博士学位论文

汉语依存句法分析关键技术研究

RESEARCH ON KEY TECHNOLOGIES OF CHINESE DEPENDENCY PARSING

李 正 华

哈尔滨工业大学

2013年3月

国内图书分类号：TP391.2　　学校代码：10213
国际图书分类号：681.324　　密级：公开

工学博士学位论文

汉语依存句法分析关键技术研究

博士研究生：李正华

导　　师：刘　挺　教授

申请学位：工学博士

学　　科：计算机应用技术

所在单位：计算机科学与技术学院

答辩日期：2013年3月

授予学位单位：哈尔滨工业大学

Classified Index: TP391.2
U.D.C: 681.324

Dissertation for the Doctoral Degree in Engineering

RESEARCH ON KEY TECHNOLOGIES OF CHINESE DEPENDENCY PARSING

Candidate: Zhenghua Li
Supervisor: Professor Ting Liu
Academic Degree Applied for: Doctor of Engineering
Specialty: Computer Application Technology
Affiliation: School of Computer Science and Technology
Date of Defence: March, 2013
Degree-Conferring-Institution: Harbin Institute of Technology

摘 要

依存句法分析的目标是分析输入句子的句法结构，将词语序列转化为树状的依存结构。一条依存弧两个词语构成搭配关系，依存弧上的标签表示搭配的具体类型，如主语、宾语、状语等。在不同语法体系中，依存语法以其形式简洁、易于标注、便于应用等优点，逐渐受到研究人员的重视。计算自然语言学习国际会议（CoNLL）联合举办的公开评测任务从2006年至2009年连续四年关注面向多语言的依存句法分析，大大推动了依存句法分析的发展。依存句法分析也越来越广泛的应用于机器翻译、问答系统、文本挖掘、信息检索等。

依存句法分析的研究工作旨在提高依存分析的准确率和效率。高准确率的分析结果可以为上层应用提供更可信的句法结构。随着互联网数据的迅速膨胀，上层应用系统需要迅速处理海量的信息，因此效率对于依存句法分析的应用也很关键。本文的研究内容涵盖这两个问题，包括以下四个方面。

1. 提出了基于柱搜索和标点切分的快速高阶依存句法分析方法。针对前人提出的面向高阶依存句法分析模型的动态规划解码算法时间复杂度高的问题，本文提出使用柱搜索的近似解码算法，一方面允许模型可以方便的融入丰富的高阶句法子树特征，另一方面保证较低的时间复杂度，我们实现的基于柱搜索的高阶依存句法分析系统在CoNLL 2009年多语依存句法分析和语义角色标注联合评测任务上取得了优异的成绩。进而，我们针对汉语的特点，提出一种利用标点符号进行长句切分的二阶段依存句法分析方法，进一步提高依存句法分析模型处理长句时的效率。实验证明，这种方法可以大幅度提高依存句法分析的速度，长句子的句法分析准确率也有提高。

2. 提出了汉语词性标注和依存句法分析联合模型。由于缺少词语的形态变化信息，汉语词性标注和其他语言如英语相比，准确率较低。这对对汉语依存句法分析带来严重的错误蔓延问题。实验表明使用自动词性时依存句法分析准确率比使用正确词性时低大约6%。对此，本文提出并深入系统的研究了汉语词性标注和依存句法分析联合模型。首先，我们扩展了前人提出的面向依存句法分析的解码算法，提出了相应的面向联合模型的基于动态规划的解码算法。并且，为了解决联合解码算法的时间复杂度过高的问题，我们又提出了一种有效地基于边缘概率的词性裁剪方法。实验结果表明联合模型可以提高词性和句法准确率。深入的错误分析表明联合模型可以帮助消解句法敏感的词性歧义。

3. 提出了面向联合模型的分离被动进取训练算法。词性标注和依存句法分

析联合模型中句法特征占据主导地位，导致词性特征无法贡献其消歧作用。对此，本文提出一种面向词性标注和依存句法分析联合模型的训练算法。算法分别不同的步长对词性特征和句法特征的权重进行更新。和传统的平均感知器和被动进取训练算法相比，分离被动进取训练算法可以很自然的增大词性特征的权重，从而更好的平衡联合模型中词性特征和句法特征的消歧作用。实验发现，我们的基于分离被动进取训练算法的联合模型在汉语和英语数据上都可以取得最好的词性和句法准确率。

4. 提出了基于准同步文法的多树库融合方法。汉语存在多个异构树库，而利用多个树库以提高依存句法分析准确率是一个非常有吸引力的课题。本文提出一种基于准同步文法的多树库融合方法，充分利用标注规范不同的多个单语树库，以提高句法分析准确率。我们设计了丰富的转换模式来刻画不同标注规范间的对应规律，然后基于这些转化模式形成准同步文法特征，从而增强基准依存句法分析模型。准同步文法特征用来指导句法模型做出更好的决策，并且可以很自然的融入到基于图的句法分析解码算法中。实验结果表明，我们的方法可以充分利用源树库的知识。从而提高句法模型在目标树库上的准确率。

总之，本文针对汉语特点，深入研究了基于标点的快速高阶依存句法分析方法、词性标注和依存句法分析的联合模型和多树库融合问题，大大提高了汉语依存句法分析处理实际文本数据的效率和准确率。本研究取得了一些初步的成果。我们期待这些研究成果可以进一步推动自然语言处理领域和其他上层应用如机器翻译、信息抽取的发展。

关键词： 依存句法分析；柱搜索；联合模型；分离被动进取训练算法；多树库融合

Abstract

Dependency parsing aims to analyze the syntactic structure of a given sentence, and converts the word sequence into a tree. A dependency consists of two words, and the one word modifies the other one. The label of the dependency represents the specific relation between the two words, such as subject, object, adverbial modifier, and so on. Among different syntax formulizations, the dependency grammar has gained more and more interest in parsing community due to its characteristics: (1) representation simplicity, (2) easy to annotate, (3) easy to make use of. The international conference of Computational Natural Language Learning (CoNLL) has organized shared tasks on multilingual dependency parsing from 2006 to 2009, which largely promotes the research on dependency parsing. Meanwhile, dependency parsing has been more extensively applied to machine translation, question answering, text mining, information retrieval, and so on.

Research on dependency parsing has two important goals. One goal is to improve the parsing accuracy, while the other is to improve the parsing efficiency. Accurate parsing results can provide reliable syntactic structures for higher-level applications. Along with the quick growth of the web data, higher-level applications need to process a large amount of information in some limited time. Therefore, parsing efficiency is also important. This thesis covers the two issues and consists of four parts.

1. We propose a fast high-order dependency parsing method based on beam search and punctuation. The previously proposed decoding algorithm for high-order dependency parsing is based on dynamic programming and has high time complexity. To address this issue, we propose a beam-search based decoding algorithm which on one hand allows the model to incorporate rich high-order syntactic features, and on the other hand is able to find the approximate optimal parse tree under lower time complexity. Our beam search based high-order dependency parser attended the CoNLL 2009 shared task on multilingual dependency parsing and semantic role labeling and achieved good results. To further improve the parsing efficiency for long sentences, we analyze the characteristics of Chinese and propose to use punctuation to segment an input sentence into several subsentences and then apply two-stage dependency parsing. Experimental results show that this punctuation-based two-stage parsing method can largely improve the parsing speed for

long sentences. Meanwhile, the parsing accuracy on long sentences is also substantially increased.

2. We propose joint models for Chinese POS tagging and dependency parsing. Due to little morphological changes, Chinese POS tagging accuracy is much lower than other languages like English. This leads to severe error propagation for Chinese dependency parsing. Our experiments show that parsing accuracy drops by about 6% when replacing manual POS tags of the input sentence with automatic ones generated by a state-of-the-art statistical POS tagger. To address this issue, this paper proposes a solution by jointly optimizing POS tagging and dependency parsing in a unique model. 1) We propose for our joint models several dynamic programming based decoding algorithms by extending the decoding algorithms for dependency parsing. 2) A novel and effective pruning strategy based on marginal probabilities is presented to reduce the search space of candidate POS tags. Experimental results show that our joint models significantly improve both the state-of-the-art tagging and parsing accuracies. Detailed analysis shows that the joint method can help resolve syntax-sensitive POS ambiguities.

3. We propose a separately passive-aggressive training algorithm for joint models. Joint models for POS tagging and dependency parsing are dominated by syntactic features. As a result, the POS features fails to fully contribute their disambiguation power. To solve this issue, we propose a separately passive-aggressive learning algorithm (SPA), which is designed to separately update the POS features weights and the syntactic feature weights with different update steps under the joint optimization framework. Compared with the traditional training algorithms averaged perceptron (AP) and passive aggressive (PA), SPA can naturally raise the weights of the POS features, and therefore better balance the discriminative power of the POS and syntactic features of the joint models. Experimental results show that our joint models trained with SPA achieve the best tagging and parsing accuracy on both Chinese and English datasets.

4. We propose a new multiple treebank exploitation method for dependency parsing with quasi-synchronous grammar (QG). There exist multiple treebanks of different annotation styles for Chinese, and it is attractive to exploit multiple treebanks to improve the parsing accuracy. We present a simple and effective framework based on QG for exploiting multiple monolingual treebanks with different annotation guidelines for parsing. Several types of transformation patterns (TP) are designed to capture the systematic annotation inconsistencies among different treebanks. Based on such TPs, we design QG

features to augment the baseline parsing models. The QG features can guide the parsing model to make better decisions, and they naturally fit into the decoding algorithms of the baseline graph-based parsing models. Experimental results show that our method can effectively exploit the knowledge of the source treebank, and significantly improve the parsing accuracy on the target treebank.

In conclusion, based on the characteristics of Chinese, this thesis conducts thorough study on fast high-order dependency parsing using punctuation, joint POS tagging and dependency parsing, and multiple treebank exploitation, and substantially improve the efficiency and accuracy of dependency parsing on real-world texts. We have accomplished several primitive achievements, which we hope can further motivate the progress of natural language processing and other high-level applications like machine translation and information retrieval.

Keywords: Dependency Parsing; Beam Search; Joint Models; Seperately Passive-aggressive Training Algorithm; Multiple Treebank Exploitation

目 录

Contents

第 1 章 绪论

1.1 课题背景及意义

1.1.1 课题背景

自然语言处理的分析技术，可以大致分为三个层面。第一层是词法分析，包括分词（Word Segmentation）、词性标注（Part-of-speech Tagging）、命名实体识别（Named Entity Recognition）、词义消歧（Word Sense Disambiguation）。和大部分西方语言不同，汉语书面语词语之间没有明显的空格标记，文本中的句子以字串的形式出现。因此汉语自然语言处理的首要工作就是要将输入的字串切分为单独的词语，然后在此基础上进行其他更高级的分析。词性标注的目的是为每一个词性赋予一个类别，这个类别称为词性标记（Part-of-speech tag），比如，名词（Noun）、动词（Verb）、形容词（Adjective）等。一般来说，属于相同词性的词，在句法中承担类似的角色。命名实体识别的任务是识别句子中的人名、地名、机构名等命名实体。每一个命名实体由一个或多个词语构成。根据标注规范的不同，命名实体可能存在嵌套的情况，但是目前大部分研究工作不考虑嵌套的情况。比如，“王义和老师”是人名，“哈尔滨南岗区”是地名，“哈尔滨工业大学”是机构名或地名。词义消歧根据句子上下文语境，判断出每一个或某些词语的真实意思。比如“打”有很多意思。“打篮球”和“打酱油”的意思完全不同。甚至不同的语境下，“打酱油”中的“打”也会有不同的意思。词义消歧一般需要一个词义体系，比如针对汉语的同义词词林扩展版和HowNet，针对英文的WordNet。不同的词义体系对词义区分的粒度不同。粒度越细，词义消歧的结果可以提供更细的信息，但是词义区分的难度也越大，标注的代价也越大。分词、词性、和命名实体识别方面的研究已经日趋成熟，并且广泛应用于机器翻译（Machine Translation，MT）、问答（Question Answering，QA）、文本挖掘（Text Mining，TM）、和信息检索（Information Retrieval, IR）等上层技术。目前的研究主要关注领域移植和面向开放域的处理。词义消歧受限于标注语料规模比较小，并且词义消歧对于上层语言处理和应用的帮助不明朗，因此词义消歧的研究似乎处于低潮。

自然语言处理的第二层为句法分析。句法分析将输入句子从序列形式变为树状结构，从而可以捕捉到句子内部词语之间的远距离搭配或修饰关系，因

此是自然语言处理最关键的一环。目前研究界存在两种主流的句法标注体系：短语结构句法体系（phrase-structure grammar, Context-Free Grammar, CFG）[1]、依存结构句法体系（dependency grammar）[2] 。另外，近年来人们也越来越关注对组合范畴文法（Combinatory Categorial Grammar, CCG）和头驱动的短语结构语法（Head-driven Phrase Structure Grammar, HPSG）[3]等的研究。与短语结构语法相比，依存语法有以下优势[4]：

(1) 依存语法表示形式简洁，易于理解和标注。依存语法直接表示词语之间的关系，而短语结构语法则需要非终结符（Non-terminal symbol），如S、NP、VP等，来刻画句子内部组块的句法结构和功能。当然，这些非终结符可以包含更丰富的信息。

(2) 依存语法可以很容易的表示词语之间的语义关系，比如句子成分之间可以构成施事、受事、时间等关系。这种语义关系可以很方便的应用于语义分析、信息抽取等。

(3) 简洁的表示形式的另一个好处是，可以实现更高效的解码算法。目前，基于转移的依存分析方法使用贪心搜索或者柱搜索（beam-search），可以在线性时间内找到最优的依存树；而基于图的依存分析方法利用动态规划解码算法（Dynamic Programming, DP），可以在$O(n^3)$或$O(n^4)$找到最优的依存树。从实践的角度看，目前流行的依存句法分析系统比短语结构句法分析系统的效率更高[5]。

(4) 依存语法可以方便的表示一些语序比较灵活的语言，如荷兰语和捷克语。这些语言对应的依存句法结构中存在大量的交叉弧现象，即非投影结构（non-projective structure）。而短语结构句法分析就很难表示这种句法结构。并且，研究者们也提出了多种面向非投影结构的句法分析算法。

自然语言处理的第三个层面是语义分析。语义分析的最终目的是理解句子表达的真实语义。但是，语义应该采用什么表示形式一直困扰着研究者们，至今这个问题也没有一个统一的答案。语义角色标注（Semantic Role Labeling，SRL）是目前比较成熟的浅层语义分析技术[6, 7]。给定句子中一个谓词，语义角色标注的任务是从句子中标注出这个谓词的施事、受事、时间、地点等参数（argument）。谓词一般是动词，人们也开始研究名词、形容词作为谓词的情况[8]。语义角色标注一般都在句法分析的基础上完成，句法结构对于语义角色标注的性能至关重要。

综合以上介绍，可以看出句法分析在自然语言处理中处于一个非常重要的位置，输入文本在这一步被转化为结构化句法结构。句法结构进而帮助语义分

析以及其他上层应用如机器翻译、问答、文本挖掘、信息检索等。

1.1.2 课题意义

在不同语法体系中，依存语法以其形式简洁、易于标注、便于应用等优点，逐渐受到研究人员的重视。目前，依存语法标注体系已经被自然语言处理领域的许多专家和学者所采用，应用于不同语言中，并对其不断地发展和完善。研究者们提出并实现了多种不同的依存分析方法，达到了较好的准确率。在计算自然语言学习国际会议（Computational Natural Language Learning，CoNLL）联合举办的公开评测任务中，2006、2007连续两年举行了多语依存句法分析评测，对包括汉语在内的十几种语言进行依存分析[9, 10]；2008、2009年则对依存分析和语义角色标注联合任务进行评测[8, 11]。国内外多家大学、研究机构和商业公司都参加了这些评测任务。这些评测一方面提供了大规模、多语言的标准评测数据集，另一方面提供了研究者们就依存句法分析进行集中交流、讨论的平台。近几年，在这几届评测任务的推动下，依存句法分析在多语言处理、非投影结构处理、和语义分析的联合分析等方面快速发展。然而，面向汉语的依存句法分析相对英文而言，还比较滞后，准确率相差比较大。英文上依存分析准确率约93%（自动词性），而汉语上只能达到约80%（正确分词、自动词性）。因此句法分析已经成为汉语信息处理的瓶颈，面向汉语的依存句法分析需要更多的关注。本论文旨在通过分析汉语的特点，提高汉语依存句法分析的准确率和效率。

作为自然语言处理的核心技术，依存句法分析广泛应用于各种上层信息处理任务。

(1) 应用于语义角色标注。语义角色标注是一种浅层的语义分析，与句法结构紧密联系。句法结构可以为语义角色标注提供非常有效地信息。2008年以前，语义角色标注主要利用短语结构句法分析的结果。在CoNLL 2008和CoNLL 2009两届依存句法分析和语义角色标注联合评测任务中，都在依存句法结构上标注语义角色。

(2) 应用于机器翻译。随着基于句法树的机器翻译技术的发展，如“树到串”、“串到树”、“树到树”等机器翻译方法，句法分析在机器翻译上的应用也越来越深入。早期工作主要是基于短语结构句法分析树。近年来，越来越多的研究者尝试使用依存句法分析的结构，尤其是用于翻译结果调序。

(3) 应用于问答系统。句法分析的结果可以从多个方面帮助问答系统的性

能。对于输入问题（query），传统的处理方法是平等的对待所有问题中包含的词。但是，实际上某些关键词的重要度是不同的。句法结构可以帮助分析问题中哪些关键词更重要。另外，传统的答案抽取方法采用关键词匹配或者基于词袋模型的相似度计算。但是，如果可以通过深层匹配候选答案和查询之间的句法结构，那么很有可能提高答案抽取的质量。

(4) 应用于文本挖掘。文本挖掘中的事件抽取、关系抽取等核心问题都需要句法分析提供比较的知识。

(5) 应用于信息检索。和问答系统类似，句法分析可以在查询分析和相关网页排序上为信息检索系统提供深层次的帮助。

依存句法分析不仅可以帮助各种上层应用，并且可以帮助底层的自然语言处理。传统的自然语言处理系统都采用级联式的框架，即先分词，然后做词性标注，然后进行句法分析。以词性标注为例，很多词性标注的歧义，如名词动词歧义，仅仅利用序列标注模型中使用到的局部上下文信息是无法正确消解的，而长距离的句法结构可以提供有用的信息。又比如错误的分词结果会导致句法模型得到很差的句法结构，然而分词任务中最难解决的一些歧义可以考虑借助句法结构进行消解。基于以上分析，今年来一些研究者开始尝试将句法分析和词性标注联合起来处理[12–14]，甚至建立句法分析、词性标注、和分词三个任务的联合模型[15–17]。

1.2 研究现状及分析

1.2.1 依存句法分析的形式化定义

依存语法历史悠久，最早可能追溯到公元前几世纪Pānini提出的梵文语法，之后发展成为传统语法学家使用的句法表示形式，尤其是欧洲古典语言和斯拉夫语。依存语法的现代理论通常被认为起源于法国语言学家Lucien Tesniére的工作。十九世纪五十年代末，在Lucien Tesniére去世后，他的工作才被发表。从那时起，人们提出了很多不同的依存语法框架，其中最著名的包括：Prague School的Functional Generative Description，Mel’čuk的Meaning-Text Theory和Hudson的Word Grammar[18]。

各种依存语法存在一个共同的基本假设：句法结构本质上包含词和词对之间的关系。这种关系称为依存关系（dependency relations，或dependencies）。一个依存关系连接两个词，分别是核心词（head）和修饰词（dependent）。依存

关系可以细分为不同的类型，表示两个词之间的句法关系（dependency relation types/labels）。

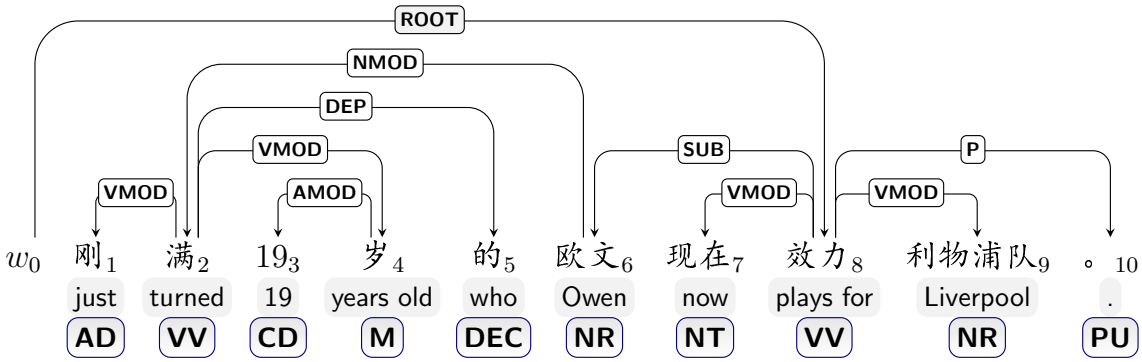


图 1-1 依存句法树实例。

Fig. 1-1 An example dependency tree.

依存句法分析的任务是针对给定句子分析其依存句法结构。输入句子可以被表示为：$\mathbf{x} = w_0 w_1 ... w_n$。$w_i$表示输入句子的第$i$个词（也可以认为包含了词和词性）。由于依存句法分析时需要利用每个词的词性信息构成丰富的特征，因此输入句子对应的词性序列通常也作为依存句法分析的输入，记为$\mathbf{t} = t_0 t_1 ... t_n$。形式化地，一个依存句法树表示为$\mathbf{d} = \{(h, m, l) : 0 \leq h \leq n, 1 \leq m \leq n, l \in \mathcal{L}\}$，其中$(h, m, l)$表示一个从核心词（head, father）$w_h$到修饰词（modifier，dependent, child）w_m的依存弧，l表示依存弧的关系类型，$\mathcal{L}$是依存弧关系类型的集合。依存弧的关系类型用来表示两个词之间的句法或语义关系。图1-1给出了一棵依存树的例子，最后一列给出的是每个词的人工标注的词性，其中句法结构和词性都遵循CTB5中汉语数据集的标注规范。w_0是一个伪词（pseudo-word），在依存结构中指向整个句子的核心词。w_0的引入可以简化依存句法分析的形式化表示。依存弧$(8, 6, \text{SUB})$表示“欧文”修饰“效力”，并且依存关系类型为主语（SUB）。这条依存弧也可以表示为$6 \xleftarrow{\text{SUB}} 8$。下面我们给出一些详细的定义。

依存图（Dependency Graph）：输入句子对应的一个依存图$G = (V, A)$是一个有向多重图（multigraph），其中节点和边的定义为：

(1) $V = \{0, 1, ..., n\}$。其中每一个节点i对应一个输入词w_i。依存图必须包含所有词。

(2) $A \subseteq V \times V \times \mathcal{L}$。

依存树（Dependency Tree, Well-formed Dependency Graph）：依存树$T = (V, A)$是一个依存图，同时需要满足以下几个条件：

(1) 单核心（Single-headed）：每个词只能修饰一个核心词。这个条件包括三方面含义：

- $\forall i \in V \setminus \{0\}, l \in \mathcal{L} : (i, 0, l) \notin A$。意思是：$w_0$没有核心词。
- 如果$(i, j, l) \in A$，那么$\forall l' \in \mathcal{L} \setminus \{l\} : (i, j, l') \notin A$。意思是：每个词只能以一种依存关系修饰其核心词。
- 如果$(i, j, l) \in A$，那么$\forall r \in V \setminus \{i\}, l' \in \mathcal{L} : (k, j, l') \notin A$。意思是：每个词只能修饰唯一的核心词。

(2) 弱连通（Weakly-connected）：用$i \rightarrow j$表示一条不考虑依存关系的依存弧$(i, j, *)$。$i \rightarrow^* j$表示从节点i出发，沿着依存弧，可以到达节点j。连通性的含义是在依存树中，可以从节点0任何其他节点。形式化描述为：$\forall i \in V \setminus \{0\}, 0 \rightarrow^* i$。

(3) 无环（Acyclic）：$\forall i \in V, j \in V$，如果$i \rightarrow^* j$，那么$j \nrightarrow^* i$。

条件(1)可以得知，一个依存树包含n个有向边，即$|A| = n$。不难发现，条件(1+2)等价于条件(1+3)。即如果依存图满足单核心，并且弱连通，那么肯定无环。反之如果依存图满足单核心，并且无环，那么肯定弱连通。

投影依存树（Projective Dependency Tree）：一个依存弧$(i, j, l) \in A$，如果对于所有满足$\min(i, j) < k < \max(i, j)$的$k$，都有$i \rightarrow^* k$，那么称之为投影依存弧。如果一个依存树$T = (V, A)$中所有的依存弧都是投影依存弧，那么这个依存树称为投影依存树。

非投影依存树（Non-projective Dependency Tree）：如果一个依存树$T = (V, A)$包含非投影依存弧，那么称为非投影依存树。直观来讲，投影依存树可以在平面上画出来，并且不存在交叉弧。而非投影依存树包含交叉弧。

依存句法分析（Dependency Parsing）：给定输入句子$\mathbf{x}$，依存句法分析的目标是给出分值最大的依存树$\hat{\mathbf{d}}$，如公式(1-1)所示。

$$\hat{\mathbf{d}} = \arg\max_{\mathbf{d}} \mathrm{Score}(\mathbf{x}, \mathbf{d}) \tag{1-1}$$

因此，依存分析的两个基本问题：(1) 如何定义$\mathrm{Score}(\mathbf{x}, \mathbf{d})$，即计算句子和对应的依存树的分值。这是建模问题。(2) 给定模型参数，即特征权重，如何建立满足依存树约束的$\mathbf{d}$。这是解码问题。另外，根据$\mathbf{d}$需要满足的约束，可以将依存分析分为两种：面向投影结构的依存分析和面向非投影结构的依存分析。

1.2.2 依存句法分析的评价方法

从依存句法分析的形式化定义中的“单核心”性质可知，在一个依存结构中，除了伪节点w_0，句子中每个词都有唯一的核心词。目前最常用的两个评价指标都以词为单位。

- 只考虑依存骨架，不考虑依存关系类型：Unlabeled Attachment Score (UAS)。

$$UAS = \frac{\text{核心节点正确的词数}}{\text{总词数}} \times 100\% \tag{1-2}$$

- 同时考虑依存骨架和关系类型：Labeled Attachment Score (LAS)。

$$LAS = \frac{\text{核心节点正确、并且对应依存关系类型也正确的词数}}{\text{总词数}} \times 100\% \tag{1-3}$$

其他两个常用评价指标如下：

- 评价整个句子的依存结构准确率：Complete Match (CM)。

$$CM = \frac{\text{整个依存结构完全正确的句子数}}{\text{总句子数}} \times 100\% \tag{1-4}$$

- 考察句子根节点准确率：Root Accuracy (RA)。

$$RA = \frac{\text{根节点正确的句子数}}{\text{总句子数}} \times 100\% \tag{1-5}$$

需要注意的是，依存句法分析评价时一般不考虑标点符号，即不关心标点符号的核心节点是否正确。一般来说，标点符号依赖于句子或从句的核心词，因此确定其核心节点比较困难，其对应核心节点的准确率和其他词相比较低。如果评价指标考虑标点符号，准确率指标会有小幅度的下降。

1.2.3 基于图的依存句法分析方法

依存句法分析两个主流的方法分别是：一种是基于图的方法，将依存句法分析看成从完全有向图中寻找最大生成树的问题。另一种主流的依存句法分析方法是基于转移的方法，通过一个移进规约转移动作序列构建一棵依存句法树，将依存分析问题建模为寻找最优动作序列的问题。基于图的方法和基于转移的方法。这两种方法从两个不同的角度解决这个问题。实验表明这两种方法各有所长，并且可以互补。

1. 模型定义

McDonald等(2005)首先提出基于图的依存句法分析方法，将问题建模为从

一个有向多重图（完全依存图）中找到概率（分值）最大的依存树的问题[19]。为了保证基于动态规划的全局搜索算法的效率，基于图的模型需要做很强的独立假设。假设句子对应的依存树中，只有存在于某些特殊结构（子树，subtree）中的依存弧之间才互相联系和影响，其他依存弧之间则互相独立。在这种假设下，一个依存树的分值便分解为一些子树的分值的和。

$$\begin{aligned}\mathrm{Score}(\mathbf{x},\mathbf{d}) &= \mathbf{w}\cdot\mathbf{f}(\mathbf{x},\mathbf{d}) \\ &= \sum_{p\subseteq\mathbf{d}} \mathrm{Score}_{\mathrm{subtree}}(\mathbf{x},p)\end{aligned} \tag{1-6}$$

其中，p表示一个独立假设允许的子树。p包含一个或多个$\mathbf{d}$中的依存弧。$\mathrm{Score}_{\mathrm{subtree}}(\mathbf{x},p)$表示由$p$贡献的分值。$\mathbf{f}(\mathbf{x},\mathbf{d})$是$(\mathbf{x},\mathbf{d})$对应的聚合的句法特征向量，$\mathbf{w}$是句法特征的权重向量。由于本论文主要采用基于图的依存句法分析方法，然后对其进行改进和提高，因此，我们此处简要介绍前人提出的基于图的依存句法分析模型。这些模型在后续章节会有详细的介绍。另外，基于图的依存句法分析模型命名时，一般会使用一阶/二阶/三阶模型的区分方法。虽然研究界并没有给出严格的定义，但是根据文献中的命名方法，本文在此澄清一下：基于图的模型的阶数是指模型融入的最复杂的子树中包含的依存弧的数目。例如，二阶模型指模型融入的最复杂的子树中包含两条依存弧。

McDonald的一阶模型（first-order model） McDonald等(2005)提出一阶模型[19]。其中，一个依存树$\mathbf{d}$的分值由所有依存弧(i,j,l)的分值累加得到，即依存弧之间互相独立，因此可以看成将一棵依存树按照弧分解（arc-factorization）。

McDonald的二阶模型（second-order model） 一阶模型假设依存树中依存弧之间相互独立，模型过于简化。McDonald和Pereira (2006)提出了二阶模型，弱化了这种独立假设，允许同方向的相邻兄弟弧之间不再独立[20]，即一棵依存树的分值增加每对相邻兄弟弧（adjacent sibling）的分值。相邻兄弟弧指满足如下条件的两条依存弧(h,m)和(h,s)：(1) m和s在h的同一侧；(2) m和s之间没有其他词修饰h。

Carreras的二阶模型（second-order model） Carreras (2007)提出更复杂的二阶模型，进一步弱化了独立假设[21]。Carreras的模型中，句法树祖孙弧之间也不再独立，即一棵依存树的分值继续增加祖孙弧（head-modifier-grandchild）对应的分值。祖孙弧指两条依存弧h,m和(m,g)。可以看到，h和g构成祖孙关系，所以简称为祖孙弧。需要注意的是，Carreras的模型中，只考虑g是m的最左或最右儿子时的情况，也可以理解为g是h的最远孙子（outermost grandchild）。

Koo的二阶模型（second-order model） Koo等(2010)提出另一种二阶模型，

可以融入所有祖孙弧对应的分值[22]。和Carreras的模型不同的是，Koo的模型没有最远孙子节点的条件。我们的实验结果表明，采用同样的解码算法，Koo的二阶模型比Carreras的模型的准确率高一些。

Koo的三阶模型（third-order model） Koo等(2010)进一步提出三阶模型，考虑包含三条依存弧的子树对应的分值[22]。他们提出了两种不同的三阶模型，分别命名为Model 1和Model 2。Model 1可以融入祖孙兄弟子树（grandparent-head-sibling）特征，而Model 2可以进一步融入三兄弟子树（tri-sibling）特征。在英语和捷克语上的实验结果表明，Model 1和Model 2的依存分析准确率基本相同。

2. 解码算法 已知模型的特征权重向量**w**，搜索输入句子对应权值最大的依存树的过程称为解码（decoding）。在此我们关注投影依存结构的解码算法，对于非投影依存结构的解码算法我们在后面给出。一阶模型对应的解码算法为Eisner算法[23, 24]。Eisner算法的本质是动态规划，不断合并相邻子串的分析结果，直到得到整个句子的结果。Eisner算法可以保证分析结果是一个投影依存树，其时间复杂度为$O(n^3)$。对于二阶模型，McDonald对Eisner算法进行了改进，增加了表示相邻兄弟节点的数据类型，保证了算法仍然满足动态规划算法的性质[20, 25]。其算法的复杂度仍然是$O(n^3)$。Carreras在Eisner算法的基础上，介绍了如何对解码算法进行扩展，以满足高阶模型，其解码算法复杂度为$O(n^4|\mathcal{L}|)$[21]。Koo和Collins (2010)的二阶和三阶模型的解码算法时间复杂度相同，为$O(n^4)$[22]。另外，需要注意的是，Carreras的模型在依存句法分析时确定依存关系类型，又称为带依存关系类型的依存句法分析（labeled dependency parsing），因此时间复杂度跟依存关系类型的数目有关。而其他模型一般都不考虑依存关系类型（unlabeled dependency parsing）。

3. 训练算法

训练算法通过训练数据学习特征权重向量**w**。基于图的依存分析方法通常采用的训练算法是在线算法，包括平均感知器算法（averaged perceptron, AP）[26]，passive-aggressive算法（PA）[27]及averaged margin infused relaxed算法（MIRA）[28]。简洁起见，我们将在后续章节中介绍这些训练算法。

4. 其他基于图的方法

置信传播算法（Belief Propagation） Smith和Eisner (2008) 提出一种基于BP算法的依存分析方法，融入高阶特征。解码过程中，通过消息传递

（message passing），高阶特征不断调整每条弧的权重[29]。

ILP解码 Riedel和Clarke (2006)提出一种基于整数线性规划（Integer Linear Programming）的依存分析方法[30]。他们使用了一阶图模型，同时加入了一些语言学启发的强约束。他们的模型需要指数数量的约束，因为每一个可能的环都需要引入一个约束。由于无法在解码时同时考虑所有的约束，他们采取了增量式的解码算法：当产生的解中包含某一个环时，则加入相应的约束，继续搜索。

Martins等(2009) 提出一种简约的基于ILP解码的依存分析方法[31]。他们观察到依存树的条件“单核心+连通”等价于“单核心+无环”。而ILP刻画“连通”只需要多项式数目约束和变量。他们的模型中可以融入更多特征：兄弟特征、祖孙特征、配价（儿子数目）等。

1.2.4 基于转移的依存句法分析方法

基于转移的方法将依存树的搜索过程建模为一个动作序列，依存分析问题转化为寻找最优动作序列的问题，如公式(1-7)所示[32–35]。

$$\begin{aligned}\hat{\mathbf{d}} &= \arg\max_{\mathbf{d}} \mathrm{Score}(\mathbf{x}, \mathbf{d}) \\ &\simeq \arg\max_{a_1..a_m} \prod_{i=1}^{m} \mathrm{Score}_{\mathrm{action}}(\mathbf{x}, h_i, a_i)\end{aligned} \tag{1-7}$$

其中，h_i表示前$i-1$个动作构成的历史，a_i表示根据当前历史采取的动作。$\mathrm{Score}_{\mathrm{action}}(\mathbf{x}, h_i, a_i)$表示根据$i-1$个动作构成的历史，采用动作$a_i$的分值或概率。$\mathrm{Score}_{\mathrm{action}}(\mathbf{x}, h_i, a_i)$通常由一个分类器给出，如支持向量机（support vector machines，SVM），最大熵（Maximum Entropy, ME）分类器等。Zhang和Clark (2008)首次提出使用线性模型的全局训练方法，并且利用提前更新（early update），可以提高句法分析准确率[36]。

1. 转移系统

基于转移的方法中，最重要的是构造一个转移系统，并且这个系统是正确的（correct，sound），完整的（complete）。正确性指当系统转移到一个接受状态时，得到的依存树是正确的、良构的。完整性指对于任意的一个正确的依存树，都可以对应一个转移（动作、状态）序列。一个转移系统应该包含如下几方面：

(1) 如何表示一个状态（state, configuration）。

(2) 动作集合及每个动作引起的状态转移。

(3) 初始状态。

(4) 接受状态。（到达这种状态则退出。）

下面介绍两种典型的基于栈的方法[37]。

Arc-standard方法 这种方法中，一个状态表示为三元组$< S, Q, A >$。其中S表示一个栈，Q表示一个队列，A表示目前已经产生的弧的集合。

- 初始状态：$< [w_0], [w_1 w_2 ... w_n], \emptyset >$（初始状态整个句子都存储在队列中。）
- 接受状态：$< [w_0], [\,], A >$（栈中只有伪节点w_0，队列为空。）
- LEFT-ARC$_l$：$< S|w_i, w_j|Q, A > \Rightarrow < S, w_j|Q, A \cup (j, i, l) >, i \neq 0$
- RIGHT-ARC$_l$：$< S|w_i, w_j|Q, A > \Rightarrow < S, w_i|Q, A \cup (i, j, l) >$
- SHIFT: $< S|w_i, w_j|Q, A > \Rightarrow < S|w_i w_j, Q, A >$

这种方法有如下几个特点：

(1) 一旦一个节点找到自己的父亲，则立刻消失（栈和队列中都不再存储）。因此采取动作LEFT-ARC$_r$时，应该保证栈顶元素w_i已找到所有的儿子；采取动作RIGHT-ARC$_r$时，应该保证列表头元素w_j已找到所有儿子。

(2) 栈中元素之间不存在弧。

(3) 一个节点只能通过SHIFT入栈。一个节点入栈有三种可能：这个节点存在没有找到的右儿子；这个节点的父亲在右边；错误的入栈选择。

(4) 一个节点可能多次通过SHIFT入栈，然后通过RIGHT-ARC$_r$，找到一个右儿子，并回到队列头。当一个节点处于队列头时，可以通过LEFT-ARC$_r$得到一个左儿子。从这个角度看，对于arc-standard方法，一个依存树对应的动作序列不是唯一的。因为当一个节点有多个左儿子，多个右儿子时，可以选择先找到所有左儿子，然后找到所有右儿子；或者反之先找到所有右儿子，然后找到所有左儿子；或者交替进行。在模型训练阶段，需要对每一个依存树确定一个唯一的动作序列。通常的做法是采取“最短栈”原则，即先从内向外找到一个节点的所有左儿子，然后再找右儿子[38]。

(5) 这种方法最难解决的歧义是：当w_i和w_j存在右弧时，会引起RIGHT-ARC$_r$和SHIFT冲突。此时取决于w_j是否已经找到所有的右儿子。而这是一个很难判断的条件。

Arc-eager方法 Arc-eager方法与arc-standard很相似，状态也表示为三元组$< S, Q, A >$，初始状态和接受状态也相同。区别在于动作集合不同。

- 初始状态：$< [w_0], [w_1 w_2 ... w_n], \emptyset >$
- 接受状态：$< [w_0], [\,], A >$

- LEFT-ARC$_l$: $< S|w_i, w_j|Q, A > \Rightarrow < S, w_j|Q, A \cup (j, i, l) >, i \neq 0$
- RIGHT-ARC$_l$: $< S|w_i, w_j|Q, A > \Rightarrow < S|w_i w_j, Q, A \cup (i, j, l) >$
- REDUCE: $< S|w_i, w_j|Q, A > \Rightarrow < S, w_j|Q, A >$
- SHIFT: $< S|w_i, w_j|Q, A > \Rightarrow < S|w_i w_j, Q, A >$

与arc-standard相比，arc-eager存在以下特点：

(1) LEFT-ARC$_l$导致儿子节点消失，而RIGHT-ARC$_l$导致儿子节点入栈。

(2) 栈中元素之间可能存在右弧。被右弧连接的两个节点在栈中应该相邻。

(3) 一个节点有两种方式入栈：SHIFT（其父亲在右边，通过LEFT-ARC$_l$消失）；RIGHT-ARC$_l$（其父亲在左边，通过REDUCE消失）。一个节点入栈前需要找到所有左儿子。入栈后找到所有右儿子。

(4) 一个节点入栈后无法再回到队列中。

(5) 如果栈顶节点和队列头节点存在左弧，那么必须执行LEFT-ARC$_l$，而不能执行SHIFT，否则由于入栈元素无法回到队列中，以后没有机会建立这条弧。同样，如果存在右弧，那么也必须执行RIGHT-ARC$_l$。因此这种方法被命名为*arc-eager*。

(6) 这种方法最难解决的歧义是：当栈顶节点和队列头节点不存在弧，并且栈顶节点已经找到父亲（RIGHT-ARC$_l$入栈）时，会引起SHIFT和REDUCE冲突。如果队列头节点已经找到所有左儿子，并且其父亲在右边，则可以SHIFT；如果栈顶节点已经找到所有右儿子，那么可以REDUCE。这两个条件都很难判断。从这个角度看，对于arc-eager方法，一个依存树对应的动作序列也不是唯一的。因为存在SHIFT和REDUCE都可行的情况。

2. 解码算法

解码算法的任务是找到一个概率或分值最大的动作序列。存在三种解码算法：贪心搜索、柱搜索（beam search）和加入动态规划的柱搜索。贪心算法每一次根据当前状态，选择并执行分值最大的动作，进入到下一个状态。如此反复直至到达接受状态[39]。贪心算法的最大问题是错误级联问题：即前期动作选择错误会导致后面更多的错误，无法回溯。柱搜索使用agenda来组织搜索过程。agenda中保存了经过相同数目的动作后到达的K个状态。遍历agenda中的所有状态。对每一个状态，都选择执行分值最大的K个动作，产生K个新的状态。从所有新的状态选择分值最大的K个，更新agenda[40, 41]。和贪心算法相比，柱搜索可以增加搜索空间，从一定程度上解决错误级联的问题。Huang和Sagae

(2010)提出在柱搜索的过程中加入动态规划[34]。其基本思想是，根据后续动作需要的原子特征（atomic/kernel feautres），将等价的状态（所有原子特征相同）合并。加入动态规划策略后，解码算法的搜索空间进一步扩大，从而提高句法分析准确率。

1.2.5 依存句法分析融合方法

基于图和基于转移的方法近年来成为最主流的两种方法，并且都达到了最好的性能。可以从如下几方面对这两种方法进行对比。

- 解码算法　基于转移的方法基于当前历史和分类器结果，贪心的选择下一个动作。搜索过程无法回溯，因此错误级联比较严重。柱搜索策略从一定程度上缓解了这种贪婪性。而基于图的方法通过动态规划，穷举的从图中找到分值最大的依存树。从解码效率来看，基于转移的方法一般是$O(n)$，而基于图的方法一般是$O(n^3)$，因此基于转移的方法在解码速度上更有优势。
- 训练算法　基于转移的方法训练的目标是得到一个更准确的预测下一个动作的分类器。而基于图的方法则训练出一个能够让正确的依存树分值更高的模型。当然，随着Zhang和Clark (2008)首次提出针对基于转移的模型使用线性模型的全局训练方法，基于图和基于转移的方法在训练算法上趋于一致。
- 特征表示　基于转移的方法使用的特征更丰富，可以使用当前已构建出来的所有子树信息。而基于图的方法由于要考虑解码算法的效率，因此只能使用局部的特征，如单弧相关的特征，兄弟弧相关的特征，祖孙弧相关的特征等。

在CoNLL2006，2007多语依存分析评测任务上，基于图和基于转移的方法都取得了最好的性能。两者的性能基本相同。但是McDonald和Nivre (2007)通过详细比较发现，这两种方法的错误分布不同。与基于图的方法相比，基于转移的方法在短距离（核心词和依存词之间的距离）依存弧上准确率高，在距离根节点较远的依存弧（从这条弧反向到达根节点）上准确率高。反之，基于图的方法准确率较高。由于基于转移的方法存在错误级联，因此越晚执行的动作错误率越高[42]。

