学校代码：10285

学　　号：20144227003

蘇州大學

SOOCHOW UNIVERSITY

# 硕士学位论文

（学术学位）


基于多形态非主属性数据的实体匹配算法研究

Entity Matching Based on Polymorphic Non-Key Attributes

研 究 生 姓 名　杨强

指导教师姓名　李直旭

专 业 名 称　计算机科学与技术

研 究 方 向　数据管理与数据分析

所 在 院 部　计算机科学与技术学院

论文提交日期　2017 年 3 月

# 苏州大学学位论文独创性声明

本人郑重声明：所提交的学位论文是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含其他个人或集体已经发表或撰写过的研究成果，也不含为获得苏州大学或其它教育机构的学位证书而使用过的材料。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式标明。本人承担本声明的法律责任。

论文作者签名：_____日　期：_____

# 苏州大学学位论文使用授权声明

本人完全了解苏州大学关于收集、保存和使用学位论文的规定，即：学位论文著作权归属苏州大学。本学位论文电子文档的内容和纸质论文的内容相一致。苏州大学有权向国家图书馆、中国社科院文献信息情报中心、中国科学技术信息研究所（含万方数据电子出版社）、中国学术期刊（光盘版）电子杂志社送交本学位论文的复印件和电子文档，允许论文被查阅和借阅，可以采用影印、缩印或其他复制手段保存和汇编学位论文，可以将学位论文的全部或部分内容编入有关数据库进行检索。

涉密论文□

本学位论文属　　在_____年___月解密后适用本规定。

非涉密论文□

论文作者签名：_____日　期：_____

导 师 签 名：_____日　期：_____

# 基于深度学习技术的信息抽取工具的研究与开发

## 摘　　要

信息抽取表示自动的从文本中抽取出结构化的信息，比如实体，实体间的关系，事件或者包含在有噪音的非结构化文本中的属性描述性实体，将非结构化文本抽取出有用的信息存储在结构化形式中，比如表格，XML，图谱，有着非常重要的价值，信息抽取是构建知识图谱的基础理论和实践支撑。信息抽取又包含了若干个子问题，比如，命名体识别，开放域信息抽取，关系抽取以及文本分割。分割式信息抽取是一个特殊的信息抽取问题，它是通过分割半结构化文本信息来抽取出隐含其中的属性值。

目前主流的解决方案是采用机器学习的方法，包括使用人工标注的训练集使用监督式机器学习方案，或者利用事先存在的知识库来辅助实施非监督式式的机器学习方案，在非监督式方法中，一个杰出的解决方案是借助于知识库，讲整个信息抽取流程分为分割，匹配，调整等若干个模块，取得了目前为止最好的表现。但是，当采用监督式方案是，获取标注好的训练集要花费非常昂贵的代价，并且往往只限制在某个领域内。而通过知识库辅助的方案，则会出现两个主要问题，1. 固定领域文本的属性顺序限制在固定的顺序上，2.匹配准确率底下。

为了解决上述方案的问题，本文提出了结合深度学习和概率模型的方案，充分利用深度学习强大的特征抽取和组合能力，并结合概率模型的适用性和可解释性。构建了完整的高效的解决方案。本文提出的方法同样借助知识库来克服人工标注的问题，并且有效克服了匹配率底下以及限定属性顺序的问题。具体研究内容如下：

(1)如何选择深度学习模型，如何将深度学习模型与概率模型完美的结合起来，并探索深度学习模型构建和优化的策略，包括构建有效的丰富的特征。

(2)提出了基于卷积神经网络的贪婪式概率匹配算法。本算法采用深度卷积神经网络模型，提出了一种贪婪式的概率匹配算法，并在抽取过程中学习出一个双向的序列与位置模型来完成抽取任务，采用了步步提升的设计方案，有效的解决了上述的问题，经过比较F1值取得了非常明显的提升，并在抽取效率上也有非常好的表现。

(3)提出CNN+LSTM+attention model+CRF的解决方案

**关键词：** 深度学习, 信息抽取, 文本分割

**作　　者：** 胡猛

**指导教师：** 李直旭

# Information Extraction based on Deep Learning technology

## Abstract

With the rapid development of information technology, the data from different field explosively grow, which makes data quality problems increasingly highlight, such as data distortion, data staleness, data missing, inconsistent data expression and so on. This paper mainly do research on one of the important data quality subjects, so-called Entity Matching (EM for short). EM aims at identifying records referring to the same entity within or across databases.

So far, most existing EM algorithms depend on string similarity metrics to measure the similarity between key attribute values of entities and then make decisions according to a predefined similarity threshold. But an arbitrary threshold is bad for either the matching precision or the recall.

So as to solve the problems of the existing methods, we propose entity matching algorithms based on polymorphic non-key attributes by analysing textual data and computing similarity do EM. Our methods are orthodox to the existing EM methods based on key attributes. We mainly pay attention on how to use non-key attributes smartly to improve the precision and recall of EM based on key attribute only. More details are shown as follows:

(1) We focus on the problem of EM in the paper. Some existing EM methods are introduced here and the advantages and disadvantages of them are also analysed.

(2) We propose non-key attributes based EM algorithms which select non-key attributes according to their identification ability to do EM. With the proposed methods, we can not only solve the problem of different expression but also overcome the problem of missing values.

(3) We propose textual data based EM algorithms which mine the key information from textual data. The precision and recall of EM using the proposed methods are greatly improved .

We demonstrate the effectiveness and availability of the proposed methods on on real-

world datasets. Our empirical study shows that our proposed EM methods outperform the state-of-the-art EM methods by reaching a higher EM precision and recall. And the efficiency of EM is also improved greatly by employing the proposed data block algorithm to reduce the times of comparison.

**Keywords:** Data Quality, Entity Matching, Polymorphic Non-key Attributes Data, Accuracy, Efficiency

**Written by** Meng Hu

**Supervised by** Zhixu Li

# 目　录

# 第一章 绪论

## 1.1 课题研究背景

在这个瞬息万变的信息时代，我们接触的信息量呈现爆炸式的增长，特别是文本信息，它们在购物网站，新闻网站，论坛交流以及社交活动等中被广泛地创造出来，是一种最丰富也是最杂乱的信息格式。通常，这些文本信息可以免费获取，但是却覆盖了各种主题和内容，内容的书写格式也是各种各样，没有任何一个统一的严格要求的的出版格式。图1展示了几种在互联网上常见的文本信息来源。大量有价值的信息和知识就隐藏在这些杂乱的文本信息中，如何从文本中抽取出有价值的东西是一个非常重要的任务。

这些信息来源蕴含了大量的各种各样的有价值的数据，根据信息种类，我们能从中找到个人信息，商品信息，出版信息，公司信息，城市信息，天气信息等，我们从这些信息中便可以推理出一些有价值的关系，去认识用户的偏好，来检测某种趋势或者去命名一下东西等。但是，文本多样性在赋予文本信息丰富的表述意义的同时，也带来了更大的限制，这种限制体现在这些文本信息中存储的信息和知识很难被通用化的组织起来。更特殊的一种情形，对于一些语义丰富的文本段，比如商品信息描述，引文信息，地址信息，分类广告，电影评论等等，这些文本来源也是一种无结构的形式，所以抽取文本内容很难找到某种自动化处理的方法。此外，在这种文本类型中，隐含其中的信息是由若干个没有关系的个体属性值组合起来的，这就更增加了抽取的难度。

现如今人工智能的发展，在算法和硬件上都得到了长足的进步，但现在的人工智能大多还处于感知智能阶段，比如如何去识别一个东西，但是从感知智能到认知智能的进化，就需要赋予机器认知能力。人之所以具备认知能力，就是因为人的大脑中存储了大量的知识，这些知识不断的增加，被完美的组织起来，并且可以流畅的推理和运用这些知识。而对机器来讲，构建知识图谱——通过不同知识的关联性形成成一个网状的知识结构，这个过程的本质就是建立认知，理解世界。现在越来越多的人意识到知识图谱的对于人工智能发展的意义，构建知识图谱是一个庞大的工程，这个工程的第一步便是信息抽取，或者称为知识获取，包括命名题识别，关

系抽取，属性值抽取，构建三元组等。无疑，信息抽取的能力直接决定了知识图谱构建的准确性和多样性，为了构建一个准确的、丰富的知识图谱，首先需要我们能够从各种各样的文本信息中抽取出有价值、有联系的知识点。在如今的信息洪流中，有无穷无尽的知识点蕴藏在这些文本中，这导致了信息抽取的必要性和价值，但是也因为文本格式和内容的多样性，又导致了信息抽取的棘手性和难度。分割式信息抽取作为信息抽取中重要的一类，这种信息抽取的文本格式具有无噪音、属性值间无关联等特点，是一种复杂又重要的抽取任务，同样具有相同重要的研究价值。因此，如何建立快速、准确的信息抽取模型成为了研究如何构建知识工程和知识图谱的重点和难点问题。

## 1.2　课题研究现状

信息抽取大致分类可以从两个层次分析，一是封闭语料和开放语料的区别，即文本信息源来自限定领域的新闻语料和整个Web页面，二是限定类别和开放类别的区别，即有限的实体、关系、事件和维基百科级别的条目。分割式信息抽取，属于封闭式语料和限定类别的信息抽取任务，由于它在实际应用中的重要性，有很多研究致力于解决这个问题，主要的解决方案是使用机器学习的技术，分为监督式[1,2,3,4,5]的和非监督式的[1,2,4,5]。

在监督式机器学习方法中，基于隐马尔可夫模型和随即向量场模型的概率图模型被广泛应用，目前表现最好的是基于随即向量场模型的解决方案，但是，取得了非常好的效果，但是，当采用监督式机器学习方法时，需要大量的标注数据，这在实际的而应用中将付出非常昂贵的代价，并且在某些领域根本无法构建标注语料。

为了避免监督式机器学习方法的缺点，提前构建一个语料库，就是固定领域的文本信息，把这个语料库看做一个参考表，实现非监督的抽取模型。简要来说，首先使用机器学习模型，比如隐马尔可夫模型或者随机向量场模型，先从这个语料库中学习到这一语料的属性值顺序，然后使用这个模型在这个领域新的文本中做抽取任务，这中方案限制了属性值顺序，只能应付固定的应用场景，对于属性值顺序不固定的领域便无法适用。最近出现了一种使用知识库，这种知识库中存的不再是整句文本记录，而是将某一领域的各个属性值分开存储，不显性的训练模型来实现非

监督的抽取模型。文献[1]提出了一种基于这种知识库的非监督式方法，输入文本首先基于知识库做初始的分割，得到很多粗略的文本块，这些文本块再通过一些固定的匹配方程来匹配到指定的属性上，最后通过一个位置和序列模型做进一步的修改和补充。但是由于这些匹配函数表征力的缺失，以及这种位置和序列模型的先天缺陷，使得这种方法的抽取质量差强人意，但是这种方案无疑提供了一种很好的实施非监督式模型的解决思路。

上述方法虽然可以解决部分领域或部分场景的信息抽取任务，但仍旧存在一定的缺陷与不足。因此，如何引入深度学习模型，并将深度学习模型和概率模型结合来更好的完成分割式信息抽取的任务，并以小见大，探索深度学习在信息抽取领域更多场景的使用，是本文的研究重点。

## 1.3 课题研究内容

本文研究了基于深度学习模型的分割式信息抽取问题，提出了结合深度卷积神经网络和概率模型的非监督式信息抽取模型，大大提高了抽取质量，抽取效率也得到了明显的提升。此外，还搭建了一个定制训练模型、执行抽取任务、展示抽取结果的可视化系统，可以更方便地将此模型移植到更多场景，覆盖更多的领域，并简化训练模型的操作难度。本文主要解决三个问题：(1)如何引入深度学习模型，如何通过构造特征将文本信息输入变成丰富的向量化表示。(2)如何将深度学习模型的结果与知识库结合起来，实现文本分割和属性标注。(3)如何更好的学习到输入文本信息的内在规律和特征表述，用以进一步优化抽取表现。本文的主要内容如下：

(1)本文主要研究分割式信息抽取问题，介绍已有的信息抽取方法，描述具有代表性的各类算法和优势，并分析其中存在的问题。

(2)介绍一些关于深度学习和信息抽取相关的理论知识和研究现状。

(3)提出了一种基于深度学习模型的贪婪式概率标注算法。传统的机器学习算法受限于标注语料缺失、应用场景限制、匹配函数表征力弱等影响，使得抽取的准确率难以得到保障。而随着应用场景的扩增，越来越多领域的文本信息需要被考虑进来，面对的文本格式也越来越杂乱，因此必须提出一种适用更强、准确率更高的模型。另外，我们需要处理大量的文本信息，因此抽取效率也是不得不考虑的一个重

要方面。因此，本文开创性的使用深度学习模型来解决分割式信息抽取这个问题；提出了一种基于深度学习模型输出结果的贪婪式概率标注算法；进一步改进了现有方法，提出了一种双向的序列与位置模型。

(4)介绍可操控和可视化的系统，包括如何定制模型训练参数，查看模型训练过程，查看抽取各步骤的结果，查看学习到的双向位置和序列模型等。

## 1.4　课题研究意义

文本信息抽取是指从自然语言文本中自动抽取指定类型的实体，关系，事件等信息的应用技术。面对日益增多的海量信息，人们迫切需要一种自动化工具来帮助自己从这些海量文本信息中快速发现有价值的信息，并将这些信息自动化的进行分类，提取，重构并最终存储在结构化的容器中，便可以做进一步的查询，处理和分析。在这种背景下，信息抽取技术应运而生，信息抽取处理的对象可以是文本，图像，语音和视频等多种媒体，但通常指的是文本信息抽取。

