北京大学

本科生毕业论文

题目：基于图重写的语义分析方法实现

Implementation of Graph Rewriting Based Semantic Parsing

姓　　名：王希豪

学　　号：1600012794

院　　系：信息科学技术学院

专　　业：计算机科学与技术

导　　师：孙薇薇 副教授

二〇二〇年五月

北京大学本科毕业论文导师评阅表

学生姓名	王希豪	学生学号	1600012794	论文成绩	良
学院 (系)	信息科学技术学院			学生所在专业	计算机科学与技术（科学方向）
导师姓名	孙薇薇	导师单位/所在研究所	王选计算机研究所	导师职称	副教授
论文题目（中、英文）	基于图重写的语义分析方法实现 Implementation of Graph Rewriting Based Semantic Parsing				

导师评语

（包含对论文的性质、难度、分量、综合训练等是否符合培养目标的目的等评价）

基于图的语义分析是近些年自然语言处理学界兴起的一种句子级语义表示方法。面向语义图，本文复现了一种基于神经图重写的分析系统，该系统的主要模块包括：（1）短语结构分析器、（2）图重写文法的自动抽取器和（3）语义解析模块。该系统较为复杂，包括较为复杂的符号计算与深度学习计算模块。

论文的研究工作反映出该同学已经掌握了扎实的专业基础知识和编程实现能力，同时具有一定的研究能力。

导师签名：

年　　月　　日

版权声明

任何收存和保管本论文各种版本的单位和个人，未经本论文作者同意，不得将本论文转借他人，亦不得随意复制、抄录、拍照或以任何方式传播。否则，引起有碍作者著作权之问题，将可能承担法律责任。

摘要

人工智能是计算机领域最前沿的研究方向。最初，人工智能研究的目标是创造出能够像人一样思考的计算机程序，能否通过图灵测试是判断程序是否为人工智能的标准。如今，人工智能领域相比较于设计智能程序，更关注数据分析、处理和表示。

表示学习是人工智能的重要任务，研究如何自动从数据中提取出有效的特征。人工神经网络是一种通过拟合非线性变换、将原始数据的特征直接表示为高层次抽象特征的表示学习模型，在许多计算机领域都有重要的应用。本文介绍了人工神经网络在自然语言处理的两个任务中的应用。

自然语言处理是利用计算机自动分析和处理人类语言数据的学科。将人类语言语句表示成计算机能够理解和计算的形式，是自然语言处理的重要问题。语言学家们从词语、句法和语义等多个层次研究了语言的形式化表示。

句法分析是将句子表示成为树形结构、研究句子构成的自然语言任务。句法分析任务主要有三种方法：基于状态转移的方法、基于翻译的方法和基于因子分解的方法。本文介绍了一种利用人工神经网络获取数据特征的基于因子分解的句法分析方法。

语义分析是另一种自然语言形式化表示的任务，从句子深层语义的角度考虑句子的表示。语义图是利用图结构表示句子语义的方法。图结构能够直观地表示复杂的语义关系，图分析也有成熟的算法，因此，语义图成为了近几年来语义分析的热点问题。本文介绍了一种基于图重写的语义分析方法。这种方法根据句法分析的结果，通过图重写的方式生成得到语义图。

本文说明了两种模型的工程实现细节，在 DeepBank 语料库数据上验证了模型效果。同时，我们还针对不同句法结构对基于图重写的语义分析方法的影响进行了实验，说明了 DeepBank 等树库的句法树包含一定的语义信息。

关键词： 句法分析, 语义图, 人工神经网络

ABSTRACT

Artificial Intelligence is one of the most advanced research fields of computer science. At first, the goal of AI research is to create a computer program that can think like human beings. Whether a program can pass Turing Test is the standard to judge whether it is AI. Nowadays, compared with the design of intelligent programs, the field of Aritificial Intellence pays more attention to data analysis and processing.

Representation Learning is an important task of Artificial Intelligence. It learns how to extract effective features from data. Artificial neural network is a kind of Representation Learning model, which directly represents high-level abstract features of original data by fitting nonlinear transformation. It has important applications in many computer fields. This paper introduces the application of neural network in two tasks of Natural Language Processing.

Natural Language Processing is a subject that uses computers to analyze and process human language data automatically. It is an important problem in Natural Language Processing to express human language sentences in the form that computers can understand and calculate. Linguists have studied the formal representations of language from the perspectives of words, syntax and semantics.

Syntactic analysis is a natural language task to express sentences as tree structures and to study sentence construction. There are three main methods of syntactic analysis: transition-based approach, translation-based approach and factorization-based approach. This paper introduce a factorization-based method that extracts features from data by using neural network.

Semantic analysis is another task of formal representation of natural language, which represents sentences from the perspective of deep semantics. Semantic graph is a method that uses graph to express sentence semantics. Graph can express complex semantic relations intuitively, and there are plenty of mature algorithms of graph processing. Therefore, semantic graph has become a hot issue of semantic analysis in recent years. In this paper, we introduces a graph rewriting based semantic parsing method. According to the result of syntactic analysis, this method generates semantic graph by graph rewriting.

This paper describes the engineering implementation details of the two models, and evaluates the models on DeepBank corpus. At the same time, we also do experiments on the influence of different syntactic structures on the graph rewriting based semantic parsing

method, which shows that the syntactic trees of DeepBank and other treebanks contain some semantic information.

KEYWORDS: syntactic analysis, semantic graph, artificial neural network

全文目录

第一章 引言 **1**
1.1 研究背景 1
1.1.1 自然语言理解 2
1.1.2 自然语言生成 4
1.2 本文工作 5

第二章 相关工作 **6**
2.1 语义的形式化表示 6
2.1.1 基于图的语义表征 6
2.1.2 英语资源语义 7
2.2 形式文法 7
2.2.1 上下文无关文法 8
2.2.2 超边替换文法 8
2.3 人工神经网络 10
2.3.1 多层感知机 10
2.3.2 长短期记忆网络 11

第三章 基于连续词串分类的句法分析模型 **13**
3.1 句法分析 13
3.1.1 基于状态转移的方法 13
3.1.2 基于翻译的方法 14
3.1.3 基于因子分解的方法 14
3.2 模型实现 15
3.2.1 连续词串特征提取 15
3.2.2 CYK 算法 16

第四章 基于图重写的语义图分析模型 **18**
4.1 同步超边替换文法 18
4.2 文法规则抽取 18
4.3 模型实现 20
4.3.1 特征提取 20

4.3.2 基于树的束搜索算法 21

第五章 实验 **23**

5.1 数据处理 23

5.2 句法分析结果 23

5.3 语义图分析结果 25

第六章 总结 **26**

参考文献 **27**

本科期间的主要工作和成果 **29**

致谢 **30**

第一章 引言

1.1 研究背景

当人类有意识地开始用声音、手势和图形表达特定含义、进行信息交换时，最初的语言就诞生了。原始语言能表达的含义和信息是极其有限的。在人类从原始社会进入奴隶社会的过程中，语言的表达能力逐渐增强，从只能利用单个符号表达简单含义发展成为了能够利用符号组合表达丰富意义的复杂系统。几千年来，伴随着历史、文化的变迁，语言也不断地发展，最终形成了如今的稳定系统。

从个体角度来说，语言是人类天生的学习、产生和理解特定类型信息的心理能力，与人脑的生理机能息息相关，心理学家通过心理实验研究语言产生的心理机制和人的认知规律；从社会和历史的角度来说，语言是特定团体间约定俗成的交换信息的方式，受到地域历史和文化的影响，社会学家和历史学家通过横向和纵向比较语言差异来研究社会和历史的变化规律；从符号的角度来说，语言是特定符号按照特定规则组合表达信息的结构系统，语言学家通过研究不同语言现象的规律来研究语言的内在特性和规律。

传统的语言学家们对语言的研究是直觉性的，主要依赖自身的知识和灵感。在分析语言数据时，语言学家们有意识或无意识地发现新的语言现象，通过研究这种语言现象的产生机制和原理，再发掘更多符合这一语言现象的例子来支持自己的理论。然而，这种研究方式存在三个问题：

1. 研究门槛高，依赖个人经验和知识储备。
2. 数据庞大，人的精力是有限的，语言数据却是无限的。语言学家能够观察到的特定语言现象的例子是极其有限的，只占语言数据极小的部分，约束了对该语言现象产生机制的研究。
3. 不同语言现象之间的研究割裂。由于语言现象并不是系统性地根据固定框架发现的，不同语言现象之间很难产生联系，甚至很多语言现象之间相互矛盾，影响了对语言现象、乃至语言的深层机制的研究。

随着计算机的发明和进步，对大量数据的自动处理成为了可能。于是，语言学家们开始研究如何利用计算机自动分析和处理自然语言数据，这类研究称为自然语言处理（Natural Language Processing，简称 NLP）或者计算语言学（Computational Linguistics，简称 CL）。

现代计算机都是冯诺依曼结构的计算机，计算机以二进制为计算基础，根据存储

在系统内的预先编写的计算机程序对数据进行计算得到结果。想要利用计算机处理自然语言数据，就必须先将语言转化成计算机能够理解的形式，也就是形式表示（Formal Representation）。

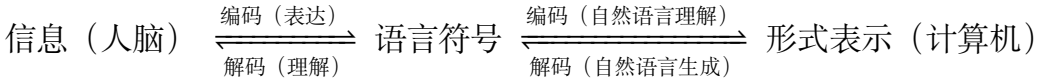


图 1.1 信息、语言符号和形式表示的关系

如图1.1所示，如果把语言看做是一种编码体系：发送方按照规则将信息编码成为语言符号，再通过说话、书写等方式传输信息；接收者则通过聆听、阅读等方式接受语言符号，再根据语言规则进行解码，获取其中的信息。

