哈尔滨工业大学

博士学位论文开题报告

题目：中文语义依存图分析技术研究

院　　(系)　计算机学院

学　　科　计算机科学与技术

导　　师　车万翔副教授

研　究　生　王宇轩

学　　号　16B903007

开题报告日期　2018 年 4 月 16 日

研究生院制

二〇一五年三月

目录

中文语义依存图分析技术研究

1 课题来源及研究的目的和意义

让机器理解自然语言（Natural Language Understanding, NLU），是自然语言处理（Natural Language Processing, NLP）领域最根本也是最重要的问题之一。要实现在语义层面上理解自然语言，一般来说需要对原始文本自底向上进行分词、词性标注、命名实体识别、句法分析，最后才能进行语义分析。

然而，由于中文严重缺乏形态变化，词类与句法成分没有严格的对应关系，导致中文句法分析的精度始终不高。例如，在英文宾州句法树库（Penn Treebank, PTB）上目前句法分析准确率已经能达到 95%，而在中文宾州句法树库（Chinese Penn Treebank, CTB）上则只能达到 88% [1]。为了解决这些问题，哈工大社会计算与信息检索研究中心（HIT-SCIR）结合中文重意合、在形式分析上有劣势的语言特点，于 2011 年在世界上最早提出了跨越句法分析直接进行语义依存分析的思路，并与北京语言大学合作标注了中文语义依存树语料库，用树结构融合依存结构和语义关系。此后又将其扩展为语义依存图，从而更全面地刻画句中词之间的语义关系。

语义依存分析（Semantic Dependency Parsing）是通往语义深层理解的一条蹊径，它通过在句子结构中分析实词间的语义关系来回答句子中“Who did what to whom when and where”等问题。以图 1中的中文句子“张三前天告诉李四一件事”为例，通过语义依存分析，我们能获知“谁告诉李四一件事?”、“张三告诉谁一件事?”、“张三何时告诉李四一件事?”及“张三告诉李四什么?”等问题的答案。

此外，与句法分析不同，语义分析可以跨越句子的表层结构直接获取深层语义表达的本质，例如图 2中的句子“昨天，张三将一个秘密告诉李四。”虽然与图 1中的句子表述形式不同，但含义相同，因此它们的语义依存结构相同。因此语义依存分析能为机器翻译、问答系统、对话生成等对语义信息要求较高的下游任务提供很大帮助。

语义依存分析建立在依存理论基础上，是对语义的深层分析，具有形式简洁，易于理解和运用的优势。具体可分为两个阶段，首先是根据依存语法建立依存结构，即找出

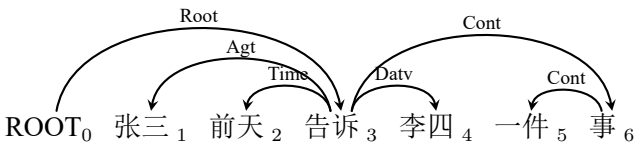


图 1 语义依存表示示例

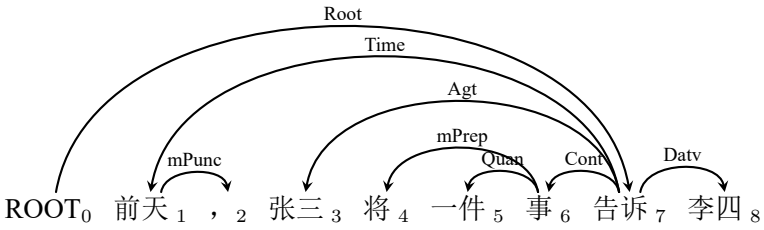


图 2 语义依存表示示例

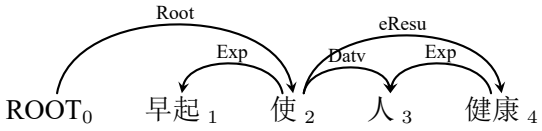


图 3 语义依存图表示示例

句子中的所有修饰词与核心词对，然后再对所有的修饰词与核心词对指定语义关系。因此，语义依存分析可以同时描述句子的结构和语义信息。

与分词、词性标注、句法依存分析等历史悠久的研究课题相比，语义依存分析算得上是一个新兴课题。由于其与句法依存分析在结构上的相似之处，语义依存树的分析基本可以沿用已经有诸多成果的句法依存分析领域的方法。然而近年来人们发现树结构已经无法胜任刻画句子中复杂的语义关系的任务，研究者们逐渐将目光聚集到约束更少的图结构上，希望用图结构来表示这些语义关系。因此，无论在中文还是在以英文为代表的一些外语中，越来越多图结构的语义依存语料库应运而生。例如图 3的句子“早起使人健康”中，“人”同时作为“使”的与事者（Dative, Datv）和“健康”的当事者（Experiencer, Exp）。

图结构在带来更全面的语义表示的同时，也给语义依存分析任务带来了巨大的挑战。对于原来的依存树结构，每个词只有一个父节点，而在依存图中，每个词可能有多个父节点，这就给分析系统带来了很大的不确定性，从而增加了语义依存分析任务的难度。因此，本学位论文的研究目标集中在图结构的中文语义依存分析上，希望能通过实现更好的语义依存分析器，提高中文语义依存分析的精度，为下游任务提供更好的语义信息，弥补中英文由于语言特性在句法分析领域产生的差距。

2 国内外在该方向的研究现状及分析

2.1 国内外研究现状

语义依存分析是一个新兴课题，虽然已经引起了许多研究者的关注，但是相关的数据集和研究工作相比分词、词性标注等历史悠久的任务都比较有限，而国内的研究更是非常少，因此这里将国内外的研究现状总结到一起描述。

语义依存树与句法依存树主要的区别在于依存弧上的标签不同，这源自于两种任务目标上的区别。因此语义依存树的分析基本可以沿用句法依存分析方法，对于后者目前已经存在大量的研究工作，目前应用最多的主要可以分为两大类：基于转移（Transition-based）和基于图（Graph-based）的依存分析。基于图的算法将依存分析转换为在有向完全图中求解最大生成树的问题，是基于动态规划的一种图搜索算法。该算法由McDonald et al. (2005)[2] 提出，是一种全局最优的搜索算法。Yamada and Matsumoto (2003)[3] 提出的基于转移的依存分析算法将句子的解码过程建模为一个转移序列的构造过程。其依存分析模型的目标是通过学习得到一个能够准确的预测下一步转移动作的分类器。

语义依存图与句法依存树除了上述的依存弧标签不同之外，结构上也有所差异。语义依存图打破了传统的依存树结构，允许多父节点的存在（在依存树中一个词只能有一个父节点），因此能够刻画句中词之间更丰富的语义关系。但多父节点的存在同时也使得对语义依存图的分析更具挑战性。目前对语义依存图的分析工作主要使用的方法都是对上述基于转移和基于图的依存分析方法的扩展。下面我们首先简要介绍目前国内外的语义依存图数据集，然后分别对基于转移和基于图的语义依存图分析工作进行总结。

2.1.1 语义依存图数据集

目前国内的语义依存图数据集只有由哈工大社会计算与信息检索研究中心（HIT-SCIR）和北京语言大学合作标注的 SemEval-2016 Task 9 中文语义依存图数据集 [4]。为了解决 SemEval-2012 Task 5 语义依存树数据集 [5] 中语义关系彼此易混淆、语义关系数量太大、刻画语义不全面等问题，在中文语义依存图数据集中用依存图结构代替依存树。形式上类似于依存句法结构，但必要时突破树形结构（BH-SDP-v2 标注体系）。这样的突破使得对连动、兼语、概念转位等汉语中常见的现象的分析更全面深入，当然这也给依存分析器的构建带来了很大的难度，因为任何词都可能有多个父节点。图 4直观地展示了语义依存树与依存图的区别。

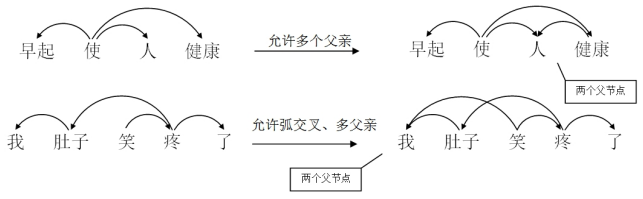
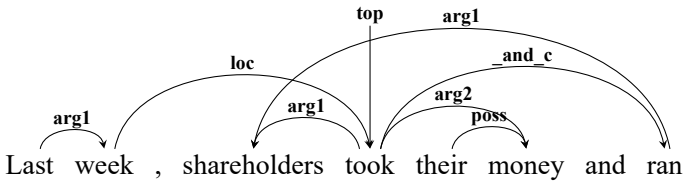
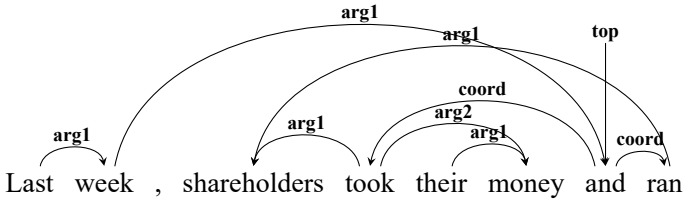


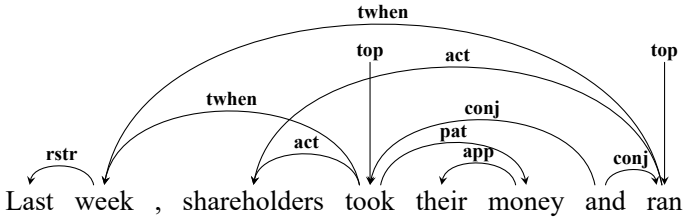
图 4 中文语义依存树与语义依存图对比示例



a) DM



b) PAS



c) PSD

图 5 广义语义依存图三种标注体系示例

BH-SDP-v2 标注体系压缩了语义关系类型的数量，重新组织并缩减了语义关系，将关系分为主要语义角色、事件关系、关系标记，从而减少不必要的类间关系混淆。语义关系在保留了一般语义关系、反关系基础上，还增加了嵌套关系，用来标记一个事件降级充当了另一个事件的成分。SemEval-2016 Task 9 中文语义依存图数据集中包含 10068 句新闻语料和 15000 句课文句子。新闻句子平均长度是 31 个词，课本句子平均长度是 14 个词。该数据集被用于 SemEval-2016 Task 9 国际公开评测上。

国外的语义依存图数据集主要是 SemEval-2015 Task 18 广义（Broad-Coverage）语义依存图数据集。其目标是获取句子内部所有实词之间的谓词论元关系，即获取表示该句子含义的语义结构。构建该数据库的目的是寻找更具普适性的图结构，从而提供对“Who did what to whom”等问题更直接的分析方式。该语料库中使用了三种不同的标注体系（图 5给出了三种标注的例子），分别是：

1. **DM (DELPH-IN MRS-Derived Bi-Lexical Dependencies)** 该语义依存图标注来源于参考 LinGO 英文资源语法（English Resource Grammar）给出的句法、语义信息进行人工重标注的 DeepBank 语料。
2. **PAS (Enju Predicate–Argument Structures)** Enju 树库和分析器来源于对宾州树

