第33卷 第24期 计 算 机 工 程 2007年12月
Vol.33 No.24 Computer Engineering December 2007
·人工智能及识别技术· 文章编号：1000—3428(2007)24—0212—03 文献标识码：A 中图分类号：TP391.4

基于远距离依存关系的中文依存关系解析

周惠巍，杨 洋，黄德根
（大连理工大学计算机科学与工程系，大连 116024）

摘 要：依据中文语法的特点，提出了Nivre算法和一种远距离依存关系的确定性中文依存关系解析方法。在中文句子中，有些相互依存的词距离较远，使用传统的确定性解析方法进行解析比较困难。在不忽略远距离依存关系的情况下进行确定性依存关系解析，采用支持向量机识别中文依存关系。实验结果表明，依存关系解析精度达到78.30%，提高了5.32%。

关键词：中文依存关系解析；Nivre算法；支持向量机

Chinese Dependency Analysis Based on Long-distance Dependency

ZHOU Hui-wei, YANG Yang, HUANG De-gen
(Department of Computer Science and Engineering, Dalian University of Technology, Dalian 116024)

【Abstract】 According to Chinese syntax and Nivre algorithm, this paper presents a deterministic dependency parser, which parses Chinese text with consideration of long-distance dependency. It is difficult to parse long-distance dependency with conventional deterministic dependency analysis method. The improved method parses a sentence deterministically without ignoring long-distance dependency. Support vector machines are applied to identify Chinese dependency. Experimental results show that the method outperforms previous system by 5.32% accuracy. The dependency accuracy achieves 78.30%.

【Key words】 Chinese dependency analysis; Nivre algorithm; Support Vector Machines(SVMs)

依存关系解析是句法分析的一个重要方法。依存关系可以明确地表明中心词之间的句法依存关系，并能方便地转化为语意依存描述。英文依存关系解析[1-2]与日语依存关系解析已经取得了较好的研究成果。近年来，由于大规模语料库的创建，使得人们利用机器学习方法构建中文依存关系解析器。许云基于支持向量机(SVMs)[3]构建了中文依存关系解析器[4]。首先将句子划分为若干个短语，然后基于SVMs计算句中每个短语与其后各个短语的依存关系，构建该句的依存关系矩阵。该方法假设句中各个短语依存且仅依存于其后方某个短语，而实际中文句中有些词依存于其前方某个词。并且，该方法需要计算句中每个短语与其后各个短语的依存关系，计算量较大。郑育昌将Nivre算法[1]和Yamada算法[2]应用于中文依存关系解析，基于最大熵和SVMs进行确定性依存关系解析[5]。郑育昌利用中文句中大多数词依存于其近旁词的语法特点，通过解析句子中各个词与其前后词的依存关系解析整个句子。该方法简单高效，取得了较高的依存关系解析精度。但是在中文句中，有些词距离其孩子节点较远，只解析句中各个词与其近旁词的依存关系，有时无法正确解析整个句子。

1 支持向量机(SVMs)

在支持向量机中，正负两类的训练集为

$$(\boldsymbol{x}_1,\boldsymbol{y}_1),\cdots,(\boldsymbol{x}_l,\boldsymbol{y}_l),\boldsymbol{x}_i\in R^n,\ \boldsymbol{y}_i\in\{+1,-1\}$$

其中，$\boldsymbol{x}_i$是数据i的n次特征向量，即$\boldsymbol{x}_i=(f_1,f_2,\cdots,f_n)\in R^n$；$\boldsymbol{y}_i$表示数据$i$是正例(1)、负例(-1)的类标。

SVMs是在n次空间构造一个最优分类超平面，可以转化为最优化以下目标函数。

$$\begin{cases}\min L(\boldsymbol{w})=\|\boldsymbol{w}\|^2\\ \text{s.t. } \boldsymbol{y}_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)\geq 1\ (i=1,2,\cdots,l)\end{cases}\tag{1}$$

