文章编号：1003-0077(2010)06-0014-09

基于序列标注模型的分层式依存句法分析方法

鉴 萍，宗成庆

（中国科学院 自动化研究所 模式识别国家重点实验室，北京 100190）

摘 要：该文提出了一种全新的分层式依存句法分析方法。该方法以依存深度不大于 1 的依存层作为分析单位，自底向上构建句子的依存结构。在层内，通过穷尽搜索得到层最优子结构；在层与层之间，分析状态确定性地转移。依存层的引入，使该模型具有比典型的基于图的方法更低的算法复杂度，与基于转换的方法相比，又一定程度上缓解了确定性过程的贪婪性。此外，该方法使用典型序列标注模型进行层依存子结构搜索，证明了序列标注技术完全可以胜任句法分析等层次结构分析任务。实验结果显示，该文提出的分层式依存分析方法具有与主流方法可比的分析精度和非常高的分析效率，在宾州树库上可以达到每秒 2 500 个英语单词。

关键词：依存句法分析；依存层；序列标注

中图分类号：TP391　　**文献标识码**：A

Layer Based Dependency Parsing by Sequence Labeling Models

JIAN Ping, ZONG Chengqing

(National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Science, Beijing 100190, China)

Abstract: A layer-based projective dependency parsing approach is presented. This novel approach works layer by layer in a bottom-up manner, in which the depth of token dependency is allowed no more than one. Inside the layer the dependency graphs are searched exhaustively while between the layers the parser state transfers deterministically. Taking the dependency layer as the parsing unit, the proposed parser has a lower computational complexity than graph-based models which search for a whole dependency graph, alleviating the error propagation in transition-based models to some extent. Furthermore, our parser adopts the sequence labeling models to find the optimal sub-graph of the layer, which demonstrates the sequence labeling techniques qualified for hierarchical structure analysis tasks. Experimental results indicate that the proposed approach offers desirable accuracies and especially a very fast parsing speed, with 2500 words per second for Penn Treebank.

Key words: dependency parsing; dependency layer; sequence labeling

1 引言

目前，基于图的分析模型[1-2]和基于转换的（也称为基于分析动作的）分析模型[3-4]是主流的数据驱动依存句法分析方法。McDonald 和 Nivre[5]详细比较了这两类模型并给出了二者因算法上的本质差别而导致的分析性能上的差异。简单来说，以最大生成树方法为代表的基于图的依存分析将一棵依存树的打分看作是所有依存关系分值的总和，把依存分析问题转化为如何寻找得分最高的依存树。此方法在全句范围内穷尽搜索，算法的时间复杂度在投射性条件下是 $O(n^3)$（n 是输入句子所含词的个数）；基于转换的方法使用分类器选择当前分析状态下最可能的分析动作来实现状态的确定性转移，以得到输入句子的最优依存树，每一步决策只作用于

收稿日期：2010-03-22　**定稿日期**：2010-05-18

项目基金：国家自然科学基金资助项目(60975053,60736014)；国家 863 计划资助项目(2006AA010108-4)

作者简介：鉴萍(1982—)，女，博士生，主要研究方向为句法分析；宗成庆(1963—)，男，博士，研究员，主要研究方向为自然语言处理理论与方法、机器翻译和人机对话技术。

当前格局(configuration)的焦点词对上。由此看来，基于图的方法和基于转换的方法分别以整个句子和一个词对为搜索最优结构的基本单位，是依存分析的两个极端。本文中，我们将引入一个处于中间位置的结构单元——依存层，来建立句法分析模型。

这里所指的依存层由在依存树中依存关系深度不大于 1 的词组成。输入句子的依存结构用依存层分割开来，便有可能采用不同的处理方式。在层内，我们穷尽搜索这些词之间的最优依存关系组合；在层与层之间，已得到的依存结构可以被确定性地传递。这种做法一方面可以降低搜索整棵树所带来的计算代价，另一方面则能减轻决策的完全确定性所导致的错误传递。因为是将树结构分解为层结构，本方法适用于分析满足投射性条件的语言。

我们知道，语块分析可以作为完全句法分析一个有效的前处理单元，而序列标注技术已能很好地处理这类任务[6-7]。受此启发，我们尝试用序列标注模型来构建依存层最优子结构，将依存句法分析问题转化为一系列典型序列标注问题进行求解，从而考查序列标注模型求解层次结构问题的能力。

为方便起见，我们将这种基于序列标注模型的分层式方法称作基于层的依存分析方法。在英汉标准树库上的实验表明，基于层的分析方法能够获得与主流方法可比的分析精度，但在分析效率上有很大优势。在相同系统环境下，分析速度可达其他方法现有实现的十几到上百倍。

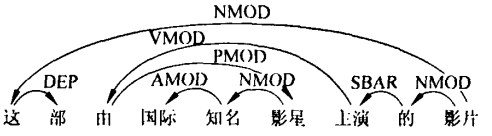
文章组织如下：在第 2 节中详细阐述基于层的依存分析方法及相关问题；第 3 节给出实验结果和相应的分析；与现有方法的比较在第 4 节中给出；第 5 节是结论和未来工作。

2　基于层的依存分析方法

2.1　算法描述

Wu 等[8]设计了一种相邻依存关系分析器(Neighbor parser)来识别句子中相邻两个词之间的依存关系。我们依照类似的方式将词之间的依存关系表示成序列形式，如图 1 所示。