库的 HPSG 形式自动标注。PAS 语义依存图是直接从 Enju 树库中抽取的，没有进行内容转换。

3. **PSD (Prague Semantic Dependencies)** 布拉格捷克语-英语依存树库（Prague Czech-English Dependency Treebank）是包括宾州树库华尔街日报部分及其捷克语翻译的依存树的语料库。PSD 语义依存图是从该树库的构造语法标注层（tectogrammatical annotation layer）中抽取出来的。

该数据集中包括英语、中文和捷克语的语料。其中英语部分来自宾州树库（PTB）华尔街日报和布朗部分，包括 35657 句训练语料、1410 句同领域测试语料及 1849 句不同领域（来自布朗部分）测试语料，拥有 DM、PAS 和 PSD 三种标注规范的语义依存图。中文部分来自中文宾州树库（CTB 7.0），包括 31113 句训练语料和 1670 句同领域测试语料，只有 PAS 规范的标注。捷克语部分来自宾州树库华尔街日报部分的捷克语翻译，包括 42076 句训练语料、1670 句同领域测试语料及 5226 句不同领域测试语料，只有 PSD 规范的标注。该语料的英语部分首先用于 SemEval-2014 Task 8 评测 [6]。此后的 SemEval-2015 Task 18 评测任务 [7] 中，该语料库又加入了中文和捷克语部分语料。

除此之外，还有一些数据集与语义依存图在结构或内容上有相似之处，包括 CoNLL-2009 数据集 [8] 的语义角色标注（Semantic Role Labeling）部分、CCD（CCG Dependencies）[9]、AMR（Abstract Meaning Representation）数据集 [10]、丹麦语依存树库 [11]（DDT）和 HPSG 树库 [12] 等。其中前两个属于语义角色标注范畴，其目的是找出句中一些谓词与其论元之间的关系，因此该任务一般分为两步，第一步是识别出句中的谓词，第二步才是找到每个谓词对应的论元。这里所说的谓词，一般来说指的是动词和名词。而语义依存分析的目的是找到句中所有词之间的语义关系，对词的种类没有限制，因此不需要进行上述的第一步。从两个任务目的上的不同能够发现，语义依存分析能提供一个句子更全面的语义信息。AMR 属于语义网络（Semantic Network），其中图节点表示的是语义概念，不需要与句中的词一一对应。而语义依存图则用有向弧直接表示的句中词之间的语义关系。此外，AMR 目前只有英文数据，在中文上没有与之类似的数据集。丹麦语依存树库和 HPSG 树库本质上来说也属于依存句法树库，但由于其中也包含一些图结构的依存句法表示，因此在语义依存图没有提出之前，一些对图结构分析算法的研究是在这些数据集上进行的，这些方法对我们也有一定的启示作用。

2.1.2 基于转移的分析方法

在基于转移的依存图分析方向上，Sagae and Tsujii (2008)[13] 发表了通过向转移系统中增加新的转移动作实现基于转移的依存图分析的先驱性工作。在传统的基于转移的依存树分析系统中，由于每个词只有一个父节点，因此一旦找到一个词的父节点就要将该词从系统中删除（LEFT-REDUCE 和 RIGHT-REDUCE 动作），Sagae 等人在转移系统中加入两个新的生成依存弧的动作（LEFT-ATTACH 和 RIGHT-ATTACH），只生成依存弧而不删除词，从而实现了多父节点结构的生成。他们在丹麦语依存树库和 HPSG 树库两

个含有图结构的依存句法树库中进行了实验。此后该方向上的研究工作主要都集中在通过修改转移系统处理依存图结构。增加两个新的动作是一个最直观的解决依存图分析问题的方法，但是新增的两个动作只生成依存弧，不改变转移系统状态，在执行这两个动作前后转移系统的状态是基本不变的，这也就导致了从当前状态中提取出的用于预测下个转移动作的特征也是基本相同的，这会影响分类器预测的准确性，从而降低系统的整体性能。Titov et al. (2009)[14] 提出了另一种转移系统，在生成弧的同时不改变系统中的词，而是用独立的 REDUCE 动作删除已经无用的词，同时利用 SWAP 动作改变系统中词的顺序。

Zhang et al. (2016)[15] 提出了两种新的转移系统，第一种在线重排序（Online re-ordering）系统与 Titov 的系统类似，第二种基于双栈的（Two-Stack-Based）系统拥有一个额外的栈，用于保存从正在处理词的栈中暂时移除的词，通过 MEM 和 RECALL 两个动作实现向该额外的栈中保存词和从该栈中取词。除了转移系统外，基于转移的依存分析中十分重要的另一个部分就是用于在给定转移状态下预测下一个转移动作的分类器。Zhang 等人在实验中也发现如果转移动作集合中存在只生成依存弧而不改变转移系统的动作，会影响分类器准确性。为了解决这一问题，他们将与生成依存弧有关的动作（NO、LEFT 和 RIGHT，分别表示不生成弧，生成向左或向右的弧）和改变转移状态的动作（SHIFT、REDUCE、SWAP 等）两两组合，在最终的系统中使用组合后动作，保证了每个动作都会改变转移系统状态，成功提高了系统性能。

2.1.3 基于图的分析方法

在基于图的依存分析方向上，McDonald and Pereira (2006)[16] 首先提出了利用近似 Eisner 算法 [17] 解决基于图的依存分析中多父节点的生成的方法。他们首先使用原始 Eisner 算法生成一个投射树（依存弧存在交叉的树），然后对树中的边逐条替换，替换时允许出现多父节点，保留使整体评分增加的替换，从而生成图结构。McDonald 等人也在丹麦语依存树库上进行了实验。这种方法本质上来说只是在传统的依存树分析算法得到的结果的基础上进行后处理，将树结构改成图结构，面对较复杂的图结构依然不能算是一个很好的解决方案。

Martins and Almeida (2014)[18] 提出了一种利用二阶特征的方法，在单个弧的一阶特征基础上，增加了连续兄弟节点、连续多父节点等二阶特征，使用 AD^3 算法 [19] 进行解码，实现了依存图的预测。AD^3 算法是一种基于对偶分解的近似离散优化算法。由于这里采用的分解方式不是以弧为因子的（arc-factored），各结构之间存在重叠，所以需要使用该算法进行解码。与 McDonald 等人的方法相比，该方法能够直接预测出图结构，而不是通过对依存树的后处理生成依存图。

Peng et al. (2017)[20] 利用了广义语义依存图语料库中英语部分有三种不同标注的特点，采用了多任务学习方法，仅用一阶特征就在该数据集上取得了当时最好结果。不仅如此，他们还借鉴了基于图的依存树分析研究中对神经网络的成功应用 [21]，将句中每个

词的词向量输入双向长短时记忆网络（Bidirectional Long-Short Term Memory, Bi-LSTM），用其隐层输出作为每个词的表示，然后利用多层感知器（Multi-Layer Perceptron, MLP）计算每条候选弧的分数。通过这种方法，他们的模型即使不使用多任务学习方法，也有很好的表现。

2.2 国内外文献综述的简析

2.2.1 现有工作的总结及其不足

由于语义依存图的概念才提出不久，中文及其他语言上的语义依存图相关数据集构建工作也是近几年才初步完成，语料有限，目前为止国内外对于语义依存图分析的研究工作还比较少。因此，我们不但调研了专门研究语义依存图分析的工作，也从此前对于句法依存图等图结构分析的研究工作中获取灵感。总的来说，目前基于转移的依存分析方法和基于图的依存分析方法都开辟出了各自对图结构进行分析的道路。

在基于转移的分析方法中，目标依存结构是通过在包括待分析句子信息的转移系统（Transition System）中执行一系列的类似移进、规约的转移动作修改转移状态（转移系统中各部分的状态）来生成的，其最终目的是训练一个给定当前转移状态能够预测出接下来要执行的转移动作的分类器。因此，在这类方法中生成依存图的最大困难来自于每个词的父节点由原来的一个变成了任意多个，导致分类器难以判断何时对一个词进行规约。针对这一问题一般的解决办法是修改转移系统，提出新的转移动作替代或补充原有转移动作，从而增强转移系统的处理能力，使其可以产生依存图结构。因此如何设计或修改转移系统，就成为了基于转移的分析方法中一个十分重要的研究问题。此外，由于这类方法的核心是用于预测转移动作的分类器的训练，而分类器预测的依据则是当前的转移状态，因此如何更好、更快、更全面地获得当前转移状态的信息将其作为分类器的分类依据就成为了这类方法中另一个十分值得研究的问题。

在基于图的分析方法中，首先要将目标分成许多子结构，计算出所有可能的子结构的分数。分解方式有很多种，以一条弧为单位的分解称为一阶分解（first-order factorization），以两条弧一组为单位进行的分解称为二阶分解（second-order factorization）。然后利用动态规划算法计算出总分最大的子结构组合。当分析目标是依存树时，一般使用最大生成树（Max Spanning Tree）算法。在面对依存图结构时，由于每个词的父节点个数由一个变为任意多个，而依存图又有非投射性（弧之间存在交叉），原来的动态规划算法不再适用，只能对每个子结构进行独立判断。如果使用了多种分解方式，则需要采用 AD^3 算法解决各子结构存在重叠时的解码问题。

上述两类方法中的绝大部分工作都使用了诸如线性模型、最大熵模型、结构化感知器等传统统计学机器学习方法。它们都是通过人工特征工程实现从环境中获取信息的，这样人工设计特征模板不但繁琐，需要该领域的专家知识，而且往往无法涵盖所有的信息，同时，对特征字符串的组合也会耗费大量时间。

近年来，模型融合（Model Ensemble）技术在依存句法分析领域已被证明十分有效，但是目前的模型融合方法不是利用多个不同模型的最终结果进行投票决定最终预测结构，就是对同一模型使用不同的随机初始化种子多次训练然后同时使用这些独立训练的模型进行预测。而且后者仅仅被应用在基于转移的依存分析中，并没有工作将这种模型融合方式应用于基于图的依存分析中。这两种模型融合方式虽然简单有效，但没有利用依存分析任务的特性，仅仅是把其它领域的模型融合方式简单搬运到依存分析领域，缺乏针对性。

作为一个新兴研究课题，中外文的语义依存图语料数量都十分有限，这给语义依存图分析研究的进行和高精度语义依存图分析器的训练都带来了巨大挑战。在这种情况下，更应该充分利用现有多种语言的语料资源，同时利用依存句法分析、语义角色标注等已有大量人工标注数据的相关任务帮助提高语义依存图分析器的精度。Peng et al. (2017)[20] 已经在 SemEval-2015 Task 18 英文广义语义依存图数据集上证明了通过多任务学习方法同时学习该数据集的三种标注体系能够有效提高最终的实验结果。但目前这方面的其它工作十分有限，仍有很大的研究空间。

2.2.2 有待深入研究的问题

与分词、句法分析等历史悠久的任务相比，语义依存图分析是一个十分年轻的任务，许多相关研究才刚刚起步，无论是在分析方法本身的研究上，还是在提高分析系统性能、利用多语言及其他任务数据的研究上，都有许多工作可以做。对于分析方法来说，基于转移的和基于图的两类依存图分析方法的研究都刚刚开始，孰优孰劣未有定论，因此这两类方法的研究都很有价值和意义。而在有了依存图分析方法之后，提出适应语义依存图任务特性的方法来提高系统性能则成为最重要的问题。另一方面，由于目前语义依存图语料较少，如何充分利用现有多种语言的语料资源以及依存句法分析、语义角色标注等相关任务的数据也是一个十分有意义的研究方向。因此，未来对于语义依存图分析的研究将集中在一下几个方面：