从编码和解码的角度考虑自然语言处理，可以将任务分成自然语言理解和自然语言生成两类。

1.1.1 自然语言理解

自然语言理解（Natural Language Understanding，简称 NLU）的目标是分析自然语言数据并将其转化成形式表示。

几十年来，计算语言学家们发明了许多语言的形式表示方式。“词袋子”模型是最简单的表示形式之一。这种模型从符号角度将词语当作是基本单位，把句子看做是词的集合，不考虑词和词之间的内在联系和句子的深层含义。在此基础上，计算机科学家们提出了分布式表示（Distributed Representation）技术 [1, 2]，根据“上下文相似的词语具有相似的特征”的假设，把词语嵌入到高维向量空间中，以词和词在向量空间中的距离来表示它们之间的相似相关程度，这一技术称为词嵌入（Word Embedding）。Mikolov 等 [3] 提出的 `word2vec` 就是一种高效的词嵌入方法。

词的分布式表示能够很好地表达词汇信息，在自然语言处理中已经被广泛应用，但其中只蕴含少量的句子信息。为了解决这一问题，使得词向量能够蕴含更多句子层面的信息，Peters 等 [4] 和 Devlin 等 [5] 先后提出了 ELMo（Embeddings from Language Models）和 BERT（Bidirectional Encoder Representations from Transformers）模型，但却没有从根本上解决句子的形式表示问题。

语言学家们认为，句子是最小的独立表达意义的单位 [6]，想要理解自然语言需要从句子层面分析语言。一些语言学家从句法层面考虑句子的结构。

句法是特定语言构造句子的原则和过程 [7]。

短语结构（Phrase Structrue）和依存结构（Dependency Structure）是两种主要的句法结构 [8]。

- 短语结构又称为成分结构（Constituency Structure）。连续的一组词语可以看做是一个单位，称为成分。成分分析逐步地将句中相邻的成分组合，直到最后只剩下由句中所有词语组合成的最大成分。分析的过程可以表示成为一棵二叉树，就是句子的短语结构。
- 依存结构是根据依存关系对句子分析得到的树形结构。依存关系是一种词和词之间的二元关系，两个词中处于中心的称为头词（head），处于旁位的称为依存者（Dependent）。依存分析根据词和词之间的依存关系，将句子组合出一棵树，从而突出句子中处于中心地位的词语

句法结构在一定程度上表现出了词和词、词和句子之间的关系，却依旧不能够表达句子的深层含义。图1.2和1.3分别是是乔姆斯基名言：

Colorless green ideas sleep furiously.

的短语结构树和依存结构树。这个句子符合现有的句法规则，通过分析可以得到句子的句法结构，但这个句子在语义上却充满矛盾。

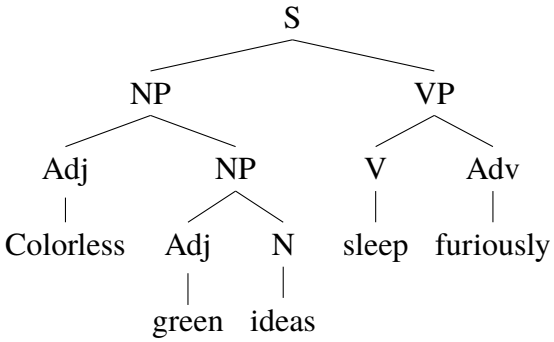


图 1.2 “Colorless green ideas sleep furiously”的短语结构。

为了挖掘句子的深层含义，计算语言学家们提出了语义分析的任务，使得计算机能够理解句子的意义。

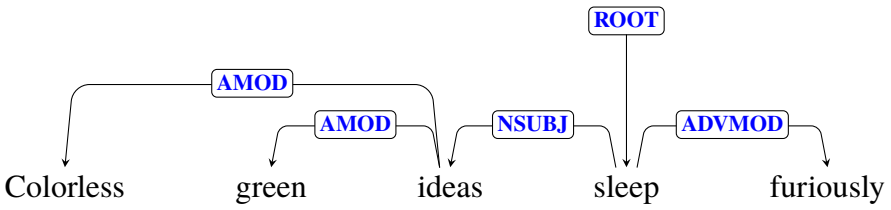


图 1.3 “Colorless green ideas sleep furiously”的依存结构。

基于一元谓词逻辑的逻辑表达式是一种基础的语义表示方式。图1.4是句子“I am a student.”的逻辑表达式。可以看到，逻辑表达式将句子表示成为变量之间的简单一元和二元关系的组合。

$$\exists e,x\ Being(e) \wedge Subject(e,Speaker) \wedge Identity(e,x) \wedge Student(x)$$

图 1.4 “I am a student.”的逻辑表达式。

和逻辑表达式相比，复杂的图结构可以表现更多的信息。基于图的语义表示能够直观地表达出复杂的语义关系，是目前主流的语义表示形式之一。

语义分析不仅为计算机理解和分析自然语言提供了支持，同时，也为语言学家们发掘、分析语言现象提供了帮助。特定的语义表示为自然语言构建了大的框架，在这一框架下发现的各种语言现象反过来使得框架更加完善。

1.1.2 自然语言生成

自然语言生成（Natural Language Generation，简称 NLG）的目标是将形式化的数据自动转化成为符合人类习惯的自然语言语句。和自然语言理解相比，自然语言生成是一个更加复杂和困难的任务。

机器翻译、自动摘要系统、基于知识库的问答系统等都是自然语言生成的重要任务。语义分析在这些任务中都有重要的应用。

机器翻译（Machine Translation）是利用计算机自动地将一种语言（称为源语言）的语句转化为另外一种语言（称为目标语言）的任务。目前，最流行的机器翻译技术是基于序列到序列（Sequence to Sequence，简称 seq2seq）技术，将源语言的句子编码成为长度固定的向量，再将其解码为目标语言的句子。Bahdanau 等 [9] 提出的基于循环神经网络（一种人工神经网络，见2.3.2）的序列到序列机器翻译模型是一种经典的极其翻译模型。Song 等 [10] 提出的图到序列模型则在序列到序列模型的基础上，应用了语义分析的结果，将源语言的语义图转化为目标语言的句子。

自动摘要系统（Automatic Abstracting System）需要分析篇章的整体结构，得到篇章的语义表示，再将其转化为简短的摘要。基于知识库的问答系统（Question Answering over Knowledge Base）同样需要分析问题的语义结构，将其转化为特定的语义表示，再根据知识库生成答案。

1.2 本文工作

本文的主要工作是复现了 [11] 提出的基于超边替换文法的语义图分析方法。这是一种基于句法分析结果，以图重写的形式生成语义图的技术。在此基础上，还比较了两种不同的句法树对于语义分析的影响。

本文一共分成六章，其余各章节安排如下：

第二章介绍了相关工作的情况。对于语义的形式化表示，主要介绍了本文关注的基于图的语义表示形式，特别是概念语义图和英语资源语义。随后解释了上下文无关文法和超边替换文法的基本概念，本文实现的同步超边替换文法分析模型就是通过两者文法的结合实现的。最后介绍了人工神经网络的两个模型，多层感知机模型和长短期记忆模型在本文的模型中都起到了重要作用。

第三章首先举例说明了基于状态转移的方法、基于翻译的方法和基于因子分解的方法三种常见的句法分析方法的基本原理，再说明了本文实现的基于连续词串的句法分析模型的原理。

第四章解释了同步超边替换文法的原理，并说明了如何根据句子的短语结构抽取语义规则，以及在同步超边替换文法的框架下对句子进行语义分析的方法。

第五章设计实验展示了本文所述模型在具体语料上的表现，并且比较了两种不同结构的推导树对实验结果的影响。

第六章总结了全文，指出了存在的问题和未来的研究方向。

第二章 相关工作

2.1 语义的形式化表示

语义分析是将给定句子转化成特定语义表示的过程，需要根据给定的语义表示形式进行。词语的分布式表示就是一种简单的语义表示，根据“分布相似的语言现象具有相似含义”的假设，从而得到词汇语义。然而，分布式表示所蕴含的句子信息是极其有限的，想要挖掘句子的深层信息，需要考虑句子层面的语义表示信息。

句子有许多复杂的语义现象，其中，句子中主要动词、形容词等和其支配的带有语义角色的成分之间的关系是一种广泛存在的基础语言现象。这种由谓词和它支配的一组带有语义角色的成分（称为论元）构成的语义结构称为谓词论元结构（Predicate-Argument Structure）。谓词论元结构以句子的主要谓词为核心，考察谓词的主体、客体、旁体部分之间的关系，从而展示出句子所蕴含的事件语义。对句子的谓词论元结构和相关语义成分的抽象分析称为语义角色标注（Semantic Role Labeling，简称 SRL）。

谓词论元结构表示的事件语义是句子最基础的语义现象之一，自然语言语句还有许多其他层面的语义。想要更加全面地表示句子的语义，需要使用更加复杂的模型。图就是一种用来表示丰富语义信息的有力工具。

2.1.1 基于图的语义表征

图是由点和边组成的数学模型，可以用二元组 $G =< V, E >$ 表示，其中 $V \neq \emptyset$ 是点集，E 是边集。如果 E 是无序积 $\{\{a, b\}|a, b \in V\}$ 的多重子集，那么图 G 是无向图，边 $e \in E$ 称为无向边；如果 E 是笛卡尔积 $V \times V = \{(a, b)|a, b \in V\}$ 的多重子集，那么图 G 是有向图，边 $e \in E$ 称为有向边 [12]。