图 1 中，第一列和第二列分别是句子的词和 POS(part of speech)标注，第三列表示该位置的词是否依存于它相邻的词：如果依存于它左边相邻的词，则用“LH”(left-headed，即“父节点在左边”)表示；如果依存于右边的词，则是“RH”(right-headed)；否则，标记为“O”。下划线“_”后面的字串是依存关系成立时相应的依存关系类型。



这	DT	O
部	M	LH_DEP
由	P	O
国际	NN	RH_AMOD
知名	JJ	RH_NMOD
影星	NN	O
主演	VV	RH_SBAR
的	DEC	RH_NMOD
影片	NN	O

图 1　相邻依存关系分析器

Wu 等使用可以得到全局最优解的条件随机场(conditional random fields，CRFs)模型来标注相邻词间依存关系，以避免采用序列分类器进行标注[9]所带来的决策的贪婪性。另外，考虑到直接输送相邻关系分析结果中的父节点到后续过程可能造成错误传递，Wu 在他们的模型中仅把该结果看作特征加入后续的分析器。实际上，这种做法没有充分发挥相邻关系分析对句法结构预测的能力。在我们的方法中，所有依存关系都由这种相邻关系分析来建立。

首先，从构建树结构的角度出发，我们简单地为词序列中每一个与其父节点相邻并且已是完整子树的节点标注依存关系，如图 2 所示。

这	DT	O
部	M	LH_DEP
由	P	O
国际	NN	RH_AMOD
知名	JJ	RH_NMOD
影星	NN	O
主演	VV	O
的	DEC	O
影片	NN	O

图 2　相邻依存关系标注方式 I

这些词(如图 2 例中的“部”、“国际”和“知名”)被归约到其父节点上，而未建立依存关系的词则进入下一层继续分析，直到形成一棵树。对输入句子的依存分析被分解到自底向上排列的若干个分析层，每一层都有部分依存结构被建立。因为被标注

了父节点的词都在当前层被归约，建立的依存结构在后续的分析中将不再改变。这也使得后续的分析可以利用已生成的依存结构。我们同样使用CRF标注器来保证层依存结构的最优性。

序列标注模型特别是计算因子之间具有不确定性的CRF模型，不能很好地捕捉序列中的长距离依存关系。使用高阶模型可以起到一定的缓和作用，但又带来极大的计算代价。因此，我们认为只有在前一步的依存层分析中就已找到自己所有孩子的词是可归约的，在当前层依存关系分析后其子树才完整的节点将不予归约。这在相邻依存分析中表现为：连续同方向的依存关系中，只归约最低层的孩子节点。采取这种策略可以使得由于依存距离较长而标注错误的词在后续的分析中有修正的机会。如图3(a)所示，名词“影星”和其父亲“由”相隔较远，在相邻依存关系分析中，它有可能错误依存到它右边的动词“主演”上。如果我们训练标注器归约所有找到父亲并且是完整子树的节点，那么词“国际”、“知名”和“影星”都将被归约掉，“影星”失去了与“由”发生关系的机会；如果标注器每次只归约该层分析之前就已是完整子树的节点，则只有词“国际”的依存关系被确立(“知名”和“影星”都是在本层分析中才找到其所有孩子的)，“影星”便有机会与“由”毗邻并得到正确的依存结果，如图3(b)。

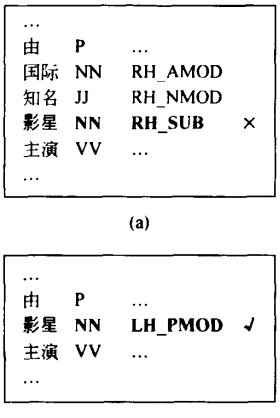


图3 相邻依存关系分析中的长距离关系依存错误

因为依存关系只发生在相邻词之间，连续同方向依存关系发生时只建立最低层关系，所以每个层内的依存深度不大于1。图4是以此设计的新的依存标注方式。

比较图4和图1，可以看出图4中的标记“O”存在一定的歧义。它可以表示当前词与相邻的词没有

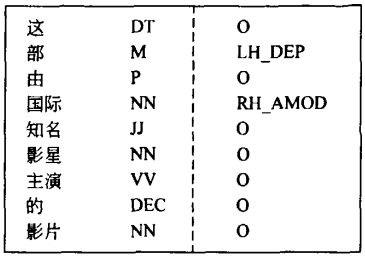


图4 相邻依存关系标注方式Ⅱ

依存关系，也可以表示当前词与相邻的词有依存关系但是暂时无法被归约而只能标记为“O”。原因可能是未找齐它的所有子节点(如词“主演”和“的”)或受连续同方向依存关系只归约最低层策略的约束(如词“知名”)。为消除这种歧义，我们回归到图1的形式，只要相邻的词之间存在依存关系，就进行标注。树结构的形成，则依靠额外的归约决策标注器来标记可以被归约的词。如图5(a)所示，最后一列的“r”表示当前词可以被归约，其依存结构在该层被建立。

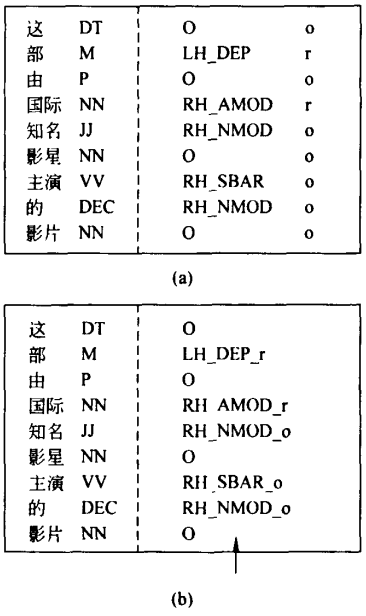


图5 相邻依存关系标注方式Ⅲ

为了保证解的全局最优性，归约决策标注器也使用CRF模型。实际上，这种依次标注并不是真正的全局最优。因此我们把依存关系和归约决策整合成一组标记，用“_”连接，如图5(b)所示，使用一个标注器来建立当前序列的依存结构。这样做的另一个原因，是避免所做的归约决策过于依赖依存关系