1. **基于转移的分析方法：**在传统基于转移的依存树分析方法基础上，如何提出新的转移系统，解决分类器不知道何时对一个词进行规约的问题，使其能生成依存图结构，是基于转移的依存图分析方法的根本。在此基础上，研究如何使用新的信息抽取方式代替原来低效、繁琐的人工特征工程，从而更好、更快、更全面地获得当前转移状态的信息是十分有意义的
2. **基于图的分析方法：**在基于图的分析方法中，第一步计算每个子结构的分数是至关重要的，因为此后使用解码算法得到的依存图结构完全是依赖于第一步中得出的分数的。所以在基于图的分析方法中如何从句子中获取更全面的信息帮助计算子结构的分数是很值得研究的。
3. **利用模型融合技术提高系统性能：**模型融合已经在许多领域证明是一个简单有效的提高系统性能的方法。现有的在依存分析系统中利用模型融合技术的工作实现

的仅仅是简单的最终结果投票融合或者是使用不同的随机初始化种子。这些方法并没有考虑到语义依存分析任务的特性，同时可解释性也不强。因此，如何设计一个符合语义依存分析任务特性，能够融合对不同维度信息进行建模的融合方法，将是十分重要的。

4. **利用多语言、多领域数据**：随着深度学习技术在各领域取得的巨大成功，数据量往往与使用深度学习技术的系统性能成正比。由于语义依存图语料有限，如何充分利用现有多种语言的语料资源以及依存句法分析、语义角色标注等已有大量人工标注语料的相关任务数据也是一个十分有意义的研究方向。

3 前期的理论研究与实验论证工作的结果

本课题相关的理论研究始于 2015 年 9 月，在经过深入的调研顺滑理论及与其紧密相关的其它自然语言处理技术的基础上，开始对顺滑任务中存在问题进行三方面的尝试。首先，针对顺滑任务中的长距离依赖问题，我们尝试用 LSTM 来对句子进行表示，同时，为了避免用 LSTM 的输出直接进行分类所带来的 label 偏置问题，我们最终的模型会在 LSTM 表示的基础上加一个 CRF 层，最终的 LSTM-CRF 模型相比于传统的模型和直接用 LSTM 的输出进行分类的模型相比，取得了更好的性能，该工作已经被 CCL2016 录用。其次，为了保证得到的顺滑句子的句法完整性，同时充分的利用全局的信息，我们尝试用注意力机制的方法，该方法将顺滑任务转化成一个生成的过程，对原句子进行编码后，直接生成目标句子，该方案取得了非常好的性能，该工作已经被 COLING2016 录用。最后，针对非顺滑块和顺滑块之间相似性，我们尝试基于转移的方案，我们的方案能很好的建模块之间的联系，同时能充分的利用全局的信息，该方案取得了目前最好的实验结果，该工作已经被 EMNLP2017 录用。由于 LSTM-CRF 模型不属于博士课题的研究点，因此本文将简要介绍以下两项工作。

(1) Transition-Based Disfluency Detection using LSTMs. (EMNLP 2017，已发表)

(2) A Neural Attention Model for Disfluency Detection. (COLING 2016，已发表)

3.1 基于转移的顺滑技术研究

3.1.1 实验动机

传统的方法将顺滑任务看作序列标注问题 (2016, 2015, 2013, 2009)[22–25]，为句子中的每一个词分配一个标签，然后根据标签判断是否要将其保留。这类方法取得了不错的结果，但其不能很好的解决长距离依赖问题，而且也没有能力利用块级别的信息。

为了利用块级别的信息，Ferguson et al. (2015)[26] 尝试使用 semi-CRF 模型来解决顺滑问题，并且取得了很好的实验效果。但是 semi-CRF 由于受到马尔科夫假设的约束，其也只能利用局部的块信息。

还有一些工作 (2013, 2014, 2015)[27–29] 尝试将句法分析和顺滑任务联合起来。这种联合模型的主要优点在于它们可以建模长距离的依赖关系，而且能够很好的利用块级别的信息。然而，它要求训练数据要同时标注句法和顺滑信息，这降低了算法的实用性，同时句法分析的噪声可能会严重影响顺滑任务的性能。

受上述观察的启发,我们使用一种不带句法信息的转移系统来解决顺滑问题 (2016)[30]。我们基于转移的方法通过采用和依存句法类似的解码算法来递增地构建和标记输入句子中的不流畅块。

3.1.2 背景介绍

在背景介绍部分，我们简要说明一下基于转移的句法分析方法和在此基础上扩展而来的句法和顺滑联合方法。一个经典的基于转移的 arc-eager 句法分析器主要由一个存储正在处理的词的堆栈 σ，一个包含将要被处理的词的缓冲区 β 和一个存储已生成的依存弧的存储器 A 四部分组成。对于该转移系统，有四种类型的转移动作 (2008)[31]

- *Shift* : 将缓冲区的顶端的词压到堆栈中去.
- *Reduce* : 弹出堆栈顶端的词.
- *LeftArc* : 将堆栈顶端的词弹出堆栈，并将其链接到缓冲区的顶端的词下面，作为其孩子节点.
- *RightArc* : 将缓冲区顶端的词弹出，并链接到堆栈顶端的词下面，作为其孩子节点

许多基于神经网络的句法分析器都遵循这套转移框架，例如Dyer et al. (2015)[32]，他们使用不同的 LSTM 结构来表示从 σ 到 β 的信息。

对于顺滑任务来说，输入是自动语音识别（ASR）后的可能带有不流畅块的句子。我们将输入句子的词序列表示为 $w_1^n = (w_1, ..., w_n)$。顺滑任务的输出是表示为 $D_1^n = (d_1, ..., d_n)$ 的标签序列，其中每个 d_i 代表对应的词 w_i 是否属于不流畅的词。因此，顺滑任务可以被建模为给定词序列 w_1^n，搜索最好的序列 D^*。

$$D^* = argmax_D P(D_1^n | w_1^n)$$

Wu et al. (2015)[29] 提出了一种句法和顺滑联合模型，其在解码过程中，通过增加一个二值分类动作（BCT）来判断哪些词属于非顺滑的词。

- *BCT* : 判断当前的词是否为不流畅词。如果是，将其从 buffer 中删除，并将其标记为不流畅词。否则就从初始的四种动作中选择一种可能性最大的动作来执行。

顺滑和句法任务进行联合的优化，

$$(D^*, T^*) = argmax_{D,T} \prod_{i=1}^{n} P(d_i | w_1^i, T_1^{i-1}) \times P(T_1^i | w_1^i, T_1^{i-1}, d_i),$$

其中 T_1^i 表示词 w_i 被处理后的部分句法树信息，d_i 是 w_i 的顺滑标签。$P(T_1^i|.)$ 表示句法模型，$P(d_1^i|.)$ 表示顺滑模型，其被用来根据部分句法树信息预测当前词的顺滑标签。

3.1.3 方法设计

BCT 模型在所有句法和顺滑联合方法中的性能是最好的。然而，它要求训练数据同时包含句法树和顺滑标注信息，这降低了算法的实用性。此外，BCT 并没有直接地利用不流畅块的信息。作为一个完全采用离散特征的模型，其性能很大程度上依赖于复杂的人工特征。

为了解决上面的约束，我们采用了一个不使用任何句法信息的基于神经网络的转移模型来解决顺滑问题。我们基于转移的方法通过执行一系列动作来递增地构建和标记输入句子中的不流畅块。任务被建模为

$$(D^*, T^*) = argmax_{D,T} \prod_{i=1}^{n} P(d_i, T_1^i | w_1^i, T_1^{i-1}),$$

其中 T_1^i 是词 w_i 被处理后的局部模型状态。d_i 是 w_i 的顺滑标签。

动作设计

我们的基于转移的方法通过执行一系列动作来递增地构建和标记输入句子中的不流畅块，其中每个时刻的状态由一个四元组 (O, S, A, B) 表示

- *output* (O) : *output* 用于表示已经被标记为流畅的词。
- *stack* (S) : *stack* 用于表示部分已经被标记为不流畅的词。
- *action* (A) : *action* 用于表示转移系统采取动作的完整历史记录。
- *buffer* (B) : *buffer* 用于表示尚未处理的句子。

给定一个可能含有不流畅块的句子，*stack*，*output* 和 *action* 初始化是空的，*buffer* 包含该句子的所有的词，然后一系列的转移动作被用于处理 buffer 中的词并构建最终的输出：

- OUT: 将 *buffer* 中的第一个词移动到 *output*，并将 *stack* 清除。
- DEL: 将 *buffer* 中的第一个词移动到 *stack*。

解码算法

基于设计好的转移系统，解码器搜索给定句子的最佳动作序列。系统初始化的时候会将所有的词及其表示逆向的送到 B 中，并将 S，O，A 置为空。

在每一步，系统根据当前的状态计算每个动作对应的概率，从而决定将要被执行的动作。当 B 为空时，解码完成（不管 S 的状态如何）。由于每一步 B 顶端的词都会被直接移动到 O 或 S，所以动作序列的总数总是等于输入句子的长度。表 1 显示了处理句子“want a flight to boston to denver”的动作序列。

如图**??**所示，t 时刻的模型状态用 e_t 表示，其定义为：

$$e_t = max\{0, W[s_t; b_t; o_t; a_t] + d\},$$

其中 W 是一个需要学习的参数矩阵，s_t 是 S 的表示，b_t 是 B 的表示，o_t 是 O 的表示，a_t 是 A 的表示，d 是一个偏置项。$(W[s_t; b_t; o_t; a_t] + d)$ 通过一个 ReLU 函数做非线性转化。

表 1 对于输入 *a flight to boston to denver* 的处理过程

Step	Action	Output	Stack	Buffer
0	NO	[]	[]	[a, flight, to, boston, to, denver]
1	OUT	[a]	[]	[flight, to, boston, to, denver]
2	OUT	[a, flight]	[]	[to, boston, to, denver]
3	DEL	[a, flight]	[to]	[boston, to, denver]
4	DEL	[a, flight]	[to, boston]	[to, denver]
5	OUT	[a, flight, to]	[]	[denver]
6	OUT	[a, flight, to, denver]	[]	[]

最后，模型状态 e_t 被用于计算 t 时刻的转移动作概率：

$$p(z_t|e_t) = \frac{exp(g_{z_t}^{\mathrm{T}} e_t + q_{z_t})}{\sum_{z' \in \mathcal{A}(S,B)} exp(g_{z'}^{\mathrm{T}} e_t + q_{z'})},$$