引入Lagrange乘数$\alpha_i\geq 0(i=1,2,\cdots,l)$，核函数$K(x,y)=\varphi(x)\cdot\varphi(y)$。通过求解最优化问题得到最终的识别函数$f$，即

$$f(x)=\mathrm{sgn}\Big(\sum_{i,\boldsymbol{x}_i\in SVs}\alpha_i\boldsymbol{y}_iK(\boldsymbol{x}_i\cdot x)+b\Big)\tag{2}$$

其中，SVs为支持向量(support vectors)。

研究者提出了多种核函数，其中多项式核函数被广泛地应用于自然语言处理领域，并取得了较好的解析结果[2,4-5]。在实验中也采用了多项式核函数，即

$$K(\boldsymbol{x}_i,\boldsymbol{x}_j)=(\boldsymbol{x}_i\cdot\boldsymbol{x}_j+1)^d\tag{3}$$

其中，d次多项式核函数可以将d个以内的特征组合起来进行学习。在本文中d等于3。

2 依存关系解析模型

中文依存关系必须满足条件：(1)句中各个词能且仅能依存于句中的某个词；(2)一个句中各依存关系彼此不能交叉。

已有的研究表明，Nivre算法[1]更符合中文的语法特点[5]，本文基于Nivre算法进行中文依存关系解析。

2.1 Nivre 算法

在Nivre算法中，解析器可以表示成一个三元组$<S, I, A>$。S和I是堆栈，I中是待解析的输入序列。A是一个集合，

基金项目：国家自然科学基金资助项目(60373095, 60373096)
作者简介：周惠巍(1969－)，女，讲师、硕士，主研方向：自然语言处理；杨 洋，硕士研究生；黄德根，教授、博士
收稿日期：2006-12-23 **E-mail：**zhou_huiwei@163.com

存放在解析过程中确定下来的依存关系项。假设给定一个输入序列 W，解析器首先被初始化成$<nil, W, \varphi>$。解析器解析栈 S 的栈顶元素 t 和栈 I 的栈顶元素 n 的依存关系，然后采取相应的动作，操作栈中的元素移动，算法迭代进行直到栈 I 为空。当栈 I 为空时，解析器停止迭代，输出保存在集合 A 中的依存关系序列。

Nivre 算法一共定义了 4 个操作，采用哪种操作是由 SVMs 分类器确定：

(1)Right。在当前三元组$<t|S, n|I, A>$中，假如存在依存关系 $t\rightarrow n$，即 t 依存于 n，则在集合 A 中添加项$(t\rightarrow n)$，同时弹出 S 中的栈顶元素 t，于是三元组变为$<S, n|I, A\cup\{(t\rightarrow n)\}>$。

(2)Left。在当前三元组$<t|S, n|I, A>$中，假如存在依存关系 $n\rightarrow t$，则在集合 A 中添加项$(n\rightarrow t)$，同时把元素 n 压入栈 S 中，于是三元组变为$<n|\ t\ |S, I, A\cup\{(n\rightarrow t)\}>$。

在当前三元组$<t|S, n|I, A>$中，如果 n 与 t 不存在依存关系，解析器根据不同情况执行下列操作。

(3)Reduce。假如 I 中不存在任何元素 n'依存于 t，并且 t 有父亲节点在其左侧，解析器从栈 S 中弹出 t，于是三元组变为$<S, n|I,A>$。

(4)Shift。上述的 3 种情况都不满足时，将 n 压入栈 S 中，于是三元组变为$< n|t|S, I,A>$。

Nivre 算法的 Reduce 操作要求 I 中不存在任何元素依存于 t，而在实际的依存关系解析过程中，这一条件比较难于判定。实际的依存关系解析器往往应用确定性 Nivre 算法，只要 I 的栈顶元素 n 不依存于 t，并且 t 有父亲节点在其左侧，即可执行 Reduce 操作。