由于基于转移和基于图的方法错误分布不同，可以互相弥补。因此出现了一些融合这两种模型的方法。主要包括三方面的工作。

后处理融合 Sagae和Lavie (2006)提出一种对多个模型的结果进行重组的方法[43]。其主要思路是：根据多个模型的分析结果，建立一个新的依存图。依存图中每条弧的权重由多个模型结果投票加权得到。然后使用基于图模型中的解码算法获得新的结果。作者尝试了不同的加权策略：如对所有模型的分析结果一视同仁，一条弧的权重为其在各个模型分析结果中出现的次数；或者将各个模型的准确率作为权重；或者更细化一些，根据每个模型在不同的词性上的性能，对相应的弧加权。实验中作者采用了多个模型，基于图的模型采用McDonald的方法，基于转移的模型包括：Nivre的方法按照从左到右处理输入句子的模型，Nivre的方法按照从右到左处理输入句子的模型，和Yamada的方法。实验结果表明，这种方法可以有效地提高分析性能。作者还将这种思想应用到短语结构句法分析，也取得了性能提高。

单向指导 Nivre和McDonald (2008)提出一种单向指导的方法，使得基于转移的方法和基于图的方法可以互相弥补[44]。其核心思想是：使用一种模型的分析结果指导另一种模型，指导的形式是加入特征。以基于转移的方法指导基于图的方法为例，在基于图的模型中，弧相关的特征向量中加入一类特征：“核心词词性+依存词词性+基于转移模型结果中这条弧是否存在”。基于转移的模型通过交叉验证的方式产生训练语料的自动分析结果。基于图的模型训练过程中，通过基于转移的模型的自动分析结果特征，学习哪些情况下应该接受基于转移模型的结果，哪些情况下应该拒绝。

联合训练，解码过程融合 Zhang和Clark (2008)提出一种细粒度的融合方法[36]。这种方法的思想是融合基于转移和基于图的模型中采用的特征，并且一起训练权重。作者使用averaged perceptron算法统一训练两种特征的权重，这样做的好处是解码阶段就可以综合考虑两者的分值。解码阶段，作者采用基于转移的方法，并且加入柱搜索。不同的是，对agenda中的状态打分时，不只考虑基于转移方法使用的特征，还要根据基于图方法使用的特征。

1.2.6 利用未标注数据的半指导方法

由于标注数据的规模有限，领域覆盖面有限，而标注树库又费时费力。同时，未标注数据很丰富，甚至可以认为是无限的。因此越来越多的研究者尝试利用未标注数据来提高句法分析的性能。事实证明，未标注数据确实对句法分析有帮助，但是如何利用未标注数据至关重要。下面介绍近几年的一些工作：

词类特征（word class） Koo等(2008)提出从未标注数据提取词类信息，

作为新的特征，以提高依存分析的性能[45]。他们采用Brown算法将未标注语料中的所有词层次聚类，聚类结果中每个词都有一个分层的编号。实验中作者使用了第4层和第6层编号作为新的特征。实验表明，词类信息可以提高分析性能。

利用自动分析结果中的子树 Chen等(2009)对大规模未标注数据自动依存分析，从结果中提取各种子树的频率信息[46]。根据频率将子树分为四类：高频（HF），中频（MF），低频（LF），未出现（ZERO）。作者实验中采用了二阶图模型依存分析方法。而抽取的子树包括两种类型：单个依存弧，兄弟弧，恰好对应依存分析模型中的弧特征，兄弟弧特征。解码过程中，模型可以利用这些子树频率信息，如：当确定$w_i \rightarrow w_j$的权重时，可以参考子树$w_i \rightarrow w_j$的频率，子树$w_{i-1} \rightarrow w_j$的频率，子树$w_{i+1} \rightarrow w_j$的频率等。

利用双语对齐信息 Huang等(2009)提出利用双语对齐信息提高依存分析性能的方法[38]。各种语言的歧义是不同的。例如：英语中存在介词短语附着歧义。汉语中的介词短语作定语没有歧义，但是介词短语边界有歧义。因此，利用词对齐信息可以消解一些歧义现象。作者采用了arc-standard依存分析方法，实验发现利用词对齐信息可以提高依存分析的性能。

自学习（self-training） McClosky等(2006,2008)在短语结构句法分析中使用自学习方法，提高了句法分析的性能[47–49]。他们使用了Eugene Charniak的重排序句法分析方法，包括一个生成式句法分析模型和一个判别式重排序模型[50]。实验发现，重排序模型对性能的提高至关重要。

1.2.7 面向非投影依存结构的方法

在一些西方语言中，非投影依存结构很常见。例如，捷克语的布拉格依存树库中，23%的句子都对应非投影依存结构。而德语和丹麦语中比例更大。非投影依存结构对解码算法的约束更少，因此理应比投影结构简单。但是事实并非如此。原因是，实际自然语言中，非投影结构一般都是近似投影结构（nearly projective），即只存在少量的非投影依存弧。下面介绍几种面向非投影依存结构的方法：

Chu-Liu-Edmond算法 Chu-Liu-Edmond算法是一种在有向图中找到一个最大生成树的算法[51]。McDonald等(2005)使用这种算法作为一阶模型的解码算法[52]。算法的思想是：开始阶段，每个节点（除了伪节点w_0）从指向自己的依存弧中，选择分值最大的一个依存弧。这样如果构成了一个环，那么打开环中

的一个依存弧，这样这条依存弧对应的儿子节点就没有父亲。这时需要从非环上节点中，重新为这个节点找到一个父亲节点。每一次这种操作，都会导致整个分值下降。那么选择分值下降最少的依存弧打开。如此反复，直至没有环。最终得到一个合格的非投影依存树。

二阶图模型的近似解码算法 二阶图模型的解码算法是一个NP完全问题。McDonald和Pereira (2006)观察到非投影结构一般都是近似投影的，从而设计了一种近似算法[20]。基本思想是，首先利用投影结构对应的解码算法得到一个投影结构依存树。然后重复修正依存结构，每次只改变一个依存弧。直到修正依存结构不增加依存树的分值。

Pseudo-projective方法 Nivre和Nilsson (2005)提出一种pseudo-projective方法[53]。首先将非投影结构通过规则转化为投影结构。转化后的投影结构通过依存关系标签记录了转化的过程。这样，非投影结构依存分析就转化为了投影结构依存分析。测试阶段，首先得到一个伪投影结构的依存结构，然后利用依存关系标签转化为非投影结构。

List-based转移系统 Nivre (2008)提出一种面向非投影结构的转移系统，称为list-based方法[37]。系统中每个状态由四部分构成：两个列表用来保存已经处理过的节点，一个队列保存未处理的节点，目前已经产生的弧集合。定义了4种动作。与arc-standard和arc-eager不同的是，所有节点永远不会消失。这种方法通过第二个列表作为缓冲，通过复杂的状态转移建立非投影弧。这种算法的最坏时间复杂度为$O(n^2)$。

加入SWAP操作的转移系统 Nivre对list-based方法进行了改进，提出了一个$O(n)$时间的转移系统[54]。这个系统与arc-standard相似，但是增加了一个新的动作，SWAP，可以将栈顶两个元素位置交换。这样便可以得到非投影结构。

1.3 依存句法分析的挑战与前景

自从2006年开始，CoNLL国际评测一直关注依存句法分析，不但提供了多语言、高质量的树库，并通过对各种方法的比较分析，让研究者们对依存分析问题理解更加清晰，极大的促进了依存句法分析的发展。依存分析已经成为自然语言处理的一个热点问题，方法也越来越成熟，并且在许多领域得到了应用。然而，目前依存句法分析还存在很多挑战，这些挑战也可能是未来依存分析发展的趋势。

(1) **提高依存分析准确率** 目前主流的两种依存分析方法都存在一定的缺

陷。基于图的方法很难融入全局特征。而基于转移的方法虽然原理上可以利用丰富的特征，但是实际使用的特征还是属于局部特征，另外也还存在错误级联的问题（柱搜索只能缓解这个问题）。融合不同依存分析模型的方法可以提高分析性能，但是提高幅度比较有限。似乎只有从新的角度理解这个问题本身，提出新的建模方法，或者应用新的机器学习方法，才有希望大幅度的提高依存分析性能。一些学者提出的利用未标注数据帮助依存分析模型是一个很好的思路，值得深入研究。

(2) **提高依存分析效率** 基于图的依存分析方法融入高阶特征可以提高性能，但是效率很低，无法适应实际应用的需求。在不明显降低分析性能的前提下，如何提高依存分析效率也是一个很有实际价值的问题。

(3) **领域移植问题** 研究发现，当训练数据领域与测试数据领域不相同时，即使差距不大，也会导致句法分析性能下降很大。以英语为例，从华尔街日报树库移植到Brown语料时，句法分析性能下降近8%[55]。目前依存树库所覆盖的领域、规模都很有限，而标注树库的代价很大。因此解决领域移植问题，对于依存分析的实际应用至关重要。

(4) **语言相关的依存分析** 目前最主流的两种依存分析方法都是语言无关的，纯粹依靠机器学习方法从数据中学习，加入人类知识只能限于特征选择。然而，每种语言都有其特点。因此语言相关的依存分析研究，如针对每种语言的特点设计更有效的模型和算法，利用一些语言特有的资源等，也是很有必要的。近年来，国内学者已经在汉语依存句法分析上做出了很多成绩[56–59]，然而如何利用汉语的特点，提高汉语句法分析的准确率和效率，仍然是一个开放的问题。

1.4 本文的研究内容及章节安排

本研究通过分析汉语的特点，尝试从效率和准确率两个角度提高汉语依存句法分析技术。总的来说，我们做了三个方面的工作。第一方面，我们从算法和汉语本身特点两个角度出发，对基于图的依存句法分析方法进行改进，大幅度提高了分析的效率。第二方面，我们系统深入的研究了汉语词性标注和依存句法分析联合模型，以解决词性标注错误对依存句法分析引入的错误蔓延问题。进而，我们提出一种面向联合模型的分离被动进取训练算法，可以更好的利用联合模型中的词性特征集合，以提高联合模型的词性和句法准确率。第三方面，提出一种基于准同步文法的多树库融合方法，尝试充分利用多树库中包

含的语言学知识，以提高依存句法分析的准确率。整篇论文结构框架如图1-2所示。具体的，本文共含5章，各章内容组织如下：

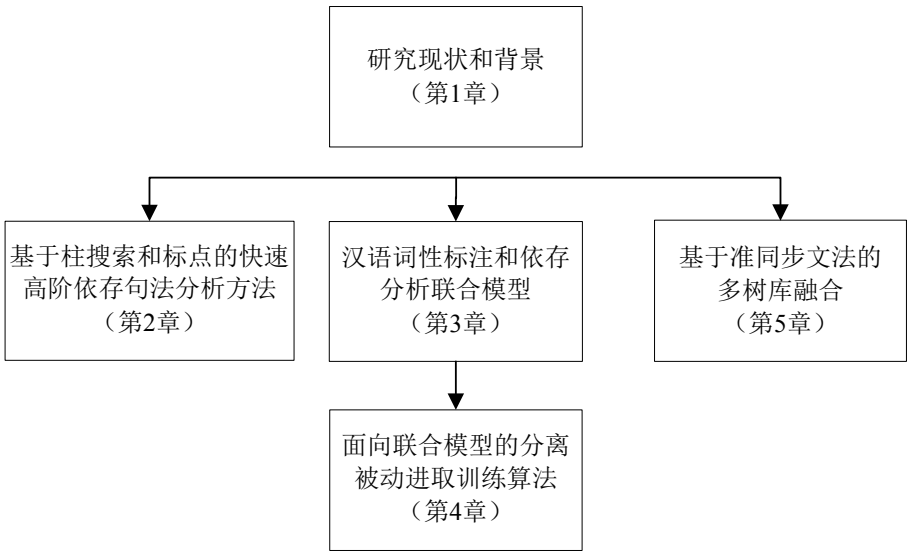


图 1-2 论文总体框架

Fig. 1-2 Structure of this thesis

第1章：本章首先阐明了依存句法分析的课题背景和意义；然后给出了依存句法分析的形式化定义和评价方法；进而，详细介绍了依存句法分析的国内外最新研究现状和面临的挑战；最后列出了本文的内容组织。

第2章：本章针对前人提出的基于动态规划解码算法时间复杂度较高的问题，提出一种基于柱搜索的快速解码算法，不但能够方便的融入高阶特征以提高句法分析准确率，并且可以保证较低的时间复杂度；进而，我们针对汉语频繁使用逗号来标识停顿的特点，提出一种利用标点符号进行长句分短句的依存句法分析方法，可以大幅度的提高汉语依存句法分析在长句上的效率，并且准确率也有所提高。

第3章：针对汉语词性标注准确率较低，对依存句法分析引入严重的错误蔓延问题，本章系统深入的研究了汉语词性标注和依存句法分析联合模型。我们对基于图的依存句法分析解码算法进行扩展，提出基于动态规划的联合解码算法；为了解决搜索空间巨大的问题，我们提出一种基于边缘概率的词性候选裁剪策略，有效的约束了搜索空间，保证了联合解码算法的效率。实验表明，联合模型可以提高句法分析准确率。

第4章：通过分析，我们发现传统的训练算法不适合联合模型，导致联合模型中的词性特征无法充分贡献其消歧作用。本章对传统的被动分离（Passive-

Aggressive）训练算法进行改进，提出一种分离被动进取训练算法（Separately Passive-Aggressive, SPA）在线训练算法，在特征权重更新阶段，针对词性特征和句法特征采用两个单独的更新步长。SPA算法可以更好的平衡联合模型中的词性特征和句法特征的权重。实验证明，SPA可以同时提高词性和句法分析准确率。

第5章：由于标注数据的规模很大程度上决定了统计模型的准确率，并且汉语存在多个句法树库，因此本章从增大标注数据规模的角度，提出一种基于准同步文法（Quasi-synchronous Grammar, QG）的多树库融合方法，尝试充分利用多树库中包含的人类语言学知识，以提高依存句法分析的准确率。对于多树库融合的想法，最大的挑战是如何处理多个树库的标注规范的不一致，即多个树库对同一语言现象采用不同的标注策略。我们提出使用转换模式（Transformation Pattern，TP）来刻画标注规范之间的不一致，然后基于TP形成QG特征。QG特征可以很自然的融入到依存句法分析模型中，并且用来指导模型作出更好的决策。实验证明，我们的多树库融合方法可以提高句法分析的准确率。

需要说明的是，本论文提出的基于柱搜索和标点的快速高阶依存分析方法、词性标注和依存分析联合模型、多树库融合这三个方面的工作之间并不矛盾，是互相联系、互相促进的。基于标点的二阶段分析方法同样可以提高联合模型的解码速度；基于柱搜索的解码算法同样可以降低联合模型的解码时间复杂度；多树库融合显然也可以提高联合模型的准确率。另外，虽然本文的工作专注于汉语依存句法分析，但本文提出的方法大部分并不局限于某种语言，可以方便地移植到其他语言。

第 2 章 基于柱搜索和标点的快速高阶依存句法分析方法

2.1 引言

依存句法分析存在两种主流方法。一种是基于图的方法，将依存句法分析看成从完全有向图中寻找最大生成树的问题。为了设计快速的基于动态规划的解码算法，基于图的模型需要做比较强的独立假设，只允许模型中融入有限几种子树的特征。目前的研究表明，当模型可以融入更复杂的子树信息，那么句法分析的准确率就会提升。但是如果模型尝试加入更复杂的子树特征时，就需要对解码算法进行大的调整，并且解码算法的复杂度也变得非常高。另一种主流的依存句法分析方法是基于转移的方法，通过一个移进规约转移动作序列构建一棵依存句法树。依存分析问题被建模为寻找最优动作序列的问题。基于转移的方法最初采用贪心搜索，后来开始使用柱搜索（beam search）增大搜索空间[35, 36, 60, 61]，再后来Huang和Sagae (2010)提出在柱搜索中加入动态规划策略，将等价状态合并，进一步增大搜索空间[34]。和基于图的方法不同，基于转移的方法由于采用近似的解码算法，因此可以使用之前已形成的所有子树信息。增加新的复杂特征时，只需要在解码过程中直接抽取并且计算分值即可。

由于基于图的方法在一些国际评测人中比基于转移的方法表现更好，因此我们专注于基于图的方法。从前面的分析可以看出，对于基于图的方法，存在分析速度和分析准确率的矛盾。为了提高分析准确率，我们需要在基于图的模型中融入更复杂的子树特征，但是这样又会导致解码算法复杂度变高，解码速度下降。因此，本文尝试在基于图的方法中引入近似的柱搜索算法，从而可以在保证搜索时间复杂度的前提下，尽量融入更复杂的子树特征。我们参加了CoNLL 2009多语语义依存分析评测任务[8], 在包括加州大学伯克利分校、爱丁堡大学、日本东京大学、瑞士日内瓦大学、瑞典隆德大学、德国DFKI、香港城市大学、中科大、武汉大学等在内的21家参加单位中，我们的系统在联合任务上取得第1名，在依存句法分析任务上列第3名。

然而，我们实验发现，即使使用了基于柱搜索的近似解码算法，高阶依存句法分析模型的效率仍然很低。尤其对于长句子而言，依存分析时间随着输入句子长度增大快速增长。这制约了这些模型在高层自然语言处理系统（NLP）

如机器翻译（MT）中的应用。为此，为了提高高阶依存句法分析模型的分析速度，我们针对汉语，利用标点符号将长句切分为短句，然后以分治的方式先处理每个子句的句法结构，最后分析子句间的句法结构，构成完整的句法结构。由于一些子句无法构成一棵完整的子树，我们对典型的依存分析模型进行了变形，允许子句对应的句法结构为一个森林，从而在第一阶段的子句内依存分析时，很自然的处理这种情况。我们在两个汉语树库上做实验，发现这种基于标点的长句分短句的方法，可以大幅度提高高阶依存句法分析模型的效率，并且可以提高长句的分析准确率。

2.2 相关工作

2.2.1 高阶依存句法分析的相关工作

Eisner (1996)提出了一种时间复杂度为$O(n^3)$的句法分析解码算法[62]。基于Eisner的解码算法，McDonald等(2005)提出了一阶（first-order）依存分析模型[19]。一阶模型假设依存树中依存弧之间是互相独立的，因此依存树的分值为每条依存弧的分值之和，也就是说，一阶模型中贡献分值的子树只包含一条依存弧。进而，McDonald和Pereira (2006)提出了二阶（second-order）依存分析模型[20]。二阶模型中依存树的分值增加了由依存树中所包含的所有兄弟子树（sibling）的分值。他们对Eisner算法进行了扩展，提出一种新的解码算法，其时间复杂度为$O(n^3)$。Carreras (2007)提出一种表达能力更强的二阶依存分析模型，可以融入祖孙子树特征（grandparent parts）[21]。然而这种二阶模型需要时间复杂度为$O(|\mathcal{L}|n^4)$的解码算法，其中$\mathcal{L}$表示依存句法关系类型集合。另外，这种二阶模型融入的祖孙子树特征仅限于两边最远的孙子节点（outermost grandchildren）。Koo和Collins (2010)提出了三阶依存模型（third-order）的解码算法，其复杂度为$O(n^4)$[22]。他们提出了两种不同的三阶模型，分别命名为Model 1和Model 2。Model 1可以融入祖孙兄弟子树（grand-sibling parts）特征，而Model 2可以进一步融入三兄弟子树（tri-sibling parts）特征。在英语和捷克语上的实验结果表明，Model 1和Model 2的依存分析准确率基本相同。

Chen等(2010)提出在基于图的模型中融入更加丰富的历史信息[63]。其思想和本文的工作非常类似。每个span中存储更加丰富的历史信息（signature），这样span合并时就可以使用更丰富的特征。对于这样的模型，很难设计出基于动态规划的解码算法。因此，作者采用柱搜索的解码算法。另外，作者尝试了不

同的策略来决定融入哪些历史特征，如一个子句内的（由标点符号决定）的依存弧历史，或使用跨度不超过一个阈值（如3、5、或10）的依存弧等。

Zhang和McDonald (2012)提出在基于图的模型中融入高阶特征[64]，包括三阶特征、配价特征（valency）、复杂的依存关系类型特征等。他们也采用柱搜索的解码算法，同时使用cube pruning策略以$O(klogk)$的时间复杂度快速得到k-best结果。另外，他们还使用了Huang和Sagae (2010)提出的面向转移的依存句法分析方法的动态规划策略[34]，将beam中等价的状态进行合并，进一步增大搜索空间。他们的实验表明，由于使用了高阶特征，基于柱搜索的方法可以取得和基于动态规划的方法更高的准确率，并且句法分析的效率也更高。

本章中的工作是我们2009年参加CoNLL 2009国际评测任务时提出的。然而，由于当时我的研究水平、视野非常有限，没有继续深入的做这个想法。非常遗憾，后来的学者们基于这个想法做出了有价值的研究成果。本章阐述了我们当时的方法及实验结果，并且结合最新的研究结果，进行讨论和分析。

2.2.2 利用标点符号帮助句法分析的相关工作

标点符号在书面文本中扮演着非常重要的作用。标点符号通常用来指示句子的边界，从句的边界，语气的中间停顿，引用等。没有标点符号，书面文本会变得很晦涩，并且会产生更多的歧义。汉语文言文以前就没有标点，需要根据人们的理解去寻找停顿，不利于让读者更好的理解。Nunberg (1990)从语言学角度讨论了标点符号对于理解文本有哪些帮助[65]。Collins (2003)在他的短语结构句法分析模型中融入了针对标点符号的约束和参数，提高了句法分析的准确率[66]。White和Rajkumar (2010)发现在CCG语法中加入细粒度的针对标点符号的规则，可以提高文本生成（surface realization）的效果[67]。

一些学者之前也曾提出利用标点符号将一个句子切分成多个片段，然后进行句法分析的想法。这个想法最大的困难是如何处理不适当的切分，即简单切分后产生的片段可能不对应一个完整的句法结构。而不适当的切分会影响后续的句法分析。Jin等(2004)首先将逗号分为两种：从句间逗号（inter-clause commas）和从句内逗号（intra-clause commas），只使用从句间逗号间进行切分。这样，他们保证了切分出的每一个片段都会对应一个完整句法结构[68]。然而，他们的方法只考虑逗号，并且似乎只对包含两个从句的简单句子有效，很难广泛应用于大规模实际数据中。Li等(2005)在片段内句法分析之后，利用一些启发式规则，检测出不适当的切分并对相应的子树进行合并[69]。他们的方法针对

短语结构句法分析，并且他们使用的启发式规则只考虑了并列结构。感觉很难在依存分析上借鉴这种思路。毛奇等(2007)首先识别出哪些片段对应完整的句法结构，称之为可单独解析块（separately parsing phrases），在片段内分析时只对这些可单独解析块进行分析[70]。这种方法最大的困难在于如何自动识别可独立解析块。识别可独立解析块本身需要准确的句法结构信息，如果单纯利用文本上下文信息，很难达到较高的准确率。本文提出一种基于标点的二阶段依存句法分析方法，不需要显式的识别无法构成完整子树的子句，而是在分析子句内句法结构的同时，将未来和其他子句产生关系的词识别出来，交给第二阶段的子句间分析。

2.3 基于柱搜索的高阶依存句法分析模型

对于输入句子$\mathbf{x} = w_1...w_n$及其词性序列$\mathbf{t} = t_1...t_n$，基于图的句法分析模型尝试从有向图中寻找一颗最大生成树。

$$\hat{\mathbf{d}} = \arg\max_{\mathbf{d}} \mathrm{Score}(\mathbf{x}, \mathbf{d}) \tag{2-1}$$

注意，词性序列**t**和词序列**x**一样，都是依存句法分析的输入，并且对依存句法分析起着非常关键的作用。简洁起见，公式中省略了词性序列**t**。形式化地，一个依存句法树表示为$\mathbf{d} = \{(h, m, l) : 0 \le h \le n, 1 \le m \le n, l \in \mathcal{L}\}$，其中$(h, m, l)$表示一个从核心词（head, father）$w_h$到修饰词（modifier，dependent, child）w_m的依存弧，l表示依存弧的关系类型，$\mathcal{L}$是依存弧关系类型的集合。依存弧的关系类型用来表示两个词之间的句法或语义关系。图2-1给出了一棵依存树的例子，最后一列给出的是每个词的人工标注的词性，其中句法结构和词性都遵循CoNLL 2009中汉语数据集的标注规范。其中，依存弧$(6, 2, \mathrm{TMP})$表示“今年”修饰“为”，并且依存关系类型为时间状语（TMP）。这条依存弧也可以表示为$2 \xleftarrow{\mathrm{TMP}} 6$。

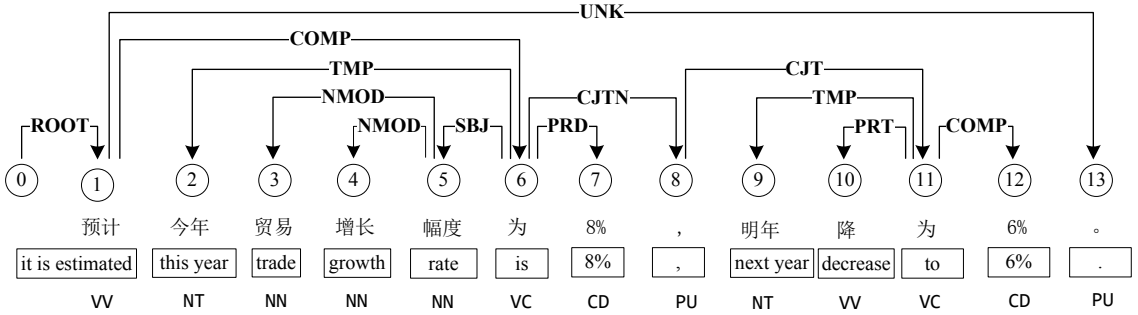


图 2-1 一棵句法树的例子

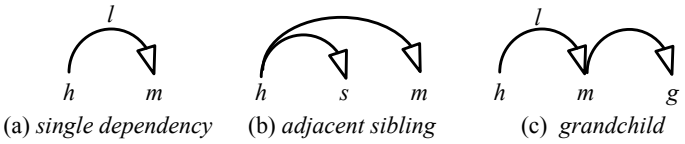
Fig. 2-1 An example dependency tree.

2.3.1 高阶依存句法分析模型定义

基于图的依存句法分析模型将句子看成一个完全有向图，然后尝试从图中找出一个分值最高的依存树。为了保证基于动态规划的全局搜索算法的效率，基于图的模型需要做很强的独立假设。假设句子对应的依存树中，只有存在于某些特殊结构（子树，subtree）中的依存弧之间才互相联系和影响，其他依存弧之间则互相独立。在这种假设下，一个依存树的分值便分解为一些子树的分值的和。

$$\begin{aligned}\mathrm{Score}(\mathbf{x},\mathbf{d}) &= \mathbf{w}\cdot\mathbf{f}(\mathbf{x},\mathbf{d}) \\ &= \sum_{p\subseteq\mathbf{d}} \mathrm{Score}_{\mathrm{subtree}}(\mathbf{x},p)\end{aligned} \tag{2-2}$$

其中，p表示一个独立假设允许的子树。p包含一个或多个$\mathbf{d}$中的依存弧。$\mathrm{Score}_{\mathrm{subtree}}(\mathbf{x},p)$表示由$p$贡献的分值。$\mathbf{f}(\mathbf{x},\mathbf{d})$是$(\mathbf{x},\mathbf{d})$对应的聚合的句法特征向量，$\mathbf{w}$是句法特征的权重向量。



(a) *single dependency* (b) *adjacent sibling* (c) *grandchild*

图 2-2 高阶模型中使用的几种子树。

Fig.2-2 Different types of scoring subtrees used in our high-order graph-based models.

从第2.2.1节的讨论可知，基于图的句法分析模型融入更复杂的子树特征时可以提高依存分析准确率。为此，本文工作借鉴Carreras (2007)的工作[21]，融入了三种子树结构，如图2-2所示。第一种子树中只包含一个依存弧；第二种子树考虑同方向的相邻的两条兄弟依存弧；第三种子树考虑构成祖孙关系的两条依存弧。注意，和Carreras的模型不同的是，我们的高阶模型中对于所有的孙子节点（如图2-2-(c)中的g），都产生一个特征向量[1]；而在Carreras的模型中，只考虑最远孙子节点构成的特征向量，也就是说，只有当g是m最左（或最右）的修饰词时，才会触发一个祖孙特征向量。

根据高阶模型融入的子树特征，一个依存树的分值可以细化为：

[1]从这一点看，我们的方法和Koo和Collins (2010)的工作更加接近[22]。我们在汉语上的结果表明，使用所有的孙子节点对应的祖孙特征向量，可以提高句法分析准确率。具体的结果是，当同样是无句法关系类型的二阶模型时，Koo的模型比Carreras的模型准确率更高。另外，当Carreras模型带句法关系类型时，准确率会比Koo的无句法关系类型的模型高一些。

$$\begin{aligned}\mathrm{Score}(\mathbf{x},\mathbf{d}) = &\sum_{\{(h,m,l)\}\subseteq\mathbf{d}} \mathrm{Score}_{\mathrm{dep}}(\mathbf{x},h,m,l)\\ &+\sum_{\{(h,m),(h,s)\}\subseteq\mathbf{d}} \mathrm{Score}_{\mathrm{sib}}(\mathbf{x},h,s,m)\\ &+\sum_{\{(h,m,l),(m,g)\}\subseteq\mathbf{d}} \mathrm{Score}_{\mathrm{grd}}(\mathbf{x},h,m,l,g)\end{aligned} \tag{2-3}$$

其中$\mathrm{Score}_{\mathrm{dep/sib/grd}}(.)$表示图2-2中三种子树对应的分值。由于我们采用线性模型（linear model），这些分值可以表示为：

$$\begin{aligned}\mathrm{Score}_{\mathrm{dep}}(\mathbf{x},h,m,l) &= \mathbf{w}_{\mathrm{dep}}\cdot\mathbf{f}_{\mathrm{dep}}(\mathbf{x},h,m,l)\\ \mathrm{Score}_{\mathrm{sib}}(\mathbf{x},h,s,m) &= \mathbf{w}_{\mathrm{sib}}\cdot\mathbf{f}_{\mathrm{sib}}(\mathbf{x},h,s,m)\\ \mathrm{Score}_{\mathrm{grd}}(\mathbf{x},h,m,l,g) &= \mathbf{w}_{\mathrm{grd}}\cdot\mathbf{f}_{\mathrm{grd}}(\mathbf{x},h,m,l,g)\end{aligned} \tag{2-4}$$

其中$\mathbf{f}_{\mathrm{dep/sib/grd}}(.)$表示图2-2中三种子树对应的特征向量，而$\mathbf{w}_{\mathrm{dep/sib/grd}}$表示对应的特征权重向量。

2.3.2 高阶依存句法分析模型的特征集合

我们借鉴McDonald (2006)在其博士论文中使用到的依存弧和兄弟弧特征[25]，以及Carreras (2007)提出的祖孙弧特征[21]，如表2-1所示。其中，$\circ$表示字符串的连接；r表示依存弧$h \curvearrowright m$的方向，例如左弧为“*L*”，右弧为“*R*”；d表示依存弧$h \curvearrowright m$中两个词的距离（我们将$|h-m|$离散化到$\{1,2,3,[4,6],[7,\infty)\}$）；r′表示依存弧$g \curvearrowright h$的方向；*b*表示介于*h*和*m*之间的一个下标。例如，图2-1中的依存弧$(1,6,\mathrm{COMP})$对应的编号为07的特征实例化后为：“07 = COMP + NN + 为+ R + [4,6]”；兄弟弧$(1,6)$和$(1,13,\mathrm{UNK})$对应的编号为20的特征实例化后为：“20 = 预计+ VC + PU + R”；祖孙弧$(1,6,\mathrm{COMP})$和$(6,5)$对应的编号为28的特征实例化后为：“28 = COMP + VV + VC + 幅度+ R + L”。对于所有和依存关系类型*l*相关的特征，我们去掉关系类型构成一个新的特征。

CoNLL 2009年多语依存句法分析和语义角色标注联合任务中，有些语言还提供了词干（lemma）和其他形态特征。对于词干信息，我们将表中2-1中每一个与词语w_i相关的特征中的词语替换为词干，构成一个新的特征。由于每个词的形态信息可能包含多个项，我们的做法和Bohnet参加评测时的方法类似[71]，将一个依存弧中两个词的形态信息两两组合，加上依存弧的方向和距离，构成新的特征。简便起见，我们没有在表中列出，具体请参考Bohnet (2010)的论文[72]。值得注意的是，由于汉语数据中没有词干和形态变化信息，因此我们只采用表中所列的特征集合。

表 2-1 高阶句法分析模型中使用的特征集合。

Table.2-1 Syntactic features used in our high-order dependency parsing model.

Dependency Features: $\mathbf{f}_{\text{dep}}(\mathbf{x}, h, m, l)$		
01: $w_h \circ \mathsf{r}$	02: $t_h \circ \mathsf{r}$	03: $w_m \circ \mathsf{r}$
04: $t_m \circ \mathsf{r}$	05: $l \circ t_h \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	06: $l \circ w_h \circ t_m \circ \mathsf{r} \circ \mathsf{d}$
07: $l \circ t_h \circ w_m \circ \mathsf{r} \circ \mathsf{d}$	08: $l \circ w_h \circ w_m \circ \mathsf{r} \circ \mathsf{d}$	09: $l \circ t_h \circ t_{h+1} \circ t_{m-1} \circ t_m \circ \mathsf{r} \circ \mathsf{d}$
10: $l \circ t_h \circ t_{h+1} \circ t_m \circ t_{m+1} \circ \mathsf{r} \circ \mathsf{d}$	11: $l \circ t_{h-1} \circ t_h \circ t_{m-1} \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	12: $l \circ t_{h-1} \circ t_{h+1} \circ t_{m-1} \circ t_m \circ \mathsf{r} \circ \mathsf{d}$
13: $l \circ t_{h-1} \circ t_h \circ t_{h+1} \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	14: $l \circ t_h \circ t_{m-1} \circ t_m \circ t_{m+1} \circ \mathsf{r} \circ \mathsf{d}$	15: $l \circ t_h \circ t_{h+1} \circ t_m \circ \mathsf{r} \circ \mathsf{d}$
16: $l \circ t_h \circ t_{m-1} \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	17: $l \circ t_h \circ t_m \circ t_{m+1} \circ \mathsf{r} \circ \mathsf{d}$	18: $t_h \circ t_b \circ t_m \circ \mathsf{r} \circ \mathsf{d}$
Sibling Features: $\mathbf{f}_{\text{sib}}(\mathbf{x}, h, s, m)$		
19: $t_h \circ t_s \circ t_m \circ \mathsf{r}$	20: $w_h \circ t_s \circ t_m \circ \mathsf{r}$	21: $t_h \circ w_s \circ t_m \circ \mathsf{r}$
22: $t_h \circ t_s \circ w_m \circ \mathsf{r}$	23: $t_s \circ t_m \circ \mathsf{r}$	24: $w_s \circ w_m \circ \mathsf{r}$
25: $t_s \circ w_m \circ \mathsf{r}$	26: $w_s \circ t_m \circ \mathsf{r}$	
Grandparent Features: $\mathbf{f}_{\text{grd}}(\mathbf{x}, h, m, l, g)$		
27: $l \circ t_h \circ t_m \circ t_g \circ \mathsf{r} \circ \mathsf{r}'$	28: $l \circ t_h \circ t_m \circ w_g \circ \mathsf{r} \circ \mathsf{r}'$	29: $l \circ w_h \circ t_m \circ t_g \circ \mathsf{r} \circ \mathsf{r}'$
30: $l \circ t_h \circ w_m \circ t_g \circ \mathsf{r} \circ \mathsf{r}'$	31: $l \circ t_m \circ t_g \circ \mathsf{r} \circ \mathsf{r}'$	32: $l \circ w_m \circ w_g \circ \mathsf{r} \circ \mathsf{r}'$
33: $l \circ t_m \circ w_g \circ \mathsf{r} \circ \mathsf{r}'$	34: $l \circ w_m \circ t_g \circ \mathsf{r} \circ \mathsf{r}'$	

2.3.3 基于柱搜索的高阶依存句法分析解码算法

Eisner (1996)提出一种通用的面向双词汇化语法（bilexical grammar）的动态规划解码算法，时间复杂度为$O(n^3)$[24]。这个算法通常被称为*Eisner 算法*。Eisner使用chart来组织搜索过程，Chart中的每个元素记录了对应输入子串的分值最高的span。一个span指一个输入子串对应的不完整分析结果，只有边界词（首尾两个词）可以和其他span建立弧。Eisner算法自底向上的每次将两个小span合并成一个大span，直到得到整个句子的分析结果[23, 24]。我们在第三章第3.3.2.2节对Eisner算法进行了详细介绍。

为了融入高阶特征，并且不会大幅度增大算法的时间复杂度，我们对Eisner算法进行扩展，提出一种柱搜索的近似解码算法。本文使用图2-3和算法2-1详细解释我们的解码算法。柱搜索的解码算法沿用Eisner算法中采用的数据类型和合并操作。最大的区别在于，每个span中包含更多的依存弧历史信息（signature），这样在span合并时就可以使用这些依存弧历史信息，构成复杂的高阶特征。但是，这也导致算法无法满足动态规划的性质，因此我

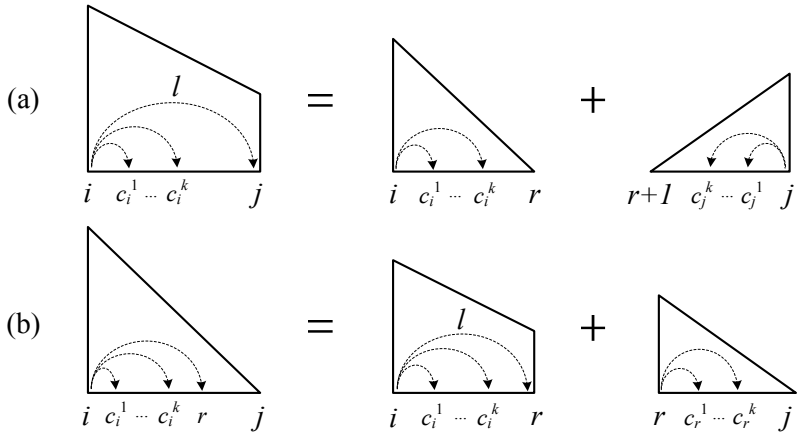


图 2-3 基于柱搜索的高阶依存分析模型的解码算法中使用到的基本数据结构和操作。

Fig. 2-3 The data structures and derivations of beam-search based decoding algorithm for the high-order parsing model.