其中 g_z 为转移动作 z 的表示的列向量，q_z 是动作 z 的偏置项。集合 $\mathcal{A}(S,B)$ 表示给定当前状态所有可能被采取的动作。由于 $e_t = f(s_t, b_t, a_t, o_t)$ 包含了转移系统所有先前动作的历史信息，因此任意可能的转移动作 z 的概率可以表示为：

$$p(z|w) = \prod_{t=1}^{|z|} p(z_t|e_t)$$

因此我们有

$$\begin{aligned}(D^*, T^*) &= argmax_{D,T} \prod_{i=1}^{|z|} P(d_i, T_1^i | w_1^i, T_1^{i-1}) \\ &= argmax_{D,T} \prod_{t=1}^{|z|} p(z_t|e_t),\end{aligned}$$

这样顺滑任务就被很自然的融入到转移系统中。

柱搜索算法

贪心搜索的主要缺点是误差传播。一个不正确的动作将对其后所有的动作产生负面影响，从而导致误差的累计。减少误差传播的一种方式是柱搜索 (2011)[33]。因为对于每一条可能的转移路径，我们模型所采取的动作数总是等于输入句子的长度，因此可以直接使用柱搜索策略。我们在训练和测试过程中都采用柱搜索策略。在训练过程中，我们使用早期更新策略 (2004, 2001)[34, 35]。具体来说，在每个训练实例被解码的过程中，我们会记录正确路径的位置，如果正确路径在步骤 t 时不在前 K 个路径里面，则解码停止，并且使用正确路径作为正例，柱内的前 K 个候选作为负例来执行参数更新。我们使用全局优化方法 (2016, 2015)[36, 37] 来训练我们的柱搜索模型。

Scheduled Sampling

Scheduled sampling (2015)[38] 也可用于减少错误传播。贪心搜索的训练目标是最大化正确动作的概率，也就是说训练过程中每一步都会采取正确的动作。但是模型被用于测试数据的时候，其每一步采取的是所预测的概率最大的动作。训练与测试之间的差异可能会导致搜索过程中错误的快速累积。Scheduled sampling 使用随机采样的方法来缓解这种不一致性。具体来说，在训练过程中的每一步，我们以一定的概率 p 选择执行模型预测分数最高的动作，以 $(1-p)$ 的概率选择执行正确的动作。我们在具体的解码过程中也采用了 dropout 的策略 (2014)[39]

3.1.4 状态表示

为了更好地捕获全局的上下文信息，我们使用 LSTM 结构来表示每个状态的不同组成部分，包括 *buffer*, *action*, *stack*, 和 *output*。具体来讲，我们利用 LSTM-Minus 方法 (2016)[40] 对 *buffer* 进行建模，传统的 LSTM 对 *action* 和 *ouptut* 进行建模，stack LSTM (2015)[32] 来对 *stack* 段进行建模。

Buffer 表示

为了获得更丰富的信息表示，我们参考Wang and Chang (2016)[40] 的工作，使用 Bi-LSTM 来表示 *buffer*。首先，*buffer* 的前向和后向 LSTM 分别进行减法操作，具体操作为 $b_f = h_f(l) - h_f(f)$ 和 $b_b = b_b(f) - b_b(l)$，其中 $h_f(f)$ 和 $h_f(l)$ 分别是前向 LSTM 中第一个和最后一个词的隐藏向量。类似地，$h_b(f)$ 和 $h_b(l)$ 分别是后向 LSTM 中第一个和最后一个词的隐藏向量。然后这些相减结果被拼接成最终的 *buffer* 表示，即 $b_t = b_f \oplus b_b$。如图**??**所示，*buffer* 的前向和后向减法分别是 $b_f = h_f(to) - h_f(denver)$ 和 $b_b = h_b(denver) - h_b(to)$。这里 *to* 是 *buffer* 中的第一个词，*denver* 是最后一个词。然后 b_f 和 b_b 被拼接成 *buffer* 的最终表示。

Action 表示

针对每一个动作 a，我们都用一个对应向量 $e_a(a)$ 来表示，这些向量最终组成一个 looking-up 表 E_a。我们用传统的 LSTM 来表示转移系统所采取动作的完整历史。模型一旦采取行动 a，a 的向量表示 $e_a(a)$ 将被添加到 LSTM 的最右边的位置。

Stack 表示

我们使用 stack LSTM (2015)[32] 来表示正在被处理的不完整的非顺滑块。Stack LSTM 尝试用一个“堆栈指针”来扩展常规的 LSTM。对于传统的 LSTM，最新的输入总是添加在最右边的位置; 但是在 stack LSTM 中，当有新的输入进来时，由堆栈指针所指的位置来确定 LSTM 中的哪个单元提供 c_{t-1} 和 h_{t-1}。除了在序列末尾添加元素之外，stack LSTM 提供了一个 *pop* 操作，能将堆栈指针移动到前一个元素。因此，stack LSTM 可以被理解为一个堆栈，从而使得内容不会被覆盖。当采取 OUT 动作时，通过将堆栈指针移动到初始位置来清空 *stack*。当采取动作 DEL 时，*buffer* 顶部的元素将被直接添加到 stack LSTM 中。

duplicate features
$Duplicate(i, w_{i+k}), -15 \le k \le +15$ and $k \neq 0$: if w_i equals w_{i+k}, the value is 1, others 0
$Duplicate(p_i, p_{i+k}), -15 \le k \le +15$ and $k \neq 0$: if p_i equals p_{i+k}, the value is 1, others 0
$Duplicate(w_i w_{i+1}, w_{i+k} w_{i+k+1}), -4 \le k \le +4$ and $k \neq 0$: if $w_i w_{i+1}$ equals $w_{i+k} w_{i+k+1}$, the value is 1, others 0
$Duplicate(p_i p_{i+1}, p_{i+k} p_{i+k+1}), -4 \le k \le +4$ and $k \neq 0$: if $p_i p_{i+1}$ equals $p_{i+k} p_{i+k+1}$, the value is 1, others 0
similarity features
$fuzzyMatch(w_i, w_{i+k}), k \in \{-1, +1\}$: $similarity = 2 * num\_same\_letters/(len(w_i) + len(w_{i+k}))$. if $similarity > 0.8$, the value is 1, others 0

表 2 我们的模型用到的离散特征. *p*-词性. *w*-词.

Output 表示

我们使用传统的 LSTM 来表示 *output*。当采取 OUT 动作时，*buffer* 顶部的元素将被直接添加到该 LSTM 的最右侧。因为 *output* 保留的是最终输出句子的连续子序列，所以该 LSTM 表示可以被看作是一种伪语言模型，因此其一定程度上能保证生成句子的语法完整性，这对顺滑任务来说是非常重要的。

3.1.5 输入表示

对于每一个位置的输入，我们使用四个向量来表示：一个需要学习的词表示 w ; 一个固定的预训练好的词表示 $\widetilde{w}$ (2015)[41]; 一个需要学习的词性的表示 p; 人工提取的的特征表示 d。四个向量被拼接在一起后，经过一个 ReLU 层以学习特征组合：

$$x = max\{0, V[\widetilde{w}; w; p; d] + b\},$$

其中 V 表示向量的拼接。

参照Wang et al. (2016)[42] 的工作，针对句子中的每一个位置，我们抽取两种类型的离散特征（如表2所示），然后将它们转换成 0-1 向量 d。d 的维数为 78，和离散特征的个数相同。对于每一个词 x_t，如果对应的第 i 个特征模板为真，则对应的 d_i 为 1。Duplicate 类特征表示 x_t 在一定距离内是否具有相同的词或者词性。Similarity 特征表示 x_t 的字符串是否类似于其周围的词。

3.1.6 实验结果

我们的训练集采用英文的 Penn Treebank 中的 Switchboard 数据 (1993)[43]。对于英文的 Switchboard 数据，其提供两种标注文件：一个是同时标注有句法和顺滑的（MRG 文件），另一个是只标注有顺滑的（DPS 文件）。由于只标注顺滑的成本较低，因此许多 DPS 文件都没有相对应的 MRG 文件，也就是说 MRG 文件的规模要小于 DPS 文件的规模。为了直接与句法和顺滑联合方法 (2014, 2015)[28, 29] 进行比较，我们使用了规模相对

较小的 PAESED/MRG/SWBD 中的 MRG 文件。参照Charniak and Johnson (2001)[44] 中的实验设置，训练集包括所有 sw [23] * .mrg 文件，开发集包括 sw4 [5-9] * .mrg 文件，测试集包括 sw4 [0-1] *.mrg 文件。跟Honnibal and Johnson (2014)[28] 一样，我们将英文词统一转化为小写，并删除所有标点符号和标记为“XX”或以“-”结尾的词。“um”和“uh”会先被删除，并将“you know”和“i mean”合并成单个词。在我们的实验中，采用Qian and Liu (2013)[24] 提供的工具包生成的自动词性。

我们将基于转移的神经网络模型与目前的五个高性能系统进行比较。如表 3所示，我们的模型性能超过之前最好的方法，达到 87.5% 的 F1 分数。我们的模型比最好的句法和顺滑联合模型 UBT (2015)[29] 有 2.4 点的提升。序列标注方法中获得最好性能的是 Ferguson et al. (2015)[26] 的 semi-CRF 方法，其通过加入韵律特征达到了 85.4% 的 F1 分数。与此相比，我们的模型获得了 2.1 点的提升。我们的模型相对于之前性能最好的基于注意力机制的模型 (2016)[42] 也提高了 0.8 点。我们将模型的性能提升归功于其强大的学习全局的块级别信息的能力以及良好状态表示，如 stack-LSTM。

Method	**P**	**R**	**F1**
Our	91.1	84.1	**87.5**
Attention-based (2016)[42]	91.6	82.3	86.7
Bi-LSTM (2016)[22]	91.8	80.6	85.9
semi-CRF (2015)[26]	90.0	81.2	85.4
UBT (2015)[29]	90.3	80.5	85.1
M^3N (2013)[24]	-	-	84.1

表 3 和之前最好的方法在英文 Switchboard 测试集上的比较。

3.2 基于注意力机制的顺滑技术研究

3.2.1 实验动机

建模长距离依赖问题是解决顺滑问题的一个核心要点。以前的序列标注的方法 (2013, 2009, 2015)[24–26] 需要复杂的人工特征来捕捉长距离的依赖信息，但是其往往面临特征稀疏的问题。句法和顺滑联合的方法尝试将非顺滑块的识别融合到句法分析的过程中，这样非顺滑块和其对应的顺滑块就可以进行充分的交互。但是，这种方法需要额外在口语数据上标注句法数据，实用性不是很强。而且，其顺滑的性能严重依赖于句法分析的性能。另外，如何保证删除非顺滑块的句子的结构完整性也是顺滑任务中一个很重要的指标。传统的序列标注方法是没有能力建模句子的内部结构的。顺滑和句法联合任务虽然理论上能建模句子的内部句法结构，但是其性能受到句法分析模型的限制。针对上述

两个挑战，我们首次利用 seq-to-seq 的方法来解决顺滑问题。采用 seq-to-seq 方法主要有两个动机，一个是 seq-to-seq 框架在编码阶段会对输入句子学习一个全局的表示，这个全局表示可能会有助于解决长距离依赖问题，另一方面，seq-to-seq 方法本身可以被看做一个基于条件的语言模型，原始的输入句子相当于语言模型的条件，解码阶段相当于一个语言模型的生成过程，这样就有一定得能力保证生成句子的句法完整性。

3.2.2 背景介绍：seq-to-seq 模型

RNN Encoder-Decoder

基于 RNN Encoder-Decoder 的模型已经被应用到很多的序列生成任务中，如Sutskever et al., Cho et al. (2014, 2014)[45, 46]. 在 Encoder-Decoder 框架中，输入句子序列 $X = [x_1, ..., x_T]$ 首先被 encoder RNN 转化成一个固定的向量 c，i.e.

$$h_t = f(x_t, h_{t-1}); \qquad c = q(\{h_1, ..., h_T\}) \tag{1}$$

其中 $h_t \in R^n$ 是时刻 t 的隐层状态，c 是一个从隐层状态表示生成的向量。f 和 q 是一些非线性函数。比如，Sutskever et al. (2014)[45] 用 LSTM 来表示 f 和 $q(\{h_1, ..., h_T\}) = h_T$。

decoder 端经常被用来在给定上下文向量 c 和之前预测的词 $\{y_1, ..., y_{t-1}\}$ 的基础上预测下一个词 y_t。换句话说，decoder 将翻译 y 的联合概率表示成：

$$p(y) = \prod_{t=1}^{T} p(y_t \mid \{y_1, ..., y_{t-1}\}, c), \tag{2}$$

其中 $y = \{y_1, ..., y_{t-1}\}$. 通过 RNN，每个条件概率被建模成

$$p(y_t \mid \{y_1, ..., y_{t-1}\}, c) = g(y_{t-1}, s_t, c), \tag{3}$$

其中 g 是一个非线性的多层的表示 y_t 概率的函数, s_t 表示 RNN 的隐层状态.

