 A 和 B：

$$a[i,j,k] = P(i \to jk|G), b[i,m] = P(i \to m|G)$$

其中 $P(i \to jk|G)$ 的意义与上节不同，它是非终结符 jk 归约到非终结符 i 的概率. 类似地，$P(i \to m|G)$ 是终结符 m 归约到非终结符 i 的概率，于是 A 和 B 描述的是规则的归约概率.

我们重新定义内(Inner)概率 e 如下：

$$e(s,t,i) = P(O(s)\ldots O(t) \overset{*}{\Rightarrow} i|G)$$

$e(s,t,i)$ 表示观察序列 $O(s)\ldots O(t)$ 经过若干步归约到非终结符 i 的概率.

重新定义外(Outer)概率 f 如下：

$$f(s,t,i) = P(O(1)\ldots O(s-1)\ i\ O(t+1)\ldots O(T) \overset{*}{\Rightarrow} S|G)$$

$f(s,t,i)$ 表示序列 $O(1)\ldots O(s-1)\ i\ O(t+1)\ldots O(T)$ 经过若干步归约到文法 G 的开始符号 S 的概率. 这里内外概率的计算方法需要在上节的计算方法(公式(3),(7),(8),(9))基础上，增加计算关于终结符的有关计算：

$$e(s,s,m) = \begin{cases} 1 & \text{if } O(s) = m \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

$$f(s,s,m) = \sum_{i \to m} f(s,s,i) \tag{11}$$

另外，为了估算归约概率，我们增加一个邻接外(Adjacent-outer)概率 af. 对于所有的满足 $,e(s,r,j)>0$ 且 $e(r+1,t,k)>0$ 的 s,t,r,j,k，我们定义

$$af(s,t,r,j,k) = P(O(1)\ldots O(s-1)\ jk\ O(t+1)\ldots O(T) \overset{*}{\Rightarrow} S|G)$$

$af(s,t,r,j,k)$ 表示序列 $O(1)\ldots O(s-1)\ jk\ O(t+1)\ldots O(T)$ 经过若干步归约到文法 G 的开始符号 S 的概率. 它可以按照下列公式进行计算：

$$\begin{aligned} af(s,t,r,j,k) = &\sum_{i \to jk} f(s,t,i)a[i,j,k] + \sum_{i \to hj}\sum_{r_0=0}^{r-1} af(r_0,t,r,i,k)a[i,h,j]e(r_0,s-1,h) \\ &+ \sum_{i \to hk}\sum_{r_0=t+1}^{T} af(s,r_0,r,j,i)a[i,k,h]e(t+1,r_0,h) \end{aligned} \tag{12}$$

其中 h 是非终结符.

在计算过程中，先求出内概率 e 和外概率 f，然后自顶向下地计算邻接外概率 af. 相应地，我们也修改了重新估算矩阵 A 和 B 的公式：

设 $$P = P(O \overset{*}{\Rightarrow} S|G) = e(1,T,S)$$

相应地，重新估算 $a[i,j,k]$ 和 $b[i,m]$ 的公式变为：

$$\hat{a}[i,j,k]=P(jk \text{ reduced to } i \mid jk \text{ occur})=\frac{P(jk \text{ reduced to } i, jk \text{ occur})}{P(jk \text{ occur})}$$

$$=\frac{\frac{1}{P}\sum_{s=1}^{T}\sum_{t=s+1}^{T}\sum_{r=s}^{t-1}a[i,j,k]e(s,r,j)e(r+1,t,k)f(s,t,i)}{\frac{1}{P}\sum_{s=1}^{T}\sum_{t=s}^{T}\sum_{r=s}^{t-1}e(s,r,j)e(r+1,t,k)af(s,t,r,j,k)} \quad \text{for all } i,j,k \tag{13}$$

$$\hat{b}[i,m]=P(m \text{ reduced to } i \mid m \text{ occur})=\frac{P(m \text{ reduced to } i, m \text{ occur})}{P(m \text{ occur})}=\frac{\frac{1}{P}\sum_{t\in O(t)=m}b[i,m]f(t,t,i)}{\frac{1}{P}\sum_{t\in O(t)=m}e(t,t,m)f(t,t,m)} \tag{14}$$

从上面的推导可以看到,估算归约概率的算法的时空复杂度是相当大的.但是在实际的计算过程中,当内概率 e 和外概率 f 计算之后,我们发现邻接外概率 af 有非常多的值为零的点,因此我们只计算和保存其中非零的元素,这样,算法对时间和空间要求大大减少,使得算法能够在我们的实验条件下得以实现.在第 4 节中我们可以看到,计算归约概率的算法所用的时间是计算推导概率的算法的 1.67 倍.