依据确定性 Nivre 算法进行依存关系解析，如果输入句长为 N，最多只需 $2N$ 个动作就可以完成解析。以图 1 中的依存关系例句为例，说明上述 4 种操作，见图 2。

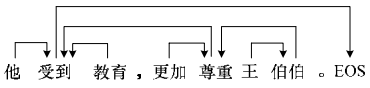

图 1　依存关系例句

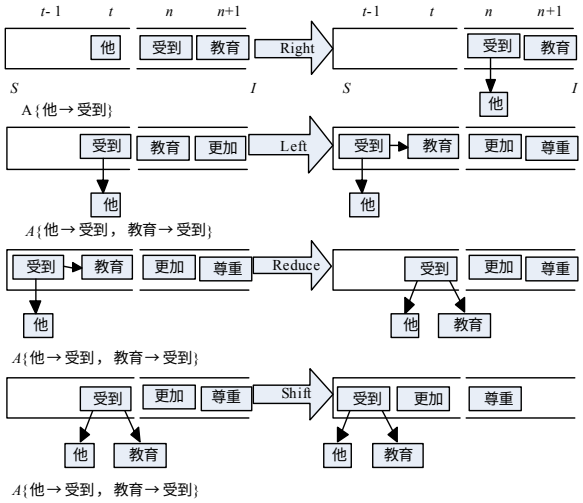

图 2　Nivre 算法的 4 种操作

确定性 Nivre 算法简化了 Reduce 操作的判定条件。但是 I 的栈顶元素 n 不依存于 t，并不能保证 I 中任何元素都不依存于 t。在实际的中文语料中，虽然大多数词依存于其近旁的词，仍然存在许多词距离其孩子节点较远。未经明确判定 t 不存在孩子节点，不应从栈 S 中弹出 t，而应执行 Shift 操作。否则，如果 t 存在孩子节点却被弹出，将不再被解析，随后会产生一些错误的依存关系解析结果。

因此，确定性 Nivre 算法对 Reduce 操作与 Shift 操作的划分不十分准确。针对此问题，提出一种改进的确定性 Nivre 算法。

2.2　考虑远距离依存关系的确定性 Nivre 算法

考虑远距离依存关系的确定性 Nivre 算法同样定义了 4 个操作。Right 与 Left 操作的定义没有改变。Reduce 与 Shift 操作重新定义如下：

(1)Reduce。假如两栈顶元素 n 与 t 不存在依存关系，t 有父亲节点在其左侧，并且该父亲节点与 n 存在依存关系，解析器从栈 S 中弹出 t，于是三元组变为$<S, n|I, A>$。

(2)Shift。当 Right, Left, Reduce 操作的条件都不满足时，将 n 压入栈 S 中，三元组变为$< n|t|S, I, A>$。

Reduce 操作的定义利用了依存关系不交叉的依存公理，从栈 S 中弹出 t，而不影响随后的解析。考虑远距离依存关系的确定性 Nivre 算法流程如下：

(1)解析器解析两栈顶元素 t 和 n 的依存关系，然后采取相应的动作，操作栈中的元素移动，直到 I 中为句子结束符(<EOS>)。如果 S 中只剩余 1 个元素，转至(3)。

(2)解析器继续对 S 中的剩余元素从栈顶开始解析，对于已经解析过的依存关系，不必重新解析。当执行 Shift 操作时，将 t 压入栈 I 中，三元组变为$<t\text{-}1|S, t|n|I, A>$。如果 S 中只剩余一个元素时，I 中剩余元素不只是句子结束符(<EOS>)，转至(1)。

