

本 科 毕 业 设 计（论 文）

学院(部) 计算机科学与技术学院

题　　目 基于局部标注的依存句法分析主动学习研究与实现

年　级 2016 级　专业 软件工程

班　级 软件工程　学号 1627406031

姓　名 侯洋

指导老师 李正华　职称 副教授

论文提交日期 2020 年 5 月 20 日

目 录

摘要 1
Abstract 2
第 1 章 绪论 3
1.1 课题研究背景 3
1.2 课题研究现状 4
1.3 论文内容和结构安排 5
第 2 章 基于图的神经依存句法分析 6
2.1 依存句法 6
2.2 依存句法分析 7
2.3 基于图的依存句法分析 8
2.4 Eisner 解码算法 9
2.5 传统的特征抽取做法 10
2.6 从机器学习到深度学习 12
2.7 深度双仿射神经依存句法分析器 13
2.7.1 深度双向长短期记忆网络 13
2.7.2 特征编码 14
2.7.3 维度缩减 15
2.7.4 双仿射打分 16
第 3 章 依存句法分析主动学习 17
3.1 主动学习 17
3.1.1 主动学习简介 17
3.1.2 主动学习与半监督学习 17
3.2 基于池的主动学习 18
3.2.1 基于池的场景 18
3.2.2 查询策略 19
3.3 依存句法分析主动学习 20
3.3.1 边缘概率 20

3.3.2 句子级全标注 22
3.3.3 单一依存弧级局部标注 23
3.3.4 批量依存弧级局部标注 24
第 4 章 模拟实验 **25**
4.1 实验设置 25
4.2 句子级全标注对比单一依存弧级局部标注 26
4.3 单一依存弧级局部标注对比批量依存弧级局部标注 28
4.4 测试集上的结果 30
第 5 章 总结和展望 **31**
5.1 总结 31
5.2 展望 31
参考文献 **32**
致谢 **35**

摘要

依存句法分析是自然语言处理的核心基础任务之一。给定输入词序列，其目标是得到一棵依存句法树，来刻画句子的句法结构。在传统的机器学习时代，高精度的依存句法分析器需要大量的树库进行训练。在深度学习时代，得益于深层神经网络强大的表示学习能力，尽管使用的是相同的树库，依存句法分析器的性能却有了较大提升。但是，深度学习对于大规模树库的需求却没有丝毫减少。

现实中，标注好的树库资源十分稀少，而标注大规模、高质量的树库又是非常耗时且昂贵的。解决该问题的一种有前景方法是主动学习。主动学习通过选择最富有信息的任务进行标注，从而减少标注代价。

在依存句法分析中，常见的主动学习方法是完整标注不确定的句子。实际上，在训练一个句法分析器时，句子中的一些部分并不包含有用的信息。本文表明选择整个句子进行标注并不是最优的方法，以依存关系为单元，选择不确定性度量低的依存关系进行标注更灵活、更有效。此外，为了比较全标注和局部标注，本文使用几种不确定性度量进行了模拟实验。

关键词：主动学习；依存句法分析；局部标注；不确定性度量；深度双仿射神经句法分析器

Abstract

Dependency parsing is one of the core basic tasks of natural language processing. Given an input sentence, It aims to get a parse tree which describes the syntactic structure of the sentence. At the age of traditional machine learning, high-precise parsers require large treebanks to estimate their parameters. At the age of deep learning, parsers which apply neural networks get better performance with the same treebanks. However, for deep learning, the needs for large treebanks don't decrease at all.

As a matter of fact, resources for labeled treebanks are hardly available while treebanks of large scale and high quality are time-consuming and expensive to produce. One of the most promising approaches to this issue is active learning. Active learning can reduce the cost of annotation by choosing and annotating tasks having most informativeness.

The common method used in active learning for dependency parsing is annotating uncertain sentences completely. Actually, some part of the sentences doesn't contain useful information when training a parser. This paper shows that annotating the whole sentence isn't the optimal method compared with annotating dependencies which is more flexible and effective. Furthermore, to compare full annotation and partial annotation, this paper conduct simulation experiments with several uncertainty metrics.

Keywords: Active Learning; Uncertainty Metrics; Dependency Parsing; Partial Annotation; Deep Biaffine Parser

第 1 章 绪论

1.1 课题研究背景

依存句法分析（Dependency Parsing）是自然语言处理领域中的一项基础性任务，也是机器翻译、信息检索等其它任务的前提。依存句法分析通过分析出句子的结构来为其它的应用提供帮助。

随着神经网络的流行，过去几年里，依存句法分析在标准文本上的性能有了较大提升，表 1.1 和表 1.2 显示了准确性的提升。但是，这些句法分析方法大多数是有监督的，即需要预先准备大量的句子和它们对应的句法结构（即树库）。一般而言，可以通过考虑更复杂的句法特征、使用更精致的网络结构来提升模型的性能。不幸的是，当数据十分稀疏时，分析器并不能可靠地评估参数从而提升模型的准确率。因此，树库的大小成为了句法分析器准确率的一个限制。

表 1.1：多种依存句法分析模型在 PTB-SD 3.3.0 和 CTB 5.1 上的准确率

模型	年份	PTB-SD		CTB	
		UAS	LAS	UAS	LAS
Zhou and Zhao (BERT)	2019	97.00	95.43	91.21	89.15
Ma et al.	2018	95.87	94.19	90.50	89.29
Dozat and Manning	2017	95.74	94.08	89.30	88.23
Kiperwasser and Goldberg	2016	93.10	91.00	86.60	85.10
Chen and Manning	2014	91.80	89.60	83.90	82.40

尽管准确率的记录被不断打破，使用通用数据进行训练得到的句法分析器在特定领域下的表现却并不让人满意，一旦应用到了新的领域，分析器的准确率往往会断崖式下跌。但是，在无领域的依存句法分析技术方面，研究人员们迟迟没有进展。另一方面，网络数据（如微博，商品评价）不断涌现，这种非规范文本的出现，给现行的句法分析技术带来了巨大的挑战。解决上述问题的一种可行方法是，对于目标领域或体裁，人工标注一定数量的文本。

考虑到给新句子标注句法结构的代价，在标注一个新的句子并将其添加到树库前，自然要询问新句子包含的信息是否有利于句法分析器的训练。与此同时，主动学习（Active Learning，AL）旨在通过选择并人工标注对模型训练最有价值的无标签样本减少标注代价 (Olsson, 2009)[1]。

1.2 课题研究现状

主动学习已经被应用到了许多自然语言处理的任务中，例如语音识别、信息抽取、词性标注和句法分析。本文主要关注它在句法分析中的应用。

传统上，在句法分析中主动学习中，一个句子作为标注任务，被标注成一棵完整的句法树。这种将完整的句子作为基本单元的标注方式称为全标注（Full Annotation，FA）。Tang et al. (2002)[2]和 Hwa (2004)[3]都使用不确定性采样的方法，提出将句法分析的概率分布熵作为不确定性度量。他们认为熵越大，分析器对选出的句子就越不确定。这两个工作的共同点就是在任务选择时，只考虑了完整的句子。但是，正如 Mejer 和 Crammer (2012)[4]评价的那样，当需要全标注时，标注过程是复杂的、缓慢的、且容易出错的。在很多情况下，句子的句法结构只有一部分是不确定的。标注这部分结构可以使分析器受益，而剩余的结构可以直接由当前的模型得到 (Flannery and Mori, 2015)[5]，无需浪费精力来标注它们。

只标注句子部分结构的标注方式称为局部标注（Partial Annotation，PA）。近年来，对于基于局部标注的依存句法分析主动学习，研究人员们报告了比较有前景的结果。他们发现在选择富有信息的结构上，比起整个句子，更小的子句单元可以提供更高的灵活性。Sassano 和 Kurohashi (2010)[6]已经在使用子句单元进行主动学习，他们用一个分类器选出得分最小的那些（日文）短语对，并让标注者判断短语对之间是否存在依存关系。相较于全标注，本文转向了另一个极端：只考虑一个句子中不确定的单词，而非考虑整个句子。更准备的说，我们考虑的是单词间的依存关系，即标注任务是选择句子中最不确定的依存关系。

深度学习流行以来，对于依存句法分析主动学习的研究实际上已经停滞，人们更加关注如何利用神经网络来提升句法分析的准确率。本文是第一个利用较先进的神经模型进行依存句法分析主动学习的。

1.3 论文内容和结构安排

首先，本文对依存句法和依存句法分析进行了简单的介绍，随后说明了基于图的依存句法分析框架，并讲解了 Eisner 解码算法。之后，比较了机器学习时代的做法和深度学习时代的做法，并详细描述了深度双仿射神经句法分析器。

其次，介绍了主动学习，具体描述了基于池的主动学习方法，并列举了常用的以及实验中使用的不确定性度量。

最后进行了模拟实验，通过对实验结果的分析，证明了使用局部标注进行依存句法主动学习的有效性。

本文分为 5 章：

第 1 章：绪论。本章主要说明了依存句法分析主动学习的研究背景和研究现状，然后概括了本文每章的内容。

第 2 章：基于图的神经依存句法分析。本章先简单介绍了依存句法，以及依存句法分析任务的定义。随后，描述了一种基于图的依存句法框架，并详解如何用 Eisner 算法得到最大的树得分。接着，举例说明了机器学习时代是如何用特征模板进行特征组合的，并通过对传统做法缺陷的总结，引出了神经网络如何解决特征抽取问题。最后，详细描述了深度双仿射神经句法分析器。