信息抽取技术作为很多现实应用的支撑和补充，具有非常高的研究意义。从满足用户需求来讲，信息检索是一项综合了各种技术的重要问题，用于从一个大的文档集合中找出用户需求的相关文档，信息抽取技术可以从文档中抽取出粒度更小的实体，关系或事件，变可以满足用户更细粒度的请求和检索，从这个角度来讲，信息抽取可以作为信息索引的有益补充。信息抽取是随着互联网发展而兴起的信息处理技术，信息抽取是获取额外信息的有力补充手段，在人工智能的发展中，特别是从感知智能进化到认知智能，知识的获取和应用是至关重要的环节，知识是人工智能的基石，只有赋予机器更多知识，并让这些知识融会贯通，才能让机器学会推理和认知，知识图谱看做机器的知识库，被赋予了很多的期待。要想从更多的信息源、更复杂的信息格式、更多的信息量中准确的、高效的抽取出知识，信息抽取是至关重要的第一步。如何从巨大的网络信息洪流中抓获有用信息，构建一个覆盖面更广的知识图谱，是一个非常重要的研究方向，目前也是一个非常困难的问题。

正因为文本信息来源的多样性和文本格式的复杂性，传统的信息抽取方法渐渐展现出很多不足之处，包括抽取质量差，应用领域受限制或者抽取效率低等。随着深度学习算法的发展，首先在图像和语音上取得了突破性的进展，现如今已经成为

发展人工智能的基础算法，越来越多成熟的解决方案都采用深度学习算法，深度学习以其强大的特征抽取和特征组合特质，具有意想不到的表征能力。现在，深度学习也被越来越多的应用到处理自然语言上，信息抽取作为自然语言理解领域重要的一部分，同样也不会错过深度学习这个强大的工具。因此，探索如何将深度学习应用到本文的研究重点——分割式信息抽取中，具有很重要的研究意义和现实意义。

## 1.5 文章组织结构

全文共分为五章，其组织结构如下：

第一章简要介绍了该课题的研究背景、研究现状、研究内容、研究意义与文章的组织结构。

第二章介绍了一些相关理论及方法，给出了信息抽取的相关概念，并结合例子阐述了分割式信息抽取问题。此外，介绍了深度学习的一些理论基础，着重介绍了深度学习在自然语言理解上的应用。通过例子说了信息抽取遇到的主要问题，面临的挑战。最后介绍了一些常见的分割式信息抽取方法，并简述了各个方法的优缺点。

第三章介绍了本文提出的基于深度学习的贪婪式概率标注模型，首先给出问题定义，然后介绍如何构建深度学习模型，介绍如何结合知识库和深度学习模型结果来处理分割和标注问题，并介绍本文提出的贪婪式概率标注算法，最后介绍双向序列和位置模型，用来对之前的标注结果做进一步的补充和修改。

第四章完整信息抽取系统的可视化页面展示，包括在线定制参数，模型训练结果可视化，提出方法的抽取过程展示，最终结果的展示等。

第五章对本文所做的工作进行总结，并对未来的工作进行展望。

# 第二章　相关理论及方法

本章首先介绍了深度学习用于自然语言处理的一般处理方案，介绍了深度学习在一些信息抽取领域的应用，并分析了深度学习用于自然语言理解上的优缺点。然后介绍了信息抽取的相关概念及其定义，重点阐述了分割式信息抽取的问题定义。最后详细介绍了常见的分割式信息抽取方法，并分析了各类方法的优缺点。

## 2.1　深度学习在自然语言处理上的应用

最近几年来，深度学习架构和算法在诸如图像识别和语音处理上取得了突破性的进展，不断刷新各个榜单，应用场景越来越多，在工业界也有越来越多的落地产品，可以说深度学习的发展引导了这一波人工智能的热潮。图像和语音是自然界的产物，它的特征是一种更自然更丰富的表征，而深度神经网络的优势就在于它的特征抽取和特征组合能力，这是深度学习能在图像和语音上取得成功的原因。然后，人类的语言，属于人类文明创造的事物，不具备自然的表征能力，因此表达形式更主观，具有高度结构化、高抽象化、数据量相对小等特点，是一种更粗粒度的表现形式。因此，一开始，深度学习在自然语言处理领域的应用效果非常一般，随着算法的发展和思路的提升，特别是词向量的引入使得我们可以更好地将自然语言向量化，现在已经在越来越多的自然语言处理任务中证明，深度学习可以有更优秀的表现，甚至是最佳的表现。比如，深度神经网络模型在诸如文本分类，关系抽取，命名体识别，机器翻译等任务中的表现已经大大超越了传统方法，并且深度学习也在继续拓展它在自然语言理解上的应用。

在很长一段时间，自然语言处理的研究方法都是采用这些浅层的模型，来学习到非常高纬且稀疏的特征表示。在传统的机器学习中，我们使用各种算法的基础是需要手工设计特征，因此特征工程是一个非常基础性的工作。只有当人们对特定领域的知识有非常透彻的理解时，才能构造出足够多、足够优质的特征。比如，对于命名题识别任务，当要识别地名和机构名，我们首先需要构造出如下的特征列表：然后将特征喂给某个机器学习算法，比如线性分类器，分类器构造出目标函数，再通过凸优化策略不断调整模型的权重和偏置，使误差优化到最小，在这样的过程中为这些特征找到最合适的权重。这样的人工设计的特征，常常需要定义过多，并且

| Feature | NER |
|---|---|
| Current Word | ✓ |
| Previous Word | ✓ |
| Next Word | ✓ |
| Current Word Character n-gram | all |
| Current POS Tag | ✓ |
| Surrounding POS Tag Sequence | ✓ |
| Current Word Shape | ✓ |
| Surrounding Word Shape Sequence | ✓ |
| Presence of Word in Left Window | size 4 |
| Presence of Word in Right Window | size 4 |

图 2-1 命名题识别任务特征列表

一般是不可能做到完整的，需要花费大量的时间去设计和验证。

然而在自然语言处理上，由于语言的特殊性，设计有价值的特征是非常困难的事情，想要从如此抽象的文本信息中抽取出有用的特征，必须经过不断的迭代和实验，这是非常耗时的。深度学习是一种端到端的模型，我们只需要提供输入，不需要做额外的特征工程，深度神经网络会自动的进行特征抽取、特征组合。我们知道，图像信息是以像素点作为表示单位，声音是以声波作为表示单位，这些都输入较为底层的原始输入信号，可以直接数字化，作为深度神经网络的输入，进行端到端的模型训练。然而，一句文本却是以一个独立的单词或字组成的，虽然单词或字是独立的，但是组合成一起就成为了一句有意义的表述，如何将一句文本向量化表述，同时体现单词之间和字之间的联系，是一个重要的切入点。近些年，基于稠密向量表示的深度神经网络在很多自然语言处理任务上取得了更好的表现，也就是词向量的横空出世，可以说是将深度学习引入到自然语言处理领域的重要转折点。下面介绍一下词向量以及词向量结合深度学习在自然语言中的应用。

### 2.1.1 词向量

想要将自然语言理解问题转化成机器学习问题，首先需要将自然语言数字化，就如何将图像、语音数字化相同。最直观的一种方式，也是最传统的方法——采

用one-hot 的编码，将每个单词表示成$R^{|V|\times 1}$ 纬的向量，其中$|V|$ 表示词典中词语的个数，对于第$i$ 个词语的向量，只有$i$ 下标处为1，其它纬都为0。比如'food', 'eat', 'laptop' 作为前三个词，则他们的one-hot 向量表示为：

$$w^{food} = [1,0,0,0,\cdots,0] \qquad w^{eat} = [0,1,0,0,\cdots,0] \qquad w^{laptop} = [0,0,1,0,\cdots,0] \qquad \cdots$$

这种向量的表示形式，实现了将自然语言数字化的目的，在实际的应用中，配以Hash的处理，再结合一些机器学习算法就可以很好的解决自然语言处理领域各种主要的任务了。但是它存在两个最主要的问题：

(1)纬度灾难[Bengio 2003]。当某一语料词典数目过大，这个向量的纬度会变得很大，数据会变得特别稀疏，导致统计语言模型会出现很多为零的条件概率，这需要花费大量的精力来处理零概率的问题。这导致在训练模型进行矩阵时计算会非常困难，特别是应用在深度学习模型上，容易导致纬度灾难。

(2)不能很好地刻画单词之间的相似性。也就是常常说的'词汇鸿沟',对于两个词性或者语义上有关联的词语，他们的向量表示应该能体现出一些关联性和区分性，但是使用ont-hot 编码无法实现这个目的。比如对于'food','eat','laptop'这三个词：

$$(w^{food})^T \times w^{eat} \quad = \quad (w^{eat})^T \times w^{laptop} \quad = \quad 0$$

他们的向量相乘结果是相同的，且都等于0，这就无法体现'food'和'eat'之间的相关性，也不能体现这三者的区分性了。

因此我们想要去学习出一种使用低纬度空间的分散式表示来向量化一个词语，同时又需要将相似性和关联性融入到这个低维向量中，这就需要我们用语言模型来训练这个词向量，因此构建词向量的原理依据就是具有相似含义的词语往往出现在相似的语境之中。使用神经网络训练语言模型在[Bengio 2003]中被首次提出，作者阐述了传统基于统计的语言模型的主要问题，并说明了使用神经网络训练语言模型的优势。词向量得到革命性的发展是[Mikolov et al]提出的CBOW模型和Skim-gram模型，也就是著名的**word2vec**。

首先对词典中的所有单词进行one-hot编码，词典中词语数目为$V$。CBOW 模型

是使用周围的$2C$ 个词语来预测目标词语，比如对于一句话"狗喜欢吃熟的骨头"，我们可以将{′Σ′′U′,′ œ′,′ ¯′} 看做周围语境(两边各$C$ 个词语)，来预测和生成目标词语{′′}，而Skim-gram 模型是跟这个思路相反的，它使用目标词语来预测它周围的$2C$ 个词语。CBOW 模型的简单形式如图 2-3 所示,这里只考虑了句子中的一个词语。CBOW 模型本质上就是一个三层的全连接神经网络(包含一个隐藏层)，输入层和输出层都具有$V$ 个神经元结点, 隐藏层具有$N$ 个神经元结点，往往$N$ 要大大小于$V$。则CBOW模型的输入层就是通过one-hot 编码的$V$ 维度向量，这样one-hot 编码的向量的每一维的值便可与输入层每个神经元结点对应起来。更一般的形式如图 2-4 所示，模型的输入就是目标词语的周围语境，即$C$ 个词语。假设输入的one-hot 词向量表示为$x^{(i)}$，模型的输出表示为$y^i$，这个模型的唯一输出$y^i$ 也就是我们的目标词语$y$, 同样是使用one-hot编码的向量。另外，我们需要创造两个矩阵，也是这么模型输入层到隐层和隐层到输出层的权重矩阵，$W^1 \in \mathbb{R}^{N\times|V|}$ 和$W^2 \in \mathbb{R}^{|V|\times N}$ , 这里$N$ 就是隐层神经元数目，也表示最终词向量的纬度空间。

这个语言模型的训练过程可以分解为如下几个步骤，对于第$i$个目标词语：

(1)首先生成$2C$ 大小上下文词语的one-hot向量，记做$(x^{i-C},\cdots,x^{i-1},x^{i+1},\cdots,x^{i+C})$

(2)对于这个上下文，计算其中所有词语的嵌入向量$(u^{i-C} = W^1x^{i-C}, u^{i-C+1} = W^1x^{i-C+1},\cdots,u^{i+C} = W^1x^{i+C})$

(3)计算这些向量的平均向量$h = \frac{u^{i-C}+u^{i-C+1}+\cdots+u^{i+C}}{2C}$

(4)计算输出层的得分向量$z = W^2h$

(5)使用softmax将输出层得分向量变成归一化的概率值$\widehat{y} = softmax(z)$

(6)使用这个输出概率值与真正的目标词语$y$ 比较，通过计算交叉熵$H(\widehat{y^i}, y^i)$ 来定义损失函数，再通过梯度下降一步一步调整权重矩阵，以此来优化这个模型。

而最终训练处的模型的输入层到隐藏层的权重矩阵$W^1$ 才是我们需要的词向量表示，具体来讲，第$i$ 个词语的词向量就是$W^1$ 的第$i$ 列的向量$w^i \in \mathbb{R}^{N\times1}$ 。图 2-2展示了词向量样例，可以看到可以体现词语之间的相似性。当然这只是最基础的词向量构造方法，还有很多版本的词向量构造思路，最近基于字符级别的字符向量可以捕获到更多句法级别的信息，展现了巨大的应用潜力。

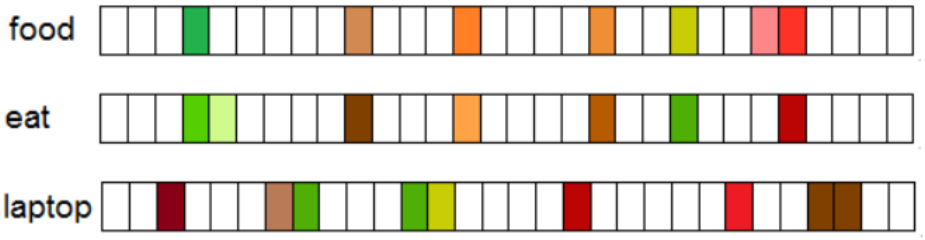



图 2-2 词向量的样例

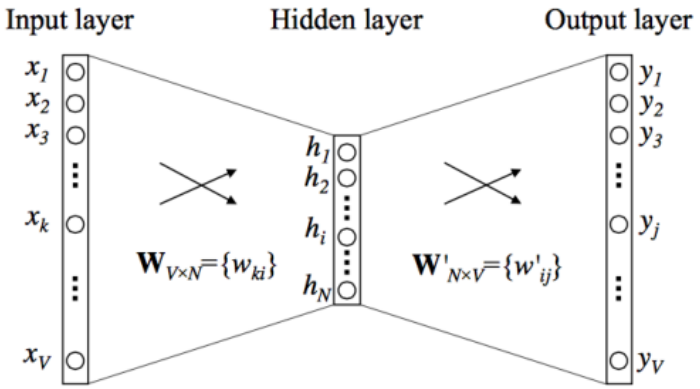



图 2-3 CBOW模型的简单形式

### 2.1.2 各种深度学习模型的使用

理解了词向量的构造原理，就解决了深度学习应用在自然语言处理上最基本的问题——输入的表示。剩下的就是根据具体问题，如何选择模型，如何设计解决方案的问题了。有很多深度学习的模型可供选择，比如深度神经网络DNN, 卷积神经网络CNN，循环神经网络RNN，长短期记忆神经网络LSTM，GRU(LSTM的一种变体)，递归神经网络，基于注意力机制的深度神经网络模型，Encoder-Decoder模型，记忆网络模型等等。

