上海交通大学博士学位论文

大规模任务型口语理解

博 士 研 究 生：朱　苏

学　　号：x

导　　师：俞凯教授

申 请 学 位：工学博士

学　　科：计算机科学与技术专业

所 在 单 位：计算机科学与工程系

答 辩 日 期：2020 年 9 月 2 日

授予学位单位：上海交通大学

Dissertation Submitted to Shanghai Jiao Tong University
for the Degree of Doctor

TOWARDS LARGE-SCALE TASK-ORIENTED SPOKEN LANGUAGE UNDERSTANDING

Candidate:	Su Zhu
Student ID:	x
Supervisor:	Prof. Kai Yu
Academic Degree Applied for:	Doctor of Engineering
Speciality:	Computer Science and Technology
Affiliation:	Department of Computer Science and Engineering
Date of Defence:	September 02, 2020
Degree-Conferring-Institution:	Shanghai Jiao Tong University

大规模任务型口语理解

摘　要

在任务型对话系统中，从用户输入的语句中提取结构化语义形式的语义理解非常重要。受益于近年来深度学习技术在自然语言处理领域的成功应用，语义理解也得到了迅速发展。但是在真实对话系统的部署与开发中，还有如下几方面没有得到充分的研究：针对口语的语义理解存在输入不确定性的问题，以及大规模任务中的数据稀疏问题、跨领域迁移问题。在本论文中，我们围绕上述三个核心问题，基于数据驱动与知识嵌入相结合的基本思想开展大规模任务型口语理解的相关研究工作。

关于口语模式中的输入不确定性问题，我们从不确定性编码和精准口语理解两方面开展研究工作：首先，针对语音识别输出的不确定性编码，本文为非词对齐式的口语理解提出了一种高效的基于“变形器（Transformer）”的词混淆网络（WCN）编码方法。该方法将 WCN 的图结构扁平化，利用“基于时刻的位置编码”和“融合语音识别后验概率的自注意力机制”编码不确定信息。在 DSTC-2 数据集上的实验表明该方法可以取得了当前最高的性能水平。其次，针对语音识别结果上词对齐式语义标注难的问题，本文提出了一种基于强化学习的数据与规则双驱动的精准口语理解框架。我们采用非参数化的槽值纠错规则模型对语义形式中的识别错误进行纠正，并基于纠错后语义形式的正确程度结合策略梯度的强化学习算法来指导优化词对齐的序列标注模型。在目前最大的中文口语理解公开数据集上，该方法的语义理解精度相比以往方法获得了显著的提升。

为了缓解单领域的数据稀疏问题，我们提出了一种基于对偶学习数据扩充的任务型语义理解半监督学习框架。对于无文本对应的语义形式，我们利用语义理解的对偶（反向）模型为其生成流畅的自然语

言句子，并结合对偶学习技术为“主模型”和“对偶模型”提供合作式的闭环学习。该方法的优势在于无文本对应的语义形式可以通过少量的领域本体知识自动获取。该方法的有效性在多个任务型语义理解的标准数据集上得到了验证。

针对跨领域的迁移问题，我们分别从数据和模型层面展开了以下研究：1）在跨领域的数据迁移方面，本文为前面语义理解的对偶模型提出了一种新型的基于原子模板的“语义到文本”生成架构。我们针对细粒度的语义单元（语义槽）设计短语模板（原子模板），将领域相关的知识包装其中；然后把语义形式映射为一个短语集合，并最终构建领域无关的“短语集到句子”的文本重写模型。在 DSTC-2&3 数据集上的实验表明该方法可以在目标领域仅有少量种子数据的情形下显著提升语义理解的性能。2）针对模型层面的领域迁移，本文提出了一种基于标签分布式表征的语义理解模型，通过共享标签信息来深度融合不同领域的数据。我们在标签分布式表征中引入了多种先验知识（原子概念、语义槽描述、语义槽标注样例、少样本支撑集），并提出相应的标签编码网络。在多个中英文数据集上不同程度的领域迁移实验表明该方法相比以往的领域迁移方法有明显提升。

综上所述，针对口语模式特性，本文从不确定性编码和错误恢复的角度提出稳健且精准的口语理解方法；并提出基于对偶学习的数据扩充方法，有效缓解了单领域数据稀疏问题；针对跨领域的迁移问题，本文分别从数据和模型层面加深领域的共享适配。

关键词：任务型对话系统，口语理解，语音识别不确定性，槽值纠错，对偶学习，数据扩充，标签分布式表征，少样本学习，领域迁移

TOWARDS LARGE-SCALE TASK-ORIENTED SPOKEN LANGUAGE UNDERSTANDING

ABSTRACT

Language understanding is a key component of task-oriented dialogue systems, which aims to convert user utterances into structured semantic forms. Recently, inspired by the success of deep learning in natural language processing, researches about Natural Language Understanding (NLU) grow rapidly. However, for the deployment and development of real-world dialogue systems, there are still several problems in lack of in-depth research: *input uncertainty problem* for Spoken Language Understanding (SLU), *data sparsity problem* and *domain adaptation problem* for large-scale domains. In this paper, the following researches about large-scale task-oriented SLU are carried out with the basic idea of *bridging knowledge-driven and data-driven methods*, focusing on the above three core problems.

To tackle the problem of *input uncertainty for spoken language*, the first part of this paper is centered on uncertainty encoding and robust SLU methods. Firstly, for the uncertainty encoding of Automatic Speech Recognition (ASR) outputs, we propose an efficient Transformer-based Word Confusion Network (WCN) encoder, which converts SLU into unaligned tasks. The method considers the graph structure of WCN as a plain form and exploits "time-step based position embeddings" and "ASR post-probability aware self-attention mechanisms" to involve uncertain information of ASR decoding. Experimental results on DSTC-2 (an SLU benchmark) show that our methods can achieve a new state-of-the-art. Secondly, to avoid word-aligned annotations on ASR outputs, we propose a reinforcement learning-based robust SLU framework which is driven by data and rules. We adopt a

non-parametric value error recovery module to recovery ASR errors in predicted slot-value pairs. Meanwhile, the learning of the slot tagging model on ASR outputs is supervised by the recovered slot-value pairs and optimized by the policy gradient-based reinforcement learning. Experiments on CAT-SLU (the largest open dataset of Chinese SLU to the best of our knowledge) show that our methods can obtain significant improvements over previous methods.

To eliminate the problem of *data sparsity problem for a single domain*, we propose a dual learning-based semi-supervised NLU framework from the perspective of data augmentations. For unexpressed semantic forms (i.e., without sentence expressions), a dual model of NLU is proposed to generate the corresponding natural sentences, which is named as Natural Language Generation (NLG) model. The dual learning algorithm is introduced to improve the NLU and NLG models iteratively in a closed-loop of the primal and dual models. Exploiting semantic forms for semi-supervised learning could be more affordable and effective than collecting in-domain sentences, since semantic forms are well-structured and could be automatically synthesized under domain knowledge. Experiments on multiple NLU standard datasets confirm the effectiveness of our methods.

To address the *domain adaptation problem*, we propose two methods at data and model levels: 1) For the domain adaptation of NLU data augmentation models, we propose a novel NLG architecture for the dual model of NLU, which incorporates atomic templates. A simple phrase template (atomic template) is designed for each fine-grained semantic item (i.e., slot-value pair), which is a kind of domain knowledge. With atomic templates, a semantic form (i.e., a set of slot-value pairs) can be mapped to a set of phrases. Afterward, a domain-independent text paraphrase model is built to convert "phrase set to sentence". Experiments on DSTC-2&3 show that this method can significantly improve NLU models with only a little of seed data

(109 samples) in the target domain. 2) For the domain adaptation of NLU models, we propose a label embedding based NLU model, which can utilize data of different domains efficiently by incorporating label relations. Several different types of domain prior-knowledge are involved, and the corresponding encoding models are proposed to extract label embeddings. Experiments of domain adaptation on multiple English and Chinese datasets show that our methods can significantly outperform strong baselines.

In conclusion, we firstly propose a series of robust and accurate SLU methods with the encoder of ASR uncertainty outputs and an error recovery algorithm for slot-value pairs for the input characteristic of spoken language. To mitigate the problems of data sparsity in a single domain, we propose a dual learning-based data augmentation method. For the domain adaptation problem, we develop methods sharing patterns of different domains efficiently at both data and model levels.

KEY WORDS: Task-oriented Dialogue System, Spoken Language Understanding, Uncertainty in Speech Recognition, Error Recovery for Slot Values, Dual Learning, Data Augmentation, Label Embedding, Few-Shot Learning, Domain Adaptation

目 录

插图索引

表格索引

算法索引

主要符号对照表

数学符号

s	普通标量采用普通小写字母
$\mathbf{v}$	列向量采用黑体加粗的小写字母
$\mathbf{M}$	矩阵采用黑体加粗的大写字母
$\mathcal{L}$	损失函数
$p(\cdot)$	概率密度函数
$p(\cdot\|\cdot)$	条件（或者后验）概率密度函数
$\{\cdot\}^\top$	矩阵或向量的转置
$\|\cdot\|$	向量取模（默认为二范数）
$[\cdot]_i$	取向量的第 i 维元素（或者矩阵的第 i 行向量）
$[\cdot]_{i,j}$	取矩阵的第 i 行第 j 列元素
$\boldsymbol{s}$	离散单元（字、词、类别标签等）的序列采用斜体加粗的小写字母
$\boldsymbol{x}$	输入词（字）序列
x_i	输入序列中第 i 个词（字）
y^I	输出意图类别
$\boldsymbol{y}^S$	输出语义槽标记序列
y_i^S	输出序列中第 i 个语义槽标记
$\boldsymbol{y}^C$	输出语义槽值对集合（或者序列）
y_i^C	第 i 个语义槽值对（slot-value pair）
$\|\cdot\|$	序列长度或者词汇表大小
$\oplus$	向量（或者序列）的拼接
$\mathcal{V}_{\text{in}}$	文本输入的词汇表
$\mathcal{V}_{\text{intent}}$	意图类别的词汇表
$\mathcal{V}_{\text{act}}$	对话动作类型类别的词汇表
$\mathcal{V}_{\text{slot}}$	语义槽类别的词汇表
$\mathcal{V}_{\text{tag}}$	带 IOB（In-Out-Begin）的语义槽标记的词汇表
$\text{id}(x_i)$	第 i 个词（字）在词汇表里的序号（取值为 $\{0, 1, \cdots,$ 词汇表大小 $- 1\}$）
$\mathbf{o}(x_i)$	第 i 个词（字）的独热向量（one-hot vector，除第 $\text{id}(x_i)$ 维为 1 其它均为 0）
d_e	词（字）向量维度
d_h	循环神经网络隐层向量维度（单向，双向网络隐层维度为 $2d_h$）

d_x	变形器（Transformer）的隐层向量维度
$\mathcal{R}$	收益（reward）函数
$\mathbb{I}\{\cdot\}$	指示函数，满足给定条件则函数返回 1，否则返回 0

英文缩写

ASR	语音识别（Automatic Speech Recognition）
SLU	口语理解（Spoken Language Understanding）
NLU	语义理解或者自然语言理解（Natural Language Understanding）
NLG	自然语言生成（Natural Language Generation）
CER	字错误率（Character Error Rate）
WER	词错误率（Word Error Rate）
OOV	词汇表以外的未登录词（Out-Of-Vocabulary）
DSTC	对话状态跟踪挑战赛（Dialogue State Tracking Challenge）
CATSLU	中文口语语义理解挑战赛（Chinese Audio-Textual Spoken Language Understanding challenge）
SVM	支持向量机（Support Vector Machine）
CRF	条件随机场（Conditional Random Field）
CNN	卷积神经网络（Convolutional Neural Network）
RNN	循环神经网络（Recurrent Neural Network）
GRU	门控循环单元（Gate Recurrent Unit）
LSTM	长短时记忆网络（Long-Short Term Memory）
BLSTM	双向长短时记忆网络（Bidirectional Long-Short Term Memory）
SC-LSTM	语义可控的 LSTM（Semantically Conditioned LSTM）
WCN	词混淆网络（Word Confusion Network）
BCE	二元交叉熵（Binary Cross Entropy）
STC	语义元组分类器（Semantic Tuple Classifier）
HD	层级式解码器（Hierarchical Decoder）
UA	无监督自适应（Unsupervised Adaptation）
DA	数据扩充（Data Augmentation）
CDT	折叠依赖迁移（Collapsed Dependency Transfer）
NER	命名实体识别（Named Entity Recognition）
AT	原子模板（Atomic Template）
RL	强化学习（Reinforcement Learning）

第一章　绪论

1.1　自然语言理解与任务型口语理解

人机口语对话是语言智能的集中体现，对社会经济进步、科学发展具有极其重要的作用，因此一直以来受到国内外政府机构、学术界和产业界的高度重视。例如，美国国防部高级研究计划署（DARPA）从 90 年代开始相继设立了 SLS（Spoken Language System）(1989-1995)、Communicator (1999-2002) 和 CALO (2003-2008) 项目计划，用于资助口语对话系统的相关技术的研究；欧洲紧随其后，在步入 21 世纪后设立了 TALK (2004-2006)、CLASSiC (2008-2011) 和 PARLANCE (2011-2014) 等口语对话系统项目计划。特别是近年来，随着移动互联网和物联网的快速发展，人机口语对话系统的需求被进一步拉大，从而引发了全世界的产业浪潮。比如，苹果（Apple）公司最早在 iPhone 4S 上推出的风靡全球的个人语音助手 Siri ①，微软（Microsoft）在 2014 年推出的 Cortana 智能助理 ②，亚马逊（Amazon）紧接其后推出的命名为 Alexa 的个人智能助理服务 ③，以及后续百度发布的对话式人工智能系统 DuerOS ④、谷歌（Google）发布的谷歌智能助理 ⑤，等等。如今，人机口语对话系统已经成为人机交互领域的热门研究方向之一，而在其中搭建人类和机器沟通桥梁的语义理解意义重大。

从功能上来看，人机口语对话系统可以大致分为聊天系统、问答系统、任务型对话系统，三种不同对话系统的架构和核心技术各不相同。其中任务型口语对话系统 [1, 2]（系统与用户交互的目标是为了帮助用户完成某种特定任务，比如“查找餐馆”，“订火车票”）在实际应用和研究中尤其被关注。上述的微软 Cortana、亚马逊 Alexa、谷歌智能助理、百度 DuerOS 等都是以任务型对话为主的产品。如图 1–1所示，在一般的任务型口语对话中，包括如下几个模块：语音识别（Automatic Speech Recognition，ASR），将语音转换为文本；口语理解（Spoken Language Understanding，SLU），将识别出的文本解析为语义表示；对话管理（Dialogue Management，DM）根据解析得到的用户语义信息以及对话历史信息给出当前的机器反馈；语言生成（Natural Language Generation，NLG）将机器反馈（语义形式）用文本来重新表达；

① https://www.apple.com/siri
② https://www.microsoft.com/en-us/cortana
③ https://www.alexa.com
④ https://dueros.baidu.com
⑤ https://assistant.google.com

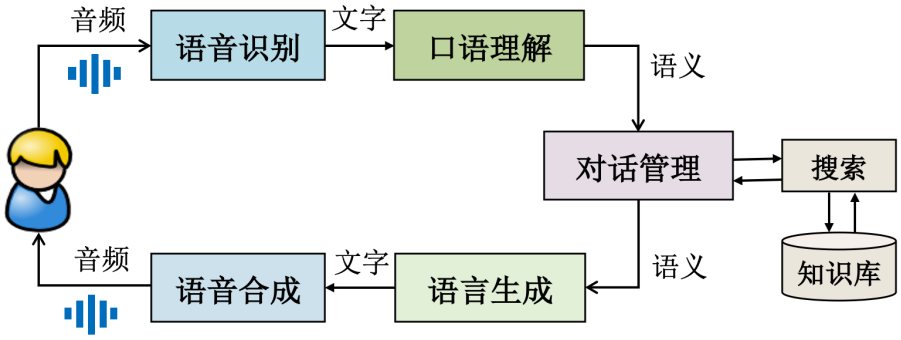


图 1–1 任务型口语对话的经典架构，以及口语（语义）理解模块所处的位置

Figure 1–1 Architecture of task-oriented spoken dialogue systems and the inside spoken language understanding module

语音合成（Text-To-Speech，TTS）最后将文本转换为语音并播放给用户听。

在任务型口语对话系统中，口语理解是服务于语音识别和对话管理之间的中间模块。其旨在理解用户（人）所说的话，以便对话系统后续能够做出适当的回答来帮助用户完成特定任务，口语理解承担着从感知智能到认知智能的桥梁沟通作用 [3-6]。本文我们将针对任务型口语对话中的口语理解进行一系列的研究和探索，并称之为“**任务型口语理解**”（在本文中“任务型”是面向领域、场景的意思）。任务型语义理解的直接目的是从用户说的一句话（口语输入）中解析出相关的语义形式（一种结构化的逻辑表达）。口语输入容易包含 ASR 识别错误，比如用户输入的一句话是“我想买一张今天下午从上海飞往北京的机票”，但 ASR 的识别结果可能是“想买张今天夏我从上海飞完北京的机票”。该例子的目标语义形式为“{场景: 机票购买; 出发时间: 今天下午; 出发城市: 上海; 到达城市: 北京; 机票数量: 一张;}”，其中语义可以用成对的语义槽（比如“出发时间”）和它相应的填充值（比如“今天下午”）进行表示。

以往绝大多数的研究选择忽略口语理解（Spoken Language Understanding，SLU）的口语（语音）特性而单纯地对自然语言文本进行语义理解（Natural Language Understanding，NLU），我们称之为“**任务型语义理解**”（为了和更一般的自然语言理解进行区分）。一般情况，任务型语义理解可以细分为场景分类和语义槽值填充任务。由于槽值填充任务更为复杂、语义信息更具体，本文将集中关注语义槽值填充任务（在后文中，如无特别指明，我们所说的语义理解将特指语义槽值填充任务）。任务型语义理解通常被定义为一个序列标注（sequence labelling）问题，即给定输入的字词序列，预测其相应的输出序列。如图1–2所示，任务型语义理解的输入输出序列是等长的、词对齐的。但如果考虑语音识别错误，非对齐

输出序列	O	O	O	B-number	B-DepartTime	I-DepartTime	O	B-FromCity	O	B-ToCity	O	O
输入句子	我	想	买	*一张*	*今天*	*下午*	从	*上海*	飞往	*北京*	的	机票

图 1–2 任务型语义理解的数据标注样例。其中输入序列中的每个标签均为语义槽的 IOB (In-Out-Begin) 标注格式

Figure 1–2 An example of data annotation for task-oriented language understanding. The slot tags for each word is in IOB (In-Out-Begin) schema.

式的语义标注（如上一段的例子）会更加便捷，但是会增加语义理解建模的难度（相关内容我们将在后文详细展开）。

简单来说，任务型语义理解的技术发展水平大致经历了三个阶段。

第一阶段：基于规则的语义理解技术（20 世纪 70 年代 ~20 世纪 90 年代）。早期的语义理解方法往往基于规则，例如商业对话系统 VoiceXML [7, 8] 和 Phoenix Parser [9]。领域专家根据相应的对话领域，设计与之对应的语言规则来匹配得到输入句子的语义信息。比如，Phoenix Parser 将输入的一个句子（即词序列）映射成由多个语义槽（Slot）构成的语义框架 [10] 里。在基于规则的系统（有时也称为基于知识的系统）[11-14] 中，开发人员会写一些句法以及语义的规则，并用这些规则来解析输入文本以抽取语义信息。这类方法最大的好处是不依赖大量的训练数据。但是其缺点是规则系统很难有足够多的规则将一个领域的各种句式、语义表达覆盖完全，即规则系统的泛化能力一般较差。而且，当系统中规则数量不断增多的时候，规则之间的相互矛盾与冲突也会使得规则系统难以被维护。

第二阶段：基于（浅层）机器学习的语义理解（21 世纪初期）。基于统计机器学习方法的语义理解则很大程度解决了上述规则方法的问题，比如逻辑回归（Logistic Regression, LR）、支持向量机（Support Vector Machine, SVM）、条件随机场（Conditional Random Field, CRF）等。这类方法可以利用句子样例以及它相应的语义标注（语料库）自动学习模型参数。与手工编写规则相比，数据标注需要的特定专业知识要少的很多。而且统计方法可以通过一些半监督、无监督学习等技术向新数据进行自适应。然而这类语义理解方法的一个缺点就是数据稀缺的问题，因为从真实世界中获取大量标注数据是费时费力的。

第三阶段：基于深度学习技术的语义理解（21 世纪 10 年代）。近十年来，深度学习技术在人工智能领域的各个领域都取得了突破性的进展，包括语音处理、图像处理、自然语言处理等领域。任务型语义理解在此阶段也获益良多。深度学习技术主要有如下好处：1) 模型复杂度和潜力的上升带来了语义理解绝对性能的提

升；2) 预训练词向量等技术的引入极大提高了语义理解模型的泛化能力；3) 深度神经网络的结构性也为多模态输入融合、多任务数据融合带来了很多便利。早期词向量、循环神经网络等技术的引入为任务型语义理解研究带来了大发展，近年来带上下文信息的预训练语言模型技术又为相关技术发展注入了新的活力。

1.2 任务型口语理解在真实场景中面临的挑战

任务型语义理解在近年来获得了长足的进步，但是在真实场景下，由于语音识别错误无法完全避免且对话领域扩增中的数据标注代价高昂，我们还面临着几个重要的挑战：*口语模式中的输入不确定性问题*、*单领域的数据稀疏问题*以及*跨领域的迁移问题*。口语模式中的输入不确定性提高了对语音识别输出的编码难度和对齐式标注的难度；在单领域的数据稀疏问题中，以往的方法关注于输入的多样性而缺乏对语义形式多样性的研究；在跨领域的迁移方面，不同领域的标签集不一致会降低数据或者模型迁移的效率。

1.2.1 口语模式中的输入不确定性问题

传统自然语言处理中，对于 “任务型语义理解” 和 “任务型口语理解” 是不做区分的，但在真实情况下，口语模式给语义理解带来了新的挑战。过去绝大多数对于任务型口语理解的研究都存在一个理想的假设，即假设语音识别的准确率是百分之百的，从而仅在准确无误的人工转写文本上进行研究。但事实上语音识别往往无法做到接近完美的识别精度，因为环境噪声、信道质量、方言口音、多说话人现象等影响识别效果的因素是无法完全避免的。因此在真实场景下语音识别的结果是具有不确定性的、带噪的，语音识别的错误也会继续向下游任务传递。而作为语音识别的后端模块，任务型口语理解在实际应用中所面临的第一个挑战就是输入的不确定性问题。带有不确定性信息的语音识别输出形式往往趋于复杂化，并具体在以下两方面影响任务型口语理解：

- **语音识别输出的编码问题**：不同于一般的文字序列，包含丰富不确定信息的语音识别输出形式往往更为复杂，比如简单的最佳候选句子（1-best hypothesis）、N 最佳候选列表（N-best hypotheses list）、词格（word lattices）、词混淆网络（Word Confusion Networks，WCN）。我们需要构建相应的模型对这些语音识别输出形式进行高效且有效的编码，使之能支撑后续的语义理解任务，并从不确定信息中获取用户真实表达的意图。
- **语音识别输出上词对齐式语义标注难的问题**：上文提到任务型语义理解一般采用词对齐的标注形式，但是在语音识别输出上进行词对齐的语义标注

是更加难的一件事情。首先，语音识别错误会使句子晦涩难懂，语音识别输出的形式也比一般词序列更复杂，这些原因使得在识别结果上进行语义标注的成本非常高。其次，口语理解模块是和上游语音识别模块相配合的。如果语音识别模型得到了更新（即语义识别的错误机理发生了改变），这种词对齐的标注也是有可能需要修改的。因此，在语音识别输出上进行词对齐式语义标注似乎是不可能的。如果选择了非词对齐式的标注，语义理解的机器学习范式则会发生变化，从原先的序列标注（sequence labelling）转变为序列解码（sequence decoder）问题。

1.2.2 单领域的数据稀疏问题

近年来，在移动互联网和移动智能设备的驱动下，人们对于任务型口语对话系统的需求剧烈增加。具体表现为任务型口语对话系统当前覆盖的任务或者领域范围在不断扩大和细化[①]，新领域（即没有定向标注数据的单一领域）的出现越来越频繁。因此，如何快速地为新的单个领域构建可用的口语理解系统是一个非常有吸引力的问题。然而，短时间内为新领域收集、标注足够多的数据是非常困难的。在当前数据驱动的机器学习以及深度学习算法应用中，这类单领域的数据稀疏问题仍然十分严峻。

任务型口语理解中常见的数据稀疏问题大致可以分为以下两类：

- **句子表达单一：**在某个领域（比如购买火车票）内，收集的用户句子说法可能会比较单一、不够丰富，即同样的语义没有足够多样性的句子表达。比如，买一张去北京的高铁票可以表达为“帮我买一张去北京的高铁票”、“我要买一张高铁票去北京”等等。
- **语义形式的稀疏：**某个领域采集到的数据可能非常有限，以致于有些语义表达还没有被包含，即部分语义标签或者一些语义标签的组合很少出现或者几乎没有出现在已有的训练数据中。比如在购买火车票领域采集的有限数据中同时提供“出发城市”、“到达城市”以及“车票等级”的样本数可能微乎其微。

1.2.3 跨领域的迁移问题

上述两类数据稀疏问题可以出现在单个领域内，也可以发生在多个领域之间。由于数据采集与标注往往耗时耗力，多领域而小数据的情况（即领域杂多但每个领域能采集的数据极为有限）在真实场景中更为普遍。那么，如何利用已有领域

① 不同领域代表不同的任务，比如订餐、查天气、听歌等。

（源领域）的数据去帮助构建新领域（目标领域）的口语理解系统成为了一个新的挑战（即跨领域的迁移问题）。不同于单领域的数据稀疏问题，跨领域的迁移问题涉及的所有领域的数据总量还是可观的，但是不同领域之间数据分布以及标签集的差异性使得相关问题的解决思路需要改变。

任务型口语理解的跨领域迁移问题主要集中在数据和模型两个层面：

- **数据层面**：不同领域之间的数据分布大多不同，但不是完全没有联系和相关性。如何利用源领域的数据帮助目标领域中标注数据的产生与扩充是上述单领域数据稀疏问题的一个直接延伸。
- **模型层面**：不同领域之间的标签集往往是有很多差异性的，比如购买火车票领域关注出发地点、到达地点、时间、日期等等，预订餐馆领域则关注菜品、口味、人均价格、时间等。在口语理解模型做领域迁移的时候，如何考虑不同领域之间标签集的相关性，并在文本输入的共享上增加标签信息的共享适配，将会是一个非常重要的研究问题。

此外，在模型层面的跨领域迁移中，还涉及两个不同需求的情况：1）领域自适应[①]的思想，即结合大量源领域的数据以及目标领域的少量数据，为每一个目标领域构建一个自有的口语理解模型。2）领域通用模型的需求，即预设目标领域中口语理解系统的构建是即时的，不同的目标领域共享唯一的、通用的口语理解模型。目标领域只需要提供必要的领域知识（包括本体、数据样例等）而不调整模型参数，即可完成口语理解系统的部署。这两类需求都是实际场景中经常会面临的问题。

1.3 论文主要内容与创新点

本论文围绕大规模任务型口语理解中口语模式的输入不确定性、单领域的数据稀疏以及跨领域的迁移三个核心技术挑战开展了一系列研究与探索。本文主要研究内容如下：

1. **为解决口语理解中语音识别输出编码难的问题**，本文首先将任务型语义理解简化为非词对齐式的分类或者生成式任务，针对语音识别结果的多种不确定性形式（最佳候选句子、N 最佳候选列表、词混淆网络），研究采用特定的神经网络模型对相应的口语理解不确定性输入进行结构化编码。本文针对 N 最佳候选列表采用了循环神经网络与自注意力机制融合的方案。针对信息更丰富的、结构更复杂的词混淆网络，本文提出了一种高效的基于 Transformer 的编码网络，将词混淆网络的图结构扁平化，并利用“基于

① 本文中“领域迁移”和“领域自适应”代表相同的意思，在后文中我们将对这两个名词不加区分地混用。

时刻的位置编码”和“融合语音识别后验概率的自注意力机制”将不确定性信息编码到网络中。在口语理解标准数据集 DSTC-2 上的实验表明，我们的方法相比以往基于循环神经网络的方法取得了显著性的提升，并达到了当前最好的性能水平。

2. **为解决语音识别结果上难以进行词对齐式语义标注的问题**，本文研究如何将人工转写文本上的词对齐标注自动迁移到语音识别输出文本上，并将该问题看成是口语理解模型的输入自适应问题。本文提出了一种基于强化学习的数据与规则双驱动的精准口语理解框架。我们采用了非参数化的槽值纠错规则模型，使用词序列以及发音序列的相似度算法对语音识别结果上提取的槽值对就行错误恢复。错误恢复后的槽值对被用来与正确答案对比计算收益（Reward），并结合策略梯度的强化学习算法对以语音识别结果为输入的序列标注模型进行优化。在目前最大的中文口语理解基准数据集上的实验表明，我们在语音识别结果上的语义理解精度获得了显著的提升。
3. **为缓解单领域的数据稀疏问题**，本文从*数据扩充*入手，研究如何为缺乏数据的新领域自动标注更多的丰富数据，并提出了一种基于对偶学习数据扩充的任务型语义理解半监督学习框架。我们利用任务型语义理解的对偶任务（即语义理解的反向任务，“语义到文本”）为无文本对应的结构化语义形式生成自然流畅的文本，并结合对偶学习技术为“主任务模型”和“对偶任务模型”提供合作优化的闭环学习。我们方法的有效性在多个任务型语义理解的标准数据集上得到了验证。
4. **为实现跨领域的迁移中的数据迁移**，本文对数据扩充模型的跨领域迁移进行了研究，为上述对偶任务提出了一种新型的、基于原子模板的“语义到文本”生成模型。我们首先将特定领域内的语义槽解释为一个短语模板（原子模板），再把语义形式（即语义槽值对的集合）映射为一个短语集合，最终在短语集合的基础上生成完整的句子文本。我们把特定的领域知识包装在原子模板中，把数据扩充中领域无关的部分用“短语集到句子”的文本重写模型来描述。在语义理解标准数据集 DSTC-2&3 上的实验表明，我们的方法可以实现数据扩充模型的领域迁移，并在目标领域仅提供少量种子数据的情况下取得非常高的语义理解性能。
5. **为解决跨领域的迁移中的语义理解模型迁移问题**，本文从*领域自适应学习*出发，研究如何利用已有领域（源领域）的数据帮助构建新领域（目标领域）的语义理解模型。我们关注语义空间的稀疏问题以及不同领域的语义标签分布不一致的问题，研究如何将语义标签的先验知识引入到任务型语

义理解建模中，并挖掘不同领域在语义空间中的相互关系，以知识和数据双驱动来缓解数据稀疏问题。我们提出了一种基于标签分布式表征的语义理解模型，解决了不同领域之间标签不一致导致输出层参数无法完全共享的问题。我们在标签分布式表征中引入了多种先验知识（原子概念、语义槽描述、语义槽样例、少样本支撑集），并提出相应的标签分布式表征编码网络。我们在多个中英文数据集上设计了领域迁移的实验，相关实验结果表明该方法在目标领域仅提供少量数据的情形下比以往的领域自适应方法有明显提升。

在上述的研究内容中，本文的核心贡献与创新性总结如下：

1. 以往的任务型口语理解研究大都忽略口语特性（少数考虑口语特性的研究也都是基于传统机器学习的方法），基于深度学习方法的系统性口语理解特定方法的研究是本论文的重要创新。
2. 以往的数据扩充方式大都是开环学习框架下的扩充，即数据扩充与模型优化是两个独立过程，而本论文首次将两个过程耦合在一起，以闭环学习方式动态进行数据扩充和模型优化的联合建模。
3. 过去跨领域迁移的口语理解方法大多没有考虑标签之间的相关性，而本论文在口语理解中首次提出基于分布式标签表征的思想，将蕴含标签相互关系的多种先验知识引入到口语理解模型中，极大提升了模型对数据的学习效率。

1.4 论文组织结构与章节安排

本文共分为八章，剩余章节内容安排如下：首先，第二章介绍任务型口语理解的基本概念以及文献调研；第三章介绍针对语音识别结果多种不确定性形式的稳健口语理解方法；第四章介绍基于槽值纠错算法和梯度策略的强化学习技术的精准口语理解方法；第五章介绍基于对偶学习数据扩充的语义理解半监督学习框架；第六章介绍结合原子模板的跨领域数据迁移方法；第七章介绍领域知识驱动的标签分布式表征方法以及它在语义理解模型领域自适应学习中的应用；第八章总结全文的工作并给出对未来研究的展望。

第二章　任务型口语理解的技术综述

在任务型口语对话系统中，口语理解是服务于语音识别和对话管理之间的中间模块。其旨在理解用户（人）所说的话，以便对话系统后续能够做出适当的回答来帮助用户完成特定任务。但在以往的绝大多数工作中，为了方便自然语言处理技术的研究和应用，口语模式的特性往往被忽略。这种理想情况假设语音识别模块是准确无误的。于是，在忽略口语特性的情形下，我们也将任务型口语理解更具体地称为任务型语义理解。

本章是对任务型口语（语义）理解技术的综述。我们首先在第2.1节介绍任务型语义理解中的一些基本概念，接着在第2.2节介绍任务型语义理解的基本设定，第2.3节介绍任务型语义理解的经典建模方法，第2.4节介绍对语音识别错误稳健的任务型口语理解方法，第2.5节介绍任务型语义理解中新领域构建和领域自适应的相关方法，第2.6节是本章小结。

2.1　基本概念

2.1.1　意图和语义槽

任务型语义理解的直接目的是将用户说的话转换为语义表示的结构化形式（简称“语义形式”）。在本文涉及的任务型语义理解中，这种语义形式主要由两方面组成：意图（*intent*）和语义槽（*semantic slot*）[10]。其中意图代表一句话抽象性、概括性的意思，通常为一个句子类别，较为简单。而语义槽则表示句子中出现的一些属性（或者实体类型）和它相应的值（value）。比如用户说的一句话是“我想买一张今天下午从上海飞往北京的机票”，其意图类别为“机票购买”，语义槽信息为“{出发时间 = 今天下午; 出发城市 = 上海; 到达城市 = 北京; 机票数量 = 一张;}”。其中“出发时间”、“出发城市”、“到达城市”、“机票数量”均为语义槽，“今天下午”、“上海”、“北京”、“一张”依次为上述语义槽相应的值，它们共同组成了一对一对的语义槽值对（*slot-value pair*）。

2.1.2　对话动作类型

为了在口语对话系统中增加“对话行为（dialogue act）”信息，Traum [15] 在1999年发展了对话系统中行为的概念，考虑了对话的轮次信息以及行为来表达对话的意义，其中包括“陈述（inform）、请求确认（confirm）、询问（request）、否

定（deny）”等行为。比如，“请求确认”可以用来表示句子“是明天上午十点出发的吗？”的行为，“询问”可以被用来表示句子“这家饭店的地址是什么？”的行为。这类对话行为类别也被称为对话动作类型（dialogue act type）[16]。对话动作类型搭配语义槽可以表示更加丰富的语义信息，具体示例如表2–1所示。

表 2–1 带对话动作类型的意图和语义槽示例

Table 2–1 Examples of intents and slots with dialogue act types.

输入句子	意图	语义槽值对
从上海到北京的机票	机票查询	{出发城市 = 上海；到达城市 = 北京}
是明天上午十点出发的吗	机票信息确认	{confirm-出发日期 = 明天； confirm-出发时间 = 上午十点}
现在去北京的机票价格是多少	request-机票价格	{到达城市 = 北京}
错了，不是去苏州，我要去宿州	信息纠正	{deny-到达城市 = 苏州； 到达城市 = 宿州}

2.1.3 领域本体

在任务型口语对话系统中，用户与系统在某一特定领域（任务）内开展的有效交互往往是受限于一定范围的事物，而这个范围被领域本体（*ontology*）[17] 所框定。经典领域本体提供了特定领域下涉及的事物（实体与概念）集合以及相关术语的定义清单。据此，任务型语义理解中的领域本体主要由意图、语义槽、语义槽特性、语义槽所包含的取值集合、意图与语义槽的关系以及语义槽之间的相互关系组成。

- 意图和语义槽已经在前文介绍过。但除了意图和语义槽的名称外，本体还可以包含它们的定义描述，比如“出发城市”可以被描述为“航班（飞机）起飞（出发、离开）的城市名称”。
- 语义槽特性则一般包括语义槽的取值类型（比如数字、字符串、布尔类型、时间、日期等）和一些其他特征（比如，是否可以被查询）。
- 语义槽所包含的取值集合则是每个语义槽可能被用户提及的所有取值，比如语义槽“出发城市”和“到达城市”的取值都可以包括“上海”、“北京”等。但实际场景中很多语义槽的取值无法完全枚举，本体中一般只给出少量的取值样例。
- 意图与语义槽的关系一般为共现或者框架关系，即某一意图的出现会涉及某些语义槽（一部分语义槽是必然出现的，而另一部分则是可选的），比如

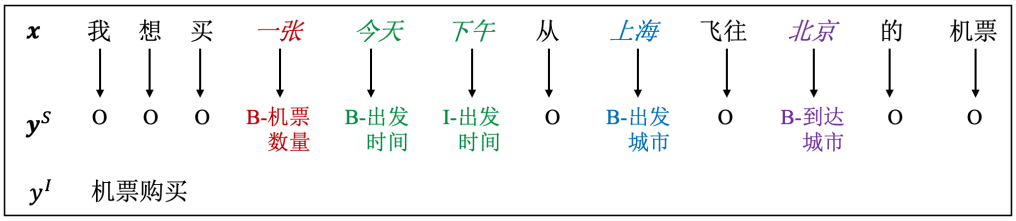


图 2–1 一个任务型语义理解的示例，其中包括意图识别和语义槽填充

Figure 2–1 An example of task-oriented language understanding which involves intent detection and slot filling.

“机票查询” 会涉及 “出发城市” 和 “到达城市”。

- 在一些复杂领域中，还需要定义语义槽之间的相关关系来丰富领域本体知识。这些关系通常包括上下位关系、值域交叠关系、组成成分关系等。

2.2 任务型语义理解基本设定

2.2.1 意图识别与语义槽填充

任务型语义理解一般可以分为两个子任务：意图识别（*intent detection*）和语义槽填充（*slot filling*），如图2–1所示。

意图识别一般被看成句子分类问题，将输入的句子 $\boldsymbol{x} = (x_1, \cdots, x_{|\boldsymbol{x}|})$ 分类到多个意图类别中的一个，$\hat{y}^I \in \mathcal{V}_{\text{intent}}$，其中 $\mathcal{V}_{\text{intent}}$ 是当前领域中所有可能意图的合集。给定 $\boldsymbol{x}$，根据最大化类别后验概率 $p(y^I|\boldsymbol{x})$ 选取 $\hat{y}^I$：

$$\hat{y}^I = \arg\max_{y^I \in \mathcal{V}_{\text{intent}}} p(y^I|\boldsymbol{x}).$$

意图类别一般在句子的表达方面会有很大的自由度。比如用户说 “我想下周从上海飞往北京” 和 “我先买一张下周从杭州到成都的机票” 在意图方面表达的可能是一样的意思。

语义槽填充任务是为了实例化语义槽，即从句子中抽取出相关语义槽的值。在统计学习方法中，语义槽填充任务往往被形式化为一个序列标注任务（即词对齐式的语义标注）。给定词序列 $\boldsymbol{x}$，其目标是根据最大后验概率 $p(\boldsymbol{y}^S|\boldsymbol{x})$ 预测出相应等长的语义标签序列 $\boldsymbol{y}^S = (y_1^S, \cdots, y_{|\boldsymbol{x}|}^S)$，

$$\hat{\boldsymbol{y}}^S = \arg\max_{\boldsymbol{y}^S} p(\boldsymbol{y}^S|\boldsymbol{x}).$$

其中 $y_i^S \in \mathcal{V}_{\text{tag}}$，$\mathcal{V}_{\text{tag}}$ 是当前领域所有可能的语义标签的合集。如图2–1所示，大多数词并不属于某个语义槽的值，于是 “O” 标签被用来表示语义槽之外词。此外，一个语义槽对应的值可以是由连续的多个词组成，为了区分不同的分段，一般会采用 “B、I” 分别表示每一个值的开始词和中间词。这就是序列标注中常见的 IOB (In-Out-Begin) 的标注形式。

除了一般常见的词对齐式的语义标注，前一章也提到非词对齐的语义标注。针对这种情况，相应的机器学习范式也从序列标注变为了序列解码（比如分类或者生成问题）。我们记给定词序列 $\boldsymbol{x}$，以及相应的非对齐标注为 $\boldsymbol{y}^C = (y_1^C, \cdots, y_{|\boldsymbol{y}^C|}^C)$。其中 y_i^C 一般为一个语义槽值对，比如图2–1中的 “出发时间 = 今天下午”。

意图识别任务通常相对简单很多，而在本文中我们将重点关注语义槽填充任务（尤其是基于序列标注的类型）的研究。

2.2.2 常用评测指标

- 句子级的意图识别准确率（intent accuracy）：该指标衡量的是有多少比例句子的意图识别是完全正确的。

$$intent\_Acc = \frac{\#\text{ 意图识别正确的句子}}{\#\text{ 所有句子}} \tag{2–1}$$

- 语义槽预测的精准率 (Precision)、召回率 (Recall)、调和平均值 (F_1 score)：

$$\begin{aligned} \text{Precision} &= \frac{\#\text{ 预测出来的语义槽值对并且与人工标注一致的部分}}{\#\text{ 所有预测出来的语义槽值对}} \\ \text{Recall} &= \frac{\#\text{ 预测出来的语义槽值对并且与人工标注一致的部分}}{\#\text{ 所有人工标注的语义槽值对}} \\ slot\_F_1 &= \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}} \end{aligned} \tag{2–2}$$

  上述指标都是语义槽值对级别的。由上文可知，一句话对应的语义标注是一个语义槽值对的集合，这里的精准率、召回率、调和平均值衡量的是预测集合与标注集合之间的重合程度。
- 语义槽预测的句子级准确率：该指标衡量的是有多少比例句子的语义槽值预测结果与标注完全一致的。

$$slot\_Acc = \frac{\#\text{ 语义槽预测完全正确的句子}}{\#\text{ 所有句子}} \tag{2–3}$$

其中 # 表示数量。

2.3 任务型语义理解的经典模型

在以往的大多数工作中，任务型口语理解的口语（语音）特性常常被忽略，人们大量采用经典的自然语言处理技术对任务型语义理解进行研究。

2.3.1 传统方法回顾

关于口语理解的研究开始于 20 世纪 70 年代美国国防先进研究项目局（Defense Advanced Research Projects Agency, DARPA）的言语理解研究和资源管理任务。在早期，像有限状态机和扩充转移网络等自然语言理解技术被直接应用于口语理解 [18]。直到 20 世纪 90 年代，得益于 DARPA 赞助的航空信息系统（Air Travel Information System, ATIS）评估项目 [19]，口语理解的研究才开始快速发展。许多来自学术界和产业界的研究组织参与其中，并试图理解用户关于航空信息的自然语言询问（其可以包括航班信息、地面换乘信息、机场服务信息等），然后从一个标准数据库中获取答案。在 ATIS 项目的研究过程中，人们开发了很多基于规则和统计学习的口语理解系统。

受自然语言处理领域中理性主义的影响，早期的语义理解方法往往基于规则，例如商业系统 VoiceXML [7] 和 Phoenix Parser [9]。其中 Phoenix Parser 把输入的一句话（词序列）映射到相应的语义结构（由多个语义槽组成）。该方法需要根据对话领域人工设计与之对应的语言规则，以此来解析语音识别产生的文本。但是，这类基于规则的系统（有时也称为基于知识的系统）[11-14] 非常依赖于领域专家设计的文法规则，对于没见过的句子泛化能力很差。

随着自然语言处理领域里经验主义方法的回归，基于统计学习的语义理解方法也陆续出现。基于统计学习的方法可以具有更好的泛化能力，并且可以减少对于领域专家的依赖转而依赖大量标注数据。统计语义理解模型可以进一步分为两类：生成式模型（generative model）和判别式模型（discriminative model）。生成式模型学习的是输入 $\boldsymbol{x}$ 和标注 $\boldsymbol{y}$ 之间的联合概率分布 $p(\boldsymbol{x}, \boldsymbol{y})$，而判别式模型则直接对后验概率 $p(\boldsymbol{y}|\boldsymbol{x})$ 进行建模。

生成式的统计语义理解方法包括组合范畴语法（Combinatory Categorial Grammars, CCG）[20]、基于短语的统计机器翻译（Statistical Machine Translation，SMT）[21]、随机有限状态变换机（Stochastic Finite State Transducers，SFST） [21, 22]、动态贝叶斯网络（Dynamic Bayesian Networks，DBN） [21, 23, 24] 等。

判别式模型则直接评估给定输入句子时语义标注的后验概率。与生成式模型不同，这类模型不需要做特征集之间的独立性假设，因此这类模型可以更随意地引入一些潜在可能有用的特征。研究表明，在语义理解任务中判别式模型一般会显

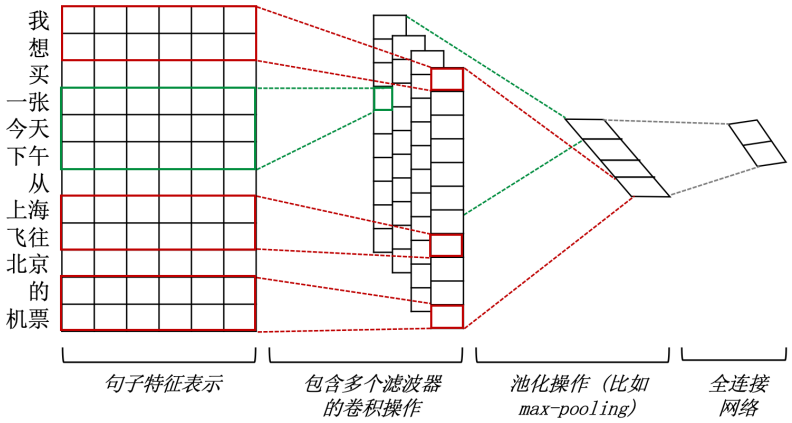


图 2–2 一个用于文本分类的 CNN 模型结构示例

Figure 2–2 An example of CNN architecture for text classification.

著地优于生成式模型 [25]。常见的判别式语义理解方法包括支持向量机（Support Vector Machines，SVM) [26, 27]、最大熵（Maximum Entropy，ME）[26]、最大熵马尔科夫（Maximum Entropy Markov Models，MEMM）[21]、条件随机场（Conditional Random Fields，CRF）[26, 28] 等等。此外，通过使用三角 CRF（Triangular-CRF），条件随机场也可以同时预测一个句子的语义槽和句子类别（比如意图） [29]。

2.3.2 基于深度学习的方法

2.3.2.1 意图识别

如上文所述，意图识别可以被形式化为句子分类问题。通过将变长序列编码为定长的表征向量，卷积神经网络（Convolutional Neural Network, CNN）和循环神经网络（Recurrent Neural Network, RNN）是典型的用于解决句子分类问题的深度学习模型。

图2–2展示了一个用于句子分类的典型 CNN 架构 [30]。一个卷积操作包含了一个滤波器（filter），其作用在输入句子中 h 个词的固定窗口上产生新的特征，即对 h 个词的特征做线性加和。将该固定窗口沿着句子开头滑动到句子结尾处，可以获得不同窗口的新特征。由于不同句子的长度是不定的，为了得到一个固定长度的句子级的特征向量，最大池化（max pooling）操作被应用到不同窗口的新特征上获取最大值，即沿着时间序列取最大的特征值。如果我们有 K 个滤波器（比如图2–2中 K 为 4），则可以取得长度为 K 的特征向量。这些特征被送入到全连接的 softmax 层最终得到在 M 个意图类别上的概率分布。

除了使用 CNN 提取句子的特征表示来进行分类外，也有一些工作采用循环神

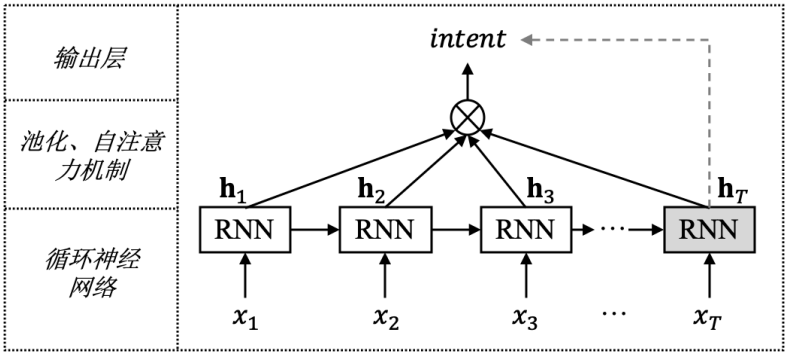


图 2–3 一个用于文本分类的 RNN 模型结构示例

Figure 2–3 An example of RNN architecture for text classification.

经网络（RNN）来编码输入的句子信息。如图2–3所示，这概括了 RNN 句子编码器的多种经典结构。首先，我们可以像 CNN 一样对 RNN 提取出来的隐层向量做池化操作来获取固定维度的句子特征 [31]。其次，自注意力机制可以对所有隐层向量计算相应权重并获取它们的加权和 [32]。此外，还有一种方法则是仅使用 RNN 最后一个时刻的隐层向量作为整体句子的表示 [33]，因为最后一个时刻的隐层向量是由前面所有的输入决定的，理论上具备表示整体句子信息的能力。近年来，单向的 RNN 已经逐渐被上下文建模能力更强的双向 RNN [34] 和 Transformer [35] 所替代。

2.3.2.2 语义槽填充

在基于序列标注的语义槽填充任务上，循环神经网络率先取得进展。Yao 和 Mesnile 将单向循环神经网络（RNN）应用到语义槽填充任务上，且在 ATIS 基准数据集上显著超越 CRF 模型 [36, 37]。图2–4展示了一个用于语义槽填充任务的 RNN 模型结构，其上中下三层分别为输出层、隐层和输入层网络。每一层均由一系列的神经元构成，且层与层之间通过一个权值矩阵相连（比如图中的 **U**，**W**，**V**）。输入层中 x_t 表示输入序列第 t 时刻的词，它在输入层中被转换为一个独热向量（one-hot vector），即 $\mathbf{o}(x_t) \in \mathbb{R}^{|\mathcal{V}_{\text{in}}|}$。其中 $\mathcal{V}_{\text{in}}$ 为输入词表，独热向量中 x_t 对应的的那一维值为 1，其他值全为 0。输出层向量 $\mathbf{z}_t$ 表示模型预测得到的概率分布（在所有语义槽标签上），该输出向量的维数是所有语义槽标签的总数。该模型结

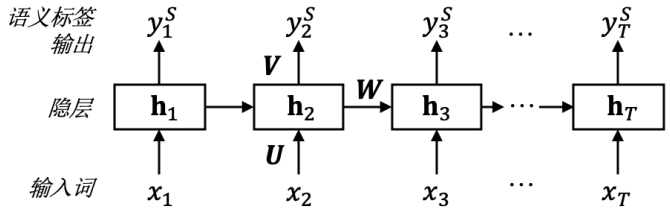


图 2–4 用于口语理解中序列标注任务的循环神经网络模型

Figure 2–4 RNN architecture for slot tagging in spoken language understanding.

构在隐层和输出层上的计算如下：

$$\mathbf{h}_t = \sigma(\mathbf{U}\mathbf{o}(x_t) + \mathbf{W}\mathbf{h}_{t-1}) \tag{2–4}$$

$$\mathbf{r}_t = \mathbf{V}\mathbf{h}_t \tag{2–5}$$

$$\mathbf{z}_t = \text{softmax}(\mathbf{r}_t) \tag{2–6}$$

其中，$\sigma(h) = \frac{1}{1+e^{-h}}$，$z_{tm} = \frac{e^{r_{tm}}}{\sum_k e^{r_{tk}}}$。$\sigma$ 为 sigmoid 激活函数（神经网络的激活函数有很多变种，比如 ReLU、tanh 等），softmax 则将输出向量归一化为在所有类别上的概率分布。公式中 $\mathbf{U}\mathbf{o}(x_t)$ 将独热向量（离散）映射为一个连续的向量，该连续向量的词表征常被称为词嵌入或者词向量（word embedding）。

该模型参数的优化目标是最小化数据样本的负条件对数似然：

$$-\log p(\boldsymbol{y}^S|\boldsymbol{x}) = -\sum_{t=1}^{|\boldsymbol{x}|} \log p(y_t^S|x_1, \cdots, x_t) \tag{2–7}$$

该模型没有考虑语义槽标签在输出层上的相互依赖关系，且第 t 时刻的预测输出只依赖于当前时刻以及之前时刻组成的历史词序列。因此，为了在一定程度上看到 t 时刻之后的词信息，一些方法选择以当前时刻为中心划定一个固定大小的输入窗口（假设窗口大小为 $2k+1$ 个词），于是第 t 时刻的输入就从 x_t 变成 $\boldsymbol{x}_{t-k}^{t+k}$。

但是这种原始的循环神经网络在训练过程在存在一些问题，即梯度消失（gradient vanishing）和梯度爆炸（gradient exploding）问题。长短时记忆单元（Long Short-term Memory, LSTM） [34, 38] 则有效地解决了这两个问题。Yao 等 [39] 首次将结合 LSTM 的循环神经网络应用于任务型语义理解领域，并在 ATIS 数据集上超越了原始 RNN。然而 LSTM 计算比较复杂，后续一些工作提出了一系列更简易的门控单元，比如著名的门控循环单元（Gated Recurrent Units，GRU）[40, 41]。

上述基于单向 RNN 的模型只能考虑当前时刻以及之前的词信息，而忽略了当前时刻往后的词。由于语义理解任务一般是对给完整的一句话预测相关语义信

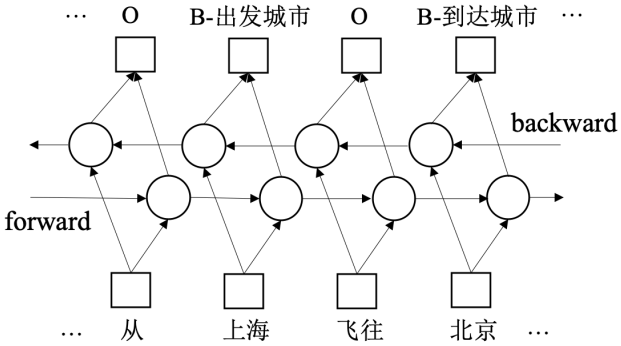


图 2–5 用于口语理解中序列标注任务的双向循环神经网络模型

Figure 2–5 Bidirectional RNN architecture for slot tagging in spoken language understanding.