第 3 章：依存句法分析主动学习。本章先介绍了什么是主动学习，其次具体描述了基于池的主动学习框架及其采用的查询策略。然后，介绍了本文为什么要使用边缘概率，最后说明了依存句法分析主动学习的流程，并列举了使用的不确定性度量。

第 4 章：模拟实验。本章先说明了实验的具体设置，然后在全标注和局部标注之间进行了实验对比，之后又在单一依存弧级局部标注和批量依存弧级局部标注间进行了实验对比，最后结合测试集上的结果进行了总结。

第 5 章：总结和展望。本章先对本文的工作进行了总结，接着表述了可以从哪些方面对工作内容进行扩展。

第 2 章 基于图的神经依存句法分析

本章会先简单介绍什么是依存句法分析，随后会讲解如何采用基于图的方法进行依存句法分析。接着，说明 McDonald 的传统特征模板做法。最后，详细描述深度双仿射神经句法分析器。

2.1 依存句法

句法研究的是句子中的各个组成部分，基于依存关系的依存句法是现代句法理论的一个类别，最早可以追溯 Lucien Tesnière 的工作。依存语法有一个基本假设：句法结构是由词和词之间的依存（修饰）关系组成的。一个依存关系连接着两个单词，分别是核心词和修饰词。处于支配地位的单词称为核心词，处于从属地位的单词称为修饰词。例如，“美丽/的/花”中，“美丽”修饰了“花”，这两个词之间就存着依存关系。其中，“花”是核心词，“美丽”则是修饰词。在依存句法中，动词被认为是子句的结构中心，其它词通过依存关系直接或间接连接到动词，因此动词一般作为核心词出现。

一个句子中会有主语、谓语和宾语，而人们常说的主谓关系、动宾关系就是依存关系。依存句法对“主谓”、“动宾”这些具体的关系类型其实并不怎么关心，而是关注两个词之间是否存在依存关系，即无标签依存句法分析。但是，在实际应用中，一般会标明依存关系的具体类型，丰富依存关系传达的句法信息，即有标签依存句法分析。而文本并不关心依存关系的具体类型，只关注于两个单词之间是否存在依存关系。因此，本文之后所描述的都是无标签依存句法分析。

依存句法树是一棵刻画了句子的句法结构的树，可以用来表示句子中词与词的依存关系，即捕捉单词之间的修饰关系。图 2.1 给出了一棵依存句法树的示例。依存句法树中最基本的单元是依存弧，它对应着依存句法中的依存关系。一条依存弧由一个核心词和一个修饰词组成，并由核心词指向修饰词。在本文之后的内容中，会混合使用“依存弧”和“依存关系”，因为这两个词表达了同一概念。图 2.1 中，$表示根节点，这是额外添加的伪结点，为了让树更规范，也可以使句法分析更顺利。

一棵合法的依存句法树需要满足以下三个条件：

（1）唯一根节点：只有一个指定的根节点，并且没有指向它的依存弧。

（2）唯一父节点：除了根节点以外，每个子节点（修饰词）有且仅有一个父节点（核心词）。

（3）无环：从根节点到其它节点只有一条路径，即多条弧不能构成环。

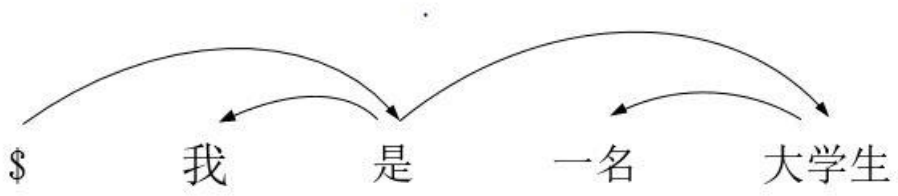


图 2.1：依存句法树示例

2.2 依存句法分析

依存句法分析是自然语言处理中的关键技术之一，它的任务是对输入的句子进行分析以得到句子的句法结构，并定义了核心词和修饰词之间的关系。对于一个输入的句子 $s = w_1 ... w_n$，依存句法分析的目标是建立一个有向的依存句法树 $y = \{h \to m : 0 \le h \le n, 1 \le m \le n\}$，其中，$|y| = n$，$h \to m$ 代表一条从核心词 h 到修饰词 m 的依存弧。实际的依存句法分析会结合单词的词性信息，最终找出每个词的核心词。一个句法分析的示例如表 2.1 所示。

表 2.1：“我是一名大学生”的句法分析示例

序号	0	1	2	3	4
单词	$	我	是	一名	大学生
词性		PN	VC	CD	NN
核心词		是	$	大学生	是
核心词序号		2	0	4	2

结合表 2.1，判断一条依存弧是否标注正确就变得很简单。对于无标签的依存句法分析，一般采用无标签依存正确率（UAS）来评估分析的结果和分析器的性能：

$$UAS = \frac{\text{正确标出核心词的单词数}}{\text{总单词数}} \tag{2.1}$$

通过依存句法分析得到句子的句法结构，这是语言理解的内在需求，另一方面也可以帮助其它自然语言处理任务。依存关系揭示了谓语和论元之间的语义关系，这在机器翻译、问答系统、信息抽取等其它应用中是十分有用的。例如，机器翻译中，需要对待翻译的文本进行依存句法分析，以得到句子的结构，而这些句法信息可以提升翻译器的性能。

2.3 基于图的依存句法分析

现代的依存句法分析方法可以粗略地分为两类：基于迁移的和基于图的。基于迁移的依存句法分析将分析视为生成一棵句法树的一系列动作（移进和规约），并训练一个分类器给处理过程每个阶段中的可能动作打分来指引分析过程。基于图的依存句法分析把分析过程当作一个基于搜索的结构化预测问题，最终目标是学习一个打分函数。通过打分函数，正确的句法树的得分会大于其它错误的树的得分。考虑到结构化预测拥有的全局性优势，本文使用基于图的方法进行依存句法分析。

基于图的句法分析 (McDonald et al., 2005)[7]定义了一个无需使用生成语法进行句法分析的框架:

$$parse(s) = \arg\max_{y \in \mathcal{Y}(s)} score_{global}(s, y) \tag{2.2}$$

$$score_{global}(s, y) = \sum_{part \in y} score_{local}(s, part) \tag{2.3}$$

在该框架下，对于一个输入的句子 $s = w_1 ... w_n$， s 的任意一棵句法分析树都是 s 的一个合法句法结构表示。一个基于图的句法分析器的核心是打分函数 $score(s, y)$。该函数给每一颗树 $y \in \mathcal{Y}(s)$ 赋予一个得分，$\mathcal{Y}(s)$ 表示句子 s 的所有合法依存句法树的集

合。我们要在$\mathcal{Y}(s)$中找到得分最大的那棵句法分析树。

为了方便打分，一棵句法树的得分通常会被分解为树的的得分总和。最简单的方法就是一阶的以弧为因子的模型 (McDonald, 2006)[8]，这也是本文使用的模型。该模型把一棵树分解为各个弧，然后对弧独立进行打分。最后，树的得分就是所有弧的得分和。其它更复杂的模型虽然考虑到了树中更多的重叠部分，但需要更复杂的训练算法。以弧为因子的句法分析把一个树的得分分解成从核心词h指向修饰词m的依存弧$h \to m$的得分和：

$$parse(s) = \arg\max_{y \in \mathcal{Y}(s)} \sum_{h \to m \in y} score(s, h, m) \tag{2.4}$$

有了全部依存弧的得分，得分最大的树可以使用 Eisner 解码算法高效地得到。

2.4 Eisner 解码算法

如果暂时忽略打分函数$score(s, h, m)$的具体实现，并假设已经知道了s中所有可能的依存弧$h \to m$的得分，那么句法分析的任务就变成了寻找树得分最大的一棵或几颗句法树。而最主要的困难是：对于一个长度为 n 的句子，它所有合法的依存句法树的数量随着句子长度的增加而指数级增长。如果稍微修改并使用短语结构句法中所用的 CKY 算法，一个句子会有$O(n^5)$数量的句法树，且要花费$O(n^5)$的时间来生成这些树。

在 CKY 算法中，一个核心词及其左右所有子孙修饰词构成了一个“成分”，而为了表示两个相邻的成分，需要记录五个位置：左边界的位置，第一个成分的核心词位置，成分交汇处的位置，第二个成分的核心词位置，右边界的位置。Eisner (2000)[9]提出的动态规划算法将句子的“成分”重新定义成两类：完整跨度（Complete Span）和非完整跨度（Incomplete Span）。该算法对于一个核心词左边所有修饰词和右边所有修饰词各自进行分析，在之后的阶段才重新组合到一起。这种方法使得核心词一定位于“跨度”的边界处，就避免记录核心词的位置，将时间复杂度降低到了$O(n^3)$。该算法仅需添加两个额外的二元变量，一个变量指明方向：构成跨度的是核心词与它左边的修饰词还是与它右边的修饰词；另一个变量指明跨度是否完整：在上述方向上，

核心词是否还有未包含到跨度中的修饰词。

“完整/非完整”是 Eisner 算法的一个核心概念。完整跨度由一个核心词和它一边的所有后继修饰词组成。非完整跨度由一个核心词和它的一个修饰词间的所有单词组成。每个大跨度可以自底向上地用两个相邻的更小的跨度组合而成，小跨度组合成大跨度，大跨度再组合成更大的跨度，最终所有的跨度组合成了一个完整的句子，具体算法如下：

