第 36 卷 第 21 期 计 算 机 工 程 2010 年 11 月
Vol.36 No.21 Computer Engineering November 2010
·人工智能及识别技术· 文章编号：1000—3428(2010)21—0217—03 文献标识码：A 中图分类号：TP391

基于 SVM 的汉语决策式依存分析

姚文琳，王玉丹
(中国海洋大学信息科学与工程学院，山东 青岛 266100)

摘 要：决策式分析有着贪婪的特性，容易引起错误增殖。针对该问题，提出一种基于 SVM 的汉语决策式依存分析算法。利用 SVM 构建根查找器，用根结点将句子划分为 2 个子句。从子句中识别出介词短语，采用改进后的 Nivre 算法分析子句。该算法在分析句子之前做预处理从而降低句子复杂度，减少错误增殖，分析准确率也相应得到提高。实验结果表明，该分析策略的准确率比 Nivre 算法提高了 3.38%。
关键词：决策式；依存分析；根；介词短语

Deterministic Dependency Parsing for Chinese Based on SVM

YAO Wen-lin, WANG Yu-dan
(College of Information Science and Engineering, Ocean University of China, Qingdao 266100, China)

【Abstract】 Deterministic parsing has the greedy characteristic that easily brings the error propagation. Aiming at this question, this paper proposes a deterministic dependency analysis algorithm for Chinese including three steps. It utilizes SVM to construct a root finder to divide a sentence into two sub-sentences and extracts the prepositional phrases from sub-sentence. Improved Nivre's algorithm is adopted to parse sub-sentence. It does pre-processing before parsing sentences to decrease the complexity of the sentence and reduce the error propagation, and improve the parsing accuracy consequently. Experimental evaluation shows the accuracy of this parsing strategy is higher by 3.38% than Nivre's.
【Key words】 deterministic; dependency parsing; root; prepositional phrase

1 概述

句法分析是自然语言处理的主要任务之一。近几年，依存句法分析得到了越来越广泛的关注。目前为止最有效的依存句法分析策略是基于分类器的训练算法和决策式的句法分析模型相结合。本文构建了一个支持向量机(Support Vector Machine, SVM)分类器和决策式分析相结合的汉语依存句法分析器。

支持向量机是基于 Large Margin 的分类算法，是学习分类器的有效方法。它是建立在统计学习理论的 VC 维理论和结构风险最小化原理基础上的，在高维特征空间具有很高的泛化性能，可以自由选取多种特征组合。另外引入核函数，解决了线性不可分以及高维特征空间稀疏问题，不会增加时间复杂度。

决策式分析是指以特定的方向逐步取一个待分析的词，为每次输入的词产生一个单一的分析结果，直到词序列的最后一个词。算法在每一步的分析中根据当前状态做出决策，具有鲁棒、有效和确定的特性。因为决策式算法在每一步做出决策后不可更改，属于贪婪算法，容易引起错误增殖，所以会在分析句子之前做预处理，减少错误增殖。

2 预处理阶段

本节实现了分析器的预处理工作。分析器做了 2 步预处理：首先构建根查找器，找到输入句子的根结点，用根结点将句子划分为 2 个子句；然后在子句上识别出介词短语。

2.1 根查找器构建

Robinson 在 1970 年提出，依存结构满足 4 条公理，即单一父结点、连通、无环和可投影，保证了句子的依存分析结果是一棵有根的树结构。每个句子有且仅有一个根结点，且两边的依存关系不能跨越根结点，这样根结点就可以把句子划分成独立的 2 个子句。分割句子可以降低句子的分析复杂度，理论上，短句的分析准确率要比长句的分析准确率高。文献[1]已经有这方面的相关工作。

通过下面的例子来描述这一情形。在图 1 中，原句有一个根结点“推动”，两边的依存关系均没有交叉，所以可以把原句分割成 2 个子句：“采取 有效 措施 推动”和“推动 公司 快速 发展”。然后分析子句得到依存子结构，最后把 2 个“推动”合二为一，合并 2 个子结构，就完成了整个句子的分析。