息，所以理论上可以同时考虑当前时刻之前词和后续词的信息。在这方面，最典型模型结构的就是双向循环神经网络。该模型由两个单向 RNN 组成，其中一个向右正向传播（forward），另外一个向左反向传播（backward），具体结构如图2–5所示。在 ATIS 数据集上，双向循环神经网络模型达到了比单向循环网络更优越的性能水平 [42, 43]。基于双向循环神经网络模型的语义标签序列标注可以表示为如下后验概率公式：

$$p(\boldsymbol{y}^S|\boldsymbol{x}) = \prod_{t=1}^{|\boldsymbol{x}|} p(y_t^S|\boldsymbol{x}) = \prod_{t=1}^{|\boldsymbol{x}|} p(y_t^S|x_1, \cdots, x_{|\boldsymbol{x}|}) \tag{2–8}$$

其中 $\boldsymbol{x}$、$\boldsymbol{y}^S$ 分别表示输入词序列和输出标签序列，y_t^S 表示 t 时刻的语义槽标签。

除循环神经网络外，卷积神经网络（CNN）也被前人应用到语义理解的序列标注任务中 [44, 45]。卷积神经网络可以利用当前词及其周围固定窗口中的上下文提取相关特征用于序列标注。

上述方法对于不同时刻的输出预测是相互独立的，没有考虑输出标签之间的相互依赖关系。传统的条件随机场（CRF）模型则可以对相邻输出标签之间的依赖关系有较好的建模。于是，许多工作将 CRF 模型和深度神经网络（比如 RNN、LSTM、CNN 等）相互结合 [44, 46, 47]。这类方法的核心在于将深度神经网络看成一个特征提取模块，并将这些特征作为 CRF 模型输入；或者，反过来将 CRF 模型的优化准则看作深度神经网络的优化函数）。最后，CRF 模型可以直接采用反向传播算法和深度神经网络一起更新参数。

除了结合传统的 CRF 模型，基于 “序列到序列” 的编码器-解码器（encoder-decoder）模型 [48] 后续也被应用到任务型语义理解中 [49]。这类模型的编码器和解码器都是循环神经网络，encoder 对输入序列进行编码（即特征提取、变换），解

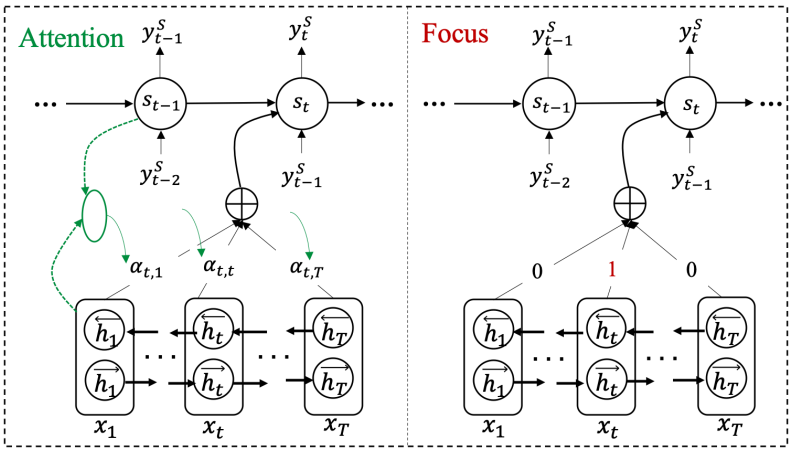


图 2–6 语义槽填充的“编码器-解码器”架构，包括注意力（attention）和聚焦（focus）两种上下文获取机制

Figure 2–6 Encoder-decoder architecture for slot tagging task, including attention and focus mechanisms.

码器则根据编码器的信息对输出序列进行预测。解码器在第 t 时刻的预测会使用第 $t-1$ 时刻的预测结果作为输入，以此考虑输出标签的时序依赖关系。基于此模型的语义槽序列标注可以表示为如下的条件概率公式：

$$p(\boldsymbol{y}^S|\boldsymbol{x}) = \prod_{t=1}^{|\boldsymbol{x}|} p(y_t^S|y_1^S, \cdots, y_{t-1}^S; \boldsymbol{x}) \tag{2–9}$$

其中 $\boldsymbol{x}$、$\boldsymbol{y}^S$ 分别表示输入词序列和输出标签序列，y_t^S 表示第 t 时刻的语义槽标签。受编码器-解码器模型的影响，Kurata 等 [50] 提出了编码器-标注器（encoder-labeler）的模型，其中编码器 RNN 先对输入序列进行逆序编码，然后解码器 RNN 的输入包括当前输入词以及上一时刻的语义标签预测结果。Zhu 和 Liu 等 [43, 51] 分别将基于注意力机制（attention）的编码器-解码器模型应用于任务型语义理解，以及提出了基于“聚焦机”（focus）的编码器-解码器模型，如图2–6所示。在第 t 时刻，attention 模型 [48] 利用解码器 RNN 中 $t-1$ 时刻的隐层向量和编码器 RNN 中所有时刻的隐层向量依次计算相关性权值 $\alpha_{t,i}, i = \{1, \cdots, |\boldsymbol{x}|\}$，再对编码器中的所有隐层向量做加权和，从而得到第 t 时刻的解码器 RNN 的输入。focus 模型则直接利用了序列标注任务中输入与输出序列等长对齐的特性，解码器 RNN 在第 t 时刻的输入就是编码器 RNN 在 t 时刻的隐层向量。他们 [43, 51] 的实验表明 focus 模型的结果明显优于 attention，且同时优于不考虑输出依赖关系的双向循环神经网络模型。

许多循环神经网络的其他变种也在语义理解中进行了相关尝试和应用，比如

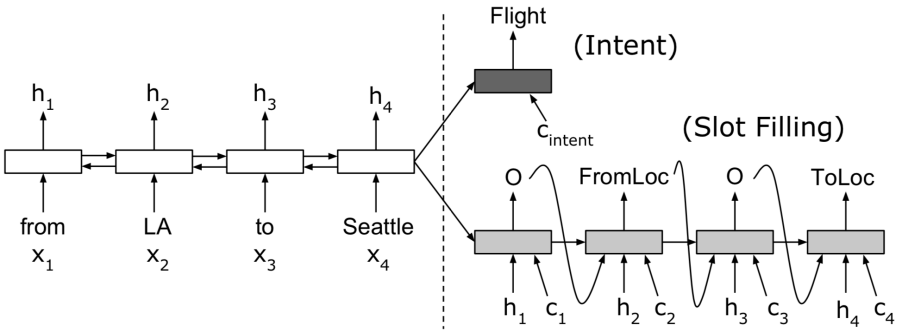


图 2–7 基于 encoder-decoder 结构的意图识别和语义槽填充联合模型（该图引自 Liu 和 Lane[51]）

Figure 2–7 Joint modelling of intent detection and slot filling based on encoder-decoder architecture. （This picture is from Liu and Lane [51]）

通过加入外部记忆单元（External Memory）来提升循环神经网络的记忆能力 [52]，以及结合指针网络将序列标注任务分为 IOB 分块任务和语义槽标注任务 [53] 等。

2.3.2.3 多任务联合学习

意图识别、语义槽填充都是语义理解相关的子任务，比如在意图为 “查询航班” 的句子中往往有 “目的地” 的语义槽出现。所以语义理解中不同子任务联合学习的机制将有助于提升任务型语义理解的性能。但在浅层机器学习时代，我们一般使用 SVM 方法解决句子分类问题，而用 CRF 模型解决序列标注问题。这两类方法完全不同，只能独立优化更新。到了深度学习阶段，由于神经网络模型的结构可塑性、可扩展性，使得不同任务之间的联合学习变得简单方便起来。

Hakkani-Tur 等 [54] 为了联合训练领域分类、意图识别、语义槽填充任务，在输入句子的开头和结尾位置各插入一个额外标记 $< BOS >$ 和 $< EOS >$，并且给这两个特殊的输入词分别标注上领域和意图标签，从而将多个任务共同转化为一个联合的序列标注问题。Zhang 和 Wang [55] 则共享 BLSTM 编码器，利用最大池化操作为意图识别提取句子的整体特征，并将语义槽序列标注和意图分类的损失函数加权联合，同步训练两个任务。Kim 等 [56] 也利用类似的思想对领域、意图、语义槽进行联合建模。Liu 和 Lane [51] 基于 encoder-decoder 结构提出了联合训练的语义槽填充和意图识别模型。如图2–7所示，其本质是利用不同的 decoder 建模不同的任务，而共享同一个句子 encoder。除了循环网络的结构，以自注意力机制为核心的 Transformer 也被应用了到意图识别和语义槽填充的联合建模中 [57]。

上述多任务联合学习的模型大多是共享了部分句子编码网络，从而隐式地相互促进其他任务。Zhang 等 [58] 则应用胶囊网络对意图识别和语义槽填充进行联

合建模。Qin 等 [59] 在词级别考虑语义槽对意图的依赖。一些工作则显式地利用意图信息构建门控单元对语义槽的预测进行显式的影响 [60-62]。

2.3.2.4 基于对话上下文的语义理解

在人机对话框架下，语义理解往往是对上下文敏感的[①]，就是在不同的对话情境中，同样一句话会有不同的语义。比如下面这两个例子：

- 用户轮次 1: “我要买一张去北京的高铁票”
- 用户轮次 2：“后天下午 *3* 点”
- 用户轮次 1：“请帮我设置一个闹钟提醒”
- 用户轮次 2：“后天下午 *3* 点”

从上述例子可以看出，两次“后天下午 *3* 点”的意思是不同的，前一个是指高铁车次的出发时间，后一个则是设置闹钟提醒的时间。在很多时候，孤立的一句话是会造成歧义的，而引入对话上下文则可以在很大程度上缓解这些语义歧义现象。

在人机口语对话系统中主要存在两类对话上下文信息，其中一类是用户之前说过的话（比如上面的两个例子），另外一类则是系统之前回复的话（一般以语义形式存储）。引入这两种上下文信息都可以对任务型语义理解提供帮助。

Henderson 等 [64] 利用系统最近的回复（语义形式）作为语义理解模型的额外特征，以此来提升语义理解的性能。该方法第一步获取系统内部最近的回复信息的语义形式，一般由意图类别、对话动作类型、语义槽值对组成。然后，将各种语义项的出现与否当做一种指示特征，与文本特征一起帮助语义理解模型的训练与测试。该工作的实验结果表明系统端的上下文信息对于任务型语义理解的帮助非常显著。

采用循环神经网络的思想，Liu 等 [65] 利用一个循环网络层记录语义理解模型的历史信息。除了用于编码上下文信息的结构外，剩下的模型是基于 CNN 和 CRF 相结合的语义理解模型 [44]。在该方法包含两种不同历史信息的引入方式：一种是将模型中间的计算结果循环连接到下一时刻的输入层和隐层（在实际训练中需要展开）；另一种则是把上一轮次的语义解析结果当做当前轮的额外特征。随着深度神经网络的发展，层级式 RNN 也被用来编码任务型语义理解的对话上下文信息 [66]。

Chen 等 [67] 提出利用记忆网络（Memory Network）来将历史上下文编码为一种表征向量，再利用该向量作为当前句子语义理解的额外特征。在多轮对话任务

① 需要指明一下本文讨论的是如何利用对话上下文对当前句子的语义理解进行消歧，而同样基于对话上下文的对话状态跟踪 [63] 则是对从对话开始到当前句子为止的语义信息的累积。

上，该方法相对于不做上下文建模的方法有了很大提升。该框架包含三个 RNN 模型，分别是对历史句子进行编码的 RNN_{mem}、对当前句子进行编码的 RNN_{in}、以及对当前句子进行语义理解（序列标注任务）的 RNN_{tagger}。该方法利用注意力机制计算当前句子编码和历史句子编码之间的权重，得到历史句子编码的加权和（即上下文信息表征向量），再将该上下文向量和当前句子共同输入序列标注模型得到融合了当前句子和历史句子信息的知识编码向量，最后通过输出层分类得到语义标签。

2.4 稳健口语理解方法

在上一节介绍中的绝大多数工作都是在人工转写文本或者自然语言文本上进行的，它们有一个共同前提，即假设语音识别系统是完美无误的。但是这类方法也忽略了语音识别错误向后传递的问题，同时也缺乏对语音识别结果不确定性编码的研究。本小节将介绍对语音识别错误具有鲁棒性的稳健口语理解技术。

如第一章提到的那样，语音识别的错误难以避免，且其错误机理也非常复杂，这就使得口语理解的输入具有不确定性 [2]。一些传统的观点认为，通过提升语音识别精度从而降低不确定性是实现稳健口语理解的唯一方法。然而，如果从认知技术的角度出发，自然语言本身就包含了一定的模糊性。另外，认知科学的观点认为，有时候使用模糊的表达方法可以减轻多余的认知负担，并有助于提高交互活动的自然度和高效性。允许不确定输入将大大提高信息的输入带宽，进而极大地提升人机交互的高效性和自然性。因此，如何在不确定条件下实现有效的口语理解是认知技术的一个重要研究范畴。

2.4.1 语音识别输出的不确定性形式与编码

口语理解输入（即语音识别输出）的不确定性具体表现为多重编码及置信度，相关的不确定性形式有很多，比如最佳候选句子（1-best hypothesis）、N 最佳候选列表（N-best hypotheses list，如图2–8所示）、词格（word lattice）和词混淆网络（word confusion network）（如图2–9所示）。

口语理解模型需要以这些不确定性形式作为输入进行建模，以此来提升语义理解模型对于语音识别错误的鲁棒能力。Henderson 等 [64] 提出过一种简单的方法，具体可以分为后续两个步骤。1）训练阶段，利用用户输入语音的人工转写文本（即语音识别的标注文本）或者语音识别输出的最佳候选句子（即预测结果中置信度最高的词序列）以及相应的语义标注训练语义理解模型。2）测试阶段，使用语音识别输出的 N 最佳候选列表，将 N 个候选句子依次进行语义理解预测，最

排序	（N最佳候选列表） 候选句子	声学模型 对数概率	语言模型 对数概率
1	it's an area that's naturally sort of mysterious	-7193.53	-20.25
2	that's an area that's naturally sort of mysterious	-7192.28	-21.11
3	it's an area that's not really sort of mysterious	-7221.68	-18.91
4	that scenario that's naturally sort of mysterious	-7189.19	-22.08
5	there's an area that's naturally sort of mysterious	-7198.35	-21.34
6	that's an area that's not really sort of mysterious	-7220.44	-19.77
7	the scenario that's naturally sort of mysterious	-7205.42	-21.50
8	so it's an area that's naturally sort of mysterious	-7198.35	-21.34
9	that scenario that's not really sort of mysterious	-7217.34	-20.70
10	there's an area that's not really sort of mysterious	-7226.51	-20.01

图 2–8 一个 N 最佳候选列表的示例，其中排序是综合了这两项对数后验概率的结果（该例子引用自 Jurafsky [68]）

Figure 2–8 An example for N-best hypotheses list which is ranked according to log probabilities of acoustic and language models. (This example is from Jurafsky [68])

后联合考虑语音识别的后验概率和语义解析的后验概率将语义理解结果进行加权合并。

早期口语理解中对于输入的不确定性建模都是基于语音识别结果中的最佳候选句子 [24, 27]，但是其它编码形式（N 最佳候选列表、词格和词混淆网络等）包含的语义识别解码信息更多，且包含正确识别结果的可能性也更高。此后，相继有工作利用 N 最佳候选列表 [64, 69, 70]、词格 [71-74]、词混淆网络 [64, 75-81] 直接提取更丰富的特征，用于后续的语义理解模型。

Tur 等 [76] 将词混淆网络（WCN）看做一个分段（bin）序列，其中每个分段则包含了音频中相邻两个时刻之间对应的所有候选词及其 ASR 后验概率，这样一来，一些传统序列标注方法（例如 CRF）就可以应用在分段序列上了。首先，该方法将输出序列上的语义标签与 WCN 分段序列进行对齐，如图2–10所示。然后，该方法对相邻的分段提取 N 元组（n-gram）特征，并基于 CRF 进行序列标注建模。

结合深度学习方法，对词混淆网络的分段序列进行直接编码近年来成为主流 [77, 79-81]。其中 Yang 等 [77] 根据分段里的候选词和它们相应的语音识别后验概率对分段进行无监督聚类，再将词混淆网络看着是 "分段类别" 的序列进行语义槽序列标注。而 Masumura 等 [81] 则直接基于词向量，并利用自注意力机制以及分段中每个词的语音识别后验概率获取每个分段的表征向量，进而再结合上层的句子级循环神经网络结构进行口语理解任务。虽然这类方法可以在词混淆网络上预测语义槽的标签序列，但仍然无法准确获取语义槽的值（因为一个分段内存在多个候选词）。

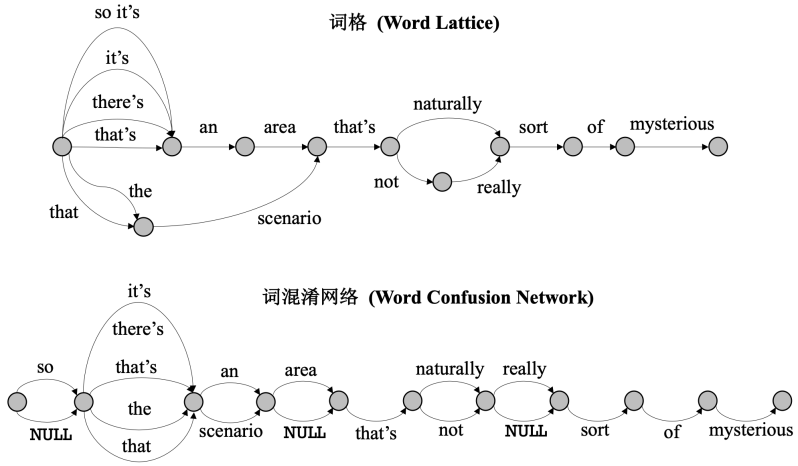


图 2–9 词格、词混淆网络是常见的总结了各种候选句子及其后验概率的结构，该示例中每条边上的置信度并没有给出（这些例子引用自 Jurafsky [68]）

Figure 2–9 Word lattice and word confusion network are typical structure to store hypotheses and posterior probabilities, while the confidence for each edge is not shown. (These examples are from Jurafsky [68])

词混淆分段 (Word Confusion Bins)				
a	t	with	ashton	kutcher
tv	series	wet	aston	
the	tv		astion	
↓	↓	↓	↓	↓
B-类型	I-类型	O	B-明星	I-明星

图 2–10 口语理解中基于词混淆网络分段的序列标注建模（该图引自 Tur [76]）

Figure 2–10 Sequence labelling on word confusion bins for spoken language understanding. (This picture is from Tur [76])

2.4.2 语言模型自适应

除了直接在更丰富的 ASR 结果上建模口语理解任务，也有一些工作专注于提升口语理解模型对 ASR 错误的自适应能力。Liu 等 [82] 和 Zhang 等 [83] 倾向于对 ASR 中的语言模型和语义理解任务进行联合建模，以提升语义理解模型对 ASR 结果的适应能力，同时联合优化后的语言模型可以提升 ASR 在当前领域的识别精度。Schumann 等 [84] 则基于 encoder-decoder 构建语音识别句子的错误恢复模型来提升口语理解对 ASR 错误的鲁棒性。Zhu 等 [85] 则利用基于共享-私有网络的迁移学习框架提升口语理解的 ASR 错误鲁棒性。

2.4.3 端到端口语理解

随着深度学习中的端到端建模技术的发展，近期一些人也开始尝试联合构建语音识别与语义理解的工作，即直接从输入语音到输出语义信息，进而消除语音识别和语义理解模块之间的错误传递问题 [86-96]。比如，Serdyuk 等 [87] 直接构建从语音特征到领域、意图分类的模型。Ghannay 等 [88] 通过将语音识别的输出序列修改为带语义标签的序列，从而将成熟的端到端语音识别技术应用过来。Haghani 等 [89] 则在输出模块上尝试了语音识别与语义理解联合建模的多种方式，比如：1）语音识别文本序列与语义槽值对组合得到的混合输出序列；2）语音识别和语义槽填充的多任务学习等。这类端到端方法最大的缺点是它需要非常多的带语义标签的音频数据 [92]，因此一些人尝试使用语音合成的方法为大量文本数据合成音频 [93]，但是合成的音频与真实音频之间还有明显不匹配的问题。

2.5 针对数据稀疏和领域迁移的语义理解方法

针对基于统计学习（包括深度学习）的语义理解方法，如果我们想要在某个对话领域内达到比较好的语义解析效果，足量且准确的标注数据是必不可少的。然而实际中获取真实数据费时费力，数据的语义标注成本也很高。此外，为了实现非限定领域的语义理解，需要研究语义的进化，即语义在不同领域的扩展和迁移。从语义进化的角度看，在传统技术框架下，如果想要扩展语义理解的领域，往往需要重新定义领域、收集数据、标注数据和构建系统。于是充分利用已有的资源进行领域自适应的语义理解研究变得尤为重要且具有很高的实用价值。

在数据稀疏的情况下，我们的任务目标是利用少量的语义标注数据单独构建一个新领域的口语理解模型。为了缓解数据稀疏问题，过去很多的工作主要围绕这些方面展开：预训练语言表示模型的引入、半监督和无监督学习、自动或者半自动地扩充标注数据、零样本与少样本学习、领域自适应学习等。为了帮助新领域的构建，这些工作的区别主要在于利用了不同的已有资源，分类说明如表2–2所示。

2.5.1 预训练语言表示模型的引入

在上一章中我们提到有一类数据稀疏问题是句子表达多样性的缺失，即相同语义的不同句子说法不够多。句子表达多样性通常可以由替换近义词、插入修饰词或者调整句子成分顺序来达到。预训练词向量的方法可以大大缓解这一类的数据稀疏问题，比如最常用的静态词向量 Word2Vec [97, 98]、Glove [99]，以及基于词形的词向量 [100]。

表 2–2 以往解决数据稀疏的语义理解方法分类

Table 2–2 Different NLU methods for the data sparsity problem.

方法	已有资源
预训练语言表示模型的引入	大量通用未标注文本
半监督学习	大量同领域未标注文本
无监督学习	通用语义知识库
数据扩充	领域知识（本体）
零样本学习	领域知识（本体）
少样本学习	其它领域的标注文本
领域自适应	其它领域的标注文本

但是静态词向量无法根据当前输入句子进行适应性的调整，难以解决相同词在不同上下文中的不同义现象。于是近年来，以 ELMo [101] 和 BERT [102] 为代表的大规模预训练语言表示模型为自然文本提供了更为准确的词向量表示，极大地提升了模型的泛化能力。其中，ELMo 基于 LSTM 网络训练双向的语言模型，即一个正向 LSTM 语言模型和一个反向 LSTM 语言模型；BERT 则是基于 Transformer 网络，结合掩蔽语言建模和上下句关系预测两个任务训练的语言模型。与以往利用静态词向量 [97-99] 的方法相比，结合 ELMo 和 BERT 等预训练语言模型的口语理解方法在性能上有明显的提升 [59, 103-105]。

2.5.2 半监督与无监督学习

另外，基于半监督学习甚至无监督学习的语义理解也是一种数据稀疏问题的解决方案。Tur 等 [106] 利用已有模型在未标注数据上产生的“虚假标签（伪标签）”进行半监督训练。Lan 等 [107] 将少量有标签数据的语义理解任务和大量无标签数据的语言模型任务进行联合训练，并使用共享-私有网络和对抗学习的策略迫使部分共享的参数趋向于任务无关化；类似的方法在其他序列标注任务上也得到了应用 [108, 109]。除了普通语言模型任务外，句子到句子的重建任务也可以和语义理解任务一起联合建模 [110, 111]。Aditya 等 [103] 用海量无标签文本训练的语言模型作为句子的特征提取器，从而减少对语义理解数据的需求。Chen 等 [112-114] 借助外部开放语义资源（FrameNet）以及知识库（FreeBase）进行了无监督的语义理解研究，但该方法要求外部开放语义资源具有完备的领域定义，所欲该方法的便捷性不高。Heck 等 [115] 利用从网页中提取的语义知识图谱对语义项构建自然文本描述，从而生成语义理解的训练数据。

近年来，基于对偶学习的无监督训练方法在神经机器翻译领域取得了成功 [116]，它为自然语言处理领域其他任务的半监督和无监督学习提供了一种新的工具。对偶学习算法在多个自然语言处理领域取得了不错的应用，比如用于知识库问答的语义解析 [117, 118]，自然语言理解 [119, 120]，开放域信息抽取 [121] 等等。

2.5.3 数据扩充

Zhu 等 [122] 提出利用源领域的数据样本模板和目标领域的本体半自动地生成目标领域的数据，其中获取源领域数据样本模板的过程是加入了人工规则的。该语义理解的数据生成方法包括五部分：

1. 源领域的数据样本模板提取（此处需要人工干预，是一个半自动过程）；
2. 目标领域的样本模板生成，即利用源领域模板和目标领域的本体通过重组合的方式合成适用于目标领域的模板；
3. 生成目标领域的文本数据，即利用目标领域的模板生成数据；
4. 语音识别错误模拟，即为生成的文本数据添加语音识别不确定性（即为输入文本添加噪音）；
5. 目标领域的语义理解模型训练，最终得到目标领域的语义解析器。

受近几年来序列生成模型发展的影响，一些人也尝试采用序列到序列（sequence-to-sequence）模型对语义理解的样本进行建模，并采用自监督学习的方式（即输入输出序列是一样的）构建语义理解样本的生成模型 [123-125]。但由于训练数据规模的限制，这类方法生成的数据基本都是对训练集样本的 “某种意义的重组”，这种优势很容易被前面的预训练语言表示模型技术打破。然而，一些传统的数据扩充方法往往更为有效，比如：随机增强（即随机增删词）、句子的反向翻译、基于词频和逆文档频率（TF-IDF）的词替换等 [126]。因为这些做法大多引入了额外的知识，可以对语义理解的训练集有一定的补充。但是，目前的这些工作仍然无法为训练集中没见过的语义标签生成数据。

2.5.4 零样本与少样本学习

除了传统的数据扩充方法，一些迁移学习方法（比如零样本学习策略，zero-shot learning）也被应用到任务型语义理解中来 [127-129]。其中 Ferreira 等 [128, 129] 利用预训练的词向量获取输入句子以及输出语义项（但预训练的词向量空间不一定与当前领域的语义空间保持一致）的向量表征；Yazdani 等 [127] 提出利用两个不同的编码网络分别对输入句子和输出语义项进行特征提取，进而通过对句

子特征和语义项特征之间的相似度（匹配度）选择最为匹配的语义输出（类似于K近邻法）。这类方法可以提升模型的泛化能力，减少对数据量的需求，甚至可以将模型自适应到在训练集中没出现过的语义标签上。

上述方法中语义槽的值域空间是有限且已知的，那么要在基于序列标注的语义槽填充中应用类似的迁移学习技术，则需要对语义槽本身进行更细粒度的编码。Zhu 等 [130] 引入原子概念（atomic concept）作为语义槽的细粒度表示，可以在语义槽编码网络中区分不同语义槽之间的同异关系，进而实现基于原子概念的语义槽自适应。可以想象，当有一个新的语义槽出现时，组成它的原子概念分别在已知的语义槽中依次出现过，则当前模型对于这个新的语义槽具有一定合理的预测能力。此外，一些工作也直接对语义标签的自然语言描述（比如语义槽名字或者领域专家附加的语义槽说明文本）进行编码，利用不同语义类别在描述上的相互关系进行迁移学习 [130-134]。在意图识别中，Chen 等 [135] 利用意图名字对应的词序列作为意图类别编码器的输入。在语义槽填充任务中，一些人将每个语义槽描述和输入句子拼接后共同编码，模型对输入的语义槽预测它相应的 IOB（In-Out-Begin）序列 [131, 132]。这种做法与机器阅读理解 [136] 非常相似，其在命名实体识别任务上也有相同的模型应用 [137]。Shah 等 [133] 在语义槽描述的基础上增加了语义槽的样例取值，为语义理解模型注入了更多的领域知识。但这类做法比较低效，因为相关语义理解模型需要对所有语义槽单独运行一遍模型运算，使得无论是训练还是测试的时间都和语义槽数量成正相关。于是，仅在模型输出层添加语义槽描述的编码信息会更为高效 [130, 134]。

零样本学习旨在将有限的训练集上训练的模型自适应到新的语义标签上，但这类方法的绝对性能往往不高。于是有些工作开始探索为没出现在训练集上的语义标签提供极少量的样本（支撑集），同时可以收获极大的性能提升。这一般被称为“少样本学习”（few-shot learning） [138, 139]。少样本学习方法最早在图像分类领域得到应用和发展。此类少样本学习方法，旨在学习一个输入编码网络，为输入数据和支撑集数据提取相应的特征表示，再利用基于特征表示的匹配计算得到输入数据和支撑集中某个标签之间的相关度。同样，在自然语言处理领域里关于少样本学习的研究一开始主要集中于文本分类问题 [140-143]。在序列标注问题上，Fritzler 等 [144] 利用原型网络（prototypical network） [145] 探索解决少样本的命名实体识别问题；Hou 等 [146] 设计了一种针对序列标注问题（语义槽填充、命名实体识别）的少样本数据构建方法，且提出一种抽象标签依赖机制，将“任务自适应映射法” [147] 应用在了相关任务上。

2.5.5 领域自适应

前文提到口语对话领域的数据非常难获取，那么如何利用不同领域的少量数据相互帮助进而提升目标领域的语义理解性能的问题（即领域自适应）变得非常有价值。一种常见的领域自适应方式是对不同领域的数据进行多任务学习，即共享不同领域数据的特征学习层。Jaech 等 [148] 在基于双向循环网络的语义理解模型上，利用多任务的框架对不同领域的数据进行共享学习。该方法共享双向循环网络的输入层和隐层结构（特征学习相关），而每个领域有一个自己的输出层（任务相关）。实验结果表明多任务学习的框架可以通过共享特征学习来节省不同领域的训练数据量。

但不同领域之间的特征学习真的是完全可以共享的吗？比如两个很不相关的领域，一个是“音乐播放”，一个是“地点导航”，完全共享是否会对各自领域的语义理解反而有坏处。这是一个值得研究的问题。kim 等 [149] 就在语义理解任务上对多领域数据的多任务学习框架进行了改进，采用了不同领域之间既有私有参数也有共享参数的方式。如图2–11所示，每一个领域 d 的语义理解模型都分为两部分，左边从 $x_t, D(x_t)$ 到 h_t 再到 z_t、y_t 的是领域 d 私有的模型结构，而右边 x_t 到 h_t^g 是领域共享的模型，其中 x_t 是领域共享的词向量，h_t 表示领域私有的循环神经网络的隐层向量，h_t^g 则表示领域共有的循环神经网络的隐层向量。由此可见，该模型可以将不同领域之间共享的特征学习模式和领域特有的特征学习模式区分开，从而进行更好的建模。基于类似思想，Liu 等 [150] 利用对抗学习让共享参数偏向于提取领域无关的特征。

2.6 本章小结

本章围绕输入不确定性和数据稀疏两个问题对任务型口语理解进行了技术调研。首先介绍了任务型语义理解的基本概念和基本的任务设定，包括意图识别和语义槽填充；然后介绍了语义理解的经典方法，包括基于规则的方法、基于传统机器学习的方法、基于深度学习的方法、多任务联合学习以及基于对话上下文的语义理解技术；接着介绍了针对语音识别输出不确定性的稳健口语理解方法；最后介绍了针对数据稀疏和领域迁移的语义理解方法，包括预训练语言表示模型、半监督与无监督学习、数据扩充、零样本与少样本学习、领域自适应的应用。

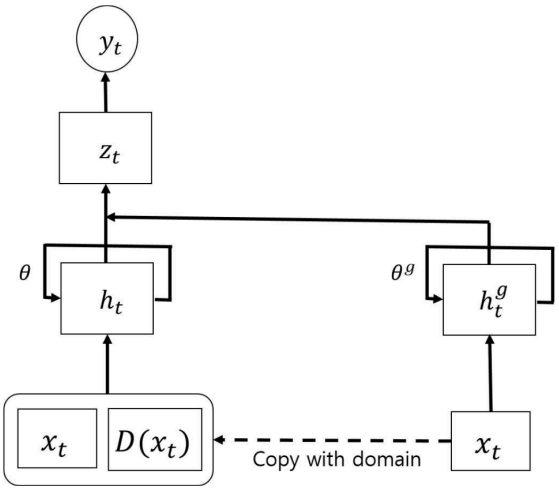


图 2–11 一种共享部分参数的多领域-多任务口语理解模型（该图引自 Kim 等 [149]）

Figure 2–11 Spoken language understanding for multi-domains and multi-tasks with parameters partially shared. (This picture is from Kim et al. [149])

第三章　针对语音识别不确定性的稳健口语理解

3.1　引言

由于环境噪声、信道质量、方言口音、多说话人现象等因素的影响，语音识别往往无法做到百分百正确，比如目前在带噪音、单说话人的公开测试集 CHiME-4 和 Aurora-4 上最好的方法仍有 7% 左右的词错误率 [151, 152]，而在有其他说话人干扰的情况下词错误率普遍高于 10% [153, 154]；甚至在会议等情况下更差，真实世界的非配合词错误率会徘徊在 20%-40% 之间。如果口语理解直接基于语音识别输出的最佳候选结果进行语义解析，会很容易受潜在语音识别错误的干扰而造成理解错误。因此，为了构建更为稳健的口语理解，我们迫切需要从语音识别结果的不确定性形式中抽取出准确的用户意图。如前文第1.2.1小节所述，语音识别输出结果的不确定性给任务型口语理解带来了两个难题：*语音识别输出编码难*与*语音识别结果上词对齐式语义标注难*的问题。

为了针对性研究解决上述第一个难题，本章节先将词对齐式的任务型口语理解等价转换为非词对齐式的分类或者生成式问题（即先忽略第二个难题），据此提出了一种高效的基于 “变形器（Transformer）” 的词混淆网络编码结构。该方法将词混淆网络扁平化，并利用 “基于时刻的位置编码” 和 “融合语音识别后验概率的自注意力机制” 将包含图结构和后验概率的不确定性信息编码到 Transformer 的隐层向量中。同时，Transformer 完全基于自注意力机制的模型结构也会让编码运算非常的高效。

本章的结构如下：我们首先在第3.2节介绍语音识别的多种输出形式（即任务型口语理解的输入）；然后在第3.3节介绍基于层级解码的非对齐式口语理解模型；在第3.4节介绍基于词混淆网络与对话历史信息的口语理解架构，包括基于 Transformer 的词混淆网络编码网络；第3.5节是实验部分，包括实验设置、主要结果与实验分析；第3.6节是本章小结。

3.2　语音识别的输出及其不确定性

如上一章所述，语音识别的输出形式可以有很多种，除了最简单的最佳候选句子（1-best hypothesis），还有 N 最佳候选列表（N-best hypotheses list）、词格（word lattice）、词混淆网络（Word Confusion Network, WCN）等。相比最佳候选句子，N 最佳候选列表、词格、词混淆网络都能包含更多的语音识别解码信息和更

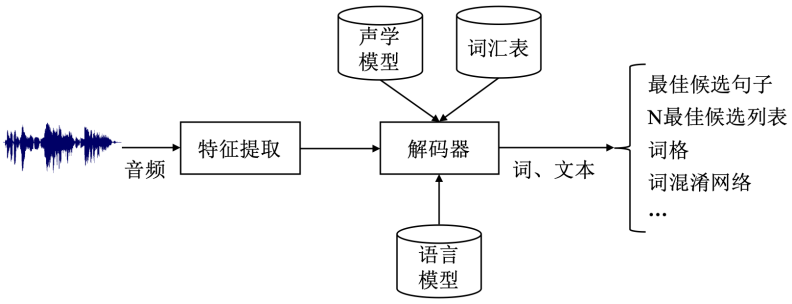


图 3–1 语音识别解码过程的示意图

Figure 3–1 A diagram of ASR decoding.

多的不确定性（即更有可能包含完全正确的识别句子）。如图3–1所示，语音识别解码涉及声学模型和语言模型两部分的评估。

最佳候选句子是语音识别解码得到的后验概率最高的输出序列，比如正确文本为“买一张去苏州的火车票”可能会被识别为“买一张去宿州的火车票”。最佳候选句子可以被形式化地表示为一个词序列，即 $\boldsymbol{x} = (x_1, x_2, \cdots, x_{|\boldsymbol{x}|})$，其中 $|\boldsymbol{x}|$ 表示序列长度（即词的个数）。

N 最佳候选列表则是包含了语音识别解码得到的按后验概率从大到小排列的前 N 个输出序列，比如表3–1中就提供了一个 N 最佳候选列表的示例（N 等于 10）。每一个候选句子的后验概率是同时考虑了声学模型以及语言模型置信度后的综合得分。同样，N 最佳候选列表可以被形式化地表示为一组词序列以及相应的语音识别后验概率，即 $\text{N-best}(\boldsymbol{x}) = \{(\boldsymbol{x}^{(k)} = (x_1^{(k)}, x_2^{(k)}, \cdots, x_{|\boldsymbol{x}^{(k)}|}^{(k)}), p(\boldsymbol{x}^{(k)}))|k = 1, 2, \cdots, N\}$，其中 $\boldsymbol{x}^{(k)}$ 为第 k 个候选句子，以及 $p(\boldsymbol{x}^{(k)})$ 是它的 ASR 后验概率。

相对于 N 最佳候选列表，词格和词混淆网络则包含的解码信息更全，包含的不确定性更丰富。如图3–2上半部分所示，词格（word lattice）是一个有向无环加权图。它被用来近似大规模词汇连续语音识别的词搜索空间，每个节点表示的是输入音频中的某个时刻点，而其中每个节点之间的转移边表示的是相关时间段的音频可能对应的词以及这个词的置信度（声学模型和语言模型的综合置信度）。相比最佳候选句子和 N 最佳候选列表，词格更有可能包含输入音频的人工转写文本（即完全正确的识别结果）。但是，词格的图结构一般都会非常庞大，不利于计算。

在词格的基础上，人们提出了一种更加高效的词图形式，即词混淆网络（Word Confusion Network, WCN）。WCN 具有更小的图规模和结构，却又不损失语音识别解码的精度。一般化的 WCN 结构如图3–2下半部分所示。WCN 通过强迫词格中发生在相近时间区域里的词相互对齐，得到了更为紧凑的词图形式。其中对齐在同一时间区域的词构成了一个组或者分段（bin）。于是形式化地，我们可以把

表 3–1 一个 N 最佳候选列表的示例（N=10）

Table 3–1 An example of 10-best hypotheses list.

排序	候选句子	归一化的后验概率
1	找乐今天武汉市是什么天气	0.950
2	找乐今天武汉是是什么天气	0.011
3	找乐今天武汉事是什么天气	0.009
4	找乐今天武汉是什么天气	0.007
5	找勒今天武汉市是什么天气	0.007
6	找乐今天武汉市什么天气	0.006
7	找勒今天武汉是是什么天气	0.005
8	找乐今天武汉事事什么天气	0.003
9	找乐今天武汉世事什么天气	0.001
10	找勒今天武汉事是什么天气	0.001
正确句子	小乐今天武汉市是什么天气	1.000

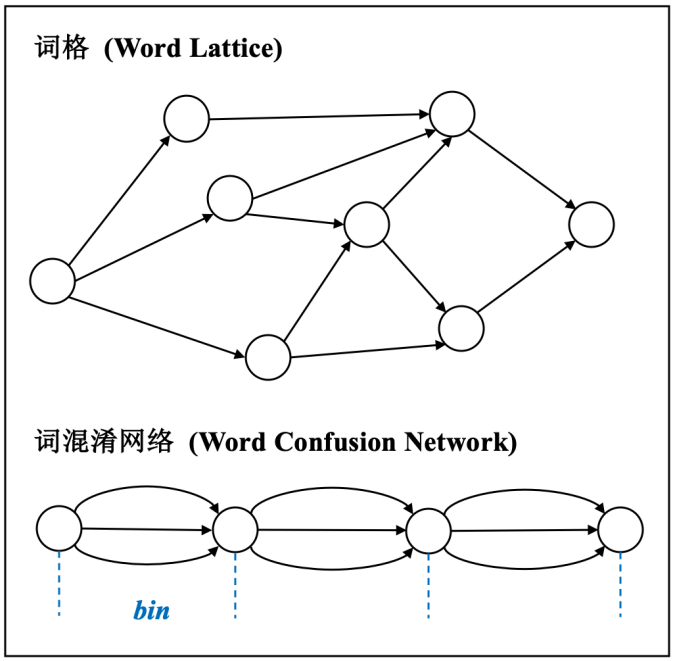


图 3–2 词格和词混淆网络的经典结构

Figure 3–2 Typical structures of word lattices and word confusion networks.

WCN 表示为一个分段序列，即 $\boldsymbol{b} = (b_1, \cdots, b_{|\boldsymbol{b}|})$，其中 $|\boldsymbol{b}|$ 为分段个数。每个分段则包括了相关时间区域里所有可能的词以及相应的后验概率，比如第 k 个分段被表示为 $b_k = \{(x_k^1, p(x_k^1)), \cdots, (x_k^{|b_k|}, p(x_k^{|b_k|}))\}$，其中 $|b_k|$ 表示该分段内的候选词的个数，x_k^i 和 $p(x_k^i)$ 分别是该分段内的第 i 个候选词以及它的语音识别后验概率。一般情况，我们会对每个分段内的候选词的后验概率归一化，即 $\sum_{i=1}^{|b_k|} p(x_k^i) = 1$。注意，有些时间区域对应的音频可能是没有说话内容的，这中情况通常会使用一个特殊的候选词（Null）来表示空的情况。

本章节将基于上述最佳候选句子、N 最佳候选列表和 WCN 三种语音识别结果展开针对语音识别不确定性的口语理解研究。

3.3 基于层级解码的非对齐式口语理解

为了针对性地研究语音识别输出编码难的问题，我们先将词对齐式的任务型口语理解转换为非词对齐式的分类或者生成任务，即将 “语音识别输出编码难的问题” 和 “语音识别结果上词对齐式语义标注难的问题” 分解开。于是，在研究上述语音识别结果的多种不确定表现形式的编码问题之前，本小节先介绍我们提出的适用于非词对齐式语义理解方法：基于层级解码的非对齐式口语理解。

首先，对于非词对齐式的口语理解数据，我们的目标是预测输入句子对应的一个特殊语义表示结构：一个由多个 “动作类型-语义槽-值（*act-slot-value*）” 三元组构成的集合。根据前面第二章介绍的那样，由动作类型引领的三元组一般分为三类：

1. 不包含语义槽值，单独一个语义动作类型的，比如像 “thankyou()” 和 “bye()” 这种用于对话开始和结束的语义表达。
2. 包含语义槽但是不包含值的，比如像 “request(phone_number)” 这类请求语义槽的值信息的语义表达。
3. 包含语义槽值的真正的三元组，比如像 “inform(pricerange=expensive)” 和 “confirm(food=Chinese)” 用来提供信息或者对信息请求确认的语义表达。

因此，根据上述 *act-slot-value* 三元组的层级结构，本文提出了相应的一种层级预测的口语理解模型。模型框架如图3–3所示，该模型主要包括四个模块：

- 共享的**输入编码器**：在本小节，我们先简单地考虑句子输入，比如人工转写句子或者最佳候选句子。
- **动作类型分类器**：该分类器以上述编码器提取的句子向量为输入，预测当前输入句子会涉及哪些动作类型（所以这是一个多类别预测的分类器）。
- **语义槽分类器**：该分类器同时以句子向量和一个动作类型为输入，预测当前

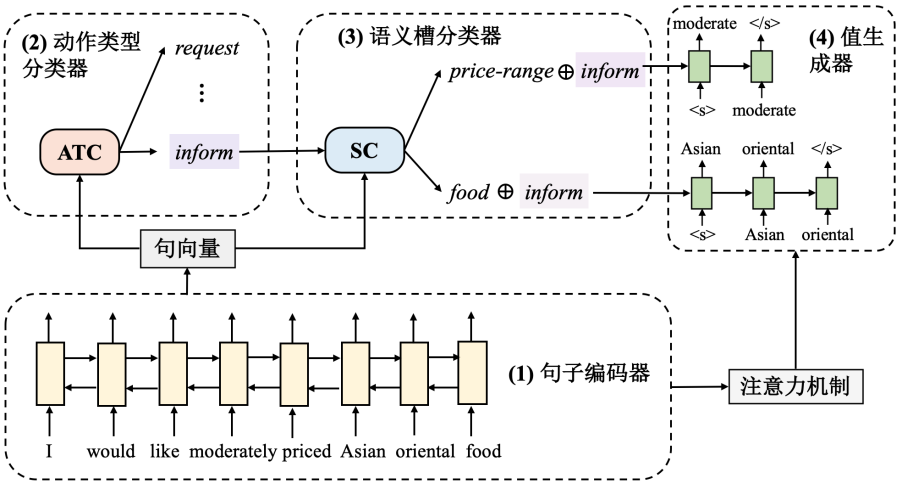


图 3–3 基于层级解码的非对齐式口语理解模型结构

Figure 3–3 A hierarchical decoding model for spoken language understanding from unaligned data.

输入句子和动作类型会涉及哪些语义槽（也是一个多类别预测的分类器）。

- **值生成器**：上述两个分类器可以分层次预测得到的 *act-slot* 两元组，该模块则为每个 *act-slot* 两元组生成相应的值（value）。

问题定义：于是，非对齐式的口语理解可以被形式化定义为，给定输入句子 $\boldsymbol{x} = (x_1, x_2, \cdots, x_{|\boldsymbol{x}|})$（也可以是 N 最佳候选列表和 WCN），预测它相应的 *act-slot-value* 三元组集合 $\boldsymbol{y}^C = \{y_1^C, y_2^C, \cdots, y_{|\boldsymbol{y}^C|}^C\}$。每一个三元组均由动作类型、语义槽和值三部分组成，即 $y_i^C = (a_i, s_i, \boldsymbol{v}_i)$，其中 a_i 表示它的动作类型，s_i 表示它的语义槽，$\boldsymbol{v}_i$ 表示语义槽的值（也是词序列）；如果不包含语义槽和值，则分别用 NULL 代替。我们的目标是根据训练数据 $\mathcal{D}_{\text{train}}$ 有效地估计给定输入预测语义信息的后验概率：

$$p(\boldsymbol{y}^C|\boldsymbol{x}) = \prod_{i=1}^{|\boldsymbol{y}^C|} p(y_i^C|\boldsymbol{x}) = \prod_{i=1}^{|\boldsymbol{y}^C|} p(a_i|\boldsymbol{x}) p(s_i|\boldsymbol{x}, a_i) p(\boldsymbol{v}_i|\boldsymbol{x}, a_i, s_i) \tag{3–1}$$

其中 $p(a_i|\boldsymbol{x})$ 为第 i 个语义三元组的动作类型的后验概率，$p(s_i|\boldsymbol{x}, a_i)$ 为第 i 个语义三元组的语义槽的后验概率，$p(\boldsymbol{v}_i|\boldsymbol{x}, a_i, s_i)$ 为第 i 个语义三元组的值的后验概率。

3.3.1 模型结构与层级式解码算法

本小节将介绍上述四个模块的具体细节。

1) **输入编码器（input encoder）**：对于输入的用户句子，我们采用双向 LSTM（BLSTM） [34, 155] 进行编码。首先，每个词都会被映射成一个词向量，即 $\mathbf{x}_i = \mathbf{W}_{\text{in}}\mathbf{o}(x_i)$，其中 $\mathbf{W}_{\text{in}} \in \mathbb{R}^{d_e \times |\mathcal{V}_{\text{in}}|}$ 是一个词向量矩阵，$\mathbf{o}(x_i) \in \mathbb{R}^{|\mathcal{V}_{\text{in}}|}$ 是一个独热向

量（one-hot vector），$\mathcal{V}_{\text{in}}$ 表示输入字典，d_e 为词向量维度，$\mathbf{x}_i \in \mathbb{R}^{d_e}$。对于输入句子 $\boldsymbol{x} = (x_1, x_2, \cdots, x_{|\boldsymbol{x}|})$，BLSTM 可以进行循环编码并获得每个词对应的隐层向量（计算过程见附录A.1）：

$$(\mathbf{h}_1, \cdots, \mathbf{h}_{|\boldsymbol{x}|}) \leftarrow \text{BLSTM}_{\Theta_{\text{enc}}}(\mathbf{x}_1, \cdots, \mathbf{x}_{|\boldsymbol{x}|}) \tag{3–2}$$

其中，BLSTM 在第 i 时刻的隐层向量由相应时刻上正向和反向 LSTM 的两个隐向量拼接得到，即 $\mathbf{h}_i = \overrightarrow{\mathbf{h}}_i \oplus \overleftarrow{\mathbf{h}}_i$，且 $\mathbf{h}_i \in \mathbb{R}^{2d_h}$，$d_h$ 表示单个 LSTM 的隐层向量维度；Θ_{enc} 为该 BLSTM 包含的所有参数。每个时刻的隐层向量代表了每个词综合句子上下文信息后的词表示向量，在最后的值生成器中会被用到。而最终输入句子的整体表示（即句向量）是由正向和反正 LSTM 的最后一个隐层向量共同拼接得到，

$$\mathbf{u} = \overrightarrow{\mathbf{h}}_{|\boldsymbol{x}|} \oplus \overleftarrow{\mathbf{h}}_1 \tag{3–3}$$

其中，$\mathbf{u} \in \mathbb{R}^{2d_h}$。句子向量的提取也可以有一些其它方案，比如常见的自注意力机制（self-attention），

$$\mathbf{u} = \sum_{i=1}^{|\boldsymbol{x}|} \alpha_i \mathbf{h}_i \tag{3–4}$$

$$\alpha_i = \frac{\exp(\mathbf{w}_u^\top \tanh(\mathbf{W}_u \mathbf{h}_i + \mathbf{b}_u))}{\sum_{j=1}^{|\boldsymbol{x}|} \exp(\mathbf{w}_u^\top \tanh(\mathbf{W}_u \mathbf{h}_j + \mathbf{b}_u))} \tag{3–5}$$

其中 $\mathbf{W}_u \in \mathbb{R}^{2d_h \times 2d_h}$，$\mathbf{w}_u \in \mathbb{R}^{2d_h}$，$\mathbf{b}_u \in \mathbb{R}^{2d_h}$ 均为模型参数。

句子向量 $\mathbf{u}$ 将在后续的语义动作分类器和语义槽分类器中使用，隐层向量 $\{\mathbf{h}_1, \cdots, \mathbf{h}_{|\boldsymbol{x}|}\}$ 将在最后的基于注意力机制和指针网络 [156-158] 的值生成器中用到。

2) **动作类型分类器**（act classifier）：因为同一句话可能对应多个不同的动作类型，动作类型的预测被定义为一个多类别分类问题。比如，“帮我找一家中餐馆并把它的电话给我” 这句话包含了 *inform* 和 *request* 两种动作类型。遵从一般的做法，我们为每一个动作类型类别训练一个二分类器。具体地，我们采用两层前馈神经网络计算每个类别标签存在的可能性：

$$\mathbf{r}_{\text{act}} = \text{ReLU}(\mathbf{W}_{\text{act}}^1 \mathbf{u} + \mathbf{b}_{\text{act}}^1) \tag{3–6}$$

$$\mathbf{z}_{\text{act}} = \sigma(\mathbf{W}_{\text{act}}^2 \mathbf{r}_{\text{act}} + \mathbf{b}_{\text{act}}^2) \tag{3–7}$$

其中，$\mathbf{W}_{\text{act}}^1 \in \mathbb{R}^{d_h \times 2d_h}$，$\mathbf{W}_{\text{act}}^2 \in \mathbb{R}^{|\mathcal{V}_{\text{act}}| \times d_h}$，$\mathbf{b}_{\text{act}}^1 \in \mathbb{R}^{d_h}$，$\mathbf{b}_{\text{act}}^2 \in \mathbb{R}^{|\mathcal{V}_{\text{act}}|}$ 均为模型参数，$\mathcal{V}_{\text{act}}$ 为所有动作类型组成的词表。ReLU 为整流线性单位函数（Rectified Linear

Unit），具体定义为 $\mathrm{ReLU}(x) = \max(0, x)$。$\sigma$ 为 S 函数（Sigmoid 函数），可以将动作类型的预测值归一化为 0 到 1 之间的数值。最后 $\mathbf{z}_{\text{act}} \in \mathbb{R}^{|\mathcal{V}_{\text{act}}|}$ 输出向量包含了每个动作类型出现的概率值。于是，第 i 个语义三元组 y_i^C 的动作类型的后验概率为 $\mathbf{z}_{\text{act}}$ 对应维度的数值，即 $p(a_i|\boldsymbol{x}) = [\mathbf{z}_{\text{act}}]_{\text{id}(a_i)}$，其中 $\text{id}(a_i)$ 是动作类型 a_i 在 $\mathcal{V}_{\text{act}}$ 中的编号，$\text{id}(a_i) = \{1, 2, \cdots, |\mathcal{V}_{\text{act}}|\}$。

在训练阶段，二元交叉熵（Binary Cross Entropy，BCE）被用来计算预测概率和真实标注之间的损失误差:

$$\mathcal{L}_{\text{act}}(\boldsymbol{x}, \boldsymbol{y}^C) = -\sum_{a \in \mathcal{V}_{\text{act}}} \Big(\mathbb{I}\{a \in \boldsymbol{y}^C\} \log p(a|\boldsymbol{x}) + (1 - \mathbb{I}\{a \in \boldsymbol{y}^C\}) \log(1 - p(a|\boldsymbol{x})) \Big) \quad (3\text{–}8)$$

其中 $\mathbb{I}\{a \in \boldsymbol{y}^C\}$ 是表示任意动作类型 $a \in \mathcal{V}_{\text{act}}$ 是否出现在语义标注中的指示函数（返回值 0 表示未出现，返回值 1 表示出现）。在测试阶段，预测概率大于 0.5 的类别将被接受并用于后续模块的计算。

3) **语义槽分类器**（slot classifier）：语义槽分类器和上述动作类型分类器的结构几乎一摸一样，唯一的不同时语义槽分类器还会把上一个模块预测得到的动作类型作为输入。如果第 i 个语义三元组的动作类型 a_i 还需要后续的语义槽，那么我们继续进行语义槽的预测。我们首先得到动作类型的向量表示 $\mathbf{a}_i = \mathbf{W}_{\text{act}}\mathbf{o}(a_i)$，其中 $\mathbf{W}_{\text{act}} \in \mathbb{R}^{d_e \times |\mathcal{V}_{\text{act}}|}$ 是一个动作类型向量矩阵。同样，我们采用两层前馈神经网络计算每个语义槽存在的可能性：

$$\mathbf{r}_{\text{slot}} = \mathrm{ReLU}(\mathbf{W}^1_{\text{slot}}(\mathbf{u} \oplus \mathbf{a}_i) + \mathbf{b}^1_{\text{slot}}) \quad (3\text{–}9)$$

$$\mathbf{z}_{\text{slot}} = \sigma(\mathbf{W}^2_{\text{slot}}\mathbf{r}_{\text{slot}} + \mathbf{b}^2_{\text{slot}}) \quad (3\text{–}10)$$

其中，$\mathbf{W}^1_{\text{slot}} \in \mathbb{R}^{d_h \times (2d_h + d_e)}$，$\mathbf{W}^2_{\text{slot}} \in \mathbb{R}^{|\mathcal{V}_{\text{slot}}| \times d_h}$，$\mathbf{b}^1_{\text{slot}} \in \mathbb{R}^{d_h}$，$\mathbf{b}^2_{\text{slot}} \in \mathbb{R}^{|\mathcal{V}_{\text{slot}}|}$ 均为模型参数，$\mathcal{V}_{\text{slot}}$ 为所有语义槽组成的词表。最后 $\mathbf{z}_{\text{slot}} \in \mathbb{R}^{|\mathcal{V}_{\text{slot}}|}$ 输出向量包含了每个语义槽出现的概率值。于是，第 i 个语义三元组 y_i^C 的语义槽的后验概率为 $\mathbf{z}_{\text{slot}}$ 对应维度的数值，即 $p(s_i|\boldsymbol{x}, a_i) = [\mathbf{z}_{\text{slot}}]_{\text{id}(s_i)}$，其中 $\text{id}(s_i)$ 是动作类型 s_i 在 $\mathcal{V}_{\text{slot}}$ 中的编号，$\text{id}(s_i) = \{1, 2, \cdots, |\mathcal{V}_{\text{slot}}|\}$。

在训练阶段，同样使用 BCE 计算损失误差：

$$\begin{aligned} \mathcal{L}_{\text{slot}}(\boldsymbol{x}, \boldsymbol{y}^C) = -\sum_{i=1}^{|\boldsymbol{y}^C|} \sum_{s \in \mathcal{V}_{\text{slot}}} \Big(& \mathbb{I}\{(a_i, s) \in \boldsymbol{y}^C\} \log p(s|\boldsymbol{x}, a_i) \\ & + (1 - \mathbb{I}\{(a_i, s) \in \boldsymbol{y}^C\}) \log(1 - p(s|\boldsymbol{x}, a_i)) \Big) \end{aligned} \quad (3\text{–}11)$$

其中 $\mathbb{I}\{(a_i, s) \in \boldsymbol{y}^C\}$ 是表示任意语义槽 $s \in \mathcal{V}_{\text{slot}}$ 是否和动作类型 a_i 一起出现在语义标注中的指示函数（返回值 0 表示未出现，返回值 1 表示出现）。在测试阶段，预测概率大于 0.5 的语义槽将被接受并用于后续值生成模块的计算。