算法 2. 1：Eisner 解码算法

输入： $s = w_0 w_1 ... w_n$

初始化： $E[i][i][d][c] = 0,\ for\ i \in [0,n],\ d,c \in [0,1]$

$for\ m : 1...n$

$for\ i : 0...(n-m)$

$j = i + m$

$E[i][j][0][1] = \max_{i \le q < j}(E[i][q][1][0] + E[q+1][j][0][0] + Score(w_j, w_i))$

$E[i][j][1][1] = \max_{i \le q < j}(E[i][q][1][0] + E[q+1][j][0][0] + Score(w_i, w_j))$

$E[i][j][0][0] = \max_{i \le q < j}(E[i][q][0][0] + E[q][j][0][1])$

$E[i][j][1][0] = \max_{i \le q < j}(E[i][q][1][1] + E[q][j][1][0])$

输出： $\arg\max(E[0][n][1][0])$

2.5 传统的特征抽取做法

无论使用什么样的句法分析框架，设计一个句法分析器的关键步骤是给模型选择正确的特征函数。传统上，句法分析器使用手工挑选的特征函数并结合线性模型进行训练。这些特征函数会关注依存弧的一些核心属性：核心词是什么，修饰词是什么，核心词的词性是什么，修饰词的词性是什么，核心词和修饰词间的距离是多少，……。特征函数通常由特征模板组成，每个特征模板都实例化了一个二元指示函数，该二元指示函数的工作机制就像在问问题，如果满足了问题中的条件，函数就会返回 1，反之，返回 0。通过结合所有的特征模板，对于“我/爱”间的依存弧，其特征最终形如：核心词是“爱”，修饰词是“我”，“爱”的词性是“动词”，“我”的词性是“代

词”，两个词之间的距离是 1，……。

下面介绍一下 McDonald et al. (2005)[7]使用的特征模板。如表 2.1 所示。基本的特征列在了表 2.1 的 a)和 b)中。其中，p-word 指的是核心词，c-word 指的是修饰词，p-pos 指的是核心词的词性，c-pos 指的是修饰词的词性。其主要关注的是单词本身和它们的词性。除此以外，还考虑到了依存弧的方向以及核心词与修饰词间的距离。

但是，仅仅使用依存弧两端的单词作为特征却不够精确，因为脱离单词出现的上下文来判断两个词之间是否存在依存关系会导致很大的偏差。为了解决这个问题，McDonald 又加入了其它两类特征，列在了表 2.1 的 c)中。其中，p-pos+1 指的是核心词右边的单词的词性，p-pos-1 指的是核心词左边的单词的词性，c-pos+1 指的是修饰词右边的单词的词性，c-pos-1 指的是修饰词左边的单词的词性，b-pos 指的是依存弧间的单词的词性。第一类特征考虑了核心词和修饰词间的所有单词的词性，表示成三元组的形式：核心词的词性，中间词的词性，修饰词的词性。这类特征能够排除一些罕见的语言现象。第二类特征考虑到了核心词和修饰词前后的单词，表示成四元组的形式：核心词的词性，修饰词的词性，核心词前后单词的词性，修饰词前后单词的词性。这类特征提供了依存关系所处的局部语境的信息。添加这两类特征较好地提升了模型的性能。

表 2.2：McDonald 的特征模板

a)	**b)**	**c)**
一元特征	**二元特征**	**依存弧间的单词的词性特征**
p-word, p-pos	p-word, p-pos, c-word, c-pos	p-pos, b-pos, c-pos
p-word	p-pos, c-word, c-pos	**周围的单词的词性特征**
p-pos	p-word, c-word, c-pos	p-pos, p-pos+1, c-pos-1, c-pos
c-word, c-pos	p-word, p-pos, c-pos	p-pos-1, p-pos, c-pos-1, c-pos
c-word	p-word, p-pos, c-word	p-pos, p-pos+1, c-pos, c-pos+1
c-pos	p-word, c-word	p-pos-1, p-pos, c-pos, c-pos+1
	p-pos, c-pos	

借助特征模板，一个特征抽取函数 $\phi(s,h,m)$ 给一条依存弧 $h \to m$ 分配了一个稀疏

的特征向量。特征向量中的每个元素值不是 0 就是 1，形如：[1, 0, 0, 1, 0, 0, 0,...]。McDonald 以及大部分后续的工作都通过一个以权重向量 w 为参数的线性模型来评估一条弧的得分，最终的得分是 $\phi(s,h,m)$ 与 w 的点积：

$$score(s,h,m) = \phi(s,h,m) \bullet w \tag{2.5}$$

2.6 从机器学习到深度学习

在传统的依存句法分析中，设计特征函数是一个巨大的挑战，因为要考虑使用什么语法属性，以及如何组合这些属性。一旦有一篇文章提出了一个优秀的特征函数，基本上之后其它的工作也会使用该函数。上述 McDonald 给基于图的句法分析提出的 18 个特征模板以及 Zhang 和 Niver (2011)[10]给基于迁移的句法分析提出的特征集（包含 20 个核心特征和 72 个特征模板）都表现得不错，且得到了广泛的应用。

由于想出一套优秀的特征集是一个困难且耗时的任务，许多研究人员都在尝试减少需要的人力。随着神经网络的流行，二元指示函数与稀疏、高维的特征向量逐步被抛弃，稠密、低维的词向量（Word Embedding）的使用频率不断提高。Chen 和 Manning (2014)[11]将句法分析器中的每个核心特征编码成了一个稠密的低维向量，这些向量随后被连接起来，然后输入到一个神经网络中，该网络可以潜在地捕捉到任意的特征组合。尽管转向神经网络缓解了手工进行特征组合的需求，但人们还是需要定义一些核心特征。在 Chen 和 Manning (2014)[11]的工作中，特征就表示为 18 个词向量，18 个词性向量和 12 个标签向量的连接。

词向量减轻了特征组合需要的精力，而另一种解决特征工程问题的方法是使用更新颖的神经网络架构。在 Dyer et al. (2015)[11]的工作中，一个基于迁移的句法分析器的整个栈和缓冲都被编码成了一个 stack-LSTMs，每个栈元素都是句法树的一个组合表示。但是，这种方法却从特征工程的极端转向了架构工程的极端。

2.7 深度双仿射神经依存句法分析器

本文使用了由 Dozat 和 Manning (2017)[13]提出的深度双仿射神经依存句法分析器。比起之前的工作，该分析器一方面只需要两个核心特征作为输入，另一方面其神经网络架构也非常简单。其基本结构如图 2.2 所示：

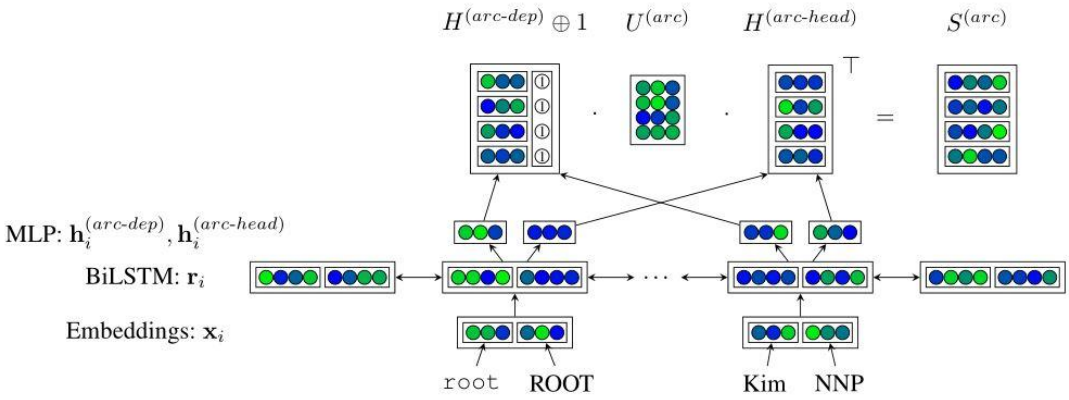


图 2.2：深度双仿射神经句法分析器的架构，此图引用自[13]

本节中，将使用大写的英文字母表示矩阵，使用小写的英文字母表示向量，行向量与列向量不做区分，视具体的情况而定。

2.7.1 深度双向长短期记忆网络

循环神经网络（RNN）是用来建模序列数据的，例如一个输入句子 $\mathbf{s} = w_1 ... w_n$。传统上，人们使用马尔可夫假设（Markov Assumption）来建模序列对象。而 RNN 模型提供了一个全新的框架，把从 1 到 i 的整个历史作为条件。基于过去的从 1 到 i 个元素，一个 RNN 允许人们建模序列中的第 i 个元素。双向循环神经网络（BiRNN），是 RNN 的一个扩展，由一个正向的 RNN 和一个反向的 RNN 组成。BiRNN 既考虑到了过去的 1 到 i 个元素，又考虑到了未来的 n 到 i 个元素。本文使用的的是 RNN 的一个变体，叫做长短期记忆网络（LSTM）。它既继承了 RNN 的优点，又通过门机制解决了 RNN 存在的梯度爆炸和梯度消失问题，因此，LSTM 是目前最广泛使用