注意力机制

注意力机制首先被Bahdanau et al. (2014)[47] 引入到 seq-to-seq 模型中，用来解决之前用一个固定向量来表示输入句子的问题，其在很多任务上取得了非常好的效果 (2017)[48]。在 decoding 过程中，注意力机制用一个动态变化的上下文向量 c_t 来表示输入端的信息。在 decoding 的每个时间点 t，c_t 的计算公式如下:

$$\begin{aligned} u_{tj} &= v^T tanh(W_1 h_j + W_2 s_{t-1}) \qquad j \in (1, ..., n) \\ a_{tj} &= \frac{exp(u_{tj})}{\sum_{k=1}^{T} exp(u_{tk})} \qquad j \in (1, ..., n) \\ c_t &= \sum_{j=1}^{n} a_{tj} h_j \end{aligned} \tag{4}$$

其中 s_{t-1} 是 $t-1$ 时刻 decoder 端的隐层状态. v, W_1, 和 W_2 是要学习的模型参数。最后, c_t 和 s_{t-1} 被拼接到一起用来预测下一个 y_t，并作为 decoder RNN 的下一步输入。我们称呼 u_{tj} 为相关性得分或者第 j 个输入对当前预测的权重。

3.2.3 方法设计

对于顺滑任务，其输出必须是对应输入的有序子序列。传统的 seq-to-seq 方法 (2014, 2015)[45, 49] 的输出候选是被事先固定好的，这就意味着其每一步只能从候选词典中选择一个概率最大的词作为输出。因为这个限制，我们不能直接将传统的 seq-to-seq 方法直接用到我们的顺滑任务中，其主要原因包括，（1）其可能会生成一些输入句子之外的词，（2）其不能生成一些不在词表中，但是却在输入句子中出现的词，（3）其没有能力保证生成词组的有序性。为了突破这些限制，我们提出采用了一个新的基于注意力机制的模型，如图**??**所示。

Encoder 设计

我们采用基于双向 LSTM 的 RNN 来将输入句子转化成一系列的隐层状态，其中每个隐层状态 h_t 对应于输入句子中的词 x_t。前向的 RNN 从左往右的读入输入句子 $x = (x_1, ..., x_T)$，产生隐层状态序列 $(\overrightarrow{h_1}, ..., \overrightarrow{h_T})$。对应的，后向 RNN 从右往左的读入输入句子，并产生隐层状态序列 $(\overleftarrow{h_1}, ..., \overleftarrow{h_T})$。我们将每个词对应位置的前向和后向的隐层状态进行拼接，组成最终的状态表示 $(h_1, ..., h_T)$，其中 $h_j = [\overrightarrow{h_j}; \overleftarrow{h_j}]$。每个向量 h_j 将第 j 个词及其周围的词的信息进行编码。

基于注意力机制的 decoder 端设计

参照Vinyals et al. (2015)[50] 的工作，我们通过修改 decoder 的框架来突破传统 seq-to-seq 方法对固定词表的限制。主要的改变包括：

- 我们将输入句子中的词序列 $(x_i, ...x_j)$ 作为候选，在 decoding 的每一步，我们从其中选一个词作为输出，而不是从一个固定词表中选词。这种机制保证了生成的词都包含在输入句子中，而且还能生成一些不在固定词表中出现，但是在输入句子中出现的词
- 一旦一个词 x_k 被选中，所有的词 $(x_i, ..., x_{k-1})$ 将会被删除，下一步的候选词序列将变成 $(x_{k+1}, ...x_l)$，这种机制能保证生成的词序列与它们出现在输入句子中的顺序相一致。

现在的关键问题是在每一步如何从候选词序列 $(x_i, ...x_j)$ 中选取正确的词。我们在每一步选择具有最高相关权重 u_{tk} 的词 x_k 作为最优的选择。在背景介绍部分，传统的注意力机制算法每一步都会在所有的输入句子序列 $(x_1, ..., x_T)$ 上计算相关权重向量 u_t。这种机制并不适合我们的任务，因为我们的任务在每一步只需要计算在 $(x_i, ...x_j)$ 上面的相关权重。因此，我们对传统的注意力机制进行修改，新的 $p(y_t \mid \{y_1, ..., y_{t-1}\}, c)$ 的计算公式如下：

$$\begin{aligned} u_{tk} &= v^T tanh(W_1 h_k + W_2 s_{t-1} + W_3 d_{k-i}) \qquad k \in (i, ..., j) \\ p(y_t \mid \{y_1, ..., y_{t-1}\}, c) &= softmax(u_t) \end{aligned} \tag{5}$$

其中 d_{k-i} 是上一个被选中的词 x_{i-1} 和对应的词 x_k 的距离的表示。距离信息对我们的模型是非常重要的，因为如果两个候选词有相似的上下文信息表示，那么后面的词被选中

的概率往往较大。一旦 x_k 被选中，我们将用其更新 decoder RNN，并且从 $(x_{k+1}, ..., x_l)$ 中选择下一个词。为了学习我们的基于注意力机制的模型的参数，我们在训练的时候，对于给定的输入句子 $\{(x_i, y_i)\}_{n=1}^{N}$ ，最小化其对应输出的负的对数概率：

$$-\sum_{i=1}^{N} log(p(y_i|x_i)) = -\sum_{i=1}^{N} log(\prod_{t=1}^{T} p(y_t \mid \{y_1, ..., y_{t-1}\}, c)) \quad (6)$$

详细的学习算法参见图 5 中的算法流程图。

3.2.4 实验结果

我们的训练集和对应的处理方法与上一节中的基于转移的方法一致。我们将基于注意力机制的神经网络模型与目前的四个高性能系统进行比较。如表 4所示，我们的模型性能超过之前最好的方法，达到 86.7% 的 F1 分数。我们的模型比最好的句法和顺滑联合模型 UBT (2015)[29] 有 1.6 点的提升。序列标注方法中获得最好性能的是 Ferguson et al. (2015)[26] 的 semi-CRF 方法，与此相比，我们的模型获得了 1.9 个点的提升。注意到我们的模型在准确率和召回率上同时取得了最好的结果。实验结果的分析显示我们的基于注意力机制的模型对于顺滑任务来说是一个不错的解决方案。

Method	**P**	**R**	**F1**
Attention-based	91.6%	82.3%	**86.7**%
M^3N (2013)[24]	-	-	84.1%
Joint Parser (2014)[28]	-	-	84.1%
semi-CRF (2015)[26]	90.1%	80.0%	84.8%
UBT (2015)[29]	90.3%	80.5%	85.1%

表 4 我们的基于注意力机制的模型与之前的四个最好的方法在英文 SwitchBoard 数据的测试集上的表示.

4 学位论文的主要内容、实施方案、及可行性论证

4.1 主要研究内容

本课题致力于解决顺滑问题，其主要有三个难点和挑战。一个是长距离依赖问题，这个问题可以从两方面来看，一方面 Edit 类型的短语块本身长度可能会很长，在 English Switchboard 数据中，最大的 Edit 块甚至包含有 15 个词，另一方面，Edit 类型和其对应的修正部分之间并不总是相邻，中间可能隔着很多的词，因此要判断一个词是不是 Edit 类型，需要看到非常远距离的信息。另一个难点是要保证顺滑后的句子的句法完整性，如果一个句子的关键成分被误认为 Edit 类型，将会导致非常严重的后果，另一个是非顺

滑块和对应的顺滑块之间存在着很紧密的联系，充分利用块之间的相似性将会变得非常重要。另一方面，公开的标注有顺滑现象的数据是十分稀少的，因而，利用来自互联网上的大量的未标注的口语数据（如字幕数据），以及大量的不带不流畅现象的数据（如新闻领域的数据）来帮助顺滑任务，将是一个十分有意义的研究点。

本课题针对上述难点和挑战，结合自然语言处理技术进行了相应的理论证明及实验尝试, 其具体内容如下：

(1) 传统的序列标注等方法很难充分的利用全局信息和建模句子的结构完整性。句法和顺滑的联合方法不但需要额外的句法标注，其性能也受到句法性能的制约，因此，我们提出了一个基于注意力机制的方案。我们的方法首先会对整个句子进行编码，之后再次序的生成顺滑块，其可以理解为一个全局的决策过程，因此能够充分的利用全局信息，而且其生成过程本身可以理解为一个语言模型的生成过程，其有一定的能力保证句法的结构完整性。实验结果也证明，我们的模型在标准的数据集上取得了较好的实验效果。（本部分工作已完成，并发表在自然语言处理的重要国际会议 COLING2016 上）

(2) 块之间的相似性对于顺滑任务是非常重要的，然而之前只有很少的工作尝试显示的去利用块信息。有工作尝试用 semi-CRF 模型去建模块信息，但是由于受到马尔可夫假设的约束，其也只能利用局部的块信息。我们的工作尝试利用一种不带句法信息的转移系统来解决顺滑问题。基于转移的方法本身能很好的利用全局信息，所以其能很好的解决长距离依赖问题，而且通过引入一个 *stack* 来存储被标注为非顺滑的词序列，我们的模型能很好的利用块之间的一些相似性等信息。实验结果也证明，我们的模型在标准的数据集上取得了较好的实验效果。（本部分工作已完成，并发表在自然语言处理的重要国际会议 EMNLP2017 上）