1. $\forall 0 \le k < K, 0 \le i \le n, t_i \in \mathcal{T}:\ C^k_{i\rightarrow i} = 0, C^k_{i\leftarrow i} = 0$ // *initialization*
2. **for** $w = 1$ **to** n **do** // *span width*
3. **for** $i = 0$ **to** $(n - w)$ **do** // *span start index*
4. $j = i + w$ // *span end index*
5. $I^K_{i\curvearrowright j} = \max^K_{l\in\mathcal{L};0\le r<j;0\le k_1,k_2<K}\{C^{k_1}_{i\rightarrow r} + C^{k_2}_{r+1\leftarrow j} + \mathrm{Score}_{\mathrm{dep}}(\mathbf{x}, i, j, l) + \mathrm{Score}_{\mathrm{sib}}(\mathbf{x}, i, c^k_i, j) + \sum_{c^b_j} \mathrm{Score}_{\mathrm{grd}}(\mathbf{x}, i, j, l, c^b_j)\}$ // *corresponding to Fig. 2-3-(a).*
6. $I^K_{i\curvearrowleft j} = \max^K_{l\in\mathcal{L};0\le r<j;0\le k_1,k_2<K}\{C^{k_1}_{i\rightarrow r} + C^{k_2}_{r+1\leftarrow j} + \mathrm{Score}_{\mathrm{dep}}(\mathbf{x}, j, i, l) + \mathrm{Score}_{\mathrm{sib}}(\mathbf{x}, j, c^k_j, i) + \sum_{c^b_i} \mathrm{Score}_{\mathrm{grd}}(\mathbf{x}, j, i, l, c^b_i)\}$
7. $C^K_{i\rightarrow j} = \max^K_{i<r\le j;0\le k_1,k_2<K}\{I^{k_1}_{i\curvearrowright r} + C^{k_2}_{r\rightarrow j} + \sum_{c^b_r} \mathrm{Score}_{\mathrm{grd}}(\mathbf{x}, i, r, l, c^b_r)\}$
// *l is the label of* (i, r) *stored in* $I^{k_1}_{i\curvearrowright r}$. *corresponding to Fig. 2-3-(b).*
8. $C^K_{i\leftarrow j} = \max^K_{i\le r<j;0\le k_1,k_2<K}\{C^{k_1}_{i\leftarrow r} + I^{k_2}_{r\curvearrowleft j} + \sum_{c^b_r} \mathrm{Score}_{\mathrm{grd}}(\mathbf{x}, j, r, l, c^b_r)\}$
// *l is the label of* (j, r) *stored in* $I^{k_2}_{r\curvearrowleft j}$.
9. **end for**
10. **end for**
11. **return** $C^K_{0\rightarrow n}$

算法 2-1 基于柱搜索的高阶依存句法分析模型的解码算法

Algo. 2-1 The beam-search decoding algorithm for higher-order dependency parsing

们采用基于柱搜索的近似搜索算法。例如，图2-3-(a)最左侧的span，记为$I_{i\curvearrowright j}$，表示一个覆盖$w_i...w_j$的局部依存结构，并且w_i和w_j构成一个依存弧$i \curvearrowright j$。我们称$I_{i\curvearrowright j}$为一个*不完整*（*incomplete*）*span*，意思是，在将来的操作中，w_j还有可能获得新的右侧的儿子，即w_j右侧的修饰词集合是*不完整*的。注意，在$I_{i\curvearrowright j}$中，w_j已经找到了其所有的左儿子。除此之外，在我们的算法中，$I_{i\curvearrowright j}$还会额外存储依存弧$i \curvearrowright j$的依存关系类型l，以及i的其他（右）儿子节点$\{c_i^1, ..., c_i^k\}$。图2-3-(b)左侧的span，记为$C_{i\rightarrow j}$，表示一个覆盖$w_i...w_j$的局部依存结构，并且w_j是w_i的一个子孙。我们称$C_{i\rightarrow j}$是一个*完整*（*complete*）*span*，意思是，w_j找到了其所有的左儿子，并且w_j没有右儿子。因此在将来的操作中，w_j不能再增加新的修饰词，即w_j的修饰词集合是*完整*的。除此之外，$C_{i\rightarrow j}$需要记录i的所有右儿子$\{c_i^1, ..., c_i^k, r\}$。这样在未来的合并操作中，算法可以使用这些信息构建高阶特征。在$I_{i\curvearrowright j}$和$C_{i\rightarrow j}$中，w_i又称为span的核心词（span head）。类似的，$I_{i\curvearrowleft j}$和$C_{i\leftarrow j}$表示以w_j为核心的相反方向的两种span。简洁起见，图2-3中省略了产生这两种span的操作。

算法2-1的第5行尝试找到K个近似最优的不完整span $I_{i\curvearrowright j}$。K称为柱宽度（beam size）。给定l和r，通过合并$C_{i\rightarrow r}$和$C_{r+1\leftarrow j}$产生一个候选的$I_{i\curvearrowright j}$，如图2-3-(a)所示。$C_{i\rightarrow r}^{k_1}$表示top-K的$C_{i\rightarrow r}$中的第k_1个；$C_{r+1\leftarrow j}^{k_2}$表示top-$K$的$C_{r+1\leftarrow j}$中的第$k_2$个。这个合并操作引入一个新的依存弧$(i, j, l)$，因而产生的span增加这个依存弧对应的分值：$\mathsf{Score}_{\mathrm{dep}}(\mathbf{x}, i, j, l)$。另外，$C_{r+1\leftarrow j}^{k_2}$包含了所有$j$的左儿子$c_j^1...c_j^k$。对于每一个左儿子$c_j^b$，都需要增加一个祖孙弧特征对应的分值$\mathsf{Score}_{\mathrm{grd}}(\mathbf{x}, i, j, l, c_j^b)$。最后，$C_{i\rightarrow r}^{k_1}$包含了所有$i$的右儿子$c_i^1...c_i^k$，新增加的右儿子$j$和最后一个$c_i^k$构成一对兄弟弧，因此需要累加对应的分值$\mathsf{Score}_{\mathrm{sib}}(\mathbf{x}, i, c_i^k, j)$；如果$i$的右儿子为空，那么$j$就是$i$的第一个右儿子，那么分值函数变为$\mathsf{Score}_{\mathrm{sib}}(\mathbf{x}, i, -, j)$。McDonald(2006)的博士论文详细讨论了如何使用第一个儿子对应的兄弟弧特征[25]。另外，我们还可以很容易的在算法中加入最后一个儿子对应的兄弟弧特征。简单起见，我们省略这部分介绍。算法遍历所有的l、r、k_1和k_2，寻找K个近似最优的$I_{i\curvearrowright j}$，存储在$I_{i\curvearrowright j}^K$中。

算法2-1第7行尝试确定最优的完整span $C_{i\rightarrow j}$。给定r，通过合并$I_{i\curvearrowright r}$和$C_{r\rightarrow j}$可以产生一个候选的$C_{i\rightarrow j}$，如图2-3-(b)所示。类似的，$I_{i\curvearrowright r}^{k_1}$表示top-$K$的$I_{i\curvearrowright r}$中的第$k_1$个；$C_{r\rightarrow j}^{k_2}$表示top-$K$的$C_{r\rightarrow j}$中的第$k_2$个。$C_{r\rightarrow j}^{k_2}$中存储了所有$r$的右儿子$c_r^1...c_r^k$。对于每一个右儿子$c_r^b$，都需要增加一个祖孙弧特征对应的分值$\mathsf{Score}_{\mathrm{grd}}(\mathbf{x}, i, r, l, c_r^b)$。算法遍历所有的$r$、$k_1$和$k_2$，寻找$K$个近似最优的$C_{i\rightarrow j}$，存储在$C_{i\rightarrow j}^K$中。

类似的，算法第6行和第8行分别确定以w_j为核心的非完整和完整span。简洁起见，我们不做详细解释。最终，span $C_{0,n}^{K}$保存了top-K的完整依存树的分值，通过回溯（backtracking）的方法，可以获得这个近似最优的依存树。可以证明，这个算法按照公式2-3正确计算每一个span的分值。

算法需要存储$2Kn^2$个完整span和$2Kn^2$个不完整span。因此空间复杂度为$O(Kn^2)$。不考虑柱宽度，并且假设每个词的修饰词数目很小（即遍历儿子节点构成祖孙弧特征的复杂度为常数），那么第5行和第6行的时间复杂度都为$O(n|\mathcal{L}|)$；第7行和第8行的时间复杂度都为$O(n)$；这个算法的复杂度为$O(n^3|\mathcal{L}|)$。我们采用一阶依存弧特征对依存弧关系类型进行裁剪。对任意一条依存弧(i, j)，我们求出最高分值的依存关系类型$\hat{l} = \arg\max_l \mathrm{Score}_{\mathrm{dep}}(\mathbf{x}, i, j, l)$，然后只保留和$\hat{l}$的分值的差距在一定阈值的依存关系类型。裁剪后，每条依存弧对应的依存关系类型集合也会变得很少。Huang和Chiang提出一种基于cube pruning的快速近似求解K-best解的方法[73]。我们采用他们的方法，这样柱宽度K带来的时间复杂度负荷为$K\log K$。综上所述，算法的时间复杂度为$O(n^3 K\log K)$，远小于Carreras算法的$O(|\mathcal{L}|n^4)$。

最后需要指出的是，这种基于柱搜索的解码算法很容易扩展，从而融入更复杂的子树特征。具体方法是在span中记录所需的额外信息，解码过程中从span中提取相应的信息构成高阶特征。

2.3.4 高阶依存句法分析训练算法

借鉴McDonald的方法[25]，我们使用在线学习算法K-best MIRA (Margin Infused Relaxed Algorithm) 学习特征权重向量$\mathbf{w}$[28]。在线算法（online algorithm）的基本思想是：每次考察一个训练实例。根据当前权重向量$\mathbf{w}^{(k)}$，获得当前实例的预测结果，根据预测结果中的错误，更新当前特征向量，得到新的特征向量$\mathbf{w}^{(k+1)}$。算法通过多次迭代确定最终的权重向量。每次迭代需要遍历所有训练实例。算法2-2给出伪代码。

算法2-2中第6行根据当前的权重向量$\mathbf{w}^{(k)}$，利用基于柱搜索的解码算法，得到top-K的近似最优的依存树，存储在$\hat{\mathbf{d}}^K$。第7行根据$\hat{\mathbf{d}}^K$对当前的权重向量进行更新，更新后的权重向量为$\mathbf{w}^{(k+1)}$。K-MIRA的更新准则为：

$$\begin{aligned} &\min \|\mathbf{w}^{(k+1)} - \mathbf{w}^{(k)}\| \\ &s.t. \forall \mathbf{d} \in \hat{\mathbf{d}}^K \quad \mathbf{w}^{(k+1)} \cdot (\mathbf{f}(\mathbf{x}^{(j)}, \mathbf{d}^{(j)}) - \mathbf{f}(\mathbf{x}^{(j)}, \mathbf{d})) \geq loss(\mathbf{d}^{(j)}, \mathbf{d}) \end{aligned} \tag{2-5}$$

1. **Input:** Training Data $\mathbb{D} = \{(\mathbf{x}^{(j)}, \mathbf{d}^{(j)})\}_{j=1}^{N}$
2. **Output: w**
3. **Initialization:** $\mathbf{w}^{(0)} = \mathbf{0}; \mathbf{v} = \mathbf{0}; k = 0$
4. **for** $i = 1$ **to** I **do** // *iterations*
5. **for** $j = 1$ **to** N **do** // *traverse the samples*
6. $\hat{\mathbf{d}}^{K} = \arg\max_{\mathbf{d}}^{K} \mathbf{w}^{(k)} \cdot \mathbf{f}(\mathbf{x}^{(j)}, \mathbf{d})$ // *decode based on current feature weights*
7. $\mathbf{w}^{(k+1)}$ = update $\mathbf{w}^{(k)}$ according to $(\mathbf{d}^{(j)}, \hat{\mathbf{d}}^{K})$
8. $\mathbf{v} = \mathbf{v} + \mathbf{w}^{(k+1)}$
9. $k = k + 1$
10. **end for**
11. **end for**
12. $\mathbf{w} = \mathbf{v}/(I \times N)$ // *average the weights*

算法 2-2 K-MIRA训练算法

Algo. 2-2 K-MIRA training algorithm

即对当前的特征向量$\mathbf{w}^{(k)}$做最小的调整，使得对于top-K中任何一个依存树$\mathbf{d}$，正确依存树$\mathbf{d}^{(j)}$的分值和$\mathbf{d}$的分值之差不小于$\mathbf{d}$的错误数。错误函数$loss(\mathbf{d}^{(j)}, \mathbf{d})$的定义为：如果依存弧错误，那么错误数加2，如果依存弧正确而依存关系类型错误，错误数加1。

2.4 基于标点的二阶段快速依存句法分析方法

基于柱搜索的高阶依存句法分析模型融入更多的特征，如兄弟特征、祖孙特征后，依存分析准确率会有较大幅度的提高。但同时特征抽取部分的时间消耗也变大，在解码算法中占的比例越来越大。尤其对于长句子而言，高阶模型的分析时间迅速增长。这制约了高阶依存句法分析模型在高层自然语言处理系统如机器翻译、信息抽取等的应用。为此，我们针对汉语的特点，提出一种基于标点符号的快速依存句法分析方法。

我们观察发现，对于汉语数据，当使用逗号、分号等标点符号将长句切割为子句后，大部分子句都对应一个完整的子树结构。图2-4给出了一个示例。句子使用标点切分后，得到两个子句。第二个子句的依存句法结构对应一个完整的子树，即只有一个根节点。如果每个子句都对应一个完整的子树，我们就可以先对每个子句进行句法分析，然后使用每个子句对应的子树根节点代表整个

子句，最后分析子句间的结构，就可以得到整个句子的依存句法结构。对于不对应完整子树的子句，我们对依存句法分析模型进行了调整，允许句法树存在多个根节点。

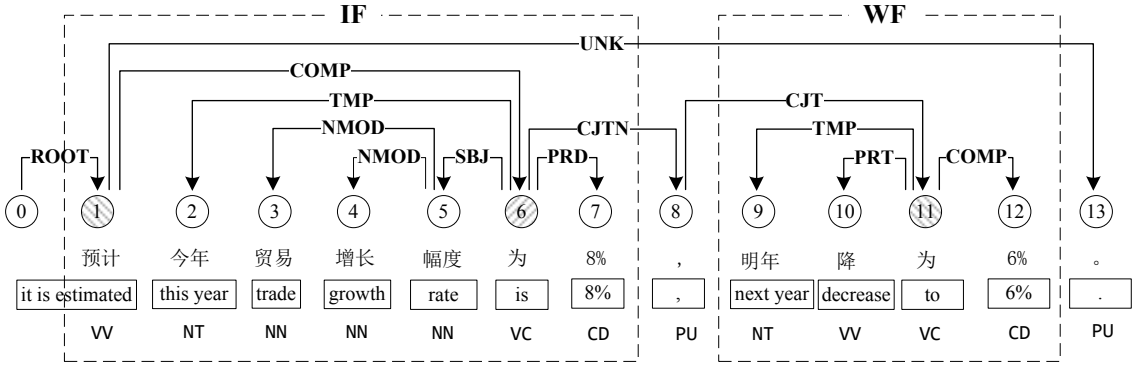


图 2-4 基于标点的长句分短句示例。

Fig.2-4 Example of sentence segmentation using punctuation.

在详细介绍我们的方法之前，我们定义以下术语。

- **切分标点**（split punctuation）： 基于前人的工作和对树库的观察，我们使用逗号、冒号、分号将句子切分为片段，称之为切分标点。之所以使用这些标点符号，是因为按照这些标点切分之后，得到的片段中90%都对应完整的句法结构。而其他标点如顿号，引号等则不满足这个要求。
- **片段**（fragment）： 使用切分标点将句子分成多个部分，每一部分称为一个片段。图2-4中的句子被节点8处的逗号和节点13的句号切分为两个片段，分别为节点1-7和节点9-12。
- **子根**（sub-root）： 如果片段中的一个节点的父亲在这个片段外，或者这个节点的子孙在这个片段外，那么这个节点称为这个片段的子根。前一种情况称为第一类型子根（sub-root#1），后一种情况称为第二类型子根（sub-root#2）。图2-4中，节点1和11属于sub-root#1，节点6属于sub-root#2。
- **良构片段**（well-formed fragment, 简称WF）： 如果一个片段只包含一个子根，如图2-4中的第二个片段，则称它为一个良构片段。
- **病构片段**（ill-formed fragment, 简称IF）： 如果一个片段包含多于一个子根，如图图2-4中的第一个片段，则称它为一个病构片段。

基于标点的快速依存句法分析方法分为两个阶段。

第1步：分析片段内结构 对输入句子的每一个片段进行依存分析。如图2-5所示。一个片段的依存结构允许有多个根节点，每个根节点对应片段的一个子根。第一个片段为病构片段，存在两个子根，其第一阶段的依存结构如

图2-5-(a)所示。我们让两个子根都修饰w_0，并且依存关系都是ROOT。第二个片段为良构片段，只存在一个子根，其第一阶段的依存结构如图2-5-(b)所示。值得提出的是，对第二种类型的子根(sub-root#2)，如图2-5-(a)中的节点6，第一阶段的句法分析比较难识别出来，因为这类子根的父亲节点本身存在于片段内。但是我们的方法和毛奇等提出的利用上下文文本信息直接识别可单独解析块相比，还是有优势的。因为我们的方法在第一阶段的依存分析过程中，至少可以借助句法结构，更好的识别第二类型的子根。

第2步: 分析片段间结构　依存分析所有片段的子根和切分标点构成的伪句子，如图2-6所示。第二阶段尝试分析子句间的依存结构，其分析结果和第一阶段的子句内结构组合，就可以得到整个句子的依存结构。

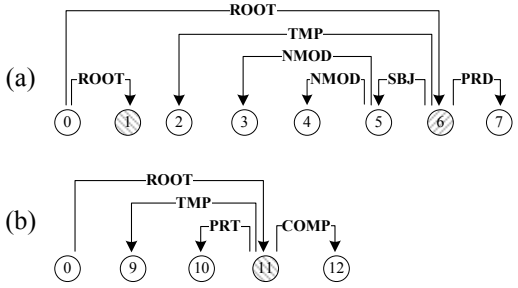


图 2-5 第一阶段的子句内依存分析。

Fig.2-5 Stage 1: parsing the intra-fragement structure.

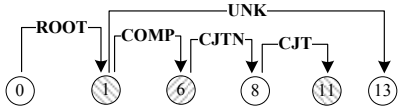


图 2-6 第二阶段的子句间依存分析。

Fig.2-6 Stage 2: parsing the iner-fragement structure.

2.5 实验结果与分析

2.5.1 CoNLL2009国际评测的实验结果

我们参加了CoNLL 2009多语语义依存分析评测任务。CoNLL 2009评测任务包括汉语、英语、日语、捷克语、德语、西班牙语、加泰罗尼亚语七种语言。其中英语、德语、捷克语还提供了跨领域（out of domain, OOD）的测试语料，用以评价系统对不同领域语料的适应能力。七种语言中，捷克语、德语、英语都不同程度的含有非投影依存树。对于这三种语言，本文采用了Nivre和Nilsson

(2005)的pseudo-projective方法[53]，先将含有交叉弧的句子转化为不含有交叉弧，用以训练高阶依存分析器。最后，将分析结果逆向转化，恢复为含有交叉弧的依存树。

本文实验的机器配置是2.5GHz Xeon CPU, 16G内存。依存分析器内存使用峰值为15G，训练时间最多约60个小时。由于CoNLL 2009年评测涉及语言比较多，并且高阶依存模型对内存等资源要求很高，我们没有进行细致的特征选择和参数优化。训练模型过程中，针对不同语言，参数设置为：迭代次数10次，解码生成*K*-best结果中的*K*在训练和测试阶段都采用10。参加评测时，我们采取将依存分析和语义角色标注级联的策略。在21家参加单位中，我们的系统在语义依存分析联合任务上取得第1名的好成绩，依存分析子任务上排名第3。表2-2中给出了最好的三个依存分析系统的性能。评价指标为带依存句法关系的附着分值（LAS）。其中，“average”表示7种语言的测试数据上的平均准确率。我们在日语上取得了最好的成绩。另外，评测组织方基于三种语言在其他领域的数据集，报告了各个系统在领域移植问题上的性能。可以看到，当测试训练语料领域不同时，依存句法分析的准确率大幅度下降。因此，领域移植问题对于依存句法分析而言是一个很大的挑战。

表 2-2 参加CoNLL 2009评测中最好的三个依存分析系统的性能

Table. 2-2 The performance of the top three systems for the CoNLL 2009 shared task

	Gesmundo等[74]	Bohnet[71]	Che等[75]
Average	**85.77**	85.68	85.23 (-0.5)
Catalan	**87.86**	86.35	86.56 (-1.3)
Chinese	76.11	**76.51**	75.49 (-1.0)
Czech	**80.38**	80.11	80.01 (-0.3)
English	88.79	**89.88**	88.48 (-1.4)
German	87.29	**87.48**	86.19 (-1.3)
Japanese	92.34	92.21	**92.57** (—)
Spanish	**87.64**	87.19	87.33 (-1.3)
Czech (ood)	**76.41**	76.4	76.03 (-0.4)
English (ood)	80.84	**82.64**	81.57 (-1.1)
German (ood)	76.77	**77.34**	76.11 (-1.2)

2.5.2 基于标点的二阶段依存分析方法的实验结果

实验中我们使用CoNLL 2009多语语义依存评测中使用的汉语数据[8]。这个依存树库由短语结构树库Penn Chinese Treebank 6.0自动转化而来[76]。

实验中我们使用了两种标准的评测指标：LAS和UAS。另外我们还使用了Dan Bikel开发的randomized parsing evaluation comparator[2]进行显著性检验。

我们使用切分标点对数据集进行句子切分，表2-3给出切分后各种结构的数目。每个句子平均得到3个片段。大约90%的片段都是良构的。子根中大约4.6%属于第二类型子根。

表 2-3 基于标点句子切分后各种结构的数目

Table. 2-3 Statistics after sentence segmentation with punctuation

	句子数	片段数	WF	Sub-root	Sub-root#2
训练集	22,277	66,111	58,172	76,989	3,512
开发集	1,762	5,204	4,556	6,105	317
测试集	2,556	7,724	6,844	8,912	407

表 2-4 测试集上第一阶段分析在子根上的性能

Table. 2-4 Performance of subroot recognition of the first-stage parsing

	Gold	System	Correct	Recall	Precision
Sub-root	8,912	8,527	7,251	81.4%	85.0%
Sub-root#1	8,505	-	7,144	84.0%	-
Sub-root#2	407	-	107	26.3%	-

表 2-5 测试集上的最终性能。

Table. 2-5 Final results on the test set.

Methods	LAS	UAS
One-stage	75.76	80.55
Two-stage	**76.13**	**81.20**
Bohnet[71]	**76.51**	-
Gesmundo等[74]	76.11	-
Che等[5]	75.49	-

第一阶段需要分析句子中的所有片段。为此，我们从训练语料中抽取所有片段对应的依存结构，如图2-5，然后使用这些结构训练一个依存分析器，称

[2]http://www.cis.upenn.edu/~dbikel/software.html

为*FRG-parser*。值得注意的是，FRG-parser允许分析结果中包含多个根节点，每个根节点对应一个子根。

表2-4给出了测试集上第一阶段分析在子根上的性能。可以看到，FRG-parser在sub-root#2上效果很差，召回率很低。可以从三个方面解释。(1) sub-root#2在训练集中数目较少，因此模型很难充分学习。(2) 不同于sub-root#1，sub-root#2实际上依存于本片段内的节点，因此这条依存弧本身的权重可能比较大，因此模型很难将其作为根节点，依存于伪节点w_0。(3) 判断sub-root#2需要根据是否有子孙在其他片段中，而分析片段内结构时无法获得这个信息。不过，sub-root#2的数目很少，因此对分析性能影响不大。

表 2-6 在不同句长上的性能比较。

Table.2-6 Performance comparison on sentence of different length.

句长范围	$len \le 27$		$28 \le len \le 50$		$51 \le len$	
句子数	1,420		826		310	
平均句长	15.6		36.8		66.7	
	LAS	UAS	LAS	UAS	LAS	UAS
One-stage	79.69	84.28	75.53	80.38	71.90	76.81
Two-stage	79.75	84.59	76.01	81.18	72.43	77.72
	+0.06	+0.31	+0.48	+0.80	+0.53	+0.91
	$p = 0.42$	$p = 0.17$	$p = 0.05$	$p = 0.003$	$p = 0.07$	$p = 0.008$

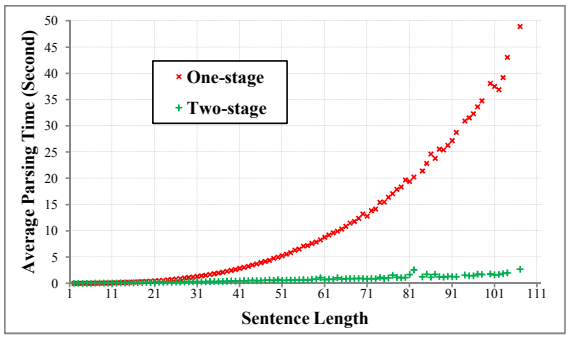


图 2-7 在不同句子长度上的效率对比

Fig.2-7 Efficiency comparison on sentences of different length.

第二阶段，需要分析由所有片段子根和切分标点构成的伪句子。为此，我们从训练语料中抽取所有如图2-6中的结构，然后训练得到一个依存分析器，称为*INTER-parser*。

两个阶段的分析结果结合起来构成整个句子的完整结构。表2-5给出各种方法在测试集上的最终性能。我们提出的二阶段方法优于传统的一阶段方法。

二阶段方法与一阶段方法相比，在LAS上提高了0.37% (显著性测试$p = 0.02$)，UAS上提高了0.65% ($p = 10^{-4}$)。

表2-5中也列出了CoNLL 2009评测任务性能最好的三个系统在汉语上的LAS。Bohnet也使用了采用了Carreras (2007)的二阶依存句法模型，使用了动态规划解码算法和丰富的特征[71]。Gesmundo等使用了一个基于转移的生成式模型，通过潜在变量表示特征[74]。Che等是我们参加评测时的结果。

图2-7比较了不同方法在不同句子长度上的效率。清晰起见，我们略去了长度大于110的句子。分析时间包括从输入句子到输出分析结果中的所有操作，如特征提取、解码等。可以看到，二阶段方法大幅度提高了分析长句子的效率，句子长度对分析时间的影响非常小。

表2-6比较了各种方法在不同句长上的性能。由于语料的平均句长为27 。我们使用了三个区间：小于28的句子；大于50的句子，介于中间的句子。可以看到，二阶段方法相比一阶段方法，随着句子长度增大，句法准确率提高的幅度变大。对于长度大于27的句子，二阶段方法的准确率提高较大。

2.6 本章小结

本文提出一种利用标点符号的快速高阶依存句法分析方法。基于图的高阶依存分析方法可以融入比较丰富的特征，因此可以提高依存句法分析的准确率。但是，基于动态规划的解码算法的时间复杂度很高$O(\mathcal{L}n^4)$，因此制约了高阶模型在上层语言处理任务中的应用。为此，我们提出一种基于柱搜索的近似解码算法，在有效融入高阶特征的同时，拥有更有优势的时间复杂度$O(n^3K\log K)$，其中K为柱宽度，一般取10。我们的系统参加了CoNLL 2009年多语种依存句法分析和语义角色标注联合任务，在21参赛单位中，我们在联合任务上名列第1名，在依存句法分析子任务上取得第3名的成绩。

进而，我们针对汉语的特点，提出一种利用标点符号进行长句切分的二阶段依存句法分析方法，进一步提高依存句法分析模型处理长句时的效率。实验证明，这种方法可以显著提高依存句法分析的速度，并且可以提高在长句上的分析准确率。

第 3 章 汉语词性标注和依存分析联合模型

3.1 引言

依存句法分析模型中，词性是非常重要且有效的特征。如果只使用词语特征，会导致严重的数据稀疏问题。自然语言处理中，词性标注和依存句法分析这两个问题通常被当成两个独立的任务，以级联的方式实现。即对于一个输入句子，假定其分词结果已知，先对句子进行词性标注，然后在词性标注结果的基础上进行依存句法分析。这种级联的方法会导致错误蔓延。也就是说，词性标注的错误会严重的影响依存分析的准确率。这种错误蔓延的问题对于汉语尤其严重。由于汉语缺乏词形变化信息（如英语中的词后缀变化如-ing，-ed，-es，-ly等），因此汉语的词性标注比其他语言如英语更具挑战性。目前最好的词性标注系统在汉语上的性能仅能达到大约94%，远低于英语的97%[26]。我们的实验结果表明，当使用自动词性标注结果时，汉语的依存分析准确率比使用正确词性标注时下降约6%（见表3-4和表3-5）。

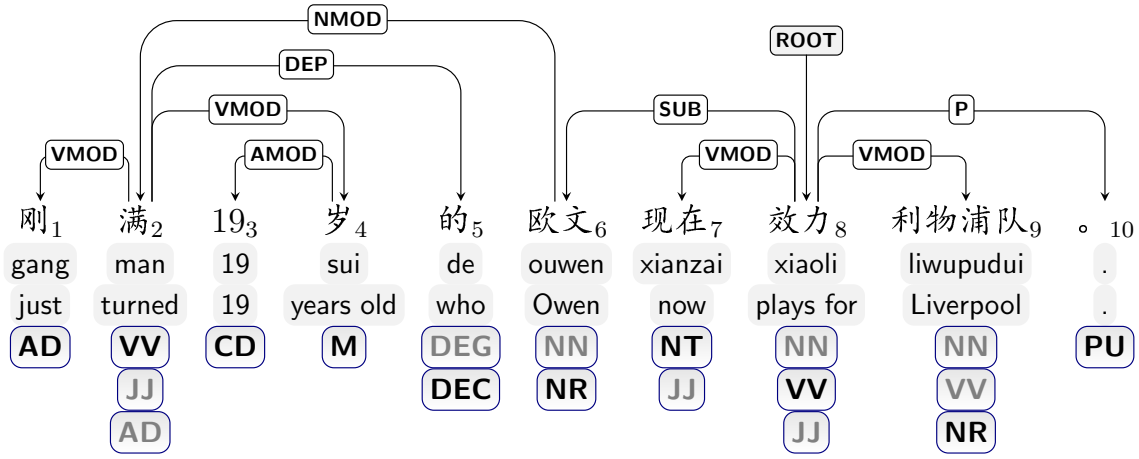


图 3-1 CTB5中的一个依存树的例子。

Fig. 3-1 A labeled dependency tree from CTB5.

图3-1给出了Penn Chinese Treebank 5.1 (CTB5)[76] 中的一个依存树的例子。最下面三行给出了根据我们提出的词性剪枝方法得出的的每个词语的词性候选。这些候选是根据基于条件随机场（Conditional random field, CRF）的词性标注模型给出的边缘概率得到的。详细介绍参见第3.4.3节。对于每个词，黑色标出的词性为人工标注的正确词性，灰色的词性为错误的候选。

可以看出，基于CRF的词性标注器的1-best结果中，存在四个错误，分别是：“的/DEC→DEG”, “欧文/NR→NN”, “效力/VV→NN”, 和“利物浦队/NR→NN”。事实上，类似{NN, VV}和{DEC, DEG}的词性歧义与句法结构紧密相关。一方面，这些词性歧义如果不能正确消解，那么会对后续的句法分析工作制造很大的麻烦。反过来，消解这些词义或多或少需要局部甚至整个句子的句法结构。我们称这样的词性歧义为**句法敏感的词性歧义（syntax-sensitive ambiguities）**。反之，类似{NN, NR}的词性歧义与句法结构联系很少。NN和NR都是表示名词，区别在于NR表示更具体的专有名词如人名、地名等，而NN表示一般的普通名词。这两种名词在句法结构中经常扮演很类似的作用，如它们都可以作为主语和宾语，也都可以作为定语修饰其他的名词。我们称类似{NN, NR}的词性歧义为**句法不敏感的词性歧义（syntax-insensitive ambiguities）**。句法敏感的词性歧义对于序列标注模型而言非常困难，因为消解这些歧义通常需要全局的句法信息。图3-1的例子中，基于CRF的词性标注模型将“效力”错误识别为名词NN，严重影响上层的依存分析模型。由于很少有名词作为句子核心词（谓语），依存分析基准模型发现整个句子缺少合适的谓语，因此输出了一个很不合理的依存树。

在我们的工作之前，研究者们并没有充分重视汉语依存句法分析面临的词性错误蔓延问题。近年来研究者面对汉语依存句法分析时，一般会忽略这个问题，简单采用正确的词性标注[34, 36, 60]。

本文工作中，我们尝试通过建立词性标注和依存句法分析的联合模型来解决这个问题。直觉上来讲，联合学习和处理词性标注和依存句法分析应该可以同时帮助这两个紧密联系的子任务。一方面，联合模型中，句法信息可以用来指导词性标注，从而帮助解决一部分需要句法结构才能够消解的歧义，即句法敏感的词性歧义。另一方面，更准确的词性标注，也可以反过来帮助依存分析。以图3-1中的词性标注错误“效力/VV→NN”为例。由于“效力”是一个未登录词，并且上下文特征也诉法提供有用的线索，因此基于CRF的序列标注方式的词性标注模型很难正确识别其为动词VV，而非名词NN。在联合模型的框架下，情况则有所不同。联合模型可能发现，如果将“效力”识别为名词的话，整个句子就没有了谓语，进而会得到一个分值很低的句法树；反之，如果将“效力”识别为动词的话，模型会得到一个分值较高的句法树。通过这种对比，联合模型很有可能正确标注这个未登录词的词性。

对于词性标注和依存句法分析联合模型而言，最大的挑战是如何设计有效的解码算法，一方面可以融合丰富的词性和句法特征，另一方面可以在一个巨大的搜索空间中，快速寻找到最优的联合解。本文对依存句法分析解码算法进

行扩展，面向词性标注和依存句法分析联合模型，提出了两种基于动态规划（dynamic programming, DP）的解码算法。这两种解码算法可以融合不同复杂度的句法特征。进而，我们又提出一种基于边缘概率的词性剪枝方法，有效地约束每个词语的词性候选集合，从而加速联合模型的解码速度。CTB5数据集上的实验结果表明，联合模型可以同时帮助词性标注和依存句法分析。与基准的级联式的方法相比，联合模型使得词性准确率提高0.4%（6.7%的错误下降率），依存句法准确率提高约0.7%（3.5%的错误下降率）。

3.2 相关工作

联合模型是一种同时解决多个相关问题的有效方法。近年来，研究者提出了不少有效的联合模型。比如：

- 汉语分词和词性标注联合模型[77–79]
- 词干提取（lemmatization）和词性标注联合模型[80]
- 词法分析（morphological analysis）和句法分析联合模型[81, 82]
- 命名实体识别和句法分析联合模型[83],
- 句法分析和语义角色标注联合模型[84]
- 词义消歧和语义角色标注联合模型[85]
- 词法分析和机器翻译联合模型[86, 87]
- 句法分析和机器翻译联合模型[88]

值得注意的是，上面所列的“句法分析”全部指的是“短语结构句法分析”，又称为“组块句法分析”。据我们所知，在我们的工作之前，几乎没有学者提出有效的联合模型，用来成功的解决依存句法分析和其他相关任务。国际计算自然语言学习会议（Conference on Computational Natural Language Learning，CoNLL）于2008年、2009年连续举办了两届依存句法分析和语义角色标注联合评测任务（shared task）。参加联合评测任务的一些学者，尝试使用依存句法分析和语义角色标注联合模型提高两个任务的性能。然而评测中，成绩最好的几个系统都采用级联的方法[8, 11]。

Eisner (2000)从理论角度简要的讨论了如何扩展其提出的句法分析解码算法，从而处理一词多义（polysemy）的情况[24]。在此，词义可以理解为词的词性，词义的歧义可以理解为词性的歧义。这样，基于Eisner (2010)的讨论，我们可以对依存分析的解码算法进行扩展，在句法分析的同时，确定出每个词的词性，即所谓的联合模型的解码过程。McDonald (2006)在其博士论文中，

基于Eisner的思想，扩展了他的二阶依存分析模型[89]，从而实现了词性标注和依存句法分析联合模型[25]。为了提高训练和解码的速度，McDonald根据一个基于最大熵的词性标注器给出的概率，对每个词只保留最可能的2个词性。他在英语语料English Penn Treebank（PTB）上做了实验，发现联合模型可以让句法准确率从91.5%提高到91.9%。然而，由于没有更有效的词性裁剪策略，McDonald的联合模型速度非常慢。本文基于Eisner的思想，扩展了当前最好的基于图的依存句法分析模型，并将其应用于汉语树库，做了广泛而深入的实验和分析。我们进而提出了一种简单有效的裁剪策略，用以约束词性搜索空间。

Smith和Eisner (2008)将置信度传播算法（loopy belief propagation, LBP）应用于依存句法分析上，并且指出LBP可以很自然的以潜在变量（latent variables）表示词性，因而可以在解码过程中联合消解词性和句法结构[29]。Lee等(2011)扩展了Smith和Eisner的工作，利用LBP研究词语形态分析和依存句法分析的联合模型[90]。他们的工作针对形态变化很丰富的语言，包括拉丁语（Latin）、捷克语（Czech）、古希腊语（Ancient Greek）以及匈牙利语（Hungarian）。对于这些语言，词语的形态分析需要消解词语的词性、性别、格等方面。他们的实验结果表明，联合模型可以很好的刻画形态变化和句法之间的联系，并且可以同时提高两个子任务的准确率。Rush等(2010)提出使用对偶分解（Dual Decomposition, DD）对多个相关的自然语言处理任务联合求解[91]。他们做了两组实验。一个是将短语结构句法分析模型和依存句法分析模型联合求解。另一个将短语结构句法分析模型和词性标注模型联合起来。两组实验都发现联合求解可以同时帮助两个子任务。从我的理解看，DD只能在测试阶段对多个模型联合求解，而无法应用于多个模型的联合训练。Auli和Lopez (2011)针对组合范畴文法（Combinatorial Categorial Grammar，CCG）中的supertagging和句法分析联合求解问题，全面比较了LBP和DD[92]。他们实验发现，和级联模型相比，联合模型能够提高句法的F值；并且，LBP和DD下联合模型的句法F值很接近，但是它们的收敛速度差距比较大。他们的工作局限于在测试阶段将两个独立训练的模型进行联合求解。并且，他们的初步试验发现利用LBP训练联合模型，会导致句法F值较大幅度下降。

Hatori等(2011)在基于转移的依存句法分析模型的框架下研究汉语词性标注和依存句法分析联合模型[13]。他们发现联合模型可以提高句法准确率，但只对词性标注准确率的提高很小。本文与他们的结果进行了对比，发现我们提出的基于图的联合模型的句法准确率和他们的结果接近，并且我们的词性准确率更高。

3.3 级联方法

级联方法将词性标注和依存句法分析看成两个独立的有先后顺序任务。首先，对于输入句子$\mathbf{x} = w_1...w_n$，确定一个最优的词性序列$\hat{\mathbf{t}}$。

$$\hat{\mathbf{t}} = \arg\max_{\mathbf{t}} \mathrm{Score}_{\mathrm{pos}}(\mathbf{x}, \mathbf{t}) \tag{3-1}$$

形式化地，一个词性序列表示为$\mathbf{t} = t_1...t_n$，其中$t_i \in \mathcal{T}$ $(1 \le i \le n)$，$\mathcal{T}$ 表示一个人工定义的词性标记集合。笼统来讲，计算语言学中引入词性标记是为了表示词类，相同词性的词在句法结构中通常扮演类似的角色。$\mathcal{T}$的大小，即词性标记的总数，一般远远小于词汇的数目。因此，对于句法分析而言，词性是对词汇的抽象化，可以用来克服词汇化特征的数据稀疏问题。

然后，基于$\mathbf{x}$ and $\hat{\mathbf{t}}$，确定一个最优的依存树$\hat{\mathbf{d}}$。

$$\hat{\mathbf{d}} = \arg\max_{\mathbf{d}} \mathrm{Score}_{\mathrm{syn}}(\mathbf{x}, \hat{\mathbf{t}}, \mathbf{d}) \tag{3-2}$$

形式化地，一个依存句法树表示为$\mathbf{d} = \{(h, m, l) : 0 \le h \le n, 1 \le m \le n, l \in \mathcal{L}\}$，其中$(h, m, l)$表示一个从核心词（head, father）$w_h$到修饰词（modifier，dependent, child）w_m的依存弧，l表示依存弧的关系类型，$\mathcal{L}$是依存弧关系类型的集合。依存弧的关系类型用来表示两个词之间的句法或语义关系。比如，图3-1中的依存弧(8, 6, SUB)表示“欧文”是“效力”的主语（SUB）。为了方便问题形式化和算法描述，我们引入一个伪节点w_0。w_0总是指向整个句子的核心词。由于本章内容不考虑依存弧的关系类型，因此我们表示依存弧时省略关系类型，用(h, m)或$w_h \curvearrowright w_m$表示一条依存弧。

3.3.1 基于条件随机域的词性标注模型

词性标注是一个典型的序列序列标注问题。研究者们成功的将很多序列标注模型应用于词性标注，如最大熵模型（Maximum Entropy, ME）[93]、条件随机域（Conditional Random Fields, CRF）[94]、和感知器（perceptron）[26]。本文采用CRF实现我们的基准词性标注模型，有两个方面原因。第一，我们以前的实验表明基于CRF的词性标注模型比基于ME和perceptron的模型准确率略高。第二，基于CRF的词性标注模型可以提供词性的边缘概率。本文使用其提供的边缘概率对词性候选进行裁剪（见第3.4.3节）。

CRF属于条件对数线性（log-linear）概率模型。给定输入句子$\mathbf{x}$，一个词性序列$\mathbf{t}$的条件概率为：

$$P(\mathbf{t}|\mathbf{x}) = \frac{\exp(\text{Score}_{\text{pos}}(\mathbf{x}, \mathbf{t}))}{\sum_{\mathbf{t}'} \exp(\text{Score}_{\text{pos}}(\mathbf{x}, \mathbf{t}'))} \tag{3-3}$$

$$\begin{aligned} \text{Score}_{\text{pos}}(\mathbf{x}, \mathbf{t}) &= \mathbf{w}_{\text{pos}} \cdot \mathbf{f}_{\text{pos}}(\mathbf{x}, \mathbf{t}) \\ &= \sum_{1 \le i \le n} \text{Score}_{\text{pu}}(\mathbf{x}, i, t_i) + \text{Score}_{\text{pb}}(\mathbf{x}, i, t_{i-1}, t_i) \end{aligned} \tag{3-4}$$

$$\begin{aligned} \text{Score}_{\text{pu}}(\mathbf{x}, i, t_i) &= \mathbf{w}_{\text{pu}} \cdot \mathbf{f}_{\text{pu}}(\mathbf{x}, i, t_i) \\ \text{Score}_{\text{pb}}(\mathbf{x}, i, t_{i-1}, t_i) &= \mathbf{w}_{\text{pb}} \cdot \mathbf{f}_{\text{pb}}(\mathbf{x}, i, t_{i-1}, t_i) \end{aligned} \tag{3-5}$$

其中$\mathbf{f}_{\text{pos}}(\mathbf{x}, \mathbf{t})$表示$(\mathbf{x}, \mathbf{t})$对应的聚合的词性特征向量，$\mathbf{w}_{\text{pos}}$表示词性特征的权重向量。我们使用两类词性特征，分别为：一元词性特征（POS unigram features），记为$\mathbf{f}_{\text{pu}}(\mathbf{x}, i, t_i)$；二元词性特征（*POS bigram features*），记为$\mathbf{f}_{\text{pb}}(\mathbf{x}, i, t_{i-1}, t_i)$。

表 3-1 词性特征集合，用于基准词性标注模型和联合模型。

Table. 3-1 POS tagging features used in both tagging and joint models.

POS Unigram Features: $\mathbf{f}_{\text{pu}}(\mathbf{x}, i, t_i)$
01: $t_i \circ w_i$
02: $t_i \circ w_{i-1}$
03: $t_i \circ w_{i-2}$
04: $t_i \circ w_i \circ c_{i-1,-1}$
05: $t_i \circ w_i \circ c_{i+1,0}$
06: $t_i \circ c_{i,0}$
07: $t_i \circ c_{i,-1}$
08: $t_i \circ c_{i,k}, 0 < k < \#c_i - 1$
09: $t_i \circ c_{i,0} \circ c_{i,k}, 0 < k < \#c_i - 1$
10: $t_i \circ c_{i,-1} \circ c_{i,k}, 0 < k < \#c_i - 1$
11: **if** $\#c_i = 1$ **then** $t_i \circ w_i \circ c_{i-1,-1} \circ c_{i+1,0}$
12: **if** $c_{i,k} = c_{i,k+1}$ **then** $t_i \circ c_{i,k} \circ$ *"consecutive"*
13: $t_i \circ \text{prefix}(w_i, k), 1 \le k \le 4, k \le \#c_i$
14: $t_i \circ \text{suffix}(w_i, k), 1 \le k \le 4, k \le \#c_i$
POS Bigram Features: $\mathbf{f}_{\text{pb}}(\mathbf{x}, i, t_{i-1}, t_i)$
15: $t_i \circ t_{i-1}$

表3-1列举了所有的词性特征模版。其中，$\circ$表示两个字符串的拼接；$c_{i,k}$表示w_i的第k个字符（一个汉字作为一个字符；从0开始计数）；$c_{i,0}$表示w_i的第0个

字符；$c_{i,-1}$表示w_i的倒数第0个（即最后一个）字符；$\#c_i$表示w_i中包含的字符数。$\text{prefix}(w_i, k)$表示w_i的前k个字符构成的前缀；$\text{suffix}(w_i, k)$表示w_i的后k个字符构成的后缀。这个特征集合主要借鉴了Zhang和Clark (2008)的工作[77]。他们发现大量使用基于字符的特征对词性标注很有帮助。第13和14项的前后缀特征借鉴于Ranaparkhi (1996)的工作[93]。Ranaparkhi (1996)使用前后缀特征帮助提高英语中未登录词的标注准确率。[1] 我在联合模型上的实验表明，这两个前后缀特征可以提高词性准确率，并且提高句法准确率。

我们使用Collins等(2008)提出的*exponentiated gradient (EG)*算法[95]来学习特征权重$\mathbf{w}_{\text{pos}}$[2]。解码时，针对输入句子，我们采用经典的Viterbi算法搜索最优的词性序列。

3.3.2 基于图的依存句法分析模型

近年来，研究者们对基于图的依存句法分析方法越来越重视。一方面是因为这种方法从图的角度对依存句法分析进行建模，因此可以应用一些成熟的图搜索算法。另一方面，基于图的依存句法分析方法在多种语言、多个数据集上都取得了最好的准确率。基于图的依存句法分析模型将句子看成一个完全有向图，然后尝试从图中找出一个分值最高的依存树。为了保证基于动态规划的全局搜索算法的效率，基于图的模型需要做很强的独立假设。假设句子对应的依存树中，只有存在于某些特殊结构（子树，subtree）中的依存弧之间才互相联系和影响，其他依存弧之间则互相独立。在这种假设下，一个依存树的分值便分解为一些子树的分值的和。

$$\begin{aligned}\text{Score}_{\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) &= \mathbf{w}_{\text{syn}} \cdot \mathbf{f}_{\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \\ &= \sum_{p \subseteq \mathbf{d}} \text{Score}_{\text{subtree}}(\mathbf{x}, \mathbf{t}, p)\end{aligned} \tag{3-6}$$

其中，p表示一个独立假设允许的子树。p包含一个或多个$\mathbf{d}$中的依存弧。$\text{Score}_{\text{subtree}}(\mathbf{x}, \mathbf{t}, p)$表示由$p$贡献的分值。$\mathbf{f}_{\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d})$是$(\mathbf{x}, \mathbf{t}, \mathbf{d})$对应的聚合的句法特征向量，$\mathbf{w}_{\text{syn}}$是句法特征的权重向量。

[1]在此感谢张梅山同学建议我使用这两个特征。

[2]Carreras和Koo实现并共享了EG算法的代码，他们将程序包命名为egstra。见http://groups.csail.mit.edu/nlp/egstra/。程序包中还实现了基于CRF的一阶依存句法分析模型。我们对程序包进行了简单扩展，实现了基于CRF的词性标注模型、基于边缘概率的词性裁剪、和基于边缘概率的依存弧裁剪。Koo和Collins最先提出基于边缘概率对依存弧进行裁剪，以提高高阶依存分析模型的效率[22]。在此，我们对他们表示感谢！

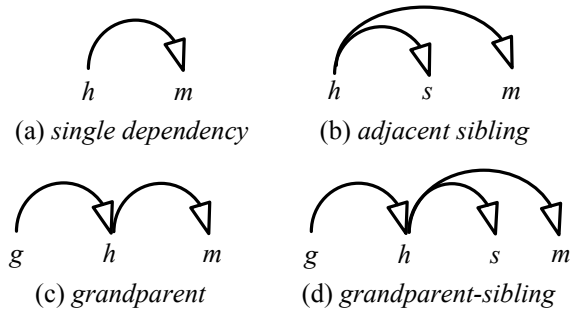


图 3-2 基于图模型中使用的几种子树。

Fig.3-2 Different types of scoring subtrees used in current graph-based models[22].