(3)S 中剩余元素为该句的根节点，根节点指向句子结束符(<EOS>)，整句分析结束。

仍然以图 1 中的依存关系例句为例，说明考虑远距离依存关系的确定性 Nivre 算法解析过程，见图 3。

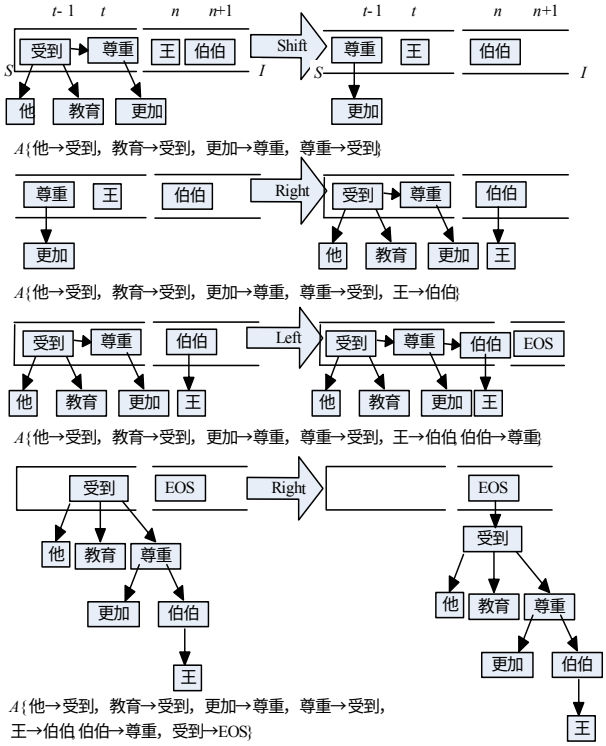

图 3　考虑远距离依存关系的确定性 Nivre 算法解析过程

考虑远距离依存关系的确定性 Nivre 算法对 Reduce 操作与 Shift 操作做了更加明确的划分，使解析过程更加准确。如

果采用从前的确定性 Nivre 算法解析图 3 中第一行 S 的栈顶元素“尊重”与 I 的栈顶元素“王”，会执行 Reduce 操作，将“尊重”弹出，不再解析。在随后的解析中，“尊重”的孩子节点“伯伯”将被错误地解析为依存于句中其他元素，使整个句子解析失败。

使用本文算法解析句长为 N 的句子，同样最多只需 $2N$ 个动作就可以完成解析，解析时间与原算法基本相同。

3 基于 SVMs 的中文依存关系解析

3.1 多值分类

SVMs 为二值分类器，而中文依存关系解析为多值分类问题。本文采用 pairwise 法将二值分类器扩展为多值分类器。

按照 pairwise 法将样本分为 4 类，共需构建 6 个分类器。为减少训练代价，本文首先将训练样本分为 Right、Left、不存在依存关系 3 类，训练生成 3 个二值分类器。然后依据 Reduce、Shift 的定义，对不存在依存关系的词作进一步划分。

3.2 特征

在 Nivre 算法中，解析器解析当前三元组的两个栈顶元素(t,n)的依存关系。可以选取 t 节点及其前两个节点，n 节点及其后两个节点，以及它们的孩子节点的特征，解析(t,n)的依存关系。Nivre 算法的特征向量如下：(1) 节点 t-2, t-1, t 的词；(2) 节点 t-2, t-1, t 的词性；(3)节点 t-2, t-1, t 的孩子节点的词；(4)节点 t-2, t-1, t 的孩子节点的词性；(5)节点 n, n+1, n+2 的词；(6)节点 n, n+1, n+2 的词性；(7)节点 n, n+1, n+2 的孩子节点的词；(8)节点 n, n+1, n+2 的孩子节点的词性；(9)节点 t 与 n 在句中的距离。

4 实验

4.1 实验结果

实验语料来自哈尔滨工业大学信息检索研究室加工的依存关系语料库的一部分，其中 1~4 000 句的语料用作训练，4 001~5 000 句的语料用作测试。语料平均句长为 22 个词。