基于深度学习模型的方法已经在很多自然语言理解领域取得了更好的表现，比如, 在文本分类问题上，文章[1][2]提出了基于卷积神经网络的模型，文章[1]使用比较普通的卷积神经网络结构，文章[2]中使用一种多层的交织最大池化层和卷积层的网络结构。文章[3][4] 都使用基于循环神经网络模型，两者的对比体现了LSTM 相比较普通RNN 的优势，并且文章[4] 使用的是一种树形长短期记忆网络(Tree-LSTM)，证明了树形LSTM 相比于双向的LSTM 的优势，这表明Tree-LSTM 能够更好的捕捉到自然语言的句法信息。文章[5] 使用的基于记忆网络模型目前为止取得了最优的表

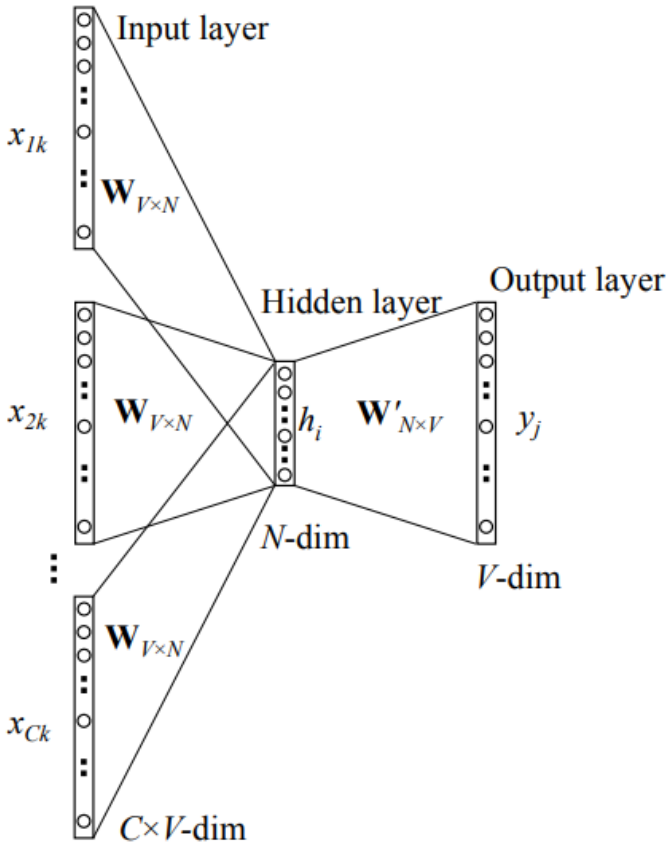



图 2-4 CBOW模型的一般形式

现。

在机器翻译任务上，文章[]使用一个四层的seq2seq 模型，使用LSTM，然后用这个模型对从传统SMT方法产生的1000个候选翻译重新排序。文章[]使用一个带有8个编码器和8 个解码器的深度LSTM 网络组成，并使用了注意机制（attention）和残差连接（residual connections）。最近，[]提出了一种基于CNN的seq2seq模型，输入的每个词语通过CNN由并行方式组成的注意力架构计算出其表示，另外，解码状态需要同时考虑之前已经产生的结果。[]中不在是使用RNN或者CNN来作为encoder-decoder的模型基础，只使用注意力机制处理序列模型相关的问题，比如机器翻译，这样可以高度并行的工作，在提升翻译性能的同时也大大提升了翻译速度。

在信息抽取任务上，文章[]提出的基于卷积神经网络的模型，特殊之处在于作者将位置信息与词向量组合作为模型的输入矩阵，在关系抽取上取得了非常好的表现。文章[]利用共享神经网络底层表示来进行联合学习，同时进行实体识别和实体的关系

抽取。具体来讲，对于输入的句子共享词向量输入层，通过双向LSTM对输入进行编码，然后分别使用LSTM进行命名体识别和CNN来进行关系分类，这种方法可以很好的解决传统的流水线方法的很多弊端。

以上列出的一些自然语言处理的几个领域，使用深度学习已经取得了突破性的进展，在其他一些领域，比如POS标注，语义解析，问答系统等，使用深度学习模型也都是目前最好的途径。当然也有很多工作是使用深度学习结合传统的机器学习算法实现的，深度神经网络具有它天生的优势，会有更多的应用场景。

## 2.2　分割式信息抽取概念以及现有的方法

分割式信息抽取是一个经常被研究的问题，用来分割一段连续文本，然后抽取出隐含在这段文本中的各属性值。比如，对于这样一段连续文本，' Mercedes-Benz E250 Auto, $84888, Obsidian Black, 7speed Sports Automatic. 2 doors 4 seat Coupe, 4 cylinder Petrol TurboIntercooled2.0L;6L/100km'，它有这样几个特点：(1)文本中包含若干属性项，且各个属性值之间是语义相关但不是语法相关的，(2)各属性值不通过固定标点分割，(3)通常属性值顺序不固定，(4)同一语料中不同记录所包含的属性值数目不一定相同，(5)文本中没有噪音数据。这些半结构化文本出现在很多场景下，比如商品信息描述，引文信息，个人信息，地址信息，分类广告，商品评价等等，图[]列出了一些例子。正因为这种文本格式在互联网上广泛的出现，且文本中包含了大量的有价值的信息，研究如何抽取这类文本在学术界和产业界都引起了大量的关注。表[]列举了各个场景下的一些文本样例，可以观察到这类文本的上述一些特征。

对于上面的例子，一个正确的分割结果如图 3-1所示，这个任务主要解决两个问题，一是正确的分割出各属性值，二是给各个分块打上正确的标签。各属性值之间虽然是语义相关的，但是因为其在语法上没有任何关联，所组成的一段文本不是一个常规的自然语言，不符合语法规则，这给这个任务带来了巨大的挑战，我们无法单纯从内容或者语法上来解觉这个问题。解决这个问题的方法大体可以分为三个阶段：(1)使用基于图的概率模型,比如隐马尔可夫模型(HMM)和随机向量场模型(CRF)，(2)使用知识库实现无监督的抽取模型

### 2.2.1 基于图的概率模型

因为分割式信息抽取可以看做是一种序列标注问题，使用机器学习技术，特别是一些基于图的概率模型，比如隐马尔可夫模型和随机向量场模型，展现了很好的效果。当使用机器学习技术时，既有使用人工标注语料的监督式模型也有非监督式模型。文章 [?]首次提出了采用基于图的概率模型来解决信息抽取问题，作者训练独立的隐马尔可夫模型来识别各个属性值，这种方法被扩展成一个DATAMOLD抽取工具 [?]，在这种方法中，各属性的隐马尔可夫模型，也可称作内部隐马尔可夫模型，被包裹起来组成一个外部隐马尔可夫模型。这些外部的隐马尔可夫模型目的是来刻画目标文本的各属性值序列状态。内部隐马尔可夫模型和外部隐马尔科夫模型使用标注数据分别训练，在两个数据集上展现了非常出色的效果。

隐马尔科夫模型是一种生成式序列模型，它设计出一种交叉概率来匹配目标文段和标注序列，但是这种模型不擅长表达多层交叉特征和处理较长依赖的序列。后来，基于随机向量场模型的方法被提出来解决此类问题 [?, ?]。相比较隐马尔科夫模型，随机向量场模型具有更强的推理能力，能够使用复杂的，重复的，非独立的特征来序列和推理，因此它能够更充分地利用上下文信息，并且不同于隐马尔可夫模型，它可以使用其他的外部特征来获取更丰富的信息。 [?]分析，对于不同的输入序列，在隐层状态具有不同的状态转移和发射概率，随机向量场模型更适合用来对这种情况进行建模。尽管这些基于图的概率模型的监督式信息抽取算法表现非常好，但是上面这些方法都是监督式模型，这需要我们去标注大量的训练数据，这个工作的花费往往非常昂贵。还有一些情况，训练数据根本就难以获得。为了解决上述的这些问题，出现了一些使用事先存在的数据集来减轻训练过程的方法 [?, ?, ?, ?]，这些方法使用已经收集的大量的语料，无需使用人力或者只需要一点点人力，就可以完成抽取的任务。这种方法的策略是使用这些已经存在的数据集中的属性值训练出模型，再来在新的输入文本中识别各属性值。 [?]提出的方法基于随机向量场模型来抽取属性值，先从数据集中学习到一些内容相关的特征，再用一些人工标注的数据来学习结构相关的特征，然后结合两者完成抽取过程。 [?, ?]提出的方法能够仅仅依赖数据集来训练模型，然后再使用这个模型在分割的文本中识别属性值，这种方法不需要使用人工标注的数据集，一旦属性值被识别到，整个抽取过程就算完成了。但

是，这个方法基于一个假设：一个数据集中所有文本中的属性值服从同一个序列顺序，这个顺序就是从一批测试样例中学习到的。这两篇文章的不同之处在于，[?]使用隐马尔可夫模型，[?]使用随机向量场。但是，这种策略虽然避免了人工的依赖，但是因为它的应用前提限制在具有固定属性值顺序的语料上，所以在现实中的使用到限制。然后，[?]提出的方法能够解决不同顺序的问题，但是却需要依赖人力的帮助来学习结构相关的特征，增加一些人工的依赖和花费，以便应用在更多的实际场景中，是一种折中的方法。

### 2.2.2　基于知识库实现无监督的抽取模型

不使用上面讨论的基于图的概率模型，也有一些借助知识库实现非监督式模型。[?]提出了一种不一样的思路，首先，作者需要构建一种特殊的文本库，使用简单的空间向量模型计算输入文本和文本库中文本集的相似度，系统可以自动找到对于指定抽取任务最相关的文本集。既然文本集已经确定，系统再借助已经定义好的文本相似度矩阵，例如Jaro-Winkler 和Smith-Waterman，来调整一个阈值抽取出有用的信息。这个思路不同于上述的基于机器学习模型的算法，上述方法需要先学习到一些内容相关的特征再做抽取，这种方法是使用一个事先定义好的文本相似度函数来直接做抽取，但是这种方法需要大量的文本集做支撑。

ONDUX [?]通过一种不同的思路实现了无监督的抽取模型。这种方法依赖的知识库与之前的都有所不同，它存储的内容不再是整条文本，而是将各个属性值分开来存储构建知识库。整个结构大体可以分为三步，第一步，根据词语在知识库中的共现来分割输入文本，将输入文本分割成若干个文本块。第二步，作者定义了一些匹配函数，比如匹配文本类属性值函数，匹配数字类属性值函数，匹配URL和e-mail类属性值函数，使用这些匹配函数直接去匹配第一步中生成的文本块。第三步，在执行抽取的过程中，作者构建了一个位置和序列模型$PSM$，这个模型用来刻画文本的结构特征，然后使用$PSM$模型最后再对标注结果进行补充和修改。这种方法主要的瓶颈在两点，第一，匹配函数的表现力不足，第二，$PSM$模型不能够充分的描述文本的结构特征，它只能单向的提现序列和位置的转移和分布特征，虽然可以提供很大的帮助，但这个模型作为最终的修改参考还不够完美。

## 2.3 本章小结

本章第一部分，首先介绍了深度学习在自然语言理解上的应用，特别是详细地介绍了词向量的生成原理和应用优势，然后列举了一些深度学习模型，并简要介绍了在一些自然语言理解领域使用深度学习模型的方法。第二部分，首先给出了分割式信息抽取任务的概念，然后介绍了两种主要的解决思路，分别列举了一些典型的解决方案，并分析了这些方法的优缺点。

# 第三章　基于卷积神经网络的贪婪式概率标注算法

本章首先给出了基于卷积神经网络的贪婪式概率标注算法(简称为*CNN – IETS*)概述，然后给出了分割式信息抽取的正式定义，并介绍了知识库的结构。接下来详细介绍了算法每一部分的内容，最后给出了详细的实验结果来证明算法的效果和表现。

## 3.1　算法概述

这一章，我们提出了一种非监督式概率标注算法，它基于深度卷积神经网络(CNN)来构建模型，借助提前构造的知识库来克服需要人工标注语料的难题。卷积神经网络是一种端到端的模型，先天具有抽取特征和组合特征的优势，使用深度卷积神经网络能够自动得选择出高质量的特征来更好的刻画各个属性值。而这个知识库有两个作用，第一个作用是用来训练我们的卷积神经网络模型分类器，第二个作用是协助生成文本段，再输入到我们的分类器中。具体一点讲，如图 3-1，输入一个文本,我们首先根据词语在知识库中的共现来初始分隔这条文本，形成若干个文本段。对于这些文本段，如果它在某个属性组中出现的频率比较高而在其他属性组中出现的频率比较低，并且得分高于某个阈值，它就会被标注成这个属性。然后根据这些初始的分割状态和标注结果，哪些没有标注的文本段会被输入到下一步的CNN分类器中。在下一步，我们并不是直接接受CNN分类器的输出结果，我们根据CNN分类器的输出概率发明了一种贪婪式概率标注算法，这个算法会从全局考虑分割和标注状态来为输入文本执行最合理的分割和标注方式。对于一些有上面过程产生的漏标注和错误标注的情形，我们提出了一种双向位置与序列模型，进一步对标注结果进行调整和改善。

为了构建我们的卷积神经网络模型，我们使用了一组全面的语义特征和句法特征加入到模型的输入中，这些特征都是直接从我们的知识库中构造的，这样我们的模型就可以充分利用这些特征来标注每一个独立的文本段。虽然，我们的模型可以直接提供一个标注的结果，但是这些结果仅仅是面对单独的文本段，没有考虑在同一个输入文本中各文本段之间的内在关系，这也不能找到最佳的分割和标注结果。因此，我们需要从全局考虑分割和标注状态来执行概率标注。因此，最理想的标注