过去几年来，研究者们不断弱化基于图的依存分析模型中的独立假设，允许模型使用更复杂的子树，从而融入更丰富更全局的特征。虽然解码算法的复杂度越来越高，但是模型的准确率也有较大提高[19–22]。图3-2中给出了目前基于图的模型中使用到的四种子树结构。如果模型使用所有四种子树，那么一个依存树的分值可以细化为：

$$\begin{aligned}\text{Score}_{\text{syn}}(\mathbf{x},\mathbf{t},\mathbf{d}) = &\sum_{\{(h,m)\}\subseteq\mathbf{d}} \text{Score}_{\text{dep}}(\mathbf{x},\mathbf{t},h,m) \\ &+ \sum_{\{(h,s),(h,m)\}\subseteq\mathbf{d}} \text{Score}_{\text{sib}}(\mathbf{x},\mathbf{t},h,s,m) \\ &+ \sum_{\{(g,h),(h,m)\}\subseteq\mathbf{d}} \text{Score}_{\text{grd}}(\mathbf{x},\mathbf{t},g,h,m) \\ &+ \sum_{\{(g,h),(h,s),(h,m)\}\subseteq\mathbf{d}} \text{Score}_{\text{gsib}}(\mathbf{x},\mathbf{t},g,h,s,m)\end{aligned} \tag{3-7}$$

其中$\text{Score}_{\text{dep/sib/grd/gsib}}(.)$表示图3-2中四种子树对应的分值。由于我们采用线性模型（linear model），这些分值可以表示为：

$$\begin{aligned}\text{Score}_{\text{dep}}(\mathbf{x},\mathbf{t},h,m) &= \mathbf{w}_{\text{dep}} \cdot \mathbf{f}_{\text{dep}}(\mathbf{x},\mathbf{t},h,m) \\ \text{Score}_{\text{sib}}(\mathbf{x},\mathbf{t},h,s,m) &= \mathbf{w}_{\text{sib}} \cdot \mathbf{f}_{\text{sib}}(\mathbf{x},\mathbf{t},h,s,m) \\ \text{Score}_{\text{grd}}(\mathbf{x},\mathbf{t},g,h,m) &= \mathbf{w}_{\text{grd}} \cdot \mathbf{f}_{\text{grd}}(\mathbf{x},\mathbf{t},g,h,m) \\ \text{Score}_{\text{gsib}}(\mathbf{x},\mathbf{t},g,h,s,m) &= \mathbf{w}_{\text{gsib}} \cdot \mathbf{f}_{\text{gsib}}(\mathbf{x},\mathbf{t},g,h,s,m)\end{aligned} \tag{3-8}$$

其中$\mathbf{f}_{\text{dep/sib/grd/gsib}}(.)$表示图3-2中四种子树对应的特征向量，而$\mathbf{w}_{\text{dep/sib/grd/gsib}}$表示对应的特征权重向量。

3.3.2.1 基于图的依存句法分析模型采用的特征集合

对于句法特征，我们充分借鉴前人的特征集合[21, 22, 25]。值得一提的是，Bohnet提出了整理并总结了一套丰富有效的句法特征集合[72]。表3-2列出了所

表 3-2 句法特征集合，用于基准依存句法分析模型和联合模型。

Table. 3-2 Syntactic features used in both parsing and joint models.

Dependency Features: $\mathbf{f}_{\text{dep}}(\mathbf{x}, \mathbf{t}, h, m)$		
01: $w_h \circ t_h \circ \mathsf{r} \circ \mathsf{d}$	02: $w_h \circ \mathsf{r} \circ \mathsf{d}$	03: $t_h \circ \mathsf{r} \circ \mathsf{d}$
04: $w_m \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	05: $w_m \circ \mathsf{r} \circ \mathsf{d}$	06: $t_m \circ \mathsf{r} \circ \mathsf{d}$
07: $w_h \circ t_h \circ w_m \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	08: $t_h \circ w_m \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	09: $w_h \circ w_m \circ t_m \circ \mathsf{r} \circ \mathsf{d}$
10: $w_h \circ t_h \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	11: $w_h \circ t_h \circ w_m \circ \mathsf{r} \circ \mathsf{d}$	12: $w_h \circ w_m \circ \mathsf{r} \circ \mathsf{d}$
13: $t_h \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	14: $w_h \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	15: $w_h \circ t_m \circ \mathsf{r} \circ \mathsf{d}$
16: $t_h \circ t_{h+1} \circ t_{m-1} \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	17: $t_h \circ t_{h+1} \circ t_m \circ t_{m+1} \circ \mathsf{r} \circ \mathsf{d}$	18: $t_{h-1} \circ t_h \circ t_{m-1} \circ t_m \circ \mathsf{r} \circ \mathsf{d}$
19: $t_{h-1} \circ t_{h+1} \circ t_{m-1} \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	20: $t_{h-1} \circ t_h \circ t_{h+1} \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	21: $t_h \circ t_{m-1} \circ t_m \circ t_{m+1} \circ \mathsf{r} \circ \mathsf{d}$
22: $t_h \circ t_{h+1} \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	23: $t_h \circ t_{m-1} \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	24: $t_h \circ t_m \circ t_{m+1} \circ \mathsf{r} \circ \mathsf{d}$
25: $t_h \circ t_b \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	26: $t_h \circ \#\mathsf{verb}(h, m) \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	27: $t_h \circ \#\mathsf{conj}(h, m) \circ t_m \circ \mathsf{r} \circ \mathsf{d}$
28: $t_h \circ \#\mathsf{punc}(h, m) \circ t_m \circ \mathsf{r} \circ \mathsf{d}$		
Sibling Features: $\mathbf{f}_{\text{sib}}(\mathbf{x}, \mathbf{t}, h, s, m)$		
29: $t_h \circ t_s \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	30: $w_h \circ t_s \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	31: $t_h \circ w_s \circ t_m \circ \mathsf{r} \circ \mathsf{d}$
32: $t_h \circ t_s \circ w_m \circ \mathsf{r} \circ \mathsf{d}$	33: $t_s \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	34: $w_s \circ w_m \circ \mathsf{r} \circ \mathsf{d}$
35: $t_s \circ w_m \circ \mathsf{r} \circ \mathsf{d}$	36: $w_s \circ t_m \circ \mathsf{r} \circ \mathsf{d}$	37: $t_s \circ t_{s+1} \circ t_m \circ \mathsf{r}$
38: $t_{s-1} \circ t_s \circ t_m \circ \mathsf{r}$	39: $t_s \circ t_{m-1} \circ t_m \circ \mathsf{r}$	40: $t_s \circ t_m \circ t_{m+1} \circ \mathsf{r}$
41: $t_s \circ t_{s+1} \circ t_{m-1} \circ t_m \circ \mathsf{r}$	42: $t_{s-1} \circ t_s \circ t_{m-1} \circ t_m \circ \mathsf{r}$	43: $t_s \circ t_{s+1} \circ t_m \circ t_{m+1} \circ \mathsf{r}$
44: $t_{s-1} \circ t_s \circ t_m \circ t_{m+1} \circ \mathsf{r}$		
Grandparent Features: $\mathbf{f}_{\text{grd}}(\mathbf{x}, \mathbf{t}, g, h, m)$		
45: $t_g \circ t_h \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$	46: $w_g \circ t_h \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$	47: $t_g \circ w_h \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$
48: $t_g \circ t_h \circ w_m \circ \mathsf{r} \circ \mathsf{r}'$	49: $t_g \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$	50: $w_g \circ w_m \circ \mathsf{r} \circ \mathsf{r}'$
51: $w_g \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$	52: $t_g \circ w_m \circ \mathsf{r} \circ \mathsf{r}'$	53: $t_g \circ t_{m-1} \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$
54: $t_g \circ t_m \circ t_{m+1} \circ \mathsf{r} \circ \mathsf{r}'$	55: $t_{g-1} \circ t_g \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$	56: $t_g \circ t_{g+1} \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$
57: $t_{g-1} \circ t_g \circ t_{m-1} \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$	58: $t_{g-1} \circ t_g \circ t_m \circ t_{m+1} \circ \mathsf{r} \circ \mathsf{r}'$	59: $t_g \circ t_{g+1} \circ t_{m-1} \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$
60: $t_g \circ t_{g+1} \circ t_m \circ t_{m+1} \circ \mathsf{r} \circ \mathsf{r}'$		
Grandparent-sibling Features: $\mathbf{f}_{\text{gsib}}(\mathbf{x}, \mathbf{t}, g, h, s, m)$		
61: $t_g \circ t_h \circ t_s \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$	62: $w_g \circ t_h \circ t_s \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$	63: $t_g \circ w_h \circ t_s \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$
64: $t_g \circ t_h \circ w_s \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$	65: $t_g \circ t_h \circ t_s \circ w_m \circ \mathsf{r} \circ \mathsf{r}'$	66: $w_g \circ w_h \circ t_s \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$
67: $t_g \circ t_s \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$	68: $w_g \circ t_s \circ t_m \circ \mathsf{r} \circ \mathsf{r}'$	

有的句法特征模版，包含四个类别，分别对应四种子树。其中，r表示依存弧$h \curvearrowright m$的方向，例如左弧为“L”，右弧为“R”；d表示依存弧$h \curvearrowright m$中两个词的距离（我们将$|h-m|$离散化到$\{1, 2, 3, [4,6], [7,\infty)\}$）；r′表示依存弧$g \curvearrowright h$的方向；$b$表示介于$h$和$m$之间的一个下标；#verb$(h,m)$表示介于$h$和$m$之间动词的数目；#conj$(h,m)$表示介于$h$和$m$之间连词的数目；#punc$(h,m)$表示介于$h$和$m$之间标点的数目。另外，我们将每个词最后一个字符作为词干，对表3-2中每一个和词相关的特征，用词干替换词构成一个新的特征。Che等发现这样可以提高依存句法分析的准确率[5]。

为了深入的研究联合模型对词性标注和依存句法分析的影响，我们实现了三个不同复杂度的依存句法分析基准模型。

3.3.2.2 一阶依存句法分析模型（O1）

Eisner (1996) 提出一种通用的面向双词汇化语法（bilexical grammar）的动态规划解码算法，时间复杂度为$O(n^3)$[96]。这个算法通常被称为*Eisner 算法*。基于这个算法，McDonald等提出一阶（first-order）依存句法分析模型。一阶模型假设依存弧之间相互独立，依存树的分值只由单个依存弧（single dependency）的分值构成。也就是说，公式3-7中只保留$\text{Score}_{\text{dep}}(.)$项，其他高阶子树对应的分值都略去。

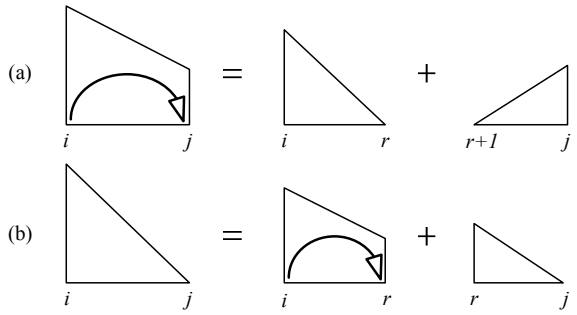


图 3-3 一阶依存分析模型的解码算法中使用到的基本动态规划数据结构和操作。

Fig. 3-3 The DP structures and derivations of Eisner algorithm for the first-order parsing model.

我们使用图3-3和算法3-1详细解释解释Eisner算法。Eisner算法中，*span*是最基本的数据结构。算法以迭代的方式、自底向上的、按照一定的规则、不断将两个小的相邻的span合并为一个更大的span，如图3-3所示。图3-3-(a)最左侧的span，记为$I_{i\curvearrowright j}$，表示一个覆盖$w_i...w_j$的局部依存结构，并且w_i和w_j构成一个依存弧$i \curvearrowright j$。我们称$I_{i\curvearrowright j}$为一个*不完整（incomplete）span*，意思是，在将来的操作中，w_j还有可能获得新的右侧的儿子，即w_j右侧的修饰词集合是*不完整*的。注意，在$I_{i\curvearrowright j}$中，w_j已经找到了其所有的左儿子。图3-3-(b)左侧的span，记

1. $\forall 0 \le i \le n, t_i \in \mathcal{T}$: $C_{i\rightarrow i} = 0, C_{i\leftarrow i} = 0$ // *initialization*
2. **for** $w = 1$ **to** n **do** // *span width*
3. **for** $i = 0$ **to** $(n - w)$ **do** // *span start index*
4. $j = i + w$ // *span end index*
5. $I_{i\curvearrowright j} = \max_{i\le r<j}\{C_{i\rightarrow r} + C_{r+1\leftarrow j} + \text{Score}_{\text{dep}}(\mathbf{x}, \mathbf{t}, i, j)\}$ // *corresponding to Fig. 3-3-(a).*
6. $I_{i\curvearrowleft j} = \max_{i\le r<j}\{C_{i\rightarrow r} + C_{r+1\leftarrow j} + \text{Score}_{\text{dep}}(\mathbf{x}, \mathbf{t}, j, i)\}$
7. $C_{i\rightarrow j} = \max_{i<r\le j}\{I_{i\curvearrowright r} + C_{r\rightarrow j}\}$ // *corresponding to Fig. 3-3-(b).*
8. $C_{i\leftarrow j} = \max_{i\le r<j}\{C_{i\leftarrow r} + I_{r\curvearrowleft j}\}$
9. **end for**
10. **end for**
11. **return** $C_{0\rightarrow n}$

算法 3-1 Eisner算法：一阶依存句法分析模型的解码算法（O1）

Algo. 3-1 Eisner algorithm for the first-order dependency parsing

为$C_{i\rightarrow j}$，表示一个覆盖$w_i...w_j$的局部依存结构，并且w_j是w_i的一个子孙。我们称$C_{i\rightarrow j}$是一个完整（*complete*）*span*，意思是，w_j找到了其所有的左儿子，并且w_j没有右儿子。因此在将来的操作中，w_j不能再增加新的修饰词，即w_j的修饰词集合是完整的。在$I_{i\curvearrowright j}$和$C_{i\rightarrow j}$中，w_i又称为span的核心词（span head）。类似的，$I_{i\curvearrowleft j}$和$C_{i\leftarrow j}$表示以w_j为核心的相反方向的两种span。简洁起见，图3-3中省略了产生这两种span的操作。

算法3-1的第5行尝试确定最优的不完整span $I_{i\curvearrowright j}$。给定r，通过合并$C_{i\rightarrow r}$和$C_{r+1\leftarrow j}$产生一个候选的$I_{i\curvearrowright j}$，如图3-3-(a)所示。这个合并操作引入一个新的依存弧$i \curvearrowright j$，因而产生的span $I_{i\curvearrowright j}$的分值包含三个部分：$C_{i\rightarrow r}$和$C_{r+1\leftarrow j}$的分值，加上新引入的依存弧贡献的分值$\text{Score}_{\text{dep}}(\mathbf{x}, \mathbf{t}, i, j)$。算法遍历所有可能的分割点$r$，从而找到分值最高的$I_{i\curvearrowright j}$。

算法3-1第7行尝试确定最优的完整span $C_{i\rightarrow j}$。给定r，通过合并$I_{i\curvearrowright r}$和$C_{r\rightarrow j}$可以产生一个候选的$C_{i\rightarrow j}$，如图3-3-(b)所示。这个合并操作没有引入新的依存弧，因此产生的span $C_{i\rightarrow j}$的分值等于两个组件span $I_{i\curvearrowright r}$和$C_{r\rightarrow j}$的分值之和。算法遍历所有可能的分割点r，从而找到分值最高的$C_{i\rightarrow j}$。

类似的，算法第6行和第8行分别确定以w_j为核心的非完整和完整span。简洁起见，我们不做详细解释。最终，span $C_{0,n}$保存了最优的完整依存树的分值，通过回溯（backtracking）的方法，可以获得这个最优的依存树。算法3-1的正确

性包含两个方面：1）算法按照公式3-7正确计算每一个span的分值；2）算法最终可以找到最优的依存树。Eisner (2000)讨论了算法的正确性证明[24]。

算法需要存储$2n^2$个完整span和$2n^2$个不完整span。因此空间复杂度为$O(n^2)$。第5行至第8行的时间复杂度一共是$O(4n)$，而外层的w和i的循环需要$O(n^2)$。因此，算法的时间复杂度为$O(n^3)$。

3.3.2.3 二阶和三阶依存句法分析模型（O2 & O3）

Koo和Collins (2010)扩展了Eisner算法，提出一种更复杂的动态规划解码算法，允许依存句法分析模型融入更复杂的子树[22]。我们简称这个算法为Koo算法。Koo算法可以融入图3-2中的所有子树。如果模型使用了图3-2中的所有子树，那么称为三阶（third-order）模型。如果模型不使图3-2中最复杂的granparent-sibling子树，那么便退化为二阶（second-order）模型。值得注意的是，二阶模型和三阶模型使用同样的解码算法，区别只在于二阶模型解码过程中不考虑grandparent-sibling子树引入的分值。因为后面会详细介绍三阶联合模型的解码算法，而三阶联合模型的解码算法是对Koo算法的一个很直接的扩展（见图3-3和算法3-3)），因此我们略过对Koo算法的详细介绍。

3.4 联合模型

在联合模型的框架下，我们同时求解词性标注和依存句法分析两个问题。

$$(\hat{\mathbf{t}}, \hat{\mathbf{d}}) = \arg\max_{\mathbf{t},\mathbf{d}} \mathrm{Score}_{\mathrm{joint}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \tag{3-9}$$

一个联合解的分值等于词性序列的分值和依存树分值之和。而词性序列分值和依存树分值的定义和我们在基准模型中的定义一致。

$$\begin{aligned} \mathrm{Score}_{\mathrm{joint}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) &= \mathrm{Score}_{\mathrm{pos}}(\mathbf{x}, \mathbf{t}) + \mathrm{Score}_{\mathrm{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \\ &= \mathbf{w}_{\mathrm{pos}\oplus\mathrm{syn}} \cdot \mathbf{f}_{\mathrm{pos}\oplus\mathrm{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \\ &= \mathbf{w}_{\mathrm{joint}} \cdot \mathbf{f}_{\mathrm{joint}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \end{aligned} \tag{3-10}$$

其中$\mathbf{f}_{\mathrm{joint}}(\mathbf{x}, \mathbf{t}, \mathbf{d})$表示联合解$(\mathbf{x}, \mathbf{t}, \mathbf{d})$对应的聚合的特征向量。这个特征向量相当于$\mathbf{f}_{\mathrm{pos}}(.)$ 和$\mathbf{f}_{\mathrm{syn}}(.)$这两个特征向量的拼接（词性特征和句法特征在特征空间中分别拥有不同的维度范围，比如词性特征的维度为N，对应$[0, N-1]$的空间；而句法特征的维度为M，对应$[N, N+M-1]$的空间），因此又记为$\mathbf{f}_{\mathrm{pos}\oplus\mathrm{syn}}(.)$。特征权重向量为$\mathbf{w}_{\mathrm{joint}}$，又记为$\mathbf{w}_{\mathrm{pos}\oplus\mathrm{syn}}$。和级联方法中的基准词性标注模型和依存句法分析模型不同，联合模型同时学习词性和句法特征的权重，因此这两种特征可以互相交互和影响，模型也可以更好的平衡这两种特征在确定联合解中的作用。

对联合模型而言，最大的挑战是如何设计有效的解码算法，一方面可以融入丰富的特征，另一方面可以在巨大的搜索空间中迅速找到最优的联合解。Eisner (2000)提出可以扩展句法分析解码算法，从而处理一词多义（*polysemy*）的情况[24]。基于这个想法，我们扩展了前人提出的依存句法分析解码算法，即Eisner算法和Koo算法，提出两个基于动态规划的面向联合模型的解码算法。为了和基准依存句法分析模型深入比较，我们对应的实现了三个不同复杂度联合模型。同样的，根据$\mathbf{f}_{\text{syn}}(.)$包含的子树类型，分别为一阶、二阶、和三阶联合模型。

3.4.1 一阶词性标注和依存句法分析联合模型（JO1）

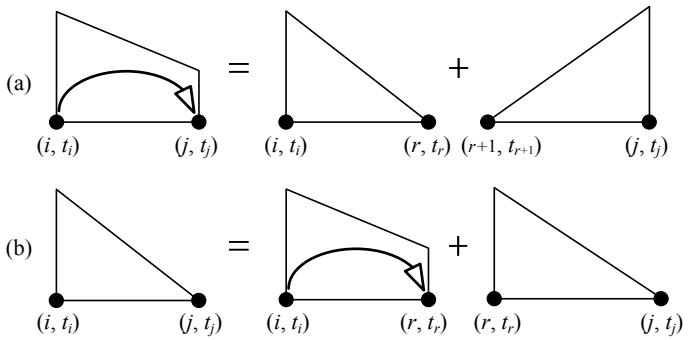


图 3-4 一阶联合模型的解码算法中使用到的基本动态规划数据结构和操作。

Fig. 3-4 The DP structures and derivations of the decoding algorithm of the first-order joint models.

和基准一阶依存句法分析模型相同，一阶联合模型中依存句法树的分值$\text{Score}_{\text{syn}}(.)$只包含由单个依存弧贡献的分值$\text{Score}_{\text{dep}}(.)$。我们基于Eisner (2000)的想法[24]，通过扩展Eisner算法，提出了基于动态规划的一阶联合模型的解码算法。图3-4和算法3-2给出了算法的详细描述。和Eisner算法不同的是，每一个span增加了边界词的词性。$I_{i\curvearrowright j}^{t_i,j}$表示一个不完整的span，并且$w_i$的词性为$t_i$，$w_j$的词性为$t_j$，如图3-4-(a)中左侧的span。$C_{i\rightharpoonup j}^{t_i,j}$表示一个完整span，并且边界词性分别为$t_i$和$t_j$，如图3-4-(b)中左侧的span。

算法3-2同样以自底向上的方式不断迭代，按照一定的规则，每次将两个相邻的span合并为一个大的span。算法第6行尝试找到最优的不完整span $I_{i\curvearrowright j}^{t_i,j}$。给定$r$、$t_r$和$t_{r+1}$，通过合并两个完整span $C_{i\rightharpoonup r}^{t_i,r}$和$C_{r+1\leftharpoondown j}^{t_{r+1},j}$，可以产生一个候选的$I_{i\curvearrowright j}^{t_i,j}$，如图3-4-(a)所示。这个合并操作引入了一个新的依存弧$i \curvearrowright j$。和Eisner算法类似，这条依存弧共享的分值会累加到生成的span中。除此之外，联合解码算法还需要额外累加词性特征引入的分值。最终，生成的候选span

$I_{i\curvearrowright j}^{t_i,t_j}$的分值包含5部分，即：两个组件span的分值；新的依存弧贡献的句法分值$\mathsf{Score}_{\mathrm{dep}}(\mathbf{x}, t_i, t_j, i, j)$；将$w_j$标注为$t_j$引入的词性分值$\mathsf{Score}_{\mathrm{pu}}(\mathbf{x}, j, t_j)$；以及将$w_r$标为$t_r$、$w_{r+1}$标为$t_{r+1}$引入的词性分值$\mathsf{Score}_{\mathrm{pb}}(\mathbf{x}, r+1, t_r, t_{r+1})$。稍后我们会专门讨论为什么这样累加词性特征引入的分值。算法遍历所有可能的r、t_r和t_{r+1}，最终确定分值最高的span $I_{i\curvearrowright j}^{t_i,t_j}$。

值得注意的是，$\mathsf{Score}_{\mathrm{dep}}(\mathbf{x}, t_i, t_j, i, j)$的参数列表中，只提供了$w_i$和$w_j$这两个词性$t_i$和$t_j$。根据表3-2可知，$\mathbf{f}_{\mathrm{dep}}(.)$需要句子中其他词的词性信息，如$t_{i\pm1}$，$t_b$（$i < b < j$），甚至$i$和$j$之间所有动词的数目（$\#\mathrm{verb}(h, m)$）。Span作为算法的基本数据结构，没有也无法存储这些上下文词性甚至是全局的词性信息，因此，我们只能向特征函数提供几个有限的词性。为了解决这个问题，我们借鉴McDonald (2006)在他博士论文中的做法[25]：对于这些动态规划算法无法动态确定的词性，我们一律使用基准CRF词性标注器提供的1-best词性。我们以前的工作显示，这种近似做法对词性和句法准确率的影响不大。

算法3-2的第8行尝试确定最优的完整span $C_{i\rightarrow j}^{t_i,t_j}$。给定$r$和$t_r$，算法通过合并$I_{i\curvearrowright r}^{t_i,t_r}$和$C_{r\rightharpoonup j}^{t_r,t_j}$得到一个候选span，如图3-4-(b)所示。这个操作没有引入新的依存弧。最终，生成的候选span $C_{i\rightarrow j}^{t_i,t_j}$分值等于两个组件span的分值之和。算法遍历所有的r和t_r以确定分值最高的$C_{i\rightarrow j}^{t_i,t_j}$。

第7行和第9行的目标分别是确定以w_j为核心的最优的不完整span $I_{i\curvearrowright j}^{t_i,t_j}$和完整span $C_{i\leftarrow j}^{t_i,t_j}$。最后，我们遍历所有可能的$t_n$，找到分值最高的span $C_{0\rightarrow n}^{t_0,t_n}$，我们将伪节点$w_0$的词性确定为“#”。通过回溯，我们就可以得到最优的联合解。类似于Eisner算法，我们可以证明算法3-2按照公式3-7正确的计算了每一个span的句法分值。

接下来我们说明算法3-2可以根据公式3-4正确的计算词性分值。事实上，我们在算法3-2中遵循了两个原则分别获取一元词性特征和二元词性特征引入的词性分值。

1. 如果合并操作确定了w_k的父亲节点，即引入了一条以w_k为修饰词的依存弧，那么生成的span的分值需要累加上w_k相关的一元词性特征引入的分值。例如，图3-4-(a)中的操作确定了w_i是w_j的父亲，那么生成的span $I_{i\curvearrowright j}^{t_i,t_j}$的分值中需要累加$w_j$的词性为$t_j$引入的词性分值$\mathsf{Score}_{\mathrm{pu}}(\mathbf{x}, j, t_j)$。由于算法3-2为每一个词赋予一个并且唯一一个父亲，因此算法可以正确计算一元词性特征对应的词性分值。
2. 当两个完整span合并为一个不完整span时，假设分割点为r，那么生成的span需要累加w_r标为t_r同时w_{r+1}标为t_{r+1}时产生的二元词性特征对应

的词性分值。例如，图3-4-(b)中的操作中，算法需要累加$\text{Score}_{\text{pb}}(\mathbf{x}, r+1, t_r, t_{r+1})$。要证明算法3-2通过这种方式，可以正确的计算二元词性特征对应的词性分值，只需要证明，按照算法3-2产生任意一个依存树的过程中，对任意r，都会有以r的为分割点的两个完整span $C_{i\rightarrow r}^{t_i,r}$和$C_{r+1\leftarrow j}^{t_{r+1},j}$被合并为一个不完整span $I_{i\curvearrowright j}^{t_i,j}$或$I_{i\curvearrowleft j}^{t_i,j}$。根据$r$和$r+1$在依存树的关系，可以分为以下几种情况。(1) 如果$r+1$是r的儿子，那么显然需要通过合并$C_{r\rightarrow r}^{t_r}$和$C_{r+1\leftarrow r+1}^{t_{r+1}}$构成$I_{r\curvearrowright r+1}^{t_r,r+1}$。当$r$是$r+1$的儿子时的情况类似。(2) 如果$r+1$是$r$的子孙，那么需要合并$C_{r\rightarrow r}^{t_r}$和$C_{r+1\leftarrow j}^{t_{r+1},j}$构成$I_{r\curvearrowright j}^{t_r,j}$。当$r$是$r+1$的子孙时的情况类似。(3) 如果不属于前两种情况，那么r和$r+1$拥有共同的祖先节点，这样需要合并$C_{i\rightarrow r}^{t_i,r}$和$C_{r+1\leftarrow j}^{t_{r+1},j}$被合并为一个不完整span $I_{i\curvearrowright j}^{t_i,j}$（$i$为共同祖先）或$I_{i\curvearrowleft j}^{t_i,j}$（$j$为共同祖先）。

综上所述，算法3-2可以根据公式3-4正确计算词性分值。

1. $\forall 0 \le i \le n, t_i \in \mathcal{T}: \; C_{i\rightarrow i}^{t_i} = 0, C_{i\leftarrow i}^{t_i} = 0$ // *initialization*
2. **for** $w = 1$ **to** n **do** // *span width*
3. **for** $i = 0$ **to** $(n - w)$ **do** // *span start index*
4. $j = i + w$ // *span end index*
5. **for all** $(t_i, t_j) \in \mathcal{T}^2$ **do**
6. $I_{i\curvearrowright j}^{t_i,j} = \max_{i\le r<j} \max_{(t_r,t_{r+1})\in\mathcal{T}^2}\{C_{i\rightarrow r}^{t_i,r} + C_{r+1\leftarrow j}^{t_{r+1},j} + \text{Score}_{\text{dep}}(\mathbf{x}, t_i, t_j, i, j) + \text{Score}_{\text{pu}}(\mathbf{x}, j, t_j) + \text{Score}_{\text{pb}}(\mathbf{x}, r+1, t_r, t_{r+1})\}$ // *corresponding to Fig. 3-4-(a).*
7. $I_{i\curvearrowleft j}^{t_i,j} = \max_{i\le r<j} \max_{(t_r,t_{r+1})\in\mathcal{T}^2}\{C_{i\rightarrow r}^{t_i,r} + C_{r+1\leftarrow j}^{t_{r+1},j} + \text{Score}_{\text{dep}}(\mathbf{x}, t_i, t_j, j, i) + \text{Score}_{\text{pu}}(\mathbf{x}, i, t_i) + \text{Score}_{\text{pb}}(\mathbf{x}, r+1, t_r, t_{r+1})\}$
8. $C_{i\rightarrow j}^{t_i,j} = \max_{i<r\le j} \max_{t_r\in\mathcal{T}}\{I_{i\curvearrowright r}^{t_i,r} + C_{r\rightarrow j}^{t_r,j}\}$ // *corresponding to Fig. 3-4-(b).*
9. $C_{i\leftarrow j}^{t_i,j} = \max_{i\le r<j} \max_{t_r\in\mathcal{T}}\{C_{i\leftarrow r}^{t_i,r} + I_{r\curvearrowleft j}^{t_r,j}\}$
10. **end for**
11. **end for**
12. **end for**
13. **return** $\max_{t_0=\text{“\#”},t_n\in\mathcal{T}}\{C_{0\rightarrow n}^{t_0,n}\}$

算法 3-2 一阶联合解码算法（JO1）

Algo. 3-2 The decoding algorithm of the first-order joint model (JO1)

算法需要存储$2n^2q^2$个不完整span和$2n^2q^2$个完整span，其中$q = |\mathcal{T}|$。因此，算法的空间复杂度为$O(n^2q^2)$。算法第6行和第7的时间复杂度都是$O(nq^2)$；第8行

和第9行的时间复杂度都是$O(nq)$；外层对w、i、t_i和t_j的循环需要$O(n^2q^2)$；因此，算法3-2的时间复杂度为$O(n^3q^4)$。

3.4.2 二阶和三阶词性标注和依存句法分析联合模型（JO2 & JO3）

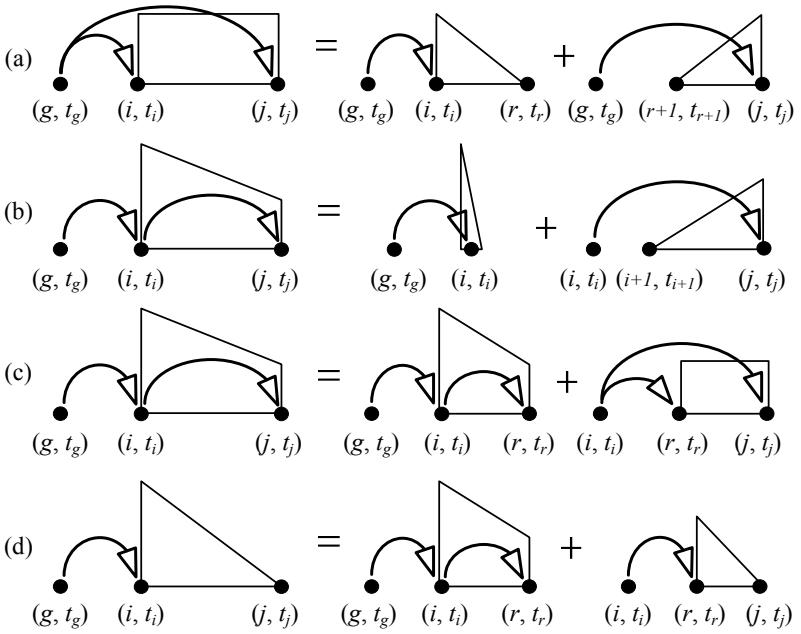


图 3-5 三阶联合模型的解码算法中使用到的基本动态规划数据结构和操作。

Fig. 3-5 The DP structures and derivations of the decoding algorithm of the third-order joint model.

Koo和Collins (2010)通过扩展Eisner算法，提出了一种基于动态规划的面向三阶依存句法分析模型的解码算法[22]，我们称之为Koo算法。我们扩展了Koo算法，提出一种基于动态规划的面向三阶联合模型的解码算法，如图3-5和算法3-3所示。为了融入更复杂子树对应的高阶句法特征，Koo算法引入了一种新的span，命名为兄弟span（*sibling span*），来表示两个词修饰同一个词（兄弟依存弧）的结构；另外，每个span增加一个父亲索引，从而表示父亲-儿子-孙子的结构。和我们在一阶联合模型解码算法中的扩展类似，我们在Koo算法的span结构上增加了边界和父亲索引位置的词性。$S^{t_g,i,j}_{g\curvearrowright i,g\curvearrowright j}$表示一个覆盖了$w_i..w_j$的兄弟span，并且$i$和$j$为$g$的两个相邻（adjacent）的儿子，$w_g$的词性为$t_g$，$w_i$的词性为$t_i$，$w_j$的词性为$t_j$，例如图3-5-(a)中左侧的span。$I^{t_g,i,j}_{g\curvearrowright i,i\curvearrowright j}$表示一个不完整span，和$I^{t_i,j}_{i\curvearrowright j}$表示的含义相同，但是额外要求$i$的父亲是$g$，并且$w_g$的词性为$t_g$。$C^{t_g,i,j}_{g\curvearrowright i,i\rightharpoonup j}$表示一个完整span，和$C^{t_i,j}_{i\rightharpoonup j}$表示的含义相同，但是额外要求$i$的父亲是$g$，同时$w_g$的词性为$t_g$。

1. $\forall 0 \le i \le n, -1 \le g \le n, (t_g, t_i) \in \mathcal{T}^2 :\ C^{t_{g,i}}_{g\curvearrowright i, i\rightarrow i} = 0, C^{t_{g,i}}_{g\curvearrowright i, i\leftarrow i} = 0$ // *initialization*
2. **for** $w = 1$ **to** n **do** // *span width*
3. **for** $i = 0$ **to** $(n - w)$ **do** // *span start index*
4. $j = i + w$ // *span end index*
5. **for all** g such that $-1 \le g < i$ **or** $j < g \le n$ **do** // *parent index*
6. **for all** $(t_g, t_i, t_j) \in \mathcal{T}^3$ **do**
7. $S^{t_{g,i,j}}_{g\curvearrowright i, g\curvearrowright j} = \max_{i\le r<j} \max_{(t_r,t_{r+1})\in\mathcal{T}^2}\{C^{t_{g,i,r}}_{g\curvearrowright i, i\rightarrow r} + C^{t_{g,r+1,j}}_{g\curvearrowright j, r+1\leftarrow j} + \text{Score}_{\text{sib}}(\mathbf{x}, t_g, t_i, t_j, g, i, j) + \text{Score}_{\text{pb}}(\mathbf{x}, r+1, t_r, t_{r+1})\}$ // *corresponding to Fig. 3-5-(a).*
8. $I^{t_{g,i,j}}_{g\curvearrowright i, i\curvearrowright j} = \max_{t_{i+1}\in\mathcal{T}}\{C^{t_{g,i}}_{g\curvearrowright i, i\rightarrow i} + C^{t_{i,i+1,j}}_{i\curvearrowright j, i+1\leftarrow j} + \text{Score}_{\text{dep}}(\mathbf{x}, t_i, t_j, i, j) + \text{Score}_{\text{grd}}(\mathbf{x}, t_g, t_i, t_j, g, i, j) + \text{Score}_{\text{sib}}(\mathbf{x}, t_i, t_j, i, -, j) + \text{Score}_{\text{gsib}}(\mathbf{x}, t_g, t_i, t_j, g, i, -, j) + \text{Score}_{\text{pb}}(\mathbf{x}, i+1, t_i, t_{i+1}) + \text{Score}_{\text{pu}}(\mathbf{x}, j, t_j)\}$ // *j is the first child of i; see Fig. 3-5-(b).*
9. $I^{t_{g,i,j}}_{g\curvearrowright j, i\curvearrowleft j} = \max_{t_{j-1}\in\mathcal{T}}\{C^{t_{i,j-1,j}}_{i\curvearrowleft j, i\rightarrow j-1} + C^{t_{g,j}}_{g\curvearrowright j, j\leftarrow j} + \text{Score}_{\text{dep}}(\mathbf{x}, t_i, t_j, j, i) + \text{Score}_{\text{grd}}(\mathbf{x}, t_g, t_i, t_j, g, j, i) + \text{Score}_{\text{sib}}(\mathbf{x}, t_i, t_j, j, -, i) + \text{Score}_{\text{gsib}}(\mathbf{x}, t_g, t_i, t_j, g, j, -, i) + \text{Score}_{\text{pb}}(\mathbf{x}, j, t_{j-1}, t_j) + \text{Score}_{\text{pu}}(\mathbf{x}, i, t_i)\}$ // *i is the first child of j.*
10. $I^{t_{g,i,j}}_{g\curvearrowright i, i\curvearrowright j} = \max\{ I^{t_{g,i,j}}_{g\curvearrowright i, i\curvearrowright j},\ \max_{i<r<j}\max_{t_r\in\mathcal{T}}\{I^{t_{g,i,r}}_{g\curvearrowright i, i\curvearrowright r} + S^{t_{i,r,j}}_{i\curvearrowright r, i\curvearrowright j} + \text{Score}_{\text{dep}}(\mathbf{x}, t_i, t_j, i, j) + \text{Score}_{\text{grd}}(\mathbf{x}, t_g, t_i, t_j, g, i, j) + \text{Score}_{\text{gsib}}(\mathbf{x}, t_g, t_i, t_r, t_j, g, i, r, j) + \text{Score}_{\text{pu}}(\mathbf{x}, j, t_j)\}\}$ // *corresponding to Fig. 3-5-(c).*
11. $I^{t_{g,i,j}}_{g\curvearrowright j, i\curvearrowleft j} = \max\{ I^{t_{g,i,j}}_{g\curvearrowright j, i\curvearrowleft j},\ \max_{i<r<j}\max_{t_r\in\mathcal{T}}\{S^{t_{i,r,j}}_{i\curvearrowleft j, r\curvearrowleft j} + I^{t_{g,r,j}}_{g\curvearrowright j, r\curvearrowleft j} + \text{Score}_{\text{dep}}(\mathbf{x}, t_i, t_j, j, i) + \text{Score}_{\text{grd}}(\mathbf{x}, t_g, t_i, t_j, g, j, i) + \text{Score}_{\text{gsib}}(\mathbf{x}, t_g, t_i, t_r, t_j, g, j, r, i) + \text{Score}_{\text{pu}}(\mathbf{x}, i, t_i)\}\}$
12. $C^{t_{g,i,j}}_{g\curvearrowright i, i\rightarrow j} = \max_{i<r\le j}\max_{t_r\in\mathcal{T}}\{I^{t_{g,i,r}}_{g\curvearrowright i, i\curvearrowright r} + C^{t_{i,r,j}}_{i\curvearrowright r, r\rightarrow j} + \text{Score}_{\text{sib}}(\mathbf{x}, t_i, t_r, i, r, -) + \text{Score}_{\text{gsib}}(\mathbf{x}, t_g, t_i, t_r, g, i, r, -)\}$ // *r is the last child of i; corresponding to Fig. 3-5-(d).*
13. $C^{t_{g,i,j}}_{g\curvearrowright j, i\leftarrow j} = \max_{i\le r<j}\max_{t_r\in\mathcal{T}}\{C^{t_{i,r,j}}_{r\curvearrowleft j, i\leftarrow r} + I^{t_{g,r,j}}_{g\curvearrowright j, r\curvearrowleft j} + \text{Score}_{\text{sib}}(\mathbf{x}, t_r, t_j, j, r, -) + \text{Score}_{\text{gsib}}(\mathbf{x}, t_g, t_r, t_j, g, j, r, -)\}$ // *r is the last child of j.*
14. **end for**
15. **end for**
16. **end for**
17. **end for**
18. **return** $\max_{t_{-1}=\text{“\#”}, t_0=\text{“\#”}, t_n\in\mathcal{T}}\{C^{t_{-1,0,n}}_{-1\curvearrowright 0, 0\rightarrow n}\}$

算法 3-3 三阶联合解码算法（JO3）

Algo. 3-3 The decoding algorithm of the third-order joint model

算法3-3的工作流程和一阶联合模型的情况类似，稍微复杂一些。第7行尝试确定分值最高的兄弟span $S^{t_{g,i,j}}_{g\curvearrowright i,g\curvearrowright j}$。给定$r$、$t_r$和$t_{r+1}$，通过合并两个完整span $C^{t_{g,i,r}}_{g\curvearrowright i,i\rightarrow r}$和$C^{t_{g,r+1,j}}_{g\curvearrowright j,r+1\leftarrow j}$可以得到一个候选span。这个操作会增加两个分值：一个是sibling特征对应的句法分值$\mathrm{Score}_{\mathrm{sib}}(\mathbf{x}, t_g, t_i, t_j, g, i, j)$；另一个是二元词性特征对应的词性分值$\mathrm{Score}_{\mathrm{pb}}(\mathbf{x}, r+1, t_r, t_{r+1})$。算法尝试所有可能的$r$、$t_r$和$t_{r+1}$，以确定分值最高的兄弟span，并将其存储在$S^{t_{g,i,j}}_{g\curvearrowright i,g\curvearrowright j}$。