(3) 以前的工作基本都是从左往右进行决策的，这种方案的一个主要问题是决策是相对局部的，因为在做当前的决策时，其只能利用左边的决策信息，对右边的部分是未知的。以“i want a flight to bosthon to denver”为例，在决定”to boston” 是否应该被删除的时候，我们如果知道”to denver” 是一个完整的块，将对决策产生很大的帮助。而且从人的思考角度来看，我们也倾向于先识别出“to boston”,“to denver”这些块，然后再决定删除哪些部分。因此，我们将尝试采用 easy-first 的方案，主要思路是先识别一些比较容易生成的块，如“to boston”等，然后再去决定删除那些块。easy-first 方案由于决策是全局的，而且能充分的利用块信息，以及建模相邻块之间的联系（与语言模型类似），因此其理论上能很好的处理顺滑任务的三个挑战。（本部分工作正在进行中）

(4) 公开的标注有顺滑现象的数据是十分稀少的，因而，利用来自互联网上的大量的未标注的口语数据（如字幕数据），以及大量的不带不流畅现象的书面语数据（如新闻领域的数据）来帮助顺滑任务，将是一个十分有意义的研究点。我们将尝试用对偶学习（dual learning） (2016)[51] 的方法来降低模型对有监督数据的依赖。对偶学习的最关键一点在于，给定一个原始任务模型，其对偶任务的模型可以给其提供反馈；同样的，

给定一个对偶任务的模型，其原始任务的模型也可以给该对偶任务的模型提供反馈；从而这两个互为对偶的任务可以相互提供反馈，相互学习、相互提高。对于我们的顺滑任务来说，从口语到书面语的转化，以及从标准数据到口语数据的转化可以建模成一个对偶学习的过程，从而通过对偶学习的过程，充分的将大量的口语数据和书面语数据结合起来，降低对有监督数据的依赖。（本部分工作正在进行中）

上述各研究内容的关系如图**??**所示。

4.2 实施方案

由于四个研究点中的前两个已经完成，并且被录用，所以实施方案部分将主要介绍后两个研究点。研究工作将从如下几个方面展开：

4.2.1 基于 easy-first 的方案

easy-first 严格来说并不是一个具体的机器学习模型，其只是一种区别于传统的从左往右进行决策的新思路。在 easy-first 的决策过程中，对于每一步，都有很多候选项，这些候选项可能来源于句子序列中的不同位置，模型会倾向于选择置信度最高的候选项，然后对模型进行更新。所以其核心思想是优先选择那些最容易做出的决策，这些决策会对后面比较难的决策进行帮助，这个思想也很符合人在面对一些问题的思考方式。

以句法分析为例，传统的模型都是从左往右进行决策的，这种搜索方式在每一步的时候，只能依赖于前面的决策步骤，也就是说只看到了句子前面部分的结构，但是看不到右边还未分析的子句的内部结构。但是采用 easy-first 的搜索方案 (2010, 2016)[52, 53]，其往往会优先选择一些比较容易识别的弧，然后再在这些已识别的弧的基础上进行更难的弧的识别，这就摆脱了传统的从左往右决策的只能看到左边信息的缺点。如图**??**所示，里面给出了一个分析“a brown fox jumped with joy”这句话的依存句法结构的过程，我们可以看到其最先识别出的往往是“a brown fox”，“with joy”这些比较容易识别的结构，这些简单的结构会对后面比较复杂的结构识别带来很大的帮助。

受到 easy-first 句法分析的启发，我们也尝试用 easy-first 的思路来解决顺滑问题。以“i want a flight to bosthon to denver”为例，模型可以先识别一些比较容易生成的块，如“to boston”等，然后再去决定删除哪些块。easy-first 方案由于决策是全局的，而且能充分的利用块信息，以及建模相邻块之间的联系（与语言模型类似），因此其理论上能很好的处理顺滑任务的三个挑战。

对于具体的搜索过程，我们设计了三种具体的动作：

- COMBINE：这个动作会将相邻的两个块给拼接成一个块。
- DEL-LEFT：该动作会把相邻的块的左边部分给删除掉
- DEL-RIGHT：该动作会把相邻的块的右边部分给删除掉

表 5 显示了处理句子“a flight to boston to denver”的动作序列。需要注意的是，对于 easy-first 方案，其每一步的正确选择并不只有一个，也就是说合理的搜索路径不止一个。

表 5 对于输入 *a flight to boston to denver* 的处理过程

Step	Action	Buffer
0	NO	[a, flight, to, boston, to, denver]
1	COMBINE	[a-flight, to, boston, to, denver]
2	COMBINE	[a-flight, to, boston, to-denver]
3	COMBINE	[a-flight, to-boston, to-denver]
4	DEL-RIGHT	[a-flight,to-denver]
5	COMBINE	[a-flight-to-denver]

对于具体的模型实现，我们首先要对句子中的每个位置的词进行表示。在这里，我们使用了循环神经网络（Recurrent Neural Net，RNN）对每个词的上下文建模。循环神经网络相比传统的前馈神经网络，引入了定向循环的结构，能够处理前后关联的、不定长的序列输入问题。在传统神经网络模型中，层与层之间是全连接的，但是每层之间的节点是无连接的。在 RNN 中，因为其特有的定向循环结构，使得隐含层之间的节点不再是无连接的，而是有连接的，即一个隐含层的输入不仅同当前输入有关，还包含上一时刻隐含层的输出。由于传统的 RNN 模型展开后相当于多层的神经网络，层数对应历史输入数据的个数，层数过多会导致训练参数时梯度消失和历史信息损失的问题。也就是说，越远的序列输入对训练权值所能起到的“影响”越小，所以训练的结果往往偏向于新的信息，即不太能有较长的记忆功能，能够利用的历史信息非常有限。长短时记忆模型（Long Short-Term Memory，LSTM）模型能够解决上述问题。图**??**是 LSTM 的单元结构图。在 LSTM 单元中，设计了专门的记忆单元（memory cell）用于储存历史信息。历史信息的更新和使用分别受三个门的控制——输入门（input gate）、遗忘门（forget gate）和输出门（output gate）。其中，x_t 为 t 时刻 LSTM 单元的输入，C_t 为 t 时刻 LSTM 单元的值，h_t 为 t 时刻 LSTM 单元的输出。LSTM 单元的更新过程如下：

$$i_t = \sigma(W^{(i)} \cdot [x_t; h_{t-1}] + b^{(i)}) \tag{7}$$

$$f_t = \sigma(W^{(f)} \cdot [x_t; h_{t-1}] + b^{(f)}) \tag{8}$$

$$o_t = \sigma(W^{(o)} \cdot [x_t; h_{t-1}] + b^{(o)}) \tag{9}$$

$$\tilde{C}_t = tanh(W^{(c)} \cdot [x_t; h_{t-1}] + b^{(c)}) \tag{10}$$

$$C_t = i_t \odot \tilde{C}_t + f_t \odot C_{t-1} \tag{11}$$

$$h_t = o_t \odot tanh(C_t) \tag{12}$$

这里，$\odot$ 表示两个向量按位相乘，$W^{(i)}, b^{(i)}, W^{(f)}, b^{(f)}, W^{(o)}, b^{(o)}, W^{(c)}, b^{(c)}$ 是神经网络的

参数，σ 表示 sigmoid 函数：

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{13}$$

用一个双向 LSTM 对句子进行表示学习之后，我们得到了每个位置的词对应的向量表示。下一步的重点是如何保存解码过程中生成的一个个的局部的块。在我们的方案中，我们尝试利用另外一个 LSTM 网络对每个块进行建模。一旦执行完 COMBINE 操作，我们会将相邻的右边的块中的词的表示依次拼接到后面的 LSTM 表示中去。图**??**是词表示和块组合的整体框架。

每一步，我们的模型都会选择最高得分的动作 (i,l)，其中 i 表示第 i 个块，l 表示对应的动作。训练的目标就是让每一步正确动作的得分高于错误动作的得分。我们采用 max-margin 作为损失函数，每一步的具体的损失函数为：

$$max\{0, 1 - max_{(i,l)\in G} Score(i,l) + max_{(i,l)\in A\setminus G} Score(i,l)\} \tag{14}$$

其中 G 表示在当前步骤下所有可能的正确的动作，A 表示当前步骤下所有可能的动作。$Score(i,l)$ 表示对应动作的得分。

4.2.2 基于对偶学习的方案

很多人工智能的应用涉及两个互为对偶的任务，例如机器翻译中从中文到英文翻译和从英文到中文的翻译互为对偶、语音处理中语音识别和语音合成互为对偶、问答系统中回答问题和生成问题互为对偶。这些互为对偶的人工智能任务可以形成一个闭环，使从没有标注的数据中进行学习成为可能。对偶学习的最关键一点在于，给定一个原始任务模型，其对偶任务的模型可以给其提供反馈；同样的，给定一个对偶任务的模型，其原始任务的模型也可以给该对偶任务的模型提供反馈；从而这两个互为对偶的任务可以相互提供反馈，相互学习、相互提高。

对于顺滑任务来说，我们的目的是将口语句子中的一些不流畅的部分删除，最终转换成偏书面语风格的句子。这个可以理解为一个原始任务。那么，相对应的对偶任务可以理解为一个书面语到口语的转化过程。对于对偶学习来说，可以把两个对偶的任务叫做智能体。我们的训练过程可以看作两个智能体互相交流的过程：

- 第一个 Agent 只懂得口语，可以将口语的信息通过一个有噪声的信道发送给第二个 Agent，这个有噪声的信道可以使用我们之前的基于转移的模型将口语转换成第二个 Agent 懂得的书面语言。
- 第二个 Agent 只懂得书面语言，她接收到第一个 Agent 发来的信息（已经被转化成书面语言），然后她会对该信息进行判断（比如语言模型得分，语句是否完整等），注意她并不能准确的判断该信息的正确性，因为原始的信息对她来说是不可见的。之后她将此时的信息通过另一个有噪声的信道再发送回第一个 Agent，该信道使用另一个基于转移的生成模型将书面语的信息转换为口语。

- 第一个 Agent 接收到第二个 Agent 传回来的信息后，就与原始的口语句子进行比对，看两者的一致性。通过这个反馈（feedback），两个 Agent 都可以知道两个信道（两个转化模型）是否执行的比较好（perform well）以及是否可以相互促进。
- 训练也可以从第二个 Agent 开始。这两个 Agent 通过这个闭环的过程，根据反馈可以不断的提升两个转化模型的性能。

从上述描述中可以看出，尽管这两个 Agent 都没有与之标记的的标签，但是她们依然可以得到关于这两个模型的反馈，并依据此反馈来提高模型的转化能力。由此也可以看出，我们并不需要大量的训练集（为了加快模型的训练速度，也可以先使用少量的标注有顺滑的数据集）。

我们准备用策略梯度方法来训练我们的模型。策略梯度方法的基本思想非常简单：如果我们在执行某个动作之后，观测到了一个很大的回报，我们就通过调整策略（在当前策略函数的参数上加上它的梯度）来增加这个状态下执行这个动作的概率；相反，如果我们在执行某个动作之后，观测到了一个很小的回报，甚至是负的回报，那么我们就需要调整策略（在当前策略函数的参数上减去它的梯度），以降低在这个状态下执行这个动作的概率。对偶学习的学习过程如图**??**所示

对偶学习已经在机器翻译等任务上取得了不错的效果，相关的实验也已经证明只用少量的有监督语料就可以取得非常好的效果。对于顺滑任务来说，如果通过对偶学习确实能减少了对有监督数据的依赖，将是一件非常有意义的工作。这块的更具体的实验方案还需要进一步完善。