3　将一般 CFG 转化为 CNF 形式

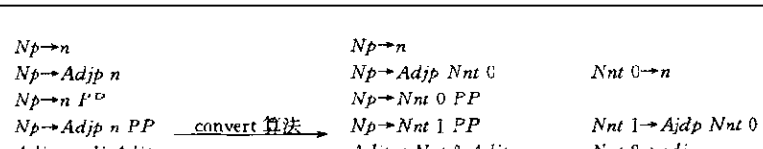
在上面的分析中,我们一直假定文法是 CNF 形式的.但在通常的 NLP 系统中,文法很少是 CNF 形式,而是采用一般形式的 CFG 文法.已经证明所有的 CFG 都可以转换成 CNF 形式.[8] 为了使 I/O 算法能够用于训练一般的 CFG 文法,我们必须将一般形式的 CFG 文法转变为 CNF 形式,并且建立二者规则之间的概率换算关系,以通过训练 CNF 形式的文法来获得原文法的概率参数.为此,我们设计了一个将一般 CFG 文法转化为 CNF 形式的算法 convert,它能够保持原文法结构,有利于方便地建立二者之间对应关系.

算法 convert 的简单描述如下:

设 S 表示原文法 G 的规则的集合;
T 表示由 G 转换成的 CNF 形式的文法规则的集合;

```
1  while S 非空 do
   {
     1.1  从 S 中取出规则 r,并将 r 从 S 中删除;
     1.2  switch 规则 r 的形式 :
          case i→jk:
            1.2.1 if j 是终结符 then
                    1.2.1.1  产生一个新的非终结符 x;
                    1.2.1.2  构造新规则:x→j 并加入 T;
                    1.2.1.3  将 r 中的 j 改成 x;
            1.2.2  if k 是终结符 then
                    1.2.2.1  产生一个新的非终结符 y;
                    1.2.2.2  构造新规则:y→k 并加入 T;
                    1.2.2.3  将 r 中的 k 改成 y;
            1.2.3  将规则 r 加入 T;
          case i→m:
            1.2.1  将规则 r 加入 T;
          case i→i1...in n≥2:
            1.2.1  产生一个非终结符 z;
            1.2.2  将规则 r 变为:i→z in;
            1.2.3  构造新规则:z→i1...in-1并加入 S;
            1.2.4  将规则 r 加入 S;
   }
2  结束.
```

另外,在我们的转换过程中,每当往 T 中加入一条规则,我们都将调用一个优化过程,这个过程可以对 T 中的所有规则进行优化,删除一些冗余的规则,保持规则集的紧凑,避免文法在转换过程中剧烈膨胀.图 1 给出了一个文法转换示例.



FG 文法 G CNF 形式的文法 G'

注：$Nnt0$，$Nnt1$，$Nnt2$和 $Nnt3$是新增的非终结符

图1 算法 convert 将文法 G 转换成 CNF 形式的文法 G'

在实验中，我们采用 convert 算法将一般形式的 CFG 转换为 CNF 形式，规则数目的增加并没有给我们的实验带来太大的困难(见第4节).

那么如何建立原文法和对应的 CNF 形式的文法的规则之间的概率换算关系呢?我们认为，换算关系的建立与文法的转换算法的设计是紧密相关的，不同的转换算法的概率换算关系也不相同. 为了说明我们的换算方法，先给出一个定义.

定义. 设 r 是原文法 G 中的规则，r_0是 G 转换成 CNF 形式后的文法 G'中的规则，若 r_0是 r 在转换过程中，经过算法 convert 的第1.2步若干次转换的操作而加入 G'的，我们称规则 r_0是规则 r 在 G'中的派生规则，并用 $D(r,G')$来表示规则 r 在 G'中的派生规则的集合.