4) **值生成器**（value decoder）：我们采用基于注意力机制和指针网络 [156-158] 的序列到序列（sequence-to-sequence）生成模型为相关的 act-slot 对预测相应的值。如果某个语义三元组 $(a, s, \boldsymbol{v})$ 需要一个值的话，记相应值的词序列为 $\boldsymbol{v} = (v_1, v_2, \cdots, v_{|\boldsymbol{v}|})$。值序列的最后一个位置 $v_{|\boldsymbol{v}|}$ 为特殊词 "</s>"，其表示序列的结尾符。我们采用一个单向 LSTM 对该序列进行解码，

$$\mathbf{h}_t^{\text{value}} = \text{f}_{\text{LSTM}}(\mathbf{v}_{t-1}, \mathbf{h}_{t-1}^{\text{value}}) \tag{3–12}$$

其中 $\mathbf{v}_i = \mathbf{W}_{\text{in}}\mathbf{o}(v_i) \in \mathbb{R}^{d_e}$ 为第 i 个词的词向量，$\mathbf{h}_i^{\text{value}} \in \mathbb{R}^{d_h}$ 为第 i 个时刻的隐层向量；v_0 是起始符 "<s>"；解码隐层向量由句子编码的隐向量初始化，即 $\mathbf{h}_0^{\text{value}} = \vec{\mathbf{h}}_{|\boldsymbol{x}|}$。为了融合相关 act 和 slot 的信息，我们定义如下新的状态表示向量 $\tilde{\mathbf{s}}_i \in \mathbb{R}^{2n}$：

$$\tilde{\mathbf{h}}_t^{\text{value}} = \mathbf{W}_{\text{value}}^1(\mathbf{h}_t^{\text{value}} \oplus \mathbf{a} \oplus \mathbf{s}) + \mathbf{b}_{\text{value}}^1 \tag{3–13}$$

其中 $\mathbf{W}_{\text{value}}^1 \in \mathbb{R}^{2d_h \times (d_h + 2d_e)}$，$\mathbf{b}_{\text{value}}^1 \in \mathbb{R}^{2d_h}$ 为相关参数；$\mathbf{a} = \mathbf{W}_{\text{act}}\mathbf{o}(a) \in \mathbb{R}^{d_e}$ 为动作类型 a 的向量表示，$\mathbf{s} = \mathbf{W}_{\text{slot}}\mathbf{o}(s) \in \mathbb{R}^{d_e}$ 为语义槽 s 的向量表示（$\mathbf{W}_{\text{slot}} \in \mathbb{R}^{d_e \times |\mathcal{V}_{\text{slot}}|}$）。$\tilde{\mathbf{h}}_t^{\text{value}}$ 被用于注意力机制中来计算上下文信息向量 $\mathbf{c}_t \in \mathbb{R}^{2d_h}$：

$$\mathbf{c}_t = \sum_{j=1}^{|\boldsymbol{x}|} \alpha_{tj}\mathbf{h}_j \tag{3–14}$$

$$\alpha_{tj} = \frac{\exp(\mathbf{h}_j^{\top}\tilde{\mathbf{h}}_t^{\text{value}})}{\sum_{k=1}^{|\boldsymbol{x}|}\exp(\mathbf{h}_k^{\top}\tilde{\mathbf{h}}_t^{\text{value}})} \tag{3–15}$$

其中 α_{tj} 表示的是输出序列第 t 个词对输入序列第 j 个词的关注（依赖）程度。引入了相关 act 和 slot 的信息的 $\tilde{\mathbf{h}}_t^{\text{value}}$ 会有助于指导注意力机制的运算。

值序列在第 t 时刻的词预测分两部分：生成和指针。首先，普通生成的话，我们采用一个基于前馈神经网络的输出层，

$$p_{\text{gen}}(v_t|\boldsymbol{v}_{<t}, \boldsymbol{x}, a, s) = \text{softmax}_{v_t}\left(\mathbf{W}_{\text{in}}^{\top}(\mathbf{W}_{\text{value}}^2(\tilde{\mathbf{h}}_t^{\text{value}} \oplus \mathbf{c}_t) + \mathbf{b}_{\text{value}}^2)\right) \tag{3–16}$$

其中 $\mathbf{W}_{\text{value}}^2 \in \mathbb{R}^{d_e \times 4d_h}$，$\mathbf{b}_{\text{value}}^2 \in \mathbb{R}^{d_e}$ 为相关参数，$\text{softmax}_{v_t}(\cdot)$ 为词 v_t 的预测概率。

为了解决值生成过程中常见的未登录词（Out Of Vocabulary，OOV）问题，我们采用指针网络 [156] 对上述基本的序列生成模型进行加强。根据前面计算的注意力权重 α_{tj}，我们可以得到当前输入句子中的所包含的词的概率分布，

$$p_{\text{copy}}(v_t|\boldsymbol{v}_{<t}, \boldsymbol{x}, a, s) = \sum_{j=1}^{|\boldsymbol{x}|}\mathbb{I}\{v_j = v_t\}\alpha_{tj} \tag{3–17}$$

其中 $\mathbb{I}$ 为 0/1 指示函数，如果 v_j 等于 v_t 则指示函数返回 1，否则返回 0。因此，我们可以得到一个在扩展的词表（基础词表加上输入句子中的词）上的最终分布：

$$p(v_t|\boldsymbol{v}_{<t}, \boldsymbol{x}, a, s) = g_t p_{\text{gen}}(v_t|\boldsymbol{v}_{<t}, \boldsymbol{x}, a, s) + (1 - g_t) p_{\text{copy}}(v_t|\boldsymbol{v}_{<t}, \boldsymbol{x}, a, s) \tag{3–18}$$

$$g_t = \sigma((\mathbf{v}_t \oplus \tilde{\mathbf{h}}_t^{\text{value}} \oplus \mathbf{c}_t)^\top \mathbf{w}_{\text{value}}^3 + b_{\text{value}}^3) \tag{3–19}$$

其中 g_t 为两种分布的平衡因子，$\mathbf{w}_{\text{value}}^3 \in \mathbb{R}^{(4d_h+d_e)}$，$b_{\text{value}}^3 \in \mathbb{R}$ 为相关参数。于是，给定输入序列 $\boldsymbol{x}$ 为相应动作类型 a 和语义槽 s 生成值 $\boldsymbol{v}$ 的后验概率为：

$$p(\boldsymbol{v}|\boldsymbol{x}, a, s) = \prod_{t=1}^{|\boldsymbol{v}|} p(v_t|\boldsymbol{v}_{<t}, \boldsymbol{x}, a, s) \tag{3–20}$$

在训练中，值生成器的损失误差由交叉熵给定：

$$\mathcal{L}_{\text{value}}(\boldsymbol{x}, \boldsymbol{y}^C) = -\sum_{i=1}^{|\boldsymbol{y}^C|} \log p(\boldsymbol{v}_i|\boldsymbol{x}, a_i, s_i) \tag{3–21}$$

在训练数据 $\boldsymbol{D}_{\text{train}}$ 上的总体损失函数为三个输出模块的损失误差之和：

$$\mathcal{L}_{\text{total}}(\boldsymbol{\Theta}) = \sum_{(\boldsymbol{x}, \boldsymbol{y}^C) \in \boldsymbol{D}_{\text{train}}} \mathcal{L}_{\text{act}}(\boldsymbol{x}, \boldsymbol{y}^C) + \mathcal{L}_{\text{slot}}(\boldsymbol{x}, \boldsymbol{y}^C) + \mathcal{L}_{\text{value}}(\boldsymbol{x}, \boldsymbol{y}^C) \tag{3–22}$$

其中 $\boldsymbol{\Theta}$ 表示所有需要训练的参数。在训练阶段，每个模块可以同时进行多任务联合学习，语义槽分类器和值生成器需要的前面模块的预测结果则是直接使用标注数据中的答案；而在测试阶段，每个模块需要按顺序执行，比如语义槽分类器和值生成器需要前面模块给出预测结果后再执行。这个层级式解码过程的伪代码如算法3–1所示。

3.3.2 在 N 最佳候选列表上的应用

上述基于层级解码的口语理解模型针对的输入类型是句子，可以是理想情况下的人工转写句子，也可以是语音识别结果中的最佳候选句子。如果要将该基于层级解码的模型应用到 N 最佳候选列表上，一般有两种方案可以选择：

- 在训练阶段使用人工转写句子或者最佳候选句子作为输入训练口语理解模型；在测试阶段，遍历 N 最佳候选列表中的所有句子，通过训练好的口语理解模型，每个候选句子都能得到相应的预测结果和后验概率，最终综合语音识别后验概率和语义理解后验概率对所有的语义解析结果进行整合。

算法 3–1 非词对齐式口语理解的层级解码

Input: 输入句子 $\boldsymbol{x}$，领域本体，模型（动作类型分类器、语义槽分类器、值生成器）
Output: act-slot-value 三元组集合 $\boldsymbol{y}^C$

初始化一个空集合 $\boldsymbol{y}^C$;
通过动作类型分类器预测动作类型集合，$\mathcal{S}_{\text{act}} = \{a | p(a|\boldsymbol{x}) > 0.5\}$;
for $a \in \mathcal{S}_{act}$ **do**
    if 根据领域本体，动作类型 a 不需要带语义槽 **then**
        将 a 添加到 $\boldsymbol{y}^C$ 集合中;
    else
        通过语义槽分类器预测语义槽集合，$\mathcal{S}_{\text{slot}} = \{s | p(s|\boldsymbol{x}, a) > 0.5\}$;
        for $s \in \mathcal{S}_{slot}$ **do**
            if 根据领域本体，语义槽 s 不需要带值 **then**
                将 act-slot 二元组 a-s 添加到 $\boldsymbol{y}^C$ 集合中;
            else
                通过值生成器预测相应的值，$\hat{\boldsymbol{v}} = \arg\max_{\boldsymbol{v}} p(\boldsymbol{v}|\boldsymbol{x}, a, s)$;
                将 act-slot-value 三元组 a-s-$\hat{\boldsymbol{v}}$ 添加到 $\boldsymbol{y}^C$ 集合中;
            end
        end
    end
end
返回 $\boldsymbol{y}^C$ 集合。

- 在训练阶段就使用 N 最佳候选列表，把所有候选句子输入到共享的句子编码器中。然后在计算句向量时，基于语音识别后验概率计算所有候选句子句向量的加权和，所有候选句子的每个词的隐层向量也乘上当前候选句子的语音识别后验概率。这样的做法相比上一个方案，只需要修改输入编码器模块的计算方式，而且训练和测试的输入是一致的。

3.4 基于词混淆网络的口语理解架构

上一小节中的输入编码器仅适用于序列型输入（人工转写句子，最佳候选句子以及 N 最佳候选列表），对于词混淆网络这种更为复杂的词图形式的语音识别结果并不适用。因此，在本小节，我们提出了一种对词混淆网络的编码器，并可以进一步对词混淆网络与对话历史信息进行联合编码。该基于词混淆网络的口语理解框架概括如图3–4所示。

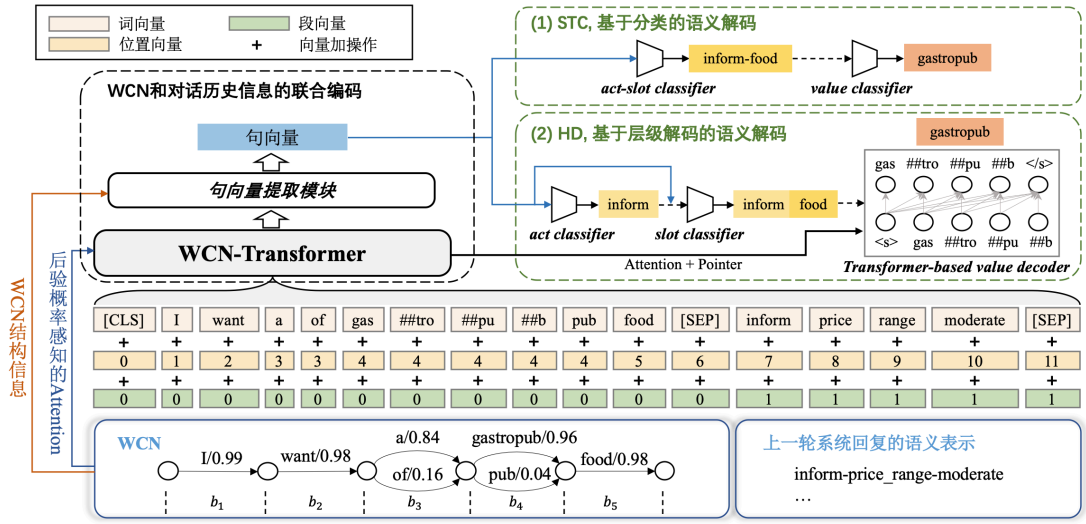


图 3–4 基于 WCN-Transformer 的稳健口语理解框架

Figure 3–4 An overview of our proposed WCN-Transformer based spoken language understanding.

3.4.1 基于 Transformer 的词混淆网络与对话历史信息的联合编码

3.4.1.1 输入层

在第3.2节中，我们把 WCN 形式化为一个 bin 序列，即 $\boldsymbol{b} = (b_1, \cdots, b_{|\boldsymbol{b}|})$，其中 $|\boldsymbol{b}|$ 为 bin 个数（比如图3–4中 WCN 例子的 bin 数量为五）。每个 bin 则包括了相关时间区域里所有可能的词以及相应的语音识别后验概率，即

$$b_k = \{(x_k^1, p(x_k^1)), \cdots, (x_k^{|b_k|}, p(x_k^{|b_k|}))\} \tag{3–23}$$

其中 $|b_k|$ 表示第 k 个 bin 里的候选词个数。在 WCN 我们采用特殊的符号 NULL 表示空的词（即没有说话内容的语音分段），但是正常编码的时候我们会选择过滤掉 NULL。为了让 WCN 的输入适用于 Transformer 模型，我们将 WCN 里包含的词全部平铺成一个序列，即

$$\boldsymbol{x}^{\text{WCN}} = (x_1^1, \cdots, x_1^{|b_1|}, \cdots, x_{|\boldsymbol{b}|}^1, \cdots, x_{|\boldsymbol{b}|}^{|b_{|\boldsymbol{b}|}|}) \tag{3–24}$$

在基于自注意力机制（self-attenion）的 Transformer 中，需要输入每个词的位置信息来编码词的顺序关系。因为同一个 bin 中的候选词之间是无序的，于是我们将所在 bin 的位置作为每个候选词的位置编号（即基于时刻的位置编号）：

$$\text{pos}(\boldsymbol{x}^{\text{WCN}}) = (1, \cdots, 1, \cdots, |\boldsymbol{b}|, \cdots, |\boldsymbol{b}|) \tag{3–25}$$

除了带有丰富的不确定信息的 WCN，我们也可以引入对话历史信息进一步对语音识别不确定性进行消歧。当前用户可能说的话和上一轮系统对他（她）的

回复有很大关系，于是我们采用上一轮系统回复的语义表示（同样也是一种 act-slot-value 的是三元组集合）作为 WCN 的补充输入。我们形式化地定义上一轮系统回复的语义表示为 $A = ((a_1, s_1, v_1), \cdots, (a_{|A|}, s_{|A|}, v_{|A|}))$，其中 $|A|$ 是三元组的个数，a_k, s_k 和 v_k 分别是第 k 个三元组的动作类型、语义槽和值。我们同样把这些三元组展开为一个词序列，

$$\boldsymbol{x}^{\mathrm{SA}} = (a_1, s_1, v_1, \cdots, a_{|A|}, s_{|A|}, v_{|A|}) \tag{3–26}$$

其中一些动作类型和语义槽的名称可能是由多个词组成的，这种情况会先被拆分为多个词的序列，比如 “price_range” 会被分为 “price” 和 “range”。不同于 WCN，$\boldsymbol{x}^{\mathrm{SA}}$ 中的每个词的位置编号是完全按顺序递增的。

为了对 WCN 和对话历史信息进行联合编码，我们把两者的输入序列拼接成一个更大的联合序列，再把它们输入到基于 Transformer 的编码模型中。拼接后的输入序列为

$$\boldsymbol{x} = (x_1, \cdots, x_T) = \texttt{[CLS]} \oplus \textsc{Tok}(\boldsymbol{x}^{\mathrm{WCN}}) \oplus \texttt{[SEP]} \oplus \textsc{Tok}(\boldsymbol{x}^{\mathrm{SA}}) \oplus \texttt{[SEP]} \tag{3–27}$$

其中 $\oplus$ 表示序列拼接，$\textsc{Tok}(\cdot)$ 表示词切割（tokenization）操作，比如一个长的英文词会被切割成好几个子词（sub-word）。在 WCN 中，分割后的子词将继承原先词的位置编号以及语音识别后验概率。另外，[CLS] 和 [SEP] 是用于分割的辅助词。T 是整个序列所有词的个数，$T = |\textsc{Tok}(\boldsymbol{x}^{WCN})| + |\textsc{Tok}(\boldsymbol{x}^{SA})| + 3$。

最终，WCN 和对话历史信息联合的输入向量是由词向量 $E_{\mathrm{token}}(\cdot)$、位置向量 $E_{\mathrm{pos}}(\cdot)$、段向量 $E_{\mathrm{seg}}(\cdot)$ 三部分组成 [102]：

$$\mathbf{x}_t = \mathrm{LayerNorm}(E_{\mathrm{token}}(x_t) + E_{\mathrm{pos}}(x_t) + E_{\mathrm{seg}}(x_t)) \tag{3–28}$$

其中 $\mathbf{x}_t \in \mathbb{R}^{d_x}$，LayerNorm 表示的是层归一化操作 [159]，另外我们采用两种段向量区分 WCN 和对话历史的输入。

3.4.1.2 联合编码器：WCN-Transformer

基于 Transformer 的 WCN 和对话历史信息的联合编码器由一系列的双向 Transformer 层 [35] 叠加构成，每一层 Transformer 都包含了一个多头的自注意力模块以及一个带残差连接的前馈神经网络。假设我们采用的 Transformer 结构共有 L 层，记第 l 层的输入为 $\mathbf{X}^l = (\mathbf{x}_1^l, \cdots, \mathbf{x}_T^l)$，其中 $l \in \{1, \cdots, L\}$。那么第 l 层输出的隐层向量（也就是第 $l+1$ 层的输入）可以通过如下自注意力机制计算得到：

$$e_{ij}^{l,h} = \frac{(\mathbf{W}_Q^{l,h}\mathbf{x}_i^l)^\top (\mathbf{W}_K^{l,h}\mathbf{x}_j^l)}{\sqrt{d_x/H}} \tag{3–29}$$

$$\alpha_{ij}^{l,h} = \frac{\exp(e_{ij}^{l,h})}{\sum_{k=1}^{T}\exp(e_{ik}^{l,h})} \tag{3–30}$$

$$\mathbf{z}_i^{l,h} = \sum_{j=1}^{T}\alpha_{ij}^{l,h}(\mathbf{W}_V^{l,h}\mathbf{x}_j^l) \tag{3–31}$$

$$\mathbf{z}_i^l = \mathbf{z}_i^{l,1} \oplus \mathbf{z}_i^{l,2} \oplus \cdots \oplus \mathbf{z}_i^{l,H} \tag{3–32}$$

$$\tilde{\mathbf{x}}_i^{l+1} = \text{LayerNorm}(\mathbf{x}_i^l + \text{FC}(\mathbf{z}_i^l)) \tag{3–33}$$

$$\mathbf{x}_i^{l+1} = \text{LayerNorm}(\tilde{\mathbf{x}}_i^{l+1} + \text{FC}(\text{ReLU}(\text{FC}(\tilde{\mathbf{x}}_i^{l+1})))) \tag{3–34}$$

其中包含了 H 个独立的自注意力模块（即多头注意力机制），$\alpha_{ij}^{l,h}$ 表示第 l 层第 h 个自注意力模块计算得到的第 i 个词对第 j 个词的关注概率；FC 是一个全连接层（即带偏置向量的线性变换），LayerNorm 表示的是层归一化操作 [159]（即对模型中间隐层向量的值进行平移和伸缩变换），$1 \le h \le H, \mathbf{W}_Q^{l,h}, \mathbf{W}_K^{l,h}, \mathbf{W}_V^{l,h} \in \mathbb{R}^{(d_x/H)\times d_x}$ 均为相关参数。

WCN 后验概率感知的自注意力机制　我们提出在 Transformer 的自注意力机制运算中融入每个词的语音识别后验概率，并对上述公式3–29进行了修改。我们记输入的每个词的语音识别后验概率为 $(p_1, \cdots, p_T)$，其中

$$p_t = \begin{cases} p(x_t) & \text{，如果} x_t \in \text{Tok}(\boldsymbol{x}^{\text{WCN}}) \\ 1.0 & \text{，否则。} \end{cases} \tag{3–35}$$

换句话说，如果一个词是 WCN 里的，则使用它的语音识别后验概率，否则默认它的语音识别后验概率为 1（即 [CLS]、[SEP] 以及 $\text{Tok}(\boldsymbol{x}^{\text{SA}})$ 中的词）。如果 WCN 中的一个词被分割为多个子词，那么每个子词都会继承它的后验概率。

然后我们通过修改公式3–29来注入 WCN 中语音识别后验概率，

$$e_{ij}^{l,h} = \frac{(\mathbf{W}_Q^{l,h}\mathbf{x}_i^l)^\top(\mathbf{W}_K^{l,h}\mathbf{x}_j^l)}{\sqrt{d_x/H}} + \lambda^{l,h}p_j \tag{3–36}$$

其中 $\lambda^{l,h}$ 是一个用于调节后验概率幅度的参数。

在经过 L 层的 Transformer 计算后，我们可以得到最终输出向量 $\mathbf{H} = (\mathbf{h}_1, \cdots, \mathbf{h}_T)$，其中 $\mathbf{h}_t = \mathbf{x}_t^{L+1}$。

3.4.1.3　句向量提取模块

类似于第3.3节，我们也需要提取输入 WCN 的句向量。在上述最终输出向量中，第一个位置 [CLS] 符号对应的向量（即 $\mathbf{h}_1$）经常被用来表示整个输入。此外，我们还提出了一种考虑 WCN 结构信息的句向量提取方法。

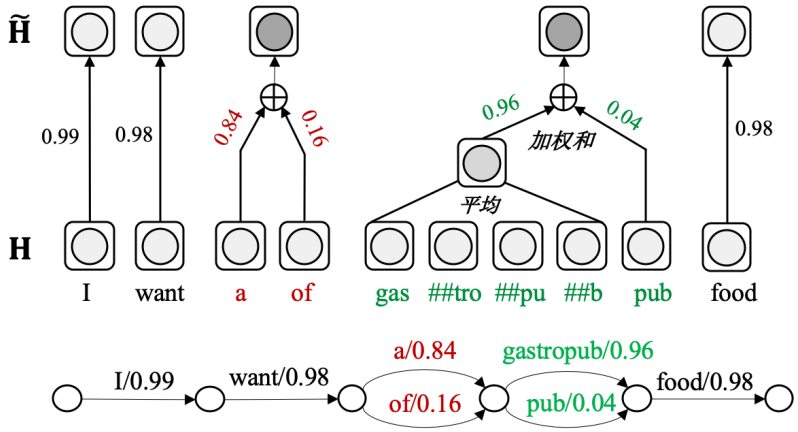


图 3–5 WCN 中向量合并的示例

Figure 3–5 An example of vector aggregations in the WCN part

首先，如图3–5所示，WCN 中分属同一个 bin 的所有隐层向量会被整合成 bin 级别的向量（即 bin 向量），具体分两步：

1. 第一步，bin 里面每个候选词的子词对应的隐层向量会被平均，然后得到候选词的向量表示；
2. 第二步，同一个 bin 中所有候选词的向量将按照语音识别后验概率进行加权合并，然后得到 bin-level 的向量表示。

我们记经过上述 WCN 向量合并后的隐层向量为 $\widetilde{\mathbf{H}} = (\tilde{\mathbf{h}}_1, \cdots, \tilde{\mathbf{h}}_{T'})$，其中，$T' = |\boldsymbol{b}| + |\textsc{Tok}(\boldsymbol{x}^{\text{SA}})| + 3$，且 $T' \leq T$。因为除了 WCN 部分，其他的隐层向量不变，所以可得 $\tilde{\mathbf{h}}_1 = \mathbf{h}_1$，$\widetilde{\mathbf{H}}_{1+|\boldsymbol{b}|:T'} = \mathbf{H}_{1+|\textsc{Tok}(\boldsymbol{x}^{\text{WCN}})|:T}$。

然后我们在新的 bin 向量基础上采用自注意力机制计算出一个句向量：

$$\mathbf{u}' = \sum_{t=2}^{T'} \left[\frac{\exp(\mathbf{w}_u^\top \tanh(\mathbf{W}_u \tilde{\mathbf{h}}_t + \mathbf{b}_u))}{\sum_{j=2}^{T'} \exp(\mathbf{w}_u^\top \tanh(\mathbf{W}_u \tilde{\mathbf{h}}_j + \mathbf{b}_u))} \cdot \tilde{\mathbf{h}}_t \right] \tag{3–37}$$

其中 $\mathbf{b}_u \in \mathbb{R}^{d_x}, \mathbf{w}_u \in \mathbb{R}^{d_x}$ 以及 $\mathbf{W}_u \in \mathbb{R}^{d_x \times d_x}$ 都是模型参数。最终，我们采用 [CLS] 符号位置的隐层向量 $\mathbf{h}_1$ 和基于自注意力得到 $\mathbf{u}'$ 共同拼接作为最终的句向量表示：

$$\mathbf{u} = \mathbf{h}_1 \oplus \mathbf{u}' \tag{3–38}$$

3.4.2 鉴别式和生成式的语义解码方法

为了衡量上述提出的基于 Transformer 的 WCN 和对话历史信息的联合编码网络，我们分别应用了鉴别式和生成式的语义解码方法。其中鉴别式语义解码方法

采用传统的分类的思想，而生成式的语义解码方法则采用第3.3节的层级式解码算法。

语义元组分类器 (Semantic Tuple Classifier, STC)：基于前文编码器得到的最终句向量 **u**，我们为每一个 “动作类型-语义槽（act-slot）对” 设置一个二元分类器预测它们存在与否，然后为每一种 act-slot 对设置一个值分类器预测相应的值[①][②]。该方法虽然简单，但是无法预测在训练集中没有出现过的值。

基于 Transformer 的层级式解码（Hierarchical Decoder，HD）：相比于前面鉴别式的分类思想，生成式的方法可以有更好的泛化能力，即有可能预测出在训练集中没有出现过的值。我们几乎完全采用第3.3节的层级式解码算法，除了以下两点修改：

1. Act 和 slot 的名称可能包含多个单词，我们采用这些单词的词向量的平均作为相应 act 或者 slot 的向量表示；
2. 层级式解码中值生成器的 LSTM 网络被替换为 Transformer 模型。

3.5 实验与分析

3.5.1 实验设置

3.5.1.1 数据集

在本章中，我们将使用第二届国际对话跟踪挑战赛的公开数据（DSTC-2）[③]来评估我们的口语理解方法。在 “查找餐馆” 的应用领域下，相关任务型对话系统与用户交互获得了这些对话的原始数据。该数据中的训练集、校验集、测试集分别包含 11677、3934、9890 个句子。虽然 DSTC-2 提供了完整的领域本体，但为了衡量口语理解模型的泛化能力，我们仍假设每个语义槽的所有可能的值不是提前给定的。

3.5.1.2 WCN 剪枝预处理

为了缩减平铺后的 WCN 序列的长度，我们会对 WCN 中的候选词做剪枝操作：

- 语气词：WCN 中容易出现很多没有意义的语气词，比如 *uh*，*oh* 等。我们采用了 Jagfeld 等 [78] 的做法。

① 有一些 act-slot 对是不需要 value 的，比如 *thankyou, goodbye, request-phone, request-food* 等等；这种情况就不需要一个值分类器。

② 一个前提假设：同一个 act-slot 在一句话中出现，只会有一个值，不存在多个值的情况。

③ http://camdial.org/~mh521/dstc

- 语音识别后验概率极低的词：我们根据 Jagfeld 等 [78] 的建议，设置了 0.001 的阈值，把语音识别后验概率低于这个阈值的候选词删除掉。

另外，DSTC-2 数据中仅 batch 字段下的识别结果提供了 WCN，所以为了公平比较本章实验采用的最佳候选句子、N 最佳候选列表以及 WCN 均来自 batch 字段，而不是 live 字段。其最佳候选句（1-best）的语音识别词错误率大约为 30%；N 最佳候选列表中包含人工转写句子的数据比例是 57.00%，而 WCN 在所有路径中包含人工转写句子的数据比例是 62.83%（剪枝预处理后是 59.48%）。因此 WCN 的一个潜在优势就是包含语音识别正确结果的可能性更高。

3.5.1.3 其它设置

在基于 Transformer 的 WCN 编码器中，我们采用了英文的 `bert-base-uncased` ①模型初始化 Transformer 里的参数（除了 $\lambda^{l,h}$）以及输入层的词向量、位置向量和段向量，其他参数则是采用均值为 0 方差为 0.02 的正态分布随机初始化。在训练过程中，我们采用 Adam [160] 优化器进行参数调优，并采用预热（warm-up）的方式调节学习率。我们设置预热率为 0.1，并从 {5e-5, 3e-5, 2e-5} 中选择峰值学习率。Adam 中的权值衰减（weight decay）设置为 0.01，梯度最大范数的修剪阈值为 1。我们设置 Transformer 模块的 dropout 率为 0.1，而句向量提取模型和语义解码模型的 dropout 率均为 0.3。

对于一些使用 BLSTM 的基线模型，我们则采用 100 维预训练的 Glove [99] 词向量初始化相应的词向量参数 ②。设置所有参数的学习率为 0.001，并在训练阶段固定不变。梯度最大范数的修剪阈值为 5，所有模块的 dropout 率均为 0.3。

评测指标采用的是 *act-slot-value* 三元组的 F_1 得分以及句子级的准确率（即要求一个句子预测的 *act-slot-value* 三元组完全正确）。所有的模型都被训练 50 轮，并保存在校验集上性能表现最好的参数。每个实验都是用了不同的随机种子运行五遍，最后平均每次的性能得分。

3.5.2 主要结果

在本小节我们将展示我们的方法以及相应的基线系统在 DSTC-2 数据集上的主要对比结果，如表3–2所示。在实验中我们对四种输入进行了研究，分别是理想情况的人工转写句子（manual）、最佳候选句子（1-best）、N 最佳候选列表（10-best）以及 WCN。其次，我们对比了三个基线系统：

① https://github.com/google-research/bert#pre-trained-models

② http://nlp.stanford.edu/data/glove.6B.zip

表 3–2 基线模型以及我们的方法在测试集上的 F_1 得分和句子级准确率

Table 3–2 F_1 scores and utterance-level accuracies of baseline models and our proposed model on the test set.

	Models	**Train→Test**	**STC**	
			F_1(%)	**Acc.**(%)
(a)	BLSTM+Self-Attention	manual→manual	98.56	97.29
(b)		manual→1-best	83.57	74.91
(c)		1-best→1-best	84.06	75.26
(d)		1-best→10-best	84.92	75.70
(e)		10-best→10-best	85.05	77.07
(f)	Neural ConfNet Classification	WCN→WCN	85.01	77.03
(g)	Lattice-SLU	WCN→WCN	86.09	78.78
(h)	WCN-Transformer	WCN→WCN	**87.91**	**81.14**
(i)	(-) w/o system act		86.69	79.56

- BLSTM+Self-Attention [81]：该系统基于 BLSTM 和自注意力机制，其编码结构类似第3.3节的输入编码器。对于 1-best→10-best 的情况，我们在测试的时候对每个候选句子预测得到语义概率分布，然后对预测结果按语音识别后验概率进行加权得到最终的结果。对于 10-best→10-best 的情况，我们在训练和测试时都对每个候选句子分别提取句向量，然后根据语音识别后验概率进行加权得到 10-best 的句向量。
- Neural ConfNet Classification [81]：Masumura 等 [81] 采用两层的网络结构对 WCN 进行编码，第一层是对 bin 进行编码的考虑语音识别后验概率的自注意力机制，可以获得 bin 向量；第二层是和 “BLSTM+Self-Attention” 类似的模型对 bin 向量编码，获取句向量。
- Lattice-SLU [73]：我们也应用了基于大规模预训练单向语言模型 GPT 的方法对 WCN 进行编码。Huang 等 [73] 利用该方法对词格进行编码，但其同样可以适用于 WCN。

在我们的实验中，上述基线系统和本章提出的方法都使用了基于分类的语义解码方法（STC）进行对比。观察表3–2中结果后，我们可以发现:

1. 对比 (a) 和 (b) 行，可以知道语音识别错误对于后续语义理解的性能会有很大的负面影响。
2. 对比 (b) 和 (c) 行，可以发现在语音识别结果上进行训练有利于提高口语理解系统的稳健性。

表 3–3 与之前在 DSTC-2 上的工作的比对

Table 3–3 Comparison with prior arts on the DSTC-2 dataset.

Models	$\mathbf{F}_1$ (%)
SLU2 [161]	82.1
CNN+LSTM_w4 [162]	83.6
CNN [163]	85.3
S2S-ATTN-PTR-NET [163]	85.8
WCN-Transformer + STC	**87.9**

3. 从 (d)、(e)、(f)、(g) 行可以看出，使用了包含语音识别解码信息更多的不确定性输入，口语理解的性能会更好。
4. Lattice-SLU 使用了大规模预训练单向语言模型 GPT，性能相比其他基线系统有较大的的提升。

通过利用 Transformer 对 WCN 和上一轮系统回复的语义表示进行联合编码，我们提出的方法可以取得最好的性能水平，并且大大超越之前在 DSTC-2 数据集上的最优结果（如表3–3所示）。上述基线系统中都没有加入对话历史信息，但在 (i) 行中显示我们的方法在不加入对话历史信息的情况下仍然可以超过所有的基线系统。而相应的 Lattice-SLU 虽然也使用了类似 BERT 的 GPT 预训练语言表示模型，但性能仍与我们的方法有明显差距，具体原因包括：它只考虑了单向的 Transformer 编码，以及它对 WCN 的利用也与我们的方法不同。

3.5.3 实验对比与分析

3.5.3.1 消融实验

本小节将对我们提出的 WCN-Transformer 编码结构进行一些对比性实验，使用到的语义解码方法包括语义元组分类器（STC) 和层级式解码（HD)，结果如表3–4所示。

基于时刻的位置编码的有效性：我们的方法在输入层中对每个词采用了基于 WCN 时刻的位置编码信息，为了验证它的有效性，我们忽略 WCN 时刻（即忽略 bin 的结构性）而是采用词序列的顺序位置编码。如表3–4-(b) 行所示，基于时刻的位置编码对性能有明显的影响。

融合语音识别后验概率的自注意力机制的有效性：在公式3–36中我们将语音识别后验概率注入到 Transformer 网络的自注意力机制计算中，为了验证它的有效性。我们在表3–4-(c) 行去掉了对语音识别后验概率的引入，结果表明其对性能的

表 3–4 消融实验与对比

Table 3–4 Ablation study on the test set.

	Model variations	STC		HD	
		$\mathbf{F}_1$ (%)	**Acc.**(%)	$\mathbf{F}_1$ (%)	**Acc.**(%)
(a)	WCN-Transformer	87.91	81.14	87.33	80.74
(b)	w/o bin position	87.57	80.63	86.64	79.51
(c)	w/o WCN Probability	87.15	80.07	86.47	79.05
(d)	$\mathbf{u} = \mathbf{h}_1$	87.75	81.02	86.99	80.23
(e)	w/o vector aggregations	87.75	80.75	86.96	79.96
(f)	GRU based value decoder	-	-	87.30	80.36
(g)	w/o BERT	86.12	77.11	86.01	78.23
(h)	w/o system act	86.69	79.56	86.18	79.07
(i)	w/o BERT and system act	85.85	77.41	85.06	77.27
(j)	Replace system act with system utterance	88.01	81.30	87.28	80.54

表 3–5 对比不同的语音识别后验概率融合方式

Table 3–5 Comparison of different methods to incorporate ASR posterior probabilities.

Model variations	STC	
	$\mathbf{F}_1$ (%)	**Acc.**(%)
WCN-Transformer	87.91±0.20	81.14±0.36
(-) None	87.15±0.02	80.07±0.02
(-) Multiply	87.65±0.10	80.94±0.17
(-) Naive plus	87.88±0.03	81.08±0.32

影响非常显著。另外，我们尝试了其它两种备选方法来融合语音识别后验概率：

- Multiply：该方法将语音识别后概率和注意力权值直接相乘，即 $e_{ij}^{l,h} = \frac{(\mathbf{W}_Q^{l,h}\mathbf{x}_i^l)^\top(\mathbf{W}_K^{l,h}\mathbf{x}_j^l)}{\sqrt{d_x/H}} p_j$。
- Naive plus：该方法将语音识别后概率和注意力权值直接相加，而没有公式3–36中调节幅度的参数，即 $e_{ij}^{l,h} = \frac{(\mathbf{W}_Q^{l,h}\mathbf{x}_i^l)^\top(\mathbf{W}_K^{l,h}\mathbf{x}_j^l)}{\sqrt{d_x/H}} + p_j$。

相关结果如表3–5所示，我们可以发现不加语音识别后验概率（即不确定性信息）的结果依然是最差的，其它的备选方法则相比本文选用的方法有小幅性能下降。

句向量提取模块的有效性：为了验证句向量提取模块，我们在表3–4-(d) 行中，我们只考虑起始特殊符号 `[CLS]` 的隐层向量作为句向量表示；另外在表3–4-(e) 行，我们忽略句向量提取模块中基于 WCN 结构的向量整合，直接利用公式3–37的

自注意力机制进行句向量提取。结果表明，利用 WCN 的结构信息提取句向量表示是有帮助的。

基于 Transformer 的值生成模型的有效性：在表3–4-(f) 行中，我们尝试把值生成模型的 Transformer 换成 GRU。对比结果表明基于 Transformer 的值生成模型会略微好于基于 GRU 的方法。

BERT 的有效性：在表3–4-(g) 行中，我们去除了利用 BERT 预训练模型对 WCN-Transformer 的参数进行初始化，而是采用随机初始化的策略，并使用 100 维的 Glove 词向量。实验结果显示去除 BERT 会对性能造成非常大的负面影响。

对话历史信息的有效性：此外在表3–4-(h) 行上，我们验证了对话历史信息的有效性。我们在输入端去掉了上一轮系统回复的语义表示，给语义理解性能造成了明显的下降。这说明了对话历史信息是非常有效的补充信息。此外，在表3–4-(j) 行上，我们将上一轮系统回复的语义表示替换成了上一轮系统回复的句子，但是性能没有特别明显的变化。这表明我们方法适合多种序列类型的上下文信息，模型的实用性比较好。

最后在表3–4-(i) 行上，我们同时去掉了 BERT 和对话历史信息，该结果可以与表3–2中的 BLSTM+Self-Attention 以及 Neural ConfNet Classification 两个基线系统公平比较。结果显示，我们的方法在同等条件下可以更好地利用 WCN，并取得更高的性能。同时，我们基于 WCN-Transformer 的方法相比 Neural ConfNet Classification 的模型运算速度在同等硬件环境（GPU 型号为 GeForce GTX 1080 Ti）下可以快大约 5 倍（一次性处理 32 句话的平均训练时间对比为 “41ms V.S. 250ms”）。

3.5.3.2 BLSTM 和 Transformer 编码器的对比

另外，我们会怀疑 Transformer 编码器是否本身就比 BLSTM 有很大的优势。于是，我们在最佳候选句子为输入的情况下对比了 BLSTM 和 Transformer 编码器，结果如表3–6所示。我们可以发现 Transformer 与 BLSTM 相比，在性能上没有明显优势，甚至会稍逊于 BLSTM。这反映出我们提出的方法的性能提升很大程度来源于对 WCN 的高效编码。

3.5.3.3 对基于生成模型的层级式解码算法的分析

从前面的实验结果中，我们可以发现基于生成模型的层级式解码（HD）的性能总是比分类方法 STC 低一些。但是理论上 HD 的方法利用了注意力机制和指针网络，应该具有更好的泛化能力，有助于解决词 OOV 的现象。为了进一步分析相应模型的泛化能力，我们从训练集里面随机保留一定比例的数据用于训练，但是

表 3–6 BLSTM 和 Transformer 编码器的对比

Table 3–6 Comparison of BLSTM and Transformer based encoders.

Models	**Train→Test**	**STC**	
		F_1(%)	**Acc.**(%)
BLSTM+Self-Attention	1-best→1-best	84.06	75.26
Transformer+Self-Attention	1-best→1-best	83.93	74.65

校验集合测试集不变，以此来模拟不同 OOV 程度的现象。并且，我们进一步将 act-slot-value 三元组分为两部分：在训练集中出现过的（seen），不在训练集中出现过的（unseen）。然后，针对性地计算这两类不同三元组的 F_1 得分。具体实验结果如表3–7所示。我们可以发现，在训练集比例比较小的时候（即 OOV 程度比较大的时候），HD 模型会明显优于 STC 的模型，并且在任何情况下 HD 模型对于 unseen 语义三元组的预测都优于 STC。

表 3–7 WCN-Transformer 模型在不同比例的训练集下得到的 F_1 值

Table 3–7 F_1 scores (%) of our WCN-Transformer SLU on the test set with varying training size.

Train size	**Models**	overall	seen	unseen
1%	+ STC	62.86	67.14	0.0
	+ HD	**71.57**	**77.00**	**29.69**
2%	+ STC	69.44	72.81	0.0
	+ HD	**76.75**	**80.89**	**30.34**
5%	+ STC	78.67	80.23	0.0
	+ HD	**81.49**	**83.07**	**35.28**
10%	+ STC	81.30	81.90	0.0
	+ HD	**82.91**	**83.80**	**22.9**
50%	+ STC	85.87	86.04	0.0
	+ HD	**86.29**	**86.56**	**7.59**
100%	+ STC	**87.91**	**88.17**	0.0
	+ HD	87.33	87.54	**4.76**

对于很多的语义槽，它们的取值范围非常庞大，难以被训练集覆盖完全，比如 *“food”*、*“name”* 以及 *“address”* 等。基于注意力机的指针网络通过从输入句子中 “拷贝” 新词可以很好地缓解这类问题。在给定 act 和 slot 的前提下，指针网络可以关注到相关的词上面。在图3–6中示例展示了指针网络的注意力机制对于不同的 act-slot 在输入句子上的 “关注度” 分布。我们可以发现 “*inform-slot*” 准确地关

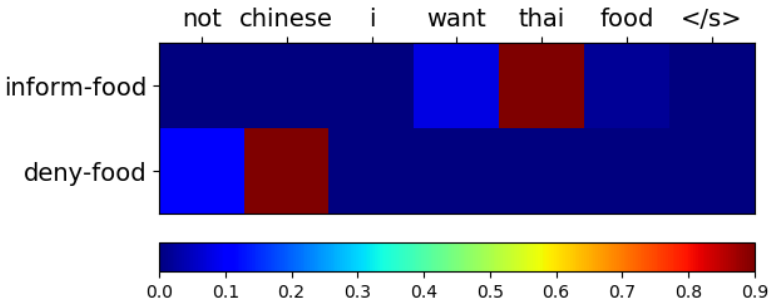


图 3–6 值生成器对于不同 act-slot 在输入句子上的注意力权重

Figure 3–6 Attention weights on input utterance of the value decoder with different act-slot pairs.

注在了 “thai” 上，而 “*deny-slot*” 这主要集中于 “chinese”。

3.6 本章小结

为了解决语音识别输出编码难的问题，本章先将词对齐式的任务型口语理解等价转换为非词对齐式的分类或者生成式问题，据此提出了一种高效的基于 Transformer 的词混淆网络和对话上下文的联合编码结构。该方法将词混淆网络扁平化，并利用 “基于时刻的位置编码” 和 “融合语音识别后验概率的自注意力机制” 将包含图结构和后验概率的不确定性信息编码到 Transformer 的隐层向量中。同时本章在该编码结构的基础上提出了一种层级式的语义解码器，可以有效缓解值 OOV 的情况。在 DSTC-2 数据集上的实验评估表明，该方法可以高效的利用 WCN 以及对话历史信息，并超越所有基线系统和以往公开结果。有限训练集的模拟实验也验证了层级式的语义解码器可以提升整体任务型口语理解的泛化能力。

第四章　基于语音识别错误纠正的精准口语理解

4.1　引言

建立在非词对齐的语义标注上，上一章介绍了对语音识别不确定性结果（词混淆网络）的编码方法。为了方便研究，上一章把口语理解的输入不确定性编码和语义解码进行了分拆，但相比对齐式的语义解码，非对齐式的语义解码往往难度更大。如第一章所描述的那样，由于语音识别错误的发生以及语音识别系统可能存在变动，要在语音识别结果上直接进行词对齐式的语义标注并不容易而且成本很高。我们依然面临着难以在语音识别结果上进行词对齐式语义标注的问题。

为了解决上述问题，本章提出了一种无需语音识别结果上词对齐语义标注的稳健口语理解训练方法。该方法只需要在人工转写句子上进行词对齐语义标注，然后利用从人工转写句子到语音识别结果上的输入自适应学习算法训练稳健精准的词对齐式口语理解模型。该输入自适应算法的核心贡献主要在以下几方面：

- 槽值纠错：受前端语音识别错误的影响，在以语音识别结果为输入时，词对齐的语义解码得到的语义槽值很可能包含了错误的文字。在本章中，我们提出一种基于语音识别错误纠正的槽值纠错算法，为模型预测的 “语义槽值对” 恢复相关的语音识别错误。
- 监督信号获取：以语音识别结果为输入时，我们是没有词对齐的语义标注的，无法直接获得口语理解模型的监督信号。但是，通过上述槽值纠错算法，语音识别错误恢复过后的语义槽值对可以与标注的语义槽值对进行对比，计算相关监督信号。
- 词对齐式口语理解模型的参数更新算法：即便有了上述的监督信号，但是槽值纠错的操作是符号运算、不可微的，我们无法直接运用一般求导计算梯度的参数更新方式。于是，基于策略梯度的强化学习算法被应用其中。

本章的结构如下：我们首先在第4.2节介绍结合槽值纠错模块的口语理解，包含基于人工转写文本数据的对齐式口语理解模型训练以及基于语音识别错误恢复的槽值纠错算法；在第4.3节介绍基于槽值纠错和强化学习算法的输入自适应训练，将对齐式口语理解模型应用到语音识别结果上；第4.4节是实验部分，包括实验设置、主要结果与实验分析；第4.5节是本章小结。

4.2 结合槽值纠错模块的口语理解

在本章中，我们的目标是把人工转写句子上的对齐式语义标注迁移到语音识别结果上。同时，为了方便构建词对齐式的序列标注模型，在本章中采用的语音识别结果是最佳候选句子。理论上，结合前一章的 WCN-Transformer 对 WCN 进行序列编码也是完全可行，后续相关方法结合的尝试我们打算放在未来的研究工作中，在本文暂不继续探索。

表 4–1 包括用户句子（人工转写文本、语音识别结果）以及相应语义标注的数据样例

Table 4–1 An example of user utterance (manual transcription and ASR hypothesis) and its semantic annotations.

名称定义	符号	样例
人工转写句子	$\hat{\boldsymbol{x}}$	我要去苏州不去上海了
语音识别结果	$\boldsymbol{x}$	喔去宿州不去商海
$\hat{\boldsymbol{x}}$ 的对齐语义标注	$\hat{\boldsymbol{y}}^S$	我 [O] 要 [O] 去 [O] 苏 [B-inform-dest] 州 [I-inform-dest] 不 [O] 去 [O] 上 [B-deny-dest] 海 [I-deny-dest] 了 [O]
语义槽值对	$\boldsymbol{y}$	inform(dest= 苏州); deny(dest= 上海)

在介绍本章的方法之前，先了解一下当前问题面临的数据样例，如表4–1所示。具体地，我们记人工转写句子为 $\hat{\boldsymbol{x}} = (\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_{|\hat{\boldsymbol{x}}|})$，记语音识别结果的最佳候选句子为 $\boldsymbol{x} = (x_1, x_2, \cdots, x_{|\boldsymbol{x}|})$，记在人工转写句子上的对齐式语义标注为 $\hat{\boldsymbol{y}}^S = (\hat{y}_1^S, \hat{y}_2^S, \cdots, \hat{y}_{|\hat{\boldsymbol{x}}|}^S)$。根据我们对对齐式口语理解任务的定义，对齐式的语义标签 $\hat{y}_i^S$ 都是基于 IOB 的格式（比如 *O*, *B*-inform-dest, *B*-deny-dest）。给定人工转写句子以及相应的对齐式语义标注，我们可以轻易获取用户要表达的真实语义，即相关的一系列语义槽值对 $\boldsymbol{y}$。（在第2.2节中我们采用 $\boldsymbol{y}^C$ 表示语义槽值对组成的语义形式，而由于本章不涉及意图识别任务并且为了公式的简洁性，我们直接使用 $\boldsymbol{y}$ 来代替 $\boldsymbol{y}^C$。）

口语理解的任务就是给定输入的语音识别结果 $\boldsymbol{x}$，解析出真实的用户语义表示形式 $\boldsymbol{y}$。但是如果直接将在人工转写句子以及相应语义标注上训练的序列标注模型（请见第4.2.1小节）应用到语音识别结果上的话，很容易抽取出带语音识别错误的值，比如表4–1中的“宿州”。为此，我们在第4.2.2小节提出了一种基于语音识别错误恢复的槽值纠错算法。

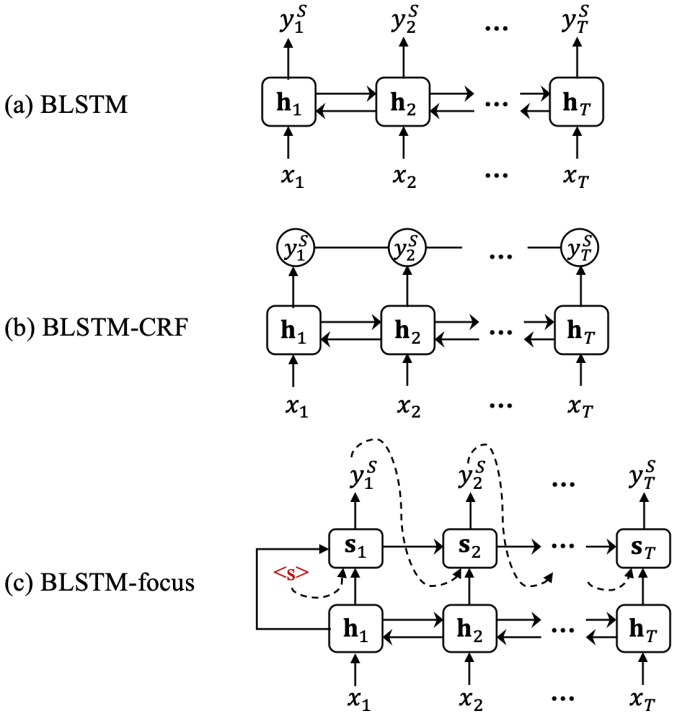


图 4–1 三个基本的语义槽标记模型

Figure 4–1 Three typical models for slot tagging.