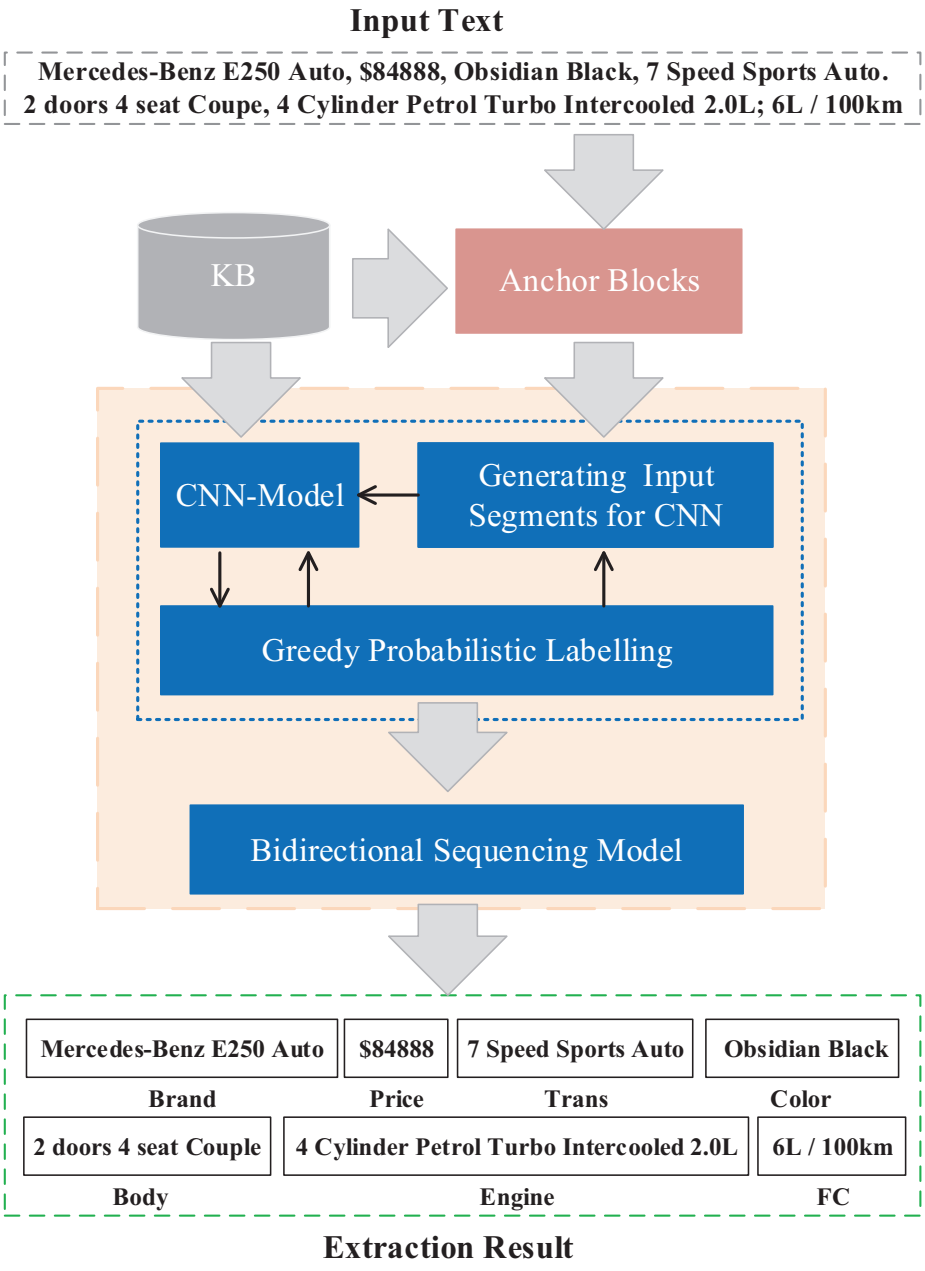



图 3-1 模型的整体结构

方式是，我们依次标注每个文本块来最大化所有标注结果的概率和，但在这个过程中，必须满足一个前提条件，即每个属性值只能包含连续的文本段。我们首先证明了这是一个NP-hard的问题，然后提出了一种贪婪式概率标注算法，此算法能够有效的实现最佳标注方式。

使用上述的基于卷积神经网络的贪婪式概率标注算法，大部分的输入文本都能

够被正确分割和标注了，但是还是会出现一些漏标注和错误标注问题。为了解决此类问题，我们在执行抽取的过程中了权衡考虑了已经标注文本段的位置和序列信息，相比较文章[?]中提出的单向序列模型，我们的是双向模型，从前向和后向同时考虑序列关系，因此能够减少更多的标注错误并提供更大的帮助。

## 3.2　基于卷积神经网络的贪婪式概率标注算法

在这一节，我们首先给出分割式信息抽取任务的定义，然后详细介绍借助知识库我卷积神经网络构建的贪婪式概率标注算法(CNN-IETS)。

**定义1.**

**(分割式信息抽取任务)** 对于某个领域的属性集$\mathbb{A}$, 假设输入文本$I$包含了一些属性集$\mathbb{A}$中的属性值, 分割式信息抽取任务目的是分割$I$成一些列的文本段$S = \{s_1, s_2, ..., s_m\}$, 然后使用一个属性值$A_i \in \mathbb{A}$来标注每一个文本段$s_i$ $(1 \leq i \leq m)$, 这里$s_i$数属性$A_i$中的一个属性值.

知识库: 我们使用实现存在的语料库来构建一个固定领域的知识库(简称为KB)，就像 [**?**, **?**]中做的一样。简单讲，知识库(KB)包含一组特定的属性以及对应的属性值，我们记做KB=$\{< A_1, V_1 >, < A_2, V_2 >, ..., < A_n, V_n >\}$，这里每一个属性$A_i$ $(1 \leq i \leq n)$都是这个领域属性集$\mathbb{A}$中的一个独立属性，$V_i$是一组词语，是在属性$a_i$中出现的典型的、合理的值，并且记录每个属性值在记录中的位置$\{(v_{i,1}, PS_{i,1}), (v_{i,2}, PS_{i,2}), ..., (v_{i,n_i}, PS_{i,n_i})\}$，这里$PS_{i,k}$是一组位置相关的数值，来表示这个属性值$v_{i,k}$出现在不同记录中的位置属性。一个简化的知识库如图 3-2所示，为了方便理解，这里我们只是使用了一些简化的单词而不是使用原始的属性名。

### 3.2.1　基于知识库的初始分割和标注

这一模块的主要目的是识别出输入文本中所谓的锚点块(Anchor Blocks)，就是对一些文本段，我们会根据知识库计算出来一个标注可信度，当这个可信度大于某个阈值时，便可以直接将这个文本段标注成某一个属性。然后，这些识别出来的锚点块将被视作一个基础锚点，执行后面步骤中的分割和标注。

**定义2.**

**(锚点块)** 给定一个输入文本$I$, 如果它满足下面的两个条件，$I$中的一个文本

KB = {<A, $V_A$>, <B, $V_B$>, <C, $V_C$>, <D, $V_D$>, <E, $V_E$>, <F, $V_F$>}

$V_A$ = { ($a_1$, {1, 1, 2}), (**$a_2$**, {1, 2, 2}), ($a_3$, {2, 2}) }
$V_B$ = { (**$b_1b_2$**, {4, 4, 5, 5}), ($a_3$, {4}) }
$V_C$ = { (**$c_1$**, {6, 6, 7}), ($c_2$, {8}), ($c_3$, {6}), ($c_4$, {9}),($e_1$, {5}) }
$V_D$ = { (**$d_1$**, {9, 9, 10}), (**$d_3$**, {10, 11, 11}), ($e_1$, {10}) }
$V_E$ = { ($e_1$, {13, 13}), ($a_1$, {12}) }
$V_F$ = { ($f_1$, {14}), (**$f_2$**, {15, 16}), ($e_1$, {10}) }

图 3-2 一个简化的知识库

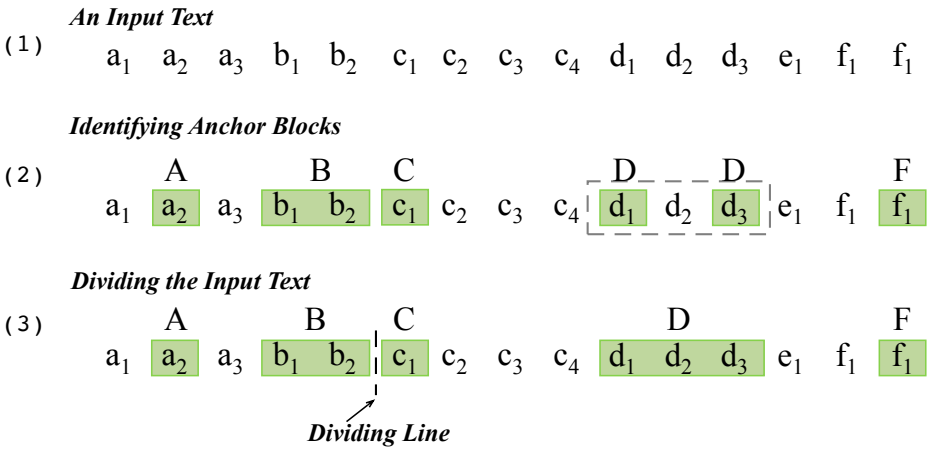



图 3-3 基于知识库的初始分割和标注例子

段$b$ 才可以被标注成具有属性$A$ 的锚点块：*1)*没有任何一个其他属性$A' \in \mathbb{A}$ 满足$p(b, A') >= p(b, A)$，这里$p(b, A)$ 是我们定义的一个方程用来计算一个文本段$b$ 属于属性$A$ 的概率。*2)* $p(b, A) > \theta$，这个阈值$\theta$ 是我们自己定义的，用来提高锚点块的标注质量。

假设一个文本段$b$ 出现在知识库中的$m$ 个不同属性$\{(A_1, A_2, ..., A_m\}$ 的属性值里面，这里$A_i \in \mathbb{A}$ $(1 \leq i \leq m)$表示$b$ 的一个可能的属性。对于每一个$A_i$，假设$b$ 出现在一组不同的属性值中，并且具有不同的出现频率，我们记做$\{(v_{i,1}, f_{i,1}), (v_{i,2}, f_{i,2}), ..., (v_{i,r_i}, f_{i,r_i})\}$，假设所有属性值都是互相独立的，则我们可以通过下面的公式，计算出这个文本段$b$ 属于属性$A_i$ 的概率为：

$$p(b, A_i) = \frac{1 + \sum_{1 \leq j \leq r_i} \sum_{1 \leq x \leq f_{i,j}} \frac{1}{x}}{1 + \sum_{1 \leq k \leq m} \sum_{1 \leq j \leq r_k} \sum_{1 \leq x \leq f_{k,j}} \frac{1}{x}} \tag{3.1}$$

这里，我们在分母使用加1的操作是为了防止分母出现为零的情况。有了这个公式和我们定义的阈值$\theta$，我们便能够在输入文本$I$中标注出一些锚点块了。因为，我们的方法的抽取质量和效率是对阈值$\theta$敏感的，所以我们会通过后面的实验部分找到一个合适的阈值。

注意，一个单独的属性值可能被分割成了两个或更多个文本块，比如图 3-3中，属性$D$便有两个文本块$d_1, d_3$。考虑到在一个输入文本中每个属性可能有一个属性值，所以，我们可以将这两个锚点块和他们之间所有的词语都合并到一起。这里可能出现的一个冲突是，在一个属性的两个锚点块之间出现了另一个属性的锚点块，通常，一个合适的阈值$\theta$便可以避免出现这种冲突。但是，如果这种情况仍然存在，我们可以通过弃掉冲突锚点块中得分更低的那个来解决这个问题。

例子一：给定图 3-2 中的知识库，对于一个图 3-3(1)中输入文本，我们能够识别出下面这些锚点块: $a_2, b_1b_2, c_1, d_1, d_3$ 以及$f_2$，就是图 3-3(2)中绿色方格这些文本段。另外, 既然$d_1$ 和$d_3$ 都是属性$D$ 的锚点块, 我们合并这三个块$d_1d_2d_3$，并共同标注为属性$D$，如图 3-3(3)所示。

### 3.2.2　基于卷积神经网络模型的构建和文本块的标注

为了处理一些未标注的文本段，比如上述例子中第一步产生的$a_1$，$a_3$ 和$c_2$，我们需要执行一个基于卷积神经网络模型分类结果的概率标注。简要来讲，我们首先使用知识库训练一个卷积神经网络分类器(CNN分类器)，假设$|\mathbb{A}|$ 表示给定领域中属性的个数，然后对于每一个输入文本$s$，我们的CNN分类器会生成一个$|\mathbb{A}|$-维的输出向量$P(s) = [\rho_1, \rho_2, ..., \rho_{|\mathbb{A}|}]$，这里每一个$\rho_i$ $(1 \le i \le |\mathbb{A}|)$表示$s$属于属性$A_i \in \mathbb{A}$ 的概率。对于每一个未标注的文本段，我们使用我们的CNN模型来决定它属于哪一个属性，然后综合考虑所有未标注文本段的CNN模型输出的概率结果和那些锚点块，得到整个输入文本的最佳分割和标注结果。下面，我们解释如何构建CNN模型，并且介绍如何找到最佳的标注方式。