依存分析得到的句法结构必须是树形结构，但其关注的词与词之间的依存关系是一种二元关系。对于图来说，每条边就代表了两个结点之间的二元关系，用图来表示句中词与词之间的依存关系是很自然的想法。打破依存分析的树形限制得到语义依存图，就能够表现更复杂的语义信息。

到目前为止，我们所谈论的语义分析都是从词语的层面出发，但是，有些抽象的语义信息并不依赖于具体某个词，只与整个句子的结构有关。为了更加全面的表示句子的语义结构，语言学家们提出了概念语义图分析的方法。概念语义图的每个顶点并不是具体的某个词，而是从句子中抽取出来的抽象的概念（Concept）。这些概念并不是单纯的符号，往往具有非常复杂的属性。有些概念与句中词汇具有非常强的关联性，

对应于某个词语或某个短语的一个含义；有些概念与句中词汇没有明显的关联性，蕴含了句子中的抽象语义属性。

常见的概念语义图分析体系有抽象语义表示（Abstract Meaning Representation，简称 AMR）[13] 和基础依存结构（Elementary Dependency Structure，简称 EDS）[14]。

2.1.2 英语资源语义

20 世纪 80 年代产生的头词驱动的短语结构语法（Head-driven Phrase Structure Grammar，简称 HPSG）理论是一种以类型特征结构（Typed Feature Structure，简称 TFS）为计算基础，用来表示词汇或短语语义的语言学框架。类型特征结构具有可组合性，在判断两个子短语结构能否合并和计算两个子短语结构合并得到的新结构上极其方便 [15]。

Delph-in[①]联盟在 HPSG 的基础上，构建了英语资源语法（English Resource Grammar，简称 ERG）。同时，以最小递归语义（Minimal Recursion Semantics，简称 MRS）为框架进行对 ERG 的逻辑语义进行了标注，构建了大型语料库 Redwoods[14]，所标注出来的逻辑语义就是英语资源语义（English Resource Semantics，简称 ERS）。

MRS 是基于带量词的一阶逻辑的计算组合语义的方法，具有完备的逻辑解释，能够进行跨语言分析 [16]。

ERS 除了遵循 MRS 的规则外，还可以转化成为语义图。ERS 最常见的语义图表示方式有基础依存结构（Elementary Dependency Structure，简称 EDS）和依存最小递归语义（Dependency Minimal Recursion Semantics，简称 DMRS）。不同的语义图有不同的特点，但其蕴含的语义特征是不会变的。本文的模型是根据 EDS 语义图实现的，但也可以很容易地迁移到 DMRS 等其他结构的语义图上。

2.2 形式文法

形式语言（Formal Language）是特定字母表上字符串的集合，形式文法（Formal Grammar）是生成形式语言的规则。

乔姆斯基提出的乔姆斯基分层体系将形式语言分成了表2.1中的四类：

本文的模型就是基于上下文无关文法以及受其启发所产生的超边替换文法来实现的。

① http://www.delph-in.net/wiki/index.php/Home

类别	文法	自动机[2]
3 型	正则文法	有限状态自动机
2 型	上下文无关文法	下推自动机
1 型	上下文相关文法	线性有界自动机
0 型	无限制文法	图灵机

表 2.1 乔姆斯基分层体系

2.2.1 上下文无关文法

上下文无关文法（Context-free Grammars，简称 CFG）是乔姆斯基分层体系中的 2 型文法。

任意上下文无关文法 G 都可以用一个四元组 $< N, T, S, P >$ 来表示 [17]，其中：

- N 是生成字符串过程中使用的临时符号的集合，这些符号称为非终结符（Non-terminal Symbols）。
- T 是特定的字母表，生成的字符串由 T 中的字母组成，这些字母被称为终结符（Terminal Symbols）。
- $S \in N$ 是起始符号，字符串的派生从起始符号出发。
- P 是派生规则的集合。每条派生规则都是 $\alpha \rightarrow \beta$ $(\alpha \in N, \beta \in (N \cup T)^*)$ 的形式，其中，箭头左边的符号被称为左部符号（Left-hand Symbols，简称 lhs），箭头右边的符号被称为右部符号（Right-Hand Symbols，简称 rhs）。

从起始符号 S 出发，根据 P 中的派生规则，逐步替换非终结符。在经过有限步数的替换之后，能够得到完全由终结符组成的字符串。这一字符串就属于当前上下文无关文法所表示的语言。

文法 $G =< \{S, T\}, \{a, b\}, S, \{S \rightarrow T, T \rightarrow aTb|\epsilon\} >$ 就是表示形式语言 $\{a^n b^n | n \geq 0$ 的上下文无关文法。

句子的短语结构分析假设句子是由特定的上下文无关文法生成的，分析得到的句法树展示了根据这一上下文无关文法派生得到句子的过程，句法树每个内部结点都是一条派生规则。

2.2.2 超边替换文法

乔姆斯基分层体系中的所有文法都只是符号层面的文法，核心思想是符号重写。Drewes 等 [18] 受到上下文无关文法的启发，提出了基于图重写的超边替换文法（Hyperedge Replace Grammar，简称 HRG）。

我们首先给出超边的定义：

定义 2.1 超边（*Hyperedge*）是普通边的扩展概念。普通的边能够连接且只能连接两个

结点，但一条超边可以只连接一个结点或连接三个及以上的结点。

带有超边的图就是超图。在本文中，我们所关注的超图都是边标记的有向超图。语义图的每个结点都含有语义信息，在利用超边替换文法对语义图重写时，可能出现节点信息冲突的情况。为了解决这一问题，我们将节点信息转移到边上，得到了边标记的有向超图，具体定义为：

定义 2.2 一个边标记的有向超图（*Edge-labeled Directed Hypergraph*）是一个四元组 $H =< V, E, l, X >$。其中，V 是有限的顶点集合；$E \subseteq V^+$ 是有限的超边集合；$l : E \rightarrow L$ 为每条超边分配了一个标签（*label*）；$X \subseteq V^*$ 是有序的点集合，该集合中的点被称为外点（*external node*），给出了超边替换时和图其他部分的连接情况。

和上下文无关文法类似，超边替换文法也可以用一个四元组 $< N, T, S, P >$ 表示，其中，N 是非终结符的集合，T 是终结符的集合。和上下文无关文法不同的是，超边替换文法中，终结符和非终结符都是带有标签的边。S 是起始符号，P 是规则集合，但每条规则的左部都是带有标签的超边，右部是一张超图。

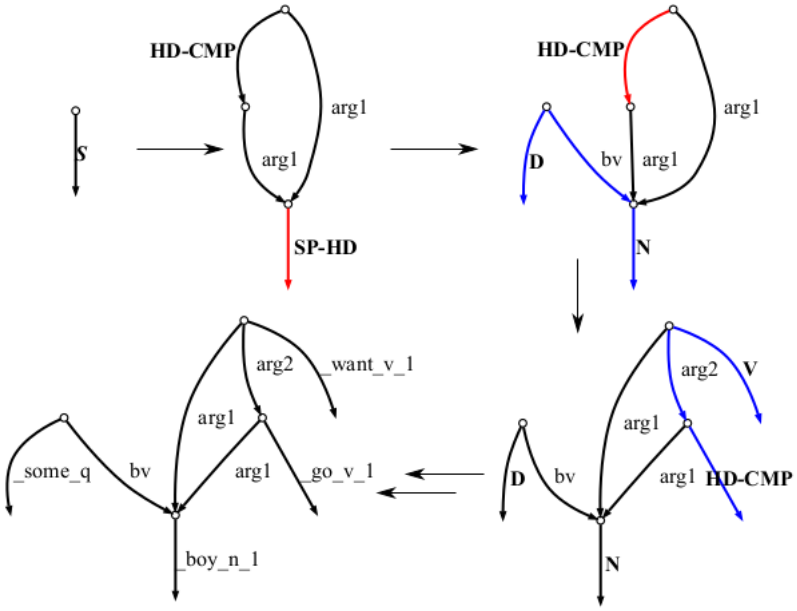


图 2.1 超边替换文法生成“Some boys want to go.”语义图的过程 [11]。图中红色的超边表示在下一步中要被替换为蓝色超边构成的子图。

图2.1是根据特定超边替换文法生成得到句子“Some boys want to go.”语义图的过程。我们从初始符号 S 出发，每次挑选一条边，将其替换为一张子图，直到所有边都是终结边为止。

2.3 人工神经网络

从古希腊时代开始，创造出能够像人类一样思考的机器就是发明家们的梦想 [19]。20 世纪中叶，计算机发明之后，许多计算机科学家致力于创造能够像人类一样思考的程序，即人工智能（Artificial Intelligence）。

有些计算机科学家人为，可以通过模拟人思考的过程和机制，来创造出能像人一样思考的人工智能程序。McCulloch、Pitts [20] 从人类神经元的工作机制得到启发，抽象得到了“M-P 神经元模型”。这一模型中，每个神经元将来自其他神经元的数据加权相加，再和设定的阈值比较，如果大于阈值，就通过激活函数将数据向其他神经元传输。由 M-P 神经元按照顺序连接起来形成的网络模型就是人工神经网络。尽管人工神经网络是对人类神经系统的模拟，但实际上是以实数向量作为输入和输出的数学模型，是对向量空间中的复杂函数的拟合。