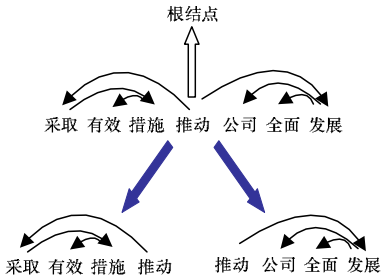

图 1 句子的分割

基金项目：国家自然科学基金资助项目“可伸缩中文语音合成系统的研究”(60602017)
作者简介：姚文琳(1967－)，女，副教授、硕士，主研方向：自然语言处理，人工智能；王玉丹，硕士
收稿日期：2010-06-14 **E-mail：**zixin-2008@163.com

用 SVM 来构建根查找器。由于 SVM 分类器只能识别数值型的对象，因此首先严格按照分类器所要求的数据格式对数据集进行了数据转换，然后再训练数据。根结点的查找是一个二类分类问题。如果当前词是根，则标记为 1，否则为 0。因为 SVM 能够处理高维特征空间，所以从语料库中提取出多个特征组合进行训练得到训练模型。表 1 是构建根查找器所用到的特征组合。

表 1　根查找器所需特征

特征	特征描述
W_n/pos_n	不同位置的词和词性，n=-2, -1, 0, 1, 2
W_n_first/W_n_last	当前词(w_n)是否位于句首或者句尾，是设为 1，否则为 0
V_left/V_right	w_n 与句首词或句尾词之间是否有动词，有设为 1，否则为 0
P_left	w_n 与句首词之间是否有介词，有设为 1，否则为 0
D_p_left	w_n 与左边的介词之间的距离，没有设为-1
Dec_right	w_n 与句尾词之间是否有“的”，有设为 1，否则为 0
D_dec_right	w_n 与右边的“的”之间的距离，没有设为-1

2.2　子句预处理——介词短语识别

从原有句法分析器可看出介词短语是引起分析器错误的主要因素之一。所以，首先识别出介词短语单独分析(分析算法和子句相同)，然后让其中心词代替整个短语参与子句分析。

英语在介词短语方面的研究主要是介词短语附加问题(PP Attachment)，文献[2]使用 preference learning 解决 PP Attachment；而汉语的介词短语研究主要是解决边界问题。因为介词短语的左边界总是介词本身，所以识别介词短语也就是识别介词短语的右边界。文献[3]利用 SVM 进行了组块分析，取得了很好的效果，本文同样使用 SVM 来识别介词短语。采用了 Tjong Kim Sang 在 1999 年定义的 IOB2 表示法来标识介词短语(PP)。

I：当前词位于 PP 内部。

O：当前词位于 PP 外部。

B：当前词位于 PP 开始位置，总是标识介词。

例如：我们/O 对/B 他/I 的/I 经历/I 一无所知/O 。/O

考虑了文献[2]用到的一些特征，并结合汉语的特点考察其他有效特征，经过多次测试，选用如表 2 所示的特征解决介词短语识别。

表 2　识别介词短语所需特征

特征	特征描述
W_n/pos_n	不同位置的词和词性，n = -2, -1, 0, 1, 2
P_is_found	是否已找到介词(w_p)，找到设为 1，否则为 0
Distance	w_p 与 w_n 之间的距离
W_i/pos_i	w_p 与 w_n 之间的词和词性
Quot_inside/ Pare inside	w_n 是否在介词之后的引号或括号内部，在内部设为 1，否则为 0
Num_com	w_p 与 w_n 之间标点符号的数目
Num_verb	w_p 与 w_n 之间动词的数目
Num_con	w_p 与 w_n 之间连接词的数目
Num_pre	w_p 与 w_n 之间介词的数目

3　子句分析阶段

本文实现的分析器的算法主要基于 Nivre 算法，首先介绍了 Nivre 算法，然后再根据汉语的特点对其进行改进。

3.1　Nivre 算法

Nivre 算法[4]是目前关于决策式句法分析最有代表性的算法之一，也是本文算法的基础。算法使用一个三元组<S,I,A>来描述分析结构，其中，S 是栈；I 是待分析词序列；A 是依存关系集合。算法定义了 4 种动作：Left-Arc(LA)，Right-Arc(RA)，Reduce(R)和 Shift(S)。

算法描述如图 2 所示(t 表示栈顶词，n 表示下一个输入词)。

Initialization	<nil, W, Ø>
Termination	<S, nil, A>
Left-Arc	<t | S, n | I, A>→<S, n | I, A∪{(n→t)}>
Right-Arc	<t | S, n | I, A>→<n | t | S, I, A∪{(t→n)}>
Reduce	<t | S, I, A>→<S, I, A>
Shift	<S, n | I, A>→<n | S, I, A>