第8行和第10行共同确定最优的不完整span$I^{t_{g,i,j}}_{g\curvearrowright i,i\curvearrowright j}$。第8行处理$j$作为$i$的第一个儿子的情况，对应图3-5-(b)。$j$作为$i$的第一个儿子，这种句法结构也会触发sibling特征，其对应的句法分值记为$\mathrm{Score}_{\mathrm{sib}}(\mathbf{x}, t_i, t_j, i, -, j)$。同样，$\mathrm{Score}_{\mathrm{gsib}}(\mathbf{x}, t_g, t_i, t_j, g, i, -, j)$表示$j$作为$i$的第一个儿子、同时$i$修饰$g$的这种grandparent-sibling结构对应的句法分值。第10行处理j不是i的第一个儿子的情况，对应图3-5-(c)。

第12尝试确定最优的完整span $C^{t_{g,i,j}}_{g\curvearrowright i,i\rightarrow j}$. 算法中span合并规则决定了一旦$C^{t_{g,i,j}}_{g\curvearrowright i,i\rightarrow j}$形成，那么$i$在以后的操作中不会再增加新的右儿子。因此这个操作会触发最后一个儿子相关的sibling和grandparent-sibling特征。$\mathrm{Score}_{\mathrm{sib}}(\mathbf{x}, t_i, t_r, i, r, -)$表示$r$作为$i$的最后一个儿子对应的句法分值。$\mathrm{Score}_{\mathrm{gsib}}(\mathbf{x}, t_g, t_i, t_r, g, i, r, -)$则表示$r$作为$i$的最后一个儿子同时$i$修饰$g$对应的句法分值。

简洁起见，我们略过对算法第9、11和13行的解释。它们的目的是确定最优的以w_j为核心词的span。搜索过程和上面的解释是类似的。和一阶联合解码算法类似，对于span无法表示、而特征函数又需要的词性如$t_{g\pm1}$和$t_{s\pm1}$，我们采用基准CRF词性标注模型产生的1-best结果。

同样，我们可以证明算法3-3的正确性，也就是说：1）按照公式3-4和公式3-7正确计算span的词性分值和句法分值；2）可以正确搜索到全局最优解。计算span的词性分值时，我们采用和一阶联合模型解码算法中类似的做法，因此可以类似的说明算法3-3按照公式3-4正确计算词性分值。关于正确计算句法分值的讨论，可以参考Eisner算法和Koo算法的证明过程。通过分析不难得出，算法的空间和实践复杂度分别为$O(n^3q^3)$和$O(n^4q^5)$。

二阶联合模型（JO2）也采用算法3-3作为解码算法，区别在于解码过程中不考虑所有三阶grandparent-sibling特征$\mathrm{Score}_{\mathrm{gsib}}(.)$贡献的句法分值。

3.4.3 基于边缘概率的词性剪枝策略

联合模型解码算法的时间复杂度很高，尤其是词性候选$q = |\mathcal{T}|$的阶数很高。McDonald (2006)为了约束搜索空间，利用基于最大熵的词性标注模型为每个词提供2-best词性候选，约束每个词只能从两个候选中选择词性，即$q = 2$[25]。然而，我们发现即使采用这种约束，联合模型的效率仍然太低，尤其对于二阶和三阶联合模型。和基准依存句法分析模型相比，二阶和三阶联合模型的时间复杂度为q^5倍。当$q = 2$时，会达到惊人的$2^5 = 32$倍。为了解决这个问题，我们使用基于CRF的词性标注模型产生每个词性的边缘概率，然后利用边缘概率对每个词的词性候选进行剪枝。实验发现，这种策略可以很有效的约束词性搜索空间。

给定输入句子$\mathbf{x}$，w_i标为t的边缘概率为：

$$\begin{aligned} P(t_i = t|\mathbf{x}) &= \sum_{\mathbf{t}:\mathbf{t}[i]=t} P(\mathbf{t}|\mathbf{x}) \\ &= \frac{\sum_{\mathbf{t}:\mathbf{t}[i]=t} \exp(\text{Score}_{\text{pos}}(\mathbf{x}, \mathbf{t}))}{\sum_{\mathbf{t}'} \exp(\text{Score}_{\text{pos}}(\mathbf{x}, \mathbf{t}'))} \end{aligned} \tag{3-11}$$

其中$\mathbf{t}[i] = t$表示词性序列$\mathbf{t}$的第i个位置词性为t，$P(\mathbf{t}|\mathbf{x})$和$\text{Score}_{\text{pos}}(\mathbf{x}, \mathbf{t})$的定义在前面的公式3-3和公式3-4中给出。利用*forward-backward*算法可以快速计算出边缘概率。

我们定义$\text{ptmax}_i(\mathbf{x})$为标注$w_i$的最大边缘概率：

$$\text{ptmax}_i(\mathbf{x}) = \max_{t \in \mathcal{T}} P(t_i = t|\mathbf{x}) \tag{3-12}$$

然后，我们定义w_i的候选词性集合为：

$$\mathcal{T}_i(\mathbf{x}) = \{t : t \in \mathcal{T}, P(t_i = t|\mathbf{x}) \geq \lambda_t \times \text{ptmax}_i(\mathbf{x})\} \tag{3-13}$$

其中λ_t为词性剪枝阈值。这样，联合模型解码算法3-2和算法3-3中，可以用$\mathcal{T}_i(\mathbf{x})$替换$\mathcal{T}$，从而约束需要遍历的$w_i$的词性。例如，算法3-3第6行中遍历词性的过程$(t_g, t_i, t_j) \in \mathcal{T}^3$可以替换为$t_g \in \mathcal{T}_g(\mathbf{x})$，$t_i \in \mathcal{T}_i(\mathbf{x})$和$t_j \in \mathcal{T}_j(\mathbf{x})$。

第3.5.1节的实验结果显示这种剪枝策略非常有效。当$\lambda_t = 0.01$时，在CTB5数据集上，每个词平均只有1.40个词性候选，而搜索空间的oracle词性准确率可以达到99.27%。我们使用10份交叉验证（10-fold jackknifing）的方式对训练集合进行词性裁剪。首先，训练集中的句子随机分为10份。然后，利用9份数据中的句子训练一个基于CRF的词性标注模型，并用这个模型对剩余一份的句子进行裁剪，如此重复10次。对于开发集和测试集，我们用整个训练集合中的句子训练CRF词性标注器，然后对开发集和测试集进行裁剪。

3.4.4 基于平均感知器的训练算法

基于平均感知器的训练算法已经在多个结构化分类问题上取得了成功，如词性标注[26]和句法分析[19, 97]。我们采用感知器训练算法学习基准依存句法分析模型和联合模型的特征权重。算法3-4给出了平均感知器算法应用于联合模型时的伪代码。算法以迭代的方式训练权重。每次迭代时需要遍历整个训练集合中的实例。每次只根据一个实例更新特征权重向量。首先，基于当前的特征权重向量，搜索得到最优的联合解，见第6行。然后，比较返回的最优解和人工标注的正确解，并根据比较结果更新特征权重。如果返回的最优解包含错误，那么需要更新特征权重，提高正确解中包含的特征的权重，降低错误解中包含的特征的权重。相反，如果返回的最优解完全正确，那么特征权重向量不变。

1. **Input:** Training Data $\mathbb{D} = \{(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)})\}_{j=1}^{N}$
2. **Output:** $\mathbf{w}_{\text{joint}}$ ($\equiv \mathbf{w}_{\text{pos}\oplus\text{syn}}$)
3. **Initialization:** $\mathbf{w}_{\text{joint}}^{(0)} = \mathbf{0}; \mathbf{v} = \mathbf{0}; k = 0$
4. **for** $i = 1$ **to** I **do** // *iterations*
5. **for** $j = 1$ **to** N **do** // *traverse the samples*
6. $(\hat{\mathbf{t}}, \hat{\mathbf{d}}) = \arg\max_{\mathbf{t},\mathbf{d}} \mathbf{w}_{\text{joint}}^{(k)} \cdot \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}, \mathbf{d})$ // *decode based on current feature weights*
7. **if** $\hat{\mathbf{t}} \neq \mathbf{t}^{(j)}$ **or** $\hat{\mathbf{d}} \neq \mathbf{d}^{(j)}$ **then**
8. $\mathbf{w}_{\text{joint}}^{(k+1)} = \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) - \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})$
9. **else**
10. $\mathbf{w}_{\text{joint}^{(k+1)}} = \mathbf{w}_{\text{joint}^{(k)}}$
11. **end if**
12. $\mathbf{v} = \mathbf{v} + \mathbf{w}_{\text{joint}}^{(k+1)}$
13. $k = k + 1$
14. **end for**
15. **end for**
16. $\mathbf{w}_{\text{joint}} = \mathbf{v}/(I \times N)$ // *average the weights*

算法 3-4 平均感知器训练词性标注和依存句法分析联合模型

Algo. 3-4 Averaged perceptron training for joint POS tagging and dependency parsing

使用平均感知器训练基准依存句法分析模型时，只需要对算法3-4进行小的改动。训练基准模型和联合模型时，对每个模型我们都运行20词迭代，然后

选择在开发集上准确率最高的参数。

3.5 实验和分析

我们采用依存句法分析最常用的Penn Chinese Treebank 5.1（CTB5）[76]作为实验数据，比便和前人的工作进行比较。借鉴前人工作[34, 36, 60]，我们将数据集分为训练集（001-815和1001-1136），开发集（886-931和1148-1151），和测试集（816-885和1137-1147）。我们使用Zhang和Clark的核心节点映射规则（head-finding rules）[36]，将短语结构转化为依存结构。

评价指标： 我们使用标准的词性准确率（*tagging accuracy, TA*）来评价词性标注。对于依存句法分析，我们使用UAS作为主要的评价指标。我们也给出了RA和CM以便和前人的结果进行更细致的比较。然而根节点准确率和完全匹配率不是很可靠，因为测试数据集一般只包含约2,000个句子。所有和句法相关的评价指标都忽略标点符号。

3.5.1 词性剪枝策略的影响

表 3-3 词性裁剪阈值λ_t的影响

Table.3-3 Impact of the POS tag pruning threshold λ_t

λ_t	$\lvert\mathcal{T}_i(\mathbf{x})\rvert$	oracle	UAS	RA	CM	TA	speed
0.001	1.83	99.55	82.85	76.75	31.50	94.52	0.5
0.01	1.40	99.27	**83.02**	**78.63**	**31.75**	94.52	1.0
0.1	1.16	98.07	82.77	77.38	31.25	**94.55**	1.7

为了研究词性裁剪的作用，我们尝试用不同的裁剪阈值λ_t对数据集合进行裁剪，然后在训练集合上训练二阶联合模型（JO2），并考察和比较模型在开发集上的性能。我们采用二阶联合模型，一方面原因是和三阶模型相比，二阶模型的效率比较高；另一方面和一阶模型相比，二阶模型可以融入丰富的特征，准确率更高。结果在表3-3中给出。$|\mathcal{T}_i(\mathbf{x})|$表示使用对应的裁剪阈值裁剪后开发集上每个词的平均候选词性的数目；“Oracle”表示搜索空间中包含的oracle词性准确率，即如果一个词的候选中包含其正确词性，那么就算这个词的词性正确；“Speed”表示模型平均每秒钟可以处理的句子数。可以发现，使用裁剪阈值$\lambda_t = 0.01$可以获得最高的句法准确率。并且，裁剪阈值的变化似乎对词性准

确率的影响不大。因此，我们采用$\lambda_t = 0.01$。这样，搜索空间中每个词平均只有1.40个词性候选，并且可以达到99.27%的oracle词性准确率。

简单起见，我们不再为其他的联合模型实验最优的λ_t。对于所有的联合模型，我们都采用$\lambda_t = 0.01$。

3.5.2 CTB5上的实验结果

3.5.2.1 正确词性下基准依存句法分析模型的实验结果

表3-4给出当给定人工标注的正确词性时，依存句法分析模型在CTB5测试集上的性能。我们还在表的下面几行罗列了一些最新的结果。Duan07表示Duan等(2007)论文中最好的结果[60]。他们利用柱搜索（beam search）增大基于转移的依存模型的搜索空间。Z&C08代表Zhang和Clark (2008)的结果[36]。他们提出一种混合模型，可以结合基于图和基于转移的依存分析模型的优势。H&S10表示Huang和Sagae (2010)的结果[34]。他们在基于转移的依存模型的柱搜索解码中加入等价状态合并，结合柱搜索和动态规划，进一步扩大了基于转移的依存模型的搜索空间。Z&N11表示Zhang和Nivre (2011)的基于转移的依存分析模型[35]。他们在一基于转移的依存模型中加入了丰富的特征，有些特征借鉴于基于图的模型。他们的模型取得了当时在CTB5上最好的句法准确率。Hatori11表示Hatori等(2011)的基准依存句法分析模型的准确率[13]。他们重新实现了Zhang和Nivre (2011)的基于转移的依存句法分析模型。我们还尝试了两个广泛使用的开源依存句法分析器：MSTParser和MaltParser。MSTParser1表示McDonald等(2005)提出的一阶依存句法分析模型[19]，MSTParser2表示McDonald和Pereira (2006)提出的二阶依存句法分析模型[20]。MaltParser是一个基于转移的依存句法分析系统，实现了多种分析算法和多种转移系统。我们采用支持向量机（Support Vector Machine, SVM）分类器和arc-standard转移系统[37]。

可以看到，在给定正确词性的情况下，我们实现的基准二阶和三阶依存句法分析模型取得了目前为止最好的句法准确率。令人意外的事，二阶模型比三阶模型的准确率稍高。我们推测可能的原因是在这个数据集上，grandparent-sibling特征过于复杂，比较稀疏，因此无法提供帮助。另外，Koo和Collins的实验结果也显示三阶模型在英语上只比二阶模型高0.23%，而在捷克语上，只提高了0.07%。

表 3-4 正确词性下，基准依存句法分析模型在测试集上的性能

Table. 3-4 Performance of the pipeline parsing models with gold-standard POS tags on the test set.

	UAS	RA	CM
O3	86.60	80.26	36.07
O2	**86.72**	80.26	36.07
O1	83.77	73.40	28.38
Hatori11[13]	85.96	**80.87**	35.03
Z&N11[35]	86.0	—	**36.9**
H&S10[34]	85.20	78.32	33.72
Z&C08[36]	85.77	76.26	34.41
Duan07[60]	83.88	73.70	32.70
MSTParser2	85.24	77.43	33.19
MSTParser1	83.04	71.52	27.59
MaltParser	82.62	69.37	29.06

表 3-5 自动词性下，基准依存句法分析模型在测试集上的性能

Table. 3-5 Performance of the pipeline parsing models with automatic POS tags on the test set.

	UAS	RA	CM	TA	speed
O3	80.50	77.33	28.22	**93.88**	0.7
O2	**80.64**	77.33	**28.59**		2.7
O1	77.19	70.05	22.72		11.5
MSTParser2	78.00	71.73	24.92		4.1
MSTParser1	76.09	67.80	21.16		5.2
MaltParser	75.98	66.44	23.98		2.6
Hatori11[13]	78.04	**75.55**	26.07	93.82	**9.0**

表 3-6 联合模型在测试集上的性能

Table. 3-6 Performance of the joint models on the test set.

	UAS	RA	CM	TA	speed
JO3	81.26	77.07	**30.47**	94.19	0.27
JO2	81.30	77.17	29.32	**94.28**	1.0
JO1	78.09	70.68	23.56	94.05	**5.8**
Hatori11[13]	**81.33**	**77.93**	29.90	93.94	1.5

3.5.2.2 自动词性下基准依存句法分析模型的实验结果

表3-5给出了自动词性下，基准依存句法分析模型在测试集上的准确率。对比表Table 3-4中的结果，可以发现自动词性下句法分析准确率下降了6-8%。这说明词性标注中的错误严重影响句法分析。因为词性为句法分析提供了非常关键的信息，因此当词性中含有噪音时，句法分析会受很大的影响。

我们的基准依存句法分析模型（二阶和三阶）比其他的模型性能好很多。表中还比较了各个模型的句法分析速度。我们的二阶模型比Hatori11稍微低一些，但是句法准确率高很多。

3.5.2.3 联合模型的实验结果

表3-6给出了联合模型在CTB5测试集上的性能。对比表3-5，我们可以看到联合模型可以同时帮助词性标注和依存句法分析。对于所有一阶、二阶和三阶的情况，句法准确率都有提高（约0.7-0.9%）。词性标注准确率也可以从基准模型的93.88%提高到94.28%，这是大约6.7%的错误率下降。这个结果说明了联合模型可以缓解错误蔓延问题，从而提高句法的准确率。另外，联合模型允许句法信息为词性标注提供反馈，从而帮助了词性标注的准确率。和Hatori11的基于转移的联合模型相比，我们的二阶和三阶联合模型取得了几乎相同的句法准确率和更高的词性准确率。Hatori11的实验结果同样说明了词性标注和依存句法分析联合模型的有效性。

从分析速度的角度看，可以发现我们提出的词性裁剪策略是非常有效的。基准二阶依存句法分析模型平均每秒钟可以处理2.7个句子，而我们二阶联合模型平均每秒钟可以处理1.0个句子。考虑到这两个模型解码算法的时间复杂度差了q^5的因子，这个速度的差距是非常小的。

3.5.3 错误分析

为了更深入的了解联合模型对词性标注和依存句法分析这两个子任务的影响，我们细致比较和分析了基准基于CRF的词性标注模型、基准二阶依存句法分析模型和二阶联合模型的结果。

3.5.3.1 联合模型对词性标注的影响

图3-6比较了CTB5测试集上基准词性标注模型和联合模型在一些高频的词性标注错误模式上的数目分布。一个词性错误模式“X → Y”表示一个焦点词的正确词性是“X”，而统计模型的标注结果是“Y”。在级联方法中，名词动词歧义对应的错误比例相当大，其中“VV → NN”占15%，“NN → VV”占11%。和级

联模型方法相比，联合模型可以更好的消解一些词性歧义如{VV, NN}和{DEG, DEC}[3]。这些词性歧义的共同特点是一旦歧义对中一种词性被错误标注为另外一种词性，那么在级联的依存句法分析模型中，局部的甚至全局的句法结构都会受到影响。换句话说，从句法分析的角度看，正确消解这些歧义非常关键。因此，我们称这类词性歧义为句法敏感的词性歧义（*syntax-insensitive ambiguities*）。从另外一个角度看，联合模型能够利用句法结构的帮助，更好的消解这类歧义。

相反，对于有的词性歧义如{NN, NR}和{NN, JJ}，如果将一种词性错误识别为另一种词性，则对句法结构的影响很小。因此我们称这类词性歧义为句法不敏感的词性歧义（*syntax-insensitive ambiguities*）。从结果来看，联合模型可以略微减少“NR → NN”和“JJ → NN”这两种模式的错误数，但同时略微增加了“NN → NR”和“NN → JJ”的错误数。

总之，我们可以得出一个结论：联合模型可以借助句法结构帮助消解句法敏感的词性歧义，而更好的消解这类词性歧义对于句法分析很有帮助。

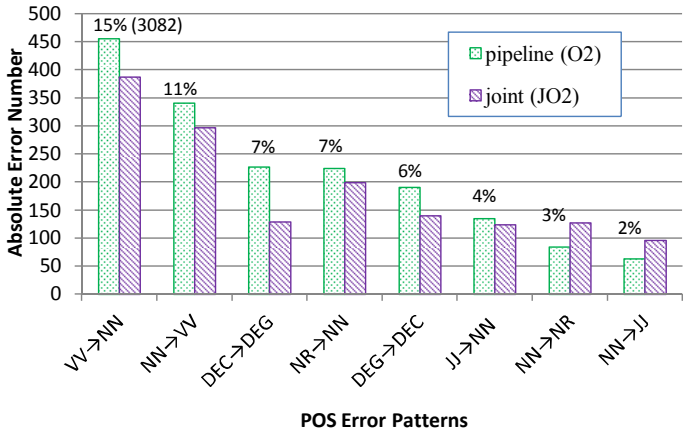


图 3-6　词性错误分布

Fig.3-6　POS error analysis

3.5.3.2 联合模型对依存句法分析的影响

表3-7考察了在CTB5测试集上联合模型如何影响不同词性标注模式的句法错误率。给定一个词性模式“X → Y”，“percent”表示正确词性为“X”而被模型标注为“Y”的词的比例；“error rate”表示对应模式的词语中被模型赋予了错误父亲的

[3]DEG和DEC是汉语中频繁使用的助词“的”对应的两个词性标记。DEG表示关联性的（associative）“的”，如“(父亲) (的/DEG) (眼睛)”。而DEC表示在关系从句（relative clause）中使用的“的”，如“(他) (取得) (的/DEC) (进步)”。

比例；最后两列给出了与级联方法（基准词性标注模型或依存句法分析模型）相比，联合模型对“percent”和“error rate”的影响（增大或减小）。

表 3-7 对比级联方法和联合模型在不同词性模式上的句法错误率。

Table. 3-7 Comparison of parsing error rates on different POS tag patterns between the pipeline and joint models.

POS patterns		pipeline		joint	
		percent	error rate	percent (±)	error rate (±)
Correct	NN → NN	95.7	16.0	-0.6	-1.3
	VV → VV	90.4	31.7	+0.3	-1.3
	DEG → DEG	85.0	9.8	+4.0	-2.8
	DEC → DEC	80.7	18.3	+8.3	-5.3
	NR → NR	90.7	14.4	+1.1	-0.6
	JJ → JJ	81.3	8.9	0	-0.6
Syntax-sensitive	NN → VV	2.5	54.3	-0.3	+20.8
	VV → NN	6.6	62.5	-1.0	+4.7
	DEG → DEC	15.0	50.3	-4.0	+49.0
	DEC → DEG	19.1	45.8	-8.2	+51.1
Syntax-insensitive	NN → NR	0.6	23.8	+0.3	+0.6
	NN → JJ	0.5	17.5	+0.2	+0.2
	NR → NN	7.1	24.6	-0.8	-8.0
	JJ → NN	10.4	17.0	-0.9	+2.4

对于正确的模式，即词性标注结果正确的模式，联合模型提高了大部分模式的比例，除了“NN”和“JJ”。这说明联合模型可以正确标注更多对应词性（如“VV”）的词语，或者说对应词性的召回率提高。更重要的是，联合模型可以降低所有正确模式的句法错误率。这说明，当词语的词性正确时，联合模型可以比级联方法更准确的找到对应词的父亲。

对于句法敏感的词性错误模式，联合模型的句法错误率急剧增大。但是这种词性错误模式的比例也大大减少。换句话说，联合模型可以很好的消解这种词性歧义，但是一旦标注错误，句法分析的结果会变得很差。这暗示在联合模型的框架下，句法敏感的词性歧义可以更好的消解，而句法结构也更信赖模型选择的词性。

对于句法不敏感的词性模式，联合模型的影响似乎比较小，没有什么规律。这个结果似乎也比较合理。因为这种词性歧义通常扮演类似的句法角色，因此句法结构不能提供更多的帮助。同样，这种词性歧义是否正确消解也对句

法结构的影响比较小。

总之，我们可以得出结论：联合模型可以更好的消解句法敏感的词性歧义，而词性标注的结果对于句法分析也变得更要有用和值得信赖。这是句法准确率提高的最根本的原因。

3.6 本章小结

本章提出并深入系统的研究了汉语词性标注和依存句法分析联合模型。我们扩展了前人提出的面向依存句法分析的解码算法，提出了相应的面向联合模型的基于动态规划的解码算法。并且，为了解决联合解码算法的时间复杂度过高的问题，我们又提出了一种有效地基于边缘概率的词性裁剪方法。我们在CTB5的实验结果表明联合模型可以提高词性和句法准确率。细致的错误分析表明联合模型可以更好地消解句法敏感的词性歧义，而正确的消解这些歧义可以进一步帮助句法分析。

第 4 章 面向联合模型的分离被动进取训练算法

4.1 引言

Li等(2011)发现词性标注和依存句法分析联合模型可以较大幅度的提高句法分析的准确率，但是却导致词性标注准确率有较大的下降[12]。详细的错误分析发现联合模型可以更好的消解句法敏感的词性歧义如{NN, VV}；但是同时在句法不敏感的词性歧义如{NN, NR}变得很差。我们认为最根本的原因是在联合模型中，由于句法特征完全淹没了词性特征的作用，因此联合模型完全被句法特征主导，因此只从句法结构的角度选择合适词性。另外，Hatori等(2011)发现基于转移的联合模型对词性标注的帮助也很小，只能提高让词性标注准确率提高约0.1%[13]。这些结果似乎都不是很合理。联合模型中句法结构可以帮助词性标注，因此应该同时较大幅度的提高词性标注的准确率。

Li等(2011)和Hatori等(2011)的联合模型都采用了基于平均感知器（Averaged Perceptron, AP）的训练算法。AP对词性特征和句法特征同等看待，采用相同的更新步长。然而，我们实验发现，在一个联合结果中，约包含3,000个非零的（non-zero）句法特征，而只包含约200个非零的词性特征。由于采用相同的步长更新权重，可以推测训练后的模型中句法特征和词性特征的权重具有接近的规模（scale）。因此，一个联合结果中，句法特征对应的分值远大于词性特征对应的分值。我们的实验发现，联合模型的最优联合结果中，平均来讲，词性分值只占句法分值的1/50。也就是说，句法特征在联合模型中占有绝对的话语权，而词性特征完全被压制，无法充分发挥其消歧作用。

因此，在本章中我们专注于改进联合模型的训练算法，目的是充分发挥词性和句法特征的消歧作用。我们对前人提出的被动进取（Passive-Aggressive，PA）训练算法进行改进，提出了一种名为分离被动进取（Separately Passive-Aggressive，SPA）的面向词性标注和依存句法分析联合模型的训练算法。SPA基于当前模型返回的最优联合解，根据这个联合解中的词性错误数为词性特征计算出一个更新步长，根据句法错误数为句法特征计算出另一个更新步长，这样在特征权重更新时，分别对词性特征和句法特征采用不同的更新步长。我们的实验发现，根据SPA的步长计算公式，词性特征的更新步长总是远大于句法特征的步长，这样自然就会使得词性特征的权重总是大于句法特征

的权重，从而可以提高词性特征在决定最优联合解时的比重。实验结果也证明了SPA的有效性。具体的，本章的贡献可以总结为以下几个方面：

- 我们针对词性标注和依存句法分析联合模型提出一种分离被动进取训练算法。我们从模型分值的角度分析和比较了PA、PA和SPA，证明SPA更加适合联合模型。实验结果证明SPA比AP和PA在词性和句法准确率上表现更好，使得联合模型比级联方法在词性标注准确率和句法分析准确率上都有较大幅度的提高。
- 我们提出并且实验了第一个包含特征丰富的、基于图的词性标注和有句法关系的依存句法的联合模型。我们在两个版本的CTB5数据集上做了实验，并在两个数据集上都取得了目前最好的词性准确率和句法准确率。具体来讲，基于SPA的联合模型取得了94.7%的词性准确率，远高于基于CRF的基准词性标注模型的93.9%。
- 我们在英语数据集Penn Treebank (PTB)上做了实验。由于英文单词中包含比较丰富的形态变化，因此英语的词性标注比汉语相对容易一些。PTB上的词性标注准确率约97%，远高于CTB5上的94%。因此，对于英语，级联方法中，依存句法分析模块面对的错误级联问题远没有汉语中中那么严重。然而，我们在PTB上的实验仍然发现联合模型可以帮助词性标注和依存句法分析。词性标注准确率提高了约0.5%，句法分析准确率提高约0.4%。我们的联合模型在这这个数据集上也取得了当前最好的词性和句法准确率。
- 我们首次系统深入的研究了句法关系对依存句法分析准确率的影响。我们发现，与无句法关系（unlabeled）的依存句法分析相比，有句法关系（labeled）的依存句法分析可以取得更高的句法准确率（无句法关系的依存弧准确率，UAS）。具体的，在两个版本的CTB5数据集上，有句法关系的依存句法分析可以使句法准确率提高0.6-0.7%。在英语数据集PTB上，依存句法分析的同时确定句法关系，可以使句法准确率提高0.3-0.4%。

4.2 相关工作

联合模型是一种同时解决和提高多个相关工作的有效方法。仅在ACL和EMNLP 2012年会议上，就有不少研究者提出了多种不同的联合模型，如面向社交文本（tweets）的命名实体识别和规划化（normalization）联合模型[98]，多词复合短语（multi-word expression）和句法分析联合模型[99]，面向

邮件文本的模版补全（template filling）联合模型[100]等。另外，一些研究者开始研究分词、词性标注和依存句法分析联合模型[15–17]。

Li等(2011)首先提出词性标注和依存句法分析联合模型[12]。利用Eisner的思想[24]，我们对基于图的依存句法分析解码算法进行了扩展，提出基于图的联合模型。当时的实验表明联合模型可以帮助句法分析，但是会导致词性标注准确率下降。和此工作相比，本章通过专注于研究联合模型训练算法，提出一种更加有效的训练算法，可以同时提高词性和句法准确率；另外，我们采用了更丰富的句法特征，并且模型可以同时确定依存句法关系类型。我们还在英语数据上做了实验和分析。

之后，Hatori等(2011)首先提出基于转移的词性标注和依存句法分析联合模型[13]。他们的实验发现联合模型可以提高句法准确率，但是对词性准确率的提高很小。尔后，Bohnet和Nivre (2012)对一种更复杂的基于转移的依存句法分析系统进行扩展，从而可以处理非投影结构（non-projective），并且在形成依存结构的同时确定依存句法关系[14]。他们在多种语言上做了实验，包括汉语、英语、捷克语和德语。和Hatori等(2011)类似，他们的实验也表明联合模型可以较大幅度提高句法准确率，但是只能小幅度的提高词性准确率。和这两个工作不同，我们发现，基于图的联合模型经过SPA算法训练后，可以同时较大幅度提高词性和句法准确率。并且，联合模型在汉语和英语数据集上都取得了最好的词性和句法准确率。

4.3 级联方法

级联方法将词性标注和依存句法分析看成两个独立的有先后顺序任务。首先，对于输入句子$\mathbf{x} = w_1...w_n$，确定一个最优的词性序列$\hat{\mathbf{t}}$

$$\hat{\mathbf{t}} = \arg\max_{\mathbf{t}} \mathrm{Score}_{\mathrm{pos}}(\mathbf{x}, \mathbf{t}) \tag{4-1}$$

然后，基于$\mathbf{x}$ and $\hat{\mathbf{t}}$，确定一个最优的依存树$\hat{\mathbf{d}}$。

$$\hat{\mathbf{d}} = \arg\max_{\mathbf{d}} \mathrm{Score}_{\mathrm{syn}}(\mathbf{x}, \hat{\mathbf{t}}, \mathbf{d}) \tag{4-2}$$

形式化地，一个依存句法树表示为$\mathbf{d} = \{(h, m, l) : 0 \le h \le n, 1 \le m \le n, l \in \mathcal{L}\}$，其中$(h, m, l)$表示一个从核心词（head, father）$w_h$到修饰词（modifier，dependent, child）w_m的依存弧，l表示依存弧的关系类型，$\mathcal{L}$是依存弧关系类型的集合。

4.3.1 基于条件随机域的词性标注模型

和第三章相同，本章采用CRF实现我们的基准词性标注模型。CRF属于条件对数线性（log-linear）概率模型。给定输入句子$\mathbf{x}$，一个词性序列$\mathbf{t}$的条件概率为：

$$P(\mathbf{t}|\mathbf{x}) = \frac{\exp(\mathrm{Score}_{\mathrm{pos}}(\mathbf{x}, \mathbf{t}))}{\sum_{\mathbf{t}'} \exp(\mathrm{Score}_{\mathrm{pos}}(\mathbf{x}, \mathbf{t}'))} \tag{4-3}$$

$$\begin{aligned} \mathrm{Score}_{\mathrm{pos}}(\mathbf{x}, \mathbf{t}) &= \mathbf{w}_{\mathrm{pos}} \cdot \mathbf{f}_{\mathrm{pos}}(\mathbf{x}, \mathbf{t}) \\ &= \sum_{1 \le i \le n} \mathrm{Score}_{\mathrm{pu}}(\mathbf{x}, i, t_i) + \mathrm{Score}_{\mathrm{pb}}(\mathbf{x}, i, t_{i-1}, t_i) \end{aligned} \tag{4-4}$$

$$\begin{aligned} \mathrm{Score}_{\mathrm{pu}}(\mathbf{x}, i, t_i) &= \mathbf{w}_{\mathrm{pu}} \cdot \mathbf{f}_{\mathrm{pu}}(\mathbf{x}, i, t_i) \\ \mathrm{Score}_{\mathrm{pb}}(\mathbf{x}, i, t_{i-1}, t_i) &= \mathbf{w}_{\mathrm{pb}} \cdot \mathbf{f}_{\mathrm{pb}}(\mathbf{x}, i, t_{i-1}, t_i) \end{aligned} \tag{4-5}$$

其中$\mathbf{f}_{\mathrm{pos}}(\mathbf{x}, \mathbf{t})$表示$(\mathbf{x}, \mathbf{t})$对应的聚合的词性特征向量，$\mathbf{w}_{\mathrm{pos}}$表示词性特征的权重向量。我们使用两类词性特征，分别为：一元词性特征（POS unigram features），记为$\mathbf{f}_{\mathrm{pu}}(\mathbf{x}, i, t_i)$；二元词性特征（*POS bigram features*），记为$\mathbf{f}_{\mathrm{pb}}(\mathbf{x}, i, t_{i-1}, t_i)$。对于汉语，我们采用第三章表3-1中的词性特征。对于英语，我们借鉴Ranaparkhi的工作[93]。表4-1中列出了所有面向英语的词性特征模版。其中，contain_uppercase(w_i)考虑w_i中是否包含大写字母；contain_hyphen(w_i)回答w_i中是否包含连字符，如“state-of-the-art”包含连字符；contain_number(w_i)回答w_i中是否包含数字“0-9”；#c_i表示w_i中的字符数；prefix(w_i, k)表示w_i的前k个字符构成的前缀；suffix(w_i, k)表示w_i的后k个字符构成的后缀。值得注意的是，Ranaparkhi只针对低频词（rare words）使用06-10特征，而我们不区分低频词和高频词，对每个词都是用所有的特征。我们认为，这种做法可以利用高频词的一些前后缀的信息来帮助低频词的标注。但是我们没有做实验严格对比。

同样，我们使用Collins等提出的*exponentiated gradient (EG)*算法[95]来学习特证权重$\mathbf{w}_{\mathrm{pos}}$。解码时，针对输入句子，我们采用经典的Viterbi算法搜索最优的词性序列。

4.3.2 基于图的依存句法分析模型

按照主流的分类，数据驱动的依存句法分析模型可以分为两种。基于转移的方法[32, 35, 37] 通过一个移动/规约（shift/reduce）的动作序列建立依存句法树，并且尝试寻找最优的动作序列。而基于图的方法[19, 21, 22] 将依存句法分析看成

表 4-1 英语词性特征集合，用于基准词性标注模型和联合模型。

Table.4-1 English POS tagging features used in both tagging and joint models.

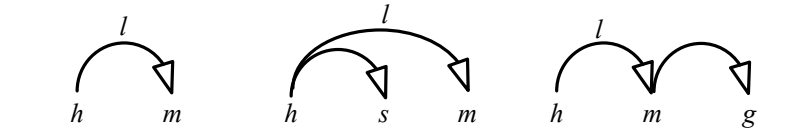
POS Unigram Features: $\mathbf{f}_{\text{pu}}(\mathbf{x}, i, t_i)$
01: $t_i \circ w_i$
02: $t_i \circ w_{i-1}$
03: $t_i \circ w_{i+1}$
04: $t_i \circ w_{i-2}$
05: $t_i \circ w_{i+2}$
06: $t_i \circ \text{contain\_uppercase}(w_i)$
07: $t_i \circ \text{contain\_hyphen}(w_i)$
08: $t_i \circ \text{contain\_number}(w_i)$
09: $t_i \circ \text{prefix}(w_i, k), 1 \le k \le 4, k \le \#c_i$
10: $t_i \circ \text{suffix}(w_i, k), 1 \le k \le 4, k \le \#c_i$
POS Bigram Features: $\mathbf{f}_{\text{pb}}(\mathbf{x}, i, t_{i-1}, t_i)$
11: $t_i \circ t_{i-1}$

一个从完全有向图中寻找最大生成树的问题。我们关注基于图的依存句法分析模型。为了保证基于动态规划的全局搜索算法的效率，基于图的模型需要做很强的独立假设。假设句子对应的依存树中，只有存在于某些特殊结构（子树，subtree）中的依存弧之间才互相联系和影响，其他依存弧之间则互相独立。在这种假设下，一个依存树的分值便分解为一些子树的分值的和。

$$\begin{aligned}\text{Score}_{\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) &= \mathbf{w}_{\text{syn}} \cdot \mathbf{f}_{\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \\ &= \sum_{p \subseteq \mathbf{d}} \text{Score}_{\text{subtree}}(\mathbf{x}, \mathbf{t}, p)\end{aligned} \tag{4-6}$$

其中，p表示一个独立假设允许的子树。p包含一个或多个$\mathbf{d}$中的依存弧。$\text{Score}_{\text{subtree}}(\mathbf{x}, \mathbf{t}, p)$表示由$p$贡献的分值。$\mathbf{f}_{\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d})$是$(\mathbf{x}, \mathbf{t}, \mathbf{d})$对应的聚合的句法特征向量，$\mathbf{w}_{\text{syn}}$是句法特征的权重向量。我们采用Carreras提出的二阶依存句法分析模型[21]，因为前人的实验表明这个模型在很多语言上获得了很高的句法准确率[22, 72]。另外，这个模型可以处理依存句法关系（label），而我们希望可以深入考察句法关系对句法准确率的影响。Carreras的模型中，一棵依存句法树的分值由图4-1三种子树对应的分值累加构成。

$$\begin{aligned}\mathtt{Score}_{\mathrm{syn}}(\mathbf{x},\mathbf{t},\mathbf{d}) = &\sum_{\{(h,m,l)\}\subseteq\mathbf{d}} \mathbf{w}_{\mathrm{dep}}\cdot\mathbf{f}_{\mathrm{dep}}(\mathbf{x},\mathbf{t},h,m,l)\\ &+\sum_{\{(h,m,l),(h,s)\}\subseteq\mathbf{d}} \mathbf{w}_{\mathrm{sib}}\cdot\mathbf{f}_{\mathrm{sib}}(\mathbf{x},\mathbf{t},h,m,l,s)\\ &+\sum_{\{(h,m,l),(m,g)\}\subseteq\mathbf{d}} \mathbf{w}_{\mathrm{grd}}\cdot\mathbf{f}_{\mathrm{grd}}(\mathbf{x},\mathbf{t},h,m,l,g)\end{aligned} \tag{4-7}$$



(a) *single dependency* (b) *adjacent sibling* (c) *outermost grandchild*

图 4-1 Carreras模型中使用的几种子树。

Fig.4-1 Different types of scoring subtrees used in the graph-based models proposed by Carreras[21].

我们采用Bohnet (2010)总结和整理的一套丰富有效的句法特征集合[72]。这个特征集合共包含三个类别，分别对应图4-1中的三种子树。表4-2简要总结每种类别中使用的原子特征。b表示h和m之间的一个下标；$dir(i,j)$和$dist(i,j)$分别表示依存弧(i,j)的方向和距离。如果读者需要完整的特征模版列表，请参考第三章表3-2，和本章使用的特征模版的区别是，每一个特征模版需要增加句法关系l；另外，g在第三章表3-2表示h的父亲节点，即m的祖父节点，而在此处表示m的儿子节点，即h的孙子节点。这种对应关系是很直接的，所以很容易转化过来。对于无句法关系的依存句法分析模型，我们只需省略表中的句法关系类型l。和Li等(2011)的工作[12]中使用的句法特征相比，这个特征集合采用更丰富的上下文词性如$t_{s\pm1}$和$t_{g\pm1}$。我们的实验发现，这些上下文词性可以提高句法准确率。另外，对于汉语，我们借鉴Che等(2012)中的做法[5]，使用每个词语的最后一个汉字作为词干信息，然后对每一个和词相关的特征，用词干替换词构成一个新的特征。CTB5上的实验结果表明这种词干相关的特征可以让基准依存句法分析模型的准确率（UAS）提高约0.3-0.4%。对于英语而言，我们借鉴Koo和Collins (2010)的做法[22]，使用粗粒度（coarse-grained）的词性来替换词性相关的特征，形成新的特征。例如，名词复数（NNP）退化为名词（NN）。PTB上的实验表明这种粗粒度词性相关的特征可以让基准依存句法分析模型的UAS提高约0.4%。

Carreras (2007)为这个依存句法模型提出了一种解码算法，我们称之为Carreras算法[21]。Carreras算法的空间复杂度为$O(|\mathcal{L}|n^3)$，时间复杂度为$O(|\mathcal{L}|n^4)$。在此，我们不详细介绍Carreras算法，因为后面详细介绍的联合

表 4-2 每类特征使用的原子特征的简要总结。

Table.4-2 Brief illustration of the atomic features used in each category of features.