标注结果(归约决策要以依存关系为特征才能达到较好的效果)。

从另一个角度看，上述标注方式更符合对“相邻”依存关系分析的直观理解，而且将归约决策独立出来，也使加入的归约约束表达地更加清晰。整个句法分析过程可以用一个流程图来表示，如图 6。

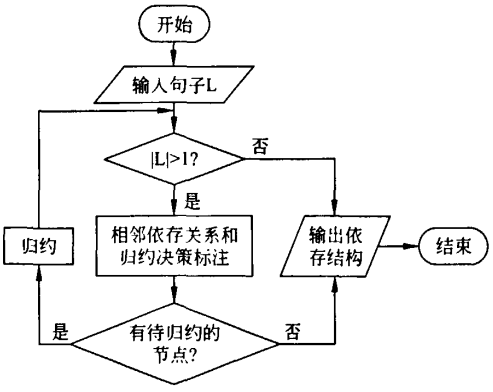


图 6　分析算法流程图

输入句子经相邻关系和归约决策标注后，部分词被归约为其相邻词的孩子，余下的词重新组合进入下一步分析，直到只剩下一个词或没有被标记为可归约的词。

根据此流程，图 5 例句的部分后续分析过程简单表示如图 7 所示。

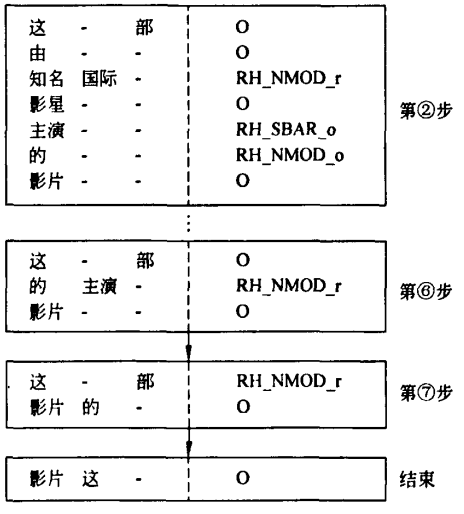


图 7　图 5 例句的后续分析过程

加上图 5 中刻画的第一层分析，输入例句共需要 7 步即 7 个依存层得到最后的分析结果。图 7 中第二列和第三列给出的是前一层分析归约掉的孩子节点，因为依存关系只发生在相邻词之间，它们分别是到目前为止当前词最左边和最右边的孩子节点。本文的实验只使用这一部分孩子节点作特征，这与一些使用基于转换方法的相关工作[4,10]的做法相同。

2.2　终止策略——*n*-best 标注结果的使用

上述算法在无可归约节点(没有节点标记为“r”)时便退出分析过程，但是由于训练语料包含每一个分析层，标记为“O”或“_o”的实例远多于标记为“_r”的实例，算法更倾向于将节点标记为“不具有依存关系”或“不归约”。我们使用标注器的多输出(*n*-best)结果来处理这个问题。

以图 5(b)所示联合标注为例，我们规定：如果当前序列长度大于 1 但是没有词被标记为可归约，分析过程仍继续，我们选取次优标注结果；如果该次优结果显示仍然没有词可归约，则退回到原最优标注序列，依据最优序列是否有词被标记为具有依存关系来决定是否强制归约。目前，我们主要使用 CRFs 提供的 2-best 输出，实验结果显示可以达到预期的效果。

2.3　特征集合

作为典型的序列标注问题，分析器使用与文献[6]中浅层句法分析类似的特征，列于表 1。包括词

表 1　标注器特征集合

词特征	$w[-3]$, $w[-2]$, $w[-1]$, $w[0]$, $w[1]$, $w[2]$, $w[3]$ $p[-3]$, $p[-2]$, $p[-1]$, $p[0]$, $p[1]$, $p[2]$, $p[3]$ $p[-2]\cdot p[-1]$, $p[-1]\cdot p[0]$, $p[0]\cdot p[1]$, $p[1]\cdot p[2]$, $p[-1]\cdot p[0]\cdot p[1]$ $w[-1]\cdot p[-1]$, $w[0]\cdot p[0]$, $w[1]\cdot p[1]$
子节点特征	$w\_lc[0]$, $w\_rc[0]$ $p\_lc[-1]$, $p\_rc[-1]$, $p\_lc[0]$, $p\_rc[0]$, $p\_lc[1]$, $p\_rc[1]$ $p[-1]\cdot p\_lc[-1]$, $p[-1]\cdot p\_rc[-1]$, $p[0]\cdot p\_lc[0]$, $p[0]\cdot p\_rc[0]$, $p[1]\cdot p\_lc[1]$, $p[1]\cdot p\_rc[1]$ $typ\_lc[-1]$,$typ\_rc[-1]$,$typ\_lc[0]$, $typ\_rc[0]$, $typ\_lc[1]$, $typ\_rc[1]$
关系特征	$rel[-3]$, $rel[-2]$, $rel[-1]$, $rel[0]$, $rel[1]$, $rel[2]$, $rel[3]$