的进行序列建模的神经网络。本文使用双向的长短期记忆网络（BiLSTM）来进行特征抽取。

本文不讨论 LSTM 的具体细节，而是将 $LSTM(\bullet)$ 看成一个特征抽取函数，它把一个由 n 个输入向量 $x_1...x_n$ 组成的序列映射成一个由 n 个输出向量 $r_1...r_n$ 组成的序列，即 $LSTM(x_1...x_n) \rightarrow r_1...r_n$。每个输出向量 r_i 都是以它之前的输入向量 $x_1...x_i$ 作为条件，有：

$$r_i = LSTM(x_1...x_n, i) \tag{2.6}$$

BiLSTM 由两个 LSTMs 组成，LSTM_F 正向读取输入序列，LSTM_R 反向读取序列。具体而言，对于特定的输出向量 r_i，可以通过下式得到：

$$\begin{aligned} r_i &= BiLSTM(x_1...x_n, i) \\ &= LSTM_F(x_1...x_n, i) \oplus LSTM_R(x_n...x_1, i) \end{aligned} \tag{2.7}$$

$\oplus$ 表示将两个向量直接连接，这是同时利用双向信息的一种简单做法。

在 BiLSTM 的基础上，研究人员们发明了一种 k 层 BiLSTM 变体：deep-BiLSTM。它是由 k 个 BiLSTM：$BiLSTM_1...BiLSTM_k$ 堆叠而成。$BiLSTM_l$ 的输出 $BiLSTM_l(x_1...x_n, 1),...,BiLSTM_l(x_1...x_n, n)$ 会成为的 $BiLSTM_{l+1}$ 的输入。Irsoy 和 Cardie(2014)[14]的研究表明，这种堆叠是有效的。本文中，选择 k=3，将 3 个 BiLSTM 堆叠在一起。为了方便表达，之后的内容中会使用 BiLSTM 代称 deep-BiLSTM。

2.7.2 特征编码

借助于 BiLSTM 强大的特征组合能力，可以只用单词本身和它的词性这两个特征，取代传统的近百个特征模板做法。。

给定一个输入句子 $\mathbf{s} = w_1...w_n$ 以及对应的词性序列 $t_1..t_n$，它们相关联的词向量分别是 $v_1^{word}...v_n^{word}$ 和 $v_1^{tag}...v_n^{tag}$。将单词向量 v_i^{word} 和词性向量 v_i^{tag} 连接：

$$x_i = v_i^{word} \oplus v_i^{tag} \tag{2.8}$$

向量序列 $x_1...x_n$ 之后会输入到 BiLSTM 中。

词向量只是独立地编码单词和词性，而没有考虑到单词实际所处的上下文。通过公式 2.6，就可以得到包含了上下文的特征向量 r_i。对于句子 $\mathbf{s} = w_1...w_n$，r_i 就是单词 w_i 的特征表示，它既编码了单词本身，又通过考虑之前的所有单词 $w_1...w_i$ 和之后的所有单词 $w_i...w_n$，近似表示了单词周围无限窗口大小的上下文，而 McDonald 的特征模板仅考虑了单词之前和之后的一个词。

2.7.3 维度缩减

Kiperwasser 和 Goldberg (2016)[14]直接用 BiLSTM 的输出计算依存弧的得分，但 Dozat 和 Manning 并没有延续之前的做法，而是在正式打分前用多层感知机（MLP）对 r_i 进行维度缩减。对于每个 BiLSTM 输出 r_i，它需要携带足够的信息来识别其核心词，找到所有修饰词，排除所有的非修饰词。但实际上，如果只计算一条依存弧 $h \rightarrow m$ 的得分，并不关注 h 的核心词信息以及除 m 外的其它修饰词信息，也不关注 m 的修饰词信息。然而，r_i 却携带了多余的信息，这会给单纯的打分任务造成干扰。以打分为目标，对 r_i 进行维度缩减，有助于剥离和当前打分任务无关的信息，如图 X 所示。对于每个 r_i，使用 ReLU-MLPs 生成两个新的向量：

$$h_i^{arc-dep} = MLP^{arc-dep}(r_i) \tag{2.9}$$

$$h_i^{arc-head} = MLP^{arc-head}(r_i) \tag{2.10}$$

$h_i^{arc-dep}$ 只保留了单词作为修饰词时需要的信息，$h_i^{arc-head}$ 只保留了单词作为核心词时需要的信息。

2.7.4 双仿射打分

有了特征表示，就可以给依存弧打分。之前的做法是用一个 MLP 打分，而 Dozat 和 Manning 使用了双仿射变换来计算得分。一般的仿射变换形如：

$$s_i = W \times h_i + b \tag{2.11}$$

双仿射变换是在仿射变换的基础上，再做一次仿射变换：

$$s_i = H^{arc-head} \times W \times h_i^{arc-dep} + H^{arc-head} \times b \tag{2.12}$$

其中，W 是参数矩阵，b 是偏置向量，$H^{arc-head}$ 是由所有的 $h_i^{arc-head}$ 堆叠而成的矩阵，$\times$表示矩阵相乘。s_i 是单词 w_i 作为其它单词的修饰词的得分向量，如：s_{ij} 表示依存弧 $j \to i$ 的得分。实际上，如图 X 所示，可以一次性得到所有依存弧的得分:

$$S = (H^{arc-dep} \oplus 1) \times U \times (H^{arc-head})^{\top} \tag{2.13}$$

$H^{arc-dep}$ 由所有的 $h_i^{arc-dep}$ 堆叠而成，$H^{arc-dep} \oplus 1$表示给 $H^{arc-dep}$ 右边添加一列，且值全为 1，U 是由 W 和 b 堆叠成的新参数矩阵，$(H^{arc-head})^{\top}$ 表示对矩阵进行转置变换。S 是所有依存弧的得分矩阵，S_{ij} 表示依存弧 $j \to i$ 的得分。之后通过 Eisner 算法就可以得到得分最大的句法树了。

第 3 章 依存句法分析主动学习

3.1 主动学习

3.1.1 主动学习简介

主动学习是人工智能的一个子领域，其最核心的思想是：如果（机器或深度）学习算法能够自主挑选用于训练的数据，即便使用的数据变少，模型也可以取得更高的准确率 (Settles, 2010)[16]。鉴于有监督的学习算法（Surpervised Learning）为了有良好的表现，经常要在成千上万的有标签实例上进行训练，该假设就成了学习算法梦寐以求的性质。有些时候，实例的标签只需要花费很少代价（设置不需要代价）就能得到。例如，人们给购物网站上的商品进行评价，并不需要网站付出什么代价。网站可以使用这些评价给用户推荐更优质的商品。在该场景下，人们免费提供了标签，但是对于许多其它更复杂的监督学习任务，获得有标签实例是非常困难、耗时、昂贵的。以下举一个例子：

在语音识别中，精确地标注语音需要受过专业训练的人，且极度耗时。Zhu (2005)[17]报告到，如果对单词进行标注，那么花费的时间比实际的音频时间要长十倍（一分钟的语音要用十分钟标注），而标注音素甚至要花上 400 倍的时间（一分钟的语音要用近七个小时标注）。当遇到稀有的语言以及方言时，情况会变得更糟糕。

解决标签瓶颈问题的一种最直接方法是将无标签实例变成有标签的。主动学习旨在让模型获得较高的准确率的同时，尽可能少地标注实例，因此最小化了获得有标签数据的代价。在许多机器学习和深度学习问题中，数据十分丰富，但是标签却很稀少，在这些领域中使用主动学习是十分有动力的[16]。在自然语言处理领域，有标签数据是稀少的，无标签文本却唾手可得。由于能够减少标注代价，主动学习获得了人们广泛的关注，大量的任务也以研究主动学习为目标。

3.1.2 主动学习与半监督学习

主动学习和半监督学习(Semi-Supervised Learning)都尽量利用无标签的数据，这

两个领域间一些重叠的概念值得思考。例如，一个非常基础的半监督技术是自训练（Self-Training）。模型首先用少量的有标签数据进行训练，然后被用来分类无标签的数据。模型最确定的无标签实例以及预测出的标签会被一起添加到训练数据中，然后继续训练，该过程会不断重复[16]。主动学习中，则存在着完全相反的情况，那就是不确定性采样（Uncertainty Sampling）。模型最不确定的无标签实例会被选出来，标注后再用于下一轮的训练。可以发现，主动学习和半监督学习从相反的方向来解决同一个问题：如何运用无标签数据。当半监督学习强化模型已知的信息时，主动学习试图利用模型未知的信息。目前，有一种声音认为主动学习是一种过时的技术，半监督学习可以取代主动学习。实际上，完全可以将两种学习方法的优点结合起来。

3.2 基于池的主动学习

3.2.1 基于池的场景

主动学习有不同的应用场景，本文所描述的是基于池（Pool-Based）的场景 (Lewis and Gale, 1994)[18]，如图 3.1 所示。现实中的许多问题中，大量的无标签数据可以被迅速收集到，这激发了基于池的主动学习方法。在基于池的场景，最初有一个较小的有标签种子数据集$\mathcal{L}$和一个大规模的无标签数据池$\mathcal{U}$。对于池中的所有实例，都会评估它们的信息量。之后，一个或多个富有信息的实例从$\mathcal{U}$中被选出。每一个被选出的实例，都会通过一个“数据库”查询它的真实标签，这样的“数据库”并不是真实存在的，通常是由标注者标注出实例的真实标签。被打上标签的实例随后会被添加到$\mathcal{L}$中。依存句法分析中，基于池的主动学习的具体工作流程如下：