#### 1.　构建CNN分类模型

图 3-4给出了我们构建的CNN分类模型的架构图，它的输入时一个文本段，输出一个关于所有可能的属性的概率向量。整体来看，我们的CNN模型包括特征输入层，

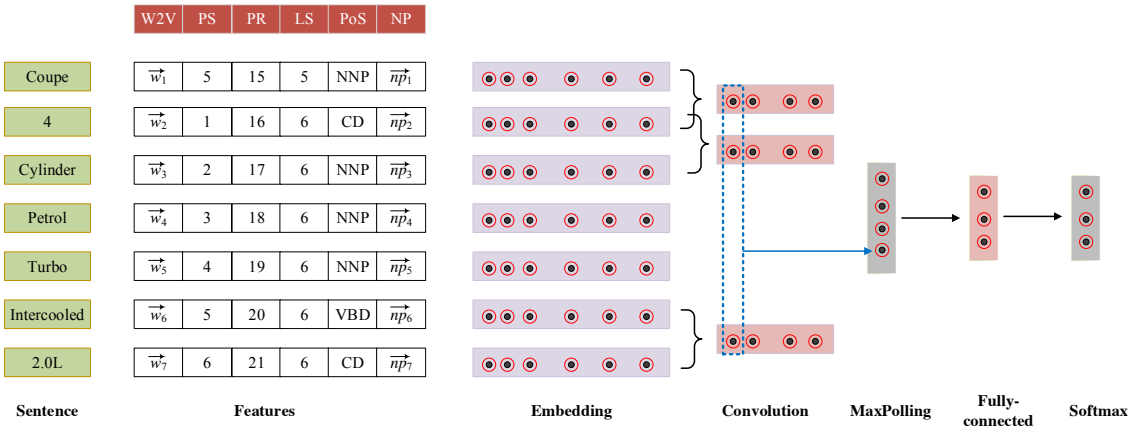



图 3-4 CNN分类模型的架构图

嵌入层，卷基层，最大值池化层，全连接层和最后的Softmax分类器层。特征输入层接受一个输入文本段的特征矩阵，这个特征矩阵包含了丰富的句法和语义特征。嵌入层负责将这些特征值对应到相应的特征向量然后将他们合并在一起。然后我们在特征矩阵上执行卷积操作来提取关于输入文本段的特征，在这一步我们使用了不同大小的过滤核来在输入文本段上尽可能多的覆盖n-gram的文本范围，这样我们就可以生成多层次的，且不同尺寸的特征矩阵。然后，我们再使用最大值池化来对卷积层的结果进行标准化，使其最终都成为相同形状的特征矩阵，且这一步也是进一步抽取特征和降维的操作。池化层得到的特征矩阵再通过一个全连接层，就是一个普通的神经网络，做进一步的特征组合和推理。最终，我们使用Softmax分类器作为我们的输出层，为每一个输入文本段产生分类结果。更多的细节，可以去参考 [?, ?]。

特征:我们为每个输入文本段都构造了一些列丰富的语义特征，包括一些合成的特征。对于文本段中的每一个单词，他的特征矩阵中包括了，提前训练的词向量Word2vec，文本段的位置信息，文本段在输入文本中的位置信息，词性标注信息(POS)，文本段长度，以及一种人为构造的归一化类别概率。下面给出每一个特征的详细介绍：

1) **Word2Vec:** 对于词向量，我们直接使用文章 [?]中用 [?]词向量模型在Pubmed文章上提前训练好的词向量，这个词向量模型是由Tomas Mikolov在Google发明，我们在之前的章节有详细介绍。对于中文实验集，我们在自己的中文语料上使

用Google的word2vec工具[1]训练了一个词向量。

2) **在文本段中位置特征:** 这个位置特征是指一个单词在输入文本段中的位置，我们之所以使用这个特征，是因为我们观察到，在一些特定的属性值中，一些特殊的词语会出现在相同的位置上。

3) **在输入文本中的位置特征:** 这个位置特征是指一个单词在原始输入文本中的位置，我们选择这个特征是因为一个单词在文本中的所在位置往往与它属于哪个属性有密切的关联，这个一个比较容易想到的特征项。

4) **文本段的长度:** 也就是文本段中单词的个数，使用这个特征是因为不同属性的属性值的平均长度往往是有差别的，这说明，文本段的长度是一个非常有区分性的特征。

5) **语法和词性标注特征(POS):** 构造POS特征的过程，对一句话中的每个单词，依赖这个单词的词性定义和上下文来给它标注一个标签，用以说明这个单词在这句话中的角色。这里，我们使用GENIA tagger[2]来得到每个词的POS特征。

6) **归一化类别概率:** 我们同样使用公式 3.1来计算一个概率向量$c\vec{p}$，向量每一维的值表示一个单词属于某个属性的概率。

### 2. 构建CNN分类模型

虽然CNN分类器可以直接输出一个输入文本段属于各个属性的概率值，因为已经存在标注了的锚点块的位置，所以每个输入文本段只会有几个可能的候选属性，因此，对于CNN分类器的结果，有些概率中是用不着的。例如，在图 3-3(3)中，文本段$a_3$ 只可能属于属性$A$，$B$ 和$E$ 中的一个，因为我们已经有了属性为$C$，$D$ 和$F$ 的锚点块，且这些锚点块都在文本段$a_3$的非邻接位置。我们在图 3-5列出了其他文本段的所有候选属性列表，其中已经得到的锚点块只有一个候选属性。

对一个文本段$s$，另$\mathbf{A}_c(s)$ 表示一组候选属性，$P(s) = [\rho_1, \rho_2, ..., \rho_{|\mathbb{A}|}]$ 表示$s$的CNN模型的输出结果，则文本段$s$ 属于一个候选属性$A \in \mathbf{A}_c(s)$ 的概率用下面

[1]https://code.google.com/archive/p/word2vec/
[2]http://www.nactem.ac.uk/GENIA/tagger/

的公式计算得到：

$$p(s,A)=\frac{\rho_{i(A)}}{\sum_{A'\in\mathbf{A}_c(s)}\rho_{i(A')}} \tag{3.2}$$

这里$i(A)$表示属性$A$在属性列$\mathbb{A}$中的下标。对于每一个输入文本段，我们都可以使用公式 3.2计算得到它属于每个候选属性的概率值。

例子二：图 3-5中列出了例子一基于CNN模型得到的各未标注文本段的属于各候选属性的概率值。对于文本段$a_3$，假设它原始的CNN模型输出的概率向量是[0.567, 0.067, 0.199, 0.133, 0.033]，因为$a_3$只可能属于属性$A$，$B$或者$E$。因此，我们能计算出文本段$a_3$属于$A$的概率为：$p(a_3,A)=\frac{0.567}{0.567+0.067+0.033}=0.85$，相同的，我们可以计算出属于$B$和$E$的概率$p(a_3,B)=0.1$和$p(a_3,E)=0.05$。

| Segments | Candidate Attrs | CNN-based Probabilities |
|---|---|---|
| $a_1$ | A, E | A(0.90), E(0.10) |
| $a_2$ | A | - |
| $a_3$ | A, B, E | A(0.85), B(0.10), E(0.05) |
| $b_1$ $b_2$ | B | - |
| $c_1$ | C | - |
| $c_2$ | C, D, E | C(0.50), D(0.50), E(0.00) |
| $c_3$ | C, D, E | C(0.80), D(0.15), E(0.05) |
| $c_4$ | C, D, E | C(0.35), D(0.00), E(0.65) |
| $d_1$ $d_2$ $d_3$ | D | - |
| $e_1$ | D, E, F | D(0.25), E(0.50), F(0.25) |
| $f_1$ | D, E, F | D(0.30), E(0.30), F(0.40) |
| $f_2$ | F | - |

图 3-5 例子中文本段的候选属性值和基于CNN模型的变换概率值

### 3.2.3 输入文本的贪婪式概率标注

尽管我们的CNN模型能够直接给出输入文本段属于各属性的概率，但是这种标注方式只是单独的考虑了的这一个文本段，没有综合考虑这个文本段与输入文本中

其他文本段的关系，因此这种方式可能不能够找到最佳的分割和标注方式。因此，我们在执行每一个概率标注的时候需要整体考虑整个文本的分割和标注状态，事实上，我们期待去找了一种最佳的标注方式来最大化整体的标注概率值得分，而这个得分就是输入文本中所有文本段属于对应属性的概率之和，但是这个过程需要遵循一个条件，这个输入文本中每个属性值只能由位置连续的文本段组成，我们把这个问题定义为基于*CNN*的受限分割式信息抽取(Constrained CNN-IETS Problem)：

**定义3.**

*(***Constrained CNN-IETS Problem***)* 给定某个领域属性集$\mathbb{A}$ 以及一个输入文本$I$ 来做分割式信息抽取任务,，假设$S(I) = \{s_1, s_2, ..., s_m\}$ 表示从$I$ 中分割出来的文本段，其中$s_1 + s_2 + ...s_m = I$, 并且$s_i \cap s_j = \emptyset\ (i \neq j)$，每一个文本段$s_i \in S$ 都有一组候选属性标签$\mathbf{A}_c(s)$，则基于CNN的受限的分割式信息抽取任务的目的是最大化下面这个方程式：

$$Prob(I, \mathbb{A}) = \sum_{s \in S(I)} p(s, A_s) \tag{3.3}$$

, 这里$A_s \in \mathbf{A}_c(s)$ 是$s$ 的一个可选属性标签，所谓受限是指，对于任何不相邻的文本段$s_a \in S(I)$ 和$s_b \in S(I)$，他们的可选属性标签$A_{s_a} \neq A_{s_b}$ ，除非他们之间的所有文本段都具有相同的可选属性标签。

但是，这中基于CNN模型结果的受限的分割式信息抽取问题(Constrained CNN-IETS Problem)是一个NP-hard问题，我们给出下面的定义：

**定理1.**

基于*CNN*的受限的分割式信息抽取问题是一个*NP-hard*问题。

**证明.** 首先，我们假设输入文本中一个没有任何锚点块的属性称作自由属性(Free-Attribute)，如果在这个输入文本中存在属性值，则它可能出现在这段文本中任何没有标记的位置上。假设，这个输入文本中有$n$ 个自由属性，我们可以将我们的基于CNN的受限的分割式信息抽取问题约简地看做是旅行商问题(Travelling SalesMan problem，TSP)。另$V$ 表示一组城市, $A = \{(r, s) : r, s \in V\}$ 表示城市之间的路径，

$C(r,s) = C(s,r)$ 表示城市$r$ 和城市$s$ 之间的路程费用，旅行商问题(TSP)是最基本的路线规划问题，就是找到最小的路程总费用而访问完所有城市。

我们可以将我们的问题简化成为旅行商问题(TSP)：将我们的问题中的$n$ 个自由属性看做旅行商问题中的那一组城市$V$ ，将公式 3.3中的$p(s,A_s)$ 看做是与边$(A_c,A_s)$ 相关的路费计算方式，当且仅当$A_c$ 是第$i$ 次出发的城市。然后，我们的问题就是找到最大的路程总费用而访问完所有城市。因此，我们可以发现，基于CNN的受限的分割式信息抽取问题跟旅行商问题一样，在旅行商问题中需要去全排列列出来所有可能的路线，然后找到最小化花费的旅行路线，我们的是要全排列列出所有可能的文本段组合方式，找到最大概率值得标注方式。于是，理论 1得证。

□

但是，列出所有可能的文本段组合方式是一个很不合实际的方式，因为，(1)因为未标注的文本段个数很多，用全排列会产生大量的组合结果，很多组合结果完全没有考虑锚点块的位置，是完全没有意义的，这增加了整体标注错误的几率，(2)得到如此多组合结果，处理起来会花费大量的时间，这种方式效率上会大受影响。为了解决这些问题，我们提出了一种贪婪式概率标注算法，这个算法基于每个文本段属于所有候选属性的概率值，然后寻找到一个对于当前输入文本的尽可能接近最佳的标注结果。在算法 1中详细介绍了这种贪婪式的标注算法：对于一个输入文本$I$ ，另$S(I) = \{s_1, s_2, ..., s_m\}$ 表示从文本$I$ 中分割得到的所有文本段，最开始，一部分这些文本段被定义为锚点块，对于剩下的未标注的文本段，我们以一种贪婪的模式来标注他们。首先，我们在输入文本中寻找有没有存在相邻的锚点块，比如图 3-3(2)中的点块$b_1b_2$ 和点块$c_1$ ,如果找到了，我们便可以将输入文本分割成两部分，比如像图 3-3(3)中的这条分割线，这样我们就可以分开来执行标注算法，这可以提高标注效率。然后，对于每一个分区，基于CNN 模型的输出概率向量，使用公式 3.2，我们便可以计算出来每个文本块属于它的所有候选属性的概率值。在这些所有未标注的文本块中，每一次我们只贪婪地选择一个出来进行标注，记做$s_c$ ，这个选出来的文本段的概率应该是剩下这一分区中未标注文本段中概率值最高的那一个。然后，我们根据概率值为文本段$s_c$ 标注上对应属性，记做$A_c$，然后，我们立即检测是不是存在一个非邻接的文本段$s'$ ，且这个文本段$s'$ 也已经具有属性$A_c$ ，如果出现这种情形，

我们需要检查$s'$和$s_c$之间是不是已经存在标注了其他属性的文本段，如果，这种情况出现了，我们便取消这次对$s_c$的标注，然后重新选择具有第二高概率值的文本段来进行相同的标注。如果这种情形没有发生，我们需要将$s'$和$s_c$之间所有的文本段合并形成一个新的文本段，且标注为$A_c$。我们重复执行这个过程，直到输入文本$I$中没有可以被标注的文本段为止。注意，在这个过程中，我们定义了一个阈值$\lambda$来防止产生低质量的标注，我们会在实验部分检测这个阈值的作用，并寻找到它的最佳值。

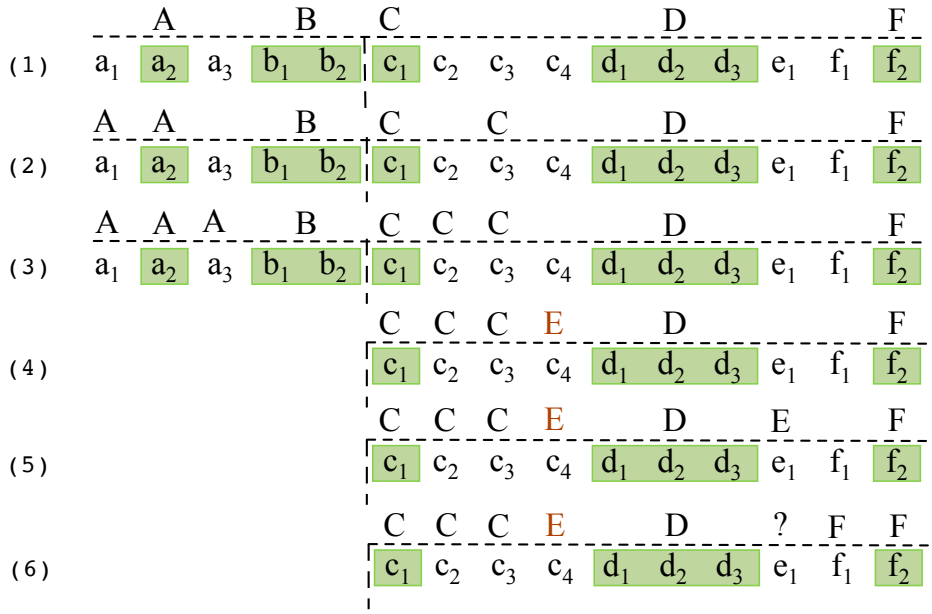



**图 3-6 贪婪式概率标注例子**

例子三：对于例子一种的输入文本，图 3-5中给出了每个未标注文本段对于对应的候选属性的基于CNN模型的变换概率值，我们设定阈值$\lambda = 0.3$，我们执行文本标注的算法 1如下：