最初的人工神经网络模型称为感知机（Perceptron），由两层 M-P 神经元连接组成，只能解决与、或、非问题等线性可分问题 [21]。由 Werbos [22] 最先提出，之后由 Rumelhart 等 [2] 重新发明的反向传播算法（Backpropagation Algorithm），解决了人工神经网络模型无法拟合复杂函数的问题，使得人工神经网络的能力得到了增强，可以拟合较为复杂的函数。近年来，随着计算机性能的提升和数据量的增加，人工神经网络模型被广泛应用于表示学习中。

人工神经网络是极其复杂的数学模型，目前没有人能在数学上说明人工神经网络内部的具体工作机制。但是，人工神经网络在数据的表示和高层次特征的提取上有着非常优秀的效果。本文所实现的模型就利用了循环神经网络对数据进行了特征的提取。

2.3.1 多层感知机

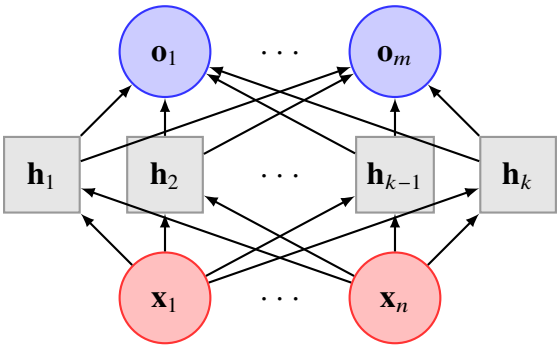


图 2.2 多层感知机模型图示

多层感知机（Multi-layer Perceptron，简称 MLP）是一种基础的人工神经网络模型。多层感知机分为输入层、隐藏层和输出层三部分，每层都包括一定数目的 M-P 神经元

模型，每个神经元都接受来自上一层所有神经元的数据，在加权计算后经过激活函数激活，再向下一层所有神经元传送数据。随着多层感知机隐藏层数目的增加，多层感知机能够拟合的函数就越复杂，训练的难度也随之增加。在实际应用中，隐藏层只需要一层就能够得到较好的效果。图2.2展示的就是只有一层隐藏层的多层感知机模型。

多层感知机模型是常见的构造分类器（Classifier）的技术。分类器是用于数据分类的模型，可以看做是从数据集合 X 到标签集合 L 的函数 $f : X \to L$。分类器根据样本 x 的特征，从有限的标签集 L 中选择一个标签赋予 x。

若标签集合 L 的大小为 m，可构造输出层有 m 个神经元的多层感知机。输出层的每个神经元都对应一个标签，将样本的特征输入多层感知机，得到每个标签的分数，再从中挑选分数最高的标签赋予 x。

2.3.2 长短期记忆网络

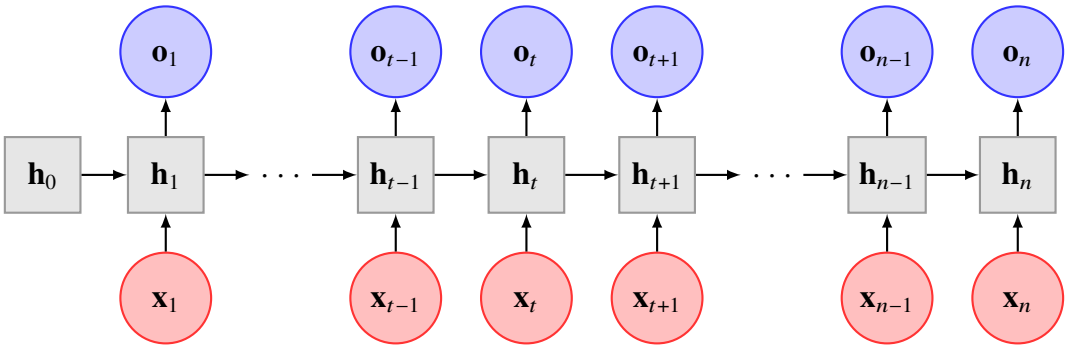


图 2.3 循环神经网络模型图示

循环神经网络（Recurrent Neural Network，简称 RNN）是一类专门用于处理序列化数据的人工神经网络模型 [2]。

如图2.3所示，循环神经网络的初始隐状态为 $\mathbf{h}_0$，输入数据为长度固定的向量序列 $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{n-1}, \mathbf{x}_n\}$。从初始状态出发，依次向循环神经网络传入数据。在 t 时刻，循环神经网络根据数据 $\mathbf{x}_t$ 和前一时刻计算得到的隐状态 $\mathbf{h}_{t-1}$，计算得到输出向量 $\mathbf{o}_t$ 和新的隐状态向量 $\mathbf{h}_t$。

传统的循环神经网络每次计算隐状态时都会进行复杂的非线性变换，在经过多次变换之后，较早的数据信息很难保留下来。因此，传统循环神经网络很难学到序列中的长距离依赖信息。为了解决这一问题，Hochreiter、Schmidhuber [23] 提出了长短期记忆网络（Long Short-Term Memory Network，简称 LSTM）。

LSTM 在传统循环神经网络的基础上增加了细胞状态 $\mathbf{c}$ 来模拟人类记忆，还引入了输入门、输出门和遗忘门三个控制结构。为了计算时刻 t 的隐状态向量 $\mathbf{h}_t$，首先要

通过输入门 i_t、遗忘门 f_t、前一时刻的隐状态向量 $\mathbf{h}_t$ 和前一时刻的细胞状态 $\mathbf{c}_{t-1}$ 计算得到细胞状态 $\mathbf{c}_t$，再根据细胞状态 $\mathbf{c}_t$ 和输出门 o_t 计算得到隐状态向量 $\mathbf{h}_t$ 并输出。具体的计算过程如下：

$$
\begin{aligned}
f_t &= \sigma(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f) \\
i_t &= \sigma(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i) \\
o_t &= \sigma(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o) \\
\tilde{\mathbf{c}}_t &= \tanh(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{c}_t &= f_t \circ \mathbf{c}_{t-1} + i_t \circ \tilde{\mathbf{c}}_t \\
\mathbf{h}_t &= o_t \circ \tanh(\mathbf{c}_t)
\end{aligned}
$$

其中，$\mathbf{W}$ 和 $\mathbf{U}$ 是参数矩阵，$\mathbf{b}$ 是误差向量，σ 是 Sigmoid 函数 $\sigma(x) = \frac{1}{1+e^{-x}}$。

第三章　基于连续词串分类的句法分析模型

3.1　句法分析

句法分析的目标是分析给定句子的句法信息，最终将句子表示成树状结构。

句法分析是非常复杂的任务，目前进行的相关工作都是根据特定的形式语言文法所展开的。也就是说，是在假设所有句子都符合特定文法规则的前提下，再根据这一文法规则对给定的句子进行分析，最后得到句子的句法结构。1.1.1提到，句法结构主要分为短语结构和依存结构两种，其中，短语结构是在上下文无关文法的基础下分析得到的，依存结构是在依存文法的基础下分析得到的。

目前，句法分析的主要方法有三种：基于状态转移的方法、基于翻译的方法和基于因子分解的方法 [8]。

3.1.1　基于状态转移的方法

基于状态转移（Transition-based）的方法按照语序依次处理句子中的词语，根据当前的状态和扫描的词语，生成相应的句法结构片段或者进行状态的转移。

移进-规约分析（Shift-reduce Analysis）法是一种基于状态转移的句法分析方法，可以用于分析特定上下文无关文法生成的语句。这种方法用栈来维护分析时的状态，每次扫描词语，都会根据栈顶元素来决定是将扫描到的词语（终结符）移进栈中，还是将栈顶的一个或多个元素按照语法规则规约为非终结符。表3.1展示了对句子“Colorless green ideas sleep furiously”进行移进-规约分析的过程，红色词语为栈顶元素。

步数	词语	栈	动作	步数	词语	栈	动作
1	Colorless		移进	9	sleep	NP	移进
2	green	Colorless	规约	10	furiously	NP sleep	规约
3	green	Adj	移进	11	furiously	NP V	移进
4	ideas	Adj green	规约	12		NP V furiously	规约
5	ideas	Adj Adj	移进	13		NP V Adv	规约
6	sleep	Adj Adj ideas	规约	14		NP VP	规约
7	sleep	Adj Adj N	规约	15		S	结束
8	sleep	Adj NP	规约				

表 3.1　“Colorless green ideas sleep furiously”的移进-规约分析过程

同个句子可能有不同的句法结构，图3.1展示了句子“I see a cat in the room”的两种不同的分析结果。两种分析结果对应了两种不同的含义：

- 我看到一只猫在房间里。
- 我在房间里看到一只猫。

句法分析器只会给出一种结果。为了得到可能性最大的分析结果，就需要进行歧义消除（Disambiguate）。基于状态转移的句法分析方法为了消除歧义得到唯一的结果，需要搜索整个解空间。但是，解空间的大小往往是指数级别的，以移进-规约分析为例，对于长度为 n 的句子，每次扫描词语都有移进或者规约两种选择，解空间的大小就是 $O(2^n)$。在实际中，为了解决这一问题，可以使用束搜索（一种基于贪心的搜索剪枝技术，见4.3.2）的方法，对于每一搜索分支，只保留分数最大的 k 个结果，从而大幅度缩减了解空间的大小，提高了分析的效率。