特征、子节点特征和关系特征。中括号内的数字标志该特征所取自的词与焦点词的相对位置，“·”表示两个原子特征的组合；w 和 p 分别是词形(word)和 POS 标注。lc 和 rc 分别表示在当前分析状态节点词最左边和最右边的孩子节点，其依存关系类型用 typ 表示。rel 表示该层依存关系标注的结果，只在依次标注模型中使用。为保证算法的高效性，这里只使用一阶 CRFs。

Cheng 等[9]认为在自底向上的确定性句法分析系统中，低层所使用的分析策略和特征要与高层区分开，因为对高层来说，分析的对象更像是一个“短语”而不是词。在我们的系统中，也将第一层和其他层区别对待，采用分离模型：训练语料中，句子第一层的实例被单独取出来训练针对第一层的分析模型，所有层的实例用来训练针对其他层的分析模型。针对第一层分析的标注器不训练子节点特征。

3 实验结果与分析

3.1 实验设置

实验分别在英语和汉语标准树库上进行。

英语数据取自宾州树库(Penn Treebank)[11]的《华尔街日报》语料，采用标准划分：02～21 节做训练，22 节作为开发集，23 节测试(共 56 684 词)。使用和文献[12]相同的中心词提取规则和依存关系类型集合，均由 Penn2Malt[13] 工具提供，得到的依存结构共含有 12 种依存关系类型。开发集和测试集中的 POS 标注由软件 MXPOST[14] 自动获得，并在测试集上达到 97.05%的标注正确率。

汉语数据取自宾州中文树库 5.0(Penn Chinese Treebank 5.0)[15]，树库的划分参照文献[16]，以平衡各种语料来源。划分后的训练集共 16 079 句，开发集 803 句，测试集 1 905 句(共 50 319 词)。树库的转化同样参照文献[12]，使用 Penn2Malt 提供的针对汉语的中心词提取规则和依存关系类型集合。开发集和测试集采用树库中的标准分词和 POS 标注。

以下是实验中用于比较的五个基线系统：

1. MaltParser[17]：文献[18]中描述的基于转换的依存分析方法的典型实现，它采用“移进－归约”的一次分析方法，具有线性的分析时间 $O(n)$，这里使用的是 MaltParser 的 1.1 版本；

2. Yamada03：我们实现的另一种基于转换的依存分析方法[3]，同样采用“移进－归约”算法，对输入句子自底向上多次分析得到句法树，时间复杂度为 $O(n^2)$；

3. MSTParser1：MSTParser[19]是基于图的方法[1,2]的典型实现，这里使用的是 0.2 版本，MSTParser1 指的是其一阶模型；

4. MSTParser2：MSTParser 的二阶模型；

5. Duan07：Duan 等[16]提出的概率动作模型。该方法在 Yamada 和 Matsumoto[3]提出的基于转换的模型之上全局搜索最优分析动作序列，使用束搜索算法和支持向量机(SVMs)学习算法。时间复杂度为 $O(BKn^2)$，B 是束搜索的宽度，K 是其动作短语模型转移性动作的步数。

对于本文提出的基于层的分析方法(LDParser)，这里共比较以下五种：

1. LDP2：图 5(a)所示依次标注法；

2. LDP：图 5(b)所示联合标注法；

3. LDPdiv：图 5(b)所示联合标注法，第一层和其他层采用分离模型；

4. LDPnrAll：图 2 所示不归约则不标注依存关系的简单标注方法，在当前层成为完整子树的节点也被归约。与 LDPdiv 一样采用分离模型；

5. LDPnr：图 4 所示不归约则不标注依存关系，但只归约当前层之前就已是完整子树的节点。与 LDPdiv 一样采用分离模型。

实验结果的评价指标我们采用常用的无标记依存正确率(UAS)、带标记依存正确率(LAS)、根正确率(RA)和完全匹配率(CM)。其中根正确率指的是所有根节点识别正确的句子所占比例。除完全匹配率以外，所有指标均不将标点符号统计在内。此外，我们还给出了各种方法的时间复杂度(Comp.)和其实现工具的分析时间(Testing time)(CPU 时间)。所有实验在同一计算机环境(64 位 Xeon 处理器，2.5GHz，64G 内存)下进行。

3.2 主要结果

在英语数据集上我们比较了除 Duan07 以外的所有系统。对于 MaltParser，arc-eager 算法[18]被选为句法分析算法，特征集合与在 Hall 等[12]实验中取得最好结果的 Ψ_5 组合一致。Hall 等还指出在该特征集上 SVM 算法在句法分析精度和速度上都要比基于记忆的学习算法(MBL)优越，对于汉语也是如此。所以本文实验均使用 SVMs 作为学习算法。另外，我们还将分类器分割策略用于 MaltParser 以

求获得更快的分析速度，按照下一个输入词的 POS 标注将原 SVM 分类器分割成多个小分类器，每个分类器训练实例数量的下限是 1 000。对于 Yamada03，特征选取的窗口宽度为 6，特征集同时还加入了孩子节点的依存关系类型，其 SVM 分类器是按左焦点词的 POS 标注划分后分别训练的。对于 MSTParser，我们重复了文献[1]和[2]的实验。

考虑到训练代价，实验中五种基于层的句法分析模型除了依次标注模型以外都只使用出现了两次以上的特征，而且表 1 中的联合子节点特征和词形与 POS 标注的联合特征也被忽略了。