1) 首先，我们像图 3-3(3)中一样将输入文本分成两个部分，然后独立的在每个分区上执行我们的标注算法。如图 3-6所示。

2) 在第一个分区，我们首先标注$a_1$为属性$A$的一个文本块，因为，$s_1$在这个分区中的所有文本块中具有最高的概率值。然后，我们继续标注$a_1$为属性$A$的一个文本块，此时，我们完成了第一个分区的标注任务。

3) 在第二个分区有5个未标注的文本段，其中文本段$c_3$ 属于属性$C$ 具有最高的概率值，将被第一个标注。然后，我们通过标注$c_2$ 为属性$C$ 来合并$c_1$ 和$c_3$。至此，我们还有三个未标注的文本段，其中，$c_4$ 具有最高的概率值首先被标注为$E$，然后，$e_1$ 应该是一下个被标注为$E$ 的，但是，因为它与$c_4$ 冲突了，所以我们撤销了对$e_1$ 的标注，然后标注文本块$f_1$。直到最后$e_1$ 任然不会被标注，因为它的概率值小于我们的阈值($\lambda = 0.3$)。

使用我们的基于CNN的贪婪式概率标注算法，大部分输入文本都可以被正确的分割和标注了，只有一些特殊的情形中出现了错误标注（mis-matching）和漏标注（un-matching），比如像例子三中的$c_4$，$e_1$。为了处理这种错误，在下一节中，我们从测试数据集中实时地学习到已经标注文本段的位置和序列信息，使用一个模型来权衡这些位置和序列信息来修正目前的标注结果。

### 3.2.4 基于序列和位置的结果修正

我们提出了一种双向的位置和序列模型来捕捉输入文本中已经标注了的文本段的位置和序列信息。对于一个输入文本$I$，假设有一组文本段$S(I) = \{s_1, s_2, ..., s_m\}$，并且他们对应的标签为$\mathbf{A}(I) = \{A_{l1}, A_{l2}, ...A_{lm}\}$。对于一个文本段$s_i$，他的两个相邻的文本段为$s_{i-1}$ 和$s_{i+1}$，这三个文本段分别被标注为属性$A_{li}, A_{l(i-1)}$ and $A_{l(i+1)}$。序列模型考虑：（1）从$A_{l(i-1)}$ 转移到$A_{li}$ 的概率以及从$A_{li}$ 转移到$A_{l(i-1)}$ 的概率。（2）属性$A_{li}$ 出现在某个位置的概率。

因为经过我们的基于CNN的贪婪式概率标注算法得到的标注结果已经具有非常高的准确率（后面的实验中会证明这一点），所以，这个结果可以用来学习序列相关的特征和输入文本中属性值位置的信息。有一点很重要，这些特征的学习是从每一组输入文本中实时的（on-demand）学习到的，不需要人工的训练，也没有属性值遵循某一个特定的顺序这个假设，然后我们构建一种类似隐马尔科夫模型的图模型来强化此前得到的标注结果，我们称之为BPSM（Bidirectional Positioning and Sequencing Model）。相比较 [**?**]中提出的单相位置和序列模型(PSM)，我们的模型从双向考虑序列之间的关联，这可以减少单向标注中出错的风险。

*双向位置与序列模型(BPSM):* 我们的模型考虑：（1）对于一组状态$L = \{begin, l_1, l_2,$

$..., l_n, end\}$，这里状态$l_i$表示在我们的基于CNN的概率标注算法中每一个文本段被分配到的标签，*begin* 和*end* 表示两个虚拟的状态，仅仅表示一个输入文本的开头和结尾，是我们人为添加上的。（2）矩阵$FT$ 中存储了在标注结果中从状态$l_i$ 转移到状态$l_j$ 的概率，这是一种前向转移，表示从状态*begin* 到状态*end*。（3）矩阵$BT$ 中存储了在标注结果中从状态$l_j$ 转移到状态$l_i$的概率，是一种后向转移，表示从状态*end* 到状态*begin*。（4）矩阵$P$ 存储了在标注结果中输入文本中的一个标签为$l_i$ 的文本段出现在*pos* 位置的概率。

我们构建矩阵$FT$，需要考虑在基于CNN的标注结果中从状态$l_i$ 到状态$l_j$ 的数量和从状态$l_i$ 出发的前向的所有数量的比例。所以，概率矩阵$FT$ 中每一个概率值$ft_{i,j}$ 的定义方式如下：

$$ft_{i,j} = \frac{\#\ of\ transitions\ from\ l_i\ to\ l_j}{Total\ \#\ of\ transitions\ out\ of\ l_i} \tag{3.4}$$

这里只是计算前向的转移，同理，我们从只考虑后向转移来构建概率矩阵$BT$，$BT$ 中的每个概率值$bt_{k,j}$ 的定义方式如下：

$$bt_{k,j} = \frac{\#\ of\ transitions\ from\ l_k\ to\ l_j}{Total\ \#\ of\ transitions\ out\ of\ l_k} \tag{3.5}$$

这里只从后向来计算转移概率。

我们使用从CNN标注的结果中观察到的状态$l_j$出现在*pos* 位置的次数比上出现在位置*pos* 所有状态的总数目来构建矩阵$PS$。因此，矩阵$PS$ 中每个概率值的定义方式如下：

$$ps_{j,pos} = \frac{\#\ of\ observations\ of\ l_j\ in\ pos}{Total\ \#\ of\ segments\ in\ pos} \tag{3.6}$$

根据CNN标注模型的标注结果，我们使用概率矩阵$TF$，$BT$，$P$ 来最大化输入文本中属性值出现的序列和位置概率和，这种方式遵循了最大化概率方式（Maximum Likelihood approach），通常被用来训练图相关的模型 [**?**, **?**]。在实际操作中，构建矩阵$TF$，$BT$，$PS$ 需要在CNN标注模型的输出结果上执行单趟的观察。这里注意，漏标注的文本段（mis-matching）在构建矩阵的时候是被自动忽略的，虽然构建矩阵时也可以加上这些漏标注文本段的数量，但这可能会刻画出谬误的转移。此外，在我们的实验中发现，CNN标注模块中产生的漏标注文本段的数量本身是非

常少的，因此我们不必考虑这些漏标注的文本段，它们也不会对整体的修正起到很大的影响。

图 3-7 和图 3-8 展示了我们使用CAR数据集构建的BPSM模型，我们可以发现，这个模型不仅可以表示分配给文本段的属性序列相关的信息，也可以表示输入文本中位置相关的信息。例如，在这个数据集中，输入文本开头的属性值，相比为‘Color’的属性值，更倾向于是为‘Brand’的属性值，还有，为‘Engine’的属性值出现在为‘Price’的属性值后面的概率会更大。这些信息都是对输入文本结构相关信息的描述。当构建好了BPSM模型，我们便可以用这些估计出来的概率值来优化标注结果。

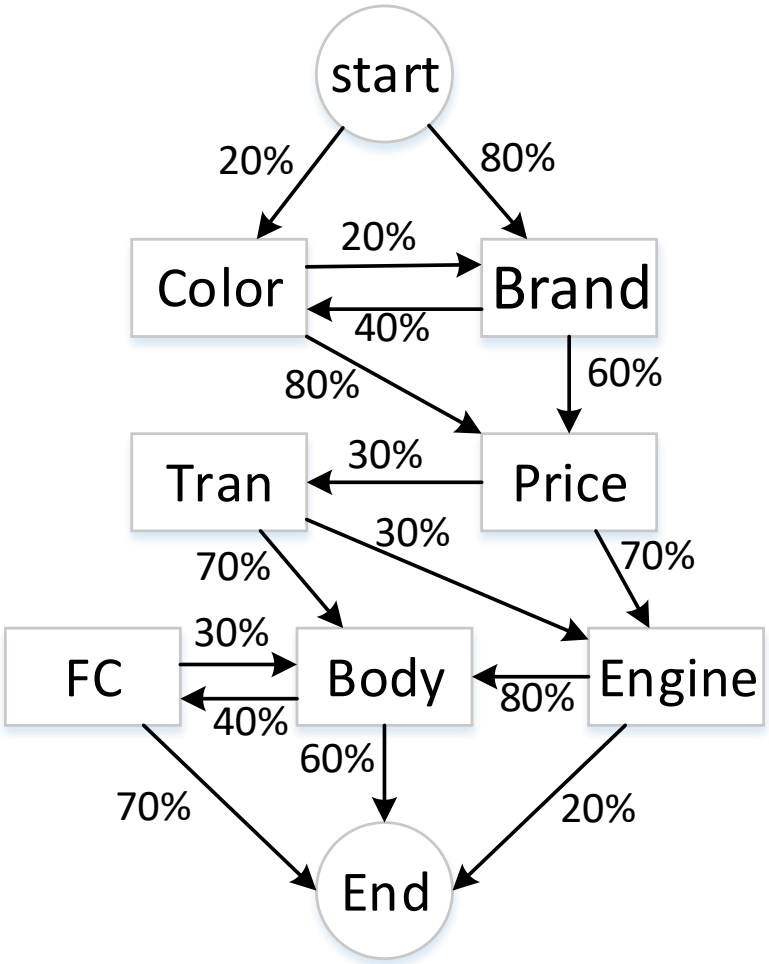



图 3-7 CAR数据集上产生的BPSM例子-前向和位置

使用*BPSM*来做修正*:* 我们在执行最后修正的步骤时，同时考虑三个因素，序列概率，

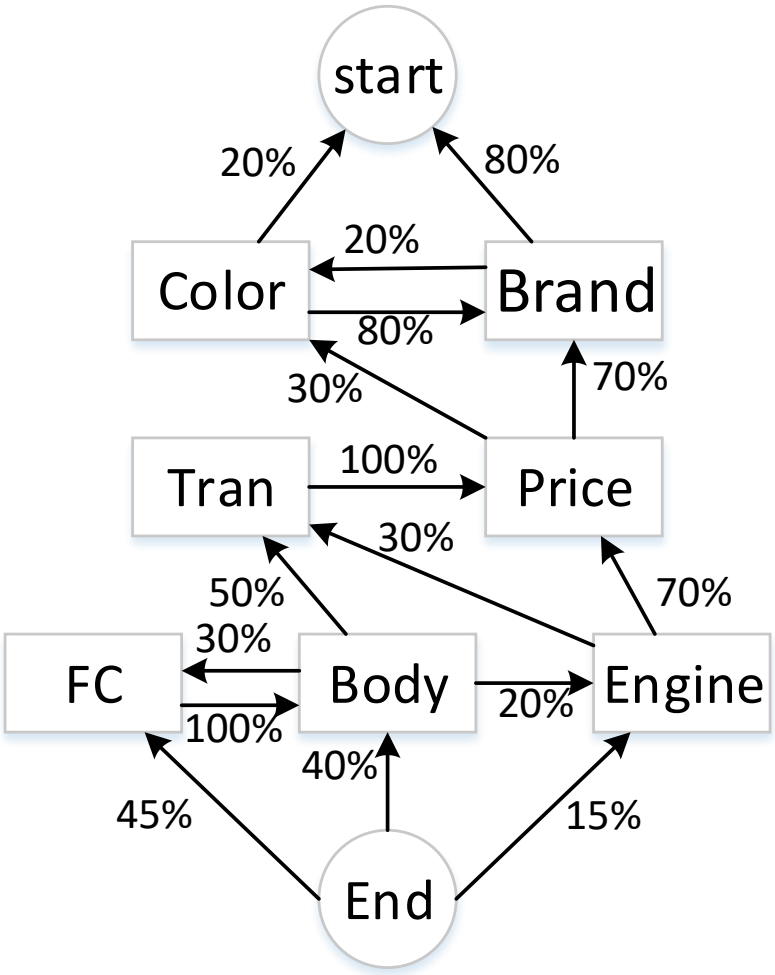



图 3-8 CAR数据集上产生的BPSM例子-后向和位置

位置概率和CNN标注模型的结果，是非常重要的。考虑到这三个因素是相互独立的，文章 [?]中讨论了这个问题，一种Noisy-OR-Gate模型 [?] 非常适合来整合这三个因素来决定每个文本段最终的标注方案。因此，对于文本段$s$ 属于属性$A_j$ 的概率分别从前向考虑和后向考虑为：

$$P_{fw}(s, A_j) = 1 - ((1 - p(s, A_j))(1 - ft_{i,j})(1 - ps_{j,pos(s)})) \tag{3.7}$$

$$P_{bw}(s, A_j) = 1 - ((1 - p(s, A_j))(1 - bt_{k,j})(1 - ps_{j,pos(s)})) \tag{3.8}$$

这里，$s$ 表示在输入文本中位置为$pos$ 的文本段，它前面是是一个被标注为$A_j$ 属性的文本段，后面是一个被标注为$A_k$ 属性的文本段。概率项$ft_{i,j}$, $bt_{k,j}$ 和$ps_{j,pos(s)}$ 分别来自矩阵$TF$, $TB$ and $PS$。