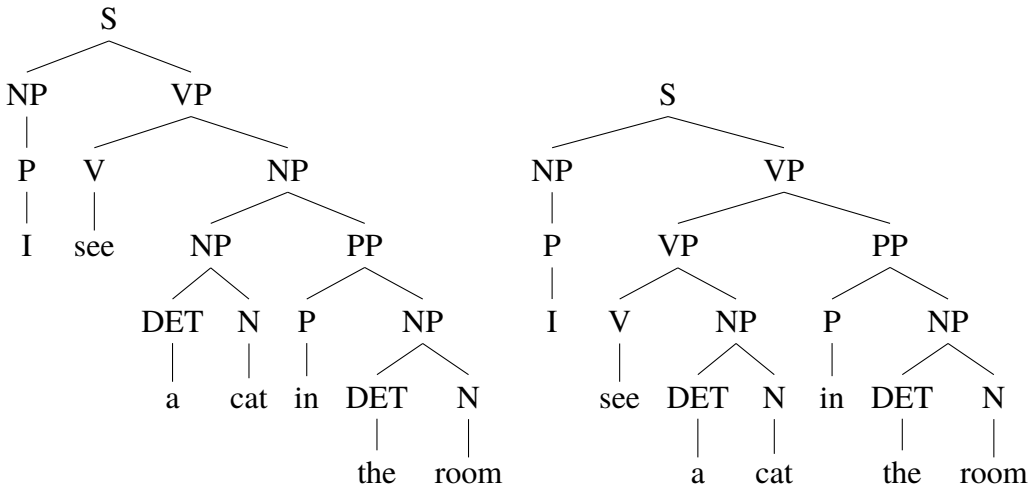


图 3.1　“I see a cat in the room”的两种句法树

3.1.2　基于翻译的方法

基于翻译的句法分析方法受到机器翻译问题的启发，将句法树转化为字符串序列作为目标语言，图3.2是图1.2的短语句法树序列化结果（为了展示方便，将序列换行展示）。之后，就可以利用已有的机器翻译的技术（如序列到序列模型），将句子“翻译”成为句法树。

这种方法对数据量要求极大，人工标注的句法结构数据较少，很难达到训练的要求。此外，单纯使用 seq2seq 模型进行句法分析的性能较差，还需要和其他句法分析方式结合起来。

3.1.3　基于因子分解的方法

基于因子分解（Factorization-base）的方法最先被应用于依存树分析中 [24]。这种方法的核心思想是将句法树分解成为若干子结构，子结构和句法树本身具有同样的性

```
( S ( NP ( Adj Colorless )
         ( NP ( Adj green )
              ( N ideas ) ) )
    ( VP ( V sleep )
         ( Adv furiously ) ) )
```

图 3.2　序列化的“Colorless green ideas sleep furiously”短语句法树

质，可以通过两两不断组合得到句法树。

为了得到最优的句法树，可以利用人工神经网络等工具设计评分函数 Score 对所有小结构打分，同时计算子结构合并的贡献，从而将问题转化成搜索所有可行的句法树结构中分数最高的结构。

最大生成树算法是一种经典的基于因子分解的依存树分析方法。该方法用 $C[i,j,a]$ 表示对词序列第 i 个位置到第 j 个位置的子序列构建依存树，其中所有以 a 为根的依存树的最大分数。用 $S[a,b]$ 表示将以 a 为根的子树和以 b 为根的子结构组合成更大结构的收益。那么就有如下的转移方程：

$$C[i,j,a] = \max_{i \le k < j, i \le b \le j} \begin{cases} C[i,k,a] + C[k+1,j,b] + S[a,b], & a < b \\ C[i,k,b] + C[k+1,j,a] + S[a,b], & a < b \end{cases}$$

根据这一转移方程，可以利用 CYK 算法（一种动态规划算法，见3.2.2）解码构建句法树。

3.2　模型实现

本文使用的句法分析模型是基于因子分解的短语结构树分析方法。模型可以分成特征提取和句法树生成两个部分。首先利用人工神经网络进行特征提取，为句法树每个子结构打分，再用 CYK 算法解码生成短语结构树。

3.2.1　连续词串特征提取

如前文所述，基于因子分解的分析方法最初被应用于依存树分析，为了将这一方法应用到短语结构的分析上，需要考虑短语句法树的子结构。

进行短语句法分析时，我们从词语出发，每次选择相邻的成分将其合并。可以观察到，每个成分有以下几个特点：

1. 由连续的一组词语组成。
2. 可以从中间划分成两个子成分。

由此，我们定义连续词串的概念：

定义 3.1 由句子中第 i 个位置到第 j 个位置的 $j-i+1$ 个词语按照在原句中的顺序排列得到的子序列称为连续词串（*Span*），用符号 $[i:j]$ 表示。

对于长度为 n 的句子来说，有 $[1:1],[2:2]\cdots,[n:n];[1:2],[2:3],\cdots,[n-1:n];\cdots;[1:n]$ 共 $\frac{n(n+1)}{2}$ 个不同的连续词串。短语句法树的每个节点都对应于一段连续词串。忽略单个词语，分析生成短语句法树的过程，就是从 $\frac{n(n-1)}{2}$ 个连续词串中挑选出 $n-1$ 个连续词串，记这个集合为 $\mathcal{S}$，则其中任意一个连续词串都存在且仅存在一种划分方法，满足下列三个条件之一：

- 划分之后得到两个词语。
- 划分之后得到一个词语和一个 $\mathcal{S}$ 中的连续词串。
- 划分之后得到两个 $\mathcal{S}$ 中的连续词串。

在确定子结构之后，我们需要设计评分函数。本文使用 Cross、Huang [25] 提出的 LSTM-Minus 模型对连续词串进行编码，得到每个连续词串的特征。

普通的 LSTM 模型从左到右依次处理词序列，t 时刻计算得到的隐状态 $\mathbf{h}_t$ 实际上是对前 t 个词组成的子序列的编码。LSTM-Minus 模型认为，第 j 个隐状态向量包含的信息，实际上是第 $i(i<j)$ 个隐状态向量包含的信息加上中间的连续词串的信息，那么 h_j-h_i 就是连续词串 $[i+1:j]$ 包含的信息。

循环神经网络是用于处理序列信息的人工神经网络，编码得到的信息会包含序列的顺序信息。如果只是从左向右编码只能得到部分信息，想要编码得到完整的序列信息，还需要逆序训练。因此，本文还利用了双向 LSTM(Bi-directional LSTM，简称 Bi-LSTM）技术，同时从左向右和从右向左对句子进行编码，得到前向编码向量 $(\mathbf{f}_1,\mathbf{f}_2,\cdots,\mathbf{f}_n)$ 和后向编码向量 $(\mathbf{b}_1,\mathbf{b}_2,\cdots,\mathbf{b}_n)$，从而得到连续词串 $[i:j]$ 的特征 $\textsc{Feature}(i,j)=(\mathbf{f}_j-\mathbf{f}_{i-1},\mathbf{b}_i-\mathbf{b}_{j+1})$。

在得到连续词串的特征之后，我们利用多层感知机模型，将特征转化为分数输出，记连续词串 $[i:j]$ 的分数为 $\textsc{Score}(i,j)$ =mlp($\textsc{Feature}(i,j)$)。

3.2.2 CYK 算法

得到每个连续词串的评分之后，需要根据连续词串的分数构造出总分最大的句法树，CYK 算法是一种常用的解码构造句法树的算法。

算法 1 CYK 算法

输入: 二维数组 SCORE，序列长度n
输出: 最大分数 $C[1,n]$，区间划分方式 BACK

```
for i = 1 → n do
    C[i, j] =SCORE[i, j]
for i = n → 1 do
    for j = i → n do
        for k = i → j − 1 do
            if C[i, j] < C[i, k] + C[k + 1, j]+SCORE[i, j] then
                C[i, j] = C[i, k] + C[k + 1, j]+SCORE[i, j]
                BACK[i, j] = k
return C[1, n], BACK
```

CYK（Cocke-Younger-Kasami）算法是一种动态规划算法。这种算法在 20 世纪 60 年代由 Cocke [26], Younger [27] and Kasami [28] 分别独立提出，因此而得名。

CYK 算法可以分析复杂的上下文无关文法，主要思路是对于字符串的每个区间，考虑所有的划分方式，查找是否有对应的文法规则。在本文中，我们构造的短语结构树是二叉树，每个内部节点都只有两个子节点，对于每个区间，只需要考虑划分一次的情况。算法**alg:cyk**是这种情况下 CYK 算法的伪代码。BACK 数组记录了每个区间最优的划分方式，根据 BACK 数组中的数据，我们可以自顶向下构造句法树。

得到句法树后的最后一项工作是为句法树的每个非叶子结点分配标签，用来说明该结点对应的两个子结构的组合方式和句法特征。本文中，我们使用了2.3.1中介绍的多层感知机模型作为分类器，根据每个连续词串的特征进行分类，为其选择分数最高的标签。

至此，我们就实现了基于连续词串的句法分析模型。

第四章 基于图重写的语义图分析模型

4.1 同步超边替换文法

同步超边替换文法（Synchronous Hyperedge Replacement Grammar，简称 SHRG）是基于上下文无关文法和超边替换文法同步分析语义图的方法。

上下文无关文法从起始符号 S 出发，根据规则对符号进行重写，生成仅有终结符的字符串；超边替换文法同样从起始符号 S 出发，根据规则对非终结边进行重写，生成仅有终结边的超图。两者的基本原理是相通的。

根据特定的超边替换文法规则，可以通过图重写的形式生成语义图，解决了语义图的组合问题。但是，超边替换文法没有规定图重写时的组合顺序。为了解决这一问题，考虑到上下文无关文法和超边替换文法的相似性，我们将超边替换文法所描述的语义组合过程和句子的句法组合过程结合，根据同一推导树（Derivation Tree），将上下文无关文法和超边替换文法相关联，得到同步超边替换文法。