采用以下 3 个解析精度评估依存关系解析器的性能：

$$\text{依存关系正确率}=\frac{\text{正确识别的依存关系个数}}{\text{所有依存关系个数}} \tag{4}$$

$$\text{根正确率}=\frac{\text{正确识别的依存树根节点个数}}{\text{测试集句子个数}} \tag{5}$$

$$\text{句子正确率}=\frac{\text{完全分析正确的句子个数}}{\text{测试集句子个数}} \tag{6}$$

分别使用确定性 Nivre 算法与本文算法进行依存关系解析。在封闭测试、开放测试中，依存关系解析结果见表 1、表 2。

表 1 封闭测试中依存关系解析结果 单位：%

算法	正确率	根正确率	句子正确率
Nivre 算法	90.54	40.93	39.45
本算法	97.64	93.15	82.93

表 2 开放测试中依存关系解析结果 单位：%

算法	正确率	根正确率	句子正确率
Nivre 算法	72.98	33.60	11.10
本算法	78.30	71.60	18.50

由表 1、表 2 的依存关系解析结果可见，使用考虑远距离依存关系的确定性 Nivre 算法进行封闭测试，依存关系解析正确率由 90.54%提高到 97.64%，几乎完全正确地解析了训练语料。根正确率(40.93%→93.15%)与句子正确率(39.45%→82.93%)也有很大的提高。表明准确定义了 Reduce 操作与 Shift 操作，显著地提高了依存关系解析结果，考虑远距离依存关系的确定性 Nivre 算法更符合中文的依存关系特性。开放测试的依存关系正确率提高到 78.30%，根正确率达到 71.60%，提高了 37.0%。

4.2 与以往方法的比较

许云构建了中文短语依存关系解析器，并假设句中各个短语依存且仅依存于其后方某个短语[4]。实验采用 Penn Chinese TreeBank，该树库是句法结构树，需自行转换为依存关系树。语料平均句长为 25 个词，短语数目应少于 25。训练语料同样为 4 000 句，任意另取 100 句作为测试语料，依存关系解析正确率为 77.3%。

郑育昌将确定性Nivre算法和确定性Yamada算法应用于中文依存关系解析，分别基于最大熵和SVMs构建了 4 个解析器[5]。实验采用台湾的CKIP Chinese Treebank(Version 2.0)，训练语料为 41 057 个短语结构，平均短语长为 5 个词，依存关系不复杂。其中基于SVMs的Nivre算法取得了较好的解析精度。教科书、报纸、文摘以及杂志的依存关系正确率分别达到 94.61%, 87.86%, 95.06%, 87.71%。

本文将郑育昌的基于 SVMs 的 Nivre 算法与改进后的基于远距离依存关系的确定性 Nivre 算法应用于哈尔滨工业大学的依存关系语料。结果表明，修改后的确定性 Nivre 算法较好地提高了依存关系解析性能。

5 结束语

本文依据中文的语法特点，提出了一种考虑远距离依存关系的确定性 Nivre 算法，并基于 SVMs 构建了依存关系解析器。实验表明，在中文依存关系解析中，考虑远距离依存关系的确定性 Nivre 算法与单纯的确定性 Nivre 算法相比，在解析正确率、根正确率以及句子正确率上都有较大的提高。基于远距离依存关系的确定性 Nivre 算法更能体现中文句法的特点，在没有增加解析复杂性的前提下，提高了依存关系解析精度。

参考文献

[1] Nivre J, Scholz M. Deterministic Dependency Parsing of English Text[C]//Proceedings of COLING'04. Geneva, Switzerland: [s. n.], 2004: 64-70.

[2] Yamada H. Statistical dependency Analysis with Support Vector Machines[C]//Proceedings of the 8th International Workshop on Parsing Technologies. Nancy, France: [s. n.], 2003: 195-206.

[3] Cortes C. Support Vector Networks[J]. Machine Learning, 1995, 20(3): 273-297.

[4] Xu Yun, Zhang Feng. Using SVM to Construct a Chinese Dependency Parser[J]. Journal of Zhejiang University Science A, 2006, 7(2): 199-203.

[5] Cheng Yuchang. Machine Learning-based Dependency Analyzer for Chinese[C]//Proceedings of the International Conference on Chinese Computing. Singapore: [s. n.], 2005: 66-73.