对于输入文本$I$的一组文本段$S(I) = \{s_1, s_2, ..., s_m\}$，在一次迭代中，我们需要比较输入文本中最前面的一个未标注文本段的前向概率，记作$P_{fw}(s_1, A_{l1})$，和最后一个未标注文本段的后向概率，记作$P_{bw}(s_m, A_{lm})$。如果$P_{fw}(s_1, A_{l1}) \geq P_{bw}(s_m, A_{lm})$，我们首先标注$s_1$，然后继续比较$P_{fw}(s_2, A_{l2})$和$P_{bw}(s_m, A_{lm})$。否则，我们首先标注$s_m$，然后继续比较$P_{fw}(s_1, A_{l1})$和$P_{bw}(s_{m-1}, A_{l,m-1})$来决定哪一个是下一个被标注的文本段。我们希望通过这种思路，同时从前向和后向考虑，找到最佳的标注结果。我们循环执行这行这个过程，直到一条输入文本中所有的文本段都被标注了为止。

## 3.3 实验结果与分析

在这一节，我们会介绍本文提出的模型在在三个数据集上的实验结果，并做充分的分析和比较。

### 3.3.1 数据集和评价标准

我们在下面这三个真实数据集上做了实验：

1) *引文数据集:* 我们从超过500个计算机领域的研究员的主页上手机了超过2.5万条的论文引文信息，这些引文信息中隐含的属性值属于*Title, Author, Journal, Volume, Pages,* and *Date*这些属性，文章 [**?**, **?**]中也使用了相同的数据集。另外，我们随机的从DBLP中选择了若干引文记录来构建我们的知识库(KB)。

2) *二手房:* 我们从Anjuke [3], Ganji [4] and Lianjia [5]上收集了超过10万的关于二手房出售和租赁相关信息的描述文本，每一条文本中包含的值属于属性：*Name, Publish-Time, Price, Charge-Method, Neighborhood, Unit, Floor,* and *Furnitures*。我们的知识库(KB)的构建使用相同来源的结构化的表格构建的。

3) *二手车:* 二手车的数据集，我们从Carsales [6]网站中收集了超过20万条的关于二手车售卖的描述信息，这个数据集的属性集包括：*Brand, Price, Transmission, Color, Body, Engine, Fuel Consuming*，我们的构建知识库(KB)的数据同样可以从这个网站上的结构化表格区域得到。

[3]www.anjuke.com
[4]www.ganji.com
[5]www.lianjia.com
[6]www.carsales.com

**评价标注：**我们使用准确率(**Precision**)，召回率(**Recall**)和F值(**F-Measure**)来评测我们提出的模型。对于一个属性$A$，另$N_l(A)$表示使用某个方法标注文本段的总数量，$N_c(A)$表示对于这个属性正确标注的文本段数量，$N_g(A)$表示测试集中应该被标注为属性$A$的文本段数量。然后，在属性$A$上的准确率可以表示成：$prec = \frac{N_c(A)}{N_l(A)}$，在属性$A$上的召回率可以表示成：$recall = \frac{N_c(A)}{N_g(A)}$，在属性$A$上的F值可以表示成：$F_1 = \frac{2(prec \times recall)}{prec+recall}$。

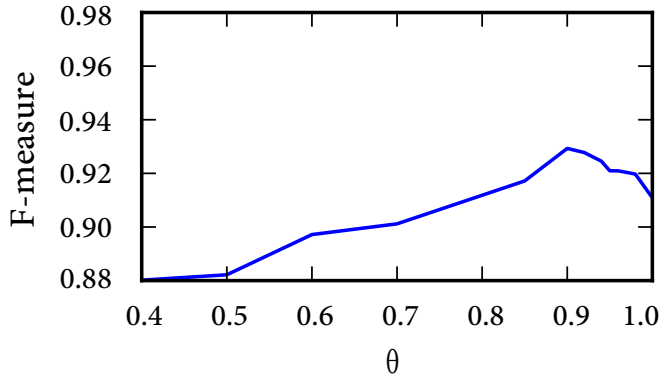
### 3.3.2　参数的设定

在我们的方法中有两个重要的参数：一个是在章节 3.2.1中用来定义锚点快的阈值$\theta$，另外一个是章节 3.2.3中算法一中用来防止低质量标注的阈值$\lambda$。在我们的实验中，我们采用单一变量分析的方法，先设置阈值$\lambda = 0.6$，然后在三个数据集上测试我们的方法中阈值$\theta$对抽取质量和效率的影响。然后，设置阈值$\theta = 0.9$，来测试阈值$\lambda$对抽取质量的影响。注意，在这两组实验中，我们的知识库(KB)在测试集中的属性值覆盖率均不超过30%。

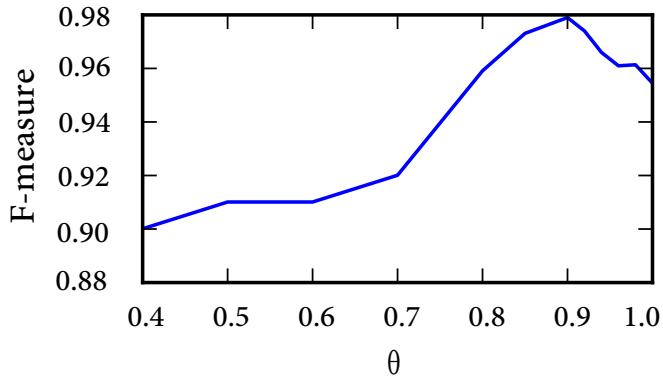
在我们的方法中，图 3-9中表明了阈值$\theta$对抽取质量和效率的影响。详细来讲，从图 3-9(a)中可以看出，当$\theta$从0.4 增加到0.9，在引文数据集中，F值首先从0.880平滑的增加到0.929，在这点之后，当$\theta$从0.9 增加到1.0 ，F值从0.929 缓慢的下降到0.910 附近。我们在图 3-9(b)和图 3-9(c)中的二手房和二手车数据集中可以看到相同的趋势。基于上面的实验结果，我们得出结论，在所有这三个数据集中，来定义锚点快时，设置阈值$\theta = 0.9$是一个合适的选择，这个值也将在后面的实验部分中来使用。在另一方面，当考虑抽取效率时，我们从图 3-9(d)中可以发现，在引文数据集中，当$\theta$从0.4 增加到1.0 时，抽取每条文本所花费的时间从大概0.05秒缓慢增加到0.31 秒。在图 3-9(e)和图 3-9(f)中的二手房和二手车数据集中可以看到相同的趋势。因此，虽然$\theta = 0.9$可能不能实现最佳的抽取效率，但是牺牲一点点效率来换取更佳的抽取质量是可以接受的。

图 3-10中表明了阈值$\lambda$对抽取质量的影响。从图 3-10(a)中可以发现，当阈值$\lambda$的值从0.2 增加到0.6 ，引文数据集上的抽取质量首先从0.817 增长到0.929 ，在$\lambda = 0.6$时达到最大值。当阈值$\lambda$的值从0.6 增加到1.0 ，我们方法的抽取质量从0.929 缓慢的下降到0.808 。在另外两个数据集中，如图 3-10(b) 和图 3-10(c)所示，可以看到

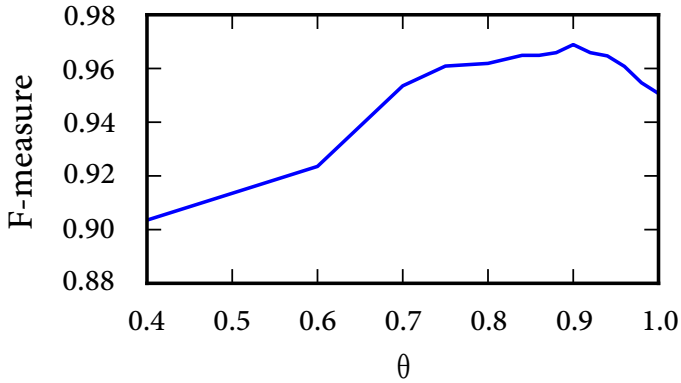
相同的现象。因此，我们可以得出结论，算法一中在过滤一些糟糕的标注的环节，$\lambda = 0.6$ 或许是一个合适的值，我们也将在下面的实验中使用这个值。




(a) F-Measure on PersonalBib



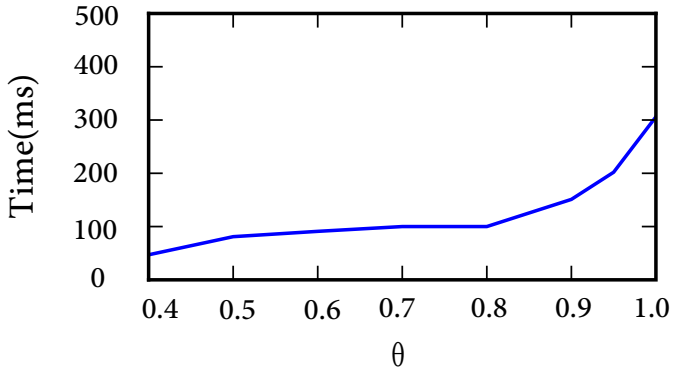

(b) F-Measure on House



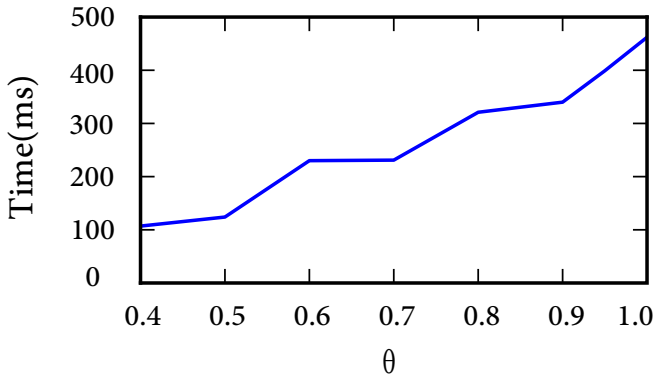

(c) F-Measure on Car

### 3.3.3 与其他方法的比较

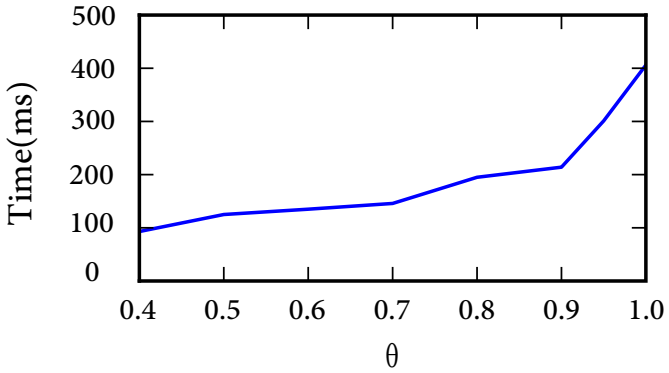
我们比较了我们的方法(CNN-IETS)的抽取质量与另外两种目前表现最佳的方法的抽取质量，为了公平的比较，对于每个数据集上的每个方法，我们都将他们调整




(d) Time Cost on PersonalBib




(e) Time Cost on House

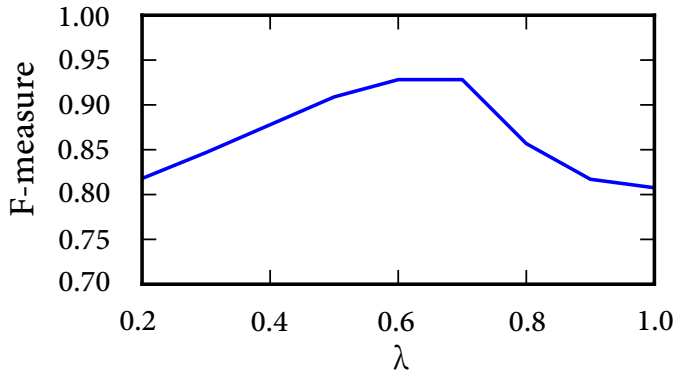



(f) Time Cost on Car

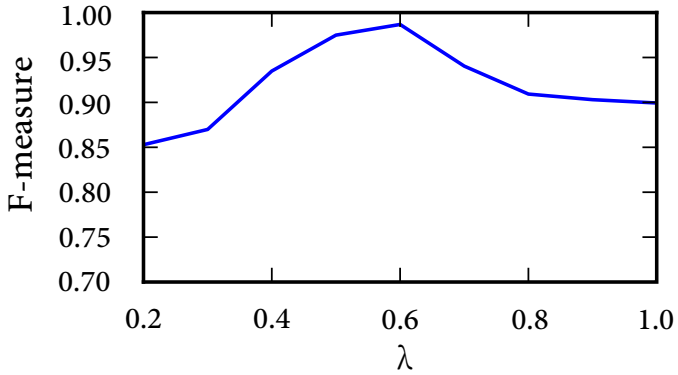
图 3-9 Evaluating the Effect of Threshold $\theta$ to the Extraction Quality and Efficiency ($\lambda = 0.6$)

成具有最佳的抽取表现。

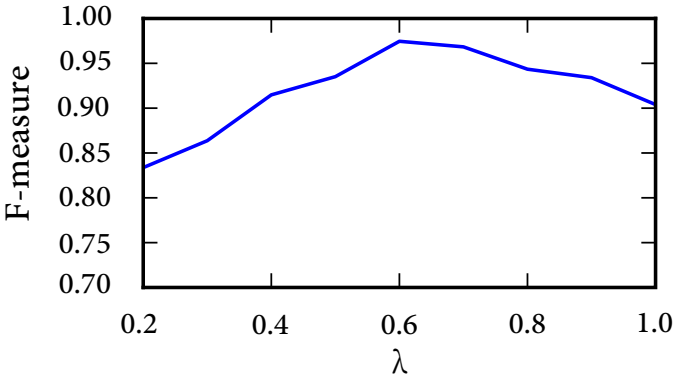
1) 基于CRF的监督式模型(**CRF**): 这是目前表现最佳的一种基于隐马尔科夫模型(CRF) 的监督式信息抽取方法 [**?**]，我们构建CRF模型所使用的特征包括文本自身特征，模型布局特征，外部词典源，文章 [**?**]中有信息的说明。这些特征也被看




(a) F-Measure on PersonalBib




(b) F-Measure on House




(c) F-Measure on Car

图 3-10 Evaluating the Effect of Threshold $\lambda$ to the Extraction Quality ($\theta$ = 0.9)