推导树是指根据一组上下文无关文法规则推导得到具体句子的树形结构，用于指导超边替换文法图重写的顺序和过程。推导树可以是句子的句法树，也可以是其他具有特定结构的二叉树。本文在实验部分探索了顺序组合的句法树对于基于同步超边替换文法的语义分析方法的影响。

推导树的每个节点都对应一条上下文无关文法规则。从推导树的树根出发，先序遍历每个结点，根据对应的上下文无关文法规则选择合适的超边替换文法规则进行图重写，最后得到语义图。

在实际实现中，考虑到实现的难度，我们并没有采用自顶向下的分析方法，而是从推导树的叶子节点出发，后续遍历整棵推导树，同时，对于每个节点所代表的句法规则，选择合适的 HRG 规则，以子图结合的形式组合得到完整的语义图。

图4.1展示了基于同步超边替换文法的语义图分析过程。

4.2 文法规则抽取

自行向下的语义图生成过程中，存在着不确定性，可能有多种语义组合方式能够得到同样的语义图。与此相反，在给定推导树和语义图的情况下，自底向上的规约分析过程是固定的。为了准确高效地提取出文法规则，本文从叶子结点出发，后序遍历整棵推导树，同时，根据推导树对应的连续词串，在语义图中寻找对应的子图，将子

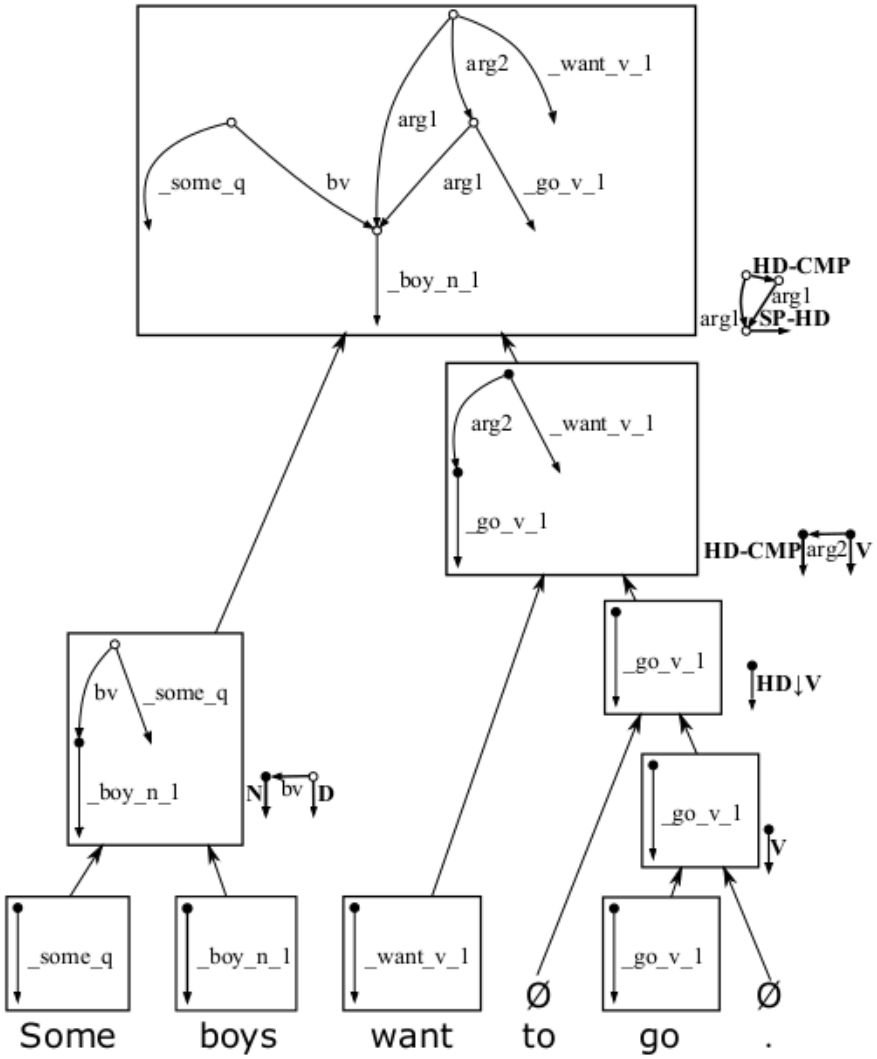


图 4.1 同步超边替换文法自底向上组合生成“Some boys want to go.”的语义图的过程 [11]。

图规约成超边。也就是说，文法规则的抽取过程实际上是超边替换文法组合语义图生成的逆过程。

算法2是文法规则抽取的伪代码。后序遍历推导树 T 上的所有顶点，对于每个顶点对应的语法规则，找到超图上对应的子图，将子图中所有顶点分为内点和外点，就得到了一条规则。在图 G 上将子图缩成一条超边。由于外点是子图和语义图其他部分的连接点，需要保留，所以超边的顶点就是子图的所有外点。

图4.2展示了根据推导树对句子“Some boys want to go.”自底向上进行自动规则抽取的过程。

算法 2 文法规则抽取

输入: 推导树 T，语义图 G
输出: 规则集合 RULES

RULES $\leftarrow \{\}$
for 树结点 $n \in T$ 按后序遍历顺序 **do**
    记 n 对应的文法规则为 $A \rightarrow B + C$
    记 A, B, C 对应的区间为 SPAN_A,SPAN_B,SPAN_C
    SPANS $\leftarrow \{\text{SPAN}_A,\text{SPAN}_B,\text{SPAN}_C\}$

    ALL-EDGES $\leftarrow \{\}$
    ALL-NODES $\leftarrow \{\}$
    for 超边 $e \in G$ **do**
        if SPAN(e) $\in$ SPANS **then**
            ALL-EDGES $\leftarrow$ ALL-EDGES $\cup \{e\}$
            ALL-NODES $\leftarrow$ ALL-NODES $\cup$ NODES(e)
    for 超边 $e \in G$ **do**
        if NODES(e) $\subseteq$ NODES **then**
            ALL-EDGES $\leftarrow$ ALL-EDGES $\cup \{e\}$

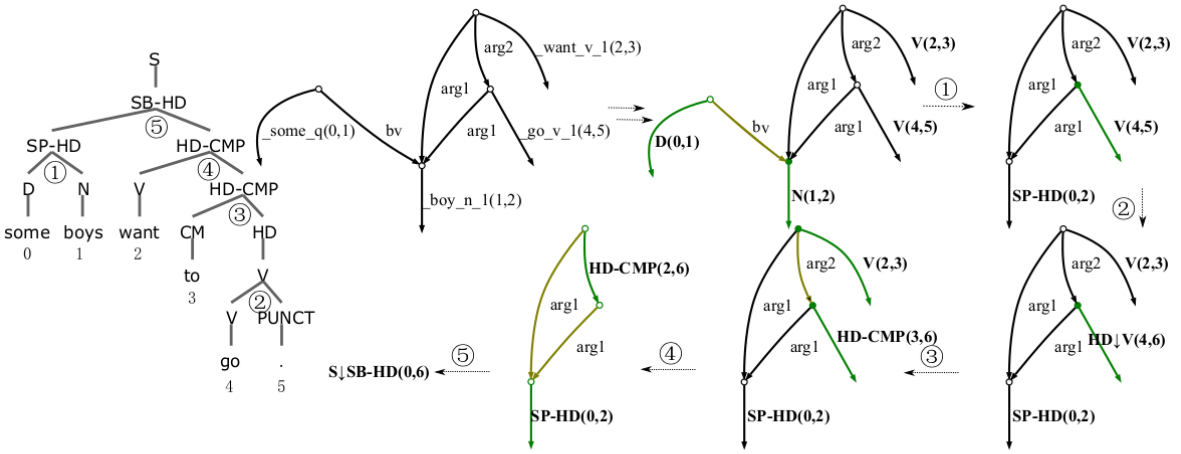
    INTERNAL-NODES $\leftarrow \{\}$
    EXTERNAL-NODES $\leftarrow \{\}$
    for 顶点 $s \in$ ALL-NODES **do**
        if EDGES(s) $\subseteq$ ALL-EDGES **then**
            INTERNAL-NODES $\leftarrow \{s\}$
        else
            EXTERNAL-NODES $\leftarrow \{s\}$
    RULES $\leftarrow$ RULES $\cup \{(A,$ALL-EDGES,INTERNAL-NODES,EXTERNAL-NODES$)\}$
    在超图 G 上将 ALL-EDGES 和 ALL-NODES 缩为一条超边
return RULES

4.3 模型实现

和第三章介绍的基于连续词串的句法分析模型类似，本章介绍的基于图重写的语义图分析模型也可以分成两部分：首先根据当前的推导树结点对应的句法规则挑选出所有可能的语义规则，通过人工神经网络对所有规则打分；再根据推导树的派生过程，搜索生成语义图。

4.3.1 特征提取

推导树的每个内部结点都是一条句法规则，为了得到对应的语义图，我们需要为每个内部结点找到一条对应的语义规则。在4.2中，我们已经介绍了文法规则的抽取算法，为每条句法规则都抽取出了对应的语义规则。但是，仍然有两个问题需要解决：



	①	②	③	④	⑤
Shared LHS	SP-HD	HD↓V	HD-CMP	HD-CMP	S↓SP-HD
RHS (syntax)	D + N	V + PUNCT	CM + HD↓V	V + HD-CMP	SP-HD + HD-CMP
RHS (semantics)	N bv D	V	HD↓V	HD-CMP arg2 V	HD-CMP arg1 arg1 SP-HD