图 2　Nivre 算法的 4 步操作

如果 n 支配 t，执行左依存 LA；如果 t 支配 n，执行右依存 RA；如果 t 与下一个词没有依存关系，且 t 有了父结点，执行归约 R；否则执行移近 S。举例说明这一分析过程，如图 3 所示。

我们 知道 你 是 最 好 的 学生 之一。

stack　　input tokens　　relation set A

Step-0: <nil，我们知道你是最好的学生之一，{}>

Step-1: ->S <我们，知道你是最好的学生之一，{}>

Step-2: ->LA <知道，你是最好的学生之一，{(知道→我们)}>

Step-3: ->S <你 知道，是最好的学生之一，{(知道→我们)}>

Step-4: ->LA <知道，是最好的学生之一，{(知道→我们)，(是→你)}>

Step-5: ->RA <是 知道，最好的学生之一，{(知道→我们)，(是→你)，(知道→是)}>

Step-6: ->R <知道，最好的学生之一，{(知道→我们)，(是→你)，(知道→是)}>

Step-7: ->S <最 知道，好的学生之一，{(知道→我们)，(是→你)，(知道→是)}>

Step-8: ->LA <好 知道，的学生之一，{(知道→我们)，(是→你)，(知道→是)，(好→最)}>

Step-9: ->LA <的 知道，学生之一，{(知道→我们)，(是→你)，(知道→是)，(好→最)，(的→好)}>

Step-10: ->LA <学生 知道，之一，{(知道→我们)，(是→你)，(知道→是)，(好→最)，(的→好)，(学生→的)}>

Step-11: ->LA <知道，之一，{(知道→我们)，(是→你)，(知道→是)，(好→最)，(的→好)，(学生→的)，(之一→学生)}>

Step-12: ->RA <之一 知道，nil，{(知道→我们)，(是→你)，(知道→是)，(好→最)，(的→好)，(学生→的)，(之一→学生)，(知道→之一)}>

输出结果:

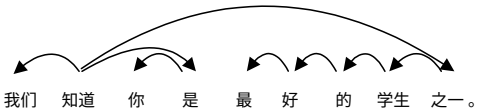

图 3　Nivre 算法的分析过程

Nivre 的算法是有缺陷的。尽管右依存并不马上归约，而是在下一步判断它与下一个输入词是否有依存关系，但这样只能处理相邻右依存问题，对长距离右依存无能为力。从图 3 也可看出，在 Step-6，“是”和“最”没有依存关系，并且“是”已经有了父结点“知道”，满足归约条件，于是执行归约操作，这样它就失去了成为“之一”的父结点的机会，产生了过早归约的问题。

正确的分析结果如图 4 所示。所以，本文根据汉语的特点对算法进行改进。

图 4　图 3 中的例子的正确分析结果

3.2 改进 Nivre 算法及特征

首先介绍分析器所用到的特征，包括栈顶词(t)和词性，下一输入词(n)和词性；还有一些上下文信息，即 t 前面的 2 个词和 n 后面的 2 个词的信息；还包括它们孩子的信息以及 t 和 n 之间的距离；t 和 n 之间的标点也考虑在内，没有则设为 0。把这些特征叫做局部特征。

3.1 节提到 Nivre 算法会导致过早归约，这是因为算法用于排歧的特征即局部特征所包含的上下文信息是有限的，缺乏长距离信息，在处理长距离依存问题时就容易出现错误。为了解决这一问题，考虑了长距离依存的信息。这里引进文献[1]定义的全局特征，即在 I 中未分析也没考虑在局部特征中的词就被认为是全局特征。

并不是在所有的操作中都要使用全局特征，因为它并不总是有效的，本文会对此做出判断。归约操作总是在确定右依存者时产生错误。汉语中，只有动词和介词有右依存者。因为已经识别了介词短语，所以只考虑动词的情况。当 t 是动词且分析器要执行归约时，推迟归约。将运用全局特征来判断在待分析词序列里是否还有 t 的依存者。如图 3 中的 Step-6，动词“是”由于缺乏长距离信息被归约掉，如果运用全局特征发现它还有依存者“之一”，就推迟归约“是”，执行 Shift 操作，让下一个词“最”进栈。为了区别于原有的 Shift 操作，把这一动作叫做 Verb_Shift(VS)。如图 5 所示，直到 Step-11“之一”找到正确的头。