4.2.1 词对齐式口语理解模型

首先我们忽略语音识别错误，介绍一些基本的适用于词对齐式口语理解的序列标注模型，也称语义槽标记（slot tagging）模型。然后在后续小节中介绍将相关序列标注模型应用到语音识别结果上的错误恢复模块以及输入自适应训练方法。

对于输入的人工转写句子，我们依然采用双向 LSTM（BLSTM）[34, 155] 进行编码。首先，每个词都会被映射成一个词向量，即 $\hat{\mathbf{x}}_i = \mathbf{W}_{\text{in}}\mathbf{o}(\hat{x}_i)$，其中 $\mathbf{W}_{\text{in}} \in \mathbb{R}^{d_e \times |\mathcal{V}_{\text{in}}|}$ 是一个词向量矩阵，$\mathbf{o}(\hat{x}_i) \in \mathbb{R}^{|\mathcal{V}_{\text{in}}|}$ 是一个独热向量（one-hot vector），$\mathcal{V}_{\text{in}}$ 表示输入字典，d_e 为词向量维度，$\hat{\mathbf{x}}_i \in \mathbb{R}^{d_e}$。对于输入句子 $\hat{\boldsymbol{x}} = (\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_{|\hat{\boldsymbol{x}}|})$，BLSTM 可以进行循环编码并获得每个词对应的隐层向量（计算过程见附录A.1）：

$$(\mathbf{h}_1, \cdots, \mathbf{h}_{|\hat{\boldsymbol{x}}|}) \leftarrow \text{BLSTM}_{\Theta_{\text{enc}}}(\hat{\mathbf{x}}_1, \cdots, \hat{\mathbf{x}}_{|\hat{\boldsymbol{x}}|}) \tag{4–1}$$

其中，Θ_{enc} 是该 BLSTM 包含的所有参数，$\mathbf{h}_i \in \mathbb{R}^{2d_h}$ 是双向隐层向量，d_h 表示 LSTM 的隐层向量维度。

基于上述 BLSTM 句子编码，有三种经典的语义槽标记模型可以应用，分别是 BLSTM，BLSTM-CRF，BLSTM-focus。它们大致模型结构如图4–1所示。

4.2.1.1 BLSTM

基于 BLSTM 的语义槽标记模型直接在上述 BLSTM 的所有隐层向量上叠加了一个线性输出层，即

$$p(\hat{\boldsymbol{y}}^S|\hat{\boldsymbol{x}}) = \prod_{i=1}^{|\hat{\boldsymbol{x}}|} p(\hat{y}_i^S|\mathbf{h}_i) = \prod_{i=1}^{|\hat{\boldsymbol{x}}|} \text{softmax}_{\hat{y}_i^S}(\mathbf{W}_1\mathbf{h}_i + \mathbf{b}_1) \tag{4–2}$$

其中 $\mathbf{W}_1 \in \mathbb{R}^{|\mathcal{V}_{\text{tag}}| \times 2d_h}$，$\mathbf{b}_1 \in \mathbb{R}^{|\mathcal{V}_{\text{tag}}|}$ 是输出层模型参数，$\mathcal{V}_{\text{tag}}$ 是所有输出标签的字典。*softmax* 函数过完后会得到在所有语义标签上的概率分布，$\text{softmax}_{\hat{y}_i^S}(.)$ 则是指该分布上对应 $\hat{y}_i^S$ 标签的概率。

4.2.1.2 BLSTM-CRF

基于条件随机场（Conditional Random Field，CRF）的输出层则可以考虑相邻时刻的输出标签之间的关系，并对输出序列进行联合解码 [46, 47, 164]。输出序列的后验概率计算过程为

$$\psi(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}^S) = \sum_{i=1}^{|\hat{\boldsymbol{x}}|} ([\mathbf{A}]_{\text{id}(\hat{y}_{i-1}^S),\text{id}(\hat{y}_i^S)} + [\mathbf{W}_1\mathbf{h}_i + \mathbf{b}_1]_{\text{id}(\hat{y}_i^S)}) \tag{4–3}$$

$$p(\hat{\boldsymbol{y}}^S|\hat{\boldsymbol{x}}) = \frac{\exp(\psi(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}^S))}{\sum_{\hat{\boldsymbol{y}}^{S\prime}} \exp(\psi(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}^{S\prime}))} \tag{4–4}$$

其中 $\text{id}(\hat{y}_i^S) \in \{1, 2, \cdots, |\mathcal{V}_{\text{tag}}|\}$ 表示标签 $\hat{y}_i^S$ 在输出词汇表 $\mathcal{V}_{\text{tag}}$ 中的序号；$\mathbf{A} \in \mathbb{R}^{|\mathcal{V}_{\text{tag}}| \times |\mathcal{V}_{\text{tag}}|}$ 是标签转移矩阵，它的每个元素 $[\mathbf{A}]_{j,k}$ 表示的是相邻时刻从第 j 个标签转移到第 k 个标签的转移置信度。输入序列 $\hat{\boldsymbol{x}}$ 和输出序列 $\hat{\boldsymbol{y}}^S$ 共同出现的置信度是由标签转移得分以及线性输出的标签得分加和得到。最终，这些置信度被归一化为后验概率。

4.2.1.3 BLSTM-focus

同样为了考虑输出标签之间的时序依赖关系（比如 “到达城市” 经常出现在 “出发城市” 之后），一些基于编码器-解码器（encoder-decoder）的网络结构被应用于语义槽标记任务 [43, 51, 53, 165]。而我们提出了一种 “聚焦” 机制（focus mechanism），利用一个单向的 LSTM 解码器来建模输出标签的时序依赖关系。在第 i 时刻的解码器隐层向量由此计算，

$$\mathbf{s}_i = \text{f}_{\text{LSTM}}(\mathbf{h}_i \oplus \hat{\mathbf{y}}_{i-1}^S, \mathbf{s}_{i-1}) \tag{4–5}$$

其中 $\hat{\mathbf{y}}^S_{i-1} = \mathbf{W}_{\text{tag}}\mathbf{o}(\hat{y}^S_{i-1})$ 是上一时刻预测得到的输出标签的向量表示，且 $\mathbf{W}_{\text{tag}} \in \mathbb{R}^{d_e \times |\mathcal{V}_{\text{tag}}|}$ 是一个标签向量矩阵。另外，$\mathbf{s}_0 = \overleftarrow{\mathbf{h}}_1$，$\hat{y}^S_0$ 则表示起始符号 "<s>"。然后，我们按如下步骤预测最终的输出序列：

$$p(\hat{\boldsymbol{y}}^S|\hat{\boldsymbol{x}}) = \prod_{i=1}^{|\hat{\boldsymbol{x}}|} p(\hat{y}^S_i|\hat{\boldsymbol{y}}^S_{<i}, \hat{\boldsymbol{x}}) = \prod_{i=1}^{|\hat{\boldsymbol{x}}|} \text{softmax}_{\hat{y}^S_i}(\mathbf{W}_2\mathbf{s}_i + \mathbf{b}_2) \tag{4–6}$$

其中 $\mathbf{W}_2 \in \mathbb{R}^{|\mathcal{V}_{\text{tag}}| \times d_h}$，$\mathbf{b}_2 \in \mathbb{R}^{|\mathcal{V}_{\text{tag}}|}$ 是相关参数。与 *BLSTM-CRF* 相比，该方法具有对长距离的输出标签依赖进行建模的能力。

最后，我们根据如下损失误差函数在人工转写句子的数据（$\mathcal{D}_{\text{tscp}}$）上训练相关序列标注模型（记所有可训练的参数为 Θ）：

$$\mathcal{L}_{\text{tscp}}(\Theta) = -\sum_{\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}^S \in \mathcal{D}_{\text{tscp}}} \log p(\hat{\boldsymbol{y}}^S|\hat{\boldsymbol{x}}; \Theta) \tag{4–7}$$

经过模型预测得到输出序列后，我们可以通过后处理提取出相应的语义槽值对，比如在表4–1中已知输入输出序列 $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}^S)$ 可以得到 $\boldsymbol{y}$。

4.2.2 基于语音识别错误恢复的槽值纠错算法

但是，当我们把上述模型直接用在语音识别结果上还有两个缺陷：1）语音识别结果上可能包含识别错误，会产生错误的语义槽值对；2）训练和测试的输入不匹配。在本小节我们先解决第一个缺陷，而第二个缺陷的解决方法会在第4.3节中继续阐述。

为了解决预测的 act-slot-value 三元组包含语音识别错误的问题，我们提出了一种基于语音识别错误恢复的槽值纠错算法。该算法依赖于领域本体和发音字典，根据预测的槽值和本体中所有候选槽值之间的文本、发音相似度进行纠正。在任务型对话系统中，每个语义槽的所有可能取值在很大程度上是可以从后端数据库中获取的，比如有效的地址、电话、商家名称等等。其中具体的槽值纠错类型分为三部分（具体流程如算法4–1所示）：

1. **语义槽置换**：第一类是语义槽错误，即值是正确的，但是语义槽与之不匹配。这种情况，需要通过值到语义槽的反向映射表（*val2slot*）查询与该值匹配的唯一的语义槽，然后替换掉原来的语义槽。比如，在天气领域中，"长沙" 的语义槽有可能会被错误预测为 "区县"，实际上应该是 "城市"。
2. **值置换**：第二类是值中包含语音识别错误，也是最主要的一类错误。如果一个预测值不在相应语义槽的本体候选值中，则被认为可能存在语音识别错误。首先，无论是预测值还是领域本体中相应语义槽的所有候选值，都

算法 4–1 基于语音识别错误恢复的槽值纠错

Input: 语义槽标记模型预测得到的语义三元组集合 $\boldsymbol{y}$；领域本体 $\mathcal{O}$；发音字典 $\mathcal{V}_{\text{pron}}$；相似度得分阈值 thr；

Output: 错误恢复后的语义三元组集合 $\tilde{\boldsymbol{y}}$；

```
val2slot ← get_val2slot_mapping(O) ;          // 获取从值到语义槽的反向映射表
ỹ ← [];
for (act, slot, val) in y do
    if val in O[slot] then
        ỹ.append((act, slot, val));
    else
        if val in val2slot and len(val2slot[val]) = 1 then
            ỹ.append((act, val2slot[val], val)) ;          // 语义槽置换
        else
            max_sim, new_val ← arg max(get_sim(val, O[slot]; V_pron)) ;   // 根据发音字典 V_pron 找到领域本体中最接近的槽值
            if max_sim ≥ thr then
                ỹ.append((act, slot, new_val)); // 值置换，否则作为无效语义槽值删除
            end
        end
    end
end
return ỹ;
```

可以轻易获得它们的词序列（或者字序列）以及对应的发音单元序列（可以通过发音字典获取）。然后，我们通过计算预测值的相应序列和每个候选值的相关序列之间的相似度（可以选择序列编辑距离、基于词袋向量的余弦距离等），寻找与该预测值最相似的候选值以及它们的相似度。如果相似度超过一定的阈值（比如 0.5），则选用该候选值替换预测值。

3. **无效语义槽值删除：**如果上述最大相似度没有大于一定的阈值，则会被认为无法纠错。于是，这种情况会删除相应的语义槽值对。

我们在此形式化地描述一下对预测值和候选值之间的相似度计算过程。首先，对于领域本体 $\mathcal{O}$，它记录了当前领域中每个语义槽的所有可能的取值。对于某语义槽 s，它的所有候选取值列表为 $\mathcal{O}[s]$。记当前需要错误恢复的语义三元组为 (a, s, v)，

我们需要计算 v 和每一个候选值之间的某种相似度：

$$\text{get\_sim}(v, \mathcal{O}[s]; \mathcal{D}_{\text{pron}}) = \{\text{sim}(v, v'; \mathcal{V}_{\text{pron}}) | \forall v' \in \mathcal{O}[s]\} \tag{4–8}$$

其中 $\mathcal{V}_{\text{pron}}$ 是发音字典，包括了每个字词对应的发音单元序列。基于文本和发音特征的相似度函数 sim 可以有多种实现方法：编辑距离和余弦距离。

1）编辑距离：该距离由基于文本序列的编辑距离和基于发音序列的编辑距离两部分组成。

$$\text{sim}(v, v'; \mathcal{V}_{\text{pron}}) = -\lambda \cdot \text{editDist}(v, v') - (1 - \lambda) \cdot \text{editDist}(\mathcal{V}_{\text{pron}}(v), \mathcal{V}_{\text{pron}}(v')) \tag{4–9}$$

其中 λ 是两者的平衡系数，$\mathcal{V}_{\text{pron}}(v)$ 是表示获取输入文本 v 的发音序列。比如，$\mathcal{V}_{\text{pron}}$(“上海”) 可以得到相应的发音序列 “*sh ang h ai*”。

2）余弦距离：由于一个语义槽的候选取值可能数量很庞大，依次计算编辑距离是非常耗时的，于是我们需要一种可以并行处理的解决方案。恰好，基于向量计算的余弦距离可以通过向量和矩阵的运算实现并行处理。

余弦距离计算用到的是基于 N 元组（n-gram）的词袋特征向量。一般，对于一个词（字）序列 $w = (w_1, \cdots, w_T)$，记它的 n-gram 集合为 $\text{Ngram}(w, n) = \{(w_i, \cdots, w_{i+n-1}) \mid i = \{1, \cdots, T - n + 1\}\}$。通过统计领域本体 $\mathcal{O}$ 中所有候选取值的 n-gram，我们可以获得一个 n-gram 词汇表，记为 $\text{Ngram}(\mathcal{O}, n)$。

对于预测值 v，我们可以得到一个 0-1 向量 $\mathbf{d}'(v) = (d_1^v; \cdots; d_L^v) \in \mathbb{R}^L$，其中向量维度为 $L = |\text{Ngram}(\mathcal{O}, n)|$，且向量每一维的值表示该维对应的 n-gram 是否出现在预测值 v 中，即 $d_j^v = \mathbb{I}\{\text{Ngram}(\mathcal{O}, n)_j \in \text{Ngram}(v, n)\}$。最后，我们把该向量归一化为一个单位向量，即 $\mathbf{d}(v) = \mathbf{d}'(v)/\|\mathbf{d}'(v)\|$.

假设语义槽 s 在领域本体中总共有 M 个候选值，记为 $\mathcal{O}[s] = (v'^1, \cdots, v'^M)$。对于每个候选值 v'，都可以按照上述方法获取它的特征向量 $\mathbf{d}(v')$。于是，预测值和每个候选值之间的文本余弦距离为：

$$\text{cosine}(v, v') = \mathbf{d}(v')^\top \mathbf{d}(v) \tag{4–10}$$

预测值和所有候选值之间的文本余弦距离为：

$$\mathbf{sim}_{\text{word}}(v, \mathcal{O}[s]) = \mathbf{D}(\mathcal{O}[s])^\top \mathbf{d}(v) \tag{4–11}$$

其中 $\mathbf{D}(\mathcal{O}[s]) \in \mathbb{R}^{L \times M}$ 的第 k 列是第 k 个候选值的特征向量 $\mathbf{d}(v'^k)$。类似地，我们还可以对它们的发音序列计算余弦距离，得到 $\mathbf{sim}_{\text{pron}}(v, \mathcal{O}[s]; \mathcal{V}_{\text{pron}})$，并最后综合两者：

$$\text{get\_sim}(v, \mathcal{O}[s]; \mathcal{V}_{\text{pron}}) = \lambda \cdot \mathbf{sim}_{\text{word}}(v, \mathcal{O}[s]) + (1 - \lambda) \cdot \mathbf{sim}_{\text{pron}}(v, \mathcal{O}[s]; \mathcal{V}_{\text{pron}}) \tag{4–12}$$

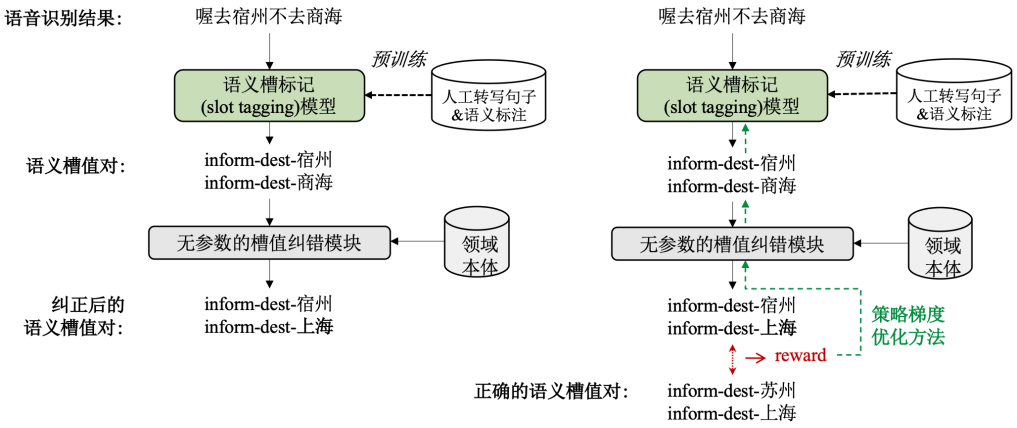


图 4–2 基于槽值纠错的口语理解框架

Figure 4–2 Spoken language understanding with value error recovery.

4.3 基于槽值纠错和强化学习的输入自适应训练

前面基于槽值纠错算法可以在测试的时候恢复槽值中包含的语音识别错误（如图4–2左半部分所示），但是语义槽标注模型在训练和测试阶段输入不匹配的问题仍然有待解决。于是我们提出了一种基于槽值纠错和强化学习的输入自适应训练方法（如图4–2右半部分所示），该方法利用槽值纠错后的语义表示与正确答案对比得到“收益（reward）”，并以此为监督信号，结合基于策略梯度的强化学习方法对语义槽标记模型进行优化。

该方法通过无参数的语义槽值纠错模块来指导语义槽标记模型的输入自适应训练，它有两点好处：1）消除了语义槽标记模型的训练和测试的不匹配问题；2）不需要在语音识别结果上有词对齐的语义标注。同时，它趋向于学习到一个更适合语义槽值纠错的语义槽标记模型。为了减小模型的搜索空间，我们先使用人工转写句子以及相应的对齐式标注预训练该语义槽标记模型（如公式4–7所示）。此外，我们利用基于策略梯度的强化学习方法最大化相应的期望“收益（reward）”。

我们将语音识别结果 $\boldsymbol{x}$ 和正确的语义槽值对集合 $\boldsymbol{y}$ 组成样本对，即 $\boldsymbol{D}_{\text{hyp}} = \{(\boldsymbol{x}, \boldsymbol{y})\}$，用于基于强化学习的输入自适应训练。给定语音识别结果 $\boldsymbol{x}$，我们可以在语义槽标记模型 $p(\boldsymbol{y}^S|\boldsymbol{x};\boldsymbol{\Theta})$ 的输出分布上进行序列解码，比如对于前文第4.2.1小节提到的 BLSTM、BLSTM-CRF 模型可以采用 Viterbi 算法，对于 BLSTM-focus 模型可以采用集束搜索（beam search）的方法。在解码结果上，我们采样前 $\boldsymbol{K}$ 个最有可能的输出序列，即 $\boldsymbol{y}^{S1}, \cdots, \boldsymbol{y}^{SK}$；随后，通过槽值纠错模块后，相应地得到 $\boldsymbol{K}$ 个预测的语义槽值对集合 $\{\tilde{\boldsymbol{y}}^k\}_{k=1}^K$。对于每一个预测的语义槽值对集合，具体的

算法 4–2 基于槽值纠错和强化学习的输入自适应训练

Input: 人工转写文本和对齐式语义标注构成的训练数据 $\mathcal{D}_{\text{tscp}} = \{(\hat{\boldsymbol{x}}, \hat{\boldsymbol{o}})\}$；语音识别结果和非对齐标注（语义槽值对集合）构成的训练数据 $\mathcal{D}_{\text{hyp}} = \{(\boldsymbol{x}, \boldsymbol{y})\}$；对齐式语义槽标记模型的损失函数 $L_{\text{tscp}}(\cdot)$；期望收益函数 $\psi(\cdot)$；学习率 η_1，η_2；槽值纠错模块；

Output: 语义槽标记模型参数 Θ；

随机初始化参数 Θ;
repeat // 预训练阶段
  从 $\mathcal{D}_{\text{tscp}}$ 中采样一个样本 $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}^S)$;
  更新模型参数：$\Theta \leftarrow \Theta - \eta_1 \nabla_\Theta L_{\text{tscp}}(\Theta)$;
until 模型收敛;
repeat // RL 训练阶段
  从 $\mathcal{D}_{\text{hyp}}$ 中采用 $(\boldsymbol{x}, \boldsymbol{y})$;
  将 $\boldsymbol{x}$ 输入到语义槽标记模型，即 $p(\boldsymbol{y}^S|\boldsymbol{x}; \Theta)$，通过在解码结果上采样以及运行槽值纠错算法获取 K 个预测的语义形式 $\tilde{\boldsymbol{y}}^1, ..., \tilde{\boldsymbol{y}}^K$;
  for $k = 1, ..., K$ **do**
    根据公式4–13计算收益 $R(\boldsymbol{x}, \boldsymbol{y}, \tilde{\boldsymbol{y}}^k)$;
  end
  根据公式4–17计算策略梯度 $\nabla_\Theta \psi(\boldsymbol{x}, \boldsymbol{y}; \Theta)$;
  更新模型参数：$\Theta \leftarrow \Theta + \eta_2 \nabla_\Theta \psi(\boldsymbol{x}, \boldsymbol{y}; \Theta)$;
  从 $\mathcal{D}_{\text{tscp}}$ 中采样 $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}^S)$ // 伴随性训练;
  更新模型参数：$\Theta \leftarrow \Theta - \eta_1 \nabla_\Theta L_{\text{tscp}}(\Theta)$;
until 模型收敛;

reward 综合考虑了语义槽值对级别和句子整体级别的评价指标：

$$\mathcal{R}(\boldsymbol{x}, \boldsymbol{y}, \tilde{\boldsymbol{y}}^k) = \mathcal{R}_{\text{triplet}} + R_{\text{utt}} \tag{4–13}$$

$$\mathcal{R}_{\text{triplet}} = 1 - \frac{|\boldsymbol{y} - \tilde{\boldsymbol{y}}^k| + |\tilde{\boldsymbol{y}}^k - \boldsymbol{y}|}{|\boldsymbol{y}|} \tag{4–14}$$

$$\mathcal{R}_{\text{utt}} = \mathbb{I}\{\boldsymbol{y} = \tilde{\boldsymbol{y}}^k\} \tag{4–15}$$

其中 $|\boldsymbol{y}|$ 表示其中语义槽值对的个数，$|\boldsymbol{y} - \tilde{\boldsymbol{y}}^k|$ 代表的是没有被预测出来的语义槽值对个数（false-positive），$|\tilde{\boldsymbol{y}}^k - \boldsymbol{y}|$ 表示的是预测结果中错误的语义槽值对个数（false-negative）；$-\infty < \mathcal{R}_{\text{triplet}} \leq 1$，$R_{\text{utt}} \in \{0, 1\}$。

我们采用策略梯度下降法，最大化 reward 的期望值并计算相关的策略梯度：

$$\psi(\boldsymbol{x}, \boldsymbol{y}; \Theta) = \mathbb{E}[\mathcal{R}(\boldsymbol{x}, \boldsymbol{y}, \tilde{\boldsymbol{y}})] \approx \frac{1}{K} \sum_{k=1}^{K} \mathcal{R}(\boldsymbol{x}, \boldsymbol{y}, \tilde{\boldsymbol{y}}^k) \tag{4–16}$$

$$\nabla_{\Theta}\psi(\boldsymbol{x},\boldsymbol{y};\Theta)=\frac{1}{K}\sum_{k=1}^{K}\Big[\mathcal{R}(\boldsymbol{x},\boldsymbol{y},\tilde{\boldsymbol{y}}^{k})-B(\boldsymbol{x},\boldsymbol{y})\Big]\nabla_{\Theta}\log p(\boldsymbol{y}^{Sk}|\boldsymbol{x};\Theta) \tag{4–17}$$

其中 $B(\boldsymbol{x},\boldsymbol{y})=\frac{1}{K}\sum_{k=1}^{K}\mathcal{R}(\boldsymbol{x},\boldsymbol{y},\tilde{\boldsymbol{y}}^{k})$ 是用于降低训练方差的基线 [166]。

最后，为了让该输入自适应训练更稳定，我们依然把人工转写数据（$\boldsymbol{D}_{\text{tscp}}$）加入进来更新语义槽标记模型（伴随性训练）。完整训练过程的伪代码请看算法4–2。

4.4 实验与分析

4.4.1 实验数据与设置

4.4.1.1 数据集构建

为了验证本章提出的方法，我们构建了一个中文口语理解的数据集。在思必驰科技有限公司（AISPEECH）[①]提供的真实口语对话系统的日志文件中，我们采集了涉及大量用户的人机交互语句，并在人工转写文本上进行了词对齐式的语义标注。该数据集包括四个领域：地图导航（**map**）、音乐搜索（**music**）、天气预报（**weather**）以及影视搜索（**video**）。每个领域的用户句子都是从用户和一个多领域口语对话系统的真实交互中获取的。

在 2019 年第 21 届多模态交互国际会议上，我们以此数据集组织了第一届中文口语理解竞赛（CATSLU）[②]。相关数据集的统计信息如表4–2所示，其中包括：用户数量，训练集、校验集和测试集的句子数量，语义槽数量，以及语音识别的字错误率（CER）。我们可以发现每个领域涉及的真实用户[③]数量是非常多的。另外我们对于天气和影视领域提供了非常有限的训练数据，以模拟数据稀缺的情况。

与其他公开的口语理解数据集的比较：与 DSTC-2&3 [④]相比，CATSLU 数据采集自真实场景，拥有更大的领域本体，出现的槽值对数量非常庞大（比如音乐领域涉及几千首歌曲）；而 DSTC-2&3 数据是通过模拟场景采集的，偏向于集中在少数的一些槽值对上。其次，DSTC-2&3 数据的语义标注并不是词对齐的。ATIS [⑤]，SNIPS [⑥]，MIT Corpus [⑦]等对齐式语义理解数据均是采用的人工转写文本（甚至是自然语言文本）数据，无法用来验证口语理解模型对语音识别错误的稳健性。类

① http://www.aispeech.com/index.php
② https://sites.google.com/view/CATSLU
③ 用户信息已经去隐私化。
④ http://camdial.org/~mh521/dstc
⑤ https://github.com/yvchen/JointSLU
⑥ https://github.com/snipsco/nlu-benchmark/tree/master/2017-06-custom-intent-engines
⑦ https://groups.csail.mit.edu/sls/downloads

表 4–2 CATSLU 数据集的统计信息

Table 4–2 Statistics of CATSLU dataset.

领域	用户数量	句子数量			语义槽数量	CER (%)
		训练集	校验集	测试集		
map	1788	5093	921	1578	24	20.66
music	268	2189	381	676	20	21.91
weather	276	341	378	2660	22	33.14
video	227	205	195	1641	28	22.90

似公开的中文口语理解数据集有：SMP2017-ECDT 评测任务一数据集[①]只涉及不到 4000 句话，同样不包含语音识别结果，且只有句子类别标注。据我们所知道的，CATSLU 数据集是目前公开的最大的中文口语理解数据集。和 DSTC-2&3 一样，该数据集也提供了一个完备的领域本体，即数据集中涉及的语义槽值对都是包含在该领域本体内的。这也满足了本章槽值纠错模块的前提假设。

4.4.1.2 参数设置

首先，为了避免中文分词模块的错误传递 [167]，我们在中文字级别上进行语义槽标记任务建模。对于预训练字向量，我们在中文维基数据上训练了一个基于 LSTM 网络的双向语言模型（类似于 ELMo[101]），以该语言模型的 200 维字向量作为初始化。

本章涉及的基线系统和提出的方法都以 BLSTM 为句子编码器，LSTM 的隐层大小均为 256，且为一层。除了预训练好的字向量部分，其他参数均以 $(-0.2, 0.2)$ 之间的均匀分布随机初始化。在训练阶段，非循环层之间设置 0.5 的 Dropout 率。我们依然采用 Adam [160] 作为优化器，设置学习率 η_1=1e-3 和 η_2=5e-4 并在训练过程中保持不变，梯度最大范数的修剪阈值为 5。在强化学习阶段，采样数 K 设置为 10。而在解码阶段，BLSTM-focus 模型采用了 beam search 且 beam 数量为 5。评测指标采用的是 *act-slot-value* 三元组的 F_1 得分以及句子级的准确率（即要求一个句子预测的 *act-slot-value* 三元组完全正确）。所有的模型都被训练 50 轮，并保存在校验集上性能表现最好的模型参数。为了更好地缓解数据稀疏问题，我们还利用了额外的字典特征。关于实验结果的显著性检测，我们采用了 McNemar's test 方法。

① http://ir.hit.edu.cn/SMP2017-ECDT

表 4–3 各方法主要结果（带词库特征），包括每个领域的 F_1 值/句子准确率。我们的方法显著性优于最好的基线系统会被标记为 † ($p < 0.05$) 和 ‡ ($p < 0.01$)

Table 4–3 Main results with lexicon features. F_1-score(%)/joint-accuracy(%) on the test set of each domain are reported.

Models	Train	Test	map	music	video	weather	Avg.
HD	hyp	hyp	87.8/84.2	90.5/82.1	88.7/76.1	89.6/82.4	89.2/81.2
BLSTM-focus	tscp	tscp	96.4/93.8	97.6/93.3	94.1/85.6	95.5/90.6	95.9/90.8
BLSTM-focus	tscp	hyp	89.0/84.9	92.8/84.8	91.5/80.9	92.6/86.7	91.5/84.3
UA	tscp+hyp	hyp	88.5/85.2	91.8/83.6	91.2/81.2	91.8/84.9	90.9/83.7
DA-GEN	tscp+hyp	hyp	88.9/85.4	92.2/84.6	**92.0**/81.4	93.1/87.1	91.5/84.6
DA-ALIGN	tscp+hyp	hyp	89.1/85.5	93.1/85.7	91.5/80.8	93.1/87.0	91.7/84.7
Proposed	tscp	hyp	89.0/85.3	93.1/85.7	91.9/81.6	93.1/87.6	91.8/85.0
	tscp+hyp	hyp	**90.0/86.7**‡	**93.8/87.0**†	**92.0/82.0**	**93.4/87.7**†	**92.3/85.8**‡

4.4.1.3 基线系统

本章提出的方法将与如下基线系统进行对比：

- **HD**: 这是前一章提出的适用于非对齐语义标注的层级式口语理解模型。该模型可以直接在数据 $\mathcal{D}_{\text{hyp}}$ 上进行训练，但是无法利用词对齐语义标注。
- **LSTM-focus**: 在第4.2.1小节提到的基于 LSTM-focus 的语义槽标记模型，该模型仅在 $\mathcal{D}_{\text{tscp}}$ 上进行训练。
- **UA**: 无监督输入自适应训练（Unsupervised Adaptation） [111] 试图利用在人工转写句子和语音识别结果上的语言模型任务将语义槽标记模型从人工转写文本迁移到语音识别结果上。该方法以上述 LSTM-focus 为基础模型。
- **DA**: 数据扩充（Data Augmentation）方法被来用给语音识别结果生成相应的词对齐的伪标签，然后这些伪样本可以被用来训练更稳健的语义槽标记模型。(1) GEN：第一种方式是将预训练好的 LSTM-focus 语义槽标记模型应用在语音识别结果上做预测，将预测得到的标签作为伪标签。(2) ALIGN：以最小编辑距离为目标，将人工转写句子和语音识别结果进行字对齐，然后把人工转写文本中字的语义标签传递给对齐的语音识别字上。该方法同样以上述 LSTM-focus 为基础模型。

在所有基线系统和我们提出的方法上，都会应用相同的槽值纠错模块。

表 4–4 各方法主要结果（无词库特征），包括每个领域的 F_1 值/句子准确率。我们的方法显著性优于最好的基线系统会被标记为 † ($p < 0.05$) 和 ‡ ($p < 0.01$)

Table 4–4 Main results without lexicon features. F_1-score(%)/joint-accuracy(%) on the test set of each domain are reported

Models	**Train**	**Test**	**map**	**music**	**video**	**weather**	**Avg.**
HD	hyp	hyp	87.9/83.5	87.0/74.1	84.0/64.9	89.2/80.0	87.0/75.6
BLSTM-focus	tscp	tscp	96.4/93.9	96.3/89.4	92.8/81.6	94.7/88.7	95.0/88.4
BLSTM-focus	tscp	hyp	89.4/86.1	92.2/83.6	90.2/77.8	92.4/85.7	91.0/83.3
UA	tscp+hyp	hyp	89.3/86.4	91.9/83.6	90.0/78.4	91.8/85.0	90.8/83.3
DA-GEN	tscp+hyp	hyp	88.7/85.7	91.4/82.5	90.3/78.2	92.2/85.8	90.6/83.1
DA-ALIGN	tscp+hyp	hyp	89.3/85.8	**92.3**/83.6	90.6/77.6	91.8/85.1	91.0/83.0
Proposed	tscp	hyp	89.5/86.3	91.8/82.8	90.8/79.0	92.2/86.1	91.1/83.6
	tscp+hyp	hyp	**89.6/86.8**†	92.2/**83.7**	**91.2/79.7**‡	**92.6/86.2**†	**91.4/84.1**‡

4.4.2 主要结果

本章主要实验结果如表4–3和4–4所示，其中 *tscp* 表示人工转写句子，*hyp* 表示语音识别结果（最佳候选句子）。首先对比表4–3和表4–4，我们可以发现词库特征会明显提升相应口语理解模型的性能。然后具体对比基线系统和我们提出的方法（Proposed），我们会发现：

1. 对齐式的语义槽标记建模（BLSTM-focus）会明显好于非对齐式的基于生成的层级式解码模型（HD）。因为序列标注建模考虑了输入和输出的对齐关系，建模难度会比生成式的模型简单很多。
2. BLSTM-focus 模型在人工转写句子上做测试会极大地好于在语音识别结果上进行测试，这体现了语音识别错误对口语理解的负面影响。
3. 通过无监督输入自适应训练，UA 方法在少数领域上有细微的性能提升，但是总体来看并没有带来好处。而数据扩充的两种方式，GEN 和 ALIGN 可以比原始基线 BLSTM-focus 有少量的性能提升。
4. 我们的方法可以在所有领域都超越原始基线 BLSTM-focus 的性能，并在四个领域平均的结果上显著地好于所有的基线系统。

随后，为了验证槽值纠错指导的输入自适应训练的有效性，我们将算法4–2中用到的语音识别结果替换为人工转写句子。以此对比实验来观察强化学习算法和输入自适应训练各自对结果的影响程度。我们发现，在不利用语音识别结果后（即倒数第二行结果），我们仍然可以从强化学习的优化算法中获取少量的性能提升，但是更多的性能提升依然来自于在训练过程中使用了语音识别结果。

表 4–5 不同语义槽标记模型的对比实验以及在四个领域上的平均性能

Table 4–5 Ablation experiments of different slot tagging models.

语义槽标记模型	词库特征	
	✓	✗
BLSTM	90.84/83.92	89.34/80.78
BLSTM-CRF	91.31/**84.49**	90.30/82.43
BLSTM-focus	**91.46**/84.31	**91.03/83.29**

4.4.3 实验对比与分析

4.4.3.1 不同语义槽标记模型的对比

在第4.2.1小节提到了三种语义槽标记模型：BLSTM，BLSTM-CRF 以及上述实验用到的 BLSTM-focus。我们在表4–5中对比了三种模型在使用和不使用词库特征两种情况下的各领域平均性能。我们发现不考虑输出标签依赖关系的 BLSTM 始终是表现最差的。BLSTM-CRF 和 BLSTM-focus 则表现相近，但我们最终还是选择了性能稍优的 BLSTM-focus 作为基础模型，用于上面的主要实验以及后续的分析性实验中。

4.4.3.2 基于语音识别错误恢复的槽值纠错模块的有效性

为了验证槽值纠错模块的有效性，我们在基线系统和本章提出的训练方法中尝试不同的槽值纠错算法：1）None：即不使用槽值纠错算法；2）Delete：仅删除不符合领域本体的语义槽值对，不进行错误恢复操作；3）Full：完整的槽值纠错算法。具体对比结果如表4–6所示。在基线系统 HD，Focus 和 DA-Align 上，槽值纠错算法作为测试时候的后处理，越完整的槽值纠错算法能获得更精准的口语理解性能。而在我们提出的基于强化学习的输入自适应训练方法中，训练和测试阶段均利用了槽值纠错算法。我们的方法在不同程度的槽值纠错算法上均明显好于上述的基线系统，而且即便在不使用槽值纠错算法的情况下也能取得显著的提升。

4.4.3.3 人工转写数据对输入自适应训练的影响

算法4–2的训练过程中除了利用必不可少的语音识别结果 $\mathcal{D}_{\text{hyp}}$，同时也利用了人工转写数据 $\mathcal{D}_{\text{tscp}}$ 作为预训练和伴随性训练（即在 RL 训练阶段依然把人工转写数据加入进来更新语义槽标记模型）。在此，我们探索人工转写数据对输入自适应训练的影响，具体实验结果如表4–7所示。从结果可以发现，如果没有在 $\mathcal{D}_{\text{tscp}}$ 上的预训练和伴随性训练的介入，强化学习要面临非常复杂的参数搜索空间，从

表 4–6 不同槽值纠错算法的对比实验（带词库特征）

Table 4–6 Ablation study of the value error recovery module.

Models	**槽值纠错算法**		
	None	Delete	Full
HD	82.47/73.90	87.30/76.05	89.16/81.22
FOCUS	88.75/81.81	91.10/83.34	91.46/84.31
DA-ALIGN	88.53/82.12	91.25/83.59	91.69/84.74
Proposed	**89.61/83.33**	**91.68/84.17**	**92.28/85.83**

表 4–7 基于强化学习的输入自适应训练的拆解分析（带词库特征）

Table 4–7 Ablation study of the training procedure for our proposed method.

$\mathcal{D}_{\text{tscp}}$ 上预训练	$\mathcal{D}_{\text{tscp}}$ 上伴随性训练	**Avg.**
✓	✓	**92.28/85.83**
✗	✓	91.89/85.12
✓	✗	91.87/85.01
✗	✗	43.06/34.69

而很难训练得到一个不错的语义槽标记模型。而使用 $\mathcal{D}_{\text{tscp}}$ 的预训练或者伴随性训练则通过拟合人工转写句子上的语义槽标注，很好地把强化学习的参数搜索空间限制在一个合理的区域附近。从另外一个角度来看，语音识别错误大多数情况下是少见的，绝大多数的语音识别句子和人工转写句子其实是非常接近甚至一模一样的。

4.4.3.4 不同语音识别错误的数据分析

为了进一步观察在不同的语音识别错误程度下本章提出的方法与基线系统有什么区别，我们将测试数据按照字错误率（CER）分成不同的组。如图4–3所示，测试数据按 CER 从大到小分为五组。首先，四个领域中的 CER 分布均为 U 形，即大部分数据的 CER 在 10% 以内或者在 90% 以上。然后我们可以发现我们的方法在 CER 10% 以内的数据上均小幅优于基线系统，在 CER 90% 以上的数据上大幅优于基线系统，在 CER 处于 10% 到 90% 之间的数据上绝大多数情况都明显优于基线系统。

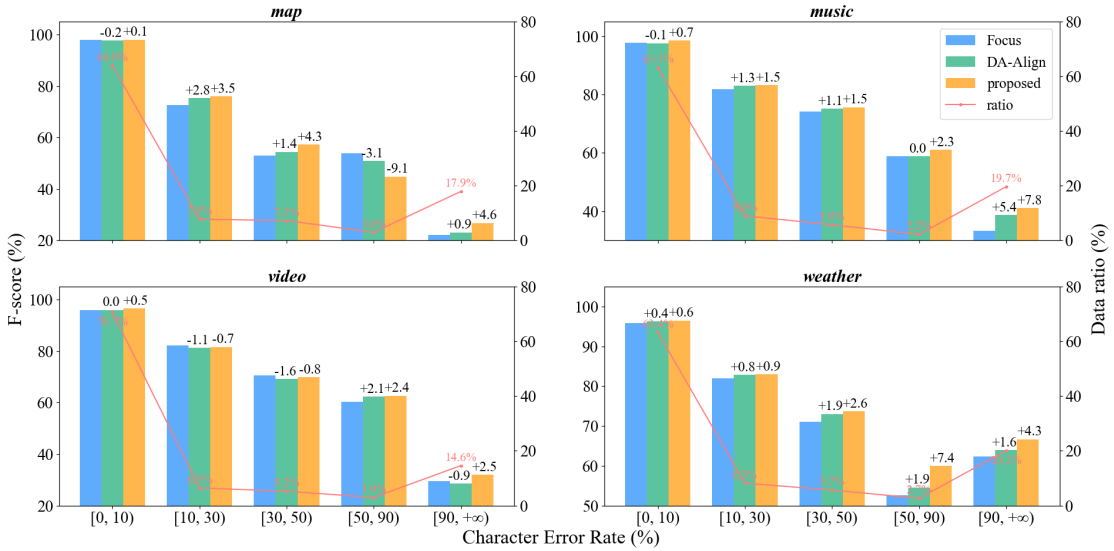


图 4–3 相关方法在不同字错误率的测试数据上的 F_1 值（带词库特征），其中红线表示不同字错误率区间的数据在测试集的占比

Figure 4–3 F_1-scores of our proposed model (with lexicon features) on the test set across four domains with various CER. The data ratio of each group is displayed in the form of line chart.

人工转写句子		*要 去 乌 苏 市 甘 河 子 镇*
语音识别结果		去 公 司 是 甘 河 子 镇
真实语义槽值对标注		inform-dest-乌苏市甘河子镇
BLSTM-focus	语义槽标记	去 公 司 是 甘 河 子 镇 O B-dest I-dest O B-dest I-dest I-dest I-dest
	语义槽值对	inform-dest-公司; inform-dest-甘河子镇
	纠正后的语义槽值对	inform-dest-公司; inform-dest-甘河子镇
Proposed	语义槽标记	去 公 司 是 甘 河 子 镇 O B-dest I-dest I-dest I-dest I-dest I-dest I-dest
	语义槽值对	inform-dest-公司是甘河子镇
	纠正后的语义槽值对	inform-dest-乌苏市甘河子镇

图 4–4 一个地图导航领域的案例分析

Figure 4–4 A case study of the map domain.

4.4.3.5 案例分析

我们提出的基于槽值纠错和强化学习的输入自适应训练方法，利用槽值纠错模块指导语义槽标记模型在语音识别结果上的自适应，旨在训练一个更适合槽值纠错模块的语义槽标记模型。我们在图4–4展示了一个案例，可以看到槽值纠错模块是如何帮助语义槽标记模型训练的。我们可以看到，基线系统 BLSTM-focus 标

记出来了两个语义槽及其值（“公司”和“甘河子镇”），且都是错误的。而我们提出的方法可以标记出更长更全的一个语义槽，且相关槽值能被正确地恢复处理。尽管中间的“是”极不可能出现在一个地名中，但我们的模型根据上下仍然认为它是某个长地名中的一部分，且有机会被槽值纠错模块正确地恢复。

4.4.4 CATSLU 比赛结果

最后我们展示一下第一届中文口语理解竞赛的各参赛队的提交结果，如表4–8所示。本次比赛总共有五组研究队伍参赛并合计提交了 14 个系统。五个队伍分别被匿名标记为 team 1-5，比赛时发布的基线系统被标记为 team 0。在实际比赛中，所有队伍事先无法获得测试数据，并被要求提交系统代码，最后由一位不参赛的组织者负责运行和评测。表4–8中具体区分了不同系统使用的输入类型，包括声学信息、ASR 识别结果、领域本体。大多数的提交系统都能超越基线系统，而且大多数提交系统也都利用了领域本体对预测的语义槽值进行了错误恢复。像 DSTC-2&3 竞赛一样，CATSLU 也允许组织者参赛。我们的队伍命名为 team 3，同时为了取得更好的比赛成绩，在利用了槽值纠错模块的基础上，把校验集添加进训练集中增加数据量，并利用了声学特征以及领域自适应训练的技术，最终获得了本次竞赛的第一名。

4.5 本章小结

为了解决在语音识别结果上进行词对齐式语义标注难的问题，本章提出了一种无需语音识别结果上的词对齐语义标注的稳健口语理解训练方法。该方法只需要在人工转写句子上进行词对齐语义标注，然后利用从人工转写句子到语音识别结果上的输入自适应学习算法训练稳健精准的词对齐式口语理解模型。为了提高口语理解的精准性，本章提出一个无参数的基于语音识别错误恢复的槽值纠错模块，并在输入自适应学习中引入相关模块。在目前公开最大的中文口语理解数据集 CATSLU 上的实验评估表明，本章提出的方法在四个领域上都可以有效提升口语理解系统在以语音识别结果为输入时的稳健性和精确性。最后，虽然本章提出的方法理论上可以和前一章的 WCN-Transformer 编码器结合将语义槽标记模型自适应到 WCN 上，但还需要未来的工作对其验证。

表 4–8 CATSLU 竞赛结果

Table 4–8 Results of the first CATSLU

Team	Entry	Used inputs			Map		Music		Weather		Video	
		Audio	ASR-text	Ontology	F1 (%)	Acc (%)	F1 (%)	Acc (%)	F1 (%)	Acc (%)	F1 (%)	Acc (%)
0	Rule		✓	✓	37.92	40.43	77.39	49.26	85.52	75.38	78.25	45.28
	Neural		✓		77.61	74.65	81.57	71.15	85.25	78.16	75.18	57.53
1	1		✓	✓	83.83	80.80	86.78	76.04	**94.16**	88.80	83.51	70.81
	2	✓	✓	✓	86.59	80.80	91.53	82.99	93.74	87.56	90.72	77.94
	3	✓	✓	✓	85.77	81.31	89.53	79.29	92.27	85.56	92.50	83.49
	4		✓	✓	85.86	81.37	92.59	84.32	89.67	81.50	**93.04**	83.91
	5		✓	✓	86.83	82.32	92.84	84.91	93.75	88.05	-	-
2	1		✓		77.82	74.46	82.68	71.75	87.28	80.94	77.23	61.91
	2		✓		78.40	74.71	82.80	72.19	87.39	80.86	77.84	62.10
3	1		✓	✓	88.07	83.84	92.63	84.76	93.35	87.44	92.18	82.75
	2		✓	✓	87.92	83.78	92.74	85.06	92.99	86.80	92.28	82.57
	3	✓	✓	✓	88.66	84.54	93.13	85.36	93.72	88.16	92.49	83.49
	4	✓	✓	✓	**89.28**	84.47	**93.53**	86.09	93.88	**89.02**	92.77	83.55
	5	✓	✓	✓	89.00	**84.54**	93.42	**86.69**	93.70	88.80	92.84	**84.28**
4	1		✓	✓	87.43	83.08	91.53	82.40	93.24	86.95	91.71	81.17
5	1		✓		77.89	74.33	80.75	70.86	86.27	79.14	76.03	61.18

第五章　基于对偶学习数据扩充的半监督语义理解

5.1　引言

在本章中，我们将针对第二个挑战“单领域的数据稀疏问题”展开研究。在该方向的研究中我们暂时选择忽略口语模式下的输入不确定性问题，假设给定的输入是无语音识别错误的人工转写文本或者是自然语言文本，并且从这章开始我们称此类的口语理解为语义理解（Natural Language Understanding，NLU）。如第二章调研的那样，在训练数据充足的情况下，基于深度学习的语义理解方法已经取得了非常好的性能水平。但是深度学习方法严重依赖于大量的领域内训练数据；但是由于语义标注成本高的原因，目前的方法大大限制了语义理解模型的规模化和可扩展性。

为了解决单领域的数据稀疏问题（即没有其它领域的辅助知识），很多半监督学习的方法利用大量未标注的句子（纯句子）来增强有效数据下的有监督训练。这一类方法旨在给未标注的文本预测相应的伪标签，以此增加一定的伪训练样本。但还没有方法研究如何使用无文本对应的语义表示形式（纯语义形式）。

在“数据扩充”思想的指导下，本章提出在半监督式的语义理解模型训练中引入除一般纯句子之外的纯语义形式数据。引入纯语义形式数据相比收集大量未标注的句子具有成本更低、效果更优的好处。因为语义形式是高度结构化的，可以通过给定的领域知识（领域本体）自动生成或者合成。首先，为了在“数据扩充”的思路下利用好纯语义形式数据，本章提出了语义理解的对偶任务（即反向任务，“语义到句子”），通过的少量的有监督数据学习相关反向模型，然后应用到大量的纯语义形式上为当前领域生成相应的伪样本。其次，考虑到正向模型和反向模型对偶特性（“闭环”特性），我们进一步提出利用对偶学习技术对两个模型进行合作式的优化学习（“闭环”优化）。该优化算法可以分为两大部分：对偶伪标签和对偶学习。图5–1展示了一个语义理解和其对偶任务的示例。

本章的结构如下：我们首先在第5.2节回顾一下任务型语义理解任务，并重点描述对偶模型的架构；在第5.3节介绍相关的对偶半监督语义理解学习框架，包括对偶伪标签和对偶学习两部分；第5.4节是实验部分，包括实验设置、主要结果与实验分析；第5.5节是本章小结。

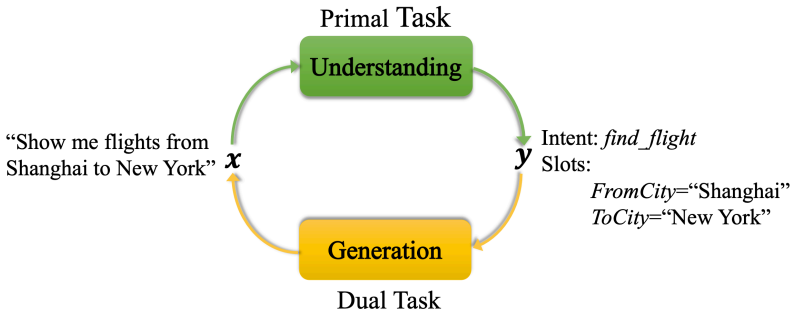


图 5–1 语义理解和其对偶任务的示例

Figure 5–1 A diagram of natural language understanding and its dual task.

5.2 语义理解及其对偶任务

在本小节我们先简单回顾一下语义理解任务的组成，然后重点介绍其对偶任务（基于语义形式的句子生成任务）。

5.2.1 语义理解：意图识别与语义槽填充

表 5–1 意图识别与语义槽填充（IOB 格式）的标注示例

Table 5–1 An example of intent and slot annotation (IOB format).

输入句子 ($\boldsymbol{x}$)	show	me	flights	from	*Shanghai*	to	*New*	*York*
语义槽标记 ($\boldsymbol{y}^S$)	O	O	O	O	B-FromCity	O	B-ToCity	I-ToCity
意图类别 (y^I)	Find_Flight							
语义槽值对 * ($\boldsymbol{y}^C$)	{FromCity=*Shanghai*; ToCity=*New York*}							

如前文介绍，任务型语义理解包括两个子任务：意图识别和语义槽填充。表5–1展示了一个意图识别与语义槽填充（IOB 格式）的标注示例，其中意图识别可以被看成句子分类问题，语义槽填充则被看成序列标注任务（所以我们采用和第4.2节一样的语义槽标注模型）。

我们用 $\boldsymbol{x} = (x_1, x_2, \cdots, x_{|\boldsymbol{x}|})$ 表示输入句子（词序列），用 y^I 表示它的意图类别标签，用 $\boldsymbol{y}^S = (y_1^S, \cdots, y_{|\boldsymbol{x}|}^S)$ 表示它的语义槽标记输出序列。其中 $|\boldsymbol{x}|$ 是指句子长度，$y_i^S \in \mathcal{V}_{\text{tag}}$ 以及 $y^I \in \mathcal{V}_{\text{intent}}$（$\mathcal{V}_{\text{tag}}$ 和 $\mathcal{V}_{\text{intent}}$ 分别是所有语义槽标记和意图类别的词汇表）。因此，任务型语义理解中的意图识别和语义槽填充任务就是为了估计给定输入 $\boldsymbol{x}$ 时意图类别 y^I 和语义槽标记序列 $\boldsymbol{y}^S$ 的联合后验概率，$p(y^I, \boldsymbol{y}^S|\boldsymbol{x})$。

大多数情况下，两个子任务是分别独立建模型的：

$$p(\tilde{\boldsymbol{y}}|x) = p(y^I, \boldsymbol{y}^S|\boldsymbol{x}) = p(y^I|\boldsymbol{x})p(\boldsymbol{y}^S|\boldsymbol{x}) \tag{5–1}$$

其中 $\tilde{\boldsymbol{y}} = (y^I, \boldsymbol{y}^S)$。

本章的任务型语义理解模型由三部分组成：句子编码，意图分类，语义槽标记。

1）句子编码：首先每个词都会被映射成一个词向量，即 $\mathbf{x}_i = \mathbf{W}_{\text{in}}\mathbf{o}(x_i)$，其中 $\mathbf{W}_{\text{in}} \in \mathbb{R}^{d_e \times |\mathcal{V}_{\text{in}}|}$ 是一个词向量矩阵，$\mathbf{o}(x_i) \in \mathbb{R}^{|\mathcal{V}_{\text{in}}|}$ 是一个独热（one-hot）向量，$\mathcal{V}_{\text{in}}$ 表示输入字典，d_e 为词向量维度，$\mathbf{x}_i \in \mathbb{R}^{d_e}$。我们使用 BLSTM 对该词向量序列进行编码（BLSTM 计算过程见附录A.1），在每个时刻得到一个隐向量 $\mathbf{h}_i^{\text{sen}} \in \mathbb{R}^{2d_h}$（$d_h$ 是 LSTM 隐层大小，$i \in \{1, \cdots, |\boldsymbol{x}|\}$）：

$$(\mathbf{h}_1^{\text{sen}}, \cdots, \mathbf{h}_{|\boldsymbol{x}|}^{\text{sen}}) \leftarrow \text{BLSTM}_{\Theta_1}(\mathbf{x}_1, \cdots, \mathbf{x}_{|\boldsymbol{x}|}). \tag{5–2}$$

2）意图分类：我们采用一个注意力机制模型（ATTN_{Θ_2}, 计算过程见附录A.2）聚集隐向量得到一个句向量，并把句向量送入一个线性输出层进行意图分类：

$$\mathbf{z}^{\text{sen}}, (a_1^{\text{sen}}, \cdots, a_{|\boldsymbol{x}|}^{\text{sen}}) \leftarrow \text{ATTN}_{\Theta_2}(\overleftarrow{\mathbf{h}}_1^{\text{sen}}, (\mathbf{h}_1^{\text{sen}}, \cdots, \mathbf{h}_{|\boldsymbol{x}|}^{\text{sen}})) \tag{5–3}$$

$$p(y^I|\boldsymbol{x}) = \text{softmax}_{y^I}(\mathbf{W}_1\mathbf{z}^{\text{sen}} + \mathbf{b}_1) \tag{5–4}$$

其中 $\mathbf{W}_1 \in \mathbb{R}^{|\mathcal{V}_{\text{intent}}| \times 2d_h}$，$\mathbf{b}_1 \in \mathbb{R}^{|\mathcal{V}_{\text{intent}}|}$ 分别是输出层的权值矩阵和偏置向量

3）语义槽标记：语义槽填充（slot filling）任务被建模为序列标注任务，也称语义槽标记（slot tagging）。如第4.2.1节一样，我们可以使用三种典型的语义槽标记模型（BLSTM，BLSTM-CRF，BLSTM-focus）分别对 $p(\boldsymbol{y}^S|\boldsymbol{x})$ 进行建模。但是，即便获取了语义槽标记的输出序列，它还只是语义表示形式的半成品，因为每个语义槽的值还没有指明。于是，我们需要利用输出输出序列的对齐关系，整理出“语义槽值对”。如表5–1所示，通过对比 $\boldsymbol{x}$ 和 $\boldsymbol{y}^S$ 可以得到语义槽值对 $\boldsymbol{y}^C = \text{summary}(\boldsymbol{x}, \boldsymbol{y}^S)$，即 *(FromCity=Shanghai, ToCity=New York)*。即最终的语义表示形式为 $\boldsymbol{y} = (y^I, \boldsymbol{y}^C)$。

给定训练样本 $\boldsymbol{x}$ 和 $\boldsymbol{y}$，语义理解模型的训练损失函数定义为

$$\mathcal{L}_{\text{NLU}}(\boldsymbol{x}, \boldsymbol{y}) = -\log p(\tilde{\boldsymbol{y}}|\boldsymbol{x}) = -\log p(y^I|\boldsymbol{x}) - \log p(\boldsymbol{y}^S|\boldsymbol{x}). \tag{5–5}$$

5.2.2 对偶任务：基于语义形式的句子生成

在本节，我们将重点介绍语义理解的对偶任务，即“语义到句子”生成任务（Natural Language Generation，NLG）。给定一个语义形式 $\boldsymbol{y}$，即一个意图类别 y^I 和

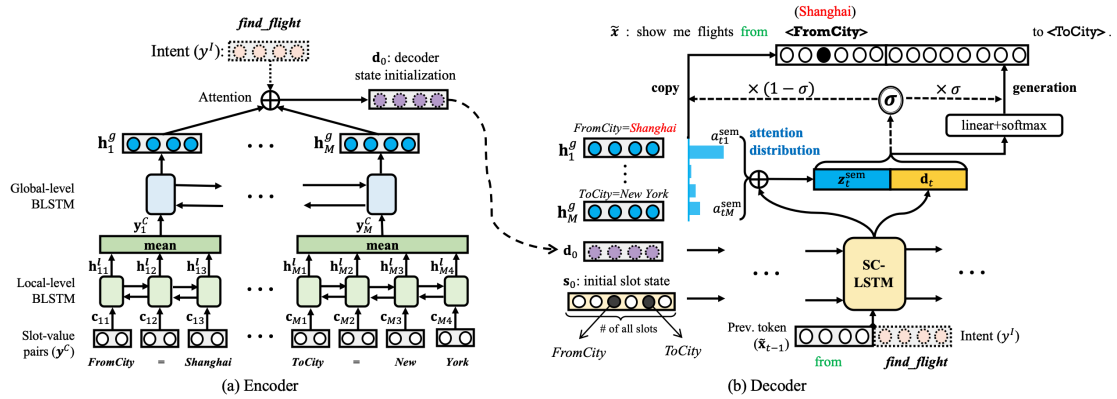


图 5–2 语义理解对偶任务的模型架构。其编码器利用一个层级式的 BLSTM 获取语义槽值对 $\boldsymbol{y}^C$ 的深度特征；解码器则使用语义可控的 LSTM 精确生成去词汇化的句子 $\tilde{\boldsymbol{x}}$，然后将 $\boldsymbol{y}^C$ 中相应的值填入到 $\tilde{\boldsymbol{x}}$ 中

Figure 5–2 The proposed architecture for the dual task of NLU, which is comprised of an encoder and a decoder. The encoder is a hierarchical BLSTM to obtain deep features for a list of slot-value pairs $\boldsymbol{y}^C$. The decoder exploits a semantically controlled LSTM to precisely generate a delexicalized form $\tilde{\boldsymbol{x}}$, and then substitute the special slot tokens with the corresponding values in $\boldsymbol{y}^C$.