表 2 和表 3 是五个基线系统以及各种基于层的分析模型在英语数据集上的实验结果。时间复杂度项中的 R 表示基于层的模型中所有在训练实例中出现的依存关系标记的个数。基于 Viterbi 算法的标准 CRF 序列模型复杂度为 $O(R^2n)$。在基于层的方法中，对一个输入句子的分析最多需要 n 层，因此算法复杂度为 $O(R^2n^2)$。

表 2　各模型英语分析结果

分析器	UAS/%	LAS/%	RA/%	CM/%	Comp.	Testing time
MaltParser	89.68	88.48	84.73	33.69	$O(n)$	3hr 54min
MaltParser (split)	89.52	88.19	84.81	33.77	$O(n)$	14min 35sec
Yamada03 (split)	89.59	88.72	88.11	34.15	$O(n^2)$	18min 20sec
MSTParser1	91.03	89.78	94.21	35.72	$O(n^3)$	5min 48sec
MSTParser2	91.72	90.46	94.41	39.53	$O(n^3)$	8min 39sec
LDPdiv	89.68	88.43	89.16	33.90	$O(R^2n^2)$	1min 18sec

表 3　各基于层的模型英语分析结果

分析器	UAS /%	LAS /%	RA /%	CM /%	Testing time
LDP2	88.60	87.34	87.96	31.13	27sec
LDP	89.16	87.91	88.70	32.62	82sec
LDPdiv	89.68	88.43	89.16	33.90	78sec
LDPnrAll	87.20	85.85	83.92	24.91	17sec
LDPnr	89.24	87.95	87.31	32.46	22sec

在相比较的模型中，基于层的方法和基于转换的方法在依存正确率和完全匹配率上相似，但根识别正确率要高于基于转换的方法。需要指出的是，MaltParser 提供了多种可选参数项，我们实验得到的结果并不代表它的最好性能。文献[12]的英语实验使用的数据集和我们的相同，其公布的该方法的最好结果是 89.4%(UAS)。虽然得到该结果的自动 POS 标注正确率是 96.5%，要低于我们的 97.05%，但其分析器的参数经过了优化。

依赖于全句范围内的搜索，MSTParser 实现的基于图的方法在英语分析中得到最好的结果。但是考虑到分析效率，基于层的方法有很大优势。将搜索范围缩小到一个依存层，复杂度比基于图的方法低(在投射性条件下)，相应的有更快的分析速度：在五种基于层的分析模型中具有最高精度的 LDPdiv 每秒钟可以分析 700 个词以上；不归约则不标注依存关系的 LDPnr 甚至可以达到每秒 2 500 个词。基于转换的方法虽然可以有线性的复杂度，但是被认为是分析效果最好的 SVMs 并不是一种非常高效的分类算法[20-21]，特别是当类别个数非常多的时候。采用分类器分割或其他加速方法[22]，或换用另外的分类器可以改善这一问题。但是，即使考虑到这些因素，基于层的方法其分析速度仍十分可观。况且很多提升分类器速度的策略往往是以牺牲分析精度和消耗更多内存为代价的。比较表 2 中两个 MaltParser 系统的分析结果也可以看出这一点。

分析表 3 中的数据，可以看出依次标注法(LDP2)因为不具有结构搜索的全局最优性，分析精度较差；采用分离模型能很好地适应不同层的特点，从而得到较好的性能；LDPnrAll 归约包括在当前层形成完整子树的节点，相当于层内可能有深度大于 1 的依存结构，高估了线性序列标注的分析能力，因此性能最差；而 LDPdiv 与 LDPnr 相比，因为消除了标记"()"的歧义，分析精度有一定提高。这五种模型分析效率的差异分属三种情况：联合标注模型和依次标注模型的差别是复杂度的一部分因子由乘积变为加和，复杂度降低；使用额外归约决策标注器的 LDP 和 LDPdiv 与只使用关系标注器的 LDPnr 和 LDPnrAll，差别在于标注集缩小，效率也就提高

了；另外，LDPnrAll 每次归约的词要比 LDPnr 多，一个句子需要的分析层数相应减少，所以 LDPnrAll 更快。实际上，建立在序列标注模型之上的基于层的依存分析方法有很强的适应性，当语料标注集较大(通常是树库依存关系类型较多)或对效率要求较高时，可以通过标注拆分或缩减标注集来满足要求；另外，因为自然语言词语的聚集性(即大部分是短距离依存)，分析进入高层，序列的平均长度会急剧减小，即使输入句子本身很长，也不会出现难以接受的时间消耗。

在汉语实验中，我们又加入了与 Duan07 的比较。这里 MaltParser 选择使用 *arc-standard* 算法[18]，因为通过对开发集的测试我们发现，对汉语的分析 *arc-standard* 算法要优于 *arc-eager* 算法，CoNLL2007 共享任务中的 MaltParser[23] 处理汉语时同样选择了 *arc-standard* 算法。特征集依然是参考 Hall 等[12]的实验，选用 Φ_5 组合。包括 Yamada03 在内，分类器设置和英语实验一致。对于 MSTParser，汉语的分析模型训练设置为只使用 1-best 句法树。