图 4.2　自底向上的规则抽取过程 [11]。

1. 一条句法规则对应多条语义规则。
2. 句法规则没有对应语义规则。

出现频率高的句法规则很容易会对应多条语义规则。此时，我们需要对每条语义规则打分，以便在生成语义图时选择最优的规则。最简单的打分模型就是统计语义规则在语料库中出现的次数，选择出现次数更高的规则。本文则是利用只有一层隐藏层的多层感知机模型对每条规则进行打分。

我们的句法分析器并不能做到绝对精准地分析句子，得到的句法规则可能是错误的，从而没有对应的语义组合规则；即使得到的句法规则是正确的，但是提取得到的规则不可能包含所有的规则，我们得到的句法规则也可能没有对应的语义规则。为了进行语义分析，我们需要为没有对应语义组合规则的句法规则分配一条语义规则。如果该句法规则对应的超边只有一个结点，那么这条边是一条终结边，根据当前推导树结点所对应的短语，为终结边分配一个语义标签；如果句法规则对应的超边有多个结点，不妨设结点数目为 n，那么就构造一张有 $n+1$ 个结点的超图，其中一个点作为中心点，从中心点往每个外点连一条有向边。

4.3.2　基于树的束搜索算法

在为推导树的每条句法规则找到备选的语义规则后，就可以通过搜索得到语义图。

搜索算法分为深度优先搜索（Depth First Search，简称 DFS）和广度优先搜索（Breadth First Search，简称 BFS）两种。搜索的过程可以抽象成为遍历一棵多叉树，多叉树的每个叶子节点对应了一种答案，内部结点的每个子节点都代表了一种可能性。深度优先搜索从根节点出发，在每个内部结点处选择一个搜索方向，直到到达某个叶子结点，将得到的答案和之前的答案比较更新后，再回溯选择其他搜索方向；广度优先搜索从根节点出发后，每次计算并维护同层所有的可能性，逐层计算得到所有的答案，再比较得到最优答案。

无论是深度优先搜索还是广度优先搜索，解空间的大小都是指数级别的，搜索算法如果遍历整个空间来求解最优答案的话，会耗费大量的时间。为了提高搜索算法的效率，往往会采用剪枝（Pruning）技术。使用剪枝技术的搜索算法，不会遍历每种可能的结果。深度优先搜索在每个内部结点考虑下一个搜索方向时，可以先评估该方向结果的可能取值情况，如果预测到这一搜索方向的结果无法超过当前已经取得的最好结果时，就不考虑该搜索方向。这一过程像是在一棵树上将旁枝剪去。

束搜索（Beam Search）则是根据贪心思想剪枝的广度优先搜索算法。传统的广度优先搜索算法要维护搜索树上同一层所有的可能搜索方向，束搜索并不维护所有的可能方向，而是只保留当前评分最高的 k 个可能方向。这样以来，每层维护的搜索方向都不大于 k，总的搜索复杂度降低到了多项式级别。

根据基于同步超边替换文法的分析方法，为了得到语义图，需要按照推导树的推导顺序进行图重写得到语义图。由于有的法规则对应多条语义规则，我们需要搜索所有可能的情况，选择可能性最大的语义图。为了提高生成语义图的效率，本文使用了基于树的束搜索算法。

考虑到代码实现的难度，我们自底向上进行语义分析，后序遍历推导树的每个结点，对每个结点对应的子树，搜索并保留 k 个分数最高的语义图。由于是后续遍历，在处理结点 A 之前，其两个子节点 B 和 C 已经通过搜索得到了语义子图。设结点 B 有 k_1 种可能的语义图结构，结点 C 有 k_2 种可能的语义图结构，那么根据结点 A 对应的语义组合规则，将结点 B 对应的语义图结构和结点 C 对应的语义图结构组合，可以得到 $k_1 * k_2$ 张语义图。再从其中挑选出分数最高的 k 张语义图作为结点 A 的语义图结构。最后，根结点对应的语义图结构中，分数最高的语义图就是最后的语义分析结果。

第五章 实验

5.1 数据处理

本文模型的训练和测试都在 Deepbank 数据集上进行。DeepBank 是根据英语资源语法规则对宾州树库（Penn Treebank，简称 PTB）的华尔街时报（Wall Street Journal，简称 WSJ）数据标注得到的语料库 [29]。我们实验中使用的 DeepBank 数据集是 1.1 版本，标注采用的是英语资源语法 1214 版本。

宾州树库的华尔街时报部分数据一共分成了 22 组，分别用 `wsj00`,`wsj01`,···,`wsj21` 命名。DeepBank 同样保留了这一数据划分方式。我们按照一般的数据分割方式，以 `wsj00`,`wsj01`,···,`wsj19` 这 20 组数据作为训练集，以 `wsj20` 数据作为开发集，`wsj21` 数据作为测试集。

本文模型实现使用了 Python 语言和 PyTorch 库。在模型训练之前，我们对数据进行了处理，使用 pyDelphin 库从数据中提取 EDS 语义图，再利用 jigsaw[①]工具分离句子中的标点符号。

DeepBank 句法树并不是标准的二叉树，许多结点只有一个子结点，为了提取语义规则，我们将这些只有一个子结点的结点按顺序合并，从而保证每个内部结点都有两个子结点。此外，DeepBank 的标签包含非常丰富的信息，每个标签都是以下划线隔开的数个类型符号，分别表示不同的含义，如 `HD-CMP_U_C`。在我们的实验中，仅选取下划线前的 `HD-CMP` 作为结点的标签进行语法的抽取。

5.2 句法分析结果

本文根据表5.1中的参数对模型进行了训练。之后，使用 evalb[②]工具将预测结果和标准数据比较，对模型进行评测。evalb 是一种括号打分工具，用于计算括号和标签的匹配程度。在评测前，需要将预测得到的短语结构树转化为图3.2所示的字符序列。

实验在 DeepBank 句法树和顺序组合句法树上进行。顺序组合句法树是将句中词语按从左到右的顺序组合得到的句法树，图5.1是 “Colorless green ideas sleep furiously” 的顺序组合句法树，我们用 stepX 表示是第 X 次组合。

为了比较 DeepBank 句法树和顺序结构句法树对语义信息预测的作用，我们在实

① www.coli.uni-saarland.de/ yzhang/files/jigsaw.jar
② https://nlp.cs.nyu.edu/evalb/

超参数	值
随机初始化词向量维数	100
LSTM 层数	2
LSTM 输出维数	256
MLP 隐藏层数目	1
词串打分 MLP 隐藏层维数	256
标签分类 MLP 隐藏层维数	512
优化器	Adam

表 5.1 句法分析模型参数

验中，将句法树结点对应的语义规则右部超图的外点数目作为语义信息，加在该结点标签的结尾处进行预测。

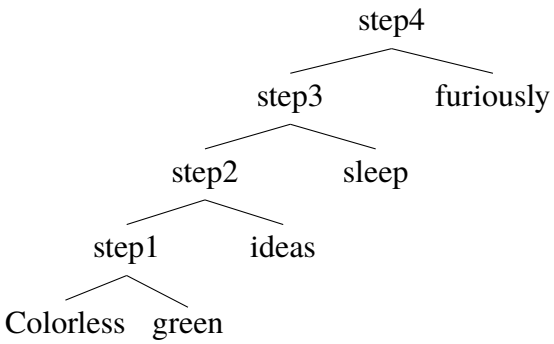


图 5.1 “Colorless green ideas sleep furiously”的顺序组合句法树

实验结果如表5.2所示，满分为 100，其中，标签准确率是叶子节点标签的预测率。从结果可以看出，顺序结构句法树由于结构简单，在不包含语义信息时，更容易预测；但在加入语义信息之后，顺序结构句法树的简单结构蕴含的语义信息较少，难以预测。DeepBank 句法树本身包含了较多的语义信息，无论是否再加入语义信息，预测准确率变化不大。

	无语义信息		有语义信息	
	括号准确率	标签准确率	括号准确率	标签准确率
DeepBank 句法树	85.73	92.99	85.67	91.95
顺序结构句法树	89.10	95.29	45.46	93.94

表 5.2 句法分析结果

5.3 语义图分析结果

语义图分析时使用训练后的句法分析器进行特征提取，再利用隐藏层维数为 100 的单隐藏层 MLP 进行规则的打分和标签的分类。

对于语义图分析器的评价，我们采用了 EDM 评价指标。Dridan、Oepen [30] 提出的 EDM 评价指标直接通过比较两张图顶点和边的重叠度来评价分析结果。和我们句法分析的方式类似，EDM 将顶点和边转化成为元组集合，通过比较元组的重叠程度，就可以得到准确率、召回率和 F 值。

同样的，我们分别测试了以 DeepBank 句法树和顺序结构句法树作为推导树的实验结果。实验结果如表5.3所示。

		Precision	Recall	F-score
DeepBank	顶点	83.68	83.13	83.41
	边	67.75	66.90	67.32
	综合	75.71	74.98	75.34
顺序结构	顶点	61.48	58.64	60.01
	边	14.52	14.68	14.60
	综合	37.21	36.55	36.88

表 5.3 语义分析结果

从实验结果可以看出，我们实现的语义分析模型在顶点预测上有着较好的效果，但在边的预测上效果较差。推导树为 DeepBank 句法树时的效果比推导树为顺序结构句法树时的效果要强很多，说明 DeepBank 句法树确实包含了大量有利于语义分析的信息。