一个语义槽值对的序列[①] $\boldsymbol{y}^C = (y_1^C, \cdots, y_M^C)$（$M$ 表示语义槽值对个数），NLG 的目标是为该语义形式生成匹配的句子 $\boldsymbol{x}$。根据词对齐式语义槽标记任务的特性，$\boldsymbol{y}^C$ 中的每个值都必定出现在句子 $\boldsymbol{x}$ 中，且相互不重叠。那么，我们选择第一步先生成去词汇化（即将句子中语义槽对应位置的词换成语义槽符号）的句子[②] $\tilde{\boldsymbol{x}}$（由词和语义槽组成）；然后在将 $\boldsymbol{y}^C$ 中的值填到相应的语义槽上，复原出 $\boldsymbol{x}$。所以，NLG 模型旨在估计如下条件概率

$$p(\tilde{\boldsymbol{x}}|\boldsymbol{y}) = p(\tilde{\boldsymbol{x}}|y^I, \boldsymbol{y}^C), \tag{5–6}$$

序列到序列的编码器-解码器架构在自然语言生成领域已经取得了很大的成功，比如机器翻译 [48, 168]，对话生成 [169] 和文本摘要 [158] 等等。然而把相关架构应用到 NLG 上并不简单，因为 NLG 输入的语义表示不是一个序列，而是一个结构化的形式（即一个意图类别和一个语义槽值对的列表）。

本章提出的 NLG 模型架构包括一个编码器和一个解码器，如图5–2所示。其中编码器将意图 y^I 和语义槽值对列表 $\boldsymbol{y}^C$ 转换为向量表示，然后解码器根据编码得到的向量生成去词汇化的句子 $\tilde{\boldsymbol{x}}$。最终，我们将 $\tilde{\boldsymbol{x}}$ 中的语义槽替换为 $\boldsymbol{y}^C$ 中相应的值。

① 和前文非对齐语义标注的语义槽值对 “集合” 不同，在对齐标注中可以根据值在句子中出现的先后顺序排序，得到语义槽值对的 “序列”。

② 比如表5–1中的句子去词汇化后是 “*show me flights from* ⟨FromCity⟩ *to* ⟨ToCity⟩”。

5.2.2.1 编码器（encoder）

我们利用层级式的 BLSTM 对语义槽值对序列进行编码，包括局部层和全局层。首先，每个单独的语义槽值对（slot-value pair）被看成一个子序列（短的词序列），即 $y_m^C = (c_{m1}, \cdots, c_{mT_m})$，其中 T_m 是序列长度，$m \in \{1, \cdots, M\}$。比如语义槽值对 *ToCity=New York* 的子序列是 ("*ToCity*", "=", "*New*", "*York*")。

局部层（local-level）：对于每个单独的语义槽值对 y_m^C，我们用一个共享的 BLSTM 模型（见附录A.1）进行编码来获取子序列的隐向量：

$$(\mathbf{h}_{m1}^l, \cdots, \mathbf{h}_{mT_m}^l) \leftarrow \text{BLSTM}_{\Theta_3}(\mathbf{c}_{m1}, \cdots, \mathbf{c}_{mT_m}) \tag{5–7}$$

其中 $\mathbf{c}_{mj}$ 是子序列 y_m^C 中第 j 个词的词向量 ①。语义槽值对 y_m^C 的局部表征被定义为这些隐向量的均值，即 $\mathbf{y}_m^C = \frac{1}{T_m}\sum_j \mathbf{h}_{mj}^l$, $\mathbf{y}_m^C \in \mathbb{R}^{2d_h}$。

全局层（global-level）：基于上述得到的每个语义槽值对的局部表征向量，另外一个 BLSTM 被用来获取全局表征，$\mathbf{h}_m^g \in \mathbb{R}^{2d_h}, m \in \{1, \cdots, M\}$:

$$(\mathbf{h}_1^g, \cdots, \mathbf{h}_M^g) \leftarrow \text{BLSTM}_{\Theta_4}(\mathbf{y}_1^C, \cdots, \mathbf{y}_M^C). \tag{5–8}$$

5.2.2.2 解码器（decoder）

为了避免在预测去词汇化句子 $\tilde{x}$ 的过程中丢失或者生成多余的语义槽，我们采用了语义可控的 LSTM 结构 (SC-LSTM) [170]。第 t 时刻的隐向量由此计算

$$(\mathbf{d}_t, \mathbf{s}_t) = \text{f}_{\text{SC-LSTM}}(\tilde{\mathbf{x}}_{t-1} \oplus \mathbf{y}^I, (\mathbf{d}_{t-1}, \mathbf{s}_{t-1})) \tag{5–9}$$

其中 $\tilde{\mathbf{x}}_{t-1}$ 是上一时刻预测出来的词的向量表示，$\mathbf{y}^I$ 是输入意图类别的向量，$\mathbf{d}_t$ 是隐向量。和一般 LSTM 不同的是，SC-LSTM 还包括了一个可以规划句子生成的槽值状态向量 $\mathbf{s}_t$。该向量在生成句子的过程中操控槽值状态特征，以此影响隐层状态并精确编码输入的语义信息。该槽值状态向量由原始语义槽的 one-hot 向量初始化，即 $\mathbf{s}_0 \in \mathbb{R}^{|\mathcal{V}_{\text{slot}}|}$，其中每个元素都为零，除了在语义槽值对列表 $\boldsymbol{y}^C$ 中的语义槽（$\mathcal{V}_{\text{slot}}$ 是当前领域的语义槽词汇表）：

$$s_{0i} = \begin{cases} 1, & \text{如果第}i\text{个语义槽出现在了}\boldsymbol{y}^C\text{中;} \\ 0, & \text{否则。} \end{cases} \tag{5–10}$$

另外，在最终的训练损失函数中会加入一个正则项

$$\mathcal{L}_{\text{SC}} = \|\mathbf{s}_{|\tilde{\boldsymbol{x}}|}\| + \sum_{t=1}^{|\tilde{\boldsymbol{x}}|} \eta \xi^{\|\mathbf{s}_t - \mathbf{s}_{t-1}\|} \tag{5–11}$$

① 每个语义槽也被映射为一个可训练的词向量

其中 $\mathbf{s}_{|\widetilde{\boldsymbol{x}}|}$ 是最后时刻的槽值状态向量，$\eta = 10^{-4}, \xi = 100$，$\|\cdot\|$ 表示向量二范数。该正则函数中第一项的优化目标是让每一个输入的语义槽都出现在生成的句子中，第二项的优化目标则是希望每一步最多生成一个语义槽。

解码器的隐层向量由汇集后的编码表征向量来初始化，即 $\mathbf{d}_0 = \mathbf{W}_0\mathbf{z}_0^{\text{sem}}$，其中 $\mathbf{W}_0 \in \mathbb{R}^{d_h \times 2d_h}$，而 $\mathbf{z}_0^{\text{sem}} \in \mathbb{R}^{2d_h}$ 是编码器隐向量的注意力（见附录A.2）向量：

$$\mathbf{z}_0^{\text{sem}}, (a_{01}^{\text{sem}}, \cdots, a_{0M}^{\text{sem}}) \leftarrow \text{ATTN}_{\Theta_5}(\mathbf{y}^I, (\mathbf{h}_1^g, \cdots, \mathbf{h}_M^g)). \tag{5–12}$$

在 SC-LSTM 的基础上，我们采用了一个包含注意力机制 [168] 和拷贝机制 [158] 的输出层来预测 $\widetilde{\boldsymbol{x}}$ 中的每个词。在解码器第 t 时刻，其关于编码器中第 m 个（$m \in \{1, \cdots, M\}$）语义槽值对的注意力权重以及向量的计算过程如下

$$\mathbf{z}_t^{\text{sem}}, (a_{t1}^{\text{sem}}, \cdots, a_{tM}^{\text{sem}}) \leftarrow \text{ATTN}_{\Theta_6}(\mathbf{d}_t, (\mathbf{h}_1^g, \cdots, \mathbf{h}_M^g)). \tag{5–13}$$

然后通过一个线性层可以得到在输出词表上的概率分布

$$p_{\text{gen}}(\tilde{x}_t|\widetilde{\boldsymbol{x}}_{<t}, \boldsymbol{y}) = \text{softmax}_{\tilde{x}_t}(\mathbf{W}_o(\mathbf{d}_t \oplus \mathbf{z}_t^{\text{sem}}) + \mathbf{b}_o) \tag{5–14}$$

其中 $\mathbf{W}_o \in \mathbb{R}^{|\mathcal{V}_{\tilde{x}}| \times 3d_h}$，$\mathbf{b}_o \in \mathbb{R}^{|\mathcal{V}_{\tilde{x}}|}$，且 $|\mathcal{V}_{\tilde{x}}|$ 表示输出词表的大小（输出词表由输入词表和语义槽词表组成）。当预测得到序列终止符 "</s>" 的时候生成过程停止。

除了上面直接预测生成的方式外，解码器还包括了拷贝机制来提升模型的泛化能力，即可以从输入的语义表示中直接拷贝语义槽。为了融合两种方式，我们使用一个 sigmoid 门控函数 σ 在生成和拷贝之间做一个软性选择：

$$p(\tilde{x}_t|\widetilde{\boldsymbol{x}}_{<t}, \boldsymbol{y}) = g_t p_{\text{gen}}(\tilde{x}_t|\widetilde{\boldsymbol{x}}_{<t}, \boldsymbol{y}) + (1 - g_t)p_{\text{copy}}(\tilde{x}_t|\widetilde{\boldsymbol{x}}_{<t}, \boldsymbol{y}) \tag{5–15}$$

$$g_t = \sigma(\mathbf{v}_g^{\top}(\mathbf{d}_t \oplus \mathbf{z}_t^{\text{sem}}) + \text{b}_g) \tag{5–16}$$

其中 $g_t \in [0, 1]$ 是一个平衡因子，$\mathbf{v}_g$ 是参数向量以及 b_g 是偏置。拷贝机制的预测 $p_{\text{copy}}(\cdot|\cdot)$ 是定义在 M 个语义槽 $(y_1^C, \cdots, y_M^C)$ 上的分布:

$$p_{\text{copy}}(\tilde{x}_t|\widetilde{\boldsymbol{x}}_{<t}, \boldsymbol{y}) = \begin{cases} a_{tm}^{\text{sem}}, & \text{如果 } \tilde{x}_t \text{ 是} y_m^C \text{ 的语义槽}, m \in \{1, \cdots, M\} \\ 0, & \text{否则。} \end{cases} \tag{5–17}$$

如果 $(y_1^C, \cdots, y_M^C)$ 中存在多个同名语义槽，那么我们会提前对它们按顺序添加编号进行区分。

最后，我们把 $\widetilde{\boldsymbol{x}}$ 中的每个语义槽替换为输入中相应的语义槽值对的值即可得到最终的句子 $\boldsymbol{x}$。给定样本 $\boldsymbol{x}$ 和 $\boldsymbol{y}$，NLG 模型的训练损失函数为

$$\mathcal{L}_{\text{NLG}}(\boldsymbol{y}, \boldsymbol{x}) = -\sum_{t=1}^{|\widetilde{\boldsymbol{x}}|} \log p(\tilde{x}_t|\widetilde{\boldsymbol{x}}_{<t}, \boldsymbol{y}) + \mathcal{L}_{\text{SC}} \tag{5–18}$$

5.3 对偶-半监督式语义理解学习框架

本节将介绍我们提出的对偶-半监督式语义理解学习框架（如算法5–1所示），其中包括两种可叠加的方法：对偶伪标签和对偶学习。在半监督学习的设定下，除了充分标注的数据 $\mathcal{D}_{\text{xy}}^{L}$，还有两种未标注的数据：纯（未标注的）句子集 $\mathcal{D}_{\text{x}}^{U}$，以及纯（没有句子对应的）语义形式集 $\mathcal{D}_{\text{y}}^{U}$。

5.3.1 对偶伪标签

“伪标签” 是在未标注数据上利用已有模型预测得到的标签，使用的时候假装它们就是正确的标注 [171]。我们首先在已标注的数据上有监督地预训练 NLU 和 NLG 模型。然后，给定一个未标注的句子 $\boldsymbol{x}$，我们可以使用 NLU 模型预测得到伪标签 $\boldsymbol{y}'$。对称地，我们也可以使用 NLG 模型为无句子对应的语义形式 $\boldsymbol{y}$ 生成伪句子 $\boldsymbol{x}'$。换句话说，我们可以构建额外的伪训练样本 $(\boldsymbol{x}, \boldsymbol{y}')$ 和 $(\boldsymbol{x}', \boldsymbol{y})$，如算法5–1（part 1）所示.

除了使用已标注的数据进行有监督的训练，我们还可以利用上述伪样本更新 NLU 和 NLG 模型，即各自最小化如下训练损失：

$$\mathcal{L}_{\text{pseudo-NLU}} = w_i(\mathcal{L}_{\text{NLU}}(\boldsymbol{x}, \boldsymbol{y}') + \mathcal{L}_{\text{NLU}}(\boldsymbol{x}', \boldsymbol{y})) \tag{5–19}$$

$$\mathcal{L}_{\text{pseudo-NLG}} = w_i(\mathcal{L}_{\text{NLG}}(\boldsymbol{y}, \boldsymbol{x}') + \mathcal{L}_{\text{NLG}}(\boldsymbol{y}', \boldsymbol{x})) \tag{5–20}$$

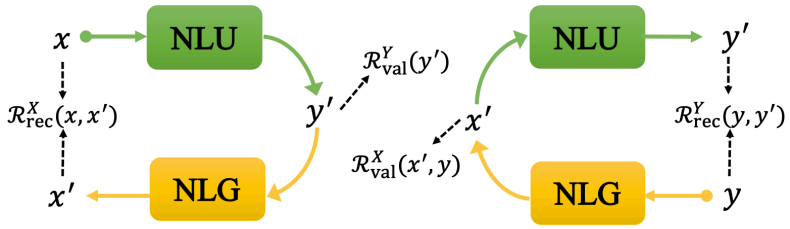
其中 i 表示第 i 轮训练，$w_i \in [0, 1]$ 则是一个重要性系数。为了防止模型掉入较差的局部最优解中，我们选择缓慢地增大 w_i。于是，随着训练的进行，越靠后生成的伪样本被赋予的置信度越高。具体来说，我们定义 $w_i = \frac{i}{N}$，其中 N 是训练的最大轮数。

除了利用未标注的句子和无句子对应的语义形式之外，我们同样也对已标注数据中的样本重新生成伪样本。首先，这样可以一定程度地增加伪样本的多样性。其次，这样做或许可以纠正一些潜在的标注错误。

之前的一些和伪标签学习相关的工作 [171-173] 仅针对无标注的输入 $\boldsymbol{x}$ 生成相应的伪标签 $\boldsymbol{y}'$。据我们所知，本文首次提出了对偶伪标签方法，且该方法可以共享 NLU 和 NLG 任务的伪样本并迭代式优化 NLU 和 NLG 模型。

5.3.2 对偶学习

上述提出的对偶伪标签的训练过程中，NLU 和 NLG 模型是独立优化的。在本小节，我们继续提出利用对偶学习方法来联合训练这两个模型。在对偶学习中，我们为 NLU 和 NLG 模型设计了一个可以提供质量反馈的闭环游戏，即便在只有



(a) *NLU2NLG* (b) *NLG2NLU*

图 5–3 对偶学习方法的示意图。其中 NLU 和 NLG 模型可以构成一个闭环连接，包括两个有向环：NLU2NLG 和 NLG2NLU，并各自从一个句子 $\boldsymbol{x}$ 和语义形式 $\boldsymbol{y}$ 出发

Figure 5–3 An overview of dual learning method. The NLU and NLG models can form a closed cycle, which contains two directed loops NLU2NLG and NLG2NLU starting from a sentence $\boldsymbol{x}$ and semantic form $\boldsymbol{y}$ respectively.

句子或者语义形式的情况下也可以运行。因为反馈的收益（reward）是不可微的，我们在参数优化过程中应用了基于策略梯度 [174] 的强化学习 [175] 方法。

如图5–3所示，NLU 和 NLG 模型参与了两个有向回环组成的协作游戏：

1. NLU2NLG 回环先从句子开始，通过 NLU 模型生成一个可能的语义形式，再试图通过 NLG 模型重构出原始的输入句子。
2. NLG2NLU 回环则从相反的反向开始。

根据在回环上定义的收益函数，每个模型都会获取相应的质量反馈。一般，NLU 和 NLG 模型需要在已标注的数据上预训练好。记 NLU 和 NLG 模型各自的参数为 Θ_{NLU} 和 Θ_{NLG}。对偶学习方法的简要描述如算法5–1 （part 2）所示，且两个有向回环的具体计算过程如下：

5.3.2.1 回环 NLU2NLG

首先，我们从所有数据（包括 D^L_{xy} 和 D^U_{x}）中采样一个句子 $\boldsymbol{x}$。给定 $\boldsymbol{x}$，NLU 模型可以通过集束搜索（beam search）预测得到 K 个可能的语义形式 $\boldsymbol{y}'^1, \cdots, \boldsymbol{y}'^K$，其中 K 是集束大小。对于每一个 $\boldsymbol{y}'^k$，我们都可以得到一个有效性收益（常量）$\mathcal{R}^Y_{\text{val}}(\boldsymbol{y}'^k)$。该收益反映了 $\boldsymbol{y}'^k$ 是一个有效合理的语义形式的可能性。随后，我们把 $\boldsymbol{y}'^k$ 传递给 NLG 模型，同样通过集束搜索解码得到 K 个不同的输出 $\boldsymbol{x}'^{k1}, \cdots, \boldsymbol{x}'^{kK}$。最终，对于每个 $\boldsymbol{x}'^{kj}$，我们都可以算得一个重构收益 $\mathcal{R}^X_{\text{rec}}(\boldsymbol{x}, \boldsymbol{x}'^{kj})$。重构收益的目标是驱使生成的句子和原始输入的句子越像越好。相关收益函数的具体定义将在

第5.3.2.3小节详细介绍。我们采用一个平衡系数 $\alpha \in [0,1]$ 来融合两种收益：

$$r_1^k(\boldsymbol{x}) = \alpha \mathcal{R}_{\text{val}}^Y(\boldsymbol{y}'^k) + (1-\alpha)\frac{1}{K}\sum_{j=1}^{K}\mathcal{R}_{\text{rec}}^X(\boldsymbol{x}, \boldsymbol{x}'^{kj}) \tag{5–21}$$

我们利用策略梯度算法 [174] 最小化负的期望收益 $\psi_1(\boldsymbol{x}) \approx -\frac{1}{K}\sum_{k=1}^{K} r_1^k(\boldsymbol{x})$ 可以计算得到关于参数 Θ_{NLU} 和 Θ_{NLG} 的梯度，分别是:

$$\nabla_{\Theta_{\text{NLU}}}\psi_1(\boldsymbol{x}) = \frac{1}{K}\sum_{k=1}^{K}\left(r_1^k(\boldsymbol{x}) - \mathcal{B}_{\text{NLU}}^1(\boldsymbol{x})\right)\nabla_{\Theta_{\text{NLU}}}\mathcal{L}_{\text{NLU}}(\boldsymbol{x}, \boldsymbol{y}'^k) \tag{5–22}$$

$$\nabla_{\Theta_{\text{NLG}}}\psi_1(\boldsymbol{x}) = \frac{1-\alpha}{K^2}\sum_{k=1}^{K}\sum_{j=1}^{K}\left(\mathcal{R}_{\text{rec}}^X(\boldsymbol{x}, \boldsymbol{x}'^{kj}) - \mathcal{B}_{\text{NLG}}^1(\boldsymbol{x}, \boldsymbol{x}'^k)\right)\nabla_{\Theta_{\text{NLG}}}\mathcal{L}_{\text{NLG}}(\boldsymbol{y}'^k, \boldsymbol{x}'^{kj}) \tag{5–23}$$

其中 $\mathcal{B}_{\text{NLU}}^1(\boldsymbol{x}) = \frac{1}{K}\sum_{k=1}^{K} r_1^k(\boldsymbol{x})$ 以及 $\mathcal{B}_{\text{NLG}}^1(\boldsymbol{x}, \boldsymbol{x}'^k) = \frac{1}{K}\sum_{j=1}^{K}\mathcal{R}_{\text{rec}}^X(\boldsymbol{x}, \boldsymbol{x}'^{kj})$ 是用于降低训练方差的基线 [166]。

5.3.2.2 回环 NLG2NLU

对称地，我们也可以从所有数据（包括 $\mathcal{D}_{\text{xy}}^L$ 和 $\mathcal{D}_{\text{y}}^U$）采样一个无句子对应的语义形式 $\boldsymbol{y}$。给定 $\boldsymbol{y}$，NLG 模型可以通过集束搜索生成 K 个可能的句子 $\boldsymbol{x}'^1, \cdots, \boldsymbol{x}'^K$。对于每一个 $\boldsymbol{x}'^k$，我们都可以得到一个有效性收益（常量）$\mathcal{R}_{\text{val}}^X(\boldsymbol{x}'^k, \boldsymbol{y})$。该收益反映了 $\boldsymbol{x}'^k$ 是一个合理且顺畅的句子的可能性。随后，我们把 $\boldsymbol{x}'^k$ 传递给 NLU 模型，同样通过集束搜索解码得到 K 个不同的输出 $\boldsymbol{y}'^{k1}, \cdots, \boldsymbol{y}'^{kK}$。最终，对于每个 $\boldsymbol{y}'^{kj}$，我们都可以算得一个重构收益 $\mathcal{R}_{\text{rec}}^Y(\boldsymbol{y}, \boldsymbol{y}'^{kj})$，以此驱使生成的语义形式和原始输入越像越好。相关收益函数的具体定义将在第5.3.2.3小节详细介绍。我们采用一个平衡系数 $\beta \in [0,1]$ 来融合两种收益：

$$r_2^k(\boldsymbol{y}) = \beta \mathcal{R}_{\text{val}}^X(\boldsymbol{x}'^k, \boldsymbol{y}) + (1-\beta)\frac{1}{K}\sum_{j=1}^{K}\mathcal{R}_{\text{rec}}^Y(\boldsymbol{y}, \boldsymbol{y}'^{kj}) \tag{5–24}$$

我们利用策略梯度算法 [174] 最小化负的期望收益 $\psi_2(\boldsymbol{y}) \approx -\frac{1}{K}\sum_{k=1}^{K} r_2^k(\boldsymbol{y})$ 可以计算得到关于参数 Θ_{NLG} 和 Θ_{NLU} 的梯度，分别是:

$$\nabla_{\Theta_{\text{NLG}}}\psi_2(\boldsymbol{y}) = \frac{1}{K}\sum_{k=1}^{K}\left(r_2^k(\boldsymbol{y}) - \mathcal{B}_{\text{NLG}}^2(\boldsymbol{y})\right)\nabla_{\Theta_{\text{NLG}}}\mathcal{L}_{\text{NLG}}(\boldsymbol{y}, \boldsymbol{x}'^k) \tag{5–25}$$

$$\nabla_{\Theta_{\text{NLU}}}\psi_2(\boldsymbol{y}) = \frac{1-\beta}{K^2}\sum_{k=1}^{K}\sum_{j=1}^{K}\left(\mathcal{R}^{Y}_{\text{rec}}(\boldsymbol{y},\boldsymbol{y}'^{kj}) - \mathcal{B}^{2}_{\text{NLU}}(\boldsymbol{y},\boldsymbol{y}'^{k})\right)\nabla_{\Theta_{\text{NLU}}}\mathcal{L}_{\text{NLU}}(\boldsymbol{x}'^{k},\boldsymbol{y}'^{kj}) \tag{5–26}$$

其中 $\mathcal{B}^{2}_{\text{NLG}}(\boldsymbol{y}) = \frac{1}{K}\sum_{k=1}^{K} r_2^k(\boldsymbol{y})$ 以及 $\mathcal{B}^{2}_{\text{NLU}}(\boldsymbol{y},\boldsymbol{y}'^{k}) = \frac{1}{K}\sum_{j=1}^{K}\mathcal{R}^{Y}_{\text{rec}}(\boldsymbol{y},\boldsymbol{y}'^{kj})$ 是用于降低训练方差的基线 [166]。

据我们所知，这是第一个将对偶学习算法应用到任务型语义理解的工作。而对偶学习首次出现在机器翻译中的工作 [116] 仅使用语言模型作为收益函数，本章节我们引入了新的适用于任务型语义理解的有效性和重构收益函数。理论上这些收益函数可以适合很多其他的结构化自然语言理解任务（比如基于知识库问答的语义解析、事件抽取等等）。

5.3.2.3 收益函数设计

本小节我们将详细介绍两种有效性和重构收益函数在两个不同回环中的定义。

有效性收益 $\mathcal{R}^{Y}_{\text{val}}(\boldsymbol{y}')$ 评估的是一个语义形式是否有效合理。我们可以通过统计意图和语义槽的两种关系来衡量它：

- *槽值对（slot-value）*：评估一个槽值对是否合法，比如 “boston” 是语义槽 `FromCity` 的一个合法取值，而 “today” 不是。
- *槽与意图对（slot-intent）*：评估一个语义槽是否可能与某个意图共同出现，比如语义槽 `FromCity` 经常会和意图 `find_flight` 一起出现在同一个句子中。

为此，我们在训练数据上构建一个词汇数据库 $\text{DB}(\cdot)$，它记录了每个语义槽 s 的所有可能取值 v。同样，根据训练数据构建一个共现矩阵 $\text{COM} \in \mathbb{R}^{|\mathcal{V}_{\text{intent}}|\times|\mathcal{V}_{\text{slot}}|}$，其中元素 $\text{COM}(i,s) \in \{0,1\}$ 表示语义槽 s 和意图 i 是否同时出现过。具体地，$\mathcal{R}^{Y}_{\text{val}}(\boldsymbol{y}')$ 定义如下：

$$\text{score}(s,v) = \max_{e\in\text{DB}(s)}(1 - \text{Edit\_Distance}(e,v)/|v|) \tag{5–27}$$

$$r_{\text{sv}}(\boldsymbol{y}^{C\prime}) = \begin{cases} \frac{1}{|\boldsymbol{y}^{C\prime}|}\sum_{(s,v)\in\boldsymbol{y}^{C\prime}}\text{score}(s,v), & \text{如果}|\boldsymbol{y}^{C\prime}|\neq 0 \\ 1.0, & \text{否则。} \end{cases} \tag{5–28}$$

$$r_{\text{si}}(y^{I\prime},\boldsymbol{y}^{C\prime}) = \begin{cases} \frac{1}{|\boldsymbol{y}^{C\prime}|}\sum_{(s,v)\in\boldsymbol{y}^{C\prime}}\text{COM}(y^{I\prime},s), & \text{如果}|\boldsymbol{y}^{C\prime}|\neq 0 \\ 1.0, & \text{否则。} \end{cases} \tag{5–29}$$

$$\mathcal{R}^{Y}_{\text{val}}(\boldsymbol{y}') = \lambda\cdot r_{\text{sv}}(\boldsymbol{y}^{C\prime}) + (1-\lambda)\cdot r_{\text{si}}(y^{I\prime},\boldsymbol{y}^{C\prime}) \tag{5–30}$$

其中 $\boldsymbol{y}' = (y^{I\prime}, \boldsymbol{y}^{C\prime})$ 包括一个意图 $y^{I\prime}$ 一个语义槽值对列表 $\boldsymbol{y}^{C\prime}$，Edit_Distance(e, v) 计算的是两个值之间的词级别的编辑距离，而 λ 则是权值因子.

有效性收益 $\mathcal{R}^{X}_{\text{val}}(\boldsymbol{x}', \boldsymbol{y})$ 从以下两方面来评估一个生成的自然语言句子 $\boldsymbol{x}'$ 是否是合理且流畅的：

- 语义完整性: 评估 $\boldsymbol{x}'$ 是否包含了输入 $\boldsymbol{y}$ 中的所有语义槽，并可以通过计算语义槽准确率来获取，即 $\text{SlotAcc}(\boldsymbol{x}', \boldsymbol{y}) = 1 - \frac{p+q}{t}$，其中 t 表示 $\boldsymbol{y}$ 中的语义槽值对的个数，p 和 q 则各自表示 $\boldsymbol{x}'$ 在 NLG 模型生成过程中的去词汇化句子里忽略掉的和多余的语义槽数量。
- 文字流畅性: 表示 $\boldsymbol{x}'$ 是一个自然语言句子的概率。我们使用所有训练数据的句子（包括 $\mathcal{D}^{L}_{\text{xy}}$ 和 $\mathcal{D}^{U}_{\text{x}}$）训练一个基于 LSTM 的语言模型 [176] 来评估 $\boldsymbol{x}'$ 的质量。同时我们采用了句子长度归一化 [177] 来对长句子和短句子进行公平比较，即 $\frac{1}{|\boldsymbol{x}'|}\log \text{LM}(\boldsymbol{x}')$.

我们采用一个权值因子 γ 合并这两方面的评估：

$$\mathcal{R}^{X}_{\text{val}}(\boldsymbol{x}', \boldsymbol{y}) = \gamma \cdot \text{SlotAcc}(\boldsymbol{x}', \boldsymbol{y}) + (1-\gamma) \cdot \frac{1}{|\boldsymbol{x}'|}\log \text{LM}(\boldsymbol{x}'). \tag{5–31}$$

重构收益 $\mathcal{R}^{X}_{\text{rec}}(\boldsymbol{x}, \boldsymbol{x}')$ 度量的是生成的句子 $\boldsymbol{x}'$ 和原始输入句子 $\boldsymbol{x}$ 的相似性。我们采用了 BLEU 得分 [178]：

$$\mathcal{R}^{X}_{\text{rec}}(\boldsymbol{x}, \boldsymbol{x}') = \text{BLEU}(\boldsymbol{x}, \boldsymbol{x}'). \tag{5–32}$$

重构收益 $\mathcal{R}^{Y}_{\text{rec}}(\boldsymbol{y}, \boldsymbol{y}')$ 反映了最后生成的语义形式 $\boldsymbol{y}' = (y^{I\prime}, \boldsymbol{y}^{C\prime})$ 和原始输入 $\boldsymbol{y} = (y^{I}, \boldsymbol{y}^{C})$ 之间的相似性。我们采用了语义槽值对的 F_1 得分以及意图准确率作为重构收益：

$$\mathcal{R}^{Y}_{\text{rec}}(\boldsymbol{y}, \boldsymbol{y}') = \omega \mathbb{I}\{y^{I} = y^{I\prime}\} + (1-\omega)\text{F}_1(\boldsymbol{y}^{C}, \boldsymbol{y}^{C\prime}).$$

其中 $\mathbb{I}$ 是 0/1 指示函数，ω 是加权系数。

5.4 实验与分析

5.4.1 数据集与实验设置

5.4.1.1 数据集

本章提出的方法在两个公开数据集上进行了实验和评测：ATIS 航空信息查询系统数据集（Airline Travel Information Systems） [179] 以及 SNIPS 语义理解标准集 [180]。两个数据集的相关统计信息如表5–2所示。

表 5–2 数据集统计信息

Table 5–2 Dataset statistics.

数据集	# 输入词表	# 训练集	# 校验集	# 测试集	语义槽数量	意图数量
ATIS	950	4478	500	893	83	18
SNIPS	14349	13084	700	700	39	7

5.4.1.2 基线系统

在实验中，我们提出的对偶半监督语义理解将与以下这些基线系统进行对比：

- 有监督学习（*Supervised*）的语义理解仅利用已标注的数据集（$\mathcal{D}_{\text{xy}}^{L}$）进行监督性训练，比如第5.2.1小节提到的 *BLSTM*, *BLSTM-CRF* and *BLSTM-focus* 模型。
- 对偶监督性学习（*Dual supervised learning*） [119] 在上述的标准有监督训练中引入了正反模型相关约束的概率对偶性（一个额外的正则项）。
- 一部分半监督（*semi-supervised*）学习方法结合额外的无监督任务进行多任务学习，比如基于 RNN 的语言模型 [103, 107-109] 或者序列到序列的句子自编码 [110, 111] 就可以额外利用上无标注的句子（$\mathcal{D}_{\text{x}}^{U}$）。在本章节的实验中，我们实现了序列到序列的句子自编码（*sentence auto-encoder*）的方式。
- 传统的伪标签（*pseudo-labeling*，*PL*）学习方法是不包含对偶任务的 [106, 171, 172]，其仅使用预训练的 NLU 模型为无标注的句子（$\mathcal{D}_{\text{x}}^{U}$）生成伪样本，以此实现数据扩充。
- 模板合成（*template synthesis*）方法是一种更基础的数据扩充方式。模板的获取是把每个已标注的句子转换为去词汇化后的形式，比如 "*show me flights from* ⟨FromCity⟩ *to* ⟨ToCity⟩"。随后，我们可以利用无句子对应的语义形式（$\mathcal{D}_{\text{y}}^{U}$）中的语义槽值为每个模板中的语义槽填上相应的值，以此合成出新的样本。

5.4.1.3 实验配置

参数设置与训练细节：在模型输入层，我们采用 400 维的预训练词向量进行初始化，其由 300 维的 Glove 词向量 ① [99] 和 100 维的字符向量 [100] 拼接而成。LSTM 的隐层大小设为 256，BLSTM 层数为一层。超参数 $\alpha, \beta, \gamma, \omega, \delta$ 均设为 0.5，λ 设为 0.25。对于对偶学习方法，集束大小 K 设为 5。除了预训练好的词向

① http://nlp.stanford.edu/data/glove.840B.300d.zip

量部分，其他网络参数均以 $[-0.2, 0.2]$ 之间的均匀分布随机初始化。我们依然采用 Adam [160] 作为优化器，设置学习率为 0.001。在训练阶段，非循环层之间设置 0.5 的 Dropout 率。对所有数据集，批处理的批次大小为 16。梯度最大范数的修剪阈值为 5，同时我们采用 l_2 二范数正则化（系数为 1*e*-5）来避免过拟合。评测指标采用的是语义槽值对的 F_1 得分以及意图类别的句准确率。F_1 得分的计算与 CoNLL 评测脚本 ①保持一致。所有的模型都被训练 50 轮，并保存在校验集上性能表现（语义槽 F_1 得分和意图准确率的平均值）最好的模型参数。

除了利用预训练的词向量，一些最新的预训练语言模型（比如，ELMo [101], BERT [102]）也被用来获取输入词向量，并在后续实验中进行了对比分析。在本章节的实验中，我们采用了预训练的 BERT 模型（*bert-base-cased*），其包括 12 层 768 维的隐层以及每层自注意力机制包括 12 个头 ②。对于该实验，我们将 Adam 的学习率设为较小的 5*e*-5。

半监督学习的数据设置：为了评测不同半监督 NLU 模型的效率和有效性，我们在此进行半监督学习的数据设置。为了模拟真实场景下的数据稀缺问题，一部分原始训练数据被随机选取作为已标注数据（$\mathcal{D}^L_{\mathrm{xy}}$），而其余的训练数据则被作为非配对的句子和语义形式（$\mathcal{D}^U_{\mathrm{x}}$ 和 $\mathcal{D}^U_{\mathrm{y}}$）来扮演未标注数据。对于已标注数据的比例，设置为 $5\%, 10\%, 20\%, 30\%$ 以及 50%。

显著性检验：我们采用了 McNemar's test 方法来计算一个方法对比另一个方法的统计显著性（$p < 0.05$）。

5.4.2 主要结果

我们首先在模拟的半监督学习以及完全监督学习的两种设定下，在 ATIS 和 SNIPS 两个数据集上对比了不同的方法（使用预训练词向量）。表5–3和表5–4展示了基线系统以及我们的方法在两个数据集上的语义槽 F_1 得分和意图准确率。从中我们可以发现：

1. 相对于语义槽填充，意图识别是一个明显更简单的任务。对比仅使用 5% 和 100% 标注数据的两种情况下的性能变化，可以看到意图准确率的变化比语义槽 F_1 得分小很多，尤其是在 SNIPS 数据集上。同时，不同的方法在意图准确率上的差别也非常小（不过最终我们的方法总能得到最好的性能）。
2. 对于有监督（*supervised*）NLU，我们选择 BLSTM-focus 模型作为 NLU 的

① https://www.clips.uantwerpen.be/conll2000/chunking/output.html

② https://github.com/google-research/bert

表 5–3 不同模型在 ATIS 数据集上的语义槽 F_1 得分以及意图准确率。‡ 表示我们的结果显著地优于最好的基线系统性能

Table 5–3 Slot F_1 scores and intent accuracies of different methods on ATIS dataset. ‡ indicates our results that significantly outperform the best baseline.

Method		**Slot F_1** (%)					
		5%	10%	20%	30%	50%	100%
supervised	**BLSTM-focus** (backbone)	82.92	89.48	92.66	93.54	95.45	95.79
	+ dual supervised learning	83.88	89.37	93.21	94.30	95.40	96.03
semi-supervised	+ sentence auto-encoder	83.16	89.65	92.74	94.52	95.36	95.87
	+ pseudo-labeling (PL)	84.75	90.08	94.07	94.91	95.52	95.75
	+ template synthesis	86.10	90.62	94.35	94.94	95.27	-
+ dual task (ours)	+ dual PL	**89.58**‡	93.49‡	94.88‡	**95.90**‡	96.02‡	95.82
	+ dual learning	88.92‡	93.40‡	95.09‡	95.50‡	95.70	96.00
	+ dual PL + dual learning	**89.58**‡	**93.53**‡	**95.37**‡	95.85‡	**96.14**‡	**96.37**‡
Method		**Intent Acc** (%)					
		5%	10%	20%	30%	50%	100%
supervised	**BLSTM-focus** (backbone)	89.03	92.61	94.40	94.85	**98.54**	98.43
	+ dual supervised learning	89.36	92.16	94.96	95.41	98.32	98.54
semi-supervised	+ sentence auto-encoder	88.80	92.50	95.18	94.62	98.32	98.32
	+ pseudo-labeling (PL)	89.47	92.50	95.18	94.85	98.32	98.32
	+ template synthesis	90.05	92.05	94.06	94.51	98.10	-
+ dual task (ours)	+ dual PL	90.37	93.28	94.62	**96.08**‡	98.43	**98.66**
	+ dual learning	89.81	93.28	**95.30**	95.86	**98.54**	98.54
	+ dual PL + dual learning	**90.48**	**93.51**‡	95.18	95.30	**98.54**	98.54

骨干模型，而不是 BLSTM 和 BLSTM-CRF。如表 5–5所示，BLSTM-focus 模型在完全有监督学习的情况下可以取得相对更好的性能。

3. 半监督（*semi-supervised*）NLU 的三个基线系统在大多数情况下都可以利用无标注的数据来提升语义理解性能，其中 "+ *sentence auto-encoder*" 引入了一个序列到序列的句子重构任务，"+ *pseudo-labeling (PL)*" 使用现有的 NLU 模型为无标注的句子生成伪标签，而 "+ *template synthesis*" 则利用已标注数据和无句子对应的语义形式来一起合成更多的样本用于后续有监督训练。
4. 和传统不带对偶任务的伪标签学习方法（+ *PL*），我们提出的对偶伪标签方法（+ *dual PL*）可以利用无句子对应的语义形式获取性能提升。
5. 相对于基线系统，本文提出的对偶学习方法也获得一定提升。而和对偶伪

表 5–4 不同模型在 SNIPS 数据集上的语义槽 F_1 得分以及意图准确率。‡ 表示我们的结果显著地优于最好的基线系统性能

Table 5–4 Slot F_1 scores and intent accuracies of different methods on SNIPS dataset. ‡ indicates our results that significantly outperform the best baseline.

Method		**Slot F_1** (%)					
		5%	10%	20%	30%	50%	100%
supervised	**BLSTM-focus** (backbone)	87.89	91.23	93.54	94.45	94.92	**96.44**
	+ dual supervised learning	88.45	91.13	93.77	94.45	94.89	96.06
semi-supervised	+ sentence auto-encoder	87.83	90.29	93.20	94.68	94.78	95.89
	+ pseudo-labeling (PL)	90.67	91.89	93.89	94.29	95.06	96.00
	+ template synthesis	90.40	92.94	93.94	94.22	94.37	-
+ dual task (ours)	+ dual PL	93.86‡	94.46‡	**95.53**‡	95.23‡	95.29	96.22
	+ dual learning	93.85‡	94.18‡	95.31‡	95.08‡	**95.45**‡	95.86
	+ dual PL + dual learning	**94.00**‡	**94.51**‡	95.22‡	**95.34**‡	95.25	96.11
Method		**Intent Acc** (%)					
		5%	10%	20%	30%	50%	100%
supervised	**BLSTM-focus** (backbone)	97.86	98.14	98.00	98.29	98.71	**99.14**
	+ dual supervised learning	97.00	98.14	98.14	98.14	99.00	**99.14**
semi-supervised	+ sentence auto-encoder	97.57	97.86	97.86	98.00	98.71	98.86
	+ pseudo-labeling (PL)	97.57	98.00	98.14	98.00	99.00	**99.14**
	+ template synthesis	97.86	98.00	**98.57**	98.14	98.86	-
+ dual task (ours)	+ dual PL	**98.57**‡	98.14	98.43	98.43	98.71	**99.14**
	+ dual learning	98.29	98.14	98.29	**98.57**	98.57	98.86
	+ dual PL + dual learning	98.29	**98.43**	**98.57**	98.14	**99.14**	98.86

标签不同，该方法引入了有效性收益和重构收益来给生成的伪样本提供一个软性的样本重要性评估。

6. 最终，我们结合提出的两种方法（如算法5–1所示）可以取得进一步的提升。在大多数情况下，两者的结合（*+ dual PL + dual learning*）可以获得最好的性能，尤其是在语义槽 F_1 得分上。
7. 我们的方法甚至可以在完全有监督的情况（100% 有标注数据）下取得一定的性能提升，比如我们在 ATIS 上的得到了 96.37% 的语义槽 F_1 得分。但是，我们的方法在 SNIPS 上并没有超过完全有监督训练的方法（语义槽 F_1 是 96.44%）。相关原因可能是 ATIS 的测试集中包含更多种类的去词汇化句子，而 SNIPS 的测试集中包含更多的训练集中没见过的语义槽值对，具体数字如图5–6所示。我们提出的方法应用在 100% 有标注数据上时，倾向

表 5–5 在有监督训练情形下，不同语义理解模型（BLSTM，BLSTM-CRF，BLSTM-focus）的对比

Table 5–5 Comparison among BLSTM, BLSTM-CRF and BLSTM-Focus for supervised NLU on ATIS and SNIPS datasets.

Method	**ATIS**		**SNIPS**	
	Slot F_1	Intent Acc	Slot F_1	Intent Acc
BLSTM	95.50	98.21	94.96	98.86
BLSTM-CRF	95.62	98.32	96.34	98.86
BLSTM-focus	**95.79**	**98.43**	**96.44**	**99.14**

表 5–6 相对于训练集的测试集数据分析

Table 5–6 Data analysis of test sets compared with training sets.

Dataset	**#Unseen Delexicalized Form**	**#Unseen Slot-Value Pair**
ATIS	680	169
SNIPS	421	522

于为已有的语义形式生成更具多样性的自然语言句子。SNIPS 的测试集中有很多训练集中没见过的语义槽值对，因此我们的方法难以在 SNIPS 上获得提升。

5.4.3 实验对比与分析

在上一小节中，我们的方法在两个数据集上取得了显著性的提升。而在本小节, 我们将用一些对比实验来分析造成这些提升的潜在因素。我们将依次进行 NLG 模型、对偶伪标签和对偶学习方法的消融研究，逐步分析不同模块的有效性。最后我们将分析预训练语言模型 BERT 对我们方法的影响。

5.4.3.1 NLG 模型的消融研究

为了验证语义理解对偶任务 NLG 模型的有效性，我们在 ATIS 和 SNIPS 数据上进行了消融实验，结果如表5–7所示。其中 BLEU 得分 [178] 是用来衡量生成的句子和真实句子之间相似度的。同时，我们也采用第5.3.2.3小节中的语义槽准确率衡量生成句子的语义完整性。从 "*(-) w/o feeding intent*" 行上的结果我们可以观察到意图对于生成句子的 BLEU 得分是非常重要的，虽然意图识别在语义理解中是一个非常简单的子任务。同样，编码器中的全局层（*global-level*）*BLSTM*，解码器里的拷贝机制以及 *SC-LSTM* 单元也是 NLG 模型中的重要部件.

表 5–7 对偶任务 NLG 模型在有监督训练情形下的消融研究

Table 5–7 Ablation studies of the NLG model for the dual task of NLU, which is supervised by full training sets on ATIS and SNIPS respectively.

Model	**ATIS**		**SNIPS**	
	BLEU	Slot Acc	BLEU	Slot Acc
supervised NLG	47.17	97.72	39.18	100.00
(-) w/o feeding intent	44.86	98.26	38.15	99.70
(-) w/o global BLSTM	41.14	96.15	31.65	98.95
(-) w/o copy mechanism	44.08	96.58	38.67	99.98
(-) w/o SC-LSTM	46.78	97.25	37.99	100.00

表 5–8 在对偶半监督语义理解中 NLG 模型的评测

Table 5–8 Evaluations of the NLG model in the proposed dual semi-supervised NLU.

Method	**ATIS** (10%)		**SNIPS** (5%)	
	BLEU	Slot Acc	BLEU	Slot Acc
supervised NLG	39.53	87.49	29.94	94.85
+ dual PL	40.28	93.90	35.26	98.85
+ dual learning	38.84	91.30	32.19	98.10
+ dual PL + dual learning	41.85	95.10	36.61	99.43

除了完全有监督训练，我们也想进一步知道 NLG 模型本身是否也可以在对偶半监督的语义理解获得性能提升。如表5–8所示，对偶伪标签方法和对偶学习方法都可以提升 NLG 模型的以及 NLU 模型的性能。该实验中我们只选取了 ATIS 和 SNIPS 中各自 10% 和 5% 的有标注训练数据来模拟数据稀缺的情况。

5.4.3.2 对偶伪标签方法的消融研究

对偶伪标签方法由以下两种方式创建伪样本：a）利用 NLU 模型为输入句子预测语义形式；b）利用 NLG 模型为给定的意图和语义槽值对列表生成句子。我们进行了相关实验对比了这两种方式，结果如表5–9所示。从中可以发现，利用 NLG 生成的伪样本会更加重要。原因可能是 NLG 模型倾向于生成和给定的语义形式语义一致的句子，即便这种句子不是很自然流畅。但是，NLU 模型则很可能会预测出错误的语义标签。同时，如果不使用 NLG 生成的伪样本，对偶伪标签方法就相当于退化成了传统的不带对偶任务的伪标签方法了。

从表5–9中 "*(+)* w_i=1" 行的结果来看，按训练轮次慢慢增加系数（w_i）是有帮

表 5–9 对偶伪标签方法的消融研究

Table 5–9 Ablation studies of the dual pseudo-labeling method.

Method	**SNIPS** (5%)	
	Slot F_1	Intent Acc
+ dual PL	93.86	98.57
(-) w/o pseudo-samples from NLU model	93.51	98.00
(-) w/o pseudo-samples from NLG model	90.67	97.57
(+) $w_i = 1$	93.65	98.14
(-) w/o iterative generation	91.49	97.71

表 5–10 对偶学习方法的消融研究

Table 5–10 Ablation studies of the dual learning method.

Method	**SNIPS** (5%)	
	Slot F_1	Intent Acc
+ dual learning	93.85	98.29
(-) w/o unlabeled sentences	92.96	98.14
(-) w/o unexpressed semantic forms	91.41	97.71
(-) w/o validity rewards	91.55	97.71
(-) w/o reconstruction rewards	93.74	97.86

助的。我们相信随着训练的增加，后面的 NLU 和 NLG 模型可以提供更高质量的伪样本。因此，如果我们在训练中一直使用第一次生成的伪样本，语义理解的性能会明显下降，如 "*(-) w/o iterative generation*" 行所示。

5.4.3.3 对偶学习方法的消融研究

同样，我们也进行了一些实验来显示对偶学习方法中不同组成部分的效用，结果如表5–10所示。从 "*(-) w/o unlabeled sentences*" 和 "*(-) w/o unexpressed semantic forms*" 两行的结果上看，可以发现无句子对应的语义形式会更加的重要。这个和对偶伪标签方法里的发现是一致的。这会让半监督语义理解更加的便捷，因为语义形式是高度结构化的，可以通过领域知识简单合成。最后两行则显示了有效性收益和重构收益都非常关键，但是有效性收益对语义槽 F_1 得分的影响更大一些。

表 5–11 基于 BERT 的方法在两个数据集上的语义槽 F_1 值以及意图准确率

Table 5–11 Slot F_1 scores and intent accuracies of BERT-based models on the two datasets.

Method	with BERT	**ATIS** (10%)		**SNIPS** (5%)	
		Slot	Intent	Slot	Intent
BLSTM-focus	✗	89.48	92.61	87.89	97.86
+ dual PL + dual learning	✗	93.53	93.51	94.00	98.29
BLSTM-focus	✓	91.41	93.51	91.53	98.14
+ dual PL + dual learning	✓	94.14	94.29	95.58	98.43

5.4.3.4 BERT 的有效性

除了预训练的词向量，BERT 模型也可以被用来获取输入向量[①]。但是，不同输入向量模型的应用和本章中半监督语义理解的研究是互不影响的。表 5–11展示了 ATIS 和 SNIPS 数据各自仅采用 10% 和 5% 的已标注训练数据的情形下的性能。相关结果显示，使用 BERT 模型可以进一步提升我们对偶半监督语义理解方法的性能，同时也能提升基线系统的性能。尽管 BERT 模型可以缩小两个方法之间的差距，但是我们的方法仍然是显著性优于基线系统的。

5.4.3.5 在完全有监督情况下与现有方法的对比

最终，我们也在 ATIS 和 SNIPS 数据集的完全有监督训练情况下（即 100% 已标注训练数据），对比我们的方法和已有的公开结果，如表5–12所示。我们提出的对偶半监督语义理解方法（*+ dual PL + dual learning*）可以在两个数据集上都获得最好的性能水平，但并不一定显著。我们还可以发现 BERT 对于 ATIS 的性能帮助是小于 SNIPS 的，一个可能的原因是 ATIS 的词表大小比 SNIPS 的小很多。对于 ATIS 数据，我们的方法加上 BERT 在语义槽 F_1 得分上有下降（从 96.4% 到 96.0%），在意图准确率上有提升（从 98.5% 到 99.1%），但整体而言两个指标的平均值还是提升了的。上述结果显示了我们的方法也可以在完全有监督训练情况下起作用。

此外，我们提出的模型是对意图识别和语义槽填充两个子任务独立建模的，依然可以超越很多对两个子任务进行依赖性建模的方法 [59-62]。但我们仍然相信在我们的方法中引入意图识别和语义槽填充两个子任务的依赖性建模肯定还是有价值的。相关课题将留在对语义理解骨干模型的未来工作中。

① 如果输入中的一个词被 BERT 分词器分为多个子词，我们仅考虑第一个子词的 BERT 向量。

表 5–12 在 ATIS 和 SNIPS 上和以往公开结果的对比

Table 5–12 Comparison with previous results of NLU on ATIS and SNIPS.

Method		ATIS		SNIPS	
		Slot	Intent	Slot	Intent
w/o BERT	Joint Seq. [181]*	94.3	92.6	87.3	96.9
	Attention BiRNN [51]*	94.2	91.1	87.8	96.7
	Slot-Gated [61]	95.2	94.1	88.8	97.0
	Self-Attentive Model [60]*	95.1	96.8	90.0	97.5
	CAPSULE-NLU [58]	95.2	95.0	91.8	97.3
	ELMo-Light for SLU [103]	95.4	97.3	93.3	98.8
	SF-ID Network [62]	95.8	97.1	92.2	97.3
	Stack-Propagation [59]	95.9	96.9	94.2	98.0
	our method	**96.4**	98.5	96.1	98.9
w/ BERT	Multi-ling. Joint BERT [104]	95.7	97.8	96.2	99.0
	Joint BERT SLU [105]	96.1	97.5	97.0	98.6
	Stack-Prop. + BERT [59]	96.1	97.5	97.0	99.0
	our method + BERT	96.0	**99.1**	**97.1**	**99.1**

* indicates a result borrowed from Qin et al. [59].

表 5–13 在 CATSLU 数据（四个领域）以语音识别结果为输入的半监督口语理解（已标注数据的选取比例是 5%）

Table 5–13 Semi-supervised spoken language understanding on CATSLU with four domains (the ratio of fully labeled data is 5%).

Method	map		music		video		weather	
	slot F_1	Joint Acc.	slot F_1	Joint Acc.	slot F_1	Joint Acc.	slot F_1	Joint Acc.
BLSTM-focus	74.73	60.01	73.80	37.43	75.62	48.63	67.35	34.89
+ dual PL	**75.86**	**61.22**	**89.09**	**48.22**	**77.80**	**53.75**	**70.71**	**38.76**

5.4.4 在语音识别结果上的实验

考虑到口语模式，我们也在带有语音识别结果（ASR 最佳候选句子）的 CATSLU 数据集（详见第4.4章节）上也进行了半监督式语义理解的性能评测。表5–13的结果显示了本章提出的对偶-半监督式语义理解学习方法对于口语模式下的语音识别结果也是有效的。

表 5–14 在 ATIS 和 OVERNIGHT 数据半监督情形下的语义解析准确率（已标注数据的选取比例是 50%）

Table 5–14 Test accuracies on ATIS and OVERNIGHT in semi-supervised settings (the ratio of fully labeled data is 50%).