4.3 可行性分析

1. 本人长期关注自然语言处理相关技术、理论等。具有一定的理论基础，已经阅读并整理了大量的相关文献，对国内外研究现状有了较清晰的了解，详细掌握了目前顺滑研究中存在的问题以及主要的研究方向。
2. 我们的初期工作已经发表在 COLING2016，EMNLP2017 等重要国际学术会议上，其创新性得到了同行的认可。
3. 所在的社会计算与信息检索研究中心经过多年的技术积累，已经基本掌握了自然语言处理中词法、句法、语义、复述以及翻译等多种底层的关键技术。其中在上述各个方面都积累了大量的代码与数据，研究中心的语言技术平台享誉国内，加之本人对顺滑技术有较深刻的理解，这些技术和数据都为完成本课题提供了良好的支持。同时科大讯飞研究院负责标注了大量的中文顺滑数据集，对本论文研究的开展起到了很大的帮助。

5 论文的进度安排与预期目标

5.1 进度安排

2015 年 9 月 - 2016 年 3 月：博士论文选题，参考文献收集。

2016 年 4 月 - 2016 年 7 月：分别完成基于 LSTM，LSTM-CRF，和注意力机制框架的顺滑分析与研究，完成论文写作并发表

2016 年 8 月 - 2017 年 4 月：完成基于转移的顺滑分析与研究，完成论文写作并发表

2017 年 5 月 - 2018 年 3 月：完成基于 easy-first 方案的顺滑分析与研究

2018 年 4 月 - 2018 年 12 月：完成基于 dual learning 方案的顺滑分析与研究

2019 年 1 月 - 2019 年 6 月：撰写毕业论文，申请毕业答辩。

5.2 预期目标

本课题致力于解决顺滑问题。经过前期的探索，我们分别从三个角度，针对顺滑问题的特点，提出了三个方法用于解决顺滑的关键问题。首先，我们提出的 LSTM-CRF 方案可以有效的避免传统 LSTM 方案的 label 偏置问题，取得了比 LSTM 更好的实验结果。其次，我们首次尝试用生成的方法来解决顺滑问题，这种生成式的方法可以利用更加全局的信息，取得了不错的实验结果。最后，针对非顺滑块和顺滑块之间存在紧密关联这一特点，我们尝试了基于转移的方案来解决顺滑问题，取得了目前最好的效果。前期的工作已经在顺滑问题上取得了目前最好的结果，下一步的工作，我们除了对模型本身进行改进，提出了 easy-first 方案外，还将尝试利用 dual learning 的方法来利用大规模的生语料，从而减少对有标注语料的依赖。综合这些工作，我们预期目标为从各个角度去探索和解决顺滑问题，并最终构建一个高效，而且只需要少量标注数据的实用的顺滑系统。

6 学位论文预期创新点

与前人已发表的研究成果相比，本学位论文的预期创新点主要有以下几点：

（1）提出了一个利用注意力机制来直接生成顺滑块的方法。传统的序列标注等方法很难充分的利用全局信息，而我们的基于注意力机制的方法首先会对整个句子进行编码，之后再次序的生成顺滑块，其可以理解为一个全局的决策过程，因此能够充分的利用全局信息。

（2）提出了一个基于转移的顺滑方案。顺滑任务的一个主要的特点是非顺滑块和对应的顺滑块之间有很强的相似性，我们设计的基于转移的方案可以很好的利用这些块之间联系，实验结果表明基于转移的方案取得了目前最好的试验效果。

（3）正在实现的基于 easy-first 的顺滑方案。以前的方案大都是从左往右进行决策，

这种方案的主要问题是决策是局部的，而且只能看到很少的右边的信息。我们设计的基于 easy-first 的方案的生成过程是一个由易到难的过程，避开了从左往右进行决策的问题，而且我们的方案在决策的过程中还能充分的利用块信息。

（4）正在实现的基于 dual learning 的顺滑方案。以前的方案大部分只关注模型本身的改进，很少有利用大量未标注数据的方案。我们的方案通过正向和逆向两个模型的互相交互，将大量的口语数据和标准的新闻数据引入进来，从而期望能利用更少的语料，取得不错的实验效果。

7 为完成课题已具备和所需的条件、外协计划及经费

1. 相关工作的理论基础的积累：包括文献整理、平台搭建等工作（已完成）；
2. 基于注意力机制的顺滑方案：首次从生成的角度来解决顺滑问题（已完成）；
3. 基于转移的顺滑方案：该方案可以充分的利用不同块之间的信息（已完成）；
4. 基于 easy-first 的方案：已完成初步的方案设计和代码编写，进一步的实验结果即将得出；
5. 基于 dual learning 的方案：已完成初步的方案设计，即将开始编写代码；
6. 本论文的课题研究经费充足，能够保证相关工作正常进行。
7. 本论文的课题研究得到了科大讯飞公司合肥研究院（总部）的鼎力支持。

8 预计研究过程中可能遇到的困难、问题，以及解决的途径

1. 在 dual learning 方案中如何约束逆向的生成过程：对于 dual learning 方案，逆向的生成过程是由顺滑的（不含不流畅块）句子生成含有不流畅块的句子，在理论上，在顺滑句子的基础上添加任何成分都是合理的，但是这样无限制的生成方案可能会带来严重的问题，也就是说正向和逆向不是严格对称的，因此如何消除或者减少这种不对称将是实验过程中需要重点关注的部分。
2. 训练语料不足问题：由于复杂的神经网络需要大规模的训练语料，现有的 English SwitchBoard 数据只有八万句左右，当模型变复杂的时候，其规模略小。由于实验室跟讯飞有相关的合作，讯飞方面将负责标注大量的中文语料，从而能很好的验证我们模型在大规模语料上的效果。
3. 实验结果不理想的问题：实验效果不理想可能是由多种不同的原因造成的。如果遇到这种问题，首先会排查代码是否有 bug。其次，会对实验结果进行详细的分析，进而根据实验结果，决定是否对模型进行调整和优化。

9 发表论文情况

1. **Shaolei Wang**, Wanxiang Che, Yue Zhang, Meishan Zhang and Ting Liu. Transition-Based Disfluency Detection using LSTMs. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2017). 2017.09. Copenhagen, Denmark.(已录用，CCF 排名 B 类，重要国际会议，poster)
2. **Shaolei Wang**, Wanxiang Che and Ting Liu. A Neural Attention Model for Disfluency Detection. In Proceedings of the 26th International Conference on Computational Linguistics (Coling 2016). 2016.12. Osaka, Japan.(已录用，CCF 排名 B 类，重要国际会议，oral)
3. **Shaolei Wang**, Wanxiang Che, Yijia Liu and Ting Liu. Enhancing Neural Disfluency Detection with Hand-Crafted Features. In Proceedings of China National Conference on Chinese Computational Linguistics (CCL 2016). 2016.10. Yantai, China.

主要参考文献

[1] Dozat T, Manning C D. Deep biaffine attention for neural dependency parsing[C]//Proceedings of ICLR. Toulon, France: [s.n.] , 2017.

[2] McDonald R, Crammer K, Pereira F. Online Large-Margin Training of Dependency Parsers[C]//Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 05). Ann Arbor, Michigan: Association for Computational Linguistics, 2005:91–98.

[3] Yamada H, Matsumoto Y. Statistical dependency analysis with support vector machines[C]//Proceedings of IWPT. Nancy, France: [s.n.] , 2003，3:195–206.

[4] Che W, Shao Y, Liu T, et al. SemEval-2016 Task 9: Chinese Semantic Dependency Parsing[C]//Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016). San Diego, California: Association for Computational Linguistics, 2016:1074–1080.

[5] Che W, Zhang M, Shao Y, et al. SemEval-2012 Task 5: Chinese Semantic Dependency Parsing[C]//SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012). Montréal, Canada: Association for Computational Linguistics, 2012:378–384.

[6] Oepen S, Kuhlmann M, Miyao Y, et al. SemEval 2014 Task 8: Broad-Coverage Semantic Dependency Parsing[C]//Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014). Dublin, Ireland: Association for Computational Linguistics and Dublin City University, 2014:63–72.

[7] Oepen S, Kuhlmann M, Miyao Y, et al. SemEval 2015 Task 18: Broad-Coverage Semantic Dependency Parsing[C]//Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015). Denver, Colorado: Association for Computational Linguistics, 2015:915–926.

[8] Hajič J, Ciaramita M, Johansson R, et al. The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages[C]//Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task. Boulder, Colorado: Association for Computational Linguistics, 2009:1–18.

[9] Hockenmaier J, Steedman M. CCGbank: a corpus of CCG derivations and dependency structures extracted from the Penn Treebank[J]. Computational Linguistics, 2007, 33(3):355–396.

[10] Banarescu L, Bonial C, Cai S, et al. Abstract Meaning Representation for Sembanking[C]//Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse.[S.l.]: Association for Computational Linguistics, 2013:178–186.

[11] Kromann M T. The Danish Dependency Treebank and the Underlying Linguistic Theory[C]//Proceedings of the Second Workshop on Treebanks and Linguistic Theories (TLT). .[S.l.]: [s.n.] , 2003.

[12] Miyao Y, Ninomiya T, Tsujii J. Corpus-Oriented Grammar Development for Acquiring a Head-Driven Phrase Structure Grammar from the Penn Treebank[C]//Natural Language Processing – IJCNLP 2004. Berlin, Heidelberg: Springer, 2005:684–693.

[13] Sagae K, Tsujii J. Shift-Reduce Dependency DAG Parsing[C]//Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1. .[S.l.]: [s.n.] , 2008:753–760.

[14] Titov I, Henderson J, Merlo P, et al. Online Graph Planarisation for Synchronous Parsing of Semantic and Syntactic Dependencies[C]//Proceedings of the 21st International Jont Conference on Artifical Intelligence. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2009:1562–1567.

[15] Zhang X, Du Y, Sun W, et al. Transition-based Parsing for Deep Dependency Structures[J]. Computational Linguistic, 2016, 42(3):353–389.

[16] McDonald R, Pereira F. Online learning of approximate dependency parsing algorithms[C]//11th Conference of the European Chapter of the Association for Computational Linguistics. .[S.l.]: [s.n.] , 2006.

[17] Eisner J M. Three new probabilistic models for dependency parsing: An exploration[C]//Proceedings of the 16th conference on Computational linguistics-Volume 1. .[S.l.]: [s.n.] , 1996:340–345.

[18] Martins A F, Almeida M S. Priberam: A turbo semantic parser with second order features[C]//Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014). .[S.l.]: [s.n.] , 2014:471–476.

[19] Martins A F. AD3: A Fast Decoder for Structured Prediction[J]. Advanced Structured Prediction. MIT Press, Cambridge, MA, USA, 2014.

[20] Peng H, Thomson S, Smith N A. Deep Multitask Learning for Semantic Dependency Parsing[C]//Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vancouver, Canada: Association for Computational Linguistics, 2017:2037–2048.

[21] Kiperwasser E, Goldberg Y. Simple and Accurate Dependency Parsing Using Bidirectional LSTM Feature Representations[J]. Transactions of the Association for Computational Linguistics, 2016, 4:313–327.