（1）在当前的$\mathcal{L}$上训练出一个新的句法分析器。

（2）分析$\mathcal{U}$中的所有句子，并选出一个最富有信息的任务集$\mathcal{U}'$。

（3）人工标注无标签实例：$\mathcal{U}' \rightarrow \mathcal{L}'$。

（4）扩展有标签数据集：$\mathcal{L} \cup \mathcal{L}' \rightarrow \mathcal{L}$。

上述步骤循环多次直到一个预先定义的停止标准被满足。例如，$\mathcal{U}$中没有数据了或不再包含富有信息的实例了。

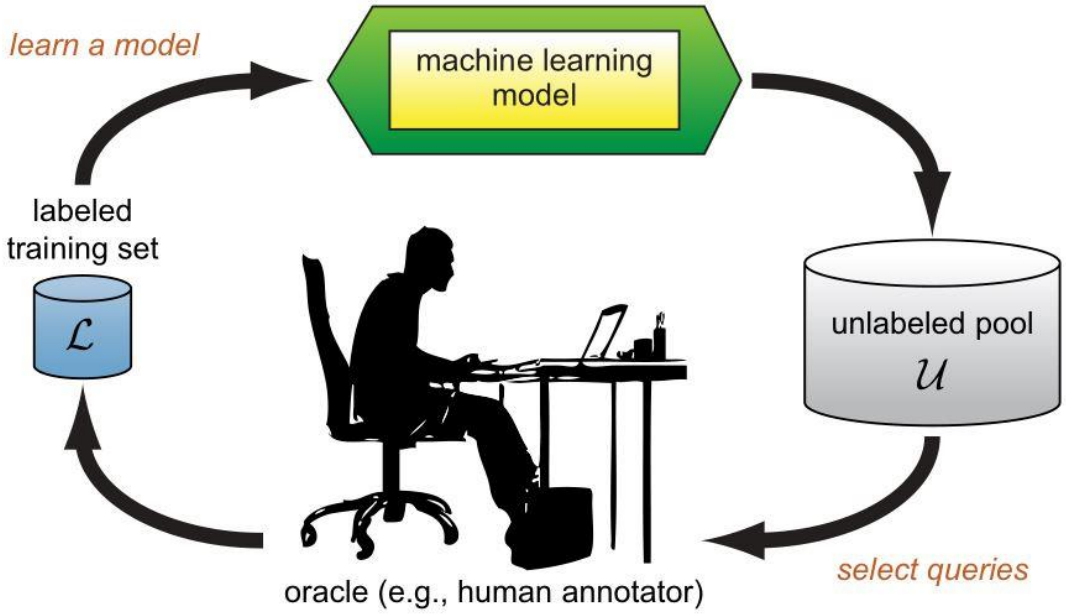


图 3.1：基于池的主动学习框架，该图引用自[16]

3.2.2 查询策略

所有的主动学习场景都需要评估无标签实例的信息量，评估信息量的方法一般称为查询策略（Query Strategy）。目前已有许多查询策略被提出，而最广泛使用的策略有两种：基于不确定性采样的和基于委员会投票的。

委员会投票方法需要维护一个委员会$\mathcal{C}=\{\theta_1,...,\theta_C\}$，每个委员会成员$\theta_i$都是是用当前有标签数据集$\mathcal{L}$训练出的模型[16]。每个委员会成员都会预测无标签实例的标签，对标签的预测相当于投票，一个实例的投票结果越不一致，那么该实例就越富有信息。

最简单且最被广泛使用的查询策略是不确定采样。在该策略中，主动学习算法会查询分类预测结果最不确定的实例。这种方法需要先定义一个不确定性度量。如果一个实例的预测标签的不确定性度量越小，那么模型就对于该标签分类结果越不确定。从而可以认为，该实例拥有模型缺少的信息，标注该实例并将其添加到$\mathcal{L}$中，就可以提升模型的性能。尽管不精确的度量标准会导致糟糕的结果，但因为该策略十分简单，本文使用该策略进行任务选择。

在不确定性采样策略中，熵经常被用作不确定性度量：

$$Confidence(x) = -\sum_{i} p(y_i \mid x) \log p(y_i \mid x) \tag{2.14}$$

y_i 取尽实例 x 所有可能的标签。信息论中，熵代表了一个分布包含的信息量，因此，在主动学习中，熵就作为一种不确定性度量。对于二元分类问题，基于熵的不确定性采样等价于选择后验概率最接近 0.5 的实例[16]。基于熵的方法也可以很简单地推广到序列和树等结构更复杂的问题上。但是，在这些更复杂的问题中，直接枚举所有可能的结构是不现实的，通常会通过最佳标签进行评估。如果对于一个实例的标签预测最不确定，就优先查询它的真实标签：

$$y^* = \arg\max_{y} p(y \mid x) \tag{2.15}$$

$$x^* = \arg\min_{x} p(y^* \mid x) \tag{2.16}$$

y^* 是实例 x 最可能的预测标签，x^* 是其最可能标签的预测概率最小的实例。

3.3 依存句法分析主动学习

本节先介绍了边缘概率是如何在主动学习中使用的，然后具体介绍了三种标注方式：句子级全标注，单一依存弧级局部标注和批量依存弧级局部标注。同时列举出了每种标注方式采用的不确定性度量，并在下一章中通过模拟实验研究它们实际的效果。

3.3.1 边缘概率

以往的依存句法分析主动学习把一个句子作为任务选择的单元，并且使用最佳树的概率作为不确定性度量，一棵句法树的概率通过以下公式得到：

$$p(y \mid s) = \frac{e^{score(s,y)}}{\sum_{y' \in \mathcal{Y}(s)} e^{score(s,y')}} \tag{2.17}$$

然而，观察标注者的实际标注过程可以发现，依存弧才是标注的基本单元。本文以依存弧为主动学习的查询目标，不再局限于完整的句子，并将这种方式称为“依存弧级局部标注”，图 3.2 展示了一棵局部标注的句法树，图 3.3 是其对应的森林，模型可以通过该森林进行学习。

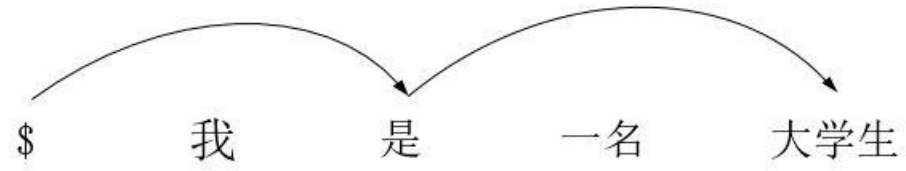


图 3.2：局部标注的句法树

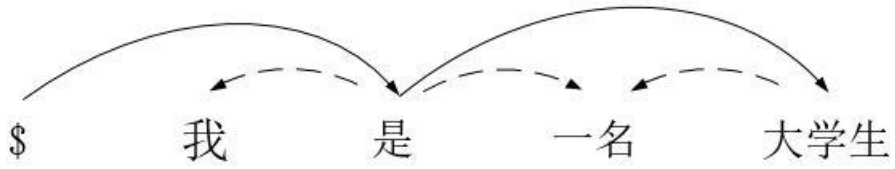


图 3.3：从图 3.2 的局部标注句法树对应的句法森林

在局部标注中，Sassano 和 Kurohashi (2010)[6]使用分类器的得分作为短语对的不确定性度量，得分越小就越不确定。而为了评估模型对一条依存弧的不确定性，Mirroshandel 和 Nasr (2011)[19]先选出 n 个最佳的句法分析结果，然后使用熵作为度量。Marcheggiani (2014)[20]的研究显示，序列标注主动学习中，局部标签的边缘概率是一个有效的不确定性度量。在依存句法分析中，一条依存弧的边缘概率是包含该依存弧的所有合法句法树的概率和：

$$p(h \rightarrow m \mid s) = \sum_{y \in \mathcal{Y}(s): h \rightarrow m \in y} p(y \mid s) \tag{2.18}$$

直觉上，比起之前基于局部分类得分或者是选出 n 个最佳的句法分析的方法，把边缘概率作为不确定性度量来评估一条依存弧，在原则上是更合适的。因为它考虑到了所有合法的句法分析。因此，本文将边缘概率运用到局部标注的不确定性度量中。