Feature category	Atomic features incorporated
Dependency features $\mathbf{f}_{\text{dep}}(\mathbf{x}, \mathbf{t}, h, m, l)$	$l, w_h, w_m, t_h, t_m, t_{h\pm1}, t_{m\pm1}, t_b, dir(h,m), dist(h,m)$
Sibling features $\mathbf{f}_{\text{sib}}(\mathbf{x}, \mathbf{t}, h, m, l, s)$	$l, w_h, w_s, w_m, t_h, t_m, t_s, t_{h\pm1}, t_{m\pm1}, t_{s\pm1}, dir(h,m), dist(h,m)$
Grandchild features $\mathbf{f}_{\text{grd}}(\mathbf{x}, \mathbf{t}, h, m, l, g)$	$l, w_h, w_m, w_g, t_h, t_m, t_g, t_{h\pm1}, t_{m\pm1}, t_{g\pm1}, dir(h,m), dir(m,g)$

模型的解码算法是对Carreras算法的直接扩展，并且很容易退化为Carreras算法。

4.4 联合模型

在联合模型的框架下，我们的目标是同时解决词性标注和依存句法分析这两个问题，寻找一个联合最优解。

$$(\hat{\mathbf{t}}, \hat{\mathbf{d}}) = \arg\max_{\mathbf{t},\mathbf{d}} \text{Score}_{\text{joint}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \tag{4-8}$$

一棵包含词性标注的依存句法树的分值是词性序列分值和句法树分值之和。而词性序列分值和依存树分值的定义和我们在基准模型中的定义一致。

$$\begin{aligned} \text{Score}_{\text{joint}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) &= \text{Score}_{\text{pos}}(\mathbf{x}, \mathbf{t}) + \text{Score}_{\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \\ &= \mathbf{w}_{\text{pos}} \cdot \mathbf{f}_{\text{pos}}(\mathbf{x}, \mathbf{t}) + \mathbf{w}_{\text{syn}} \cdot \mathbf{f}_{\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \\ &= \mathbf{w}_{\text{pos}\oplus\text{syn}} \cdot \mathbf{f}_{\text{pos}\oplus\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) = \mathbf{w}_{\text{joint}} \cdot \mathbf{f}_{\text{joint}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \end{aligned} \tag{4-9}$$

其中⊕表示向量拼接。联合模型同时学习词性特征和句法特征的权重向量（$\mathbf{w}_{\text{pos}\oplus\text{syn}}$或$\mathbf{w}_{\text{joint}}$），因此可以这两种特征可以互相交互和影响，模型也可以更好地平衡这两种特征的消歧作用，从而确定最优的联合解。

4.4.1 基于动态规划的联合模型的解码算法

和上一章的工作相似，我们基于Eisner的想法[24]，对依存句法分析模型的Carreras解码算法[21]进行扩展，提出面向联合模型的基于动态规划（Dynamic Programming, DP）的解码算法。我们使用图4-2和算法4-1详细描述算法。其核心思想是将Carreras算法中的基本数据结构span进行扩展，增加边界词和某个最远儿子的词性信息。一个span表示覆盖某个句子子串的局部依存句法结构。例如，图4-2-(a)左侧的span，表示一个覆盖了$w_i...w_j$、包含依存弧(i, j, l)的不完

整span（*incomplete span*），并且w_i的词性为t_i、w_j的词性为t_j，记为$I_{l,i\curvearrowright j}^{t_i,j}$。图4-2-(b)左侧的span，表示一个覆盖了$w_i...w_j$、$r$是$i$的的最右儿子的完整span（*complete span*），并且w_i的词性为t_i、w_r的词性为t_r、w_j的词性为t_j，记为$C_{i\rightarrow j,i\curvearrowright r}^{t_i,r,j}$。在这两个span中，$w_i$都成为span的核心词（head word）。简洁起见，我们在图中略去生成以w_j为核心词的span的操作。以w_j为核心词的不完整和完整span分别记为$I_{l,i\curvearrowright j}^{t_i,j}$和$C_{i\leftarrow j,r\curvearrowright j}^{t_i,r,j}$。

算法4-1以自底向上的方式不断迭代，按照一定的规则，每次将两个相邻的span合并为一个更大的span。第7行尝试找到最优的不完整span $I_{l,i\curvearrowright j}^{t_i,j}$。给定$r$、$t_r$、$t_{r+1}$、$s$、$t_s$、$g$和$t_g$，算法通过合并两个完整span $C_{i\rightarrow r,i\curvearrowright s}^{t_i,s,r}$和$C_{r+1\leftarrow j,g\curvearrowleft j}^{t_{r+1},g,j}$，可以产生一个候选的$I_{l,i\curvearrowright j}^{t_i,j}$。这个过程对应图4-2-(a)。这个操作会引入一些新的分值，需要累加到生成的span中。最终，产生的候选span $I_{l,i\curvearrowright j}^{t_i,j}$的分值包含7部分，即：两个组件span的分值，$\mathbf{f}_{\text{dep}}(\mathbf{x}, t_i, t_j, i, j, l)$，$\mathbf{f}_{\text{sib}}(\mathbf{x}, t_i, t_s, t_j, i, j, l, s)$，$\mathbf{f}_{\text{grd}}(\mathbf{x}, t_i, t_g, t_j, i, j, l, g)$，$\mathbf{f}_{\text{pu}}(\mathbf{x}, j, t_j)$和$\mathbf{f}_{\text{pb}}(\mathbf{x}, r+1, t_r, t_{r+1)}$。需要特别指出的是，当$i == s$时，$\mathbf{f}_{\text{sib}}(\mathbf{x}, t_i, t_s, t_j, i, j, l, s)$表示$j$作为$i$的第一个儿子节点时的sibling特征对应的句法分值（first-sibling）；而当$g == j$时，$\mathbf{f}_{\text{grd}}(\mathbf{x}, t_i, t_g, t_j, i, j, l, g)$表示$j$是$i$的儿子，而$j$没有左儿子的grandchild特征对应的句法分值（no-grandchild）；我们的实验发现：这种first-sibling和no-grandchild特征都可以帮助句法分析。算法遍历所有的r、t_r、t_{r+1}、s、t_s、g和t_g，以确定分值最高的span $I_{l,i\curvearrowright j}^{t_i,j}$。需要注意的是，对$s$和$t_s$的遍历，与对$g$和$t_g$的遍历互相没有关系，因此可以单独进行，分别寻找最优的取值。我们在伪代码中用两条下划线分别标记出来这两部分独立的寻优操作。这样，第7行伪代码的时间复杂度为：$O(n|\mathcal{T}|^2\times(n|\mathcal{T}|+n|\mathcal{T}|))$，即$O(n^2|\mathcal{T}|^3)$。

值得注意的是，句法分值函数如$\mathbf{f}_{\text{dep}}(\mathbf{x}, t_i, t_j, i, j, l)$的参数列表中，只包含了有限的几个词性信息。但是根据表4-2，句法特征函数还需要一些上下文词性如$t_{i\pm1}$和$t_{j\pm1}$。和第三章中的做法类似，对于这些上下文词性，我们一律使用基于CRF的基准词性标注模型提供的1-best词性。

第11行尝试找到最优的完整span $C_{i\rightarrow j,i\curvearrowright r}^{t_i,r,j}$。给定$r$、$t_r$、$l$、$g$和$t_g$，算法通过合并$I_{l,i\curvearrowright r}^{t_i,r}$和$C_{r\rightarrow j,r\curvearrowright g}^{t_r,g,j}$，可以产生一个候选的$C_{i\rightarrow j,i\curvearrowright r}^{t_i,r,j}$。这个过程对应图4-2-(b)。这个操作会引入一些新的分值，需要累加到生成的span中。最终，产生的候选span $I_{l,i\curvearrowright j}^{t_i,j}$的分值包含3部分，即：两个组件span的分值和$\text{Score}_{\text{grd}}(\mathbf{x}, t_i, t_r, t_g, i, r, l, g)$。需要特别指出的是，只有当$r == j$时，才允许$r == g$，此时表示$r$是$i$的儿子，而$r$没有右儿子的grandchild特征对应的句法分值（no-grandchild）。算法遍历所有的r、t_r、l、g和t_g，以确定分值最高的span $C_{i\rightarrow j,i\curvearrowright r}^{t_i,r,j}$。这一行伪代码的时间复杂度是$O(n\times|\mathcal{T}|\times|\mathcal{L}|)$。

1. $\forall 0 \le i \le n, t_i \in \mathcal{T}: C^{t_i}_{i\rightarrow i,i\curvearrowright i} = 0, C^{t_i}_{i\leftarrow i,i\curvearrowleft i} = 0$ // *initialization*
2. **for** $w = 1$ **to** n **do** // *span width*
3. **for** $i = 0$ **to** $(n - w)$ **do** // *span start index*
4. $j = i + w$ // *span end index*
5. **for all** $(t_i, t_j) \in \mathcal{T}^2$ **do**
6. **for all** $l \in \mathcal{L}$ **do**
7. $I^{t_{i,j}}_{l,i\curvearrowright j} = \max_{i\le r<j} \max_{(t_r,t_{r+1})\in\mathcal{T}^2}\{\underline{\max_{i\le s<\le r,t_s\in\mathcal{T}}\{C^{t_{i,s,r}}_{i\rightarrow r,i\curvearrowright s} + \mathrm{Score}_{\mathrm{sib}}(\mathbf{x}, t_i, t_s, t_j, i, j, l, s)\}} + \underline{\max_{r+1\le g<\le j,t_g\in\mathcal{T}}\{C^{t_{r+1,g,j}}_{r+1\leftarrow j,g\curvearrowleft j} + \mathrm{Score}_{\mathrm{grd}}(\mathbf{x}, t_i, t_g, t_j, i, j, l, g)\}} + \mathrm{Score}_{\mathrm{dep}}(\mathbf{x}, t_i, t_j, i, j, l) + \mathrm{Score}_{\mathrm{pu}}(\mathbf{x}, j, t_j) + \mathrm{Score}_{\mathrm{pb}}(\mathbf{x}, r+1, t_r, t_{r+1})\}$ // *corresponding to Fig. 4-2-(a).*
8. $I^{t_{i,j}}_{l,i\curvearrowleft j} = \max_{i\le r<j} \max_{(t_r,t_{r+1})\in\mathcal{T}^2}\{\underline{\max_{i\le g<\le r,t_g\in\mathcal{T}}\{C^{t_{i,g,r}}_{i\rightarrow r,i\curvearrowright g} + \mathrm{Score}_{\mathrm{grd}}(\mathbf{x}, t_i, t_s, t_j, j, i, l, g)\}} + \underline{\max_{r+1\le s<\le j,t_s\in\mathcal{T}}\{C^{t_{r+1,s,j}}_{r+1\leftarrow j,s\curvearrowleft j} + \mathrm{Score}_{\mathrm{sib}}(\mathbf{x}, t_i, t_s, t_j, j, i, l, s)\}} + \mathrm{Score}_{\mathrm{dep}}(\mathbf{x}, t_i, t_j, j, i, l) + \mathrm{Score}_{\mathrm{pu}}(\mathbf{x}, i, t_i) + \mathrm{Score}_{\mathrm{pb}}(\mathbf{x}, r+1, t_r, t_{r+1})\}$
9. **end for**
10. **for all** $i < r \le j, t_r \in \mathcal{T}$ **do** // *r is the outermost child of i*
11. $C^{t_{i,r,j}}_{i\rightarrow j,i\curvearrowright r} = \max_{l\in\mathcal{L}} \max_{r\le g\le j} \max_{t_g\in\mathcal{T}}\{I^{t_{i,r}}_{l,i\curvearrowright r} + C^{t_{r,g,j}}_{r\rightarrow j,r\curvearrowright g} + \mathrm{Score}_{\mathrm{grd}}(\mathbf{x}, t_i, t_r, t_g, i, r, l, g)\}$ // *corresponding to Fig. 4-2-(b).*
12. **end for**
13. **for all** $i \le r < j, t_r \in \mathcal{T}$ **do** // *r is the outermost child of j*
14. $C^{t_{i,r,j}}_{i\leftarrow j,r\curvearrowleft j} = \max_{l\in\mathcal{L}} \max_{r\le g\le j} \max_{t_g\in\mathcal{T}}\{C^{t_{i,g,r}}_{i\leftarrow r,g\curvearrowleft r} + I^{t_{r,j}}_{l,r\curvearrowleft j} + \mathrm{Score}_{\mathrm{grd}}(\mathbf{x}, t_g, t_r, t_j, j, r, l, g)\}$
15. **end for**
16. **end for**
17. **end for**
18. **end for**
19. **return** $\max_{1\le r\le n} \max_{t_r\in\mathcal{T}} \max_{t_0=\text{“\#”},t_n\in\mathcal{T}}\{C^{t_{0,r,n}}_{0\rightarrow n,0\curvearrowright r}\}$

算法 4-1 二阶联合解码算法（JO2Carreras）

Algo. 4-1 The decoding algorithm of the second-order joint model

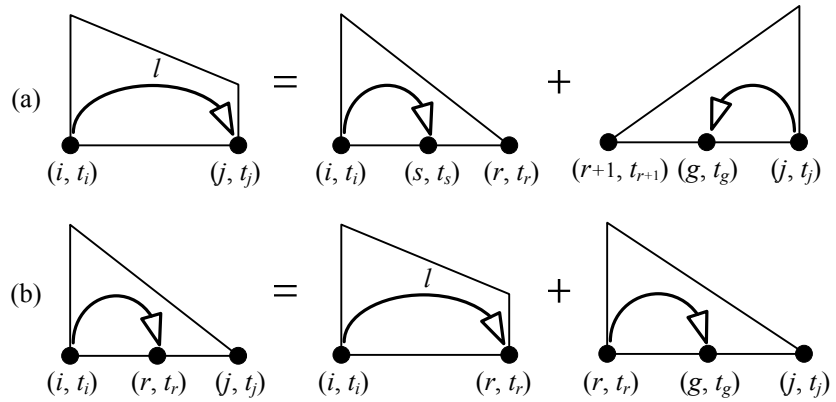


图 4-2 二阶联合模型的解码算法中使用到的基本动态规划数据结构和操作。

Fig.4-2 The DP structures and derivations of the decoding algorithm of the second-order joint models.

第8行和第14行分别尝试确定以w_j为核心词的不完整和完整span。简洁起见，我们不详细解释这两行伪代码。整个算法的时间复杂度为$O(|\mathcal{L}|n^4|\mathcal{T}|^5)$，空间复杂度为$O(|\mathcal{L}|n^2|\mathcal{T}|^2 + |n^3|\mathcal{T}|^3)$。

和Carreras算法的正确性证明类似，我们可以证明算法4-1 可以根据公式4-7正确计算span的句法分值。对于词性分值，算法4-1 采用了和第三章相同的做法，因此同样也根据公式4-4 正确计算每个span的词性分值。

4.4.2 基于边缘概率的裁剪策略

词性裁剪策略：由于联合模型解码算法的时间复杂度中每个词的词性候选数目$q = |\mathcal{T}|$的阶数很高。因此，我们采用第三章提出的词性裁剪方法，利用基于CRF的词性标注模型给出的边缘概率，对每个词的词性候选进行裁剪（裁剪阈值$\lambda_t = 0.01$）。裁剪后，CTB5上每个词平均只有1.4个候选词性，oracle词性准确率为99.27%；PTB上每个词平均有1.2个候选词性，oracle词性准确率为99.71%。

依存弧裁剪策略：由于联合模型解码算法的时间复杂度中句子长度n的阶数也很高。因此，我们使用coarse-to-fine的方法[22, 50, 101]，首先训练一个基于CRF的一阶依存句法分析模型（基于自动词性），然后利用这个概率模型得到每个依存弧的边缘概率，最后利用这个边缘概率对每个词的核心词候选进行裁剪。裁剪后，CTB5上保留了31.3%的依存弧，oracle依存弧准确率（UAS）为99.77%；PTB上保留了28.9%的依存弧啊，oracle依存弧准确率为99.91%。

4.5 一种分离被动进取训练算法

在线学习算法（online training）在多种结构化分类问题如词性标注[26]和句法分析[19, 97]上都取得了成功。算法4-2给出了在线学习算法用于训练词性标注和依存句法分析联合模型时的通用框架。在线学习算法以迭代的方式训练特征的权重。每次迭代会遍历一遍整个训练数据集合，每次根据一个训练实例的分析结果对权重向量进行调整。首先，基于当前的特征权重向量，模型为当前实例寻找到最优解，对应第6行。然后，对比返回的最优解和人工标注的正确解，更新当前的特征权重向量，对应第7行。第12行将特征权重进行平均化。这种平均化策略由Collins (2002)提出[26]，发现可以较大幅度提高感知器算法在词性标注任务上的准确率。

1. **Input:** Training Data $\mathbb{D} = \{(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)})\}_{j=1}^{N}$
2. **Output:** $\mathbf{w}_{\text{joint}}$ ($\equiv \mathbf{w}_{\text{pos}\oplus\text{syn}}$)
3. **Initialization:** $\mathbf{w}_{\text{joint}}^{(0)} = \mathbf{0}; \mathbf{v} = \mathbf{0}; k = 0$
4. **for** $i = 1$ **to** I **do** // *iterations*
5. **for** $j = 1$ **to** N **do** // *traverse the samples*
6. $(\hat{\mathbf{t}}, \hat{\mathbf{d}}) = \arg\max_{\mathbf{t},\mathbf{d}} \mathbf{w}_{\text{joint}}^{(k)} \cdot \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}, \mathbf{d})$ // *decode based on current feature weights.*
7. $\mathbf{w}_{\text{joint}}^{(k+1)} = \text{update } \mathbf{w}_{\text{joint}}^{(k)} \text{ with } (\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \hat{\mathbf{t}}, \mathbf{d}^{(j)}, \hat{\mathbf{d}})$ // *update weights according to some criterion.*
8. $\mathbf{v} = \mathbf{v} + \mathbf{w}_{\text{joint}}^{(k+1)}$
9. $k = k + 1$
10. **end for**
11. **end for**
12. $\mathbf{w}_{\text{joint}} = \mathbf{v}/(I \times N)$ // *average the weights*

算法 4-2 线学习算法用于训练词性标注和依存句法分析联合模型时的通用框架

Algo. 4-2 Generic online training for joint POS tagging and dependency parsing

根据算法4-2第7行的权重更新准则，三种在线算法在句法分析上有广泛的应用，即平均感知器算法[26]、被动进取算法[27] 和Margin Infused Relaxed算法（MIRA）[28]。在本章工作之前，研究者们都采用AP训练词性标注和依存句法分析联合模型[12–14]。我们的实验表明，当用于训练联合模型时，AP和PA可以取得类似的词性和句法准确率。然后，我们通过分析和实验，提出一种PA算法的变形，命名为分离被动进取算法。我们发现SPA可以更好的训练联合模型，同时

提高词性和句法准确率。简单起见，我们没有做实验和MIRA进行比较，因为我们以前在依存句法分析模型上的实验表明MIRA、AP和PA的准确率相似。并且，我们可以类似提出Separately Margin Infused Relaxed算法（SMIRA）。我们相信，和SPA一样，SMIRA同样可以更好的训练联合模型。

AP、PA和SPA更新权重的方向都取决于人工标注的正确解对应的特征向量$\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)},\mathbf{d}^{(j)})$和模型返回的最优解对应的特征向量$\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)},\hat{\mathbf{t}},\hat{\mathbf{d}})$之间的距离。这个方向意味着增大正确解对应的特征向量，减小返回解对应的特征向量。不同的是，它们采取不同的方式决定更新权重的步长。AP算法中，步长始终为1，如公式4-10所示。

$$\text{AP}\left\{\ \mathbf{w}_{\text{joint}}^{(k+1)}=\mathbf{w}_{\text{joint}}^{(k)}+\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)},\mathbf{d}^{(j)})-\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)},\hat{\mathbf{t}},\hat{\mathbf{d}})\right. \tag{4-10}$$

Collins (2002)证明了AP应用于词性标注时的收敛性[26]。

PA算法通过一个稍微复杂的公式计算出一个动态步长τ_{joint}。这个公式考虑了返回解中的错误数，正确解和返回解的分值之差，以及正确解和返回解对应的特征向量之差的模。

$$\text{PA}\left\{\begin{array}{l}\tau_{\text{joint}}=\dfrac{\text{Score}_{\text{joint}}(\mathbf{x}^{(j)},\hat{\mathbf{t}},\hat{\mathbf{d}})-\text{Score}_{\text{joint}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)},\mathbf{d}^{(j)})+\rho_{\text{pos}}(\mathbf{t}^{(j)},\hat{\mathbf{t}})+\rho_{\text{syn}}(\mathbf{d}^{(j)},\hat{\mathbf{d}})}{\|\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)},\mathbf{d}^{(j)})-\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)},\hat{\mathbf{t}},\hat{\mathbf{d}})\|^2}\\ \mathbf{w}_{\text{joint}}^{(k+1)}=\mathbf{w}_{\text{joint}}^{(k)}+\tau_{\text{joint}}(\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)},\mathbf{d}^{(j)})-\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)},\hat{\mathbf{t}},\hat{\mathbf{d}}))\end{array}\right. \tag{4-11}$$

其中$\rho_{\text{pos}}(\mathbf{t}^{(j)},\hat{\mathbf{t}})$指与正确解$\mathbf{t}^{(j)}$对比，返回解$\hat{\mathbf{t}}$中包含的错误词性的数目（包括标点符号）。$\rho_{\text{syn}}(\mathbf{d}^{(j)},\hat{\mathbf{d}})$指与正确解$\mathbf{d}^{(j)}$对比，返回解$\hat{\mathbf{d}}$的句法错误数。我们借鉴Johansson和Nugues (2008)的做法[102]，如果依存弧错误，则$\rho_{\text{syn}}(\mathbf{d}^{(j)},\hat{\mathbf{d}})$加1，而如果依存弧正确而句法关系错误，则$\rho_{\text{syn}}(\mathbf{d}^{(j)},\hat{\mathbf{d}})$加0.5。PA的这个计算更新步长的公式是通过求解下面的凸二次优化问题推导出来的。

$$\begin{aligned}&\mathbf{w}_{\text{joint}}^{(k+1)}=\arg\min_{\mathbf{w}_{\text{joint}}}\frac{1}{2}\|\mathbf{w}_{\text{joint}}-\mathbf{w}_{\text{joint}}^{(k)}\|^2\\ &s.t.\ \ \mathbf{w}_{\text{joint}}\cdot\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)},\mathbf{d}^{(j)})-\mathbf{w}_{\text{joint}}\cdot\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)},\hat{\mathbf{t}},\hat{\mathbf{d}})\geq\rho_{\text{pos}}(\mathbf{t}^{(j)},\hat{\mathbf{t}})+\rho_{\text{syn}}(\mathbf{d}^{(j)},\hat{\mathbf{d}})\end{aligned} \tag{4-12}$$

这个凸二次优化问题的意思是最小的修改当前的特征权重向量$\mathbf{w}_{\text{joint}}^{(k)}$，使得正确解的分值和当前返回的最优解的分值的差大约等于返回解中的错误数。

可以看到，AP和PA都采用同样的步长来更新词性特征权重$\mathbf{w}_{\text{pos}}$和句法特征权重$\mathbf{w}_{\text{syn}}$。因此，可以推测在训练得到的模型中，每个词性特征和每个句法特征的权重的绝对大小（scale）是接近的。由于句法特征的数目远大于词性特征的数目，我们认为这样训练出的模型是有问题的。我们实验发现，平均来讲，在模型找到的最优联合解对应的特征向量$\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)},\hat{\mathbf{t}},\hat{\mathbf{d}})$中，句法特征向量

部分$\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)},\hat{\mathbf{t}},\hat{\mathbf{d}})$包含大于3,000个非零（non-zero）的特征，而词性特征向量部分$\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)},\hat{\mathbf{t}})$只有少于200个非零特征。因此，在联合模型中句法特征占据了主导地位，而词性特征在决定最优联合解时只起到很小的作用。事实上，我们实验发现，使用AP和PA训练时，在返回的最优联合解中，词性特征对应的分值只占所有特征对应的分值的很小的一部分（见图4-4）。

经过了一些实验尝试，我们发现可以通过增大词性特征的更新步长以提高词性分值在联合解的分值的比重。进而，我们发现对PA算法的更新公式进行简单的变形之后，就可以很优雅、很自然的达到这个目的。我们将变形后的训练算法命名为分离被动进取算法（*Separately Passive-Aggressive Algorithm，SPA*）。具体来讲，SPA按照下面的公式为词性特征和句法特征分别计算更新步长。

$$\text{SPA}\begin{cases}\tau_{\text{pos}}=\dfrac{\text{Score}_{\text{pos}}(\mathbf{x}^{(j)},\hat{\mathbf{t}})-\text{Score}_{\text{pos}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)})+\rho_{\text{pos}}(\mathbf{t}^{(j)},\hat{\mathbf{t}})}{\|\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)})-\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)},\hat{\mathbf{t}})\|^2}\\ \tau_{\text{syn}}=\dfrac{\text{Score}_{\text{syn}}(\mathbf{x}^{(j)},\hat{\mathbf{t}},\hat{\mathbf{d}})-\text{Score}_{\text{syn}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)},\mathbf{d}^{(j)})+\rho_{\text{syn}}(\mathbf{d}^{(j)},\hat{\mathbf{d}})}{\|\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)},\mathbf{d}^{(j)})-\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)},\hat{\mathbf{t}},\hat{\mathbf{d}})\|^2}\\ \mathbf{w}_{\text{pos}}^{(k+1)}=\mathbf{w}_{\text{pos}}^{(k)}+\tau_{\text{pos}}(\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)})-\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)},\hat{\mathbf{t}}))\\ \mathbf{w}_{\text{syn}}^{(k+1)}=\mathbf{w}_{\text{syn}}^{(k)}+\tau_{\text{syn}}(\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)},\mathbf{d}^{(j)})-\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)},\hat{\mathbf{t}},\hat{\mathbf{d}}))\end{cases}\tag{4-13}$$

和PA算法类似，公式4-13从理论上分别求解了两个凸优化子问题：

$$\begin{aligned}&\mathbf{w}_{\text{pos}}^{(k+1)}=\arg\min_{\mathbf{w}_{\text{pos}}}\frac{1}{2}\|\mathbf{w}_{\text{pos}}-\mathbf{w}_{\text{pos}}^{(k)}\|^2\\ &s.t.\ \ \mathbf{w}_{\text{pos}}\cdot\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)})-\mathbf{w}_{\text{pos}}\cdot\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)},\hat{\mathbf{t}})\geq\rho_{\text{pos}}(\mathbf{t}^{(j)},\hat{\mathbf{t}})\end{aligned}\tag{4-14}$$

$$\begin{aligned}&\mathbf{w}_{\text{syn}}^{(k+1)}=\arg\min_{\mathbf{w}_{\text{syn}}}\frac{1}{2}\|\mathbf{w}_{\text{syn}}-\mathbf{w}_{\text{syn}}^{(k)}\|^2\\ &s.t.\ \ \mathbf{w}_{\text{syn}}\cdot\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)},\mathbf{d}^{(j)})-\mathbf{w}_{\text{syn}}\cdot\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)},\hat{\mathbf{t}},\hat{\mathbf{d}})\geq\rho_{\text{syn}}(\mathbf{d}^{(j)},\hat{\mathbf{d}})\end{aligned}\tag{4-15}$$

即为词性特征权重向量和句法特征权重向量计算出两个最小的更新步长，使得更新后正确解的词性分值与返回的最优解的词性分值的差值大于等于词性错误数，并且正确解的句法分值与返回的最优解的句法分值的差值大于等于句法错误数。值得注意的是，公式4-13计算出的τ_{pos}或τ_{syn}可能是负的。原因是，正确解的词性分值或句法分值本身比返回的最优解的对应分值大，但是另一个分值却小很多，因此导致正确解没有成为最优解。这种计算出负的更新步长的情况不是很多（大多数情况是是句法特征对应的更新步长为负）。当出现这种情况时，我们将更新步长设为0，即不更新相应的特征权重。

由于$\mathbf{f}_{\text{pos}}(.)$中包含的非零特征比$\mathbf{f}_{\text{syn}}(.)$少很多，因此$\|\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)})-\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)},\hat{\mathbf{t}})\|^2$也比$\|\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)},\mathbf{t}^{(j)},\mathbf{d}^{(j)})-\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)},\hat{\mathbf{t}},\hat{\mathbf{d}})\|^2$小很多。从而，$\tau_{\text{pos}}$比$\tau_{\text{syn}}$大很多。我们的实

验结果表明τ_{pos}平均来讲大约是τ_{syn}的10倍。这意味着，词性特征权重的更新步长比句法特征的高权重大很多，这导致词性特征权重比句法特征的权重更大。我们实验发现，使用SPA训练出的联合模型中，联合解的词性分值和句法分值变得比较接近（参见图4-4）。这意味着词性特征可以在联合模型中扮演更重要的作用。

4.6 实验和分析

实验数据： 我们采用CTB5[76]作为最主要的实验数据。我们采用前人的数据分割[34, 36, 60]，并使用开源工具Penn2Malt[1]和Zhang和Clark (2008)的头结点搜索规则[36] 将数据从短语结构转为依存结构。

表 4-3 本章使用的数据汇总（句子数）。

Table.4-3 Data used in this work (in sentence number).

Corpus	Train	Dev	Test
CTB5	16,091	803	1,910
CTB5-Bohnet	16,069	803	1,905
PTB	39,832	1,346	2,416

为了和Bohnet和Nivre (2012)的联合模型进行比较，我们在他们使用的另一个版本的CTB5数据上做实验[14]。[2] 我们将这个数据集记为CTB5-Bohent。我们仔细对比了CTB5和CTB5-Bohnet，CTB5-Bohnet比CTB5少了大约30个句子，并且两个数据集的依存结构和依存关系都不同。和Bohnet讨论后，我们终于了解到原因是他使用了Yue Zhang的短语转依存工具[3]，而我们使用的是Penn2Malt。虽然我们采用的都是Zhang和Clark (2008)的头结点搜索规则，转化后的依存结构和依存关系的不一致可能是两个工具的实现不同。

英语数据我们采用PTB，02-21作为训练集，24作为开发集，23作为测试集。我们使用Penn2Malt和默认的头结点搜索将短语结构转化为依存结构。表4-3总结了本章中使用的数据。

评价指标： 我们使用标准的词性准确率（*POS tagging accuracy, TA*）评价词性标注。对于依存句法分析，我们使用无句法关系的附着分值（*unlabeled*

[1]http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html

[2]在此感谢Bernd Bohnet分享他们的数据。

[3]http://sourceforge.net/projects/zpar/files/0.3/

attachment score, UAS）和有句法关系的附着分值（*labeled attachment score, LAS*）来评价[8]。为了和前人的工作一致，我们计算UAS和LAS时都忽略标点符号。

4.6.1 CTB5上的实验结果

4.6.1.1 比较三种训练算法

图4-3给出了无句法关系联合模型使用三种训练算法的词性准确率和句法准确率曲线。每次迭代结束后，我们使用当前训练出的模型（平均特征权重）分析整个开发集，并且记录对应的词性准确率和句法准确率，从而得到图中的曲线。可以看到，和AP算法相比，PA算法训练出的模型在词性和句法上都表现更好一些。SPA在UAS上的峰值比PA算法略高。最重要的是，SPA在词性准确率上高于PA和AP。另外，图4-3也表明SPA可以和其他两种训练算法收敛得一样快。我们希望以后可以从理论上证明SPA的收敛性。

为了更好的理解SPA提高词性准确率的原因，我们从模型分值的角度对联合模型进行分析。联合模型返回的最优解的分值中包含两部分，分别为词性分值$\text{Score}_{\text{pos}}(.)$和句法分值$\text{Score}_{\text{pos}}(.)$。每次迭代后，我们使用当前训练出的模型分析整个开发集，然计算出平均的词性分值和句法分值。图4-4给出了三种训练算法下，联合模型中词性分值和句法分值的变化曲线。对于AP和PA算法，词性分值大约是句法分值的1/50 ($10^{1.7} = 50$)。这意味着词性分值在联合模型中的作用比较小。SPA算法下，词性分值是句法分值的1/8 ($10^{0.9} = 8$)，这说明SPA算法可以提高词性分值的比重。从而，词性特征可以在联合模型中做出更多的贡献。这样，句法特征在消解句法敏感的词性歧义的能力，和词性特征在消解句法不敏感的词性歧义的能力可以得到很好的平衡。我们认为这是SPA使得词性准确率提高的本质原因。

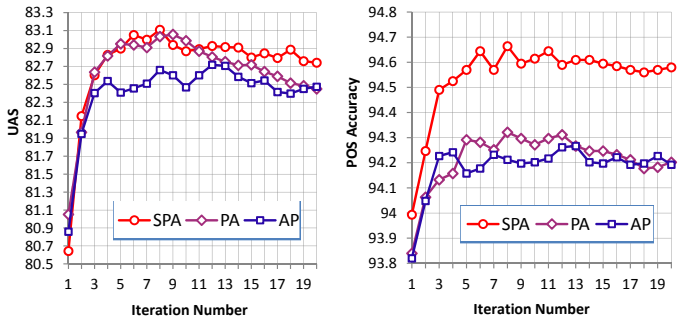


图 4-3 联合模型在CTB5开发集上的训练曲线。

Fig.4-3 Training curves of the unlabeled joint models on the CTB5 development dataset.

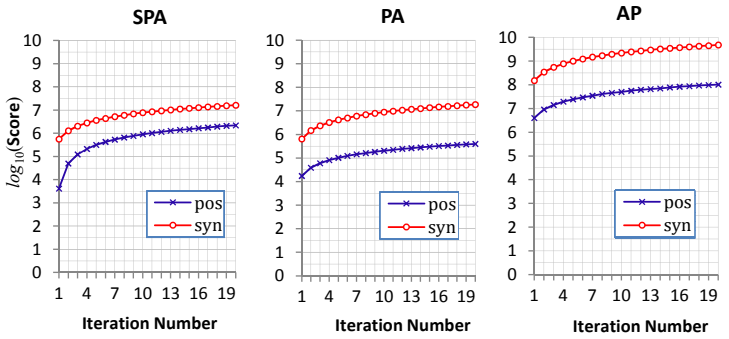


图 4-4　CTB5开发集上的联合模型分值分析。

Fig.4-4　Model score analysis on the CTB5 development dataset.

表4-4报告了SPA/PA/AP训练下无句法关系的联合模型在CTB5测试集上的性能。根据图4-3，我们使用在开发集上性能最高的迭代次数对应的模型。结果显示，使用AP和PA训练出的联合模型比基准词性标注模型的词性准确率高大约0.3%，这和第三章基于Koo解码算法的实验结果是类似的。进而，使用SPA训练的联合模型比AP和PA训练的联合模型在词性准确率上高0.3%，相当于5%的错误率下降；同时UAS提高大约0.2%。这说明了，SPA训练算法更加适合训练联合模型，可以更好的平衡词性特征和句法特征的消歧作用，充分利用词性特征消解句法不敏感的词性歧义的判别力，从而提高词性准确率。进而，更好的词性帮助也可以帮助句法分析。

表 4-4　CTB5测试集上的三种训练算法的性能比较

Table.4-4　Performance of three training algorithms on the CTB5 test dataset.

	UAS	TA
SPA	81.21	94.51
PA	80.98	94.18
AP	80.94	94.17
pipeline	80.22	93.88

4.6.1.2 最终实验结果

表4-5给出了CTB5测试集上的最终结果。根据词性配置的不同，我们给出了三组结果：“gold POS”表示基准依存句法分析模型采用人工标注的正确词性时的结果；“pipeline”表示基准依存句法分析模型采用基于CRF的词性标注基准模型标注的自动词性时的结果（级联方法）；“joint”表示联合模型的结果。对

于基准依存句法分析模型，我们采用PA训练算法训练特征权重；联合模型采用SPA训练算法。我们训练两种模型，一种是带句法关系类型的模型，一种是不带句法关系类型的模型，以研究句法关系对依存句法分析准确率的影响。可以看到，我们的模型在三种不同词性配置上都取得了最好的结果。更具体的，我们可以得出以下结论：

- 使用正确词性时，带句法关系的模型比不带句法关系的模型的句法准确率（UAS）高约0.2%；级联方法和联合模型中，带句法关系的模型使得UAS提高0.3-0.6%；联合模型中，带句法关系的模型也可以略微提高词性准确率（0.1%）。这说明了句法关系可以帮助句法分析。
- 和级联方法相比，SPA训练的联合模型可以较大幅度的提高词性准确率（约1.0%，相当于16%的错误率下降。）
- 联合模型比级联方法在UAS上1.0-1.3%，在LAS上高1.3%。

表 4-5 CTB5测试集上的最终实验结果比较

Table.4-5 Final results on the CTB5 test dataset

Models		LAS	UAS	TA
joint	this work (labeled)	**79.09**	**81.80**	**94.83**
	this work (unlabeled)	—	81.55	94.76
	JO2Koo (SPA)	—	81.50	94.76
	JO2Koo (AP)	—	81.17	94.27
	Li et al. (2011)	—	80.74	93.08
	Jun et al. (2011)	—	81.33	93.94
pipeline	this work (labeled)	**77.80**	**80.82**	**93.88**
	this work (unlabeled)	—	80.22	
	Li et al. (2011)	—	79.29	93.51
	Jun et al. (2011)	—	78.04	93.82
gold POS	this work (labeled)	**85.36**	**86.76**	100.0
	this work (unlabeled)	—	86.55	
	Li et al. (2011)	—	86.18	
	Jun et al. (2011)	—	85.96	
	Zhang and Nivre (2011)	84.4	86.0	
	Huang and Sagae (2010)	—	85.20	

另外，我们将SPA应用于第三章中的基于Terry Koo的解码算法的联合模型（JO2Koo）中，以进一步考察SPA对联合模型的影响，并和基于Carreras解码算法的模型进行比较。可以看到，SPA在JO2Koo联合模型上也取得了类似的性

能提升，句法准确率提升0.4%，词性准确率提升0.5%。同样使用SPA训练算法，JO2Koo和无句法关系的Carreras联合模型在词性和句法上的准确率都很接近。使用AP训练算法时，第三章表3-6中JO2Koo的句法准确率高一些，原因是第三章的实验中我们对JO2Koo模型进行了改进，额外使用了一些特殊的特征，如果使用这些特征，JO2Koo使用SPA训练算法的句法准确率可以达到81.65%。由于这些特征比较繁琐，我们在论文中略过这些讨论。如果读者对这些特征比较感兴趣，可以和作者联系，进一步讨论。

4.6.1.3 实验结果分析

为了考察各个特征集合对联合模型的贡献，我们每次从完整模型（full model）中减少一组特征集合，然后观察联合模型的性能变化。表4-6中给出了结果。完整模型指的是无句法关系的联合模型，使用SPA算法训练模型参数。可以看到，去掉词性特征使得词性准确率下降约0.5%，而对句法准确率影响不大，仅下降0.1%。相反，去掉三种子树对应的句法特征集合中的任何一组，都会导致句法准确率较大幅度的下降，而词性准确率下降比较小。

表 4-6 CTB5测试集上不同特征集合对联合模型的贡献分析

Table.4-6 Results of feature ablation on the CTB5 test dataset

	UAS	POS
full model	81.21	94.51
-pos	81.09	94.06
-dependency	80.85	94.32
-grand	80.57	94.39
-sibling	80.66	94.28

图4-5分析了各个模型在一些高频的词性错误模式上的表现。一个词性错误模式X→Y表示一个焦点词的正确词性是X，而统计模型的标注结果为Y。Li等(2011)的工作[12]发现，基于图的联合模型可以较大幅度的减少在句法敏感的词性歧义上的词性标注错误，但是同时大大增加了在句法不敏感的词性歧义上的错误数，这导致当时的结果中整体词性准确率下降。和Li等(2011)的结果相比，使用SPA训练的联合模型在句法不敏感的词性歧义上表现得好很多，错误数和基于CRF的词性标注基准模型很接近，甚至更好（如NR→NN和JJ→NN）。总之，我们可以得出结论：基于SPA训练算法的联合模型在句法不敏感的词性歧义上和序列标注方式的词性标注基准模型性能接近，并且在词性敏感的词性歧义上和Li等(2011)的联合模型的性能接近。这表明SPA训练算法可以更好的平

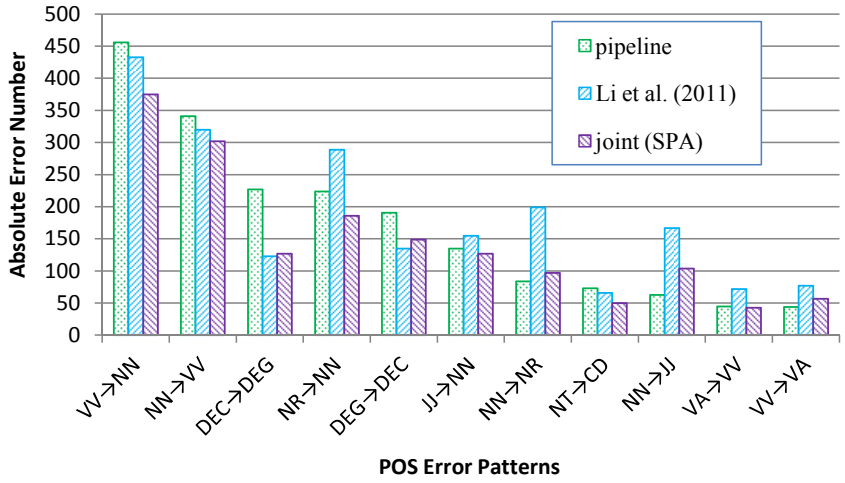


图 4-5 CTB5测试集上的词性错误模式分析。

Fig.4-5 Statistics for different POS error patterns on the CTB5 test dataset.

衡词性特征和句法特征的消歧作用。

表 4-7 CTB5-Bohnet测试集上的最终实验结果对比

Table.4-7 Final results on the CTB5-Bohnet test dataset.

Models		LAS	UAS	POS
joint	this work (labeled)	**80.18**	**83.14**	**94.71**
	this work (unlabeled)	—	82.37	94.65
	Bohnet and Nivre (2012)	77.91	81.42	93.24
pipeline	this work (labeled)	**79.01**	**82.07**	**93.89**
	this work (unlabeled)	—	81.38	
	Bohnet and Nivre (2012)	76.79	80.33	92.81

4.6.2 CTB5-Bohnet上的实验结果

为了和Bohnet和Nivre (2012)提出的基于转移的联合模型比较，我们在CTB5-Bohnet做实验并给出对比结果，如表4-7中所示。可以看到，我们的结果在级联方法和联合模型中都较大幅度的超过了他们的结果。和Hatori等(2011)提出的基于转移的联合模型的结果类似，他们的联合模型也采用AP训练算法，并且只能略微提高词性准确率（约0.4%）。需要注意的是，他们的基准词性标注模型的准确率很低，比我们的基准模型低1

另外，在这个数据集上，模型的句法准确率比在CTB5上高。我们推测原因是采

用Penn2Malt默认的头结点搜索规则转化得到的依存结构更加容易学习。我们希望以后可以深入研究这个问题。和CTB5类似，在这个数据上我们同样发现句法关系类型可以提高句法准确率（UAS）。

表 4-8 PTB测试集上的最终实验结果对比

Table.4-8 Final results on the PTB test dataset.

Models		LAS	UAS	POS
joint	this work (labeled)	**92.44**	**93.52**	**97.62**
	this work (unlabeled)	—	93.12	**97.62**
	Bohnet and Nivre (2012)	**92.44**	93.38	97.33
	Bohnet and Nivre (2012) †	*92.68*	*93.67*	*97.42*
pipeline	this work (labeled)	**92.00**	93.14	97.16
	this work (unlabeled)	—	92.85	
	Bohnet and Nivre (2012)	91.71	92.79	**97.28**
	Zhang and Nivre (2011)	—	92.9	—
	Martins et al. (2010)	—	**93.26**	—
	Koo and Collins (2010)	—	93.04	—
	Huang and Sagae (2010)	—	92.1	—
	Koo et al. (2008) †	—	*93.16*	—
	Carreras et al. (2008) †	—	*93.5*	—
	Suzuki et al. (2008) †	—	*93.79*	—

4.6.3 PTB上的实验结果

为了考察联合模型在英语上的影响，我们在PTB做了实验。表4-8给出了实验结果。我们还列举了几个目前最好的结果。其中，†标记表示对应的模型使用了额外的资源，因此不能和我们的结果直接对比。由于英语的形态变化可以为词性标注提供很有帮助的信息，因此英语上的词性标注更加容易一些，词性准确率可以达到97%（汉语上只能达到约94%）。这样，和汉语相比，英语上级联方法中面临的错误级联问题少很多。然而，结果表明联合模型仍然可以帮助词性标注和句法标注，UAS提高0.3-0.4%，LAS提高0.4%，词性准确率提高0.5%。和汉语上的结果类似，句法分析时确定句法关系类型可以帮助句法结构的准确率，UAS提高约0.3-0.4%。我们的带句法关系的联合模型比Bohnet和Nivre (2012)的联合模型（不使用额外的资源）在UAS上高0.1%，在词性准确率上高0.3%。事实上，我们的联合模型在PTB上取得了最高的词性和句

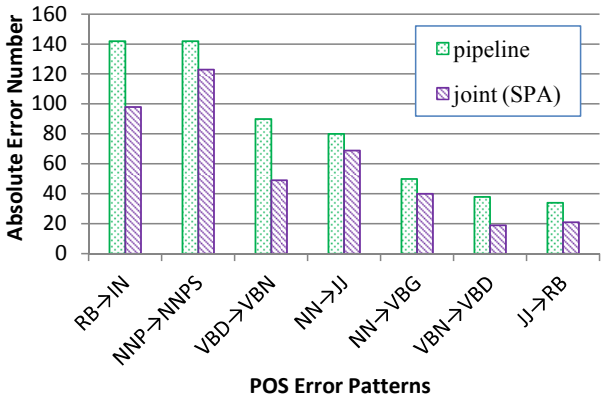


图 4-6 PTB测试集上的词性错误模式分析。

Fig.4-6 Statistics for different POS error patterns on the PTB test dataset.