第六章 总结

本文的主要工作有以下几点：

1. 比较了句法分析和语义分析的差异，说明了语义分析对于自然语言处理的必要性，介绍了语义图等语义表示形式。
2. 举例说明了上下文无关文法和超边替换文法的基本概念，介绍了结合上下文无关文法和超边替换文法的基于同步超边替换文法的语义分析技术。
3. 根据 Chen 等 [11] 的论文实现了 SHRG 语义分析模型，将模型分成基于连续词串的句法分析模型和基于图重写的语义分析模型两部分，详细说明了两个模型的工程实现细节。在实验集上进行了训练和测试，模型的表现优秀。
4. 比较了 DeepBank 句法树和顺序结构句法树对句法分析器和语义分析器的训练与测试的影响，说明 DeepBank 句法结构对语义分析确实有促进作用。

到目前为止，本文已经基本实现了基于 SHRG 的语义分析模型，但仍然存在问题：

1. 特征提取时只使用了较为简单的 LSTM-Minus 模型，可以改为 Transformer 等复杂模型提高特征提取的准确度。
2. 语义规则抽取时没有考虑到外部结点的顺序，部分结构相同的语义规则被判断成了不同的规则，从而增加了训练的难度，影响到了实验的效果。
3. 语义分析器对于语义图的边预测效果较差，需要继续训练模型来改进。

此外，目前的自动语义规则抽取算法较为简单，抽取得到的语义规则较为冗杂，其中许多语义规则可能影响到语义分析的结果。后续可以继续进行语义规则的正则化研究，设计出更加优秀的语义规则提取算法，提升语义分析的效果。

参考文献

[1] G. E. Hinton. "*Distributed representations*". **1984**.

[2] D. E. Rumelhart, G. E. Hinton and R. J. Williams. *Learning internal representations by error propagation* [techreport]. **1985**.

[3] T. Mikolov, K. Chen, G. Corrado *et al.* "*Efficient estimation of word representations in vector space*". *arXiv preprint arXiv:1301.3781*, **2013**.

[4] M. E. Peters, M. Neumann, M. Iyyer *et al.* "*Deep contextualized word representations*". *arXiv preprint arXiv:1802.05365*, **2018**.

[5] J. Devlin, M.-W. Chang, K. Lee *et al.* "*Bert: Pre-training of deep bidirectional transformers for language understanding*". *arXiv preprint arXiv:1810.04805*, **2018**.

[6] 沈阳 and 郭锐. 现代汉语. 北京: 高等教育出版社, **2014**.

[7] N. Chomsky and D. W. Lightfoot. *Syntactic structures*. Walter de Gruyter, **2002**.

[8] C. Parsing. "*Speech and language processing*". **2009**.

[9] D. Bahdanau, K. Cho and Y. Bengio. "*Neural machine translation by jointly learning to align and translate*". *arXiv preprint arXiv:1409.0473*, **2014**.

[10] L. Song, Y. Zhang, Z. Wang *et al.* "*A Graph-to-Sequence Model for AMR-to-Text Generation*". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, 2018-07: 1616–1626. https://www.aclweb.org/anthology/P18-1150.

[11] Y. Chen, W. Sun and X. Wan. "*Accurate SHRG-Based Semantic Parsing*". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, 2018-07: 408–418. https://www.aclweb.org/anthology/P18-1038.

[12] 耿素云, 屈婉玲 and 王捍贫. 离散数学教程. 北京: 北京大学出版社, **2002**.

[13] L. Banarescu, C. Bonial, S. Cai *et al.* "*Abstract Meaning Representation for Sembanking*". In: *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*. Sofia, Bulgaria: Association for Computational Linguistics, 2013-08: 178–186. https://www.aclweb.org/anthology/W13-2322.

[14] S. Oepen, K. Toutanova, S. Shieber *et al.* "*The LinGO Redwoods Treebank: Motivation and Preliminary Applications*". In: *COLING 2002: The 17th International Conference on Computational Linguistics: Project Notes*. **2002**. https://www.aclweb.org/anthology/C02-2025.

[15] C. Pollard and I. A. Sag. *Head-driven phrase structure grammar*. University of Chicago Press, **1994**.

[16] A. Copestake, D. Flickinger, C. Pollard *et al.* "*Minimal recursion semantics: An introduction*". *Research on language and computation*, **2005**, *3*(2-3): 281–332.

[17] J. E. Hopcroft, R. Motwani and J. D. Ullman. "*Introduction to automata theory, languages, and computation*". *Acm Sigact News*, **2001**, *32*(1): 60–65.

[18] F. Drewes, H.-J. Kreowski and A. Habel. "*Hyperedge Replacement Graph Grammars*". In: *Handbook of Graph Grammars and Computing by Graph Transformation: Volume I. Foundations*. USA: World Scientific Publishing Co., Inc., **1997**: 95–162.

[19] I. Goodfellow, Y. Bengio and A. Courville. *Deep Learning*. MIT Press, **2016**, http://www.deeplearningbook.org.

[20] W. S. McCulloch and W. Pitts. "*A logical calculus of the ideas immanent in nervous activity*". *The bulletin of mathematical biophysics*, **1943**, *5*(4): 115–133.

[21] M. Marvin and A. P. Seymour. *Perceptrons*. MIT Press, **1969**.

[22] P. Werbos. "*Beyond regression:" new tools for prediction and analysis in the behavioral sciences*". *Ph. D. dissertation, Harvard University*, **1974**.

[23] S. Hochreiter and J. Schmidhuber. "*Long Short-Term Memory*". *Neural Computation*, **1997**, *9*(8): 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

[24] R. McDonald and F. Pereira. *Discriminative learning and spanning tree algorithms for dependency parsing*. University of Pennsylvania, **2006**.

[25] J. Cross and L. Huang. "*Span-Based Constituency Parsing with a Structure-Label System and Provably Optimal Dynamic Oracles*". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, 2016-11: 1–11. https://www.aclweb.org/anthology/D16-1001.

[26] J. Cocke. *Programming languages and their compilers: Preliminary notes*. New York University, **1969**.

[27] D. H. Younger. "*Recognition and parsing of context-free languages in time n3*". *Information and control*, **1967**, *10*(2): 189–208.

[28] T. Kasami. "*An efficient recognition and syntax-analysis algorithm for context-free languages*". *Coordinated Science Laboratory Report no. R-257*, **1966**.

[29] D. Flickinger, Y. Zhang and V. Kordoni. "*DeepBank. A dynamically annotated treebank of the Wall Street Journal*". In: *Proceedings of the 11th International Workshop on Treebanks and Linguistic Theories*. **2012**: 85–96.

[30] R. Dridan and S. Oepen. "*Parser Evaluation Using Elementary Dependency Matching*". In: *Proceedings of the 12th International Conference on Parsing Technologies*. Dublin, Ireland: Association for Computational Linguistics, 2011-10: 225–230. https://www.aclweb.org/anthology/W11-2927.

[31] H. Shi, X. Wang, Y. Sun *et al.* "*Constructing High Quality Sense-specific Corpus and Word Embedding via Unsupervised Elimination of Pseudo Multi-sense*". In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan: European Language Resources Association (ELRA), 2018-05. https://www.aclweb.org/anthology/L18-1154.

本科期间的主要工作和成果

2018 年 5 月在会议 *International Conference on Language Resources and Evaluation* 上发表了论文。其中，石昊悦为第一作者，本人为第二作者：

H. Shi 等. “*Constructing High Quality Sense-specific Corpus and Word Embedding via Unsupervised Elimination of Pseudo Multi-sense*”. In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan: European Language Resources Association (ELRA), 2018-05. https://www.aclweb.org/anthology/L18-1154

致谢

在北京大学的四年时间转瞬而逝，不知不觉就到了毕业的时节。四年前，我怀着忐忑的心情来到燕园，担心自己不能够适应北大的学习生活。好在有许多老师和同学的帮助和支持，使我能够顺利地学习科研。在此，要特别感谢室友谷晟、吕栋杰和孙至诚在这几年对我的关心和照顾。

2016 年 9 月，我选修了胡俊峰老师开设的《计算概论（实验班）》。除了基本的计算机编程外，他还向我们介绍了计算机科学多个领域的研究内容。课程结束后，我参加了胡老师组织的讨论班，学习了机器学习和自然语言处理的基础知识，最后受邀加入了胡老师的实验室，进行了多义词向量的研究。在此，感谢胡俊峰老师对我的引导、石昊悦学姐对我的指点和孙雨奇同学对我的帮助，使我对科研产生了浓厚的兴趣。

2018 年 7 月，我加入了孙薇薇老师的实验室，开始进行语义概念嵌入的研究。孙老师不仅是非常出色的研究者，还是位优秀的导师。她在科研上的指导使我加深了对自然语言处理的认识和理解，每次和她的讨论都使我收获良多；她也关心我的学习和生活，给了我许多建议。和孙老师的交流和沟通激发了我对语言学的兴趣，坚定了我进行科研的决心。在她的帮助和指导下，我成功直博了中国语言文学系，今年九月，我将开始跟随现代汉语教研室的詹卫东老师攻读中文信息处理方向。此外，还要感谢实验室陈宇非师兄、叶亚杰师兄等人对我的帮助，使我解决了不少棘手的问题。

2020 年，新型冠状肺炎对社会秩序产生了巨大的影响。几个月来，在社会各界的通力合作下，国内疫情已经得到控制，感谢医护人员、社区工作人员等诸多在抗疫一线作出牺牲的人们，正是他们的付出保障了人民的安全、稳定了社会秩序。

最后，还要感谢我的父母，他们从小对我的教育和一直以来对我的支持造就了今天的我。今年，受到新冠肺炎的影响，我只能在家完成本文的写作，多亏了他们近几个月来对我生活上的关心和帮助，才使我能够顺利完成论文。