Method	ATIS	OVERNIGHT
Attention	78.6	65.4
+ dual learning	**80.6**	**71.5**
Attention + Pointer Net.	84.8	65.2
+ dual learning	**86.2**	**71.4**

5.4.5 在自然语言理解其它任务上的适应性

除了本文关注的任务型语义理解问题（意图识别和语义槽填充），一些其它的自然语言理解任务也会产生结构化的输出（其它结构化的语义形式），比如基于知识库问答的语义解析、语义角色标注等等。因此，在理论上我们提出的对偶半监督学习框架也可以应用到这些任务上。而使用到其它任务，我们仅需要额外调整主任务和对偶任务的模型结构，以及定义不同的收益函数来适应新的任务。

在此，我们展示了一个将本章的对偶半监督框架应用到基于知识库问答的语义解析 [182] 上的例子。根据 Jia 和 Liang [183] 的工作，我们将语义解析的主任务和对偶任务都形式化为序列生成任务（即把语义形式也看成一个序列），并采用序列到序列的 RNN 模型结构（包括注意力机制 [48] 或者指针网络 [158]）。对于有效性收益，我们分别采用了句子和语义形式上的语言模型。对于重构收益，我们则直接采用了原始输入序列在解码器上的对数概率 [116]。

我们的方法在语义解析的两个标准数据集上的进行了评测：ATIS [184] 和 OVERNIGHT [185]。为了模拟半监督学习的设定，我们也随机选取了 50% 的训练样本作为已标注数据，而其他的作为无标注数据（和第5.4.1.3小节类似）。相关实验结果如表5–14所示，其中包含了两个不同骨干模型（“Attention” 和 “Attention + Pointer Net.”）的结果。从中可以发现，我们的方法可以在两个数据集上、不同骨干模型上均获得明显的性能提升。

5.5 本章小结

为了解决单领域的数据稀疏问题，我们介绍了语义理解的对偶任务（即反向任务，“语义到句子” 生成任务），并把它应用到我们提出的对偶半监督语义理解框架中。该方法可以同时利用无标注的句子和无句子对应的纯语义形式生成相应的

伪样本，用于增强当前有限的已标注训练数据。我们提出的对偶半监督语义理解框架包括对偶伪标签和对偶学习两种方法，可以在主任务和对偶任务构成的闭环中合作式优化主模型和对偶模型。在 ATIS 和 SNIPS 两个语义理解标准数据集上的实验表明，在半监督学习设定下我们的方法可以取到非常明显的性能提升，可以有效缓解单领域构建中的数据稀疏问题。同时，即便在完全有监督学习的情形下，我们的方法依然可以发挥作用，并在 ATIS 和 SNIPS 数据集上取得了目前最好的性能水平。

本章利用纯语义形式进行“数据扩充”的方法具有一定的潜在价值：引入纯语义形式数据相比收集大量未标注的句子具有成本更低、效果更优的好处。因为语义形式是高度结构化的，可以通过给定的领域知识（领域本体）自动生成或者合成。另外，我们的对偶半监督学习框架理论上是“骨干模型不可知”的，即可以应用到其他具有类似的结构化语义形式输出的自然语言理解领域。本章的方法在骨干模型的研究、收益函数的设计上还有进一步挖掘的空间。

算法 5–1 对偶-半监督式语义理解学习算法

Input: 已标注训练集 $\mathcal{D}_{\text{xy}}^{L}$；纯（未标注的）句子集 $\mathcal{D}_{\text{x}}^{U}$；纯（无句子对应的）语义形式（即意图类别和语义槽值对列表）集 $\mathcal{D}_{\text{y}}^{U}$；集束大小 K；权值系数 w_i，δ；最大训练轮数 N；

Output: NLU 模型参数 Θ_{NLU} 以及 NLG 模型参数 Θ_{NLG}

在 $\mathcal{D}_{\text{xy}}^{L} \cup \mathcal{D}_{\text{x}}^{U}$ 上训练语言模型 LM(·)；在 $\mathcal{D}_{\text{xy}}^{L} \cup \mathcal{D}_{\text{y}}^{U}$ 上构建词汇数据库 DB(·) 以及意图-语义槽共现矩阵 COM;

在 $\mathcal{D}_{\text{xy}}^{L}$ 上预训练 NLU(·$|\Theta_{\text{NLU}}$) 和 NLG(·$|\Theta_{\text{NLG}}$) 模型，通过最小化各自的训练损失：$\sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}_{\text{xy}}^{L}} \mathcal{L}_{\text{NLU}}(\boldsymbol{x},\boldsymbol{y})$ 以及 $\sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}_{\text{xy}}^{L}} \mathcal{L}_{\text{NLG}}(\boldsymbol{y},\boldsymbol{x})$;

for $i = 1$ *to* N **do**

  repeat

    采样句子 $\boldsymbol{x} \sim \mathcal{D}_{\text{xy}}^{L} \cup \mathcal{D}_{\text{x}}^{U}$;

    采样语义形式 $\boldsymbol{y} \sim \mathcal{D}_{\text{xy}}^{L} \cup \mathcal{D}_{\text{y}}^{U}$;

    /* **Part 1:** 对偶伪标签方法 */

    用当前 NLU 模型为 $\boldsymbol{x}$ 生成伪标签，即 $\boldsymbol{y}' = \text{NLU}(\boldsymbol{x}|\Theta_{\text{NLU}})$;

    用当前 NLG 模型为 $\boldsymbol{y}$ 生成伪句子，即 $\boldsymbol{x}' = \text{NLG}(\boldsymbol{y}|\Theta_{\text{NLG}})$;

    通过最小化 $w_i(\mathcal{L}_{\text{NLU}}(\boldsymbol{x},\boldsymbol{y}') + \mathcal{L}_{\text{NLU}}(\boldsymbol{x}',\boldsymbol{y}))$ 更新 Θ_{NLU};

    通过最小化 $w_i(\mathcal{L}_{\text{NLG}}(\boldsymbol{y},\boldsymbol{x}') + \mathcal{L}_{\text{NLG}}(\boldsymbol{y}',\boldsymbol{x}))$ 更新 Θ_{NLG};

    /* **Part 2:** 对偶学习方法 */

    对 NLU($\boldsymbol{x}|\Theta_{\text{NLU}}$) 采用集束搜索获取 K 个语义形式 $\boldsymbol{y}'^1, \cdots, \boldsymbol{y}'^K$;

    对任意 $\boldsymbol{y}'^k$，在 NLG($\boldsymbol{y}'^k|\Theta_{\text{NLG}}$) 上同样采用集束搜索获取 K 个句子 $\boldsymbol{x}'^{k1}, \cdots, \boldsymbol{x}'^{kK}$;

    计算每个 k 的单独收益 $r_1^k(\boldsymbol{x})$ 以及总体期望值 $\psi_1(\boldsymbol{x})$;

    计算策略梯度 $\nabla_{\Theta_{\text{NLU}}}\psi_1(\boldsymbol{x})$ 和 $\nabla_{\Theta_{\text{NLG}}}\psi_1(\boldsymbol{x})$；// 回环 NLU2NLG

    对 NLG($\boldsymbol{y}|\Theta_{\text{NLG}}$) 采用集束搜索获取 K 个句子 $\boldsymbol{x}'^1, \cdots, \boldsymbol{x}'^K$;

    对任意 $\boldsymbol{x}'^k$，在 NLU($\boldsymbol{x}'^k|\Theta_{\text{NLU}}$) 上同样采用集束搜索获取 K 个语义形式 $\boldsymbol{y}'^{k1}, \cdots, \boldsymbol{y}'^{kK}$;

    计算每个 k 的单独收益 $r_2^k(\boldsymbol{y})$ 以及总体期望值 $\psi_2(\boldsymbol{y})$;

    计算策略梯度 $\nabla_{\Theta_{\text{NLU}}}\psi_2(\boldsymbol{y})$ 和 $\nabla_{\Theta_{\text{NLG}}}\psi_2(\boldsymbol{y})$；// 回环 NLG2NLU

    使用梯度 $\delta\nabla_{\Theta_{\text{NLU}}}\psi_1(\boldsymbol{x}) + (1-\delta)\nabla_{\Theta_{\text{NLU}}}\psi_2(\boldsymbol{y})$ 更新 Θ_{NLU};

    使用梯度 $\delta\nabla_{\Theta_{\text{NLG}}}\psi_1(\boldsymbol{x}) + (1-\delta)\nabla_{\Theta_{\text{NLG}}}\psi_2(\boldsymbol{y})$ 更新 Θ_{NLG};

    /* **Part 3:** 原始有监督训练 */

    采样训练样本对 $(\boldsymbol{x},\boldsymbol{y}) \sim \mathcal{D}_{\text{xy}}^{L}$;

    通过最小化 $\mathcal{L}_{\text{NLU}}(\boldsymbol{x},\boldsymbol{y})$ 和 $\mathcal{L}_{\text{NLG}}(\boldsymbol{y},\boldsymbol{x})$ 各自更新 Θ_{NLU} 和 Θ_{NLG};

  until $\mathcal{D}_{xy}^{L}, \mathcal{D}_{x}^{U}, \mathcal{D}_{y}^{U}$ 中所有的样本均被过一遍;

end

第六章　结合原子模板的跨领域数据迁移

6.1　引言

为了解决单领域的数据稀疏问题，在第五章中我们提出了基于语义理解对偶任务（语义到句子的生成任务）的数据扩充模型。然而，单领域中有限的标注数据同样会限制数据扩充模型的性能。为了在有限标注数据下提高数据扩充模型的能力，一种常见的领域自适应思想是利用其它领域（源领域）的已标注数据来帮助数据扩充模型的训练，从而实现跨领域的数据迁移。但是，不同领域之间的语义类别符号是不一致的（相应的文本数据分布也不一致），如何高效地利用不同领域的数据训练跨领域的 “语义到句子” 生成模型是一个实际要解决的问题。另外，当前领域的数据有限，大部分的语义标签出现频次很低甚至还没有出现过，如何为出现频次低甚至根本没出现在训练数据中的语义表示生成扩充样本也是一个重要的问题。

针对上述的问题，本章提出了一种结合原子模板的跨领域语义数据扩充方法。为了解决不同领域之间语义类别符号不一致的问题，我们首先将语义形式中语义槽值对的自然语言阐述定义为原子模板，比如 “出发城市 = 北京” 可以朴素地阐述为 “从北京出发”；那么语义形式中多个语义槽值对的集合就可以通过原子模板映射为多个短语组成的短语集合。其次，利用源领域和目标领域的标注数据训练一个 “短句集合到完整句子” 的复述生成模型。这样的好处是：

- 将语义槽等类别符号通过原子模板解释为自然语言短语，消除了语义类别符号不一致的问题，同时以短语为输入也提升了相应数据扩充模型的泛化能力。
- 通过复述生成模型的泛化能力的提升，可以实现对稀有甚至在标注数据中没有见过的语义形式生成扩充样本。

我们在 DSTC-2&3 上的实验表明，原子模板作为一种新的领域先验知识，可以有效提升语义数据扩充模型的领域自适应学习能力。

本章的结构如下：我们首先在第6.2节介绍基于原子模板的跨领域数据扩充模型；第6.3节介绍数据扩充模型与语义理解模型的领域自适应训练以及合作优化方法；第6.4节是实验部分，包括实验设置、主要结果与实验分析；第6.5节是本章小结。

6.2 结合原子模板的跨领域数据扩充模型

我们记完整句子为词序列 $\boldsymbol{x} = (x_1, \cdots, x_{|\boldsymbol{x}|})$，并且记语义表示形式为 *act-slot-value* 三元组集合 $\boldsymbol{y} = \{y_1, \cdots, y_{|\boldsymbol{y}|}\}$，即每个 y_i 都是一个三元组。我们期望对给定语义表示形式 $\boldsymbol{y}$ 后生成完整句子 $\boldsymbol{x}$ 的条件概率进行评估，即 $p(\boldsymbol{x}|\boldsymbol{y})$。

但是，直接采用上述语义表示形式作为句子生成模型的输入会有如下缺点：

- 不同垂直领域的语义类别符号的定义可能有冲突，比如相同的语义槽名称可能代表不同的意思。
- 语义类别符号的字符串名称可能无法体现它们的特定意思，比如开发人员可能会使用 “city_1” 和 “city_2” 来表示不同上下文中出现的城市。

因此，这种方式很难将模型 $p(\boldsymbol{x}|\boldsymbol{y})$ 适应到新的语义类别符号上，不利于语义数据扩充方法的领域自适应。

在传统方法中，我们可以为新的语义形式（由多个语义三元组构成的集合）人工设计模板，从而生成相关的自然语言句子。但是这类句子级的模板很难穷举所有可能的语义三元组集合（指数级规模），时间和人力成本都很高。

而在本章，我们提出了更细粒度的原子模板，即单独一个语义三元组对应的短语。传统的句子模板需要给定完整三元组集合（一般是多个）与完整句子的对应关系，模板的规模（复杂度）是指数级的（和句子包含的平均三元组个数有关）。然而，原子模板仅需要为每一个单独的语义三元组提供短语对应关系，其复杂度为线性。但是，通过原子模板为三元组集合映射得到的短语集合与自然流畅的完整句子还有一定差别。因此，我们需要利用已有的标注数据学习 “从短语集合到完整句子” 的句子复述模型。这也是原子模板方法的不平凡的地方。在实际数据扩充阶段，我们可以通过遍历组合多个语义三元组来提升扩充数据的语义多样性。但是需要指明的是，原子模板是人工定义的，原子模板本身的说法丰富性是受限于人工设计的。

如图 6–1所示，基于原子模板的语义数据扩充流程包括两部分：1）利用原子模板将语义表示形式（比如 *act-slot-value* 三元组集合）映射为短语样例集合；2）基于短语样例集合生成完整流畅的句子。在下文中，我们将详细介绍原子模板以及基于短语集合的句子生成模型。

6.2.1 原子模板

我们提出一种新的领域先验知识，将语义类别符号重新用自然语言解释一遍，然后再将语义表示形式的自然语言解释重新复述为完整的句子。由于语义表示形式一般比较复杂，要想对语义表示形式整体进行解释是不切实际的。因此，我们

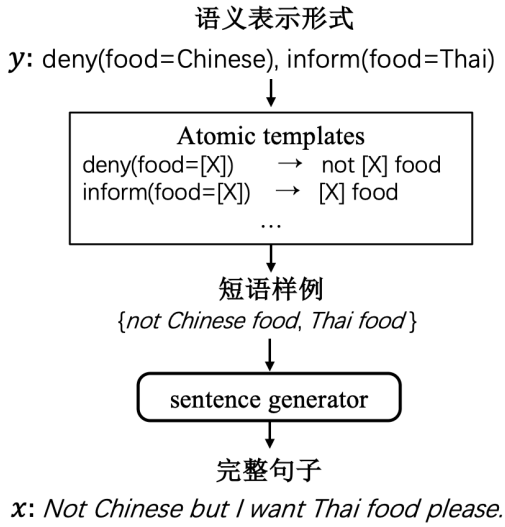


图 6–1 基于原子模板的语义数据扩充流程

Figure 6–1 Workflow of the semantic data augmentation with atomic templates.

表 6–1 DSTC-2&3 数据集中的原子模板样例，其中 “[X]” 表示相应语义槽的任意可能的取值

Table 6–1 Examples of atomic templates in DSTC-2&3 dataset. “[X]” is an arbitrary value of corresponding slot.

Triplet	Templates
bye()	goodbye bye
request(addr)	the address what’s the address
inform(food=[X])	[X] [X] food
inform(hastv=true)	television with a television

提出对语义表示形式的子结构进行解释，即对每个单独的 *act-slot-value* 三元组采用一些原子模板来描述它。

表 6–1给出了一些在 DSTC-2&3 数据集上的原子模板示例。原子模板可以为每一个单独的 *act-slot-value* 三元组 y_i 生成相应的简单短语描述 e_i。如果 y_i 可以适用于多个原子模板，那么这个过程可以产生多个简单短语描述 $\mathrm{AT}(y_i)$。在训练阶段，我们会选择和输入句子最相近的一个简单短语，即 $e_i = \arg\max_{e \in \mathrm{AT}(y_i)} \mathrm{sim}(e, \boldsymbol{x})$；而在测试阶段（即语义数据扩充阶段），我们则从 $\mathrm{AT}(y_i)$ 中随机挑选一个简单短语

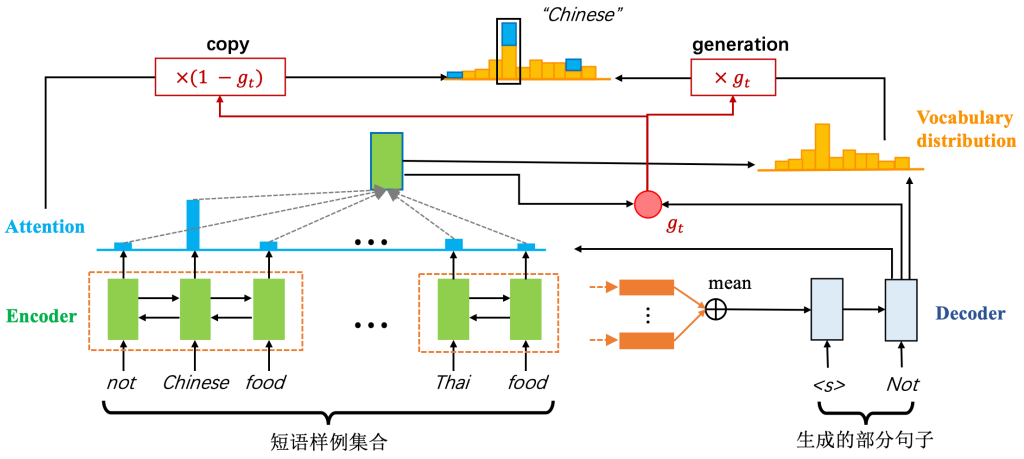


图 6–2 从短语集合到完整句子的句子生成模型架构

Figure 6–2 Architecture of the sentence generator from atomic exemplars to a complete sentence.

描述 e_i。我们采用模糊字符串匹配算法（Ratcliff-Obershelp 算法 [186]）作为上述 $\text{sim}(e, \boldsymbol{x})$ 的相似度计算。于是，经过原子模板的短语映射后，句子生成模型的条件概率公式可以替换为如下形式：

$$p(\boldsymbol{x}|\boldsymbol{y}) = p(\boldsymbol{x}|\{y_1, \cdots, y_{|\boldsymbol{y}|}\}) = p(\boldsymbol{x}|\{e_1, \cdots, e_{|\boldsymbol{y}|}\}) \tag{6–1}$$

6.2.2 句子生成模型

我们采用编码器-解码器框架来为上述短语样例的集合生成完整流畅的句子，即对 $p(\boldsymbol{x}|\{e_1, \cdots, e_{|\boldsymbol{y}|}\}$ 进行建模。具体模型结构如图6–2所示。

6.2.2.1 编码器

首先，因为短语样例的集合是内部无序的，我们对每个短语独立编码。记第 i 个短语样例为一个词序列 $e_i = (w_{i1}, \cdots, w_{iT_i})$，其中 T_i 是序列长度。那么，我们可以使用一个共享的 BLSTM 模型对这些短语样例进行编码。首先每个词都会被映射成一个词向量，即 $\mathbf{w}_i = \mathbf{W}_{\text{in}}\mathbf{o}(w_i)$，其中 $\mathbf{W}_{\text{in}} \in \mathbb{R}^{d_e \times |\mathcal{V}_{\text{in}}|}$ 是一个词向量矩阵，$\mathbf{o}(w_i) \in \mathbb{R}^{|\mathcal{V}_{\text{in}}|}$ 是一个独热（one-hot）向量，$\mathcal{V}_{\text{in}}$ 表示输入字典，d_e 为词向量维度，$\mathbf{w}_i \in \mathbb{R}^{d_e}$。将这些词向量序列输入 BLSTM（计算过程见附录A.1），在每个时刻可以得到一个隐向量 $\mathbf{h}_{ij} \in \mathbb{R}^{2d_h}$（$d_h$ 是 LSTM 隐层大小，$j \in \{1, \cdots, T_i\}$）：

$$(\mathbf{h}_{i1}, \cdots, \mathbf{h}_{iT_i}) \leftarrow \text{BLSTM}_{\Theta_1}(\mathbf{w}_{i1}, \cdots, \mathbf{w}_{iT_i}) \tag{6–2}$$

其中 $\mathbf{w}_{ij}$ 为相应的词向量。此外，我们使用正向和反向 LSTM 的终止时刻隐向量作为当前短语的短语向量。

在对集合中所有短语样例编码后，我们可以得到一系列的隐向量：

$$H(\boldsymbol{y}) = [\mathbf{h}_{11}, \cdots, \mathbf{h}_{1T_1}; \cdots; \mathbf{h}_{|\boldsymbol{y}|1}, \cdots, \mathbf{h}_{|\boldsymbol{y}|T_{|\boldsymbol{y}|}}] \tag{6–3}$$

$$= [\mathbf{h}_1, \cdots, \mathbf{h}_M] \tag{6–4}$$

其中 $M = \sum_{i=1}^{|\boldsymbol{y}|} T_i$。

6.2.2.2 解码器

在编码器中我们利用一个 LSTM 模型 [187] 逐词生成相应的完整句子 $\boldsymbol{x}$。该 LSTM 解码网络在第 t 时刻的隐向量由此计算

$$\mathbf{d}_t = \mathrm{f}_{\mathrm{LSTM}}(\mathbf{x}_{t-1}, \mathbf{d}_{t-1}) \tag{6–5}$$

其中 $\mathbf{x}_{t-1}$ 是上一时刻预测出来的词的向量表示，$\mathbf{d}_t$ 是隐向量。解码器的隐层向量由所有短语表示向量平均得到，即

$$\mathbf{d}_0 = \frac{1}{|\boldsymbol{y}|} \sum_{i=1}^{|\boldsymbol{y}|} (\overrightarrow{\mathbf{h}}_{iT_i} \oplus \overleftarrow{\mathbf{h}}_{i1}) \tag{6–6}$$

其中 $\oplus$ 表示向量拼接操作。

在 LSTM 的基础上，我们采用了一个包含注意力机制 [168] 和拷贝机制 [158] 的输出层来预测 $\boldsymbol{x}$ 中的每个词。在解码器第 t 时刻，其关于编码器中每个隐向量的注意力权重以及上下文向量 $\mathbf{c}_t$ 的计算如下

$$\mathbf{c}_t, (a_{t1}, \cdots, a_{tM}) \leftarrow \mathrm{ATTN}_{\Theta_2}(\mathbf{d}_t, (\mathbf{h}_1, \cdots, \mathbf{h}_M)). \tag{6–7}$$

该注意力机制的定义见附录A.2。然后通过一个线性层可以得到在输出词表上的概率分布

$$p_{\mathrm{gen}}(x_t | \boldsymbol{x}_{<t}, \boldsymbol{y}) = \mathrm{softmax}_{x_t}(\mathbf{W}_o(\mathbf{d}_t \oplus \mathbf{c}_t) + \mathbf{b}_o) \tag{6–8}$$

其中 $\mathbf{W}_o \in \mathbb{R}^{|\mathcal{V}_{\mathrm{in}}| \times 3d_h}$，$\mathbf{b}_o \in \mathbb{R}^{|\mathcal{V}_{\mathrm{in}}|}$。当预测得到序列终止符 “</s>” 的时候生成过程停止。

除了上面直接预测生成的方式外，解码器还包括了拷贝机制来提升模型的泛化能力，即可以从输入中直接拷贝短语词汇。为了融合两种方式，我们使用一个 sigmoid 门控函数 σ 在生成和拷贝直接做一个软性选择：

$$p(x_t | \boldsymbol{x}_{<t}, \boldsymbol{y}) = g_t p_{\mathrm{gen}}(x_t | \boldsymbol{x}_{<t}, \boldsymbol{y}) + (1 - g_t) p_{\mathrm{copy}}(x_t | \boldsymbol{x}_{<t}, \boldsymbol{y}) \tag{6–9}$$

$$g_t = \sigma(\mathbf{v}_g^{\top}(\mathbf{d}_t \oplus \mathbf{c}_t) + \mathrm{b}_g) \tag{6–10}$$

其中 $g_t \in [0,1]$ 是一个平衡因子，$\mathbf{v}_g$ 是参数向量以及 b_g 是偏置。拷贝机制的预测 $p_{\text{copy}}(\cdot|\cdot)$ 是定义在 M 个短语词汇 $(w_1, \cdots, w_M)$ 上的分布:

$$p_{\text{copy}}(x_t|\boldsymbol{x}_{<t}, \boldsymbol{y}) = \sum_{m=1}^{M} \mathbb{I}\{x_t = w_m\} a_{tm} \tag{6–11}$$

其中 $\mathbb{I}\{x_t = w_m\}$ 指示函数对于 x_t 和 w_m 相同的情况返回 1，否则返回 0。

最终，从短语集合到完整句子的句子生成模型的训练损失函数为

$$\mathcal{L}_{\text{NLG}}(\boldsymbol{y}, \boldsymbol{x}) = -\sum_{t=1}^{|\boldsymbol{x}|} \log p(x_t|\boldsymbol{x}_{<t}, \boldsymbol{y}) \tag{6–12}$$

6.2.3 与其他方法的对比讨论

在以往的方法中，Hou 等 [124] 用一个序列到序列的生成模型来建模同义句之间的翻译过程。然而，该方法无法对训练集中没见过的语义形式生成新的样本。Yoo 等 [125] 将句子和语义标注组成一个样本序列，然后利用变分自编码器学习样本序列的自动生成。但该方法同样很难为训练集中没见过的语义形式生成数据。其次，上述方法都在尝试对句子和语义形式的联合概率 $p(\boldsymbol{x}, \boldsymbol{y})$ 进行建模，从而利用该模型生成更多带标注的样本。但这类方法有两个缺点：1）首先它们无法控制生成样本的语义形式进行定向数据扩充；2）其次它们无法为包含未登录语义标签（即在训练数据中没出现过的语义标签）的语义形式生成样本。本章采用的基于条件概率的语义增强模型 $p(\boldsymbol{x}|\boldsymbol{y})$ 可以利用各种语义标签的组合作为输入，实现语义形式更多样性的数据扩充，而且原子模板作为一种先验知识可以很好地解决无法为训练集中没见过的语义标签生成数据的问题。

6.3 数据扩充模型与语义理解模型的领域自适应

数据扩充模型与语义理解模型的领域自适应训练流程如图6–3所示。在没有数据扩充方法的情况下，语义理解模型 $p(\boldsymbol{y}|\boldsymbol{x})$ 的领域自适应训练一般为在源领域数据 $\mathcal{D}_{src}$ 上进行预训练，然后在目标领域数据 $\mathcal{D}_{tgt}$ 继续微调模型参数，如图6–3(a)所示。相应地，语义数据扩充模型 $p(\boldsymbol{x}|\boldsymbol{y})$ 也需要类似的领域自适应训练，惟一的区别是两个模型的输入输出是相反的，如图6–3(b) 所示。此外，给定目标领域的原子模板，我们可以利用训练好的语义数据扩充模型（具体是从短语集合到完整句子的句子生成模型），结合大量目标领域的语义形式作为输入从而生成相应的文本数据。最后，在给定目标领域扩充数据的情形下，我们采用两次微调的策略

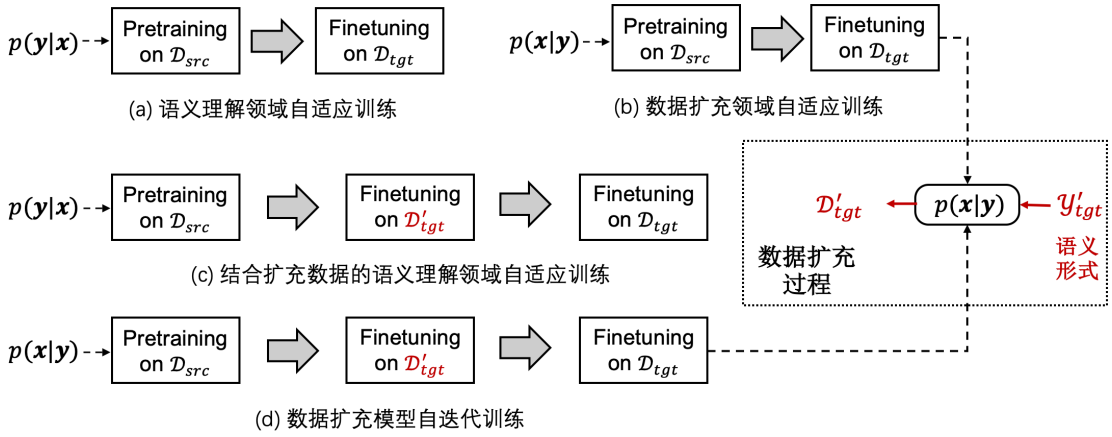


图 6–3 数据扩充模型与语义理解模型的领域自适应训练流程

Figure 6–3 Workflow of domain adaptations of data augmentation and language understanding models.

对语义理解模型进行自适应训练，如图6–3(c) 所示。因为生成的数据可能包含错误、句子不够流畅自然，所以我们采用先在目标领域的生成数据上微调，然后在目标领域的少量标注数据上再次微调的“两次微调策略”。

在数据扩充过程中，我们需要准备很多待生成相应句子的语义形式 $\mathcal{Y}'_{tgt}$。由于语义形式是高度结构化的一种数据形式，我们可以通过多种方式合成它。本文提供了两种可行的方式：

1. **重采样法**：利用目标领域已有的少量标注数据，对其中的语义形式进行重新采样，比如一个语义形式包含三个语义三元组 {A,B,C}，我们可以进行某种下采样获取它的子集，即 {A,B}, {B,C}, {A,C}, {A}, {B}, {C} 作为新的语义形式。
2. **随机组和法**：我们从领域本体中随机挑选最多 N_c（比如设为 3）个语义三元组构成一个新的语义形式。

6.3.1 数据扩充模型的自迭代训练

由于一般目标领域的数据非常有限，那么通过数据扩充方法生成的伪样本也同样可以被用来强化数据扩充模型本身的训练，如图6–3(d) 所示。首先，我们如图6–3(b) 所示，利用源领域和目标领域训练数据预训练好数据扩充模型。然后我们利用数据扩充模型为目标领域的语义形式 $\mathcal{Y}'_{tgt}$ 一一生成相应的句子，共同组成新的扩充样本集。最后，我们将该扩充样本集引入到源领域和目标领域训练数据重新训练增强模型。如此反复迭代，直至模型收敛，其大致流程如算法6–1所示。

算法 6–1 数据扩充模型的自迭代训练

Input: 源领域训练数据 $\mathcal{D}_{src}$；目标领域训练数据 $\mathcal{D}_{tgt}$；目标领域的纯语义形式集 $\mathcal{Y}'_{tgt}$；最大迭代轮数 N；

Output: 数据扩充模型 $p(\boldsymbol{x}|\boldsymbol{y};\theta)$ 以及扩充数据集 $\mathcal{D}'_{tgt}$

在 $\mathcal{D}_{src} \rightarrow \mathcal{D}_{tgt}$ 上领域自适应训练数据扩充模型 $p(\boldsymbol{x}|\boldsymbol{y};\theta^{(0)})$;

for $i = 1$ *to* N **do**

  对目标领域所有的纯语义形式，生成相应的句子，即 $\boldsymbol{x}' = \arg\max_{\boldsymbol{x}} p(\boldsymbol{x}|\boldsymbol{y}';\theta^{(i-1)}), \forall \boldsymbol{y}' \in \mathcal{Y}'_{tgt}$;

  将生成的伪样本作为扩充数据集 $\mathcal{D}'_{tgt} = \{(\boldsymbol{x}', \boldsymbol{y}')|\forall \boldsymbol{y}' \in \mathcal{Y}'_{tgt}\}$;

  在 $\mathcal{D}_{src} \rightarrow \mathcal{D}'_{tgt} \rightarrow \mathcal{D}_{tgt}$ 上领域自适应训练数据扩充模型 $p(\boldsymbol{x}|\boldsymbol{y};\theta^{(i)})$;

end

6.4 实验与分析

6.4.1 实验数据与设置

6.4.1.1 数据集

我们的实验采用了第二届与第三届国际对话跟踪挑战赛的公开数据，即 DSTC-2&3 [63, 188]。其中 DSTC-2（源领域）数据集包括了非常充足的训练数据，大约有 15000 多句和餐馆预约相关的对话句子。而 DSTC-3（目标领域）数据集则是专门为 DSTC-2 设计的领域迁移任务，其涉及了包括餐馆预约在内的旅游信息查询领域。但是，DSTC-3 只提供了非常少量的训练数据（种子数据），总计 109 句话。该数据集不仅提供每句话的人工转写文本，也提供了语音识别后的 N 最佳候选列表。

为了模拟在目标领域上有充足训练数据的情况，我们将 DSTC-3 的测试数据随机分为两半，其中一半用作模拟充足的训练数据（`dstc3_oracle_train`），另一半保留作为测试数据。数据集相关的统计信息如表6–2所示。

原子模板构建：我们为去词表化的 *act-slot-value* 三元组构建原子模板，比如表6–1中 “inform(food=[X])”。在 DSTC-2 和 DSTC-3 中各自分别有 41 和 35 个去词表化的语义三元组，其远远小于完整语义形式的数量。对于每一个去词表化的语义三元组，我们平均有两个原子模板将其解释为自然语言短语。

目标领域的额外语义形式获取：如第6.3小节介绍的那样，我们有两种方法（重采样法和随机组合法）获取目标领域的额外语义形式①。其中重采样法需要利用 `dstc3_seed` 种子数据，而随机组合法则需要 DSTC-3 的领域本体随机选择最多 3

① 相应数据划分、原子模板以及额外语义形式公开在了：https://github.com/sz128/DAAT_SLU

表 6–2 DSTC-2&3 数据集的统计信息

Table 6–2 Dataset statistics of DSTC-2&3 dataset.

数据集	领域	语义槽数量	数据划分	数据量
DSTC-2	餐馆预约（源领域）	8	dstc2_train	15611
DSTC-3	旅游信息（目标领域）	13	dstc3_seed	109
			dstc3_oracle_train	9466
			dstc3_test	9249

个语义三元组进行组合。相对而言，暴力的随机组合法的语义形式覆盖量会更大。在实际实验中，我们利用重采样法为 DSTC-3 合成了 1420 个语义形式，利用随机组合法则合成了 20670 个语义形式。通过原子模板映射和句子生成模型，我们可以为这些语义形式生成（贪心解码）相应的句子，从而得到扩充数据。

6.4.1.2 模型配置

语义理解模型：本章实验是在语义理解任务上检验数据扩充模型的好坏，于是针对 DSTC-2&3 数据集是非对齐语义标注的特性，我们采用了第3.3节基于层级解码的非对齐式口语理解模型。此外，为了提高语义理解模型领域自适应训练的效果，我们还利用语义槽描述的平均词向量作为语义槽表示，用于层级解码模型的语义槽分类器中。

语义理解模型和数据扩充中的句子生成模型都使用了一样的超参数。其中，词向量层采用的是 100 维的 *Glove*[①]预训练词向量，LSTM 的隐层大小均为 128。除了预训练好的词向量部分，其他网络参数均以 $[-0.2, 0.2]$ 之间的均匀分布随机初始化。我们依然采用 Adam [160] 作为优化器，设置学习率为 0.001。在训练阶段，非循环层之间设置 0.5 的 Dropout 率。训练数据批处理的批次大小为 20。我们在训练集上训练 50 轮，并保存在校验集上性能表现（*act-slot-value* 三元组 F_1 得分[②]）最好的模型参数。最终我们报告该模型参数在目标领域测试集上的 F_1 得分以及相应的准确率（Precision）和召回率（Recall）。为了提供句子生成模型的泛化能力，我们在输入的短语样例集合上采用了目标词 Dropout 的技术。

没有扩充数据时，语义理解模型在 dstc2_train 上预训练，然后在 dstc3_seed 上微调参数。数据扩充中的句子生成模型同样先在 dstc2_train 上预训练，然后再在 dstc3_seed 上微调。在有扩充数据时，语义理解模型在

① http://nlp.stanford.edu/data/glove.6B.zip

② http://camdial.org/~mh521/dstc

表 6–3 DSTC-3 测试集上不同系统的语义理解性能

Table 6–3 Performances of different methods on DSTC-3 test set.

Method	Precision	Recall	F1-score
NO	80.9	76.9	78.8
NAIVE	84.9	81.6	83.2
AT (重采样)	88.0	86.0	87.0
AT (随机组合)	90.2	89.0	89.6
AT (重采样 + 随机组合)	89.4	86.9	88.1
HUMAN	91.5	89.0	90.2
ORACLE	96.9	97.0	96.9

`dstc2_train` 上预训练后先用扩充数据微调，最后再在 `dstc3_seed` 上微调优化。相关流程同图6–3一致。训练好的语义理解模型最终在 `dstc3_test` 上进行评测。

6.4.1.3 对比方法

我们对比了如下的语义数据扩充的基线方法和上限方法：

- NAIVE: 该方法是通过替换槽值的方式为目标领域训练数据（`dstc3_seed`）生成更多的样本。我们将同时出现在输入句子和语义形式中的值随机替换为相应语义槽的其他可能的值。该方法随机替换，直到目标领域本体中每一个槽值至少出现了三遍。
- HUMAN: 这是人工数据扩充的方法，即人为地将源领域的句子和目标领域的本体结合，为目标领域构建完整的已标注数据。
- ORACLE: 假设目标领域数据充足的理想情况，即数据扩充方法的上限。在本实验中我们利用 `dstc3_oracle_train` 模拟充足的训练数据。

6.4.2 主要结果

6.4.2.1 人工转写文本上的结果

表6–3展示了不同数据扩充方法在目标领域（DSTC-3）上的语义理解性能。我们可以发现：

1. 首先，目标领域非常稀缺的训练数据 `dstc3_seed` 限制了目标领域语义理解模型的性能。当我们仅采用最简单的数据扩充办法（NAIVE）都可以取得明显的性能提升。

表 6–4 在语音识别识别输出上的语义理解性能（三元组 F_1 得分）

Table 6–4 Triplets F_1 scores of language understanding with ASR hypotheses.

Method	manual			1-best			10-best		
	all	seen	unseen	all	seen	unseen	all	seen	unseen
NO	78.8	86.3	44.3	70.6	78.2	33.9	72.2	79.2	36.0
NAIVE	83.2	89.6	55.8	74.0	81.4	41.1	75.5	82.1	44.0
AT (重采样)	87.0	90.6	70.0	78.2	82.7	56.3	79.2	83.5	57.7
AT (随机组合)	89.6	91.6	**80.6**	80.2	83.2	**65.9**	81.1	84.1	**66.5**
HUMAN	**90.2**	**94.5**	71.1	**80.6**	**85.5**	57.3	**81.4**	**86.1**	58.2
ORACLE	96.9	97.9	92.8	85.7	88.0	75.6	86.6	88.7	77.1

2. 我们利用原子模板（atomic templates，AT）的数据扩充方法相比 NAIVE 办法也可以明显地提升语义理解性能。一个原因是我们的方法可以为非常多样性的语义形式生成样本，而 NAIVE 办法仅替换了一些同类型的槽值。
3. 随机组合法相对于重采样法有明显的性能优势，因为重采样法受限于目标领域种子数据 `dstc3_seed` 的大小。同时，我们发现两种方法产生的数据合集并没有进一步提升，可能是因为重采样法的的扩充数据中有一些质量较低的样本。
4. 我们方法的最好性能与人工进行数据扩充的方法非常接近（F_1 得分，89.6% 相对于 90.2%），这说明我们的方法是非常有效的。而且，原子模板构建的人工成本也远远低于直接进行人工数据扩充。

6.4.2.2 在语音识别（ASR）输出上的结果

考虑到口语模式，我们在语音识别输出上也进行了语义理解的性能评测，实验结果如表6–4所示。其中 manual 指人工转写文本，1-best 指 ASR 输出的最佳候选句子，10-best 指 ASR 输出的 10 最佳候选列表。除了 DSTC-3 测试集上所有语义三元组的 F_1 得分（all），我们还把测试集中的语义三元组分为两部分计算指标：一部分为在 `dstc3_seed` 中出现过的（`seen`），另一部分为没有出现过的（`unseen`）。以此观察不同数据扩充方法生成的额外数据对目标领域的语义覆盖度。从表6–4中的结果可以发现：

- 语音识别错误会对语义理解的性能造成极大的负面影响。10-best 包含更多的不确定信息，可以表现的比 1-best 更好。
- 不同数据扩充方法在人工转写文本以及语音识别输出上相对优劣都是一致的。在人工转写文本上的扩充数据也可以帮助提升口语理解的性能以及对

表 6–5 语义理解领域自适应训练的消融实验

Table 6–5 Ablation study about domain adaptive training of language understanding.

Method	Precision	Recall	F1-score
AT (随机组合)	90.2	89.0	89.6
(-) w/o dstc2_train	88.5	84.1	86.2
(-) w/o dstc3_seed	87.0	82.1	84.5
(-) w/o dstc2_train & dstc3_seed	88.2	80.7	84.3

表 6–6 基于原子模板的数据扩充模块的消融实验

Table 6–6 Ablation study of atomic template based data augmentation.

Method	Precision	Recall	F1-score
AT (随机组合) (-) w/o dstc2_train & dstc3_seed	88.2	80.7	84.3
(-) w/o 句子生成模型	73.7	74.3	74.0
(-) w/o 原子模板	74.6	66.8	70.5

ASR 错误的稳健性。

- 数据扩充方法的确生成了很多 dstc3_seed 种没有的语义数据；而且我们可以发现基于原子模板和随机组合的方法在生成新的语义数据方面更有优势，具体表现为在 unseen 上的 F_1 得分大大超过了人工的方法（HUMAN）。

6.4.3 实验对比与分析

6.4.3.1 语义理解领域自适应训练的消融实验

在表6–5中，我们进行了一系列对比实验来验证语义理解领域自适应训练的有效性。我们尝试去除源领域的数据（“(-) w/o dstc2_train”），以及去除目标领域的训练数据 “(-) w/o dstc3_seed”），相应的结果也验证了源领域训练数据和目标领域种子数据对于语义理解领域自适应学习都至关重要。最终，如果两个数据都不用，只使用数据扩充方法生成的目标领域数据，性能是最低的。

6.4.3.2 基于原子模板的数据扩充模块的消融实验

接在上述语义理解的消融实验，我们在只使用数据扩充方法生成的数据训练语义理解模型的情况下，继续分析数据扩充模块的有效性。如表6–6所示，如果我们去除句子生成模型，即直接采用短语样例集合拼接成的词序列作为 “句子”，可

表 6–7 数据扩充模型的自迭代学习

Table 6–7 Self-iterative learning for the model of data augmentation.

Method	Precision	Recall	F1-score
AT (随机组合)	90.2	89.0	89.6
(+) 自迭代学习	92.1	87.6	89.8
(+) 数据过滤	91.1	90.0	90.6
(+) 自迭代学习	**92.4**	**90.1**	**91.3**
HUMAN	91.5	89.0	90.2

以发现语义理解性能有很大下降。因为原子模板的短语样例距离自然流畅的完整句子还是有很大区别的，而这也验证了句子生成模型的必要性和有效性。另外，我们也尝试去除了原子模板，即利用句子生成模型直接将 *act-slot-value* 三元组集合转换为一个完整句子。我们同样可以发现明显的性能下降，这说明了原子模板可以极大地提升句子生成模型领域自适应的泛化能力。另外，用短语样例拼接的“句子”甚至比从三元组生成的句子表现更好，这是因为短语样例拼接的句子虽然不流畅但语义的关键信息还有存在的，但是句子生成模型如果表现不好的话会很容易生成“文不达意”的句子。

6.4.3.3 数据扩充模型的自迭代学习

我们同样对算法6–1的数据扩充模型的自迭代学习算法进行了验证，结果如表6–7所示。如果我们利用生成的所有扩充样本进行自迭代学习，效果提升并不明显（F_1 得分从 89.6% 到 89.8%）。这可能是扩充样本中存在一些噪音的干扰，即少量扩充样本的语义完整性存在不足。于是我们对扩充样本进行过滤，即如果输入语义槽的值没有全部包含在句子中的话，我们将这类情况的生成句子直接替换为短语样例的简单拼接（即这类情况下回退到不使用句子生成模型的方法）。从结果中可以发现，过滤了语义完整性不足的扩充样本后，语义理解的性能有了进一步的提升（F_1 得分从 89.6% 到 90.6%）。在该前提下继续结合自迭代学习，最终 F_1 得分可以达到 91.3%，并超越了人工数据扩充的方法。

6.4.3.4 目标领域训练数据量的影响

目标领域的训练数据量对于语义数据扩充模型以及语义理解模型的领域自适应训练都有很重要的作用，图6–4展示了不同方法下目标领域（DSTC-3）训练数据量对语义理解性能的影响。我们可以发现即便在目标领域不提供训练数据的情

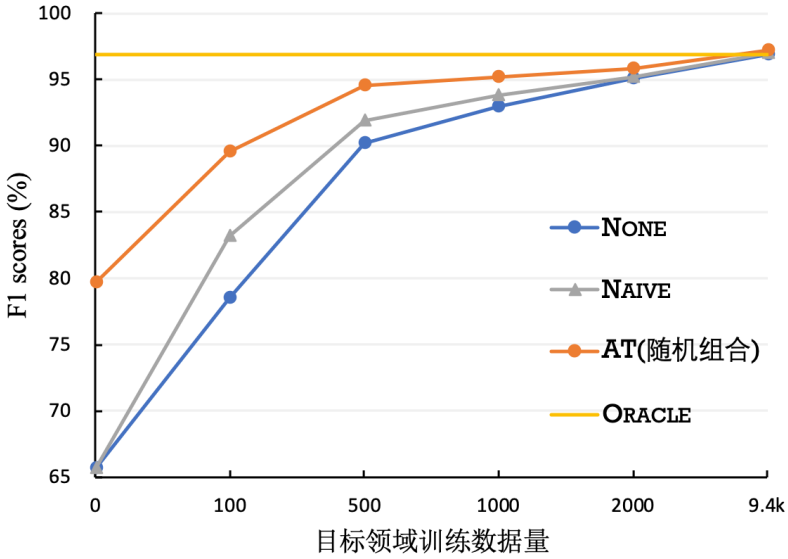


图 6–4 不同方法下目标领域（DSTC-3）训练数据量对语义理解性能的影响

Figure 6–4 LU performance of different methods with varying number of training samples of DSTC-3.

形下，我们基于原子模板的数据扩充方法仍然可以取得非常好的性能，具有良好的泛化能力。而随着目标领域训练数据量的增加，我们的方法始终保持着有效性。即使在训练数据最充足的情况下我们的方法还可以相比理想情况有细微的提升。

6.4.3.5 案例分析

表6–8展示了目标领域 DSTC-3 的部分扩充数据样例。对于一些在源领域中常见的语义元组（比如 “thankyou()”，“request(phone)”），我们的方法可以生成非常自然流畅的句子。但我们也发现一些表现不好的案例，比如表6–8中最后两行。其中的原因可能是 “request(childrenallowed)” 和 “request(hastv);request(addr)” 这两组语义形式都没有在目标领域的种子数据 `dstc3_seed` 中出现过，句子生成模型的泛化能力还有提升的空间。但是，这样的句子作为语义理解的训练数据还是有价值的。

6.5 本章小结

本章提出了一种结合原子模板的跨领域数据迁移方法。为了解决不同领域之间语义类别符号不一致的问题，我们首先将语义形式中语义槽值对集合通过原子模板映射为短语样例集合。其次，利用源领域和目标领域的数据训练一个 “短句

表 6–8 目标领域 DSTC-3 的扩充数据样例

Table 6–8 Examples of generated data samples for DSTC-3.

语义形式	原子模板映射后的短语样例	生成句子
thankyou() bye()	thank you good bye	Thank you good bye.
request(phone) request(addr)	what is the phone whats the address number	What is the address and phone number?
request(phone) request(addr)	the phone number the address	The address and phone number.
inform(type=coffeeshop) inform(hasinternet=true)	cafe has internet connection	I'm looking for a cafe has the internet connection.
affirm() inform(pricerange=moderate) inform(type=restaurant)	yes moderately priced looking for a restaurant	Yes, I'm looking for a moderately priced restaurant.
request(childrenallowed)	does it allow children	Does it have children?
request(addr) request(hastv)	the address does it have a television	Does it have the television address and address?

集合到完整句子" 的句子生成模型。原子模板消除了语义类别符号不一致的问题，同时以短语为输入也提升了相应数据扩充模型的泛化能力。我们在 DSTC-2&3 上的实验表明，原子模板作为一种新的领域先验知识，可以有效提升语义数据扩充模型的领域自适应学习能力。实验分析也验证了基于原子模板的语义数据扩充方法可以为标注数据中没有见过的语义形式生成扩充样本。

第七章　基于标签分布式表征的语义理解领域迁移

7.1　引言

目前几乎所有的任务型口语对话系统都不是通用的，而是基于有限的垂直领域（比如音乐播放、天气预报、影视搜索、餐馆预订、机票查询等等）组合而成的。在实际场景中，任务型口语理解同样要面对大量的垂直领域（任务）。在大规模任务构建中，系统往往已经积累了一定的已有领域。这样对于新领域的构建，我们可以把它看成一个语义理解模型的跨领域迁移问题。前面第五章单领域的数据稀疏问题侧重解决没有其它领域辅助知识的情况，而本章侧重大规模任务构建中的领域迁移问题。因为在实际情况里，我们往往只在少量重点领域拥有充足的训练数据，而很多低频领域或者新开发的领域却处于数据缺乏的状态。因此，在大规模任务构建中，如何利用大量其他领域（源领域）的数据来帮助当前领域（目标领域）的构建成为了一个主要问题。

为了解决上述问题，本章从领域自适应学习出发，研究如何利用已有领域（源领域）的数据构建新领域（目标领域）的语义理解模型。我们提出了一种基于标签分布式表征的语义理解模型，利用先验知识中蕴含的语义标签相关性实现不同领域之间标签信息的共享，解决了因为不同领域之间标签不一致导致输出层参数无法完全共享的问题。我们在标签分布式表征中引入了多种先验知识 (原子概念、语义槽描述、语义槽标注样例、少样本支撑集)，并提出相应标签分布式表征的编码方法。在多个数据集上的领域迁移实验表明，该方法在目标领域仅提供少量数据的情形下相比以往的领域自适应方法有明显提升。同时，利用少样本支撑集的方法为领域通用的语义理解模型提供了初步的研究思路：我们以少样本学习为切入点，为匹配式网络提出了一种基于向量投影的 “词-标签” 相似度计算方法。在语义槽填充和命名实体识别任务上，我们的方法相比以往的少样本学习方法均取得了明显的性能提升。

本章的结构如下：我们首先在第7.2节回顾一下语义理解领域自适应的传统方法；在第7.3节介绍面向语义理解自适应的标签分布式表征；在第7.4节介绍基于标签分布式表征的语义理解模型自适应，包括引入先验知识的标签分布式表征方法以及相应的先验知识编码方法；第7.5节是实验部分，包括实验设置、主要结果与实验分析；第7.6节是本章小结。

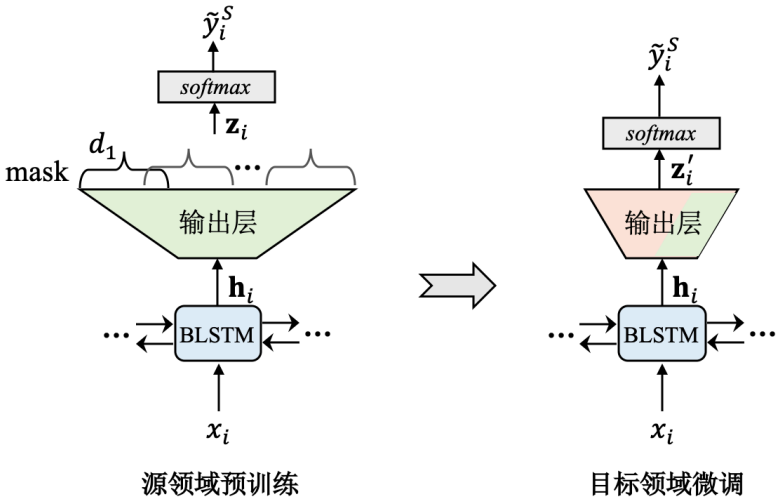


图 7–1 语义理解领域自适应的传统方法

Figure 7–1 A conventional method of NLU domain adaptation.

7.2 语义理解领域自适应的传统方法回顾

在本小节我们先大致回顾一下传统的语义理解领域自适应学习方法，即没有标签分布式表征的方法。利用源领域数据的领域自适应方法可以有效缓解目标领域的数据稀疏问题，比如特征扩充方法 [189]，参数共享方法 [148-150] 以及专家集成方法 [190, 191]。其中，共享不同领域的模型参数是一种更为常见的将源领域知识迁移到目标领域的技术。

假设我们有一个源领域集合 $\boldsymbol{d} = \{d_1, d_2, \cdots, d_M\}$ 以及每个源领域的训练数据 $\boldsymbol{D} = \{\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_M\}$，$M$ 为源领域数量。而记目标领域为 d'，目标领域的训练数据为 $\mathcal{D}'$。本章我们依然关注基于序列标注的语义槽填充任务，并记输入句子（词序列）为 $\boldsymbol{x} = (x_1, x_2, \cdots, x_{|\boldsymbol{x}|})$，以及它的语义槽标记输出序列为 $\boldsymbol{y}^S = (y_1^S, \cdots, y_{|\boldsymbol{x}|}^S)$。那么，任何领域的数据都是由一系列的 “词序列-语义槽标记序列” 构成，即 $\mathcal{D} = \{(\boldsymbol{x}^{(j)}, \boldsymbol{y}^{S(j)})\}_{j=1}^{|\mathcal{D}|}$，$|\mathcal{D}|$ 表示该数据集样本数量。如图7–1所示，领域自适应分为两个步骤：

- **源领域预训练**：首先我们将源领域的训练数据混合起来，训练一个多领域的语义槽标记模型。对于不同的目标领域，这个预训练过程是一次性的。
- **目标领域微调**：然后，将预训练的模型在目标领域中继续更新和微调参数。

表 7–1 不同领域之间的语义槽标注纠纷

Table 7–1 Examples about conflicts of data annotation from different domains.

Domain	Data Samples
DSTC-2	I'm looking for a [Thai$_{Food}$] restaurant.
DSTC-3	I'm looking for a [Thai$_{Food}$] [restaurant$_{Type}$].
Weather	我要去 [北京 $_{City}$]，那里的天气怎么样？
Travel	我要去 [北京 $_{ToCity}$]，那里的天气怎么样？

7.2.1 源领域预训练

如第4.2.1所述，我们可以使用各种典型的语义槽标记模型（BLSTM，BLSTM-CRF，BLSTM-focus）对任务型语义理解 $p(\boldsymbol{y}^S|\boldsymbol{x})$ 进行建模。本章以 BLSTM 为例，首先利用 BLSTM 对句子进行编码（BLSTM 计算过程见附录A.1），在每个时刻得到一个隐向量 $\mathbf{h}_i \in \mathbb{R}^{2d_h}$（$d_h$ 是单个 LSTM 的隐层大小，$i \in \{1, \cdots, |\boldsymbol{x}|\}$）：

$$(\mathbf{h}_1, \cdots, \mathbf{h}_{|\boldsymbol{x}|}) \leftarrow \text{BLSTM}_{\Theta_1}(\mathbf{x}_1, \cdots, \mathbf{x}_{|\boldsymbol{x}|}) \tag{7–1}$$

其中 $\mathbf{x}_i$ 表示第 i 个词的词向量。最后，我们在每个时刻上应用一个线性输出层预测相应的语义槽标签。

但是在源领域预训练阶段，不同领域不能完全共享输出层的参数，因为不同领域的语义槽标签集合是不同的。而且，不同领域所包含的语义槽标签集合代表的是一个闭合的有限语义空间，每个领域对于集外的语义槽是处于不可知的状态。表7–1给出了一些不同领域之间的语义槽标注纠纷的例子。

因此，在输出层上我们需要针对当前领域遮掩掉其它不相关语义槽标签的预测结果。我们记所有源领域组成的语义槽标签并集为 $\mathcal{V}_{\text{tag}}^{src} = \bigcup_{m=1}^{M} \mathcal{V}_{\text{tag}}^{(d_m)}$，其中 $\mathcal{V}_{\text{tag}}^{(d)}$ 表示领域 d 里所有语义槽标记的词汇表。因为不同领域之间可能存在部分相同的语义槽标签，所以 $|\mathcal{V}_{\text{tag}}^{src}| \leq \sum_{m=1}^{M} |\mathcal{V}_{\text{tag}}^{(d_m)}|$。于是在源领域预训练阶段，对于领域 d 的训练数据，线性输出层的计算公式如下：

$$\mathbf{z}_i = \mathbf{W}_o^{src}\mathbf{h}_i + \mathbf{m}^{(d)}. \tag{7–2}$$

其中 i 表示输入序列的第 i 个时刻位置，输出层权值矩阵为 $\mathbf{W}_o^{src} \in \mathbb{R}^{|\mathcal{V}_{\text{tag}}^{src}| \times 2d_h}$，$\mathbf{m}^{(d)}$ 为领域 d 的输出掩码（mask）向量（即忽略非本领域的标签）。掩码（mask）向量 $\mathbf{m}^{(d)} \in \mathbb{R}^{|\mathcal{V}_{\text{tag}}^{src}|}$ 的定义如下：

$$\mathbf{m}_j^{(d)} = \begin{cases} 0, & \text{如果 } \mathcal{V}_{\text{tag}}^{src}[j] \in \mathcal{V}_{\text{tag}}^{(d)}; \\ -\infty, & \text{否则。} \end{cases} \tag{7–3}$$

其中 $\mathcal{V}_{\text{tag}}^{src}[j]$ 表示 $\mathcal{V}_{\text{tag}}^{src}$ 的第 j 个语义标签，$j = 1, 2, \cdots, |\mathcal{V}_{\text{tag}}^{src}|$。我们用它来忽略领域 d 之外的语义槽标签。

于是，在源领域预训练阶段，我们的目标是最大化所有源领域数据上真实输出序列的后验概率。相关训练损失函数为：

$$\mathcal{L}_{src} = -\sum_{m=1}^{M} \sum_{(\boldsymbol{x},\boldsymbol{y}^S)\in\mathcal{D}_m} \log p(\boldsymbol{y}^S|\boldsymbol{x}) = -\sum_{m=1}^{M} \sum_{(\boldsymbol{x},\boldsymbol{y}^S)\in\mathcal{D}_m} \sum_{i=1}^{|\boldsymbol{x}|} \log \text{softmax}_{y_i^S}(\mathbf{z}_i) \tag{7–4}$$

其中 $\text{softmax}_{y_i^S}(.)$ 则是输出分布上对应于 y_i^S 标签的概率。

7.2.2 目标领域微调

在目标领域微调阶段，首先我们拷贝预训练好的模型用于初始化。词向量、BLSTM 编码器的参数可以完全拷贝；但是由于目标领域和源领域也存在标签集不一致的现象，输出层权值矩阵只有部分可以拷贝，如图7–1所示。因为目标领域 d' 经常会包括很多源领域里没有的语义槽标签，部分拷贝的公式化表示如下：

$$[\mathbf{W}_o^{(d')}]_j = \begin{cases} [\mathbf{W}_o^{src}]_k, & \exists k \in \{1, \cdots, |\mathcal{V}_{\text{tag}}^{src}|\} \text{满足} \mathcal{V}_{\text{tag}}^{src}[k] = \mathcal{V}_{\text{tag}}^{(d')}[j] \\ \text{随机初始化}, & \text{否则。} \end{cases} \tag{7–5}$$

其中 $j = 1, 2, \cdots, |\mathcal{V}_{\text{tag}}^{(d')}|$，$\mathbf{W}_o^{(d')} \in \mathbb{R}^{|\mathcal{V}_{\text{tag}}^{(d')}|\times 2n}$ 为目标领域的输出层权值矩阵，$\mathcal{V}_{\text{tag}}^{(d')}$ 是目标领域 d' 的语义槽标签合集，$[\mathbf{W}_o^{(d')}]_j$ 表示输出层矩阵的第 j 个行向量。从而在目标领域上，模型输出层计算公式为：

$$\mathbf{z}_i' = \mathbf{W}_o^{(d')}\mathbf{h}_i \tag{7–6}$$

相应地，目标领域预训练阶段的微调训练损失函数为：

$$\mathcal{L}_{tgt} = -\sum_{(\boldsymbol{x},\boldsymbol{y}^S)\in\mathcal{D}'} \log p(\boldsymbol{y}^S|\boldsymbol{x}) = -\sum_{(\boldsymbol{x},\boldsymbol{y}^S)\in\mathcal{D}'} \sum_{i=1}^{|\boldsymbol{x}|} \log \text{softmax}_{y_i^S}(\mathbf{z}_i'). \tag{7–7}$$

7.3 面向语义理解自适应的标签分布式表征

7.3.1 从独热向量到分布式表征

对于任何领域 d，上述语义理解模型的输出层预测的是每个语义槽标签 $y^S \in \mathcal{V}_{\text{tag}}^{(d)}$ 的置信度得分，即：

$$[\mathbf{z}_i]_{\text{id}(y^S)} = (\mathbf{W}_o^{(d)}\mathbf{h}_i)^\top \mathbf{o}(y^S), \tag{7–8}$$

其中 $\mathrm{id}(y^S) \in \{1, 2, \cdots, |\mathcal{V}_{\mathrm{tag}}^{(d)}|\}$ 是 y^S 在标签集 $\mathcal{V}_{\mathrm{tag}}^{(d)}$ 里的序号；$\mathbf{W}_o^{(d)} \in \mathbb{R}^{|\mathcal{V}_{\mathrm{tag}}^{(d)}| \times 2d_h}$ 是和领域 d 的标签预测相关的权值矩阵，$\mathbf{o}(y^S)$ 是语义槽标签 y^S 的独热向量（one-hot vector）。如果我们对输出层权值矩阵做一个低秩分解（low-rank decomposition），比如 $\mathbf{W}_o^{(d)} = \mathbf{B}^\top \mathbf{C}$，其中 $\mathbf{B} \in \mathbb{R}^{K \times |\mathcal{V}_{\mathrm{tag}}^{(d)}|}$ 以及 $\mathbf{C} \in \mathbb{R}^{K \times 2d_h}$。那么公式7–8可以被重新写为

$$[\mathbf{z}_i]_{\mathrm{id}(y^S)} = (\mathbf{B}^\top \mathbf{C}\mathbf{h}_i)^\top \mathbf{o}(y^S) = (\mathbf{C}\mathbf{h}_i)^\top (\mathbf{B}\mathbf{o}(y^S)), \tag{7–9}$$

其中 $\mathbf{B}$ 相当于语义槽标签的向量（标签分布式表征）查表矩阵，而 $\mathbf{C}$ 是输入特征的线性变换矩阵，K 为标签向量的维度。两者旨在把输出标签和输入隐向量映射到一个相同的空间上，最终通过标签分布式表征向量和输入表征向量的内积得到相应标签的预测得分。

上述内容引出了标签分布式表征的概念，这很容易让人联想到词嵌入（词向量） [97, 98]。词嵌入方法同样是将词的独热向量（one-hot vector）映射到连续向量空间，并可以在向量空间体现词之间的相互关系（近义、反义等）。但不同的是，词向量可以依托大量的纯文本数据进行预训练，然后在其它具体的自然语言处理任务（比如文本分类、语义槽标记等）上进行微调，并可以有效缓解相关任务的数据稀疏问题（特征空间）。其中大量的纯文本数据就相当于一种先验知识，提供了词与词之间先验的相互关系（比如共现关系）。

但是由于任务型对话系统中的语义槽标签是人为主观设计的符号，几乎不存在于现实的纯文本数据中。对于一个新的语义槽标签（即不在源领域的标签向量查表矩阵中），我们只能无根据地随机初始化它的标签向量。因此，为了引入不同语义槽标签的先验关系，本章提出的标签分布式表征也需要先验知识的驱动。本章提及的任务型语义理解中的标签分布式表征，简单来说是标签的向量式表示，但要求标签的向量表示中蕴含标签的相互关系。

我们旨在通过标签分布式表征实现标签的信息共享。假设我们已经拥有能反映标签关系（比如相似、相反的关系）的标签分布式表征，那么从源领域学习的语义槽标记模型或许可以直接应用到目标领域并预测训练集中没有见过的标签。这会有助于减轻目标领域的数据稀疏问题（标签空间），并和预训练词向量（特征空间）形成互补。

7.3.2 结合标签分布式表征的语义理解模型

在本小节，我们将会介绍结合标签分布式表征的语义理解模型，相关模型大致结构如图7–2所示。首先，对于任何领域 d，我们假设其先验知识是已知的，以及针对该类型先验知识的标签编码模型（Label Encoder）也是具备的（我们将在

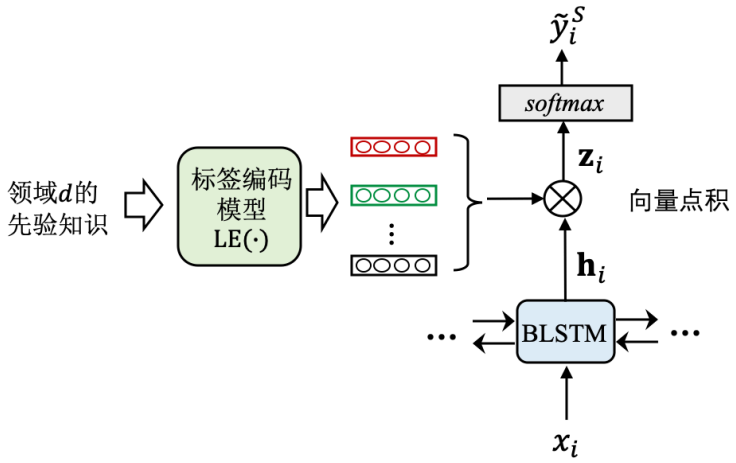


图 7–2 基于标签分布式表征的语义槽标记模型

Figure 7–2 Label embeddings based slot tagging model

下一节具体阐述基于不同类型先验知识的标签分布式表征获取方法）。我们将标签分布式表征获取过程抽象为

$$\mathbf{y}^S = \mathrm{LE}(y^S) \tag{7–10}$$

其中 $y^S \in \mathcal{V}_{\mathrm{tag}}^{(d)}$ 为领域 d 中任意一个语义标签，$\mathbf{y}^S \in \mathbb{R}^K$。

对于任意领域（无论是源领域还是目标领域）$d \in \boldsymbol{d} \cup \{d'\}$，我们可以获取它的标签分布式表征矩阵 $\mathbf{B}_d \in \mathbb{R}^{K \times |\mathcal{V}_{\mathrm{tag}}^{(d)}|}$，该矩阵的每个列向量都是相应语义槽标签的向量表示，即 $[\mathbf{B}_d^\top]_{\mathrm{id}(y^S)} = \mathrm{LE}(y^S)$。这样，源领域和目标领域的输出层计算公式（公式7–2和7–6）可以统一为

$$\mathbf{z}_i = \mathbf{B}_d^\top \mathbf{C} \mathbf{h}_i \tag{7–11}$$

其中 $\mathbf{C} \in \mathbb{R}^{K \times 2d_h}$ 是输出层的线性变换参数。

因此，该方法下所有的模型参数（句子编码器、标签编码器）都是领域无关的，可以在不同领域之间完全共享。这也使得从源领域到目标领域的领域自适应变得更加简单快捷，不同领域仅需要准备各自私有的标签分布式表征矩阵即可。

7.4 先验知识驱动的标签分布式表征获取方法

本小节将详细阐述上面提到的标签分布式表征获取方法，即 LE(·)。关于驱动标签分布式表征获取的先验知识，我们探索了四种领域知识，分别是：原子概念，语义槽描述，语义槽标注样例，以及少样本支撑集。

因为在语义槽标记任务中，每个词上的标签是基于 IOB（In/Out/Begin）格式的。我们对该标签和语义槽做一下区分：对于语义槽标签 y^S（比如 “B-from_city”），

表 7–2 语义槽表示为原子概念的样例

Table 7–2 Examples of slot representation by atomic concepts.