法准确率。

为了跟深入的理解联合模型在哪些词性歧义上做的更好，我们比较了基于CRF的词性基准模型和无句法关系的联合模型在一些高频的词性错误模式上的表现。图4-6给出了结果。可以看到联合模型可以在很多词性错误模式上做的更好。比如，RB→IN（副词被错误识别为介词或从属性连词）的错误数从142降低为98，相当于31%的错误数下降。又比如，联合模型可以很好的消解词性歧义对{VBD, VBN}（过去时态的动词，动词过去分词）。

4.7 本章小结

本章提出一种面向词性标注和依存句法分析联合模型的分离被动进取训练算法（SPA）。和平均感知器算法（AP）和被动进取算法（PA）相比，SPA可以很自然的增大词性特征的权重，从而更好的平衡联合模型中词性特征和句法特征的消歧作用。

我们实现了第一个包含丰富特征的基于图的词性标注和依存句法分析联合模型。实验发现，我们的基于SPA训练算法的联合模型在汉语和英语数据上都可以取得最好的词性和句法准确率。

另外，我们系统的研究了句法关系类型的作用，并且发现确定依存结构的同时确定依存句法关系类型，可以较大幅度的提高依存结构的准确率。

我们希望在未来工作中，可以进一步从理论上研究SPA训练算法的收敛特性等，从而更深入的理解为什么SPA可以有效的训练联合模型，以及如何进一步改进SPA算法。另外，虽然我们的工作关注基于图的联合模型，我们相信我

们提出的SPA训练算法同样适用于基于转移的联合模型[13, 14]。因此，我们希望可以将SPA算法应用到基于转移的联合模型上，及其他同时解决多个相关的联合模型上。

第 5 章 基于准同步文法的多树库融合

5.1 引言

对于统计的数据驱动的分析模型而言，标注数据的规模很大程度上影响着分析结果的准确率。依存句法分析是一种结构化分类问题，比二元分类和序列标注问题更具挑战性，因此依存句法分析更容易受到数据稀疏问题的影响，树库规模对依存句法分析的准确率影响很大。然而，标注树库是一件艰巨的任务，通常需要耗费很大的人力和物力。首先，需要总结、提炼各种语言现象，然后制定一个合理标注规范；然后，需要对标注人员进行培训，使之熟悉标注规范，以保证标注一致性；标注过程中，需要不断对标注规范进行调整和补充，反复迭代；最后，还需要自动或人工检查标注结果，保证标注的质量。因此，单独一个树库的规模和覆盖的领域都相当有限。目前的研究现状暗示着单独在一个树库上训练出的句法分析的模型似乎很难再进一步提高句法分析的准确率。例如，Koo和Collins (2010)[22]和Li等(2011)[12]的工作表明，在基于图的依存句法分析模型中融入更复杂的三阶特征，对于句法分析的准确率影响很小。融入高阶特征后的句法模型更加复杂，可以融入更全局的特征，为什么句法准确率没有较大提高呢? 我们认为原因是，这些高阶特征在规模有限的树库上变得非常稀疏，训练集上出现过的高阶特征，在测试集上出现次数非常少，因此，这些高阶特征根本无法帮助模型。这种情况下，研究者们开始使用其他资源来增强现有的句法分析模型，如使用大规模的未标注数据[45, 46, 103, 104]，和大规模双语对齐文本或跨域言的树库[38, 105, 106]。这些研究表明合理使用其他资源，可以有效的帮助句法分析模型。

与此同时，我们认为多个单语树库的存在为我们的研究工作打开了另一扇门。表5-1中列举了几个公开的汉语树库[4, 107]。这些树库由不同的组织或机构标注，遵循不同的标注规范，面向不同的应用。在本文中，我们使用了前三个树库，分别是宾夕法尼亚大学标注的Chinese Penn Treebank 5.1 (CTB5)和6.0 (CTB6)[76]，以及哈尔滨工业大学社会计算与信息检索研究中心标注的Chinese Dependency Treebank (CDT)[108]。另外两种树库分别为台湾中央研究院标注的繁体树库Sinica treebank[109] 和清华大学标注的Tsinghua Chinese Treebank (TCT)[110]。我们希望以后可以进一步利用这两种树库，探讨同时使用多个树库对句法模型的影响。

表 5-1 现有的汉语树库。

Table.5-1 Several publicly available Chinese treebanks.

Treebanks	Number of words	Grammar
CTB5	0.51 million	Phrase structure
CTB6	0.78 million	Phrase structure
CDT	1.11 million	Dependency structure
Sinica	0.36 million	Phrase structure
TCT	about 1 million	Phrase structure

尽管各个树库遵循不同的标注规范，但是它们都根据人们对汉语语法的理解而标注，因此包含很多共性的标注结构。同时，不一致的标注结果应该也是有规律可循的。基于这种考虑，我们认为充分挖掘多个树库，对于提高句法分析准确率是很有潜力的。图5-1中给了一个实例，包含了CTB5和CDT中的标注结果，上面为CTB5的标注结构，下面为CDT中的标注结果。值得注意的是，CTB5遵循短语结构语法体系标注。我们使用规则将其转化为依存句法结构。由于两个树库中的词性标注体系也不相同，因此，我们在图中也给出了词性标注结果。这个例子显示，CTB5和CDT的标注依存结构的规范很类似，比如动宾关系，CTB5中为OBJ，CDT中对应为VOB；又比如并列结构中连词的处理，都附着于紧后面的成分，CTB5中为NMOD（名词的修饰语），而CDT中为LAD（左附着成分）。在这个例子上，CTB5和CDT的差异在于处理名词并列结构中两个并列成分的方式。CTB5中，使用第二个名词作为核心词，第一个名词修饰第二个名词（NMOD）；而在CDT中恰好相反，第一个名词作为核心词，第二个名词修饰第一个名词（COO）。多树库融合中，如果可以让模型学习到这种标注规范的不一致，就可以充分的利用多个树库来提高句法准确率。

对于多树库融合，一个很自然的想法就是树库转化（*treebank conversion*）。首先，将源树库中的标注结构转化为目标树库的标注结构，最重要的就是处理标注结构中的不一致；然后，将转化后的源树库和目标树库合并，形成一个更大规模的训练数据；最后，在合并后的树库上训练句法分析模型。由于训练数据增大，句法分析模型的准确率自然就会提高。然而，处理不同树库中标注结构的不一致现象，是一个非常困难的问题。标注结构的不一致种类繁多，形式多样，使用基于规则的转化方法根本无法处理。比如，CTB5和CDT中的很多不一致是跟一些具体的词相关的，也就是说标注规范中对于这些词给出了详细的

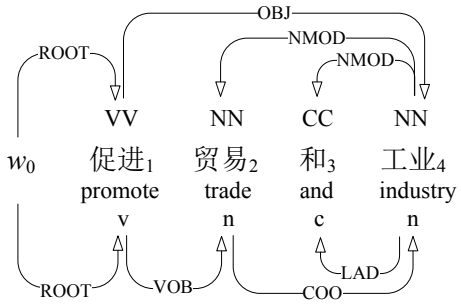


图 5-1 一个实例，上面为CTB5的标注，下面为CDT的标注。

Fig. 5-1 Example with annotations from CTB5 (upper) and CDT (under).

标注指南，而两个树库的标注规范制定者对这些词的理解有很大差异。另外，CTB5中对于复合名词短语（如“中华人名共和国国徽”）的标注是扁平的，即将复合名词短语中的词同时作为一个非终结符（如NP）的儿子节点。这样，短语结构转依存结构时，由于这种扁平结构中只能指定一个核心词。因此，CTB5的依存结构不会对复合名词短语内部的依存结构进一步细化。相反，CDT是一个纯粹的依存结构树库，其标注规范要求详细标注复合名词短语内部的依存结构。总之，我们认为尝试直接将一种标注规范转化为另一种特定的转化规范是极其困难的。Niu等(2009)[111]提出一种统计的树库转化方法，通过一个巧妙的策略降低转化后树库中包含的噪声，以提高句法分析的性能。我们在实验部分将我们的方法和他们的方法进行了间接对比，结果表明我们的方法更加有效。

本文提出一种简单而有效的框架，以充分挖掘多树库中的知识，提高句法分析的准确率。这个框架不直接尝试将一种标注规范转化为另一种标注规范，而是使用转化模式（*Transformation Patterns, TP*）对标注规范的不一致进行建模。然后基于TP，构成准同步文法（*Quasi-synchronous Grammar, QG*）特征，以增强依存句法分析模型。

5.2 相关工作

本文的工作主要受Jiang等(2009)的工作[112]和Smith和Eisner (2009)的工作[113]启发。Jiang等(2009)使用另一个大规模的分词和词性标注语料（人民日报，People Daily），以帮助CTB5上的分词和词性标注准确率。他们的方法和我们的框架类似。但是，他们的工作关注于序列标注中的标注不一致问题，而我们需要处理句法结构的标注不一致。因此，我们的工作更具挑战，也更有趣。Smith和Eisner (2009)使用QG特征以帮助句法移植（adaptation）和投影

（projection）问题。他们的第一部分工作和我们很相关，但是也存在几点重要的区别。他们主要研究如何将一个句法分析模型移植到另一种标注规范上。并且他们在一个树库上做模拟实验，基于两个启发式规则，人为的构造了一些标注不一致。不同的是，我们在两个现有的大规模汉语树库上做实验，利用QG特征以帮助一个很强的依存句法分析模型，进一步提高句法分析准确率。另外，我们对标注不一致进行深入建模，提出更加复杂的QG特征。这些QG特征可以很好的融入现有的基于图的依存句法分析模型中。总之，本文使用QG特征对结构化的标注不一致进行深入建模，有效地帮助依存句法分析模型，提高了句法分析准确率。

树库转化方面，前人主要关注于将一种标注体系的树库转化为另一种标注体系（如短语结构转为依存结构），然后在转化后的树库上开展研究[114–117]。这种工作不要求转化后的树库遵守某一种特定的标注规范。据我们所知，目前只有两个工作尝试将转化后树库和目标树库结合起来，以提高句法分析准确率。第一个是我们在中文信息学报发表的一个工作[118]。这个工作尝试统计和规则结合的方法，将短语结构的源树库CTB5转化为符合CDT标注规范的依存树库。首先，我们使用前人提出的头结点搜索规则将CTB5转化为一种依存树库。然后，我们使用一些基于人工观察提炼的规则处理了一部分有规律的标注不一致现象。最后，我们使用基于CDT训练出的依存分析器处理其他的不一致现象。实验发现，由于转化中的噪声，转化后的树库对于句法分析准确率的帮助并不大，只有在目标树库CDT规模很小时，才能较大幅度的提高句法分析的准确率。另一个工作是Niu等(2009)提出的基于统计的树库转化方法。他们尝试将依存结构的CDT树库转化为满足CTB5标注规范的短语结构树库。他们的转化方法包含三个步骤。首先，他们使用CTB5训练出一个短语结构句法分析器，称为CTB5-parser；然后，使用CTB5-parser分析整个CDT数据集，并且每个句子产生n-best结果；最后，对每个CDT中的句子，他们对n-best结果进行重排序，使用CTB5-parser对应概率较高、同时和CDT中依存结构较一致的短语句法树作为对应句子的句法结构。他们将转化后的CDT树库和CTB5合并，配合语料加权（corpus weighting）的方式，发现合并后的树库可以较大幅度的提高短语结构句法分析的准确率。他们的自动转化方法之所以有效，核心原因就是第三步中的重排序，可以有效地减少转化后树库的噪声，充分结合CTB5和CDT中人工标注结构中的句法知识。我们的方法不是直接将转化后的树库作为额外的训练数据，而是使用QG特征增强依存句法分析模型，从而柔和的学习标注规范中规律性的不一致现象。

我们的方法和栈学习（*stacked learning, SL*）也有联系。SL是一种有效融合不同模型的方法。前人将SL用于依存句法法分析，融合两种主流的依存句法分析模型，基于图（graph-based）的模型和基于转移（transition-based）的模型，以结合两种模型的优势[44, 119]。这种方法又称为融合（hybrid）方法。和本文中基于准同步文法的工作不同的是，SL在同一个树库上训练出两个句法分析器，因此不需要考虑标注规范不一致的问题。

5.3 依存分析基准模型

给定输入句子$\mathbf{x} = w_0 w_1 ... w_n$和其词性标注序列$\mathbf{t} = t_0 t_1 ... t_n$，依存句法分析的目的是建立一颗最优的依存句法树$\mathbf{d} = \{(h, m, l) : 0 \le h \le n, 0 < m \le n, l \in \mathcal{L}\}$，如图5-1所示。其中$(h, m, l)$表示一条从核心词（head, father）$w_h$到修饰词（modifier, child, dependent）w_m 的依存弧，并且句法关系类型为l，$\mathcal{L}$表示所有句法关系类型的集合。本章我们省略句法关系l，因为我们的工作关注无句法关系类型的依存句法分析。我们引入伪节点w_0以简化形式化描述，w_0用来总是指向整个句子的核心词。

依存句法分析存在两种主流的方法。基于转移的方法（transition-based）通过一个移动规约动作序列建立一棵依存树。依存句法分析的过程即搜索最优动作序列的过程[32, 33]。基于图的方法（graph-based）将依存句法分析建模为从一个有向完全图中搜索最大生成树的问题[19–22]。本文专注于基于图的模型，尝试使用QG特征提高基于图的模型的依存准确率。但是，我们的方法同样可以很自然的应用于基于转移的模型中。

为了保证基于动态规划解码算法的效率，基于图的方法需要做很强的独立性假设。在基于图的模型中，一棵依存树的分值分解为一些子树的分值之和。

$$\begin{aligned} \text{Score}_{\text{bs}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) &= \mathbf{w}_{\text{bs}} \cdot \mathbf{f}_{\text{bs}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \\ &= \sum_{p \subseteq \mathbf{d}} \text{Score}_{subtree}(\mathbf{x}, \mathbf{t}, p) \end{aligned} \tag{5-1}$$

其中，p表示一个子树，包含$\mathbf{d}$中一个或多个依存弧，$\mathbf{f}_{\text{bs}}(.)$表示基本句法特征（*basic syntactic features*），对应于QG句法特征（QG syntactic features）$\mathbf{f}_{\text{qg}}(.)$。近几年，研究者们尝试在依存句法分析模型中融入更复杂的子树，因此解码算法越来越复杂，同时句法准确率也不断提高。图5-2中列举了本文句法分析模型中采用的所有子树类型。我们没有使用三条依存弧构成的子树信息，即三阶句法特征。原因是Koo和Collins (2010)以及Li等(2011)的工作发现，三阶句法特征对句

法准确率影响不大。

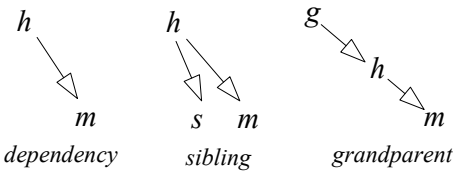


图 5-2 基于图的依存句法分析模型中使用到的子树类型。

Fig.5-2 Scoring parts used in our graph-based parsing models.

为了更好的考察QG特征对句法模型的影响，我们实现了三个基准依存句法分析模型。它们可以融入不同复杂度的子树，因此模型的复杂度和性能也不同。

- **一阶模型（The first-order model, O1）：** 即McDonald等(2005)提出的一阶模型[19]，模型中依存弧之间互相独立，只使用包含一条依存弧的子树。解码算法时间复杂度为$O(n^3)$。
- **只使用兄弟（sibling）子树的二阶模型（The second-order model using only sibling parts, O2sib）：** 对应McDonald和Pereira (2006)提出的二阶模型[20]，模型使用图5-2中的强两种子树：只包含一条依存弧的子树和包含两条兄弟弧的子树。解码算法的时间复杂度为$O(n^3)$。
- **二阶模型（The second-order model, O2）：** 对应Koo和Collins (2010)提出的二阶模型[22]，使用图5-2中的所有三种子树。解码算法的时间复杂度为$O(n^4)$。我们使用coarse-to-fine的策略[22]，在解码之前对依存弧进行剪枝，过滤掉边缘概率非常低的依存弧，这样可以大大加速解码过程。Li等(2011)的实验发现，这个二阶模型在CTB5上取得了最好的依存分析准确率[12]。

对于二阶模型，依存树的分值函数可以展开为：

$$\begin{aligned}\mathrm{Score}_{\mathrm{bs}}(\mathbf{x},\mathbf{t},\mathbf{d}) = &\sum_{\{(h,m)\}\subseteq\mathbf{d}} \mathrm{Score}_{\mathrm{dep}}(\mathbf{x},\mathbf{t},h,m) \\ &+ \sum_{\{(h,s),(h,m)\}\subseteq\mathbf{d}} \mathrm{Score}_{\mathrm{sib}}(\mathbf{x},\mathbf{t},h,s,m) \\ &+ \sum_{\{(g,h),(h,m)\}\subseteq\mathbf{d}} \mathrm{Score}_{\mathrm{grd}}(\mathbf{x},\mathbf{t},g,h,m)\end{aligned} \tag{5-2}$$

$$\begin{aligned}\mathrm{Score}_{\mathrm{dep}}(\mathbf{x},\mathbf{t},h,m) &= \mathbf{w}_{\mathrm{dep}}\cdot\mathbf{f}_{\mathrm{dep}}(\mathbf{x},\mathbf{t},h,m) \\ \mathrm{Score}_{\mathrm{sib}}(\mathbf{x},\mathbf{t},h,s,m) &= \mathbf{w}_{\mathrm{sib}}\cdot\mathbf{f}_{\mathrm{sib}}(\mathbf{x},\mathbf{t},h,s,m) \\ \mathrm{Score}_{\mathrm{grd}}(\mathbf{x},\mathbf{t},g,h,m) &= \mathbf{w}_{\mathrm{grd}}\cdot\mathbf{f}_{\mathrm{grd}}(\mathbf{x},\mathbf{t},g,h,m)\end{aligned} \tag{5-3}$$

其中$\mathbf{f}_{dep}(.)$、$\mathbf{f}_{sib}(.)$和$\mathbf{f}_{grd}(.)$为三种句法特征，分别对应图5-2中三种子树。我们采用Li等(2011)使用的句法特征[12]。和第三中表3-2中的特征集合相比，本章中使用的特征要简单一些，我们没有使用$t_{s\pm1}$和$t_{g\pm1}$相关的特征。对于O1和O2sib模型，公式5-2省略没有覆盖的子树对应的特征即可。

5.4 利用准同步文法特征的依存分析

Smith和Eisner (2006)针对机器翻译提出准同步文法（Quasi-synchronous Grammar, QG）的方法，从而允许两个语言对应句法结构更加灵活[120]。给定$\mathbf{x}'$和其句法树$\mathbf{d}'$，一个准同步文法QG定义了一种单语语法，这个单语语法给出了$\mathbf{x}'$的翻译的概率，记为：$p(\mathbf{x}, \mathbf{d}, \mathbf{a}|\mathbf{x}', \mathbf{d}')$ 其中$\mathbf{x}$表示翻译的句子，$\mathbf{d}$表示句子的句法结构，$\mathbf{a}$表示两个源语言和目标语言的对齐结果。在QG文法中，$\mathbf{d}$和$\mathbf{d}'$中的句法结构可以灵活的对齐，不受任何限制。因此，生成$\mathbf{d}$的过程可以用$\mathbf{d}$的任何一个子结构来指导。目前为止，QG被成功应用于多种任务，如双语词对齐[120]，机器翻译[121] 和句子简化[122]。

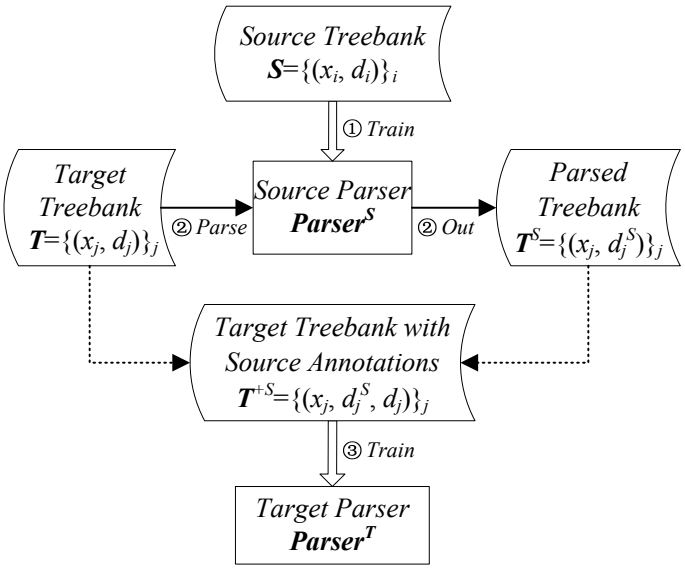


图 5-3 我们的方法的系统框架。

Fig.5-3 Framework of our approach.

本文基于QG的思想，利用单语多树库提高句法分析的准确率。核心思想是利用符合源树库标注规范的句法结构来指导目标句法树的建立。和机器翻译的过程不同，句法分析只考虑一个句子（$\mathbf{x} = \mathbf{x}'$），因此不需要考虑对齐$\mathbf{a}$的问题。图5-3给出了我们的系统框架。首先，我们使用源树库（*source treebank*）

训练出一个依存句法分析器，称为源端句法分析器（*source parser*）。然后，我们使用source parser分析整个目标树库（*target treebank*）。此时，目标树库中每个句子都包含两套句法结果：一种遵守源树库的标注规范并且可能含有噪声（因为source parser会产生错误结果），称之为源端句法结构（*source annotations*）另一种句法结构遵守目标树库的标注规范并且不包含噪声（人工标注结果），称为目标句法结构（*target annotations*）。这样，在训练和测试目标句法分析器（*target parser*）时，即target parser尝试建立符合目标树库标注规范的依存句法结构时，target parser可以把含噪声的源端句法结构作为一种指导信息，从而做出更好的决策。也就是说，在训练阶段，target parser尝试学习源端句法结构和目标句法结构之间的对应关系（包括不一致现象的对应关系）；而在测试集阶段，target parser根据源端句法结构和与学习到的对应规律，尝试建立更好的符合目标树库标注规范的句法树。这样，在增加了QG特征的句法模型中，一棵目标句法树的分值变为：

$$\begin{aligned}\text{Score}(\mathbf{x},\mathbf{t},\mathbf{d}',\mathbf{d}) =& \text{Score}_{\text{bs}}(\mathbf{x},\mathbf{t},\mathbf{d}) \\ &+\text{Score}_{\text{qg}}(\mathbf{x},\mathbf{t},\mathbf{d}',\mathbf{d})\end{aligned} \tag{5-4}$$

其中，第一部分和基准句法分析模型中的定义一致，第二部分分值由QG特征贡献，受到源端句法结构$\mathbf{d}'$影响。第二部分分值又可以分解为：

$$\text{Score}_{\text{qg}}(\mathbf{x},\mathbf{t},\mathbf{d}',\mathbf{d}) = \mathbf{w}_{\text{qg}} \cdot \mathbf{f}_{\text{qg}}(\mathbf{x},\mathbf{t},\mathbf{d}',\mathbf{d}) \tag{5-5}$$

其中$\mathbf{f}_{\text{qg}}(.)$表示QG特征集合。这些QG特征根据源端句法结构$\mathbf{d}'$，可以增大或者减小某些目标端的子树的分值，从而影响目标句法树的生成。以图5-1为例，我们使用CDT作为源端树库，CTB5作为目标树库。由于依存弧“和”$\curvearrowleft$“工业”在源端句法结构中存在，并且训练语料暗示源端和目标端标注规范对于名词并列结构中连词的处理方式一致，因此target parser可能会通过QG特征增大候选依存弧“和”$\curvearrowleft$“工业”的分值。类似的，因为“贸易”$\curvearrowright$“工业”在源端句法结构中存在，并且训练语料暗示源端和目标端标注规范对于名词并列结构中的两个并列名词构成的依存弧方向相反，因此target parser也可能增大候选依存弧“贸易”$\curvearrowleft$“工业”的分值。

为了对这种标注规范的一致和不一致现象进行建模，我们提出使用转化模式（Transformation Pattern, TP）来刻画源端和目标端句法结构的对应关系。图5-4给出了我们使用到的三种高频TP，分别对应模型中使用的三种子树。一个TP包含两个句法结构，一个为源端句法结构，另一个为目标端句法子树，并且刻画了目标端的句法子树对应哪种源端句法结构。从另一个角度看，一个TP刻画了一个源端句法结构转化为一个目标端的句法子树的过程。函

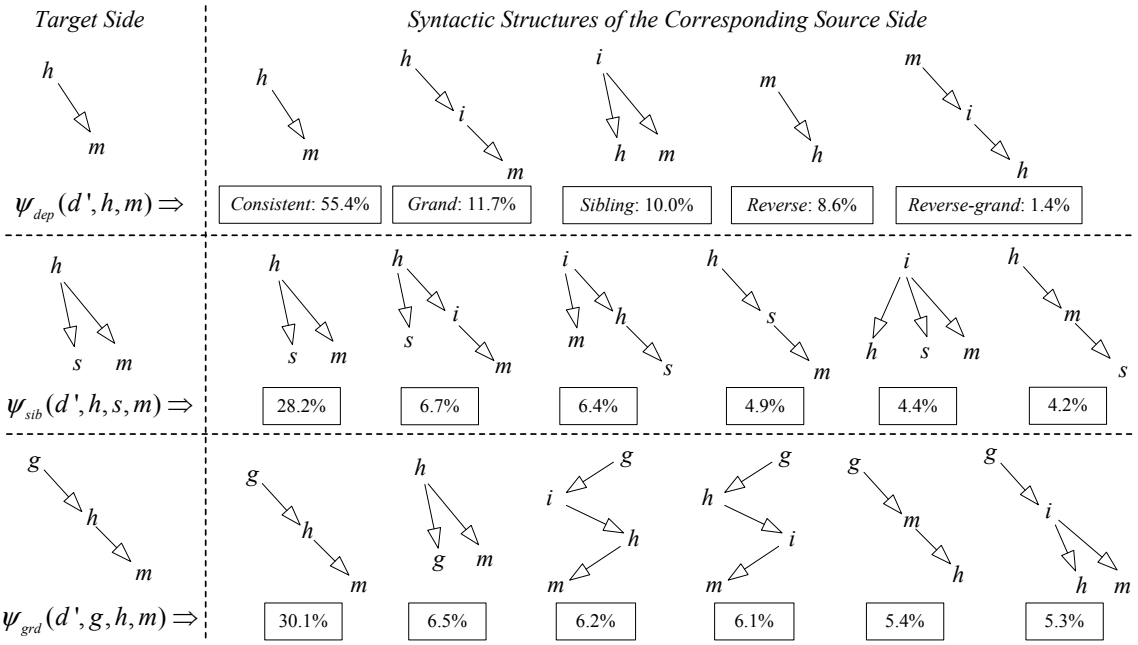


图 5-4 CDT作为源树库、CTB5作为目标树库时，统计出的最频繁的转化模式（TP）。

Fig.5-4 Most frequent transformation patterns (TPs) when using CDT as the source treebank and CTB5 as the target.

数$\psi_{\text{dep}}(.)$、$\psi_{\text{sib}}(.)$和$\psi_{\text{grd}}(.)$根据源端句法树$\mathbf{d}'$，分别返回三种目标端句法子树对应的源端句法结构的类型。每个源端句法结构下面的文本框中给出的百分比代表在训练语料T^{+S}中，对应模式所占的比例。

图中第一行给出了使用到的高频依存弧模式（dependency TP）。依存弧模式刻画的是一个目标端的候选依存弧$h \curvearrowright m$，在源端句法树$\mathbf{d}'$中，对应于哪种句法结构。可以看到，含噪音的源端句法结构和正确的目标端句法结构中，55.4%的依存弧是一致的(*"consistent"*)。我们只采用图中列出的5种模式。如果$h \curvearrowright m$对应的源端句法结构不属于这5种模式，$\psi_{\text{dep}}(\mathbf{d}', h, m)$返回一种特殊的类型*"else"*（占12.9%）。当然，我们也可以使用更加复杂的源端句法结构，如h作为m的曾祖父，但是这样的结构在训练语料中变得非常稀疏。

由于依存弧模式只能刻画只包含一条依存弧的目标端子树，我们尝试使用其他两种更复杂的模式。图5-4第二行和第三行分别给出了兄弟弧模式（sibling TP）和祖孙弧模式（grandparent TP）。我们只使用百分比高于1.0%的模式，对于其他情况，我们则返回*"else"*。这样，我们只保留了21种兄弟弧模式和22种祖孙弧模式。

表 5-2 QG特征集合，用以增强基准依存句法分析模型。

Table.5-2 QG features used to enhance the baseline parsing models.

$\mathbf{f}_{\text{qg-dep}}(\mathbf{x},\mathbf{t},\mathbf{d}',h,m)$	$\mathbf{f}_{\text{qg-sib}}(\mathbf{x},\mathbf{t},\mathbf{d}',h,s,m)$	$\mathbf{f}_{\text{qg-grd}}(\mathbf{x},\mathbf{t},\mathbf{d}',g,h,m)$
$\oplus dir(h,m)\circ dist(h,m)$	$\oplus dir(h,m)$	$\oplus dir(h,m)\circ dir(g,h)$
$\psi_{\text{dep}}(\mathbf{d}',h,m)\circ t_h\circ t_m$	$\psi_{\text{sib}}(\mathbf{d}',h,s,m)\circ t_h\circ t_s\circ t_m$	$\psi_{\text{grd}}(\mathbf{d}',g,h,m)\circ t_g\circ t_h\circ t_m$
$\psi_{\text{dep}}(\mathbf{d}',h,m)\circ w_h\circ t_m$	$\psi_{\text{sib}}(\mathbf{d}',h,s,m)\circ w_h\circ t_s\circ t_m$	$\psi_{\text{grd}}(\mathbf{d}',g,h,m)\circ w_g\circ t_h\circ t_m$
$\psi_{\text{dep}}(\mathbf{d}',h,m)\circ t_h\circ w_m$	$\psi_{\text{sib}}(\mathbf{d}',h,s,m)\circ t_h\circ w_s\circ t_m$	$\psi_{\text{grd}}(\mathbf{d}',g,h,m)\circ t_g\circ w_h\circ t_m$
$\psi_{\text{dep}}(\mathbf{d}',h,m)\circ w_h\circ w_m$	$\psi_{\text{sib}}(\mathbf{d}',h,s,m)\circ t_h\circ t_s\circ w_m$	$\psi_{\text{grd}}(\mathbf{d}',g,h,m)\circ t_g\circ t_h\circ w_m$
	$\psi_{\text{sib}}(\mathbf{d}',h,s,m)\circ t_s\circ t_m$	$\psi_{\text{grd}}(\mathbf{d}',g,h,m)\circ t_g\circ t_m$

基于这些模式，我们提出QG特征以增强基准的依存句法分析模型，如表5-2所示。包括三类QG特征，分别对应三种TP。我们将TP的类型和相关词语和词性结合起来，这样模型可以根据更细粒度的上下文信息做出更好的决策。$dir(h,m)$表示依存弧(h,m)的方向；$dist(h,m)$表示依存弧的距离$|h-m|$；$\oplus dir(h,m)\circ dist(h,m)$表示对应列中的特征和$dir(h,m)\circ dist(h,m)$结合起来形成新的特征。这样，由QG特征贡献的分值可以展开为：

$$\begin{aligned}\text{Score}_{\text{qg}}(\mathbf{x},\mathbf{t},\mathbf{d}',\mathbf{d}) = &\sum_{\{(h,m)\}\subseteq\mathbf{d}}\text{Score}_{\text{qg-dep}}(\mathbf{x},\mathbf{t},\mathbf{d}',h,m)\\ &+\sum_{\{(h,s),(h,m)\}\subseteq\mathbf{d}}\text{Score}_{\text{qg-sib}}(\mathbf{x},\mathbf{t},\mathbf{d}',h,s,m)\\ &+\sum_{\{(g,h),(h,m)\}\subseteq\mathbf{d}}\text{Score}_{\text{qg-grd}}(\mathbf{x},\mathbf{t},\mathbf{d}',g,h,m)\end{aligned} \tag{5-6}$$

$$\begin{aligned}\text{Score}_{\text{qg-dep}}(\mathbf{x},\mathbf{t},\mathbf{d}',h,m) &= \mathbf{w}_{\text{qg-dep}}\cdot\mathbf{f}_{\text{qg-dep}}(\mathbf{x},\mathbf{t},\mathbf{d}',h,m)\\ \text{Score}_{\text{qg-sib}}(\mathbf{x},\mathbf{t},\mathbf{d}',h,s,m) &= \mathbf{w}_{\text{qg-sib}}\cdot\mathbf{f}_{\text{qg-sib}}(\mathbf{x},\mathbf{t},\mathbf{d}',h,s,m)\\ \text{Score}_{\text{qg-grd}}(\mathbf{x},\mathbf{t},\mathbf{d}',g,h,m) &= \mathbf{w}_{\text{qg-grd}}\cdot\mathbf{f}_{\text{qg-grd}}(\mathbf{x},\mathbf{t},\mathbf{d}',g,h,m)\end{aligned} \tag{5-7}$$

可以看到，QG特征贡献的分值和和基准模型中的分值定义（见公式5-2）是一一对应的，因此可以直接使用基准模型中的解码算法，区别仅在于在融合基准特征的同时，额外增加相应的QG特征。

5.5 实验和结果分析

我们使用汉语依存树库（Chinese Dependency Treebank, CDT）[108]作为源树库。CDT包含来自人民日报（People Daily）90年代的约60,000个句子。对于目标树库，我们使用了两个被广泛用于依存句法分析的树库：Penn Chinese Treebank 5.1（CTB5）和6.0（CTB6）[76]。CTB中的文本来源包含新华社、香港新闻和台湾光华杂志。为了和前人的结果比较，对于CTB5，我们采用Zhang和Clark的头结点搜索规则，将短语结构转化为依存结构，并且采用Duan等的数据划分；对于CTB6，我们采用ConLL 2009公开评测数据中的设置[8]。

由于CDT和CTB采用不同的词性标记集合，并且这两个词性标记集合无法互相转化。为了解决这个问题，我们使用大规模的词性标注语料Peopel Daily corpus （PD）[1]，训练了一个基于平均感知器的词性标注模型。我们使用这个词性标注模型对源树库和目标树库标注词性。源端句法分析器使用这套统计的词性标注结果，在源树库上训练，然后分析整个目标树库。PD中包含大约30,000句子，7000,000词语，来自人民日报1998年上半年。事实上，CDT和CTB的分词标注规范也有差异，但是我们的工作不考虑这个问题，直接基于各自的正确分词进行处理。

表5-3中总结了本工作中采用的数据资源。其中，CTB5X是根据Niu等的设置重新对CTB5进行数据划分。我们在这个数据集上做了实验，以便和他们的树库转化方法进行间接的比较。（见表5-10）。

表 5-3 本章使用的数据汇总（句子数）。

Table.5-3 Data used in this work (in sentence number).

Corpus	Train	Dev	Test
PD	281,311	5,000	10,000
CDT	55,500	1,500	3,000
CTB5	16,091	803	1,910
CTB5X	18,104	352	348
CTB6	22,277	1,762	2,556

评价指标： 我们无句法关系的附着分值（*unlabeled attachment score, UAS*）作为主要的评价指标。我们也给出了根节点准确率（*root accuracy, RA*）和完全

[1]http://icl.pku.edu.cn/icl_groups/corpustagging.asp

匹配率（*complete match rate, CM*），以便和前人的结果进行更细致的比较。然而由于因为测试一般只包含约几百或者2,000左右个句子，根节点准确率和完全匹配率不是很可靠，所有和句法相关的评价指标都忽略标点符号。我们使用Dan Bikel开发的randomized parsing evaluation comparator做显著性检验[2]。对于词性标注，我们使用标准的词性准确率（*POS tagging accuracy, TA*）来评价。

对于所有本文中使用的模型，包括词性标注和句法分析模型，我们都采用平均感知器（averaged perceptron, AP）训练特征权重[26]。对每个模型，我们训练时迭代10次，然后选择在开发集上性能最好的模型。

5.5.1 前期准备工作

本小节介绍我们如何将源端标注结果映射到目标树库中。也就是说，如何在目标树库中产生符合源端标注规范的含噪声的句法结构。首先，我们使用PD训练出一个基于统计的词性标注器，记为$Tagger^{PD}$。我们采用Zhang和Clark提出的面向汉语的词性特征集合[77]。在PD的测试集上，词性准确率为98.30%。

第二步，我们使用$Tagger^{PD}$对所有的树库，包括CDT、CTB5和CTB6 词性标注。这样所有的树库都包含了一套公共的符合PD标注规范的词性结果。

第三步，我们基于这套公共的词性结果，使用CDT训练一个二阶的基准句法分析器，记为$Parser^{CDT}$，即源端句法分析器。在CDT测试集上，$Parser^{CDT}$的UAS为84.45%。

第四步，我们基于公共的词性结果，使用$Parser^{CDT}$对所有的目标树库进行句法分析。这样，目标树库CTB5和CTB6中就包含了一套符合CDT标注规范的含噪声的句法结构。

5.5.2 CTB5作为目标树库

本小节，我们采用CDT作为源树库，CTB5为目标树库，希望通过CDT提高句法分析模型在CTB5上的准确率。我们尝试了两种情景，从而更深入的理解QG特征的作用。

第一种情景中，句法分析模型使用CTB5中人工标注的正确词性。这种情况下，句法分析模型就不会受到词性标注引入的错误蔓延。表5-4给出了实验结果。Li11表示Li等(2011)的基于图的二阶模型[12]；Z&N11表示Zhang和Nivre

[2]http://www.cis.upenn.edu/~dbikel/software.html

表 5-4 正确词性下，CTB5测试集上的结果

Table. 5-4 Parsing accuracy (UAS) comparison on CTB5-test with gold-standard POS tags.

Models	without QG	with QG
O2	86.13	86.44 (+0.31, $p = 0.06$)
O2sib	85.63	86.17 (+0.54, $p = 0.003$)
O1	83.16	84.40 (+1.24, $p < 10^{-5}$)
Li11	86.18	—
Z&N11	86.00	—
H&S10	85.20	—
Z&C08	85.77	—

(2011)的融入丰富特征的基于转移的模型[35]。可以看到，我们的二阶基准模型达到了最好的句法准确率。对于一阶模型（O1），QG特征可以较大幅度的提高句法准确率，并且显著性检验说明这个提高是显著的。对于更复杂的O2sib模型，QG特征带来的准确率上升幅度较小，并且显著性降低。对于最复杂的O2模型，QG特征的贡献最小，并且显著性检验表明准确率的提升不明显。这种情况说明了，当使用目标树库的正确词性时，QG特征只有在句法模型比较简单时，才能较大提高句法准确率；而当句法模型中包含的特征很丰富时，QG特征的作用变得很有限。我们认为，这种现象是由于词性对于句法模型非常关键，当目标树库提供正确词性时，词性信息可以为句法模型提供很可信的证据，因而复杂的二阶模型只基于正确词性就可以很好的作出决策，QG特征的作用就相对变小了。

第二种情景中，我们考虑更加实际的情况，即目标树库不提供正确的词性。我们在CTB5训练集上训练一个基于平均感知器的词性标注模器，然后使用这个词性标注器在开发集和测试集上产生自动词性标注结果。测试集上的词性准确率为93.88%。我们采用10份交叉验证的方式产生CTB5训练集上的自动词性结果。当然，我们也可以使用$Tagger^{PD}$产生的自动词性结果，但是这样我们就无法和前人的结果进行公平比较。并且，由于CTB5和PD在文本来源和分词标注的不同，这样做可能会导致句法准确率下降。

表5-5给出了当句法模型采用自动词性时的实验结果。可以看到，QG特征可以较大幅度的提高所有基准句法模型（O1、O2sib和O2）的句法准确率。尤其对于最简单的O1模型，提高的幅度最大，达到2.31%。显著性检验结果表明，所有的准确率提高都是统计显著的（$p < 10^{-5}$）。和采用正确词性时相比，句法

表 5-5 自动词性下，CTB5测试集上的结果

Table.5-5 Parsing accuracy (UAS) comparison on CTB5-test with automatic POS tags.

Models	without QG	with QG
O2	79.67	81.04 (+1.37)
O2sib	79.25	80.45 (+1.20)
O1	76.73	79.04 (+2.31)
Li11 joint	80.79	—
Li11 pipeline	79.29	—

模型采用自动词性时，由于词性中含有噪声，因此无法为句法模型提供可信的证据。这样，QG特征表示源树库中的知识，可以有效地帮助句法模型做出更好的决策。Li等(2011)发现词性标注和依存句法分析联合模型可以提高句法分析准确率[12]。我们的使用QG特征的二阶模型比联合模型的句法准确率高0.25%。并且，我们相信QG特征同样可以有效帮助联合模型，从而进一步提高句法准确率。我们希望在未来工作中研究这个想法。

5.5.3 错误分析：基于自动词性的二阶模型

为了更深入的理解QG特征如何帮助句法模型，我们深入比较和分析了二阶基准模型和增加QG特征的模型在自动词性下的结果。

表 5-6 自动词性下，CTB5测试集上不同特征集合的贡献

Table.5-6 Feature ablation for Parser-O2 on CTB5-test with automatic POS tags.

Setting	UAS	CM	RA
$\mathbf{f}_{bs}(.)$	79.67	26.81	73.82
$\mathbf{f}_{qg}(.)$	79.15	26.34	74.71
$\mathbf{f}_{bs}(.) + \mathbf{f}_{qg}(.)$	81.04	29.63	77.17
$\mathbf{f}_{bs}(.) + \mathbf{f}_{qg-dep}(.)$	80.82	28.80	76.28
$\mathbf{f}_{bs}(.) + \mathbf{f}_{qg-sib}(.)$	80.86	28.48	76.18
$\mathbf{f}_{bs}(.) + \mathbf{f}_{qg-grd}(.)$	80.88	28.90	76.34

表5-6分析了不同特征集合对于句法模型的贡献。前三行分析了基本特征$\mathbf{f}_{bs}(.)$和QG特征$\mathbf{f}_{qg}(.)$的作用。当只使用表5-2中列出的简单的QG特征时，句法模型的准确率就可以达到使用所有基本特征时的准确率（对应于基准模型）。

这说明了QG特征非常有效。未来工作中，我们希望考虑更加复杂的QG特征，进一步提高句法模型的准确率。进而，同时使用基本特征和QG特征，可以较大幅度的提高句法准确率。后三行尝试分析三种QG特征的作用。每次我们在基本特征的基础上，增加一类QG特征，考察句法模型的准确率。结果表明，三类QG特征的作用是相当的，可以达到类似的准确率。

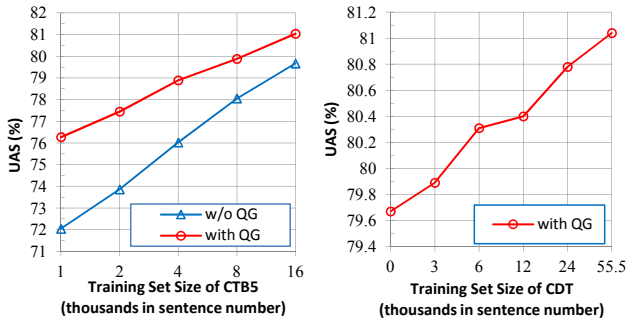


图 5-5 当源树库和目标树库规模变化时，CTB5测试集上的句法准确率变化。

Fig.5-5 Parsing accuracy curve on CTB5-test when the scale of source and target treebanks varies.