LDParser 模型只考虑出现了一次以上的特征。实验结果列于表 4。

表 4 各模型汉语分析结果

分析器	UAS/%	LAS/%	RA/%	CM/%	Testing time
MaltParser (split)	83.82	82.15	73.54	32.55	36min
Yamada03 (split)	83.91	82.44	70.38	31.32	46min 11sec
MSTParser1	83.39	81.75	70.76	26.30	10min 3sec
MSTParser2	85.23	83.47	75.70	31.81	15min 33sec
Duan07	84.38	82.94	71.28	32.17	8hr 21min
LDPdiv	83.44	81.89	70.29	29.66	1min 19sec
LDPnr	83.23	81.68	70.22	29.16	22sec

在汉语实验中，基于层的方法与一阶 MSTParser 准确率相当，比基于转换的方法稍差一些。原因是基于转换的方法有较强的特征表达能力，更适合形态变化少的汉语的分析。这也是为什么基于层的方法在汉语分析中能与一阶 MSTParser 相比：基于层的方法也可以使用已生成的句法结构，虽不及基于转换的方法有效，但比 MSTParser 在特征表达上有一定优势。Duan07 在 Yamada03 之上全局搜索最优分析动作序列，得到了比 Yamada03 等更好的分析结果①。

和在英语数据集上的实验一样，基于层的方法使用了最少的时间。LDPdiv 模型在当前系统环境下每秒钟分析多于 600 个词，而且与其他方法的差距比英语实验中要大。比如，在汉语数据集上 LDPdiv 的分析速度是 MaltParser 的 27 倍和 MSTParser2 的约 12 倍，而在英语数据集上这两个值分别是 11 和不到 7。原因可能是汉语测试集的平均句长(每句 26.4 个词)大于英语的平均句长(每句 23.5 个词)。分层结构和相邻依存关系分析模式使基于层的句法分析能更高效地处理长句。在原本就低效的基于 SVM 分类的动作序列上加入全局搜索，Duan07 是所有参与比较的分析器中最慢的一个。

总之，基于层的依存分析方法在分析精度上与现有主流方法有可比性，其准确率处于基于图的方法和基于转换的方法之间，但在分析速度上有极大优势。特别是用于处理大规模语料时，这一优势将非常突出。

3.3 其他结果和分析

为进一步分析 LDParser 的特性，我们在英语数据集上分别测试了各方法对不同长度依存关系的分析能力，列于表 5。项“=1”表示相邻依存关系，

表 5 各长度依存关系精度比较(UAS/%)(英语)

分析器	=1	≤3	>3
MaltParser	94.24	93.09	73.92
MSTParser2	94.67	93.59	83.23
LDPdiv	94.56	93.07	74.53

① 这里公布的 Duan07 分析结果与文献[16]不一致，是因为我们使用了不同的依存树转换规范。

“≤3”和“>3”分别指依存距离不大于和大于 3 个词的依存关系。这里以“3”作为分界点是考虑到实验使用的英语树库平均依存距离是 3.28。

LDParser 的相邻关系分析精度与其他两个模型不相上下，说明全局搜索的序列标注模型可以像层次分析模型一样，很好地判断相邻词间的依存关系，甚至有比基于转换的方法更高的精度。对于长距离的依存关系，LDParser 性能也比 MaltParser 稍好，这说明基于层的方法能一定程度上缓解基于转换的方法中确定性过程的贪婪性。

通过将计算因子扩展到两个相邻的依存边以保存更多的上下文信息，二阶 MSTParser 无论在短依存还是长依存关系上都有较好表现。因为使用了已生成结构信息，LDParser 对短距离依存关系的判断能力与 MSTParser 相近，但随着依存关系距离的增大，LDParser 的精度下降严重。原因是基于层的方法仍然是一种确定性的分析方法，无法摆脱错误传递对长距离依存关系分析的影响。另一个原因是目前的 LDParser 版本对所有高层实例使用同一个模型，没有各自建模，削弱了高层分析模型的排歧能力。

我们还考查了各模型对长句的分析能力。英语测试集中长度大于 40 个词的共 171 个句子(8 019 词)作为测试数据，结果列于表 6。测试时间项中同时列出了分析器分析长句时速度下降的百分比。

表 6　长句分析结果比较(英语)

分析器	UAS/%	RA/%	Testing time
MaltParser (split)	87.34	71.92	149sec (17%)
MSTParser2	89.84	94.74	167sec (56%)
LDPdiv	86.58	78.95	13sec (15%)

LDParser 的精度与 MaltParser 相当。因为使用全句搜索，MSTParser 虽然是准确率最高的一个，但也是句子变长时效率下降最严重的一个(减慢了约 56%)。平均句长由原来的 23.5 增加到 46.9，LDParser 的分析速度只降低了 15%，进一步说明了基于层的方法虽然引入了最优结构搜索，对长句仍有较好的鲁棒性，可以高效地进行长句分析。

4　与相关工作的比较

作为一种自底向上的层次分析方法，本文提出的基于层的句法分析与基于转换的 Yamada 和 Matsumoto[3] 的方法在依存树归约形式上有些相似。但是他们的模型每一次由左至右分析词序列，针对每一个分析状态做决策，算法是完全确定性的和贪婪的。为了克服此类方法决策的贪婪性，Duan 等[16] 保留部分或所有可能的分析动作序列，搜索概率最大者来分析输入句子。同样，基于层的方法也通过加入全局搜索来寻找依存关系的最优组合，但是搜索空间只限制在一个依存层内，而 Duan 的方法则和基于图的模型一样为整个图空间打分。另外，他们继承了基于转换的方法所采用的层次结构分析机制(即使用分析动作建立依存关系)，基于层的方法则引入了序列标注技术。