3.3.2 句子级全标注

在每轮主动学习中，通过不确定性采样，句子级全标注会选出 K 个句子，随后标注出它们的完整句法树结构。接下来，介绍两种不确定性度量。

在全标注中，可以用最佳树的概率作为不确定性度量。最佳树概率需要求解所有句法树才能得到，可以认为是一种全局的度量。因此，比起只选出 n 个最佳的句法树，该度量是更适当的。理论上，如果 $p(y^*|s)$ 越小，那么模型就越需要 s 包含的信息。但实际上，句子可能的句法树数量是随句子长度指数级增长的，这就导致句子越长，最佳树的概率就越小。如果直接使用树概率，那么长句子会优先被选出。使用下方的归一化策略可以缓解该问题：

$$Confidence(s) = \sqrt[n]{p(y^*|s)} \tag{2.19}$$

如前文所说，依存弧的边缘概率是一种全局的度量，尽管弧只是句子的一部分，使用边缘概率在全标注中仍然是有效的。对于最佳的树，不再直接计算它的概率，而是考虑所有弧的边缘概率和。与归一化的树概率一样，需要考虑句子长度的影响，直接平均进行归一化即可：

$$Confidence(s) = \frac{\sum_{h \to m \in y^*} p(h \to m|s)}{n} \tag{2.20}$$

上述两种不确定性度量分别叫做归一化树概率、平均边缘概率。

3.3.3 单一依存弧级局部标注

标注出句子的完整结构是一件费时费力的事，单一依存弧级局部标注则避免花费多余的代价在一些简单的依存关系上，不再考虑整个句子，而去关注那些极有价值的依存弧。。每轮主动学习从$\mathcal{U}$中选择 M 个最不确定的单词，在查询阶段中标注出这些单词修饰的核心词。不同于句子级全标注的是，由于一个句子中有价值的单词并不会在一轮主动学习中全都选出，$\mathcal{L}'$在被添加到有标签数据集$\mathcal{L}$中后，还会被放回到池$\mathcal{U}$中，这就保证了新任务还可以从$\mathcal{L}'$中选出。

借鉴复杂结构中的主动学习做法，可以使用一个修饰词最可能的核心词的概率作为不确定性度量：

$$\begin{aligned} h^0 &= \arg\max_h p(h \to i \mid s) \\ Confidence(s,i) &= p(h^0 \to i \mid s) \end{aligned} \tag{2.21}$$

i表示修饰词，h^0是i最可能的核心词。$p(h^0 \to i \mid s)$越小，模型就越不确定i的核心词。

然而，依存弧是一种简单的结构，熵与最佳标签两种做法可以进行组合。除了最可能的核心词，还考虑第二可能的核心词：

$$\begin{aligned} h^1 &= \arg\max_{h \neq h^0} p(h \to i \mid s) \\ Confidence(s,i) &= p(h^0 \to i \mid s) - p(h^1 \to i \mid s) \end{aligned} \tag{2.22}$$

h^1是i第二可能的核心词，这里用两个概率的差来评估不确定性。概率差越大，模型对i越不确定。

上述两种不确定性度量分别叫做边缘概率最大、边缘概率差。

3.3.4 批量依存弧级局部标注

单一依存弧级局部标注由于只关注依存弧，这就导致一个句子中有价值的单词可能会在主动学习的不同轮次中选出。这就意味着，标注者需要多次理解同一个句子，但每次只标注一个单词，这造成了认知努力的巨大浪费。实际上，标注者完全可以一次性标注句子中的多个单词。

受到 Flannery 和 Mori (2015)[5]工作的启发，本文使用了基于批量依存弧级局部标注的主动学习，这是一种对句子级全标注和单一依存弧级局部标注进行折中的方案。批量依存弧级局部标注中，每次从$\mathcal{U}$ 中选择出 K 个最不确定的句子，然后从每个句子中选出 r%的最不确定的单词，标注者可以一次标注出这些不确定的单词。本文参考了 Li et al. (2016)[21]中的三种相关做法。

第一种做法是，由于同时考虑到了句子和单词，可以先用平均边缘概率选出句子，然后，对于每个选出的句子，使用边缘概率差选出r%的单词。

在第二种做法中，先选出单词，再选出句子。一种做法是用边缘概率差选出 r%的最不确定单词。对于句子，用边缘概率差的平均值作为不确定性度量，不过只考虑那些属于 r%的单词。

最后一种做法和第二种几乎一样。不过，在评估一个单词时，用的是最佳树中指向它的那条依存弧的边缘概率，而非单一依存弧级局部标注中采用的两种度量。

上述三种度量称为平均边缘概率及概率差、边缘概率差、平均边缘概率。

第 4 章 模拟实验

为了评估一个不确定性度量的效果，需要进行实验。但是，对于一个实验来说，让标注者进行实际标注，在时间和金钱两个方面付出的代价太大了。现实做法是，在一份标注好的树库上进行主动学习实验。查询实例的真实标签时，只要使用标注好的标签即可，这就是模拟实验的含义。

4.1 实验设置

本文使用中文宾州树库（CTB）来处理中文，用宾州树库来处理英文（PTB）。对两个数据集，本文遵循了标准的数据划分。同时，使用 Penn2Malt 程序，基于它默认的寻找核心词的规则，将树库的括号结构转变成了依存结构。因为要在每轮主动学习中从头开始训练新的句法分析器，本文只保留长度在 50 以内的句子。表 4.1 显示了数据集的统计信息。

表 4.1：实验数据统计

		训练集	开发集	测试集
CTB	句子数	14,176	803	1,910
	单词数	312,005	20,454	50,319
PTB	句子数	38,992	1,700	2,416
	单词数	902,004	40,117	56,684

遵循之前局部标注主动学习的惯例，实验设置如下：训练集最前面的 500 个句子作为种子标签数据。在全标注主动学习中，每轮有 K=500 个新的句子被选出来标注。在单一依存弧级局部标注中，每轮主动学习选出并标注 M=10000 个单词。在批量依存弧级局部标注中，每轮选出 K=500 个句子，并且每个中文句子中选出 r=20%的单词，每个英文句子中选出 r=10%的单词。

对于不同标注方式和不确定性度量，统一使用 UAS 进行评估。

4.2 句子级全标注对比单一依存弧级局部标注

本节对全标注和局部标注进行了比较，局部标注采用的是单一依存弧级的。

遵循之前的工作，为了能够直接比较两种标注方式，本文使用标注的依存弧数量作为 x 轴来代表标注代价。使用随机选择的全标注结果作为基线，并用在所有数据上训练的模型的 UAS 作为上界。

中文 CTB 上的结果如图 4.1 所示。

对于全标注，从图中可以看出，尽管在某些点处十分接近，归一化树概率的性能大部分阶段都胜过了随机选择的基线。而平均边缘概率几乎在所有阶段都优于其它两个不确定性度量。

对于局部标注，两个度量的折线纠缠在了一起，这说明两个置信度度量间的差距非常小。在某些时候，它们的 UAS 甚至超过了用所有数据训练出来的句法分析器，这可以归因于训练时的扰动。