语义槽	原子概念
city	{*city_name*}
from_city	{*city_name*, *from_location*}
depart_city	{*city_name*, *from_location*}
arrive_airport	{*airport_name*, *to_location*}
city_of_birth	{*city_name*, *birth*}
date_of_birth	{*date*, *birth*}
deny-to_city	{*city_name*, *to_location*, *deny*}

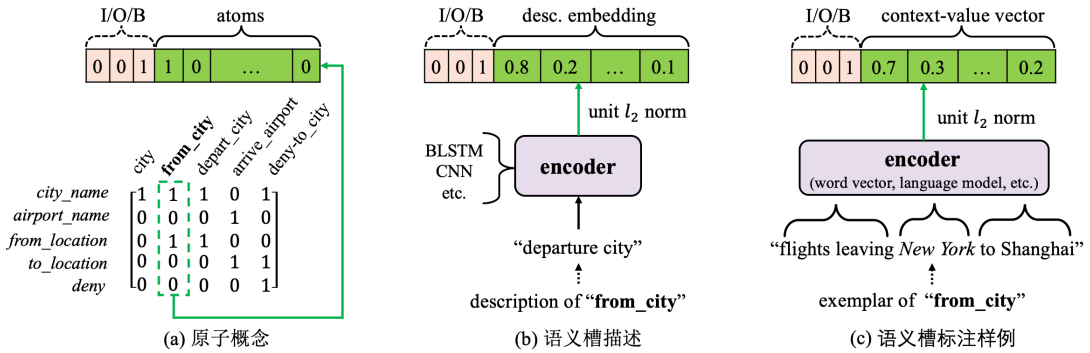
我们可以通过 IOB(y^S) 获取 IOB 符号（比如 “B”），并通过 SLOT(y^S) 获取它的语义槽名字（比如 “from_city”）。对于特殊情况 SLOT(“O”) 的语义槽名称为空。

7.4.1 原子概念

原子概念（atomic concepts）假设每个语义槽可以细分为更小的语义单元，即每个语义槽被表示为一个由多个原子概念组成的集合。原子概念可以被认为是更细粒度的语义槽，会比语义槽更加通用。表7–2给出了一些语义槽表示为原子概念的样例，比如语义槽 `from_city` 可以表示为原子概念集合 {*from_location*, *city_name*}，以及语义槽 `date_of_birth` 可以被表示为 {*date*, *birth*}。

我们期望把语义槽细粒度化之后，可以在原子概念的层面上体现不同语义槽之间的关系，从而驱动不同语义槽之间的数据复用。但在本章中，原子概念并不是可以自动获取的，而是需要领域专家或者开发人员进行人为的设计。一般情况下，原子概念可以被分为两大类：值感知原子和上下文感知原子。值感知原子一般是一些值的类别名称，比如 “New York” 是城市名（*city_name*）。上下文感知的原子则反映了当前语义槽值对周围词的隐式依赖，比如 *to_location* 往往依赖于上下文中有没有 “going to”，“fly to” 以及 “arrive at” 这类的词出现。原子概念的建模有助于发现相关语义槽之间的共享的语言模式，从而在一定程度上缓解数据稀缺的问题。

我们假设，通过领域专家或者开发人员的设计，存在一个同时囊括源领域和目标领域的、通用的原子概念词汇表 $\mathcal{V}_{\text{atom}}$。那么对于任意领域 d，每个语义槽 s 都可以被表示为一个原子概念的结合，即 $\text{atoms}(s) = \{a_1, \cdots, a_{M_s}\}, a_l \in \mathcal{V}_{\text{atom}}$，$M_s$ 是该语义槽所包含的原子概念的数量。因此，每个语义槽都可以被表示为一个 $|\mathcal{V}_{\text{atom}}|$



(a) 原子概念　(b) 语义槽描述　(c) 语义槽标注样例

图 7–3 不同先验知识驱动的标签分布式表征，包括原子概念、语义槽描述以及语义槽标注样例

Figure 7–3 Three different label embeddings consisted of IOB symbol and slot vector: (a) Atomic concept, (b) Slot description encoding and (c) Slot exemplar encoding.

维的 0/1 向量 $\mathbf{b}(\text{atoms}(s))$[①]。由于 IOB 的标注格式，标签分布式表征向量包括两部分：3 维的 IOB 独热向量（one-hot vector）以及 0/1 原子向量，如图7–3 (a) 所示。语义标签 $y^S \in \mathcal{V}_{\text{tag}}^{(d)}$ 的标签分布式表征为：

$$\text{LE}(y^S) = [\mathbf{o}(\text{IOB}(y^S)); \mathbf{b}(\text{atoms}(\text{SLOT}(y^S)))] \tag{7–12}$$

其中 $\text{LE}(y^S) \in \mathbb{R}^{3+|\mathcal{V}_{\text{atom}}|}$，即标签分布式表征维度 $K = 3 + |\mathcal{V}_{\text{atom}}|$。

原子概念的组合可以为语义槽关系提供一个显式的解释。但是，原子概念的构建需要很多的人力去设计，并且随着领域数量的增加，维护一份统一通用的原子概念表是非常难的。

7.4.2 语义槽描述

我们假设每个语义槽都有一个基于自然语言的文本描述，比如 `deny-to_city` 可以被描述为 "not this arrival city"。这类先验知识同样需要领域专家或者开发人员的预先定义，但是工作量会比原子概念低很多，即非结构化的语义槽文本描述会比前面的原子概念更容易获取。由于一些相关语义槽的描述文本很可能会出现一些相同、近义或者反义的词语，描述文本可以反映不同语义槽之间的关系。因此，我们可以从语义槽描述（slot description）中直接抽取标签分布式表征。

不同于原子概念，我们需要一个序列编码器将不定长的语义槽描述文本转换为一个固定维度的向量，如图7–3(b) 所示。记每个语义槽 s 的文本描述为一个 $\boldsymbol{T}_s$

① $\mathbf{b}(\text{atoms}(\text{SLOT}(\text{"O"})))$ 为全 0 向量。

长的词序列，$\text{desc}(s) = (w_1, \cdots, w_{T_s})$。我们分别应用了三种序列编码模型来获取标签分布式表征：平均词向量，BLSTM 以及 CNN 模型。下文将依次介绍这三种编码方法的细节。

平均词向量：首选，语义槽描述里的每个词都被映射为 d_e 维的词向量，即 $\mathbf{w}_t = \mathbf{W}_{\text{in}}\mathbf{o}(w_t)$。然后语义槽的向量表示为描述文本 $\text{desc}(s)$ 所有词的平均词向量：

$$\text{mean}(\text{desc}(s)) = \text{unit}(\frac{1}{T_s}\sum_{t=1}^{T_s}\mathbf{w}_t) \tag{7–13}$$

其中 $\text{unit}(\mathbf{v}) = \frac{\mathbf{v}}{\|\mathbf{v}\|}$ 是为了获取单位化的表示向量。向量单位化的做法有助于提升模型的泛化能力，因为输入句子和语义槽描述是不同质的数据。

BLSTM 编码器：更进一步，我们可以利用 BLSTM 对上述描述文本的词向量序列进行编码，并计算给个词所对位置的隐向量：

$$(\mathbf{h}_1^{\text{desc}}, \cdots, \mathbf{h}_{T_s}^{\text{desc}}) \leftarrow \text{BLSTM}_{\Theta_{\text{desc}}}(\mathbf{w}_1, \cdots, \mathbf{w}_{T_s}) \tag{7–14}$$

我们将两个方向 LSTM 的终止时刻的隐向量拼接，然后作为当前语义槽的向量表示：

$$\text{BLSTM}(\text{desc}(s)) = \text{unit}(\overrightarrow{\mathbf{h}}_{T_s}^{\text{desc}} \oplus \overleftarrow{\mathbf{h}}_1^{\text{desc}}). \tag{7–15}$$

CNN 编码器：CNN 也是将一个词序列映射为固定维度向量的典型模型 [30]。我们在 k 个连续词的窗口上应用一个线性变换来获取新的特征（在这里我们设 $k = 3$）。为了充分考虑序列边缘的词，我们使用零向量在词序列两端进行补充（padding）。过完所有的窗口后，我们可以获得新的特征序列，并采用最大池化（max pooling）操作 [192] 将新的特征序列转换为固定维度的特征向量 $\mathbf{c}$，即最终的语义槽向量表示：

$$\text{CNN}(\text{desc}(s)) = \text{unit}(\mathbf{c}). \tag{7–16}$$

根据上述三种方法，我们可以将语义槽文本描述编码为一个固定维度的向量。因此，每个语义槽标签 $y^S \in \mathcal{V}_{\text{tag}}^{(d)}$ 的标签分布式表示向量为

$$\text{LE}(y^S) = [\mathbf{o}(\text{IOB}(y^S)); \text{ENCODER}(\text{desc}(\text{SLOT}(y^S)))] \tag{7–17}$$

其中 ENCODER 可以是 mean，BLSTM 以及 CNN。记 $\text{LE}(y) \in \mathbb{R}^K$，$K$ 标签分布式表征维度，并注意 $\text{ENCODER}(\text{desc}(\text{SLOT}(\text{“O”}))) = \mathbf{0}$。

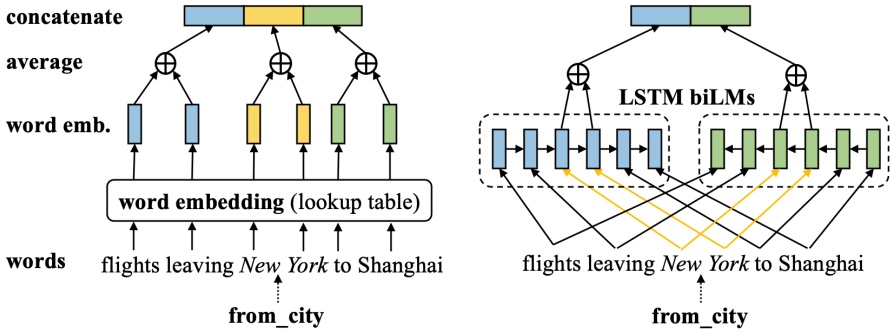


图 7–4 两种对语义槽标注样例编码的方法，包括结构化词向量以及语言模型两种方法

Figure 7–4 Two methods to encode data exemplar of a slot, including (a) Word embeddings and (b) Language models.

7.4.3 语义槽标注样例

在该方法中，我们提出将语义槽的标注样例（slot exemplar）作为提取标签分布式表征的先验知识。考虑到语义槽标注由当前的槽值以及它周围的词组成，我们分层级有结构地从中抽取标签分布式表征，如图7–3(c) 所示。一个语义槽 s 的标注样例由一句话 $\boldsymbol{e} = (e_1, \cdots, e_{|e|})$ 以及槽值的起始、终止位置 (sp, ep) 组成，其中 $\boldsymbol{e}_{sp:ep}$ 就是 s 的取值。比如语义槽 `from_city` 的一个标注样例为 “flights leaving New York to Shanghai”，以及在 $(3, 4)$ 位置的 “New York” 就是它的取值。我们这里介绍两种对语义槽标注样例进行编码的方法，它们分别使用了预训练的词向量和语言模型。一些以往方法也选择从数据样本中提取标签分布式表征 [193-195]，但它们都只关注于当前的槽值而忽略了周围词构成的上下文信息。

基于词向量的方法：首先样例句子中的每个词都被映射为词向量，即 $\mathbf{e}_j = \mathbf{W}_e \mathbf{o}(e_j)$，其中 $\mathbf{W}_e$ 一个词向量矩阵（它可以是和语义理解模型输入层共享的，也可以是其他预训练的词向量）。为了显式地区分槽值和它的上下文，我们将样例句子分为三部分：前文，值和后文，如图7–4(a) 所示。这三部分的词向量被分别平均以及拼接，最后构成语义槽的向量表示：

$$\text{emb}(s) = \text{unit}(\text{avg}(\mathbf{e}_1, \cdots, \mathbf{e}_{sp-1}) \oplus \text{avg}(\mathbf{e}_{sp}, \cdots, \mathbf{e}_{ep}) \oplus \text{avg}(\mathbf{e}_{ep+1}, \cdots, \mathbf{e}_{|e|})) \quad (7\text{–}18)$$

其中当 $sp = 1$ 和 $ep = |e|$ 的时候，前文和后文分别为零向量。

基于语言模型的方法：不同于词向量方法，基于 LSTM 的双向语言模型 (biLMs) [196] 则可以直接将上下文信息编码到隐向量中。利用正向和反向的 LSTM，我们可以得到输入句子对应的两个方向的隐向量序列，$(\vec{\mathbf{u}}_1, \cdots, \vec{\mathbf{u}}_{|e|})$ 和 $(\overleftarrow{\mathbf{u}}_1, \cdots, \overleftarrow{\mathbf{u}}_{|e|})$)，如图7–4(b) 所示。因为隐向量中已经包含了前后文的信息，于是我

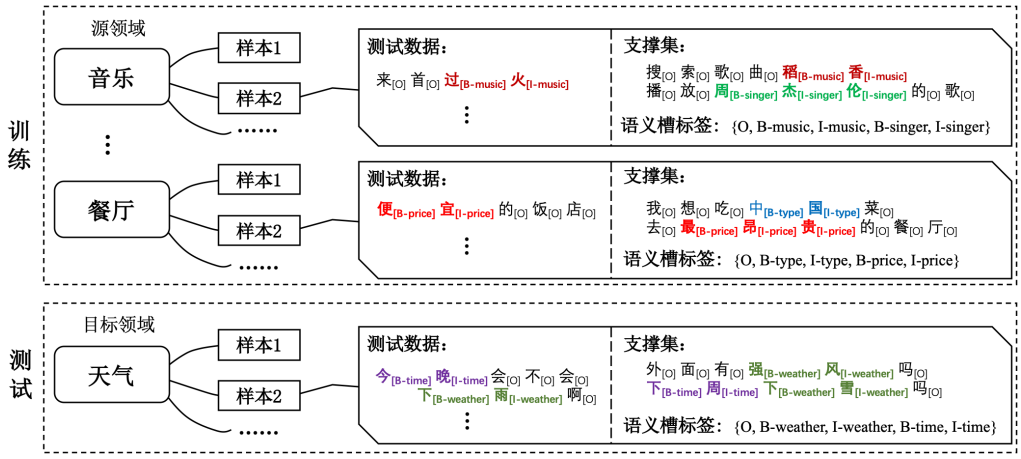


图 7–5 基于少样本学习的语义理解训练和测试概括

Figure 7–5 Overview of training and testing for few-shot slot tagging

们使用从 sp 到 ep 位置的隐向量构建相应的语义槽向量表示：

$$\mathrm{emb}(s) = \mathrm{unit}(\mathrm{avg}(\vec{\mathbf{u}}_{sp}, \cdots, \vec{\mathbf{u}}_{ep}) \oplus \mathrm{avg}(\overleftarrow{\mathbf{u}}_{sp}, \cdots, \overleftarrow{\mathbf{u}}_{ep})) \tag{7–19}$$

根据上述两种方法，我们可以通过语义槽标注样例获取语义槽的向量表示。如果一个语义槽有多个样例，则继续对多个样例获取的向量取平均。因此，我们可以得到语义槽标签 $y^S \in \mathcal{V}_{\text{tag}}^{(d)}$ 的分布式表示向量：

$$\mathrm{LE}(y^S) = [\mathbf{o}(\mathrm{IOB}(y^S)); \mathrm{emb}(\mathrm{SLOT}(y^S))] \tag{7–20}$$

其中记 $\mathrm{LE}(y^S) \in \mathbb{R}^K$，$K$ 标签分布式表征维度，并注意 $\mathrm{emb}(\mathrm{SLOT}(\text{“O”})) = \mathbf{0}$。

本文采用预训练的词向量或者语言模型参与标签分布式表征的提取，且为了计算方便我们在训练中固定相关预训练词向量或者语言模型的参数不变。语义槽标注样例相比于前两种先验知识应该是需要人工设计最少的，但是它对预训练词向量或者语言模型的质量有很大的依赖。

7.4.4 少样本支撑集与少样本学习

受上述语义槽标注样例的启发，带标注的数据也可以作为一种特殊的领域先验知识。在少样本学习 [138, 139] 中，每一个领域都只有非常少量的样本（支撑集），如图7–5所示。于是，我们借鉴少样本学习的概念，把少样本支撑集看作一种新的先验知识类型。

我们期望语义理解模型可以从大量的源领域中学习到测试句子和支撑集之间的匹配范式，从而将该模型应用到新的领域中。在应用的时候，目标领域仅提供支撑集即可让模型为目标领域进行语义理解预测。

我们记某个领域的支撑集为 $\mathcal{U} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{S(i)})\}_{i=1}^{|\mathcal{U}|}$，如果 N 个标签各自均在支撑集中有 k 个样例则被称为 N-way k-shot 设定 [139]。在目标领域上，仅有少量的标注样本（支撑集，*support set*）是提前知道的，没有其余的数据。在源领域上进行训练的时候，我们可以通过采样少量标注数据来模拟支撑集。

7.4.4.1 少样本支撑集的标签分布式表征计算

与上述第三种 “语义槽标注样例” 不同之处在于，我们将采用同样的编码器对测试句子和支撑集样本进行编码，并同时更新这个编码器。我们记句子编码器为 $\boldsymbol{E}$，对于输入词序列 $\boldsymbol{x}$，它会返回 $|\boldsymbol{x}|$ 长的词表示向量序列（即 $\boldsymbol{E}(\boldsymbol{x})$）。句子编码器 $\boldsymbol{E}$ 可以是前面用到的 BLSTM 句子编码器，也可以是其它的序列编码模型（比如 BERT 这类的预训练语言表示模型）。如果 $\boldsymbol{E}$ 为前文的 BLSTM 编码的话，则输入序列中第 i 个词的向量表示为 $\boldsymbol{E}(\boldsymbol{x})_i = \mathbf{h}_i$。

过去图片分类领域中有很多工作关注如何从支撑集 $\mathcal{U}$ 中获取标签分布式表征，比如匹配网络（matching network）[139]，原型网络（prototypical network）[145] 以及 TapNet [147]。我们以原型网络为例，定义语义槽标签 y^S 的标签分布式表征 $\mathbf{y}^S$ 是支撑集中相应标签对应的样本点词向量的中心点（即向量平均）：

$$\mathbf{y}^S = \mathrm{LE}(y^S) = \frac{1}{N_{y^S}} \sum_{j=1}^{|\mathcal{U}|} \sum_{t=1}^{|\boldsymbol{x}^{(j)}|} \mathbb{I}\{y_t^{S(j)} = y^S\} \boldsymbol{E}(\boldsymbol{x}^{(j)})_t \tag{7–21}$$

其中 $N_{y^S} = \sum_{j=1}^{|\mathcal{U}|} \sum_{t=1}^{|\boldsymbol{x}^{(j)}|} \mathbb{I}\{y_t^{S(j)} = y^S\}$ 是支撑集中标注为 y^S 的词个数。

7.4.4.2 基于向量投影距离的 “词-标签” 相似度

在少样本情形下，前文公式7–11中每个标签的输出得分计算不仅依赖输入句子，同时依赖于给定的支撑集。于是，我们记第 i 个词的输出层置信度向量 $\mathbf{z}_i$ 对应于当前领域任意一个语义槽标签 $y_k^S \in \mathcal{Y}_{\text{tag}}^{(d)}$ 的置信度为

$$[\mathbf{z}_i]_{\text{id}(y_k^S)} = f_{\text{sim}}(\boldsymbol{E}(\boldsymbol{x})_i, \mathbf{y}_k^S) \tag{7–22}$$

$\mathbf{y}_k^S$ 是语义槽标签 y_k^S 的分布式表征，$f_{\text{sim}}(\cdot,\cdot)$ 为相似度函数（比如前文公式7–11用到的向量点积运算）。于是，语义槽标签 y_k^S 的输出置信度被定义为了一个 “词-标签” 相似度，即带上下文信息的词向量 $\boldsymbol{E}(\boldsymbol{x})_i$ 和每个标签分布式表征 $\mathbf{y}_k^S$ 的相似度计算，以前他人的工作常用的有：向量点乘 [146, 147, 197]、余弦距离函数 [139]、欧拉距离 [145] 等。

如果我们延续前文对标签分布式表征先单位向量化再做向量点乘的思路，那么它在少样本支持集的应用本质上就是词表征 $\boldsymbol{E}(\boldsymbol{x})_i$ 在标签分布式表征 $\mathbf{y}_k^S$ 上的投影长度：

$$f_{\text{sim}}(\boldsymbol{E}(\boldsymbol{x})_i, \mathbf{y}_k^S) = \boldsymbol{E}(\boldsymbol{x})_i^\top \frac{\mathbf{y}_k^S}{\|\mathbf{y}_k^S\|} \tag{7–23}$$

其中 $k \in \{1, 2, \cdots, |\mathcal{V}_{\text{tag}}^{(d)}|\}$。不同于直接的向量点乘，该方法可以消除标签分布式表征 $\mathbf{y}_k^S$ 的范数大小的影响。比如一种极端情况下，$\mathbf{y}_k^S$ 的范数大到可以决定相似度的大小，这很可能会造成很多假阳性（false positive）错误。同样，为了进一步降低假阳性错误发生的可能，我们采用标签分布式表征 $\mathbf{y}_k^S$ 范数的一半作为阈值（偏置项）：

$$f_{\text{sim}}(\boldsymbol{E}(\boldsymbol{x})_i, \mathbf{y}_k^S) = \boldsymbol{E}(\boldsymbol{x})_i^\top \frac{\mathbf{y}_k^S}{\|\mathbf{y}_k^S\|} - \frac{1}{2}\|\mathbf{y}_k^S\| \tag{7–24}$$

即词向量 $\boldsymbol{E}(\boldsymbol{x})_i$ 在标签分布式表征 $\mathbf{y}_k^S$ 上的投影长度必须超过 $\mathbf{y}_k^S$ 的一半的范数长度才能得到正的相似度。偏置项一般表示的是标签的先验概率，这里 $-\frac{1}{2}\|\mathbf{y}_k^S\|$ 则表明标签分布式表征的长度越长其先验概率就越小。其次，我们猜想词向量的范数长度由每个向量的异构性决定的，词向量长度越大会驱使词向量某些维度的值相对于其它维度更大（即向量中不同维度值之间的方差变大），间接地词向量长度越大使得词的可区分性变大。

归一化线性模型的解释：我们对上述模型进行了一个简单的理论性的解释，即上述基于向量投影距离的相似度模型可以等价于为每个输出标签学习一个线性分类器。我们把上述公式重写为线性模型的表示：

$$f_{\text{sim}}(\boldsymbol{E}(\boldsymbol{x})_i, \mathbf{y}_k^S) = \boldsymbol{E}(\boldsymbol{x})_i^\top \mathbf{w}_k + b_k \tag{7–25}$$

其中 $\mathbf{w}_k = \frac{\mathbf{y}_k^S}{\|\mathbf{y}_k^S\|}$ 并且 $b_k = -\frac{1}{2}\|\mathbf{y}_k^S\|$。其权值向量被归一化为单位向量，来提升少样本模型的泛化能力，即 $\|\mathbf{w}_k\| = 1$。

7.5 实验与分析

本小节我们依次介绍实验数据与设置，以及基于标签分布式表征的语义理解领域自适应实验结果与分析。

7.5.1 实验数据与设置

7.5.1.1 数据集

本章在以下数据集进行了领域自适应的实验：

表 7–3 AICar 和 SNIPS 数据集的统计信息

Table 7–3 Dataset statistics of AICar and SNIPS.

数据集	领域	# 训练集	# 测试集	语义槽数量	原子概念数量	语义槽描述平均长度
AICar	Map	12000	20661	30	16	4.3
	Weather	10150	7236	36	20	3.4
	Flight	1500	4407	22	18	4.0
SNIPS	We	2000	100	10	-	1.6
	Mu	2000	100	10	-	1.1
	Pl	1942	100	6	-	1.5
	Bo	1956	100	8	-	2.4
	Se	1959	100	8	-	2.1
	Re	1973	100	15	-	1.6
	Cr	1954	100	3	-	1.8

- **AICar**：ATCar 数据集是从车载口语对话系统中收集的商用数据集，其包括数以千计的真实用户。该数据集共包含三个领域：地图导航（`MAP`），天气预报（`WEATHER`）和航班查询（`FLIGHT`）。这三个领域之间几乎没有重合的语义槽类别，但是不同语义槽之间却有部分联系，比如地图导航里有语义槽 `starting_point`，天气预报里有 `city_name` 以及航班查询领域有 `departure_city`。
- **SNIPS**：SNIPS [180] 是一个公开的任务型语义理解数据集，它是在语音助手应用中以众包的形式采集的自然语言数据。该数据集包含 7 个领域以及每个领域有大约 2000 句话的训练数据：天气领域（We），音乐领域（Mu），播放列表（Pl），书籍领域（Bo），搜索刷选事件（Se），餐馆（Re）以及搜索创意作品的领域（Cr）。

两个数据集的具体统计信息如表7–3所示，其中 SNIPS 是英文数据集，而 AICar 是中文数据集。为了避免中文分词错误的影响 [167]，我们直接在中文字级别上进行语义槽标记建模。

7.5.1.2 基于原子概念、语义槽描述和语义槽标注样例的实验设置

领域知识驱动的标签分布式表征：本章提出了三种领域先验知识以及相应的提出标签分布式表征的方法，分别是原子概念、语义槽描述和语义槽标注样例。表 7–3中也提供了关于原子概念数量和语义槽描述文本平均长度的统计。对于原子概念，我们请领域开发人员将 AICar 的语义槽细分为了更小的单元，比如语义

槽 deny-starting_point 被分为了 “deny”，“from_location” 以及 “poi_name”。对于语义槽文本描述，我们首先简单地使用了语义槽的名称（一般为名词短语），其次也在后续实验中对比了不同的文本描述对语义理解性能的影响。比如语义槽 deny-starting_point 可以被描述为 “不是这个起始地点”。而对于语义槽标注样例，我们采用了训练集可以访问的所有样本。

网络超参数与训练配置：我们对不同数据集采取了几乎完全相同的网络超参数与训练配置。我们采用了两层 BLSTM 作为句子编码器，LSTM 隐层大小为 200。除了预训练好的词向量部分，其他网络参数均以 $[-0.2, 0.2]$ 之间的均匀分布随机初始化。我们依然采用 Adam [160] 作为优化器，设置学习率为 0.001。在训练阶段，非循环层之间设置 0.5 的 Dropout 率。对所有 AICar 的 MAP 和 WEATHER 领域批处理的批次大小为 20，FLIGHT 的批处理的批次大小为 5，SNIPS 数据集的批处理批次大小为 10。梯度最大范数的修剪阈值为 5，同时我们采用 l_2 二范数正则化（系数为 1e-6）来避免过拟合。我们在训练集上训练 50 轮，并保存在校验集上性能表现（语义槽 F_1 得分 ①）最好的模型参数。最终我们报告该模型参数在测试集上的语义槽 F_1 得分。

网络输入层：我们采用预训练词向量对网络的输入层进行初始化。对于 AICar，我们在中文维基数据②上预训练了一个字级别的 LSTM 双向语言模型（词向量维度为 200）。对于 SNIPS，我们直接采用了英文 ELMo [196] 预训练双向语言模型③的词向量层（词向量维度为 1024）。

语义槽标注样例的编码器：前面的预训练词向量和双向语言模型还会应用到语义槽标注样例的编码上面，以获取相关标签分布式表征。为了减小训练开销，在本章中我们选择固定语义槽标注样例的编码器，而不进行更新。因此，在相关实验里语义理解的性能会严重依赖于预训练语言模型的质量。

显著性检验：我们采用了 McNemar’s test 方法来计算一个方法对比另一个方法的统计显著性（$p < 0.05$）。

领域自适应的数据设置：为了验证本章提出的方法在领域自适应情形下的有效性，我们对相关数据集的领域自适应问题进行了数据划分。在 AICar 中，我们循环地选择一个领域作为目标领域，而剩下两个作为源领域。另外，为了模拟目标领域数据的稀疏性问题，我们随机选取目标领域原始训练数据中 {1%，2.5%，5%，10%，20%，40%，60%，80%} 的样本作为实际训练数据。每次数据选择都重复做十次，并最终平均所有实验的评测结果。然后，我们对选择的训练数据按 4:1 的比

① 语义槽 F_1 得分计算脚本为https://www.clips.uantwerpen.be/conll2000/chunking/output.html

② https://dumps.wikimedia.org/zhwiki/latest

③ https://github.com/allenai/allennlp/blob/master/tutorials/how_to/elmo.md

例随机划分训练集和校验集。对于 SNIPS 的七个领域，我们也循环选择其中一个领域作为目标领域，而其余的作为源领域。

7.5.1.3 少样本学习的实验设置

少样本学习的数据设置：为了方便和已有方法的比较，我们直接采用了 Hou 等 [146] 发布的 SNIPS 和 NER 数据集的小样本数据划分 ①。该数据划分把每个领域的数据组织为 *episode* 形式 [139]，即每个 episode 都包括一个支撑集（1-shot 或者 5-shot）和一批用于训练或者评测的已标注样本。SNIPS 的数据集前面已经介绍过了。而 NER 则是由四个不同领域的命名实体识别数据集组合而成，分别是：CoNLL-2003（新闻领域，News）[198]，GUM（维基百科领域，Wiki）[199]，WNUT-2017（社交媒体领域，Social） [200] 以及 OntoNotes（混合领域，Mixed）[201]。更多的数据划分细节请参照 Hou 等 [146] 的工作。

对于 SNIPS 和 NER，我们参照 Hou 等 [146] 的做法，选择其中一个领域作为评测的目标领域，另外一个作为校验集而其余的领域作为训练使用的源领域。评测的时候我们对目标领域的每个 *episode* 单独测试并计算相应测试集上的语义槽 F_1 得分，最终我们将所有 *episode* 上的 F_1 得分平均。对于每个实验我们都使用十个不同的随机种子 ②运行了十次，并将十次评测结果进行了平均。

网络超参数与训练配置：在基于少样本学习的所有实验中，我们采用了预训练的英文 `BERT-base-uncased` [102] 作为上下文词向量提取器 $\boldsymbol{E}$。相关模型同样采用 Adam [160] 优化器训练，并使用了 1e-5 的学习率。批处理训练是一个批次为一个 episode 的数据。为了对 BERT 模型底层参数的更新小一些，我们采用了随层递减的学习率（衰减率为 0.9），即 BERT 第 l 层参数的学习率为 $1e\text{-}5 * 0.9^{(L-l)}$，其中 L 表示 BERT 模型的最大层数。而对于 CRF 模型中的标签转移参数，我们把它们初始化为零，并设置这部分参数的学习率为 1e-3。

7.5.2 原子概念、语义槽描述以及语义槽标注样例

7.5.2.1 基线系统

本章提出的方法将与不使用标签分布式表征的传统 BLSTM 模型、以及两种零样本学习的方法进行对比。

- **Original BLSTM**：我们采用两层的 BLSTM 以及一个线性输出层（CRF 层是可选的）构成的语义槽标记模型。这种情况下没有领域知识的加入，标

① https://atmahou.github.io/attachments/ACL2020data.zip

② 随机种子生产网站https://www.random.org

签分布式表征相当于是一个独热向量（one-hot vector）。

- **Concept Tagger (CT)** [131]：该方法采用了语义槽的描述文本，并将该描述文本和输入句子一起编码到了 BLSTM 中（同样是两层 BLSTM）。但是，该方法在训练和预测时需要对每一个语义槽分别单独运行一次模型，计算代价很高；此外，这种方案也容易造成语义槽值区域重叠的情况。
- **Zero-shot Adaptive Transfer (ZAT)** [132]：该方法改进了上述 CT 对描述文本和输入句子的编码结构，采用了注意力机制，而 CT 则是直接对描述文本的词向量进行相加与平均的操作。

7.5.2.2 主要结果

表7–4显示了在 AICar 数据集（三个领域）上的领域迁移实验，其中每个领域均被作为目标领域，而剩余两个领域作为源领域。我们在目标领域的不同训练数据量（从 1% 到 100% 的比例）上对所有基线系统和本章提出的基于原子概念（atomic concept）、语义槽描述（slot description）、语义槽标注样例（slot exemplar）的标签分布式表征方法进行了比较。从表中的结果我们可以发现：

- 两个零样本学习的基线系统（CT 和 ZAT）都可以比原始的 BLSTM 模型（Original BLSTM）表现的更好，因为它们也利用了语义槽描述文本。然而，这两种方法是语义槽独立建模的，存在计算时间长的缺点；以及会引起语义槽值分段的重叠问题。
- 随着目标领域的训练数据量的增加，所有方法的语义槽 F_1 得分都随之升高。这说明目标领域数据越多，领域迁移的效果自然就越好。
- 在所有不同的目标领域训练数据量下，我们提出的基于标签分布式表征的语义理解领域自适应方法都可以取得最好的结果，并且在绝大多数情况下都显著性地优于最好的基线系统。
- 最好的方法经常是在原子概念和语义槽描述两者之间。相比语义槽标注样例，前两者引入了更多的领域知识，因为语义槽标注样例已经存在于训练数据之中。但即便如此，在绝大多数情况下，基于语义槽标注样例的标签分布式表征还是可以击败基线方法。

另外，我们也在 SNIPS 数据集上进行了领域迁移的实验。我们同样将 SNIPS 中 7 个领域一一作为目标领域，而其他领域作为源领域。在目标领域中我们仅随机选取 50 个训练样本用于语义理解模型的微调更新。循环如此，最终我们将 7 个实验在目标领域上的语义槽 F_1 得分平均，结果如表7–5所示。从结果来看，我们的方法全部优于基线系统，并且基于语义槽标注样例的方法取得了最好的性能。

表 7–4 在 AICar 数据集上的领域迁移实验。我们衡量了目标领域在不同训练数据量上的领域迁移效果。其中粗体的数字表示最好的语义槽 F_1 得分

Table 7–4 Slot F_1 scores on AICar dataset with three domains. The results in bold black are the best F_1 scores.

目标领域	方法	2.5%	5%	10%	20%	40%	60%	80%	100%
Map	Original BLSTM	60.24	72.67	82.63	87.94	90.99	92.92	93.61	94.79
	CT	62.23	75.20	82.79	87.58	91.28	92.88	94.02	94.80
	ZAT	63.74	73.79	82.10	87.39	91.33	92.40	93.65	94.62
	atomic concept	66.57	77.20	84.74	89.27	92.04	93.76	**94.81**	**95.78**
	slot desc. (BLSTM)	67.26	77.11	84.83	88.68	92.02	93.81	94.56	95.65
	slot desc. (CNN)	67.41	77.88	84.85	88.79	91.86	93.77	94.41	95.31
	slot desc. (mean)	**70.12**	**79.95**	**85.92**	**89.37**	**92.48**	**93.90**	94.67	95.47
	slot exemplar (BiLM)	66.17	76.74	84.45	88.80	91.91	93.61	94.42	95.72
Weather	Original BLSTM	39.35	49.23	62.40	72.96	80.57	84.13	86.50	88.60
	CT	45.46	52.26	62.67	73.39	81.42	84.85	86.96	87.85
	ZAT	47.03	54.10	65.28	73.60	80.60	84.19	86.75	88.61
	atomic concept	**48.38**	56.51	67.39	75.67	82.48	85.08	87.35	88.79
	slot desc. (BLSTM)	47.41	**56.67**	66.84	75.72	82.70	**85.89**	**88.23**	89.43
	slot desc. (CNN)	46.47	54.73	**67.41**	75.85	**82.74**	85.37	87.86	**89.44**
	slot desc. (mean)	48.32	55.71	66.54	**76.06**	82.47	85.56	87.90	89.42
	slot exemplar (BiLM)	44.28	52.41	64.50	74.61	81.62	84.75	87.56	88.00
Flight	Original BLSTM	65.84	73.22	80.68	87.58	92.26	94.14	94.92	95.26
	CT	71.01	75.36	81.40	87.74	92.32	93.84	94.97	95.74
	ZAT	70.44	74.80	82.03	87.81	92.37	93.93	94.77	95.34
	atomic concept	72.74	78.58	83.42	88.76	92.79	93.99	95.02	95.27
	slot desc. (BLSTM)	75.93	79.93	85.49	89.96	92.94	**94.99**	95.33	95.47
	slot desc. (CNN)	73.15	78.45	84.92	89.58	93.17	94.69	**95.41**	95.83
	slot desc. (mean)	**76.02**	**80.77**	**85.79**	**90.33**	**93.33**	94.60	95.29	95.76
	slot exemplar (BiLM)	72.96	77.85	82.93	88.66	92.92	94.39	95.17	**95.98**

7.5.2.3 实验分析

零样本学习分析：由于领域知识驱动的标签分布式表征包含了不同语义槽之间的关系，从而在不需要目标领域训练数据的情况下，基于标签分布式表征的语义理解模型也有可能自动泛化到目标领域上。这种目标领域不提供训练数据的情况也被称为零样本学习。我们在 AICar 上开展了相关实验，将三个领域一一作为目

表 7–5 SNIPS 数据集上的领域迁移实验。其中目标领域仅使用 50 个训练样本用于模型微调

Table 7–5 Slot F_1 scores on SNIPS test set in the setting of domain adaptation with only 50 instances of each target intent for fine-tuning.

方法	领域平均的 $\mathbf{F}_1$ 得分
Original BLSTM	85.63
CT	83.79
ZAT	86.10
slot description (BLSTM)	86.94
slot description (CNN)	86.99
slot description (mean)	86.57
slot exemplar (BiLMs)	**87.33**

表 7–6 在 AICar 数据集上对零样本学习（即目标领域不提供训练数据）的分析

Table 7–6 Zero-shot learning on the AICar dataset.

方法	平均批次时间	领域平均的 $\mathbf{F}_1$ 得分
Original BLSTM	7.6ms	0
CT	101.2ms	2.42
ZAT	134.7ms	5.89
atomic concept	8.0ms	26.20
slot description (BLSTM)	10.6ms	3.55
slot description (CNN)	9.3ms	6.64
slot description (mean)	8.4ms	8.17
slot exemplar (BiLMs)	7.7ms	0

标领域，并最后将三个目标领域上的语义槽 F_1 得分平均，如表7–6所示。从结果中我们可以发现，原子概念可以取得最好的零样本学习情形下的性能水平。因为原子概念是人为显式定义的领域知识，相关原子可以组合成一个新的语义槽。但是基于语义槽文本描述的方法依赖于标签分布式表征编码器的参数学习，相关泛化能力被弱化。而基于语义槽标注样例的方法在完全不提供目标领域训练数据（即标注样例也无法获取）的情形下无法开展。

不同方法的计算效率：我们在 AICar 数据集上也对比了不同方法的计算效率，即单个批次数据在型号为 “GeForce GTX 1080 Ti” 的显卡上的平均训练时间。相关结果如表7–6所示，我们可以发现对语义槽独立建模的 CT 和 ZAT 基线系统相比其他方法非常耗时（其运行时间在理论上与语义槽数量成正相关）。

学习曲线：图7–6展示了不同方法在目标领域为 `FLIGHT`（AICar 数据集）且保

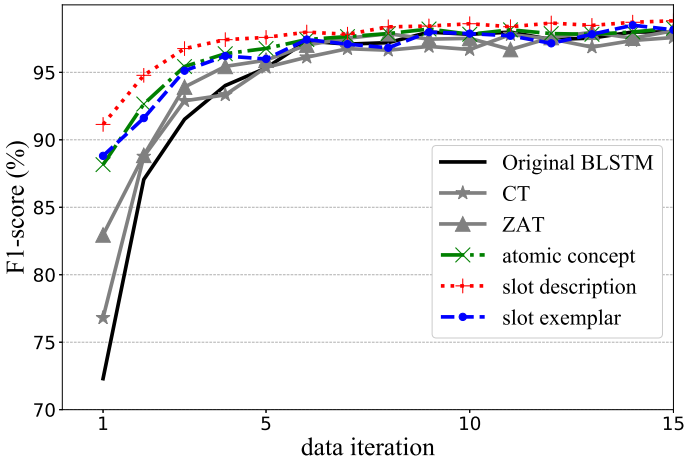


图 7–6 不同领域自适应方法在目标领域校验集上的的学习曲线。该实验的目标领域是 FLIGHT 且 100% 的训练数据被用于模型微调

Figure 7–6 Learning curves of the baselines and our methods when FLIGHT is the target domain and 100% training data is used for fine-tuning. It shows performances on the validation set at different data iterations.

留 100% 训练数据时的学习曲线。该图显示了语义槽 F_1 得分在目标领域校验集上随着训练轮数的变化。可以看到我们的方法比基线系统具有更快的收敛速度。尤其在第一轮训练结束后，我们的方法的性能会大大超过所有的基线系统。

不同语义槽描述文本的影响：语义槽描述文本是基于自然语言的非结构化文本，具有较高的自由度。尤其是不同的领域专家或者开发人员定义的描述文本肯定会有多多少少的差别。为了探究不同语义槽描述文本的影响，我们邀请了五位志愿者参与扮演 AICar 的领域专家，并被要求按照自己的理解对 FLIGHT 领域的每个语义槽给出一段相匹配的文本描述。相关实验结果如表7–7所示。最终，所有人给出的语义槽描述的平均长度在 6.2 到 7.7 个字之间。如果以 BLSTM 为语义槽描述编码器，并且以 FLIGHT 为目标领域以及保留 10% 的训练数据用于模型微调，目标领域的语义槽 F_1 得分在 84.66% 到 86.43% 之间。这说明不同语义槽描述文本对我们的方法影响并不太大。

语义槽标注样例的数量的影响：尽管我们可以使用目标领域所有的训练数据作为语义槽标注样例，但为了减低计算代价，我们有必要去探索语义槽标注样例的数量对最终领域自适应性能的影响。图7–7展示了语义槽样例数量和目标领域（FLIGHT）性能的相关结果。如果我们不使用目标领域的训练数据进行模型微调，那么越多的语义槽标注样例会带来更好的效果。但是，如果目标领域有很多训练

表 7–7 不同语义槽描述文本对领域自适应的性能影响。该实验是“slot description (BLSTM)”方法在 FLIGHT 目标领域上采用 10% 训练数据进行模型微调的结果。

Table 7–7 Performances of the “slot description (BLSTM)” method with varied slot descriptions on FLIGHT. FLIGHT is the target domain, and its 10% training data is used for fine-tuning.

语义槽描述平均长度	4.0 (ori.)	7.7	7.4	7.0	6.4	6.2
Slot F_1	85.49	85.84	84.66	85.80	86.43	85.08

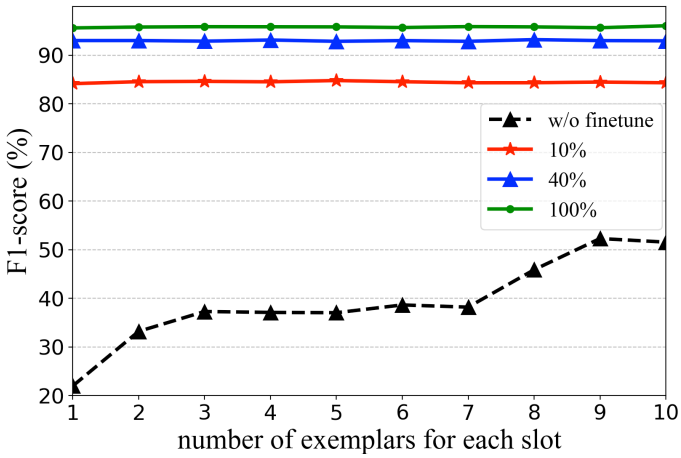


图 7–7 语义槽标注样例的数量对领域自适应的性能影响。该实验是“slot exemplar (BiLMs)”方法在 FLIGHT 目标领域上采用 {none, 10%, 40%, 100%} 训练数据进行模型微调的结果

Figure 7–7 Performances of the “slot exemplar (BiLMs)” method when FLIGHT is the target domain and {none, 10%, 40%, 100%} training data is used for fine-tuning. The number of exemplars for each slot ranges from 1 to 10.

数据可以进行模型微调的话，语义槽标注样例的数量显得就不重要了。

7.5.3 基于少样本学习的跨领域语义理解

7.5.3.1 基线系统

本章提出的方法将与如下基线系统进行对比：

- **SimBERT**：对于输入句子中的每个词 x_i，SimBERT 从支撑集中找到与之最像的词 x'_k，并把 x'_k 的输出标签分配给 x_i。词相似度计算是基于 BERT 词向量的余弦距离，且该 BERT 模型是固定不更新的。
- **TransferBERT**：对于每个领域，我们都在共享的 BERT 模型上加上一个可训练的线性分类器用作语义槽标签预测。在目标领域，该分类器是在支撑集上进行训练优化的。

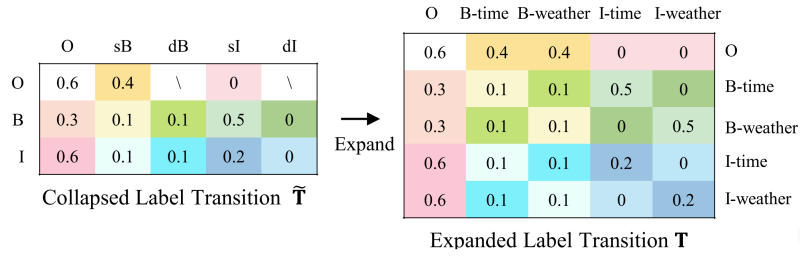


图 7–8 折叠依赖迁移机制的一个示例（该图引自 Hou 等的工作 [146]）

Figure 7–8 An example of collapsed label dependency transfer（this figure is from Hou *et al.* [146]）

.

- **L-WPZ+CDT+PWE：** WPZ 是一种少样本序列标注模型 [144]，它将序列标注看成是每个词的分类任务。该方法进一步与 BERT、基于折叠依赖迁移机制（Collapsed Dependency Transfer，CDT) 的 CRF 模型以及成对词嵌入机制（Pair-Wise Embedding，PWE） [146] 相结合。
- **L-TapNet+CDT+PWE：** 之前最高性能水平的少样本语义槽标记方法 [146]，该方法基于 TapNet [147] 并同样结合 BERT、CDT 以及 PWE。

CRF 模型中的转移得分体现了相邻时刻输出标签之间的时序依赖关系，具体表现为每个可能的 “标签对” 都有一个可以学习的标量参数。由于不同领域之间的语义槽标签集是不相同的，所以为了共享不同领域中潜在的转移因子，我们采用了折叠依赖迁移机制（Collapsed Dependency Transfer，CDT）[146]。如图7–8所示，CDT 机制仅对相邻标签之间通用、抽象的关系进行建模，即图中左边的矩阵 $\widetilde{\mathbf{T}}$。纵轴表示上一时刻标签是以 O 还是 B 或者 I 开头的，横轴这表示当前时刻的标签是一个 O、还是相同的 B（sB）、不同的 B（dB）、相同的 I（sI）、不同的 I（dI）。

为了保证实验的可比性，后续直接引用了 Hou 等 [146] 在相同数据划分上公布的上述方法的结果。

7.5.3.2 主要结果

表7–8和表7–9展示了 SNIPS 和 NER 数据集上的少样本（1-shot 和 5-shot）语义槽标记性能。我们的方法可以显著超越所有的基线系统，包括之前最高性能水平的方法。此外，之前最好的方法 “L-TapNet+CDT+PWE” 非常依赖成对的词嵌入机制，即将输入句子和每一个支撑句子依次拼接后送入后续句子编码器（实验中用的是 BERT）中。通过对比 “L-TapNet+CDT+PWE” 和 “L-TapNet+CDT+VP”，我们可以发现本章提出的向量投影（VP) 距离即可以获得更好的性能也相比 PWE 有更高的计算效率。此外，实验结果还表明原型网络（ProtoNet）相比 TapNet 会更

表 7–8 不同少样本学习方法在 SNIPS 数据上的语义槽 F_1 指标（* 表示我们的方法）

Table 7–8 F_1 scores on few-shot slot tagging of SNIPS.