目前有很多工作致力于两种主流方法的融合算法研究。Sagae 和 Lavie[24] 建立图模型重新分析一个“组合”起来的依存图，图中边的分值(即依存关系打分)来自另外三个基于转换的分析器。Hall 等[10] 采用相似的方法融合六个基于转换的分析模型。这类方法可以看作是对基于图和基于转换方法的结构投票。Nivre 和 McDonald[25] 提出了一种更有效的融合算法，将其中一种模型的输出看作是另一模型的特征。但是，所有这些融合系统都只是使用了子系统的输出结果，并没有改变它们的结构或算法。而基于层的分析方法对二者的继承，不仅引入了全新的结构而且采用了不同的分析算法。

Cheng 等[9] 和 Wu 等[8] 使用相邻依存关系分析器来提高确定性句法分析系统的性能。在 Cheng 的方法中，相邻关系由 SVM 序列分类器标注，父节点直接输送到后续的分析。Wu 使用 CRFs 进行关系分析，并用分析结果扩充特征集。这些方法中的相邻关系分析实际上只是充当完全句法分析的一个前处理单元，而我们的方法自始至终都是使用这种分析形式，并且证明它完全可以胜任层次结构预测任务。

5　结论

我们提出了一种全新的分层式依存句法分析方法。它以依存层为分析单位，在层内，像一个基于图的模型搜索最优子结构；在层之间，又像一个基于转换的模型确定性地传递已建立的依存结构。自底向上的分层结构、相邻关系分析模式和一阶 CRF 序列标注技术的应用使分析器在保证与主流方法可比的分析精度前提下，具有非常高的分析效率。

对分析器进行改进的首要工作是提高其分析精

度。像 Duan[16]的方法那样保留多个依存标注结果并加入层间搜索，是目前来看最有潜力的途径。此外，优化高层分析模型可能是另一种有效的手段。

参考文献

[1] R. McDonald, K. Crammer and F. Pereira. Online large-margin training of dependency parsers [C]// Proc. of ACL. Ann Arbor, USA: 2005, 91-98.

[2] R. McDonald and F. Pereira. Online learning of approximate dependency parsing algorithms[C]//Proc. of EACL. Trento, Italy: 2006, 81-88.

[3] H. Yamada and Y. Matsumoto. Statistical dependency analysis with support vector machines[C]//Proc. of IWPT. Nancy, France: 2003, 195-206.

[4] J. Nivre and M. Scholz. Deterministic dependency parsing of English text [C]//Proc. of COLING. Switzerland: 2004, 64-70.

[5] R. McDonald and J. Nivre. Characterizing the errors of data-driven dependency parsing models[C]//Proc. of EMNLP-CoNLL. Prague, Czech: 2007, 122-131.

[6] F. Sha and F. Pereira. Shallow parsing with conditional random fields[C]//Proc. of NAACL. Edmonton, Canada: 2003, 213-220.

[7] 李珩，朱靖波，姚天顺. 基于 SVM 的中文组块分析[J]. 中文信息学报，2004，18(2)：1-7.

[8] Y. Wu, J. Yang and Y. Lee. Multilingual deterministic dependency parsing framework using modified finite Newton method support vector machines[C]//Proc. of EMNLP-CoNLL. Prague, Czech: 2007, 1175-1181.

[9] Y. Cheng, M. Asahara and Y Matsumoto. Multi-lingual dependency parsing at NAIST [C]//Proc. of CoNLL. New York City, USA: 2006, 191-195.

[10] J. Hall, J. Nilsson, J. Nivre, et. al. Single Malt or blended? A study in multilingual parser optimization [C]//Proc. of EMNLP-CoNLL. Prague, Czech: 2007, 933-939.

[11] M. P. Marcus, B. Santorini and M. A. Marcinkiewicz. Building a large annotated corpus of English: the Penn Treebank [J]. Computational Linguistics, 1993, 19(2): 313-330.

[12] J. Hall, J. Nivre and J. Nilsson. Discriminative classifiers for deterministic dependency parsing[C]// Proc. of COLING-ACL. Sydney, Australia: 2006, 316-323.

[13] http://w3. msi. vxu. se/~nivre/research/Penn2Malt. html[CP/OL].

[14] A. Ratnaparkhi. MXPOST[CP/OL]. http://www. inf. ed. ac. uk/resources/nlp/local_doc/MXPOST. html.

[15] N. Xue, F. Xia, F. Chiou et, al. The Penn Chinese Treebank: phrase structure annotation of a large corpus[J]. Natural Language Engineering, 2005,11(2): 207-238.

[16] X. Duan, J. Zhao and B. Xu. Probabilistic models for action-based Chinese dependency parsing [C]// Proc. of ECML-PKDD. Warsaw, Poland: 2007, 559-566.

[17] J. Nivre, J. Hall and J. Nilsson. MaltParser: a data-driven parser-generator for dependency parsing [C]//Proc. of LREC. Genoa, Italy: 2006, 2216-2219.

[18] J. Nivre. Incrementality in deterministic dependency parsing[C]//Proc. of ACL. Barcelona, Spain: 2004, 50-57.

[19] R. McDonald. MSTParser[CP/OL]. http://www. seas. upenn. edu/~strctlrn/MSTParser/MSTParser. html.

[20] Y. Cheng, M. Asahara and Y. Matsumoto. Machine learning-based dependency analyzer for Chinese[C]// Proc. of ICCC. Mauritius: 2005, 66-73.

[21] M. Wang, K. Sagae and T. Mitamura. A fast, accurate deterministic parser for Chinese[C]//Proc. of COLING-ACL. Sydney, Australia: 2006, 425-432.