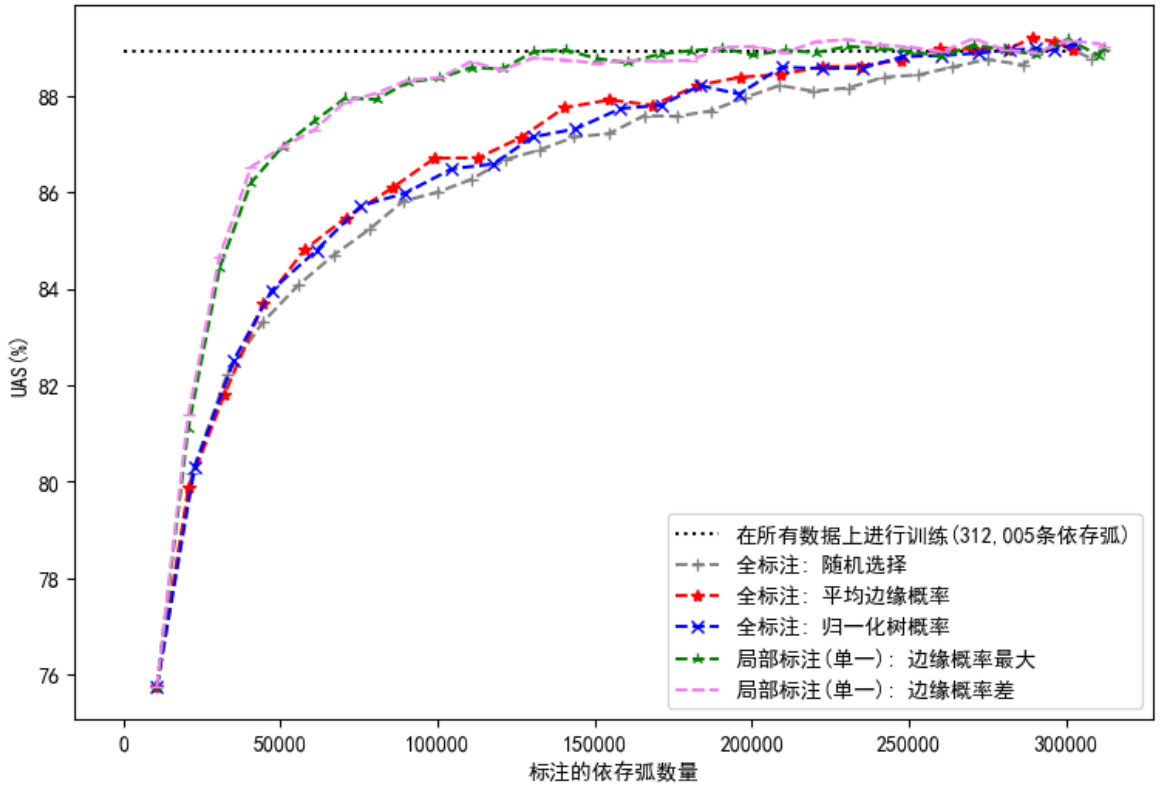


图 4.1：CTB-dev 上全标注和局部标注的对比

英文 PTB 上的结果如图 4.2 所示。折线图的结果表明，几个不确定性度量在英文上的表现和在中文上的基本一致。

对于全标注，与之前不同的是，平均边缘概率在一开始稍落后于归一化树概率，但之后两者间的差距逐渐缩小，平均边缘概率开始赶上甚至超过了归一化树概率。

对于局部标注，结果与之前一致，边缘概率最大和边缘概率差依旧交缠在了一起，很难判断那种度量是更优的。

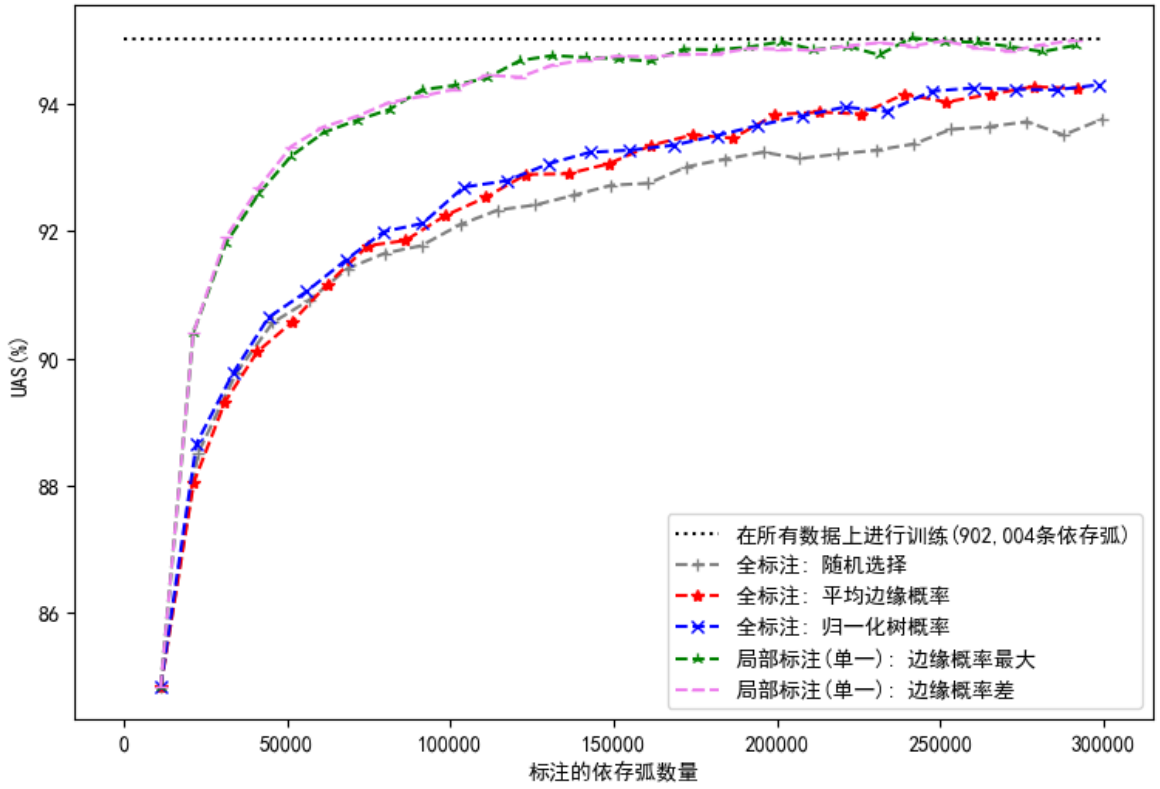


图 4.2：PTB-dev 上全标注和局部标注的对比

两个数据集的结果表明，仅从需要标注的依存弧的数量方面来看，比起全标注，局部标注确实可以减少标注代价。而为了获得和在所有数据上训练的分析器一样的准确率，在中文上，仅需要 42%的依存弧，在英文上，仅需 27%的依存弧。综上所述，依存弧级局部标注极大减少了标注代价，在实际标注中，是一种值得应用的策略。

4.3 单一依存弧级局部标注对比批量依存弧级局部标注

为了避免频繁切换到不同的句子标注，进行了批量局部标注的实验。同上述实验一样，使用标注的依存弧数量作为 x 轴，用所有数据训练出的模型作为上界，但是使用边缘概率差的单一依存弧级局部标注结果作为基线。