图5-5考察了当源树库CDT和目标树库CTB5的规模变化时，QG特征对句法模型的贡献。在左图中，我们考察当目标树库规模变化时，QG特征对于句法模型的影响。横坐标表示使用多少CTB5训练数据来训练目标句法分析器，“16”表示使用所有CTB5训练数据。源端句法分析器$Parser^{CDT}$采用整个CDT训练数据训练。可以看到，源树库规模越小，QG特征带来的准确率提升越大。这种现象是很合理的。更重要的是，图中的曲线暗示着当使用QG特征时，在16,000句目标树库上训练出的句法分析器的准确率和在两倍规模（32,000句）目标树库上训练出的基准句法分析器的准确率相当。这个结果是非常可喜的。

右图中，我们考察源树库规模变化时QG特征对句法准确率的影响。横坐标表示使用多少CDT训练集来训练源端句法分析器$Parser^{CDT}$，“55.5”表示使用整个CDT训练集。我们使用整个CTB5训练集训练目标句法分析器。显然，图中的曲线说明了当源树库规模变化时，QG特征对句法模型的帮助越大。这个结果也是合理的。源树库规模越大，源端句法分析器的准确率越高，那么QG特征的越可信，那么就可以为目标端句法分析模型提供更有用的信息。

图5-6考察了QG特征在不同跨度（$|h-m|$）的依存弧上的作用。和前人的结果一致，无论是基准句法模型还是使用了QG特征的句法模型，随着依存弧跨度的增大，对应依存弧的F值急剧下降。这说明句法模型可以很好的处理近距离的句法搭配，但是远距离搭配对于句法分析模型而言则非常困难。从图中的曲线可以看出，QG特征可以帮助句法模型缓解这个问题。QG特征在距

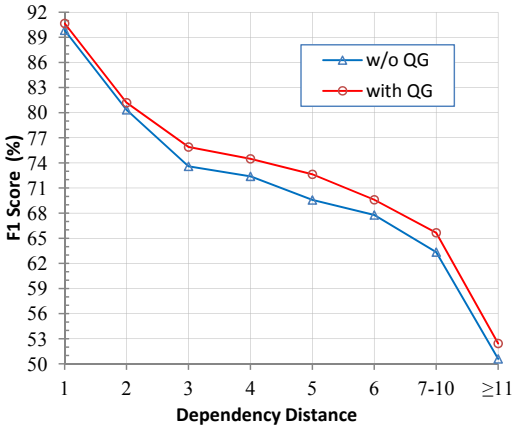


图 5-6　QG特征在不同跨度的依存弧上的作用

Fig.5-6　Effect of QG features on dependencies of different distances.

离为1或2的依存弧上的作用比较小，F值只提高了大约0.8%。但是，对于长距离搭配（$|h-m| \geq 3$），F值有大约1.8%-3.1%的提高。因此，我们可以得出结论：QG特征可以对传统的句法分析模型在长距离搭配上提供大的帮助。

表 5-7　QG特征在不同依存模式上的作用。

Table.5-7　Detailed effect of QG features on different dependency patterns.

Dependency	w/o QG	+QG	Descriptions
NN ↶ NN	858	-78	noun modifier or coordinating nouns
VV ↷ VV	777	-41	object clause or coordinating verbs
VV ↶ VV	570	-38	subject clause
VV ↷ NN	509	-79	verb and its object
w_0 ↷ VV	357	-57	verb as sentence root
VV ↶ NN	328	-32	attributive clause
P ↶ VV	278	-37	preposition phrase attachment
VV ↷ DEC	233	-33	attributive clause and auxiliary DE
P ↷ NN	175	-35	preposition and its object

表5-7分析了QG特征对于一些具体的搭配模式的作用。一个搭配模式“VV ↷ NN”表示核心词词性为动词（VV），修饰词词性为名词（NN）的方向朝右的依存弧，其中词语的词性根据人工标注的正确词性。表中“w/o QG”对应的列表示基准句法分析模型在对应搭配模式上的错误数；“+ QG”对应的列表示使用了QG特征的句法分析模型相对于基准模型在对应搭配模式上错误数的改变量。

我们只列出了该变量绝对值大于30的搭配模式。可以看出，QG特征对于各种搭配结构都有帮助，可以有效减少在这些结构上的错误数。

5.5.4 CTB6作为目标树库

我们以CDT作为源树库，CTB6作为目标树库展开进行实验，进一步验证我们提出的方法的有效性。和CTB5相比，CTB6的规模更大，并且采用了更加细致的短语转依存的规则[8]。简单起见，我们直接采用在CTB5数据集上调试出来的转化模式和QG特征集合。表5-8给出了目标树库提供正确词性时的实验结果。实验结果和CTB5的情况非常类似。

表 5-8 正确词性下，CTB6测试集上的结果

Table. 5-8 Parsing accuracy (UAS) comparison on CTB6-test with gold-standard POS tags.

Models	without QG	with QG
O2	88.32	88.61 (+0.29, $p = 0.04$)
O2sib	87.59	88.20 (+0.61, $p < 10^{-4}$)
O1	85.79	86.77 (+0.98, $p < 10^{-5}$)

表 5-9 自动词性下，CTB6测试集上的结果

Table. 5-9 Parsing accuracy (UAS) comparison on CTB6-test with automatic POS tags.

Models	without QG	with QG
O2	83.23	84.33 (+1.10)
O2sib	82.87	84.11 (+1.37)
O1	80.29	82.76 (+2.47)
Bohnet[71]	82.68	—
Che等[75]	82.11	—
Gesmundo等[74]	81.70	—

表5-9给出了当采用自动词性时的实验结果。和CTB5上的情况类似，实验结果表明，当句法模型采用自动词性时，QG特征可以较大幅度的提高句法准确率。显著性检验表明所有准确率提高都是统计显著的（$p < 10^{-5}$）。在最下面的三行，我们给出了参加CoNLL 2009公开评测任务中最好的三个系统在

汉语上的准确率。官方结果只提供带句法关系的附着分值（labeled attachment score, LAS）。我们下载了组织方提供的参赛单位的提交结果[3]，使用eval.pl评测工具，得出表中的UAS指标。可以看到，我们的基准句法分析模型比当时最好的系统的准确率高0.55%，使用QG特征的模型则高了1.65%。但是需要指出的是，这么直接比较并不公平，因为当时参赛单位采用的官方提供的词性结果只有92.38%的词性准确率，而我们的词性标注模型的准确率为94.08%。

表 5-10 CTB5X测试集上的结果比较

Table.5-10 Parsing accuracy (UAS) comparison on the test set of CTB5X.

Models	baseline	with another treebank
Ours	84.16	86.67 (+2.51)
GP [111]	82.42	84.06 (+1.64)

5.5.5 与树库转化方法的比较

Niu等(2009)提出一种基于统计的自动树库转化方法[111]，将依存结构的CDT转化为符合CTB规范的短语结构树库，并将转化后的树库作为额外的数据，通过语料加权的策略，将转化后树库和CTB5X合并，从而提高短语结构句法分析器的F值。为了和他们的方法进行比较，我们使用同样的头结点搜索规则，将他们在CTB5X测试集上的输出的短语结构树转化为依存结构，然后和我们的结果进行比较。[4] 表5-10给出了比较结果。Niu等(2009)使用了两种短语结构句法分析模型，一种是基于最大熵的生成模型（Generative Parser, GP）[123]。表中给出的就是GP的结果。另一种是更复杂但是准确率更好的重排序模型（Reranker Parser, RP）[50]。牛正雨博士没有找到RP对应的结果，所以我们没有在表中给出。Niu等(2009)的结果表明树库转化方法在GP上F值提高1.1%，在RP上提高1.1%。详细信息请参考他们论文中的表5和表6[111]。表中的结果暗示着我们的方法对句法准确率的提高更大。我们希望在未来工作中对两种方法进行更深入、更直接的比较。

[3]http://ufal.mff.cuni.cz/conll2009-st/results/results.php

[4]在此我们感谢牛正雨博士找到并共享他的实验结果。

5.6 本章小结

本章提出一种简单而有效的多树库融合方法，充分利用标注规范不同的多个单语树库，以提高句法分析准确率。我们设计了丰富的转换模式（TP）来刻画不同标注规范间的对应规律，然后基于这些TP形成QG特征。QG特征用来指导句法模型做出更好的决策，并且可以很自然的融入到基于图的句法分析解码算法中。我们使用CDT作为源树库，CTB5和CTB6作为目标树库，进行了大量实验。实验结果表明，我们的方法可以充分利用源树库的知识，从而提高句法模型在目标树库上的准确率。我们在CTB5和CTB6上都取得了目前最好的准确率。我们又做了细致的错误分析以深入理解QG特征的作用。

结 论

在自然语言处理领域，句法分析处于非常关键的位置。以序列形式输入文本，通过句法分析，转化为树状结构，从而刻画句子内部词语之间的句法组成方式和搭配关系。句法分析可以为语义分析、机器翻译、自动问答、信息抽取等上层应用提供有效地帮助。在不同的语法体系中，依存语法由于形式简单、易于标注、便于应用等特点，逐渐受到自然语言处理领域研究者的重视。近几年，计算自然语言学习国际会议（CoNLL）多次组织以依存句法分析为核心的评测任务，同时发布了多种语言的依存树库作为公开评测数据，推动了依存句法分析的快速发展。依存句法分析在提高依存句法分析效率、提高依存句法分析准确率、针对具有形态变化丰富特征的语言的处理、针对具有非投影结构特征的语言的处理、与词法分析的互动等方面取得了较大的发展。

然而，由于汉语存在分词歧义大、词语形态变化信息少等特点，导致汉语依存句法分析和其他西方语言如英语相比，在准确率上还存在较大差距。本文针对汉语的特点，对基于图的依存句法分析方法进行了深入的改进，提高了汉语依存句法分析的效率和准确率。本文的主要研究内容和成果可概括如下：

首先，针对前人提出的面向高阶依存句法分析模型的动态规划解码算法时间复杂度高的问题，本文提出了基于柱搜索的近似解码算法，一方面允许模型可以方便的融入丰富的高阶句法子树特征，另一方面保证较低的时间复杂度，从而提高依存句法分析的速度。我们实现的基于柱搜索的高阶依存句法分析系统参加了CoNLL 2009年多语依存句法分析和语义角色标注联合评测任务，在21家国内外参赛单位中，名列依存句法分析子任务3名。进而，我们针对汉语的特点，提出一种利用标点符号进行长句切分的二阶段依存句法分析方法，进一步提高依存句法分析模型处理长句时的效率。实验证明，这种方法可以大幅度提高依存句法分析的速度，长句子的句法分析准确率也有提高。

其次，本文提出并深入系统的研究了汉语词性标注和依存句法分析联合模型。我们扩展了前人提出的面向依存句法分析的解码算法，提出了相应的面向联合模型的基于动态规划的解码算法。并且，为了解决联合解码算法的时间复杂度过高的问题，我们又提出了一种有效地基于边缘概率的词性裁剪方法。实验结果表明联合模型可以提高词性和句法准确率。细致的错误分析表明联合模型可以更好地消解句法敏感的词性歧义，而正确的消解这些歧义可以进一步帮助句法分析。

再次，本文提出了面向词性标注和依存句法分析联合模型的分离被动进取训练算法（SPA）。和传统的平均感知器算法和被动进取算法相比，SPA算法可以很自然的增大词性特征的权重，从而更好的平衡联合模型中词性特征和句法特征的消歧作用。我们实现了第一个包含丰富特征的基于图的词性标注和依存句法分析联合模型。实验发现，我们提出的基于SPA训练算法的联合模型在汉语和英语数据上可以取得很高的词性和句法准确率。另外，我们系统的研究了句法关系类型的作用，并且发现确定依存结构的同时确定依存句法关系类型，可以提高依存结构的准确率。

最后，本文了提出一种简单而有效的多树库融合方法，充分利用标注规范不同的多个单语树库，以提高句法分析准确率。我们设计了丰富的转换模式来刻画不同标注规范间的对应规律，然后基于这些转转模式形成准同步文法特征。准同步文法特征用来指导句法模型做出更好的决策，并且可以很自然的融入到基于图的句法分析解码算法中。实验结果表明，我们的方法可以充分利用源树库的知识。从而提高句法模型在目标树库上的准确率。我们在多个汉语树库上都取得了当时最好的准确率。我们又做了细致的错误分析以深入理解准同步文法特征的作用。

需要说明的是，虽然本文的工作主要针对汉语依存句法分析分析，并且实验语料也基本采用汉语依存句法树库数据，但是本文提出的方法没有对特定语言的局限性，是语言无关的。基于柱搜索的高阶依存句法分析方法是语言无关的，同时基于标点的长句分短句策略也可以尝试应用于其他语言；词性标注和依存句法分析联合模型及SPA训练算法也是语言无关的技术，可以用来缓解词性标注对依存句法分析带来的错误蔓延问题，同时提高词性标注的准确率；如果其他语言拥有多个句法树库，那么也可以利用我们提出的基于准同步文法的多树库融合方法，提高依存句法分析准确率。

综上，本文在汉语依存句法分析上做了一些尝试并取得了一些初步的成果，然而，依存句法分析作为人工智能和自然语言处理领域的一个重要而具有挑战的课题，目前的研究还远远不够。因此需要我们在已有的工作基础上继续深入研究，尤其是提高面对海量的、灵活多变的、不规范的互联网数据时的依存句法分析速度和准确率。我们期待依存句法分析技术快速发展，为上层应用提供可靠的精准的分析结果，以推动科学技术和人类社会的发展！

参考文献

[1] Collins M. Head-driven statistical models for natural language parsing[D]. University of Pennsylvania, 1999.

[2] Mel'čuk I A. Dependency Syntax: Theory and Practice[M]. State University of New York Press, 1988.

[3] Pollard C, Sag I A. Head-Driven Phrase Structure Grammar[M]. Chicago: University of Chicago Press and Stanford: CSLI Publications, 1994.

[4] 马金山. 基于统计方法的汉语依存句法分析研究[D]. 哈尔滨工业大学, 2007.

[5] Che W, Spitkovsky V, Liu T. A Comparison of Chinese Parsers for Stanford Dependencies[C]. Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics. 2012:11–16.

[6] 车万翔. 基于核方法的语义角色标注研究[D]. 哈尔滨工业大学, 2008.

[7] 李军辉. 中文句法语义分析及其联合学习机制研究[D]. 苏州大学, 2010.

[8] Hajič J, Ciaramita M, Johansson R, et al. The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages[C]. Proceedings of CoNLL 2009. 2009.

[9] Buchholz S, Marsi E. CoNLL-X Shared Task on Multilingual Dependency Parsing[C]. In Proc. of CoNLL. 2006:149–164.

[10] Nivre J, Hall J, Kübler S, et al. The CoNLL 2007 Shared Task on Dependency Parsing[C]. Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007. 2007:915–932.

[11] Surdeanu M, Johansson R, Meyers A, et al. The CoNLL-2008 Shared Task on Joint Parsing of Syntactic and Semantic Dependencies[C]. CoNLL-2008. 2008.

[12] Li Z, Zhang M, Che W, et al. Joint Models for Chinese POS Tagging and Dependency Parsing[C]. EMNLP 2011. 2011:1180–1191.

[13] Hatori J, Matsuzaki T, Miyao Y, et al. Incremental Joint POS Tagging and Dependency Parsing in Chinese[C]. Proceedings of 5th International Joint Conference on Natural Language Processing. 2011:1216–1224.

[14] Bohnet B, Nivre J. A Transition-Based System for Joint Part-of-Speech Tagging and Labeled Non-Projective Dependency Parsing[C]. Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. 2012:1455–1465.

[15] Hatori J, Matsuzaki T, Miyao Y, et al. Incremental Joint Approach to Word Segmentation, POS Tagging, and Dependency Parsing in Chinese[C]. Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics. 2012:1045–1053.

[16] Li Z, Zhou G. Unified Dependency Parsing of Chinese Morphological and Syntactic Structures[C]. Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. 2012:1445–1454.

[17] Qian X, Liu Y. Joint Chinese Word Segmentation, POS Tagging and Parsing[C]. Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. 2012:501–511.

[18] Kubler S, McDonald R, Nivre J. Dependency Parsing (Synthesis Lectures On Human Language Technologies)[M]. Morgan and Claypool Publishers, 2009.

[19] McDonald R, Crammer K, Pereira F. Online Large-Margin Training of Dependency Parsers[C]. Proceedings of ACL 2005. 2005:91–98.

[20] McDonald R, Pereira F. Online Learning of Approximate Dependency Parsing Algorithms[C]. Proceedings of EACL 2006. 2006.

[21] Carreras X. Experiments with a Higher-Order Projective Dependency Parser[C]. Proceedings of EMNLP/CoNLL. 2007:141–150.

[22] Koo T, Collins M. Efficient Third-Order Dependency Parsers[C]. Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. 2010:1–11.

[23] Eisner J. Bilexical Grammars and a Cubic-Time Probabilistic Parser[C]. Proceedings of the 5th International Workshop on Parsing Technologies (IWPT). 1997:54–65.

[24] Eisner J. Bilexical Grammars and Their Cubic-Time Parsing Algorithms[M]. . Bunt H C, Nijholt A. Advances in Probabilistic and Other Parsing Technologies. Kluwer Academic Publishers, 2000:29–62.

[25] McDonald R. Discriminative Training and Spanning Tree Algorithms for Dependency Parsing[D]. University of Pennsylvania, 2006.

[26] Collins M. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms[C]. Proceedings of EMNLP 2002. 2002.

[27] Crammer K, Dekel O, Keshet J, et al. Online passive aggressive algorithms[C]. Procedings of NIPS 2003. 2003.

[28] Crammer K, Singer Y. Ultraconservative Online Algorithms for Multiclass Problems[J]. Journal of Machine Learning Research, 2001, 3.

[29] Smith D A, Eisner J. Dependency Parsing by Belief Propagation[C]. Proceedings of EMNLP 2008. 2008:145–156.

[30] Riedel S, Clarke J. Incremental integer linear programming for non-projective dependency parsing[C]. In EMNLP. 2006:129–137.

[31] Martins A, Smith N, Xing E. Concise Integer Linear Programming Formulations for Dependency Parsing[C]. Proceedings of ACL/IJCNLP 2009. 2009:342–350.

[32] Yamada H, Matsumoto Y. Statistical dependency analysis with support vector machines[C]. Proceedings of IWPT 2003. 2003:195–206.

[33] Nivre J. An Efficient Algorithm for Projective Dependency Parsing[C]. Proceedings of the 8th International Workshop on Parsing Technologies (IWPT). 2003:149–160.

[34] Huang L, Sagae K. Dynamic Programming for Linear-Time Incremental Parsing[C]. Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. 2010:1077–1086.

[35] Zhang Y, Nivre J. Transition-based Dependency Parsing with Rich Non-local Features[C]. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. 2011:188–193.

[36] Zhang Y, Clark S. A Tale of Two Parsers: Investigating and Combining Graph-based and Transition-based Dependency Parsing[C]. Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing. 2008:562–571.

[37] Nivre J. Algorithms for deterministic incremental dependency parsing[C]. Computational Linguistics. 2008，34:513–553.

[38] Huang L, Jiang W, Liu Q. Bilingually-Constrained (Monolingual) Shift-Reduce Parsing[C]. Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing. 2009:1222–1231.

[39] Nivre J, Hall J. Maltparser: A language-independent system for data-driven dependency parsing[C]. In Proc. of the Fourth Workshop on Treebanks and Linguistic Theories. 2005:13–95.

[40] Duan X, Xu J Z B. Probabilistic Parsing Action Models for Multi-Lingual Dependency Parsing[C]. Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007. 2007:940–946.

[41] Johansson R, Nugues P. Incremental Dependency Parsing Using Online Learning[C]. Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007. 2007:1134–1138.

[42] McDonald R, Nivre J. Characterizing the Errors of Data-Driven Dependency Parsing Models[C]. Proceedings of EMNLP-CoNLL 2007. 2007:122–131.

[43] Sagae K, Lavie A. Parser Combination by Reparsing[C]. Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume. 2006:129–132.

[44] Nivre J, McDonald R. Integrating Graph-Based and Transition-Based Dependency Parsers[C]. Proceedings of ACL 2008. 2008:950–958.

[45] Koo T, Carreras X, Collins M. Simple Semi-supervised Dependency Parsing[C]. Proceedings of ACL-08: HLT. 2008:595–603.

[46] Chen W, Kazama J, Uchimoto K, et al. Improving Dependency Parsing with Subtrees from Auto-Parsed Data[C]. Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing. 2009:570–579.

[47] McClosky D, Charniak E, Johnson M. Effective Self-Training for Parsing[C]. Proceedings of the Human Language Technology Conference of the NAACL. 2006:152–159.

[48] McClosky D, Charniak E, Johnson M. Reranking and Self-Training for Parser Adaptation[C]. Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics. 2006:337–344.

[49] McClosky D, Charniak E, Johnson M. When is Self-Training Effective for Parsing?[C]. Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008). 2008:561–568.

[50] Charniak E, Johnson M. Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking[C]. Proceedings of ACL-05. 2005:173–180.

[51] Chu Y J, Liu T H. On the shortest arborescence of a directed graph[J]. Science Sinica, 1965, 14:1396–1400.

[52] McDonald R, Pereira F, Ribarov K, et al. Non-Projective Dependency Parsing using Spanning Tree Algorithms[C]. Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing. 2005:523–530.

[53] Nivre J, Nilsson J. Pseudo-Projective Dependency Parsing[C]. Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL05). 2005:99–106.

[54] Nivre J. Non-Projective Dependency Parsing in Expected Linear Time[C]. Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP. 2009:351–359.

[55] McClosky D, Charniak E, Johnson M. Automatic Domain Adaptation for Parsing[C]. Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. 2010:28–36.

[56] 段湘煜, 赵军, 徐波. 基于动作建模的中文依存句法分析[J]. 中文信息学报, 2007, 21(5).

[57] 辛霄, 范士喜, 王轩, 王晓龙. 基于最大熵的依存句法分析[J]. 中文信息学报, 2009, 23(2).

[58] 鉴萍, 宗成庆. 基于序列标注模型的分层式依存句法分析方法[J]. 中文信息学报, 2010, 24(6).

[59] 计峰,邱锡鹏. 基于序列标注的中文依存句法分析方法[J]. 计算机应用与软件, 2009, 26(10).

[60] Duan X, Zhao J, , et al. Probabilistic models for action-based Chinese dependency parsing[C]. Proceedings of ECML/ECPPKDD. 2007.

[61] Titov I, Henderson J. Fast and Robust Multilingual Dependency Parsing with a Generative Latent Variable Model[C]. Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007. 2007:947–951.

[62] Eisner J. Efficient Normal-Form Parsing for Combinatory Categorial Grammar[C]. Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL). 1996:79–86.

[63] Chen W, Kazama J, Tsuruoka Y, et al. Improving Graph-based Dependency Parsing with Decision History[C]. Coling 2010: Posters. 2010:126–134.

[64] Zhang H, McDonald R. Generalized Higher-Order Dependency Parsing with Cube Pruning[C]. Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. 2012:320–331.

[65] Nunberg G. The linguistics of punctuation[M]. Lecture Notes, no. 18. Stanford, CA: Center for the Study of Language and Information, 1990.

[66] Collins M. Head-Driven Statistical Models for Natural Language Parsing[C]. Computational Linguistics. 2003，29:589–637.

[67] White M, Rajkumar R. A More Precise Analysis of Punctuation for Broad-Coverage Surface Realization with CCG[C]. Coling 2008: workshop on Grammar Engineering Across Frameworks. 2008:17–24.

[68] xun Jin M, Kim M Y, Kim D, et al. Segmentation of Chinese Long Sentences Using Commas[C]. ACL SIGHAN Workshop 2004. 2004.

[69] Li X, Zong C, Hu R. A Hierarchical Parsing Approach with Punctuation Processing for Long Chinese Sentences[C]. IJCNLP 2005: Companion Volume. 2005.

[70] 毛奇, 连乐新, 周文翠, 袁春风. 基于标点符号分割的汉语句法分析算法[J]. 中文信息学报, 2007, 21(2).

[71] Bohnet B. Efficient Parsing of Syntactic and Semantic Dependency Structures[C]. Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task. 2009:67–72.

[72] Bohnet B. Top Accuracy and Fast Dependency Parsing is not a Contradiction[C]. Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010). 2010:89–97.

[73] Huang L, Chiang D. Better k-best Parsing[C]. Proceedings of IWPT 2005. 2005.

[74] Gesmundo A, Henderson J, Merlo P, et al. A Latent Variable Model of Synchronous Syntactic-Semantic Parsing for Multiple Languages[C]. Proceedings of CoNLL 2009: Shared Task. 2009:37–42.

[75] Che W, Li Z, Li Y, et al. Multilingual Dependency-based Syntactic and Semantic Parsing[C]. Proceedings of CoNLL 2009: Shared Task. 2009:49–54.

[76] Xue N, Xia F, Chiou F D, et al. The Penn Chinese TreeBank: Phrase Structure Annotation of a Large Corpus[C]. Natural Language Engineering. 2005，11:207–238.

[77] Zhang Y, Clark S. Joint Word Segmentation and POS Tagging Using a Single Perceptron[C]. Proceedings of ACL-08: HLT. 2008:888–896.

[78] Jiang W, Huang L, Liu Q, et al. A Cascaded Linear Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging[C]. Proceedings of ACL-08: HLT. 2008:897–904.

[79] Kruengkrai C, Uchimoto K, Kazama J, et al. An Error-Driven Word-Character Hybrid Model for Joint Chinese Word Segmentation and POS Tagging[C]. Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP. 2009:513–521.

[80] Toutanova K, Cherry C. A global model for joint lemmatization and part-of-speech prediction[C]. Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP. 2009:486–494.

[81] Cohen S B, Smith N A. Joint Morphological and Syntactic Disambiguation[C]. Proceedings of EMNLP-CoNLL 2007. 2007:208–217.

[82] Goldberg Y, Tsarfaty R. A Single Generative Model for Joint Morphological Segmentation and Syntactic Parsing[C]. Proceedings of ACL-08: HLT. 2008:371–379.

[83] Finkel J R, Manning C D. Joint Parsing and Named Entity Recognition[C]. Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics. 2009:326–334.

[84] Li J, Zhou G, Ng H T. Joint Syntactic and Semantic Parsing of Chinese[C]. Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. 2010:1108–1117.

[85] Che W, Liu T. Jointly Modeling WSD and SRL with Markov Logic[C]. Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010). 2010:161–169.

[86] Dyer C. Using a maximum entropy model to build segmentation lattices for MT[C]. Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics. 2009:406–414.

[87] Xiao X, Liu Y, Hwang Y, et al. Joint Tokenization and Translation[C]. Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010). 2010:1200–1208.

[88] Liu Y, Liu Q. Joint Parsing and Translation[C]. Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010). 2010:707–715.

[89] McDonald R, Lerman K, Pereira F. Multilingual Dependency Analysis with a Two-Stage Discriminative Parser[C]. Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X). 2006:216–220.

[90] Lee J, Naradowsky J, Smith D A. A Discriminative Model for Joint Morphological Disambiguation and Dependency Parsing[C]. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. 2011:885–894.

[91] Rush A M, Sontag D, Collins M, et al. On Dual Decomposition and Linear Programming Relaxations for Natural Language Processing[C]. Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. 2010:1–11.

[92] Auli M, Lopez A. A Comparison of Loopy Belief Propagation and Dual Decomposition for Integrated CCG Supertagging and Parsing[C]. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. 2011:470–480.

[93] Ratnaparkhi A. A Maximum Entropy Model for Part-Of-Speech Tagging[C]. Proceedings of EMNLP 1996. 1996.

[94] Lafferty J, McCallum A, Pereira F. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data[C]. Proceedings of ICML 2001. 2001:282–289.

[95] Collins M, Globerson A, Koo T, et al. Exponentiated Gradient Algorithms for Conditional Random Fields and Max-Margin Markov Networks[J]. JMLR, 2008, 9:1775–1822.

[96] Eisner J. Three New Probabilistic Models for Dependency Parsing: An Exploration[C]. Proceedings of COLING 1996. 1996:340–345.

[97] Zhang Y, Clark S. Syntactic processing using the generalized perceptron and beam search[J]. Computational Linguistics, 2011, 37(1):105–151.

[98] Liu X, Zhou M, Zhou X, et al. Joint Inference of Named Entity Recognition and Normalization for Tweets[C]. Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics. 2012:526–535.

[99] Constant M, Sigogne A, Watrin P. Discriminative Strategies to Integrate Multiword Expression Recognition and Parsing[C]. Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2012:204–212.

[100] Minkov E, Zettlemoyer L. Discriminative Learning for Joint Template Filling[C]. Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics. 2012:845–853.

[101] Petrov S, Klein D. Improved Inference for Unlexicalized Parsing[C]. Proceedings of NAACL 2007. 2007.

[102] Johansson R, Nugues P. Dependency-based Semantic Role Labeling of PropBank[C]. EMNLP-2008. 2008.

[103] Bansal M, Klein D. Web-Scale Features for Full-Scale Parsing[C]. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. 2011:693–702.

[104] Zhou G, Zhao J, Liu K, et al. Exploiting Web-Derived Selectional Preference to Improve Statistical Dependency Parsing[C]. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. 2011:1556–1565.

[105] Burkett D, Klein D. Two Languages are Better than One (for Syntactic Parsing)[C]. Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing. 2008:877–886.

[106] Chen W, Kazama J, Torisawa K. Bitext Dependency Parsing with Bilingual Subtree Constraints[C]. Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. 2010:21–29.

[107] 王跃龙, 姬东鸿. 汉语树库综述[J]. 当代语言学, 2009, 11(1).

[108] Liu T, Ma J, Li S. Building a Dependency Treebank for Improving Chinese Parser[C]. Journal of Chinese Language and Computing. 2006，16:207–224.

[109] Chen K J, Luo C C, Chang M C, et al. Sinica treebank: Design criteria, representational issues and implementation[M]. Kluwer Academic Publishers, 2003: 231–248.

[110] 周强. 汉语句法树库标注体系[J]. 中文信息学报, 2004, 18(4).

[111] Niu Z Y, Wang H, Wu H. Exploiting Heterogeneous Treebanks for Parsing[C]. Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP. 2009:46–54.

[112] Jiang W, Huang L, Liu Q. Automatic Adaptation of Annotation Standards: Chinese Word Segmentation and POS Tagging – A Case Study[C]. Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP. 2009:522–530.

[113] Smith D A, Eisner J. Parser Adaptation and Projection with Quasi-Synchronous Grammar Features[C]. Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing. 2009:822–831.

[114] Wang J N, Chang J S, Su K Y. An Automatic Treebank Conversion Algorithm for Corpus Sharing[C]. Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics. 1994:248–254.

[115] Collins M, Ramshaw L, Hajic J, et al. A Statistical Parser for Czech[C]. ACL 1999. 1999:505–512.

[116] Xia F, Palmer M. Converting Dependency Structures to Phrase Structures[C]. In Proceedings of HLT 2001. 2001:1–5.

[117] Xia F, Bhatt R, Rambow O, et al. Towards a Multi-Representational Treebank[C]. In Proceedings of the 7th International Workshop on Treebanks and Linguistic Theories. 2008.

[118] Li Z, Che W, Liu T. A Study on Constituent-to-Dependency Conversion[J]. Journal of Chinese Information Processing (in Chinese), 2008, 6(22):14–19.

[119] Martins A F T, Das D, Smith N A, et al. Stacking Dependency Parsers[C]. EMNLP'08. 2008:157–166.

[120] Smith D, Eisner J. Quasi-Synchronous Grammars: Alignment by Soft Projection of Syntactic Dependencies[C]. Proceedings on the Workshop on Statistical Machine Translation. 2006:23–30.

[121] Gimpel K, Smith N A. Quasi-Synchronous Phrase Dependency Grammars for Machine Translation[C]. Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing. 2011:474–485.

[122] Woodsend K, Lapata M. Learning to Simplify Sentences with Quasi-Synchronous Grammar and Integer Programming[C]. Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing. 2011:409–420.

[123] Charniak E. A Maximum-Entropy-Inspired Parser[C]. ANLP'00. 2000:132–139.

攻读博士学位期间发表的论文及其他成果

（一）发表的学术论文

[1] **李正华**, 车万翔, 刘挺. *基于柱状搜索的高阶依存句法分析*. 中文信息学报. 2010, 24(1):37-41.

[2] **Zhenghua Li**, Ting Liu, Wanxiang Che. *Exploiting Multiple Treebanks for Parsing with Quasi-synchronous Grammar*. In Proceedings of the 50th Annual Meeting of the Association of Computational Linguistics (ACL 2012) (acceptance rate: 112/571=19.6%). 2012.07, pp. 675-684, Jeju, Republic of Korea.

[3] **Zhenghua Li**, Min Zhang, Wanxiang Che, Ting Liu, Wenliang Chen, Haizhou Li. *Joint Models for Chinese POS Tagging and Dependency Parsing*. In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011). 2011.07, pp. 1180-1191. Edinburgh, Scotland, UK. (**EI Indexed: 20114014390107**)

[4] **Zhenghua Li**, Min Zhang, Wanxiang Che, Ting Liu. *A Separately Passive-Aggressive Training Algorithm for Joint POS Tagging and Dependency Parsing.* In Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012). 2012.12, pp. 1681-1698. Mumbai, India.

[5] **Zhenghua Li**, Wanxiang Che, Ting Liu. *Improving Chinese POS Tagging with Dependency Parsing.* In the Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011). 2011.08, pp. 1447-1451, Chiang Mai, Thailand.

[6] **Zhenghua Li**, Wanxiang Che, Ting Liu. *Improving Dependency Parsing Using Punctuation*. In Proceedings of the International Conference on Asian Language Processing (IALP). 2010.12, pp. 53-56. Harbin, China. (**EI Indexed: 20110613644353**)

哈尔滨工业大学学位论文原创性声明及使用授权说明

学位论文原创性声明

本人郑重声明：此处所提交的学位论文《汉语依存句法分析关键技术研究》，是本人在导师指导下，在哈尔滨工业大学攻读学位期间独立进行研究工作所取得的成果。据本人所知，论文中除已注明部分外不包含他人已发表或撰写过的研究成果。对本文的研究工作做出重要贡献的个人和集体，均已在文中以明确方式注明。本声明的法律结果将完全由本人承担。

作者签名：　　日期：　　年　月　日

学位论文使用授权说明

本人完全了解哈尔滨工业大学关于保存、使用学位论文的规定，即：

（1）已获学位的研究生必须按学校规定提交学位论文；（2）学校可以采用影印、缩印或其他复制手段保存研究生上交的学位论文；（3）为教学和科研目的，学校可以将学位论文作为资料在图书馆及校园网上提供目录检索与阅览服务；（4）根据相关要求，向国家图书馆报送学位论文。

保密论文在解密后遵守此规定。

本人保证遵守上述规定。

作者签名：　　日期：　　年　月　日

导师签名：　　日期：　　年　月　日

致　谢

自从2004年夏天加入信息检索实验室，我在此经历了将近9年的青春时光。这些年发生了很多事情，我也从一个懵懂青年转变成为一个即将涉世的成年人。实验室成了我的另一个家。未来的日子，我会继续努力，做有意思的事情，成为一个快乐的人。

首先，我要感谢我的导师刘挺教授。感谢刘老师接收我进入SCIR这样一个快乐、团结、进步的团队，并不遗余力的关心我、培养我。刘老师不仅对我的学术研究提供很大的帮助，给我创造了去国外实习锻炼的机会，并且在做人做事上不断教导我，让我快速成长。

同时我要感谢我们LA组的老大，也是我的直接指导者车万翔老师。在LA组的这5年里，车老师给了我很多帮助。车老师视野开阔、精力充沛，每一次跟我讨论idea时都能提出很多有帮助的意见。在我遇到困难的时候，总是给我鼓励。

感谢李生教授。博士课题开题、中期报告后，李老师总能从大局上帮我分析和把握课题方向，提出很好的建议，并且督促我尽快做出成果。感谢秦兵教授。自从进入实验室，尤其是本科和硕士期间在TM组的阶段，秦老师在工作和生活上给了我很多帮助和指导，我会永远记着秦老师暖暖的微笑。感谢张宇老师。每次向张老师请教工作或生活中的困难时，张老师总是诚恳细心没有保留的给出很多具体的建议。

此外，还要感谢通信研究所张民教授在新加坡实习期间，张老师在做研究上给了我很多点拨和帮助。写完论文，张老师总是帮我精心反复修改，让人感动。感谢陈文亮研究员。新加坡实习期间，我们深入讨论了句法分析相关问题，对我之后的研究工作很有启发。并且，陈文亮对我在论文写作也有很多帮助和点拨。

感谢陈毅恒老师给我很多真诚的意见。感谢赵妍妍老师这些年和我讨论问题，并且给我很多帮助。感谢郎君师兄在实验室和新加坡对我的指导和帮助。师兄作为俱乐部的面试官，将我领进实验室，并且从一点一滴指导我，帮助我，直到我步入正轨。新加坡实习期间，师兄帮我租房子、装机器、...，直到我适应新加坡生活。感谢赵世奇师兄对我的帮助。我和师兄在学术上的直接讨论不多，但是每一次他的建议都非常中肯、一针见血。作为我做研究的榜样，师兄

让我受益良多。感谢马金山师兄。博士毕业前几个月，我开始对句法分析感兴趣，实验室安排我接手师兄的工作。师兄在最忙碌的那段时间里，甚至工作后的一段时间里，都耐心的解答我各种问题。感谢高立琦师兄。我在北京金山公司实习的一段时间里，师兄给了我很多帮助和指导。感谢张志昌师兄、贺瑞芳师姐、伍大勇师兄、宋巍、郭宇航、付瑞吉、张梅山、付博、郭江、刘一佳、唐都钰、刘安安等，感谢我的本硕博同学王宝勋、于博、吕新波、王金宝，在最珍贵的这几年里，我们一起生活、一起讨论问题、一起成长，一起见证青春。感谢王丽杰、陈鑫、韩中华、赵静、郭江、唐国华，2010年夏我们一起标注树库的艰苦岁月值得纪念，感谢你们对我的帮助。感谢我带过的两个小弟，陈鑫和陆子龙，虽然水平有限，但是指导你们的过程中我也受益良多。

感谢社会计算与信息检索研究中心的每一位成员。大家营造了一个很好的科研和生活氛围。在这个大家庭里，我们互相学习、互相鼓励、互相帮助。祝愿这个大家庭不断发展!

最后，我深深感谢这些年一直支持我、关心我的父母和爱人。你们是我的港湾。当累了、失去信心的时候，只需回到温暖的港湾中休息，就能重拾信心，精力充沛的继续战斗。

个人简历

李正华，男，1983年9月生，出生于陕西省合阳县。

教育经历

- 2008.9 – 2013.4：就读于哈尔滨工业大学计算机学院社会计算与信息检索研究中心，获得计算机应用技术学科博士学位，导师为刘挺教授。
- 2006.9 – 2008.7：就读于哈尔滨工业大学计算机学院社会计算与信息检索研究中心，获得工学硕士学位，导师为刘挺教授。
- 2002.9 – 2006.7：就读于哈尔滨工业大学计算机学院，获工学学士学位。

实习经历

- 2010.9 – 2011.3: 在新加坡信息通讯研究所实习（Institute for Infocomm Research）。在张民教授指导下，研究了词性标注和依存句法分析联合模型，在自然语言处理领域顶级会议EMNLP 2011上发表一篇论文。
- 2007.6 – 2007.12: 在北京金山公司实习。主要研究汉语依存句法树库转化问题，发表一个专利，并在中文信息学报上发表一篇论文。另外，参与了一个机器翻译项目，了解了统计机器翻译中的一些核心技术。

获奖情况

- 2010年因开发“语言技术平台”，获“钱伟长中文信息处理科学技术奖”一等奖，在获奖名单中列第5位。
- 获哈尔滨工业大学2009-2010年度“优秀学生干部”称号。
- 2008年：以第一作者撰写的论文“李正华,车万翔,刘挺. 短语结构树库向依存结构树库转化研究”，获得第四届全国学生计算语言学研讨会（SWCL 2008）优秀论文。
- 2006.9 – 2008.7：获得2次一等“人民奖学金”。
- 2004年获“联想奖学金”。
- 获得哈尔滨工业大学2003-2004年度“三好学生标兵”称号。
- 2002.9 – 2006.7：获得4次一等“人民奖学金”；1次二等“人民奖学金”。

参与项目

- 2007.1 – 2008.12：基于XML的分层交互式中文处理开放平台。国家863项目（2006AA01Z145）。

- 2009.1 – 2011.12：汉语依存句法分析若干关键技术研究。国家自然科学青年基金（60803093）。
- 2007.1 – 2009.12：汉语语义角色标注方法研究。国家自然科学基金面上项目（60675034）。
- 2008.1 – 2010.12：基于实体关系的文本内容挖掘与集成技术平台。863计划探索类专题项目（2008AA01Z144）。
- 2011.1 – 2012.12：基于分治算法的高效率依存句法分析研究。哈工大语言语音教育部-微软重点实验室开放基金（HIT.KLOF.2010064）。
- 2010.1 – 2011.12：汉语语义依存分析技术研究。哈尔滨工业大学科研创新基金（HIT.NSRIF.2009069）。

系统开发和数据标注

- 2007.11 – 2011.3：开发和维护“语言技术平台（Language Technology Platform, LTP）”。期间独立重写了LTP的数据表示层和逻辑控制层代码，独立开发了准确率更高的中文依存分析模块，协助设计并开发了可视化界面和网络服务（Web Service）程序接口。
- 2010.7 – 2010.9：作为主要负责人，组织实验室5位硕士生和本科生，聘请11位在校学生标注了6万句依存句法树库。目前，这个大规模中文依存句法树库已经在Linguistic Data Consortium（LDC）上发布。

参加评测

- 参加2012年国际面向互联网数据的句法分析评测（SANCL）。在提交的12个结果中，我们的系统排列第3名。
- 参加2009年计算自然语言学习国际会议（CoNLL）组织的多语种依存句法分析和语义角色标注联合评测任务。在21家参加单位中，我们的系统在联合任务上取得第1名，在依存句法分析任务上列第3名。
- 参加2008年计算自然语言学习国际会议（CoNLL）组织的英语依存句法分析和语义角色标注联合评测任务。在19家参加单位中，我们的系统在联合任务上取得第3名，在依存句法分析任务上列第5名。

论文及专利成果

- 在自然语言处理国际顶级和重要会议ACL, EMNLP, COLING, IJCNLP上以第一作者发表论文。
- 李正华, 高立琦, 刘挺, 王海洲. 一种树库转化方法及树库转化系统. 专利号: CN101201819