[22] Y. Goldberg and M. Elhadad. SplitSVM: fast, space-efficient, non-heuristic, polynomial kernel computation for NLP applications[C]//Proc. of ACL. Columbus USA: 2008, 237-240.

[23] J. Hall, J. Nilsson and J. Nivre. MaltParser in the CoNLL shared task 2007-Multilingual track [EB/OL]. http://maltparser. org/conll/conll07/.

[24] K. Sagae and A. Lavie. Parser combination by reparsing[C]//Proc. of NAACL. New York City, USA: 2006, 129-132.

[25] J. Nivre and R. McDonald. Integrating graph-based and transition-based dependency parsers[C]//Proc. of ACL. Columbus, USA: 2008, 950-958.

基于序列标注模型的分层式依存句法分析方法

作者： 鉴萍， 宗成庆， JIAN Ping， ZONG Chengqing
作者单位： 中国科学院,自动化研究所模式识别国家重点实验室,北京,100190
刊名： 中文信息学报 ISTIC PKU
英文刊名： JOURNAL OF CHINESE INFORMATION PROCESSING
年，卷(期)： 2010,24(6)

参考文献(25条)

1. R.McDonald,K.Crammer and F.Pereira.Online large-margin training of dependency parsers[C]//Proc.of ACL.Ann Arbor,USA:2005,91-98.

2. R.McDonald and F.Pereira.Online learning of approximate dependency parsing algorithms[C]//Proc.of EACL.Trento,Italy:2006,81-88.

3. H.Yamada and Y.Matsumoto.Statistical dependency analysis with support vector machines[C]//Proc.of IWPT.Nancy,France:2003,195-206.

4. J.Nivre and M.Scholz.Deterministic dependency parsing of English text[C]//Proc.of COLING.Switzerland:2004,64-70.

5. R.McDonald and J.Nivre.Characterizing the errors of data-driven dependency parsing models[C]//Proc.of EMNLP-CoNLL.Prague,Czech:2007,122-131.

6. F.Sha and F.Pereira.Shallow parsing with conditional random fields[C]//Proc.of NAACL.Edmonton,Canada:2003,213-220.

7. 李珩,朱靖波,姚天顺.基于SVM的中文组块分析[J].中文信息学报.2004.18(2):1-7.

8. Y.Wu,J.Yang and Y.Lee.Multilingual deterministic dependency parsing framework using modified finite Newton method support vector machines[C]//Proc.of EMNLP-CoNLL.Prague,Czech:2007,1175-1181.

9. Y.Cheng.M.Asahara and Y Matsumoto.Multi-lingual dependency parsing at NAIST[C]//Proc.of CoNLL.New York City,USA:2006,191-195.

10. J.Hall,J.Nilsson,J.Nivre,et.al.Single Malt or blended? A study in multilingual parser optimization[C]//Proc.of EMNLP-CoNLL.Prague,Czech:2007,933-939.

11. M.P.Marcus,B.Santorini and M.A.Marcink-iewicz.Building a large annotated corpus of English:the Penn Treebank[J].Computational Linguistics,1993,19(2):313-330.

12. J.Hall,J.Nivre and J.Nilsson.Discriminative classifiers for deterministic dependency parsing[C]//Proc.of COLING-ACL.Sydney,Australia:2006,316-323.

13. http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html[CP/OL].

14. A.Ratnaparkhi.MXPOST[CP/OL].http://www.inf.ed.ac.uk/resources/nlp/local_doc/MXPOST.html.

15. N.Xue,F.Xia,F.Chiou et,al.The Penn Chinese Treebank:phrase structure annotation of a large cor-pus[J].Natural Language Engineering,2005,11(2):207-238.

16. X.Duan,J.Zhao and B.Xu.Probabilistic models for action-based Chinese dependency parsing[C]//Proc.of ECML-PKDD.Warsaw,Poland:2007,559-566.

17. J.Nivre,J.Hall and J.Nilsson.MaltParser:a data-driven parser-generator for dependency parsing[C]//Proc.of LREC.Genoa,Italy:2006,2216-2219.

18. J.Nivre.Incrementality in deterministic dependency parsing[C]//Proc.of

ACL. Barcelona, Spain: 2004, 50-57.

19. R. McDonald. MSTParser[CP/OL]. http://www.seas.upenn.edu/~ strctlrn/MSTParser/MSTParser.html.

20. Y. Cheng, M. Asahara and Y. Matsumoto. Machine learning-based dependency analyzer for Chinese[C]//Proc. of ICCC. Mauritius: 2005, 66-73.

21. M. Wang, K. Sagae and T. Mitamura. A fast, accurate deterministic parser for Chinese[C]//Proc. of COLING-ACL. Sydney, Australia: 2006, 425-432.

22. Y. Goldberg and M. Elhadad. SplitSVM: fast, space-efficient, non-heuristic, polynomial kernel computation for NLP applications[C]//Proc. of ACL. Columbus USA: 2008, 237-240.

23. J. Hall, J. Nilsson and J. Nivre. MaltParser in the CoNLL shared task 2007-Multilingual track[EB/OL]. http://maltparser.org/conll/conll07/.

24. K. Sagae and A. Lavie. Parser combination by reparsing[C]//Proc. of NAACL. New York City, USA: 2006, 129-132.

25. J. Nivre and R. McDonald. Integrating graph-based and transition-based dependency parsers[C]//Proc. of ACL. Columbus, USA: 2008, 950-958.