中文 CTB 上的结果如图 4.3 所示。对于批量标注，从图中可以看到，三个不确定性度量之间很难进行比较，因为三条折线几乎交缠在了一起。

对比单一标注和批量标注，在初期，批量的三个不确定性度量超过了边缘概率差的单一标注。而在之后，也有不错的表现，最终只落后了一点。

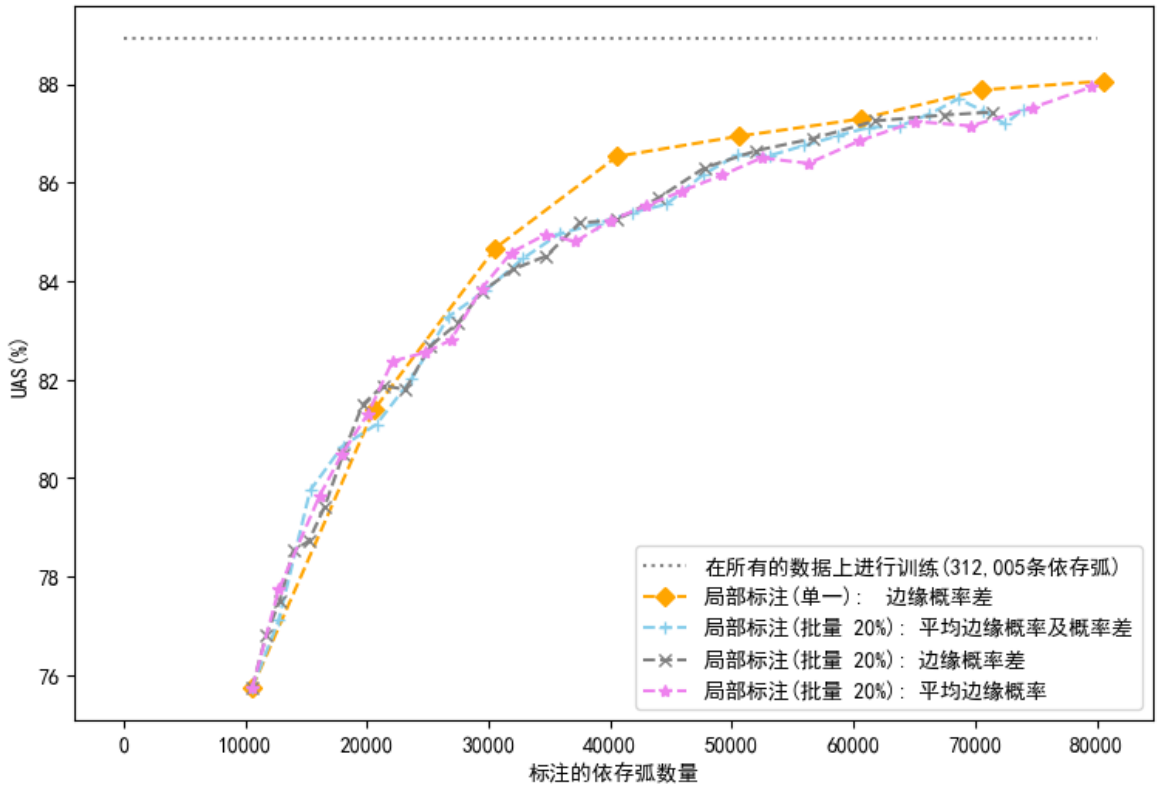


图 4.3：CTB-dev 上单一和批量局部标注的对比

英文 PTB 上的结果如图 4.4 所示。可以说，和在中文上的没有什么差别。

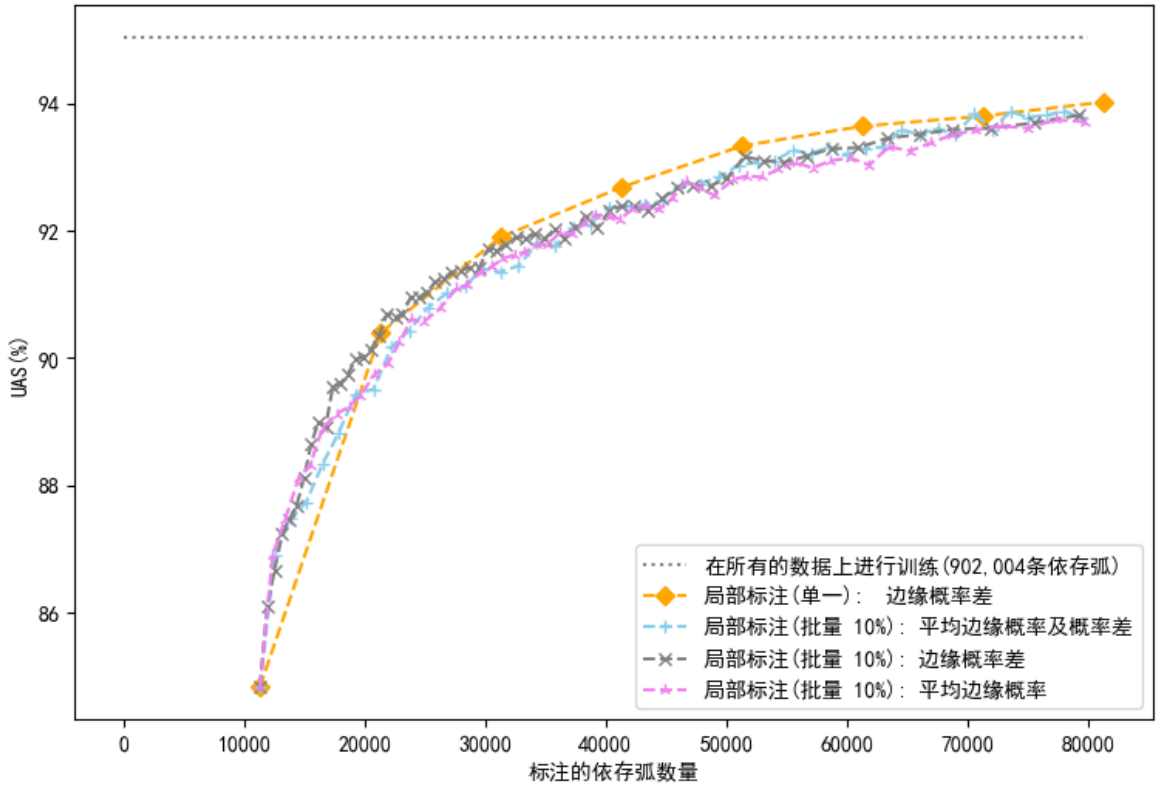


图 4.4：PTB-dev 上单一和批量局部标注的对比

尽管批量的局部标注在性能上比单一局部标注落后了一点，但是从批量依存弧级局部标注减少的标注代价来看，这是完全值得的。并且，如果有更多的语料，两种标注方法间的差距甚至有希望抹平。

4.4 测试集上的结果

前文描述的是在开发集的结果，为了更贴近真实情况，在测试集上进行了评估。结果如表 4.2 所示：

表 4.2：测试集上的结果

	中文 CTB		英文 PTB	
	标注的弧数量	UAS	标注的弧数量	UAS
所有的数据	312,005	89.19	902,004	95.17
全标注（随机选择）	154,581	87.71	264,905	93.62
全标注（平均边缘概率）	140,381	87.63	199,467	93.72
局部标注（单一）	60,560	87.79	61,307	93.71
局部标注（批量）	70,688	87.70	63,021	93.72

或许可以通过到达上界 UAS 时需要标注的依存弧数量来比较三种标注方式，但是由于批量依存弧级局部标注无法满足该要求。因此用获得比在所有数据上训练出的模型小大约 1.5%的 UAS 时，需要标注的依存弧数量来对三种标注方式进行比较。

对于全标注来说，使用平均边缘概率是最佳的。在中文上，它比随机选择的方式少标注了 9.2%的依存弧，在英文上，少标注了 24.7%的依存弧。

对于单一依存弧级局部标注，选边缘概率差作为最佳度量。在中文上，它比平均边缘概率的全标注需要的依存弧少了 56.9%，在英文上，则少了 69.3%。

对于批量依存弧级局部标注，使用的是平均边缘概率及概率差的结果。比起平均边缘概率的全标注，在中文上减少了 49.6%的标注量，在英文上减少了 68.4%的标注量。比起单一依存弧级局部标注，在中文上，需要额外 16.7%的依存弧，在英文上，需要额外 2.8%的依存弧。

综合开发集和测试集上的结果，依存弧级局部标注比句子级全标注减少了一半以上的需要标注的依存弧数量。而批量局部标注虽然比单一局部标注需要更多的依存弧，但由于避免了多次标注同一个句子，实际上，仍然减少了标注代价。

第 5 章 总结和展望

5.1 总结

主动学习是解决依存句法分析领域数据稀缺问题的一种有效方法。之前的工作认为认为训练数据必须是全标注的，而最近的研究表明使用局部标注的数据进行训练的模型仍然有相当强的竞争力。本文的主要工作如下：

（1）应用了比较先进的深度双仿射神经句法分析器进行依存句法分析，该分析器的结构比较简单，但能取得较高的准确率。

（2）使用依存弧作为主动学习的基本单元，并提出了几种不确定性度量。

（3）进行了模拟实验，系统地比较了全标注和局部标注。实验结果显示，在依存弧数量方面，比起全标注，局部标注在中文上减少了 49.6%的标注量，在英文上减少了 68.4%的标注量。这表明，在主动学习中使用依存弧级局部标注是可行的、有效的。

5.2 展望

首先，本文基于不确定性采样来选择标注任务，并且假定最不确定的任务就是最富有信息的。实际上，如果这些任务都属于一类语言现象，会出现一种“重复学习”的情况，这并不能最大程度地提升模型。在任务选择中，可以考虑把不确定性和代表性结合起来，选择那些既让模型难以进行判断又能代表常见语言现象的任务。

其次，在主动学习过程中，上一轮的句法分析器用于挑选标注任务后，就将其抛弃了，下一轮中又从头开始训练新分析器。计算机视觉中的主动学习则只稍微调整模型的参数，然后将分析器继续用于下一轮训练。希望在依存句法分析主动学习中，也能使用这种技巧，减少训练时间。

参考文献

[1] Fredrik Olsson. A literature survey of active machine learning in the context of natural language processing [R]. Technical report, Swedish Institute of Computer Science, 2009.

[2] Min Tang, Xiaoqiang Luo, and Salim Roukos. Active learning for statistical natural language parsing [A]. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics [C], 2002: 120–127.

[3] Rebecca Hwa. Sample selection for statistical parsing [J]. Computional Linguistics, 2004, 30(3): 253–276.

[4] Avihai Mejer and Koby Crammer. Training dependency parser using light feedback [A]. In Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies [C], 2012: 488–497.

[5] Daniel Flannery and Shinsuke Mori. Combining active learning and partial annotation for domain adaptation of a japanese dependency parser [A]. In Proceedings of the 14th International Conference on Parsing Technologies [C], 2015: 11–19.

[6] Manabu Sassano and Sadao Kurohashi. Using smaller constituents rather than sentences in active learning for japanese dependency parsing [A]. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics [C], 2010: 356–365.

[7] Ryan McDonald, Koby Crammer, and Fernando Pereira. Online large-margin training of dependency parsers [A]. In Proceedings of the 43th Annual Meeting of the Association for Computational Linguistics [C], 2005: 91–98.

[8] Ryan McDonald. Discriminative Training and Spanning Tree Algorithms for Dependency Parsing [D]. PhD thesis, University of Pennsylvania, 2006.

[9] Jason Eisner. Bilexical grammars and their cubic-time parsing algorithms [A]. In Advances in Probabilistic and Other Parsing Technologies [M], 2000: 29–62.

[10] Yue Zhang and Joakim Nivre. Transition-based dependency parsing with rich non-local features [A]. In Proceedings of the 49th Annual Meeting of the Association for

Computational Linguistics: Human Language Technologies [C], 2011, 2: 188–193.

[11]Danqi Chen and Christopher D. Manning. A fast and accurate dependency parser using neural networks [A]. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) [C], 2014: 740–750.

[12]Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. Transition-based dependency parsing with stack long short-term memory [A]. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing [C], 2015, 1: 334–343.

[13]Timothy Dozat, Christopher D. Manning. Deep biaffine attention for neural dependency parsing [A]. In Proceedings of the 5th International Conference on Learning Representations [C], 2017.

[14]Ozan Irsoy and Claire Cardie. Opinion mining with deep recurrent neural networks [A]. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) [C], 2014: 720–728.

[15]Eliyahu Kiperwasser and Yoav Goldberg. Simple and accurate dependency parsing using bidirectional LSTM feature representations. Transactions of the Association for Computational Linguistics, 4:313–327,2016.

[16]Burr Settles. Active Learning Literature Survey [R]. Technical Report 1648, University of Wisconsin-Madison, 2010.

[17]Xiaojin Zhu. Semi-Supervised Learning with Graphs [D]. PhD thesis, Carnegie Mellon University, 2005.

[18]David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers [A]. In Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval [C], 1994: 3–12.

[19]Seyed Abolghasem Mirroshandel and Alexis Nasr. Active learning for dependency parsing using partially annotated sentences [A]. In Proceedings of the 12th International Conference on Parsing Technologies [C], 2011: 140–149.

[20]Diego Marcheggiani and Thierry Artières. An experimental comparison of active

learning strategies for partially labeled sequences [A]. In proceedings of the 2014 conference on empirical methods in natural language processing [C], 2014: 898–906.

[21]Zhenghua Li, Min Zhang, Yue Zhang, Zhanyi Liu, Wenliang Chen, Hua Wu, and Haifeng Wang. Active learning for dependency parsing with partial annotation [A]. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics [C], 2016: 344-354.

致谢

毕业设计已经基本完成，至此，我才意识到大学生涯即将结束。回顾过去的四年，有太多的感慨，太多的留念，太多的遗憾。

首先，要感谢李正华老师在论文选题和实验过程中对我的指导。李老师是一位要求严格但实际相处过程中会感到十分和蔼的人。在实现深度双仿射句法分析器出现问题时，李老师给予的鼓励重振了我的信心。李老师深耕序列标注和句法分析多年，希望能在研究生阶段从他身上学到更多东西。

其次，张宇学长在实验和代码方面给我提供了很多建议，周厚全学长在论文书写和 GPU 服务器使用方面提供了许多帮助，在此向两位学长表达真诚的谢意。

最后，感谢我的父母对我二十多年来的培育，感谢他们对我的包容和支持，感谢他们对我的勉励和督促。