根据条件概率的思想，我们给出概率换算关系如下：

设 G 是一般 CFG 文法，G'是 G 采用算法 convert 变换而生成的 CNF 形式的文法，对于任意规则 $r\in G$，我们有：

$$P(r|G)=\prod_{r_0\in D(r,G')}P(r_0|G')$$

即我们用规则 r 在 G'中的所有派生规则的概率的乘积作为它的概率估算结果. 以图1中的文法为例，我们用下面的等式计算原文法规则的概率：

$$P(Np\to n|G)=P(Np\to n|G')$$
$$P(Np\to Adjp\ n|G)=P(Np\to Adjp\ Nnt0|G')P(Nnt0\to n|G')$$
$$P(Np\to n\ PP|G)=P(Np\to Nnt0\ PP|G')P(Nnt0\to n|G')$$
$$P(Np\to Adjp\ n\ PP|G)=P(Np\to Nnt1\ PP|G')P(Nnt1\to Adjp\ Nnt0|G')P(Nnt0\to n|G')$$
$$P(Adjp\to adj\ Adjp|G)=P(Adjp\to Nnt2\ Adjp|G')P(Nnt2\to adj|G')$$
$$P(Adjp\to adj|G)=P(Adjp\to adj|G')$$
$$P(PP\to prep\ Np|G)=P(PP\to Nnt3\ Np|G')P(Nnt3\to prep|G')$$

4 实验结果及讨论

我们在586微机环境下对上述的研究进行了实现，并取得了较好的实验结果. 在实验中，以 MATRIX 英汉翻译系统的英文 CFG 文法作为训练的对象. 该文法是由219条规则，50个非终结符，27个终结符组成的，基本上覆盖了英文的基本句型. 首先使用算法 convert 将其转换成 CNF 形式，转换后的文法有311条规则，142个非终结符，转换造成的规则数目增加42%，转换效果良好. 我们使用北京大学的俞士汶教授提供的语料进行训练和测试. 其中，训练语料由503个英文句子构成，并被用来进行闭测试(Close-Test). 另外，我们从北大的语料库中随机选择了574个句子作为开测试(Open-Test)语料.

为了评价训练结果，我们使用 MATRIX 系统的广义 LR 语法分析器，将训练后的文法用来分析英文句子. 该语法分析器采用 Tomita 算法，能够处理非确定的 LR 文法，并生成所有的语法分析树.[9,10]文法规则的概率参数被用于对每一个分析树进行评价. Magerman 认为，用语法分析树中使用的所有规则的概率的几何平均值作为评分标准比用规则概率的乘积更合理.[11]所以，对于每棵分析树，我们把树中使用的所有规则的概率的几何平均值作为它的评分，评分最高的分析树作为语法分析结果.

实验中，我们先将 MATRIX 的文法转换为 CNF 形式，以均匀概率分布为迭代的初始模型(注意，均匀分布的意义对于推导概率和归约概率来说是不一样的，因此二者的初始模型的参数不同)，再分别对 CNF 形式的文法进行多次重估算，并通过上节的换算方法得到了原文法的规则的概率参数. 我们对文法的推导概率和归约概率分别进行了估算，

并将各次的迭代结果用于LR分析器，对测试语料和训练语料进行分析统计，所得的结果如下：

表1　MATRIX文法的推导概率和归约概率估算所需时间

估算概率类型	推导概率估算	归约概率估算
每次迭代平均耗时(s)	290	484

表2　句子概率P总和($\sum P$)变化

迭代次数	0	1	2	3	4	5	6	7
推导概率	3.896×10^{-4}	2.339×10^{-1}	2.810×10^{0}	2.848×10^{1}	2.854×10^{1}	2.855×10^{1}	2.856×10^{1}	2.855×10^{1}
归约概率	8.018×10^{-4}	7.841×10^{0}	1.122×10^{1}	1.188×10^{1}	1.198×10^{1}	1.201×10^{1}	1.202×10^{1}	1.202×10^{1}

表3　将MATRIX文法的推导概率和归约概率估算的各次迭代的结果用于语法分析的统计结果

	推导概率，闭测试		推导概率，开测试		归约概率，闭测试		归约概率，开测试	
迭代次数	错误句数	正确率(%)	错误句数	正确率(%)	错误句数	正确率(%)	错误句数	正确率(%)
0	196	61.93	197	65.68	160	68.19	199	65.33
1	106	78.93	134	76.66	86	82.90	126	78.05
2	98	80.52	132	77.00	83	83.50	117	79.62
3	97	80.72	125	78.22	81	83.90	113	80.31
4	97	80.72	125	78.22	80	84.10	115	79.97
5	97	80.72	125	78.22	80	84.10	115	79.97
6	97	80.72	125	78.22	81	83.90	117	79.62
7	97	80.72	125	78.22	82	83.70	115	79.97