[22] Zayats V, Ostendorf M, Hajishirzi H. Disfluency detection using a bidirectional LSTM[J]. arXiv preprint arXiv:1604.03209, 2016.

[23] Hough J, Schlangen D. Recurrent Neural Networks for Incremental Disfluency Detection[C]//Sixteenth Annual Conference of the International Speech Communication Association. .[S.l.]: [s.n.] , 2015.

[24] Qian X, Liu Y. Disfluency Detection Using Multi-step Stacked Learning.[C]//HLT-NAACL. .[S.l.]: [s.n.] , 2013:820–825.

[25] Georgila K. Using integer linear programming for detecting speech disfluencies[C]//Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers. .[S.l.]: [s.n.] , 2009:109–112.

[26] Ferguson J, Durrett G, Klein D. Disfluency Detection with a Semi-Markov Model and Prosodic Features[C]//Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.[S.l.]: Association for Computational Linguistics, 2015:257–262.

[27] Rasooli M S, Tetreault J R. Joint Parsing and Disfluency Detection in Linear Time.[C]//EMNLP. .[S.l.]: [s.n.] , 2013:124–129.

[28] Honnibal M, Johnson M. Joint incremental disfluency detection and dependency parsing[J]. Transactions of the Association for Computational Linguistics, 2014, 2:131–142.

[29] Wu S, Zhang D, Zhou M, et al. Efficient Disfluency Detection with Transition-based Parsing[C]//Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers).[S.l.]: Association for Computational Linguistics, 2015:495–503.

[30] Zhang M, Zhang Y, Fu G. Transition-Based Neural Word Segmentation[C]//Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Berlin, Germany: Association for Computational Linguistics, 2016:421–431, http://www.aclweb.org/anthology/P16-1040.

[31] Nivre J. Algorithms for deterministic incremental dependency parsing[J]. Computational Linguistics, 2008, 34(4):513–553.

[32] Dyer C, Ballesteros M, Ling W, et al. Transition-Based Dependency Parsing with Stack Long Short-Term Memory[C]//Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). Beijing, China: Association for Computational Linguistics, 2015:334–343, http://www.aclweb.org/anthology/P15-1033.

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

[34] Collins M, Roark B. Incremental parsing with the perceptron algorithm[C]//Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics. .[S.l.]: [s.n.] , 2004:111.

[35] Giles R C S L L. Overfitting in Neural Nets: Backpropagation, Conjugate Gradient, and Early Stopping[C]//Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference. .[S.l.]: [s.n.] , 2001，13:402.

[36] Andor D, Alberti C, Weiss D, et al. Globally Normalized Transition-Based Neural Networks[C]//Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Berlin, Germany: Association for Computational Linguistics, 2016:2442–2452, http://www.aclweb.org/anthology/P16-1231.

[37] Zhou H, Zhang Y, Huang S, et al. A Neural Probabilistic Structured-Prediction Model for Transition-Based Dependency Parsing[C]//Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). Beijing, China: Association for Computational Linguistics, 2015:1213–1222, http://www.aclweb.org/anthology/P15-1117.

[38] Bengio S, Vinyals O, Jaitly N, et al. Scheduled sampling for sequence prediction with recurrent neural networks[C]//Advances in Neural Information Processing Systems. .[S.l.]: [s.n.] , 2015:1171–1179.

[39] Srivastava N, Hinton G, Krizhevsky A, et al. Dropout: A simple way to prevent neural networks from overfitting[J]. The Journal of Machine Learning Research, 2014, 15(1):1929–1958.

[40] Wang W, Chang B. Graph-based Dependency Parsing with Bidirectional LSTM[C]//Proc. of ACL. .[S.l.]: [s.n.] , 2016:2306–2315.

[41] Ling W, Dyer C, Black A, et al. Two/too simple adaptations of word2vec for syntax problems[C]//Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. .[S.l.]: [s.n.] , 2015:1299–1304.

[42] Wang S, Che W, Liu T. A Neural Attention Model for Disfluency Detection[C]//Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. Osaka, Japan: The COLING 2016 Organizing Committee, 2016:278–287, http://aclweb.org/anthology/C16-1027.

[43] Marcus M P, Marcinkiewicz M A, Santorini B. Building a large annotated corpus of English: The Penn Treebank[J]. Computational linguistics, 1993, 19(2):313–330.

[44] Charniak E, Johnson M. Edit detection and parsing for transcribed speech[C]//Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies. .[S.l.]: [s.n.] , 2001:1–9.

[45] Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks[C]//Advances in neural information processing systems. .[S.l.]: [s.n.] , 2014:3104–3112.

[46] Cho K, van Merrienboer B, Gulcehre C, et al. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation[C]//Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar: Association for Computational Linguistics, 2014:1724–1734, http://www.aclweb.org/anthology/D14-1179.

[47] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate[J]. In ICLR 2015, arXiv preprint arXiv:1409.0473, 2014.

[48] Yin Q, Zhang Y, Zhang W, et al. 2017. A Deep Neural Network for Chinese Zero Pronoun Resolution[C]//Proceedings of the 26th International Conference on Artificial Intelligence.[S.l.]: AAAI Press, IJCAI'17.

[49] Rush A M, Chopra S, Weston J. A Neural Attention Model for Abstractive Sentence Summarization[C]//Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Lisbon, Portugal: Association for Computational Linguistics, 2015:379–389, http://aclweb.org/anthology/D15-1044.

[50] Vinyals O, Fortunato M, Jaitly N. Pointer Networks[M]//. Cortes C, Lawrence N D, Lee D D, et al. Advances in Neural Information Processing Systems 28.[S.l.]: Curran Associates, Inc., 2015:2692–2700, http://papers.nips.cc/paper/5866-pointer-networks.pdf.

[51] He D, Xia Y, Qin T, et al. Dual learning for machine translation[C]//Advances in Neural Information Processing Systems. .[S.l.]: [s.n.] , 2016:820–828.

[52] Goldberg Y, Elhadad M. An efficient algorithm for easy-first non-directional dependency parsing[C]//Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. .[S.l.]: [s.n.] , 2010:742–750.

[53] Kiperwasser E, Goldberg Y. Easy-first dependency parsing with hierarchical tree LSTMs[J]. arXiv preprint arXiv:1603.00375, 2016.

[54] Huang Z, Xu W, Yu K. Bidirectional LSTM-CRF models for sequence tagging[J]. Computer Science, 2015.

[55] Ma X, Hovy E. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF[J]. arXiv preprint arXiv:1603.01354, 2016.

[56] Pascanu R, Mikolov T, Bengio Y. On the difficulty of training recurrent neural networks[J]. arXiv preprint arXiv:1211.5063, 2012.

[57] Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural computation, 1997, 9(8):1735–1780.

[58] Lafferty J, McCallum A, Pereira F C. Conditional random fields: Probabilistic models for segmenting and labeling sequence data[J]. 2001.

[59] Shriberg E E. Preliminaries to a theory of speech disfluencies[D].[S.l.]: Citeseer, 1994.

[60] Irsoy O, Cardie C. Opinion Mining with Deep Recurrent Neural Networks.[C]//EMNLP. .[S.l.]: [s.n.] , 2014:720–728.

[61] Cho E, Ha T L, Waibel A. CRF-based Disfluency Detection using Semantic Features for German to English Spoken Language Translation[J]. IWSLT, Heidelberg, Germany, 2013.

[62] Johnson M, Charniak E. A TAG-based noisy channel model of speech repairs[C]//Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics. .[S.l.]: [s.n.] , 2004:33.

[63] Zwarts S, Johnson M. The impact of language models and loss functions on repair disfluency detection[C]//Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. .[S.l.]: [s.n.] , 2011:703–711.

[64] Meteer M W, Taylor A A, MacIntyre R, et al. Dysfluency annotation stylebook for the switchboard corpus[M].[S.l.]: University of Pennsylvania, 1995.

[65] Jørgensen F. The effects of disfluency detection in parsing spoken language[J]. 2007.

[66] Shriberg E, Bates R, Stolcke A. A prosody only decision-tree model for disfluency detection[C]//Fifth European Conference on Speech Communication and Technology. .[S.l.]: [s.n.] , 1997.

[67] Wang X, Sim K C, Ng H T. Combining punctuation and disfluency prediction: An empirical study[C]//Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). .[S.l.]: [s.n.] , 2014:121–130.

[68] Hassan H, Schwartz L, Hakkani-Tür D, et al. Segmentation and disfluency removal for conversational speech translation[C]//Fifteenth Annual Conference of the International Speech Communication Association. .[S.l.]: [s.n.] , 2014.

[69] Ostendorf M, Hahn S. A sequential repetition model for improved disfluency detection.[C]//INTERSPEECH. .[S.l.]: [s.n.] , 2013:2624–2628.

[70] Hassan Awadalla H, Schwartz L, Hakkani-Tür D, et al. Segmentation and Disfluency Removal for Conversational Speech Translation[R].[S.l.]: [s.n.] , 2014. https://www.microsoft.com/en-us/research/publication/segmentation-and-disfluency-removal-for-conversational-speech-translation/.

[71] Liu Y, Shriberg E, Stolcke A, et al. Enriching speech recognition with automatic detection of sentence boundaries and disfluencies[J]. Audio, Speech, and Language Processing, IEEE Transactions on, 2006, 14(5):1526–1540.

[72] Buckman J, Ballesteros M, Dyer C. Transition-based dependency parsing with heuristic backtracking[C]//Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing; 2016 Nov 1-5; Austin, Texas, USA. Stroudsburg (USA): Association for Computational Linguistics (ACL); 2016. p. 2313-18. .[S.l.]: [s.n.] , 2016.

[73] Zhang Y, Clark S. A tale of two parsers: investigating and combining graph-based and transition-based dependency parsing using beam-search[C]//Proceedings of the Conference on Empirical Methods in Natural Language Processing. .[S.l.]: [s.n.] , 2008:562–571.

[74] 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. Portland, Oregon, USA: Association for Computational Linguistics, 2011:188–193, http://www.aclweb.org/anthology/P11-2033.

[75] Zhu M, Zhang Y, Chen W, et al. Fast and Accurate Shift-Reduce Constituent Parsing[C]//Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Sofia, Bulgaria: Association for Computational Linguistics, 2013:434–443, http://www.aclweb.org/anthology/P13-1043.

[76] Watanabe T, Sumita E. Transition-based Neural Constituent Parsing[C]//Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). Beijing, China: Association for Computational Linguistics, 2015:1169–1179, http://www.aclweb.org/anthology/P15-1113.

[77] Yao K, Peng B, Zweig G, et al. Recurrent conditional random field for language understanding[C]//Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on. .[S.l.]: [s.n.] , 2014:4077–4081.

[78] Lample G, Ballesteros M, Subramanian S, et al. Neural architectures for named entity recognition[J]. arXiv preprint arXiv:1603.01360, 2016.

[79] Graves A. Generating sequences with recurrent neural networks[J]. arXiv preprint arXiv:1308.0850, 2013.

[80] Wang S, Che W, Liu Y, et al. Enhancing Neural Disfluency Detection with Hand-Crafted Features[C]//China National Conference on Chinese Computational Linguistics. .[S.l.]: [s.n.] , 2016:336–347.