	Model	We	Mu	Pl	Bo	Se	Re	Cr	Avg.
1-shot	SimBERT	36.10	37.08	35.11	68.09	41.61	42.82	23.91	40.67
	TransferBERT	55.82	38.01	45.65	31.63	21.96	41.79	38.53	39.06
	L-WPZ+CDT+PWE	71.23	47.38	59.57	81.98	69.83	66.52	62.84	65.62
	L-TapNet+CDT+PWE	71.53	60.56	66.27	**84.54**	76.27	70.79	62.89	70.41
	L-TapNet+CDT+VP *	71.65	**61.73**	63.97	83.34	74.00	71.91	**71.02**	71.09
	ProtoNet+CDT+VP *	**73.56**	58.40	68.93	82.32	**79.69**	**73.40**	70.25	**72.37**
	ProtoNet+CDT+VPB *	72.65	57.35	68.72	81.92	74.68	72.48	70.04	71.12
	L-ProtoNet+CDT+VPB *	73.12	57.86	**69.01**	82.49	75.11	73.34	70.46	71.63
5-shot	SimBERT	53.46	54.13	42.81	75.54	57.10	55.30	32.38	52.96
	TransferBERT	59.41	42.00	46.07	20.74	28.20	67.75	58.61	46.11
	L-WPZ+CDT+PWE	74.68	56.73	52.20	78.79	80.61	69.59	67.46	68.58
	L-TapNet+CDT+PWE	71.64	67.16	75.88	84.38	82.58	70.05	73.41	75.01
	L-TapNet+CDT+VP *	78.25	67.79	70.66	86.17	75.80	78.51	75.93	76.16
	ProtoNet+CDT+VP *	79.88	67.77	78.08	87.68	**86.59**	79.95	75.61	79.37
	ProtoNet+CDT+VPB *	82.91	69.23	80.85	90.69	86.38	81.20	**76.75**	81.14
	L-ProtoNet+CDT+VPB *	**82.93**	**69.62**	**80.86**	**91.19**	86.58	**81.97**	76.02	**81.31**

好一些。如果我们继续采用标签分布式表征范数的一半作为偏置项阈值（VPB），在 5-shot（即每个语义槽标签在支撑集中至少出现 5 次）情形下可以明显提升目标领域的语义槽 F_1 得分。我们猜测是因为 5-shot 情形下每个标签都有多个支撑样本点，相比 1-shot 情形更容易引发假阳性错误。最后如果结合基于语义槽文本描述的标签分布式表征（即将两种标签分布式表征插值加和，ProtoNet 标签分布式表征权值为 0.9），整体性能还可以有细微的提升（即 L-ProtoNet+CDT+VPB）。

7.5.3.3 实验分析

不同 “词-标签” 相似度函数的对比：关于 “词-标签” 相似度函数 $f_{\text{sim}}(\mathbf{x}, \mathbf{c})$，我们也做了一些对比实验；并将我们提出的基于向量投影距离的方法和其它可能的变种形式进行了对比，包括：向量点积（$\mathbf{x}^\top\mathbf{c}$），标签分布式表征在词向量上的投影（$\frac{\mathbf{x}^\top}{\|\mathbf{x}\|}\mathbf{c}$），余弦距离（$\frac{\mathbf{x}^\top}{\|\mathbf{x}\|}\frac{\mathbf{c}}{\|\mathbf{c}\|}$），欧拉平方距离（$-\frac{1}{2}\|\mathbf{x}-\mathbf{c}\|^2$）以及基于可训练的缩放因子（$\lambda\mathbf{x}^\top\mathbf{c}$）[197]。在表7–10上的结果表明我们的方法可以显著性地超越其它的方法。同时，我们也发现欧拉平方距离可以在 5-shot 情形下取得还不错的效

表 7–9 不同少样本学习方法在 NER 数据上的语义槽 F_1 指标（* 表示我们的方法）

Table 7–9 F_1 scores on few-shot slot tagging of NER.

	Model	News	Wiki	Social	Mixed	Avg.
1-shot	SimBERT	19.22	6.91	5.18	13.99	11.32
	TransferBERT	4.75	0.57	2.71	3.46	2.87
	L-TapNet+CDT+PWE	44.30	**12.04**	20.80	15.17	23.08
	L-TapNet+CDT+VP *	44.73	8.91	**30.61**	29.39	28.41
	ProtoNet+CDT+VP *	**44.82**	11.32	26.96	29.91	28.25
	ProtoNet+CDT+VPB *	42.50	10.78	27.17	32.06	28.13
	L-ProtoNet+CDT+VPB *	43.47	10.95	28.43	**33.14**	**29.00**
5-shot	SimBERT	32.01	10.63	8.20	21.14	18.00
	TransferBERT	15.36	3.62	11.08	35.49	16.39
	L-TapNet+CDT+PWE	45.35	11.65	23.30	20.95	25.31
	L-TapNet+CDT+VP *	50.43	8.41	29.93	37.59	31.59
	ProtoNet+CDT+VP *	54.82	16.30	27.43	33.38	32.98
	ProtoNet+CDT+VPB *	**57.42**	**19.48**	35.06	44.45	**39.10**
	L-ProtoNet+CDT+VPB *	56.30	18.57	**35.42**	**44.71**	38.75

果。我们对此进行分析，并将欧拉平方距离公式按如下展开：

$$-\frac{1}{2}\|\mathbf{x}-\mathbf{c}\|^2 = -\frac{1}{2}\mathbf{x}^\top\mathbf{x} + \mathbf{x}^\top\mathbf{c} - \frac{1}{2}\mathbf{c}^\top\mathbf{c} \cong \mathbf{x}^\top\mathbf{c} - \frac{1}{2}\mathbf{c}^\top\mathbf{c} \tag{7–26}$$

其中 $-\frac{1}{2}\mathbf{x}^\top\mathbf{x}$ 对于每个标签都是一样的，可以被忽略。最后的公式分解形式中 $\frac{1}{2}\mathbf{c}^\top\mathbf{c}$ 作为一个阈值也可以缓解假阳性错误。

模型消融实验：为了分析模型不同组成部分的有效性，我们进行了相关消融实验，结果如表7–11所示。首先，我们可以发现，基于语义槽文本描述的标签分布式表征对最终性能的贡献非常微小，并不显著。其次，基于 CDT 的 CRF 输出层对模型最终的性能影响非常大，这说明了输出标签的时序依赖关系对于少样本学习是很重要的。最后，如果我们不使用支撑集（即仅利用语义槽文本描述）的话，最终性能会非常差，因为支撑集提供的领域知识（即少量的标注数据）是明显更丰富的。

在支撑集继续微调的效果：在上述我们提供了支撑集的情况下，其实语义理解模型是允许在该支撑集上进行微调优化的。只是绝大多数的少样本学习在于维护一个跨领域的单个通用模型，因为进行微调的话不同目标领域的模型参数就会不一样。那么如果考虑微调的话，我们的方法又会如何变化呢？表7–12展示了我

表 7–10 不同相似度函数的对比以及相应的领域平均的 F_1 得分

Table 7–10 Comparison among different similarity functions. Results are average F1-scores of all domains.

$f_{\text{sim}}(\mathbf{x}, \mathbf{c})$	SNIPS		NER	
	1-shot	**5-shot**	**1-shot**	**5-shot**
$\mathbf{x}^\top \frac{\mathbf{c}}{\|\mathbf{c}\|}$	**72.37**	79.37	**28.25**	32.98
$\mathbf{x}^\top \frac{\mathbf{c}}{\|\mathbf{c}\|} - \frac{1}{2}\|\mathbf{c}\|$	71.12	**81.14**	28.13	**39.10**
$\mathbf{x}^\top \mathbf{c}$	57.92	65.03	17.10	19.91
$\frac{\mathbf{x}^\top}{\|\mathbf{x}\|}\mathbf{c}$	63.87	71.16	16.72	23.65
$\frac{\mathbf{x}^\top}{\|\mathbf{x}\|}\frac{\mathbf{c}}{\|\mathbf{c}\|}$	34.02	39.21	10.40	12.26
$\lambda \mathbf{x}^\top \mathbf{c}$	48.91	68.11	5.99	21.05
$-\frac{1}{2}\|\mathbf{x} - \mathbf{c}\|^2$	66.91	79.72	20.04	34.04

表 7–11 模型消融实验。相应结果是领域平均的 F_1 得分

Table 7–11 Ablation study. Results are average F1-scores of all domains.

Method	SNIPS		NER	
	1-shot	**5-shot**	**1-shot**	**5-shot**
L-ProtoNet+CDT+VPB	71.63	81.31	29.00	38.75
(-) w/o slot description	71.12	81.14	28.13	39.10
(-) w/o CDT based CRF	64.30	75.51	24.29	33.59
(-) w/ only slot description	38.32	37.86	17.96	14.23

们上述 “ProtoNet+CDT+VP” 方法在目标领域的支撑集上继续微调 ①的效果，其中微调训练配置和在源领域上的预训练是一致。我们记录了对应不同微调训练轮数的性能变化，并发现目标领域的微调可以带来非常明显的性能提升，并随着微调轮次的增大性能也会逐步增加。因为在目标领域支撑集上的微调训练有助于鉴别式地区分不同语义标签的分布式表征向量，进一步提升性能。

7.6 本章小结

针对大规模任务构建中的领域迁移问题，本章研究如何利用已有领域（源领域）的数据构建新领域（目标领域）的语义理解模型。在领域自适应方面，我们提出了一种基于标签分布式表征的语义理解模型，解决了因为不同领域之间标签不一致导致输出层参数无法完全共享的问题。我们在标签分布式表征中引入了多

① 需要注意对于不同的 episode 具有不同的支撑集，每个 episode 在微调前需要加载同样的预训练模型。

表 7–12 少样本语义理解模型（“ProtoNet+CDT+VP”）在支撑集上微调的性能结果（领域平均的 F_1 得分）

Table 7–12 Fine-tuning few-shot based slot tagger on support sets of target domains. Results are averaged F_1-scores of all domains.

Fine-tune step	SNIPS		NER	
	1-shot	5-shot	1-shot	5-shot
0	72.37	79.37	28.25	32.98
1	73.47	80.91	29.16	34.77
3	74.92	82.98	30.76	37.49
5	75.48	83.97	31.93	39.29
10	75.72	84.87	33.41	42.03

种先验知识 (原子概念、语义槽描述、语义槽标注样例、少样本支撑集)，并提出相应的标签分布式表征编码网络。在中英文数据集（AICar 和 SNIPS）上的领域自适应实验表明，该方法在目标领域仅提供少量数据的情形下相比以往的领域自适应方法有明显提升。另外，我们以少样本学习为切入点，为匹配式网络提出了一种基于向量投影的 “词-标签” 相似度计算方法。在语义槽填充 SNIPS 数据集和命名实体识别 NER 数据集上，我们的方法相比以往的少样本学习方法均取得了非常明显的性能提升。

第八章　总结与展望

作为任务型口语对话系统中的一个重要模块，口语理解旨在从语音识别的结果中提取结构化的语义信息并提供给后续的对话管理与决策模块。作为口语对话系统中联通感知智能（语音识别）与认知智能（对话决策）的桥梁，口语理解最近在研究界和工业界受到了越来越多的关注。在本文中，我们针对真实应用中的任务型口语理解探索了一系列实际问题：1）面对口语交互模式下的输入不确定性，如何构建且精准的口语理解系统？2）如何解决实际情况下单领域的数据稀缺问题？3）在大规模任务构建中，如何利用其它领域的数据帮助目标领域的口语理解系统搭建？并针对这三个实际问题开展了一系列研究与探索。

8.1　工作总结

本论文围绕任务型口语理解中的三个核心挑战：口语模式中的输入不确定性、单领域的数据稀疏以及跨领域的迁移，进行了以下研究工作。

在第三章中，针对语音识别结果的不确定性编码问题，本文将任务型语义理解等价变换为非词对齐的分类或者生成式问题，提出了一种高效的基于 Transformer 的不确定性编码方法。该方法将词混淆网络的图结构扁平化，利用基于时刻的位置编码和融合语音识别后验概率的自注意力机制把不确定信息注入到网络中。在口语理解标准数据集 DSTC-2 上的实验表明，我们的方法比以往基于循环神经网络的方法有显著的性能提升，并取得了当前最高的性能水平。

在第四章中，针对在语音识别结果上词对齐式语义标注难的问题，本文提出了一种基于强化学习的数据与规则双驱动的精准口语理解框架。我们采用了一种基于规则的槽值纠错方法，根据领域本体提供的语义候选项，使用词序列以及发音序列的相似度对 “槽值对” 所包含的语音识别错误进行纠正。后续，我们利用 “槽值对” 级别（非词对齐）的语义监督信号并结合策略梯度的强化学习算法来指导优化面向语音识别结果的序列标注模型。在目前最大的中文口语理解公开数据集 CATSLU 上的实验表明，该方法在语音识别结果上的语义理解精度相比以往方法获得了显著的性能提升。

在第五章中，为了解决单领域的数据稀缺问题，本文提出了一种基于对偶学习数据扩充的任务型语义理解半监督学习框架。我们利用语义理解的对偶（反向）模型为无文本对应的语义形式生成流畅的自然语言句子，并结合对偶学习技术为 “主模型” 和 “对偶模型” 提供合作式的闭环学习，同步优化两个模型。对偶模型在

应用中的一个好处是，无文本对应的语义形式可以通过少量的领域本体知识自动获取。该方法的有效性在多个任务型语义理解的标准数据集上得到了验证。

在第六章中，我们进一步考虑数据扩充模型的领域自适应学习，在数据层面研究跨领域的迁移问题。本文为上述对偶模型提出了一种新型的基于原子模板的“语义到文本”生成架构。我们针对细粒度的语义单元（语义槽）设计短语模板（原子模板），将领域相关的知识包装其中。然后将语义形式（即语义槽值对的集合）映射为一个短语集合，并最终构建领域无关的“短语集合到句子”文本重写模型。在语义理解标准数据集 DSTC-2&3 上的实验表明，该方法可以实现语义理解数据扩充模型的领域迁移，并且使用扩充数据帮助目标领域构建语义理解模型，最终在目标领域仅有极少量种子数据的情形下取得较高的语义理解性能。

在第七章中，针对模型层面的跨领域迁移问题，本文提出了一种基于标签分布式表征的语义理解自适应方法，深度融合不同领域（即不同标签集）的数据。我们在标签嵌入中引入了多种先验知识（原子概念、语义槽描述、语义槽标注样例、少样本支撑集），并提出相应的标签编码网络。我们在多个中英文数据集上设计了领域迁移的实验，相关结果表明该方法在目标领域仅提供少量数据的情形下相比以往的领域自适应方法有明显提升。此外，在任务型语义理解的少样本学习方面，我们为匹配式网络提出了一种基于向量投影的“词-标签”相似度计算方法。在语义槽填充和命名实体识别的相关数据集上，我们的方法相比以往的少样本学习方法均取得了非常显著的性能提升。

8.2 研究展望

首先，尽管本文在任务型口语理解的上述三个核心挑战上取得了一定的进展，但是在一些层面上还有继续进步的空间。本文进一步的研究方向包括：

1）本文在语音识别结果的不确定编码方面针对词混淆网络（WCN）提出了一种基于 Transformer 的编码结构，可以有效编码 WCN 中的图结构和语音识别后验概率信息。但在实际应用中，WCN 数据的收集成本较高，而自然语言文本数据更易获得。如何为自然语言文本模拟生成高质量的 WCN 数据将会是一个非常有趣的研究点。

2）在本论文中我们提出了基于槽值纠错和强化学习的口语理解方法来解决语音识别结果上词对齐标注难的问题，为基于自然语言的序列标注模型在语音识别结果上的应用提供了一个解决方案。如何进一步将该方法应用到复杂度更高的语音识别结果（如 WCN）上并摆脱对槽值纠错模块的依赖将是未来的一个非常有意义的研究方向。

3）本论文针对大规模任务构建提出了一种基于标签分布式表征和少样本学习的语义理解模型，让通用语义理解模型成为可能。如何利用自然语言理解领域的大型语义数据库（比如 FrameNet ①）对该通用语义理解模型进行预训练将会是一个很有吸引力的研究点。

4）本论文在上述三个核心挑战上分别进行了研究，如何将上述三类技术进行系统性融合将会是一个非常有实际意义的研究课题。

其次，在过去的几十年里，虽然任务型口语理解取得了很多的进展，但还有一些开放性问题没有被回答。一些未来的研究展望包括：

1）目前有少量的工作已经开始探索端到端的口语理解，即直接从语音信号转换得到语义形式，以提升口语理解对语音信号噪音的稳健性。但目前这类工作的性能大多还处于较低的水平。研究如何对端到端口语理解进行建模并设计相关学习算法以取得稳健精准的语义理解性能，将会是未来备受关注的研究方向。

2）目前，任务型口语理解的语义表示形式还是比较简单的结构（比如集合形式、两层的层级结构等），缺乏对复杂语义的表达能力。研究适用于任务型对话系统且表达能力更强的语义表示形式将会是一个很有意义的研究方向。

3）现在几乎所有的任务型口语理解都需要提供领域本体（至少包括领域的意图、语义槽等设计），需要领域专家进行细致的领域划分和定义。那么，如何从大量的网页、文档等无结构或者半结构的数据中自动归纳出领域本体将是未来一个非常重要的课题。

① https://framenet.icsi.berkeley.edu/fndrupal

附录 A　一些神经网络基本模型的定义

A.1　BLSTM

基于长短期记忆记忆单元（Long Short-Term Memory，LSTM）的双向循环神经网络（Recurrent neural network，RNN），简称 BLSTM，经常被用来对序列进行建模。给定一个序列的特征向量（$\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_L$），BLSTM 在第 t（$i \in \{1, 2, \cdots, L\}$）时刻的隐层向量可以通过如下方式递归计算得到：

$$\overrightarrow{\mathbf{h}}_i = \mathrm{f}_{\mathrm{LSTM}}(\mathbf{e}_i, \overrightarrow{\mathbf{h}}_{i-1}), i = 1, 2, \cdots, L \tag{A–1}$$

$$\overleftarrow{\mathbf{h}}_i = \mathrm{b}_{\mathrm{LSTM}}(\mathbf{e}_i, \overleftarrow{\mathbf{h}}_{i+1}), i = L, \cdots, 2, 1 \tag{A–2}$$

$$\mathbf{h}_i = \overrightarrow{\mathbf{h}}_i \oplus \overleftarrow{\mathbf{h}}_i \tag{A–3}$$

其中 $\mathrm{f}_{\mathrm{LSTM}}$ 和 $\mathrm{b}_{\mathrm{LSTM}}$ 分别表示正向和反向的 LSTM 计算单元，$\oplus$ 表示向量拼接操作，$\overrightarrow{\mathbf{h}}_0$ 和 $\overleftarrow{\mathbf{h}}_{L+1}$ 都默认为 $\mathbf{0}$ 向量。$\overrightarrow{\mathbf{h}}_i \in \mathbb{R}^n$ 是正向 LSTM 在第 i 时刻的隐层向量，$\overleftarrow{\mathbf{h}}_i \in \mathbb{R}^n$ 是反向 LSTM 在第 i 时刻的隐层向量，$\mathbf{h}_i \in \mathbb{R}^{2n}$ 是两个方向的隐层向量的拼接（即考虑了前后不同的上下文信息），其中 n 表示 LSTM 的隐层向量维度。为了方便阐述，我们将上述过程重新写为一个序列级的映射操作 BLSTM_{Θ}：

$$(\mathbf{h}_1, \cdots, \mathbf{h}_L) \leftarrow \mathrm{BLSTM}_{\Theta}(\mathbf{e}_1, \cdots, \mathbf{e}_L). \tag{A–4}$$

其中 Θ 表示该 BLSTM 的所有参数。

A.2　注意力机制

注意力机制[48, 168] 经常被用来获取序列级的特征向量或者在 “序列到序列” 的架构中获取上下文表示。给定一个序列的特征向量（$\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_L$）以及一个查询向量 $\mathbf{q}$，我们可以计算得到 $\mathbf{q} \in \mathbb{R}^n$ 和每个 $\mathbf{e}_i \in \mathbb{R}^m$（$i \in \{1, 2, \cdots, L\}$）之间的关注度权重 a_i：

$$a_i = \frac{\exp(u_i)}{\sum_{j=1}^{L} \exp(u_j)} \tag{A–5}$$

$$u_i = \mathbf{v}_a^{\top} \tanh(\mathbf{W}_a(\mathbf{q} \oplus \mathbf{e}_i) + \mathbf{b}_a) \tag{A–6}$$

其中 $\mathbf{v}_a \in \mathbb{R}^d$，$\mathbf{W}_a \in \mathbb{R}^{d \times (m+n)}$，$\mathbf{b}_a \in \mathbb{R}^d$ 均为相关参数，d 为注意力机制的内部隐层大小（一般与 n 或者 m 大小相近）。在计算得到关注度权重后，相关上下文向量

是所有输入向量的加权和：

$$\mathbf{z} = \sum_{i=1}^{L} a_i \mathbf{e}_i \tag{A–7}$$

同样地，为了简洁性，我们把上述过程也重写为一个序列级的映射过程 ATTN_{Θ}:

$$\mathbf{z}, (a_1, \cdots, a_L) \leftarrow \mathrm{ATTN}_{\Theta}(\mathbf{q}, (\mathbf{e}_1, \cdots, \mathbf{e}_L)). \tag{A–8}$$

其中 Θ 表示该注意力机制的所有参数。

参考文献

[1] YOUNG S, GAŠIĆ M, THOMSON B, et al. POMDP-Based Statistical Spoken Dialog Systems: A Review[J]. Proceedings of the IEEE, 2013, 101(5): 1160-1179.

[2] 俞凯, 陈露, 陈博, 等. 任务型人机对话系统中的认知技术——概念, 进展及其未来[J]. 计算机学报, 2015, 38(12): 2333-2348.

[3] BANGALORE S, HAKKANI-TÜR D, TUR G. Introduction to the special issue on spoken language understanding in conversational systems[J]. Speech Communication, 2006, 48(3): 233-238.

[4] MCTEAR M. Spoken language understanding for conversational dialog systems[C]. in: Proceedings of IEEE SLT. 2006: 6-6.

[5] DE MORI R, BECHET F, HAKKANI-TUR D, et al. Spoken language understanding[J]. IEEE Signal Processing Magazine, 2008, 25(3).

[6] TUR G, DE MORI R. Spoken language understanding: Systems for extracting semantic information from speech[M]. John Wiley & Sons, 2011.

[7] NIKLFELD G, FINAN R, PUCHER M. Architecture for adaptive multimodal dialog systems based on VoiceXML[C]. in: Proceedings of Seventh European Conference on Speech Communication and Technology. 2001: 2341-2344.

[8] NYBERG E, MITAMURA T, HATAOKA N. DialogXML: extending VoiceXML for dynamic dialog management[C]. in: Proceedings of the second international conference on Human Language Technology Research. 2002: 298-302.

[9] WARD W. Understanding spontaneous speech[C]. in: Proceedings of the workshop on Speech and Natural Language. 1989: 137-141.

[10] WANG Y, DENG L, ACERO A. Semantic frame-based spoken language understanding[G]. in: Spoken Language Understanding: Systems for Extracting Semantic Information from Speech, Tur and De Mori Eds. New York, NY, USA: Wiley, 2011: 35-80.

[11] WARD W. Extracting information in spontaneous speech[C]. in: Proceedings of Third International Conference on Spoken Language Processing. 1994: 83-86.

[12] WARD W, ISSAR S. Recent improvements in the CMU spoken language understanding system[C]. in: Proceedings of the workshop on Human Language Technology. 1994: 213-216.

[13] SENEFF S. TINA: A natural language system for spoken language applications[J]. Computational Linguistics, 1992, 18(1): 61-86.

[14] DOWDING J, GAWRON J M, APPELT D, et al. Gemini: A natural language system for spoken-language understanding[C]. in: Proceedings of ACL. 1993: 54-61.

[15] TRAUM D R. Speech acts for dialogue agents[G]. in: Foundations of rational agency. Springer, 1999: 169-201.

[16] YOUNG S. CUED standard dialogue acts[J]. Report, Cambridge University Engineering Department, 14th October, 2007, 2007.

[17] EUZENAT J, SHVAIKO P, et al. Ontology matching[M]. Springer, 2007.

[18] WOODS W A. Language Processing for Speech Understanding[R]. DTIC Document, 1983.

[19] PRICE P. Evaluation of spoken language systems: The ATIS domain[C]. in: Proceedings of the Third DARPA Speech and Natural Language Workshop. 1990: 91-95.

[20] ZETTLEMOYER L S, COLLINS M. Online Learning of Relaxed CCG Grammars for Parsing to Logical Form[C]. in: Proceedings of EMNLP-CoNLL. 2007: 678-687.

[21] HAHN S, DINARELLI M, RAYMOND C, et al. Comparing stochastic approaches to spoken language understanding in multiple languages[J]. IEEE Transactions on Audio, Speech, and Language Processing, 2011, 19(6): 1569-1583.

[22] JURCICEK F, MAIRESSE F, GAŠIC M, et al. Transformation-based Learning for Semantic parsing[C]. in: Proceedings of InterSpeech. 2009: 2719-2722.

[23] SCHWARTZ R, MILLER S, STALLARD D, et al. Language understanding using hidden understanding models[C]. in: Proceedings of ICSLP. 1996: 997-1000.

[24] HE Y, YOUNG S. Spoken language understanding using the hidden vector state model[J]. Speech Communication, 2006, 48(3): 262-275.

[25] WANG Y Y, ACERO A. Discriminative models for spoken language understanding[C]. in: Proceedings of ICSLP. 2006: 1766-1769.

[26] RAYMOND C, RICCARDI G. Generative and discriminative algorithms for spoken language understanding[C]. in: Proceedings of InterSpeech. 2007: 1605-1608.

[27] MAIRESSE F, GAŠIĆ M, JURČÍČEK F, et al. Spoken language understanding from unaligned data using discriminative classification models[C]. in: Proceedings of ICASSP. 2009: 4749-4752.

[28] LAFFERTY J, MCCALLUM A, PEREIRA F C. Conditional random fields: Probabilistic models for segmenting and labeling sequence data[C]. in: Proceedings of ICML. 2001: 282-289.

[29] JEONG M, GEUNBAE LEE G. Triangular-chain conditional random fields[J]. IEEE Transactions on Audio, Speech, and Language Processing, 2008, 16(7): 1287-1302.

[30] KIM Y. Convolutional Neural Networks for Sentence Classification[C]. in: Proceedings of EMNLP. Doha, Qatar, 2014: 1746-1751.

[31] LEE J Y, DERNONCOURT F. Sequential Short-Text Classification with Recurrent and Convolutional Neural Networks[C]. in: Proceedings of NAACL. San Diego, California, 2016: 515-520.

[32] LIN Z, FENG M, SANTOS C N D, et al. A structured self-attentive sentence embedding[J]. ArXiv preprint arXiv:1703.03130, 2017.

[33] RAVURI S V, STOLCKE A. Recurrent neural network and LSTM models for lexical utterance classification[C]. in: Proceedings of InterSpeech. 2015: 135-139.

[34] GRAVES A. Supervised sequence labelling[G]. in: Supervised sequence labelling with recurrent neural networks. Springer, 2012: 5-13.

[35] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[C]. in: Proceedings of NeurIPS. 2017: 5998-6008.

[36] YAO K, ZWEIG G, HWANG M Y, et al. Recurrent neural networks for language understanding[C]. in: Proceedings of InterSpeech. 2013: 2524-2528.

[37] MESNIL G, HE X, DENG L, et al. Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding[C]. in: Proceedings of InterSpeech. 2013: 3771-3775.

[38] HOCHREITER S. The vanishing gradient problem during learning recurrent neural nets and problem solutions[J]. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 1998, 6(02): 107-116.

[39] YAO K, PENG B, ZHANG Y, et al. Spoken language understanding using long short-term memory neural networks[C]. in: Proceedings of IEEE SLT. 2014: 189-194.

[40] CHUNG J, GULCEHRE C, CHO K, et al. Gated feedback recurrent neural networks[C]. in: Proceedings of ICML. 2015: 2067-2075.

[41] VUKOTIC V, RAYMOND C, GRAVIER G. A step beyond local observations with a dialog aware bidirectional GRU network for Spoken Language Understanding[C]. in: Proceedings of InterSpeech. 2016: 3241-3244.

[42] VU N T, GUPTA P, ADEL H, et al. Bi-directional recurrent neural network with ranking loss for spoken language understanding[C]. in: Proceedings of ICASSP. 2016: 6060-6064.

[43] ZHU S, YU K. Encoder-decoder with Focus-mechanism for Sequence Labelling Based Spoken Language Understanding[C]. in: Proceedings of ICASSP. 2017: 5675-5679.

[44] XU P, SARIKAYA R. Convolutional neural network based triangular crf for joint intent detection and slot filling[C]. in: Proceedings of IEEE ASRU. 2013: 78-83.

[45] VU N T. Sequential convolutional neural networks for slot filling in spoken language understanding[C]. in: Proceedings of InterSpeech. 2016: 3250-3254.

[46] YAO K, PENG B, ZWEIG G, et al. Recurrent conditional random field for language understanding[C]. in: Proceedings of ICASSP. 2014: 4077-4081.

[47] HUANG Z, XU W, YU K. Bidirectional LSTM-CRF models for sequence tagging[J]. ArXiv preprint arXiv:1508.01991, 2015.

[48] BAHDANAU D, CHO K, BENGIO Y. Neural machine translation by jointly learning to align and translate[C/OL]. in: Proceedings of ICLR. 2015. https://arxiv.org/abs/1409.0473.

[49] SIMONNET E, CAMELIN N, DELÉGLISE P, et al. Exploring the use of attention-based recurrent neural networks for spoken language understanding[C]. in: Proceedings of SLUNIPS. 2015.

[50] KURATA G, XIANG B, ZHOU B, et al. Leveraging Sentence-level Information with Encoder LSTM for Semantic Slot Filling[C]. in: Proceedings of EMNLP. Austin, Texas, 2016: 2077-2083.

[51] LIU B, LANE I. Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling[C]. in: Proceedings of InterSpeech. 2016: 685-689.

[52] PENG B, YAO K, JING L, et al. Recurrent Neural Networks with External Memory for Spoken Language Understanding[G]. in: Proceedings of NLPCC. Springer, 2015: 25-35.

[53] ZHAI F, POTDAR S, XIANG B, et al. Neural models for sequence chunking[C]. in: Proceedings of AAAI. 2017: 3365-3371.

[54] HAKKANI-TÜR D, TÜR G, ÇELIKYILMAZ A, et al. Multi-Domain Joint Semantic Frame Parsing Using Bi-Directional RNN-LSTM[C]. in: Proceedings of InterSpeech. 2016: 715-719.

[55] ZHANG X, WANG H. A Joint Model of Intent Determination and Slot Filling for Spoken Language Understanding[C]. in: Proceedings of IJCAI. New York, New York, USA, 2016: 2993-2999.

[56] KIM Y B, LEE S, STRATOS K. ONENET: Joint domain, intent, slot prediction for spoken language understanding[C]. in: Proceedings of IEEE ASRU. 2017: 547-553.

[57] ZHANG L, WANG H. Using Bidirectional Transformer-CRF for Spoken Language Understanding[C]. in: Proceedings of NLPCC. 2019: 130-141.

[58] ZHANG C, LI Y, DU N, et al. Joint Slot Filling and Intent Detection via Capsule Neural Networks[C]. in: Proceedings of ACL. 2019: 5259-5267.

[59] QIN L, CHE W, LI Y, et al. A Stack-Propagation Framework with Token-Level Intent Detection for Spoken Language Understanding[C]. in: Proceedings of EMNLP-IJCNLP. 2019: 2078-2087.

[60] LI C, LI L, QI J. A Self-Attentive Model with Gate Mechanism for Spoken Language Understanding[C]. in: Proceedings of EMNLP. Brussels, Belgium, 2018: 3824-3833.

[61] GOO C W, GAO G, HSU Y K, et al. Slot-gated modeling for joint slot filling and intent prediction[C]. in: Proceedings of NAACL. 2018: 753-757.

[62] HAIHONG E, NIU P, CHEN Z, et al. A novel bi-directional interrelated model for joint intent detection and slot filling[C]. in: Proceedings of ACL. 2019: 5467-5471.

[63] HENDERSON M, THOMSON B, WILLIAMS J D. The second dialog state tracking challenge[C]. in: Proceedings of SigDial. 2014: 263-272.

[64] HENDERSON M, GASIC M, THOMSON B, et al. Discriminative spoken language understanding using word confusion networks[C]. in: Proceedings of IEEE SLT. 2012: 176-181.

[65] LIU C, XU P, SARIKAYA R. Deep contextual language understanding in spoken dialogue systems[C]. in: Proceedings of InterSpeech. 2015.

[66] GUPTA R, RASTOGI A, HAKKANI-TUR D. An Efficient Approach to Encoding Context for Spoken Language Understanding[C]. in: Proceedings of InterSpeech. 2018: 3469-3473.

[67] CHEN Y N, HAKKANI-TÜR D, TÜR G, et al. End-to-End Memory Networks with Knowledge Carryover for Multi-Turn Spoken Language Understanding.[C]. in: Proceedings of InterSpeech. 2016: 3245-3249.

[68] JURAFSKY D. Speech & language processing[M]. Pearson Education India, 2000.

[69] ROBICHAUD J P, CROOK P A, XU P, et al. Hypotheses ranking for robust domain classification and tracking in dialogue systems[C]. in: Proceedings of InterSpeech. 2014: 145-149.

[70] KHAN O Z, ROBICHAUD J P, CROOK P A, et al. Hypotheses ranking and state tracking for a multi-domain dialog system using multiple ASR alternates[C]. in: Proceedings of InterSpeech. 2015: 2022-2026.

[71] ŠVEC J, ŠMÍDL L, VALENTA T, et al. Word-semantic lattices for spoken language understanding[C]. in: Proceedings of ICASSP. 2015: 5266-5270.

[72] LADHAK F, GANDHE A, DREYER M, et al. LatticeRnn: Recurrent Neural Networks Over Lattices[C]. in: Proceedings of InterSpeech. 2016: 695-699.

[73] HUANG C W, CHEN Y N. Adapting Pretrained Transformer to Lattices for Spoken Language Understanding[C]. in: Proceedings of IEEE ASRU. 2019: 845-852.

[74] HUANG C W, CHEN Y N. Learning Spoken Language Representations with Neural Lattice Language Modeling[C]. in: Proceedings of ACL. 2020: 3764-3769.

[75] HAKKANI-TÜR D, BÉCHET F, RICCARDI G, et al. Beyond ASR 1-best: Using word confusion networks in spoken language understanding[J]. Computer Speech & Language, 2006, 20(4): 495-514.

[76] TÜR G, DEORAS A, HAKKANI-TÜR D. Semantic parsing using word confusion networks with conditional random fields[C]. in: Proceedings of InterSpeech. 2013: 2579-2583.

[77] YANG X, LIU J. Using Word Confusion Networks for Slot Filling in Spoken Language Understanding[C]. in: Proceedings of InterSpeech. 2015: 1353-1357.

[78] JAGFELD G, VU N T. Encoding Word Confusion Networks with Recurrent Neural Networks for Dialog State Tracking[C]. in: Proceedings of the Workshop on Speech-Centric Natural Language Processing. 2017: 10-17.

[79] SHIVAKUMAR P G, GEORGIOU P. Confusion2Vec: towards enriching vector space word representations with representational ambiguities[J]. PeerJ Computer Science, 2019, 5: e195.

[80] SHIVAKUMAR P G, YANG M, GEORGIOU P. Spoken Language Intent Detection Using Confusion2Vec[C]. in: Proceedings of InterSpeech. 2019: 819-823.

[81] MASUMURA R, IJIMA Y, ASAMI T, et al. Neural Confnet Classification: Fully Neural Network Based Spoken Utterance Classification Using Word Confusion Networks[C]. in: Proceedings of ICASSP. 2018: 6039-6043.

[82] LIU B, LANE I. Joint Online Spoken Language Understanding and Language Modeling With Recurrent Neural Networks[C]. in: Proceedings of SigDial. Los Angeles, 2016: 22-30.

[83] ZHANG H, ZHU S, FAN S, et al. Joint Spoken Language Understanding and Domain Adaptive Language Modeling[C]. in: Proceedings of IScIDE. 2018: 311-324.

[84] SCHUMANN R, ANGKITITRAKUL P. Incorporating ASR Errors with Attention-Based, Jointly Trained RNN for Intent Detection and Slot Filling[C]. in: Proceedings of ICASSP. 2018: 6059-6063.

[85] ZHU S, LAN O, YU K. Robust Spoken Language Understanding with Unsupervised ASR-Error Adaptation[C]. in: Proceedings of ICASSP. Calgary, Canada: IEEE, 2018: 6179-6183.

[86] QIAN Y, UBALE R, LANGE P, et al. From Speech Signals to Semantics — Tagging Performance at Acoustic, Phonetic and Word Levels[C]. in: Proceedings of ISCSLP. 2018: 280-284.

[87] SERDYUK D, WANG Y, FUEGEN C, et al. Towards end-to-end spoken language understanding[C]. in: Proceedings of ICASSP. 2018: 5754-5758.

[88] GHANNAY S, CAUBRIÈRE A, ESTÈVE Y, et al. End-to-end named entity and semantic concept extraction from speech[C]. in: Proceedings of IEEE SLT. 2018: 692-699.

[89] HAGHANI P, NARAYANAN A, BACCHIANI M, et al. From Audio to Semantics: Approaches to end-to-end spoken language understanding[C]. in: Proceedings of IEEE SLT. 2018: 720-726.

[90] TOMASHENKO N, CAUBRIÈRE A, ESTÈVE Y, et al. Recent Advances in End-to-End Spoken Language Understanding[C]. in: Proceedings of SLSP. 2019: 44-55.

[91] PALOGIANNIDI E, GKINIS I, MASTRAPAS G, et al. End-to-end architectures for ASR-free spoken language understanding[C]. in: Proceedings of ICASSP. 2020: 7974-7978.

[92] WANG P, WEI L, CAO Y, et al. Large-Scale Unsupervised Pre-Training for End-to-End Spoken Language Understanding[C]. in: Proceedings of ICASSP. 2020: 7999-8003.

[93] LUGOSCH L, MEYER B, NOWROUZEZAHRAI D, et al. Using Speech Synthesis to Train End-to-End Spoken Language Understanding Models[C]. in: Proceedings of ICASSP. 2020: 8499-8503.

[94] PRICE R. End-To-End Spoken Language Understanding Without Matched Language Speech Model Pretraining Data[C]. in: Proceedings of ICASSP. 2020: 7979-7983.

[95] HUANG Y, KUO H K, THOMAS S, et al. Leveraging Unpaired Text Data for Training End-To-End Speech-to-Intent Systems[C]. in: Proceedings of ICASSP. 2020: 7984-7988.

[96] SARI L, THOMAS S, HASEGAWA-JOHNSON M. Training Spoken Language Understanding Systems with Non-Parallel Speech and Text[C]. in: Proceedings of ICASSP. 2020: 8109-8113.

[97] MIKOLOV T, SUTSKEVER I, CHEN K, et al. Distributed Representations of Words and Phrases and their Compositionality[C]. in: Proceedings of NeurIPS. 2013: 3111-3119.

[98] MIKOLOV T, CHEN K, CORRADO G, et al. Efficient Estimation of Word Representations in Vector Space[C/OL]. in: Proceedings of ICLR. 2013. https://arxiv.org/abs/1301.3781.

[99] PENNINGTON J, SOCHER R, MANNING C D. Glove: Global vectors for word representation[C]. in: Proceedings of EMNLP. 2014: 1532-1543.

[100] HASHIMOTO K, XIONG C, TSURUOKA Y, et al. A Joint Many-Task Model: Growing a Neural Network for Multiple NLP Tasks[C]. in: Proceedings of EMNLP. 2017: 1923-1933.

[101] PETERS M, NEUMANN M, IYYER M, et al. Deep Contextualized Word Representations[C]. in: Proceedings of NAACL. 2018: 2227-2237.

[102] DEVLIN J, CHANG M W, LEE K, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding[C]. in: Proceedings of NAACL. 2019: 4171-4186.

[103] SIDDHANT A, GOYAL A, METALLINOU A. Unsupervised transfer learning for spoken language understanding in intelligent agents[C]. in: Proceedings of AAAI: vol. 33. 2019: 4959-4966.

[104] CASTELLUCCI G, BELLOMARIA V, FAVALLI A, et al. Multi-lingual Intent Detection and Slot Filling in a Joint BERT-based Model[J]. ArXiv preprint arXiv:1907.02884, 2019.

[105] CHEN Q, ZHUO Z, WANG W. Bert for joint intent classification and slot filling[J]. ArXiv preprint arXiv:1902.10909, 2019.

[106] TUR G, HAKKANI-TÜR D, SCHAPIRE R E. Combining active and semi-supervised learning for spoken language understanding[J]. Speech Communication, 2005, 45(2): 171-186.

[107] LAN O, ZHU S, YU K. Semi-Supervised Training Using Adversarial Multi-Task Learning for Spoken Language Understanding[C]. in: Proceedings of ICASSP. Calgary, Canada: IEEE, 2018: 6049-6053.

[108] REI M. Semi-supervised Multitask Learning for Sequence Labeling[C]. in: Proceedings of ACL. 2017: 2121-2130.

[109] PETERS M, AMMAR W, BHAGAVATULA C, et al. Semi-supervised sequence tagging with bidirectional language models[C]. in: Proceedings of ACL. 2017: 1756-1765.

[110] KIM Y B, STRATOS K, KIM D. Adversarial adaptation of synthetic or stale data[C]. in: Proceedings of ACL. 2017: 1297-1307.

[111] ZHU S, LAN O, YU K. Robust spoken language understanding with unsupervised ASR-error adaptation[C]. in: Proceedings of ICASSP. 2018: 6179-6183.

[112] CHEN Y N, WANG W Y, RUDNICKY A I. Unsupervised induction and filling of semantic slots for spoken dialogue systems using frame-semantic parsing[C]. in: Proceedings of IEEE ASRU. 2013: 120-125.

[113] CHEN Y N, WANG W Y, RUDNICKY A I. Leveraging frame semantics and distributional semantics for unsupervised semantic slot induction in spoken dialogue systems[C]. in: Proceedings of IEEE SLT. 2014: 584-589.

[114] CHEN Y N, WANG W Y, GERSHMAN A, et al. Matrix Factorization with Knowledge Graph Propagation for Unsupervised Spoken Language Understanding.[C]. in: Proceedings of ACL. 2015: 483-494.

[115] HECK L, HAKKANI-TÜR D. Exploiting the semantic web for unsupervised spoken language understanding[C]. in: Proceedings of IEEE SLT. 2012: 228-233.

[116] HE D, XIA Y, QIN T, et al. Dual learning for machine translation[C]. in: Proceedings of NeurIPS. 2016: 820-828.

[117] CAO R, ZHU S, LIU C, et al. Semantic Parsing with Dual Learning[C]. in: Proceedings of ACL. 2019: 51-64.

[118] YE H, LI W, WANG L. Jointly Learning Semantic Parser and Natural Language Generator via Dual Information Maximization[C]. in: Proceedings of ACL. 2019: 2090-2101.

[119] SU S Y, HUANG C W, CHEN Y N. Dual Supervised Learning for Natural Language Understanding and Generation[C]. in: Proceedings of ACL. 2019: 5472-5477.

[120] SU S Y, HUANG C W, CHEN Y N. Towards Unsupervised Language Understanding and Generation by Joint Dual Learning[C]. in: Proceedings of ACL. 2020: 671-680.

[121] SUN M, LI X, LI P. Logician and Orator: Learning from the Duality between Language and Knowledge in Open Domain[C]. in: Proceedings of EMNLP. 2018: 2119-2130.

[122] ZHU S, CHEN L, SUN K, et al. Semantic parser enhancement for dialogue domain extension with little data[C]. in: Proceedings of IEEE SLT. 2014: 336-341.

[123] KURATA G, XIANG B, ZHOU B. Labeled Data Generation with Encoder-decoder LSTM for Semantic Slot Filling[C]. in: Proceedings of InterSpeech. 2016: 725-729.

[124] HOU Y, LIU Y, CHE W, et al. Sequence-to-Sequence Data Augmentation for Dialogue Language Understanding[C]. in: Proceedings of COLING. 2018: 1234-1245.

[125] YOO K M, SHIN Y, LEE S. Data Augmentation for Spoken Language Understanding via Joint Variational Generation[C]. in: Proceedings of AAAI: vol. 33. 2019: 7402-7409.

[126] XIE Q, DAI Z, HOVY E, et al. Unsupervised Data Augmentation for Consistency Training[J]. ArXiv preprint arXiv:1904.12848, 2019.

[127] YAZDANI M, HENDERSON J. A Model of Zero-Shot Learning of Spoken Language Understanding.[C]. in: Proceedings of EMNLP. 2015: 244-249.

[128] FERREIRA E, JABAIAN B, LEFÈVRE F. Zero-shot semantic parser for spoken language understanding[C]. in: Proceedings of InterSpeech. 2015: 1403-1407.

[129] FERREIRA E, JABAIAN B, LEFEVRE F. Online adaptative zero-shot learning spoken language understanding using word-embedding[C]. in: Proceedings of ICASSP. 2015: 5321-5325.

[130] ZHU S, YU K. Concept Transfer Learning for Adaptive Language Understanding[C]. in: Proceedings of SigDial. Melbourne, Australia: Association for Computational Linguistics, 2018: 391-399.

[131] BAPNA A, TÜR G, HAKKANI-TÜR D, et al. Towards Zero-Shot Frame Semantic Parsing for Domain Scaling[C]. in: Proceedings of InterSpeech. 2017: 2476-2480.

[132] LEE S, JHA R. Zero-shot adaptive transfer for conversational language understanding[C]. in: Proceedings of AAAI: vol. 33. 2019: 6642-6649.

[133] SHAH D, GUPTA R, FAYAZI A, et al. Robust Zero-Shot Cross-Domain Slot Filling with Example Values[C]. in: Proceedings of ACL. 2019: 5484-5490.

[134] LIU Z, WINATA G I, XU P, et al. Coach: A Coarse-to-Fine Approach for Cross-domain Slot Filling[C]. in: Proceedings of ACL. 2020: 19-25.

[135] CHEN Y N, HAKKANI-TÜR D Z, HE X. Zero-shot learning of intent embeddings for expansion by convolutional deep structured semantic models[C]. in: Proceedings of ICASSP. 2016: 6045-6049.

[136] RAJPURKAR P, ZHANG J, LOPYREV K, et al. SQuAD: 100,000+ Questions for Machine Comprehension of Text[C]. in: Proceedings of EMNLP. 2016: 2383-2392.

[137] LI X, FENG J, MENG Y, et al. A Unified MRC Framework for Named Entity Recognition[C]. in: Proceedings of ACL. 2020: 5849-5859.

[138] FEI-FEI L, FERGUS R, PERONA P. One-shot learning of object categories[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2006, 28(4): 594-611.

[139] VINYALS O, BLUNDELL C, LILLICRAP T, et al. Matching networks for one shot learning[C]. in: Proceedings of NeurIPS. 2016: 3630-3638.

[140] YAN L, ZHENG Y, CAO J. Few-shot learning for short text classification[J]. Multimedia Tools and Applications, 2018, 77(22): 29799-29810.

[141] YU M, GUO X, YI J, et al. Diverse few-shot text classification with multiple metrics[J]. ArXiv preprint arXiv:1805.07513, 2018.

[142] SUN S, SUN Q, ZHOU K, et al. Hierarchical Attention Prototypical Networks for Few-Shot Text Classification[C]. in: Proceedings of EMNLP-IJCNLP. 2019: 476-485.

[143] GENG R, LI B, LI Y, et al. Induction networks for few-shot text classification[C]. in: Proceedings of EMNLP-IJCNLP. 2019: 3895-3904.

[144] FRITZLER A, LOGACHEVA V, KRETOV M. Few-shot classification in named entity recognition task[C]. in: Proceedings of ACM SIGAPP. 2019: 993-1000.

[145] SNELL J, SWERSKY K, ZEMEL R. Prototypical networks for few-shot learning[C]. in: Proceedings of NeurIPS. 2017: 4077-4087.

[146] HOU Y, CHE W, LAI Y, et al. Few-shot Slot Tagging with Collapsed Dependency Transfer and Label-enhanced Task-adaptive Projection Network[C]. in: Proceedings of ACL. 2020.

[147] YOON S W, SEO J, MOON J. TapNet: Neural network augmented with task-adaptive projection for few-shot learning[C]. in: Proceedings of ICML. 2019: 7115-7123.

[148] JAECH A, HECK L, OSTENDORF M. Domain adaptation of recurrent neural networks for natural language understanding[C]. in: Proceedings of InterSpeech. 2016: 690-694.

[149] KIM Y B, STRATOS K, SARIKAYA R. Frustratingly Easy Neural Domain Adaptation.[C]. in: Proceedings of COLING. 2016: 387-396.

[150] LIU B, LANE I. Multi-Domain Adversarial Learning for Slot Filling in Spoken Language Understanding[J]. ArXiv preprint arXiv:1807.00267, 2018.

[151] KINOSHITA K, OCHIAI T, DELCROIX M, et al. Improving noise robust automatic speech recognition with single-channel time-domain enhancement network[C]. in: Proceedings of ICASSP. 2020: 7009-7013.

[152] WANG P, TAN K, et al. Bridging the gap between monaural speech enhancement and recognition with distortion-independent acoustic modeling[J]. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2019, 28: 39-48.

[153] CHANG X, ZHANG W, QIAN Y, et al. End-To-End Multi-Speaker Speech Recognition with Transformer[C]. in: Proceedings of ICASSP. 2020: 6134-6138.

[154] Von NEUMANN T, KINOSHITA K, DRUDE L, et al. End-to-end training of time domain audio separation and recognition[C]. in: Proceedings of ICASSP. 2020: 7004-7008.

[155] SCHUSTER M, PALIWAL K K. Bidirectional recurrent neural networks[J]. IEEE Transactions on Signal Processing, 1997, 45(11): 2673-2681.

[156] VINYALS O, FORTUNATO M, JAITLY N. Pointer networks[C]. in: Proceedings of NeurIPS. 2015: 2692-2700.

[157] GULCEHRE C, AHN S, NALLAPATI R, et al. Pointing the Unknown Words[C]. in: Proceedings of ACL. 2016: 140-149.

[158] SEE A, LIU P J, MANNING C D. Get To The Point: Summarization with Pointer-Generator Networks[C]. in: Proceedings of ACL. 2017: 1073-1083.

[159] BA J L, KIROS J R, HINTON G E. Layer normalization[J]. ArXiv preprint arXiv:1607.06450, 2016.

[160] KINGMA D P, BA J. Adam: A method for stochastic optimization[C/OL]. in: Proceedings of ICLR. 2015. https://arxiv.org/abs/1412.6980.

[161] WILLIAMS J D. Web-style ranking and SLU combination for dialog state tracking[C]. in: Proceedings of SigDial. 2014: 282-291.

[162] BARAHONA L M R, GASIC M, MRKŠIĆ N, et al. Exploiting Sentence and Context Representations in Deep Neural Models for Spoken Language Understanding[C]. in: Proceedings of COLING. 2016: 258-267.

[163] ZHAO L, FENG Z. Improving slot filling in spoken language understanding with joint pointer and attention[C]. in: Proceedings of ACL. 2018: 426-431.

[164] MA X, HOVY E. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF[C]. in: Proceedings of ACL. 2016: 1064-1074.

[165] KURATA G, XIANG B, ZHOU B, et al. Leveraging Sentence-level Information with Encoder LSTM for Semantic Slot Filling[C]. in: Proceedings of EMNLP. 2016: 2077-2083.

[166] WEAVER L, TAO N. The Optimal Reward Baseline for Gradient-Based Reinforcement Learning[C]. in: Proceedings of UAI. 2001: 538-545.

[167] MENG Y, LI X, SUN X, et al. Is Word Segmentation Necessary for Deep Learning of Chinese Representations?[C]. in: Proceedings of ACL. 2019: 3242-3252.

[168] LUONG M T, PHAM H, MANNING C D. Effective Approaches to Attention-based Neural Machine Translation[C]. in: Proceedings of EMNLP. 2015: 1412-1421.

[169] VINYALS O, LE Q. A neural conversational model[J]. ArXiv preprint arXiv:1506.05869, 2015.

[170] WEN T H, GAŠIĆ M, MRKŠIĆ N, et al. Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems[C]. in: Proceedings of EMNLP. 2015: 1711-1721.

[171] LEE D H. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks[C]. in: Proceedings of ICML Workshop on challenges in representation learning: vol. 3. 2013: 2.

[172] CELIKYILMAZ A, SARIKAYA R, HAKKANI-TÜR D, et al. A New Pre-Training Method for Training Deep Learning Models with Application to Spoken Language Understanding.[C]. in: Proceedings of InterSpeech. 2016: 3255-3259.

[173] XIE Q, LUONG M T, HOVY E, et al. Self-training with noisy student improves imagenet classification[C]. in: Proceedings of CVPR. 2020: 10687-10698.

[174] SUTTON R S, MCALLESTER D A, SINGH S P, et al. Policy gradient methods for reinforcement learning with function approximation[C]. in: Proceedings of NeurIPS. 2000: 1057-1063.

[175] SUTTON R S, BARTO A G. Reinforcement learning: An introduction[M]. MIT press, 2018.

[176] MIKOLOV T, KARAFIÁT M, BURGET L, et al. Recurrent neural network based language model[C]. in: Proceedings of InterSpeech. 2010: 1045-1048.

[177] WU Y, SCHUSTER M, CHEN Z, et al. Google's neural machine translation system: Bridging the gap between human and machine translation[J]. ArXiv preprint arXiv:1609.08144, 2016.

[178] PAPINENI K, ROUKOS S, WARD T, et al. BLEU: A method for automatic evaluation of machine translation[C]. in: Proceedings of ACL. 2002: 311-318.

[179] HEMPHILL C T, GODFREY J J, DODDINGTON G R, et al. The ATIS spoken language systems pilot corpus[C]. in: Proceedings of the DARPA speech and natural language workshop. 1990: 96-101.

[180] COUCKE A, SAADE A, BALL A, et al. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces[J]. ArXiv preprint arXiv:1805.10190, 2018.

[181] HAKKANI-TÜR D, TUR G, CELIKYILMAZ A, et al. Multi-Domain Joint Semantic Frame Parsing using Bi-directional RNN-LSTM[C]. in: Proceedings of InterSpeech. 2016: 715-719.

[182] BERANT J, LIANG P. Semantic Parsing via Paraphrasing[C]. in: Proceedings of ACL. 2014: 1415-1425.

[183] JIA R, LIANG P. Data Recombination for Neural Semantic Parsing[C]. in: Proceedings of ACL. 2016: 12-22.

[184] DONG L, LAPATA M. Coarse-to-Fine Decoding for Neural Semantic Parsing[C]. in: Proceedings of ACL. 2018: 731-742.

[185] WANG Y, BERANT J, LIANG P. Building a semantic parser overnight[C]. in: Proceedings of ACL. 2015: 1332-1342.

[186] BLACK P E. Ratcliff/Obershelp pattern recognition[Z]. In *Dictionary of Algorithms and Data Structures* [online], Paul E. Black, ed. Available from: https://www.nist.gov/dads/HTML/ratcliffObershelp.html (accessed 20 May 2019). 2004.

[187] VINYALS O, KAISER Ł, KOO T, et al. Grammar as a foreign language[C]. in: Proceedings of NeurIPS. 2015: 2773-2781.

[188] HENDERSON M, THOMSON B, WILLIAMS J D. The third dialog state tracking challenge[C]. in: Proceedings of IEEE SLT. 2014: 324-329.

[189] DAUMÉ III H. Frustratingly Easy Domain Adaptation[C]. in: Proceedings of ACL. 2007: 256-263.

[190] KIM Y B, STRATOS K, KIM D. Domain attention with an ensemble of experts[C]. in: Proceedings of ACL. 2017: 643-653.

[191] JHA R, MARIN A, SHIVAPRASAD S, et al. Bag of experts architectures for model reuse in conversational language understanding[C]. in: Proceedings of NAACL. 2018: 153-161.

[192] COLLOBERT R, WESTON J, BOTTOU L, et al. Natural language processing (almost) from scratch[J]. Journal of Machine Learning Research, 2011, 12(Aug): 2493-2537.

[193] KIM Y B, STRATOS K, SARIKAYA R, et al. New Transfer Learning Techniques for Disparate Label Sets[C]. in: Proceedings of ACL-IJCNLP. 2015: 473-482.

[194] MA Y, CAMBRIA E, GAO S. Label embedding for zero-shot fine-grained named entity typing[C]. in: Proceedings of COLING. 2016: 171-180.

[195] LEE J, KIM D, SARIKAYA R, et al. Coupled Representation Learning for Domains, Intents and Slots in Spoken Language Understanding[C]. in: Proceedings of IEEE SLT. 2018: 714-719.

[196] PETERS M, NEUMANN M, IYYER M, et al. Deep Contextualized Word Representations[C]. in: Proceedings of NAACL. 2018: 2227-2237.

[197] ORESHKIN B, LÓPEZ P R, LACOSTE A. TADAM: Task dependent adaptive metric for improved few-shot learning[C]. in: Proceedings of NeurIPS. 2018: 721-731.

[198] SANG E T K, DE MEULDER F. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition[C]. in: Proceedings of NAACL. 2003: 142-147.

[199] ZELDES A. The GUM corpus: creating multilayer resources in the classroom[J]. Language Resources and Evaluation, 2017, 51(3): 581-612.

[200] DERCZYNSKI L, NICHOLS E, van ERP M, et al. Results of the WNUT2017 shared task on novel and emerging entity recognition[C]. in: Proceedings of the 3rd Workshop on Noisy User-generated Text. 2017: 140-147.

[201] PRADHAN S, MOSCHITTI A, XUE N, et al. Towards robust linguistic analysis using ontonotes[C]. in: Proceedings of CoNLL. 2013: 143-152.