从上面各表可以看出，不论推导概率还是归约概率的估算结果都能显著地提高语法分析的正确率．通过对实验结果进行分析，我们可以发现：

(1) 两种概率估算方法都使得训练语料的句子的概率之和$\sum P$逐步增大，分析正确率也显示逐步提高的趋势，这说明训练方法是有实用价值的．第1次迭代结果的提高最为显著，然后各次迭代的增幅逐渐降低，当$\sum P$的相对改变量小于0.5‰时，停止迭代．

(2) 当迭代进行到一定程度时，准确率提高很微小，甚至有略微的降低．理论上，I/O算法每次迭代，在最大相似意义上都能提高模型的准确性，即使得$P(S\Rightarrow O|G)$不断增大．但在句法分析时，追求的目标是使正确的分析树具有最大的概率值，而不是使句子的概率值(即所有的分析树的概率值之和)最大．二者既有联系，又互有区别．因此，尽管在理论上，每次迭代使得句子概率值增大，但难以保证分析准确率也相应的增大．类似现象在HMM词性标注中也有出现．[2]这个问题有待于进一步研究．

(3) 正如我们指出的一样，尽管归约概率的计算在理论上非常复杂，但是在实际的问题中，由于邻接外概率af非常稀疏，作为一种训练的方法来说，其实际的运算量仍然可以接受．如表1所示，归约概率所用的时间比推导概率增加了67%．

(4) 不论是闭测试还是开测试，归约概率的估算结果都要优于推导概率估算，这与我们提出归约概率的思想相吻合．我们认为，在大多数NLP的CFG分析器中，如广义LR分析器，都是面对句子(即观察序列)，采用自底向上的分析策略，通过使用文法的规则，从终结符序列逐步归约到文法的开始符号．这种使用规则的过程是一种归约的，而不是推导的过程，因此，归约概率估算的结果更符合语法分析的特点，用这种概率参数作评价标准更加合理和准确．实验的结果也证明了我们的观点．

5　结束语

本文提出了文法的推导概率和归约概率的概念，并通过修改I/O算法，实现了从未分析的语料中获取一般形式的CFG文法规则的概率参数．实验结果表明，将I/O算法用于训练文法的概率参数是有效的，而且归约概率的估算结果更加适合自然语言的语法分析．

致谢　感谢北京大学俞士汶教授为我们的研究提供了语料．

参考文献

1 Baum L. E., Eagon J A. An inequality with application to statistical estimation for probabilistic functions of Markov process and to a model for ecology. Bulletin of the American Mathematicians Society, 1967,73:360～363

2 Merialdo B. Tagging English text with a probabilistic model. Computational Linguistics, 1994,20(2):155～171

3 Baker J K. Trainable grammars for speech recognition. In: Klatt D H, Wolf J J eds. Speech Communication Papers for the 97th Meeting of the Acoustical Society of American. 1979. 547～550

4 Shih H-H, Young S J, Waeger N P. An inference approach to grammar construction. Computer Speech and Language, 1995,9: 235～256

5 Kupiec J. Hidden Markov estimation for unrestricted stochastic context-free grammars. In: Proceedings of the International Conference on ASSP. 1992.1:177～180

6 Black E, Lafferty J, Roukos S. Development and evaluation of a broad-coverage probabilistic grammars of English language computer manuals. In: Proceedings of the Association for Computational Linguistics. 1992. 117～121

7 Lari K, Young S J. The estimation of stochastic context-free grammars using the inside-outside algorithm. Computer Speech and Language. 1990.4:35～56

8 Chomsky N. On certain properties of grammars. Information and Control, 1959,2:137～167

9 Tomita M. Efficient parsing for natural language. Norwell, Massachusetts, USA: Kluwer Publishers, 1986

10 Shi Xiao-dong. Machine translation —— a practitioner's approach[Ph. D. Thesis]. National University of Defence Technology, 1994

11 Magerman D, Marcus M. Pearl: a probabilistic chart parser. In: Proceedings of the 2nd International Workshop on Parsing Technologies. Cancun, Mexican, 1991. 193～199

A Method of Training Grammar with Unanalyzed Corpus

WANG Ting SHI Xiao-dong CHEN Huo-wang YANG Yi

(*Department of Computer Science National University of Defense Technology Changsha* 410073)

Abstract The authors proposed the concept of reduce probability and derive probability. With the modifications on the I/O algorithm, the probability parameters of general context free grammar can be trained from unanalyzed corpus.

Key words Inside-outside algorithm, derive probability, reduce probability, corpus.

Class number TP301