我们 知道 你 是 最 好 的 学生 之一 。

…

Step-5: ->RA <是 知道，最好的学生之一，{(知道→我们)，(是→你)，(知道→是)}>

Step-6: ->VS <最 是 知道，好的学生之一，{(知道→我们)，(是→你)，(知道→是)}>

Step-7: ->LA <好 是 知道，的学生之一，{(知道→我们)，(是→你)，(知道→是)，(好→最)}>

…

Step-10: ->LA <是 知道，之一，{(知道→我们)，(是→你)，(知道→是)，(好→最)，(的→好)，(学生→的)，(之一→学生)}>

Step-11: ->RA <之一 是 知道，nil，{(知道→我们)，(是→你)，(知道→是)，(好→最)，(的→好)，(学生→的)，(之一→学生)，(是→之一)}>

输出结果:

图 5 改进算法的分析过程

4 实验结果

本文选用的实验数据来自哈工大公开中文依存语料库。选取其中的 4 000 句作为训练集，1 000 句作为测试集。本文的分类工具是采用文献[5]提供的软件包 Libsvm，用它来构建分类模型，得到分析动作。

为了评价系统的整体性能，本文主要采用了以下 2 个评价标准：

Dependency Accuracy (Dep. Acc.)=
标记正确头结点的数目/所有头结点的数目

Root Accuracy (Root Acc.)=标记正确的根的数目/句子总数

根据上述标准，得到以下测试结果，见表 3。

表 3 实验结果比较 (%)

算法	Dep. Acc.	Root Acc.
Nivre 算法	82.92	81.42
本文算法 (加上根查找器)	86.09	92.37
本文算法 (加上根查找器和介词短语识别)	86.30	92.37

从表 3 中的实验结果可以看出，本文分析器的准确率和根准确率(第 4 行)分别比 Nivre 的算法(第 2 行)高出了 3.38%和 10.95%。同时也分析比较了 2 步预处理对分析器的影响。首先，利用根结点把句子分割为 2 个子句，分析器的性能增加了 3.17%。当加入介词短语识别后，性能又增加了 0.21%，但增幅不大，可能是由于介词短语识别器的性能一般。可见，如果能够完全正确地识别出介词短语，分析器将会有很好的表现。

5 结束语

本文构建了一个基于 SVM 的汉语决策式依存分析器。为了解决决策式分析算法的贪婪性带来的错误增殖，加入了预处理部分。首先构建了 Root Finder，利用根结点来分割句子，分割句子可以降低句子的复杂度，使分析的准确率提高；然后在子句上识别介词短语，用 IOB2 表示法标识介词短语。最后运用改进后的 Nivre 算法分析子句。实验结果表明分析准确率得到显著提高。

笔者认为根查找器还有一定的发展空间，在以后的工作中，还应当继续提高其准确率。另外，介词短语的识别率不高，也会导致一些新的错误，影响了分析器的性能，所以，提高介词短语的识别率也是以后工作的重点。

参考文献

[1] Cheng Yuchang, Asahara M, Matsumoto Y. Chinese Deterministic Dependency Analyzer: Examining Effects of Global Features and Root Node Finder[C]//Proc. of the 4th SIGHAN Workshop on Chinese Language Processing. Jeju Island, Korea: [s. n.], 2005: 17-24.

[2] Isozaki H, Kazawa H, Hirao T. A Deterministic Word Dependency Analyzer Enhanced with Preference Learning[C]//Proc. of the International Conference on Computational Linguistics. Geneva, Switzerland: [s. n.], 2004: 275-281.

[3] 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, 2003, 18(2): 1-7.

[4] Nivre J, Scholz M. Deterministic Dependency Parsing of English Text[C]//Proc. of International Conference on Computational Linguistics. Geneva, Switzerland: [s. n.], 2004: 64-70.

[5] Lin C J. A Practical Guide to Support Vector Classification[EB/OL]. (2001-07-18). http://www.csie.ntu.edu.tw/~cjlin/libsvm/.

编辑 任吉慧