做是一种标准的特征。

2) 非监督式ONDUX方法(**ONDUX**): ONDUX [**?**] 是目前分割式信息抽取任务中采用非监督式方法表现最佳的一种方法。这种方法首先将根据一个知识库，将输入文本粗略的分割成若干个文本块，每一个文本块是可能组成一个属性值的一组词语，然后通过一组特定的匹配函数，将这些文本块与已知的属性进行匹配。为了比较

的公平，这种ONDUX方法与我们的方法(CNN-IETS)使用同一个知识库(KB)。

**抽取质量的比较**: 我们在三个数据集中比较了这三种方法的抽取质量，为了全面的比较，我们不仅比较各个属性的F值得分，也比较了在每个数据集上所有属性的平均F值得分。我们从表 3-1，表 3-2 和表 3-3中可以看到，我们的方法在三个数据集的几乎所有属性上的表现都好于另外两种方法。整体来讲，我们的方法(CNN-IETS)相比较另外两种目前表现最好的方法(CRF，ONDUX)，在三个数据集上，抽取质量平均提升了超过10%。

**抽取效率的比较**: 我们另外比较了使用这三种方法来抽取一条文本所花费的平均时间，从表 3-4中可以看到，相比较ONDUX和CNN-IETS，基于CRF的方法花费了更多的时间来训练自己的模型。尽管，我们的方法(CNN-IETS)也需要训练CNN模型，但是这个训练的过程可以使用已经存在的知识库在线下来运行，因此我们的方法跟ONDEX方法在做线上的信息抽取时所花费的时间差不多。

**知识库(KB)的覆盖率对抽取质量的比较**: 因为ONDUX方法和我们的方法都需要使用一个事先构建好的知识库用来理解输入文本的结构特征，因此，知识库在测试集中属性值的覆盖率对抽取质量有非常大的影响。经过试验，我们得出结论，相比较ONDEX的方法，我们的CNN-IETS方法对知识库覆盖率的要求更低。从图 **??**(a)中可以看到，当知识库的覆盖率从0.1 增加到0.6 ，在引文数据集上，CNN-IETS方法的抽取质量从0.874 缓慢地增加到0.929 ，而ONDEX方法的抽取质量急剧的从0.704 增加到0.810 。另外，不管知识库的覆盖率是多少，我们的CNN-IETS方法的表现都要好于ONDEX方法。从图 **??**(b)和图 **??**(c)中我们可以发现相同的现象，这就证明了，我们的方法相比较ONDEX方法对知识库覆盖率的依赖更低。

### 3.3.4　BPSM效果的验证

我们提出了一个全新的类似隐马尔科夫的概率图模型，双向位置与序列模型(BPSM)，用来进一步修正CNN模型得到的标注结果，相比较在ONDEX方法 [**?**]中使用的单向位置和序列模型(PSM)，我们的模型同时从前向和后向来考虑序列关系，因此可以有效的避免使用单向模型可能出现标注错误的风险。

为了验证BPSM的有效性，已经相比较PSM的优势，我们在三个数据集上做了实

**算法 1:** 基于CNN的贪婪式概率标注算法，CNN-based Greedy Probabilistic Labeling

**Input** : Input text $I$, segments $S(I) = \{s_1, s_2, ..., s_m\}$, $P = \{p(s_1, A_{c1}), p(s_1, A_{c2}), ...\}$, where $p(s_i, A_{cj})$ denotes the probability that an segment $s_i \in S(I)$ belonging to its candidate attribute $A_{cj}$ according to Equation 3.2, and a minimum labelling probability threshold $\lambda$

**Output**: Label set $\mathbf{A}(I) = \{A_{l1}, A_{l2}, ...A_{lm}\}$ for segments in $S(I)$

Initialize $\mathbf{A}(I)$ by setting each $A_{li} = -1$, where $A_{li} \in \mathbf{A}(I)$;
**Pt** $= \{Pt_1, Pt_2, ...\} \leftarrow$ Partitioning $S(I)$ into adjacent subsets;
**foreach** $Pt \in$ **Pt do**
    **do**
        $(s_c, A_c) \leftarrow$ the pair with $\max_{s \in Pt, A \in \mathbf{A}_c(s)} p(s, A)$;
        **if** $p(s_c, A_c) \leq \lambda$ **then**
            break;
        $P = P - \{p(s_c, A_c)\}$;
        $ind(s_c) \leftarrow$ the index of $s_c$ in $S(I)$;
        **if** $A_c = A_{lq} \in \mathbf{A}(I)$ **then**
            **if** $\nexists A_{lr} \neq \mathbf{A}(I)$ *between the* $ind(s_c)$*-th and* $q$*-th positions in* $\mathbf{A}(I)$ **then**
                Set the $ind(s_c)$-th element in $\mathbf{A}(I)$ into $A_c$;
                Also set all elements between the $ind(s_c)$-th and $q$-th positions in $\mathbf{A}(I)$ into $A_c$;
    **while** $P \neq \emptyset$ *&&* $\exists$ *unlabelled segment in* $Pt$;
**return** $\mathbf{A}(I)$;

表 3-1 Comparing the Extraction Quality on PersonalBib

| Attribute | CRF | ONDUX | CNN-IETS |
|---|---|---|---|
| Title | 0.829 | 0.781 | **0.833** |
| Author | 0.856 | **0.861** | 0.859 |
| Journal | 0.772 | 0.803 | **0.932** |
| Volume | 0.787 | 0.822 | **0.987** |
| Pages | 0.832 | 0.831 | **0.979** |
| Date | 0.782 | 0.817 | **0.998** |
| **Average** | 0.809 | 0.810 | **0.929** |

表 3-2 Comparing the Extraction Quality on House

| Attribute | CRF | ONDUX | CNN-IETS |
|---|---|---|---|
| Name | 0.894 | 0.867 | **0.977** |
| Publish-time | 0.829 | 0.880 | **0.985** |
| Price | 0.846 | 0.877 | **0.985** |
| Charge-Method | 0.859 | 0.907 | **0.989** |
| Neighborhood | 0.837 | 0.875 | **0.984** |
| Unit | 0.835 | 0.880 | **0.987** |
| Floor | 0.847 | 0.894 | **0.985** |
| Furnitures | 0.821 | 0.871 | **0.996** |
| **Average** | 0.846 | 0.845 | **0.986** |

表 3-3 Comparing the Extraction Quality on Car

| Attribute | CRF | ONDUX | CNN-IETS |
|---|---|---|---|
| Brand | 0.844 | 0.857 | **0.972** |
| Price | 0.794 | 0.890 | **1.000** |
| Transmission | 0.825 | 0.872 | **1.000** |
| Color | 0.773 | 0.858 | **0.868** |
| Body | 0.769 | 0.866 | **1.000** |
| Engine | 0.801 | 0.830 | **1.000** |
| Fuel-consuming | 0.813 | 0.854 | **0.979** |
| **Average** | 0.803 | 0.861 | **0.968** |

验，比较了三种模型下的抽取质量，这三种模式分别是：纯粹基于卷积神经网络的方法(CNN)，卷积神经网络结合单向位置与序列模型的方法(CNN + PSM)，卷积神经网络结合双向位置与序列模型的方法(CNN + BPSM)。从表 3-5中可以看到，CNN + PSM的方法在所有三个数据集上均取得了最佳的F值。具体来讲，使用CNN+PSM仅仅比单纯使用CNN的抽取质量在三个数据集上平均提升了大概3-5%，但是，使用CNN+BPSM比单纯使用CNN的抽取质量在三个数据集上平均提升了大概5-8%。因此，我们得出结论，我们提出的双向位置与序列模型确实好过单向的位置与序列模型。

表 3-4 三种方法花费时间的比较

| Dataset | CRF | ONDUX | CNN-IETS |
|---|---|---|---|
| PersonalBib | 2350ms | 190ms | 150ms |
| House | 4670ms | 380ms | 340ms |
| Car | 3780ms | 200ms | 210ms |

表 3-5 CNN, CNN+PSM 和CNN+BPSM 三种模型抽取质量的比较

| Dataset | CNN | CNN + PSM | CNN + BPSM |
|---|---|---|---|
| PersonalBib | 0.862 | 0.891 | **0.929** |
| House | 0.903 | 0.943 | **0.986** |
| Car | 0.897 | 0.932 | **0.968** |

## 3.4 本章小结

本章，我们给出了分割式信息抽取(IETS)的正式定义，介绍了知识库的构建，并详细介绍了本文提出的基于卷积神经网络的贪婪式概率标注算法的每一部分，最后我们在三个数据集上与另外两种目前表现最佳的方法做了充分的对比实验，给出了详细的实验结果和分析，最终得到结论，文本提出的方法不仅在抽取质量还是在抽取效率上都明显好于其他方法。

# 第四章　系统的实现与展示

## 4.1　Tensorflow介绍

TensorFlow 是一个使用数据流图进行数值计算的开源软件库。图中的节点代表数学运算，而图中的边则代表在这些节点之间传递的多维数组（张量）。这种灵活的架构可让您使用一个API 将计算工作部署到桌面设备、服务器或者移动设备中的一个或多个CPU 或GPU。TensorFlow 最初是由Google 机器智能研究部门的Google Brain 团队中的研究人员和工程师开发的，用于进行机器学习和深度神经网络研究，但它是一个非常基础的系统，因此也可以应用于众多其他领域。

## 4.2　CNN模型构建细节

本文提出的基于深度卷积神经网络(CNN)的信息抽取模型，除了事先构建知识库和第一步做初始的文本分割，其他部分的执行都是基于我们构建的CNN分类器模型的输出结果。因此，我们的分类器模型需要能更准确的刻画各属性值的区别，有更高的准确率，但是，我们知道，卷积神经网络(CNN)的优势在于它能充分抽取特征和自由组合特征的能力，但是CNN模型的这个优势，从另一个角度来讲，恰恰也是它的一个劣势——缺乏模型解释性。因此，我们在使用卷积神经网络来构建深度学习模型时，更多时候我们只能把它看做黑盒子。但是，我们还是可以通过加入一些模型训练技巧，探索合适的超参数，改变模型结构来提高模型的表现力。在这一节，将详细介绍我们构建CNN分类模型的一些细节(我们将不再给出实验结果，我们会结合实验结果尝试解释如此选择的原因，从而得到一些经验相关的知识)：

1) *CNN模型结构:* 我们使用了文章 [**?**]中类似的模型结构，在输入层之后，是一个多卷积核的卷积层，然后接一个池化层，最后是一个softmax分类器。卷积层和池化层的层层叠加构成了卷积神经网络模型(有些模型没有池化层)，通常，现实中常用的基于卷积神经网络的深度学习模型都会有很多层的卷积层加池化层，特别是在做图像和语音的任务中。因此，我们在构建自己的模型时，尝试增加卷积层和池化层的数量，但是通过实验我们发现，这并没有提升模型的表现力，反而有略微的相反作用，并且更重要的一点是，增加了层数模型训练时间会有明显的增

加。模型层数多就代表模型需要学习的参数更多，当我们要做图像识别时，模型的输入是图像的像素矩阵，这个矩阵的特征密度非常高，因此我们需要一层一层的，从低阶到高阶的，从部分到整体的对输入矩阵进行处理，并在每一层进行特征的抽象和组合。图像中特征的刻画是非常复杂的，比如从低阶的边角特征组合成不同动物是需要在非常高阶上组合的，因此，增加模型的层数，才能够用更多的参数来刻画这种高阶的特征。在我们的应用场景中，首先因为我们的模型输入矩阵是训练好的word2vec，虽然是一种压缩的高密度文本表征方式，但是这种‘高密度’也是相对于one-hot编码而言，相比较图像的像素矩阵，所包含的特征密度应该是差的很远，因此我们不需要这么深的网络。另外，我们训练CNN模型的输入是来自于知识库中的属性值，它的文本长度相对较短，即使最终将所有输入都归一化之后，也不会是很大的输入矩阵，这种输入矩阵的大小也不需要我们使用过深的网络。网络越复杂过拟合的可能性越大。因此，这种单层卷积加单层池化的结构是最适合我们的应用场景的。

2) *CNN模型通道:* CNN网络中的不同通道可以解释为‘看’输入矩阵的不同角度，比如，在做图像识别时，我们有红，绿，蓝三个通道，模型在三个通道上同时运行，然后以某种方式组合起来。在我们的模型中，也采用了类似的模式。我们使用静态通道和动态通道，静态通道是指，对于我们的输入矩阵——word2vec，在训练我们的模型时此输入矩阵保持不变，而模型的其他参数随着模型训练而改变。动态通道是指，不仅模型的其他参数随着训练的过程而改变，输入的word2vec也会在训练过程中进行微调。具体来讲，在训练时，同时给模型两组训练数据，输入的都是词向量，然后在每一组训练数据上，每一组词向量就看做是一个通道，然后每个卷积核会同时在两组训练数据上执行，但是，梯度的反向传播只在其中一个通道上进行。这样就可以保持一个通道上输入词向量不变，也就是静态通道，而微调另一个通道的输入词向量，也就是动态通道。而静态通道和动态通道的实现在Tensorflow中也是比较简单，只需要在*Variable* 类中设置$trainable = True/False$，然后相应的改变后面步骤中张量的维度就行了。这种策略在文章 [**?**, **?**] 中有详细的讨论。

3) *卷积核数量和大小:* 我们的输入是一句话，假设规则化之后，输入文本长度是$d$，

词向量的维度为$V$，则输入矩阵便是一个$d \times V$ 的矩阵，每行代表一个词语。类似于语言模型中的$n-gram$ 思路，我们需要将相邻的若干个词语同时来考虑，也就是需要卷积核每次读入$m$ 行词向量，这个行数就看做是卷积核的大小。当确定了卷积核的大小之后，不同参数的卷积核可以抽取出不同的特征，因此看做是不同的卷积核。我们通过随机设定卷积的参数，便可以得到若干个不同的卷积核，每个卷积核都能够抽取出一个特征图(future map)，最终再综合考虑所有的特征图。为了充分的抽取特征，在卷积层中，我们使用了不同大小的卷积核，并且每种大小的卷积核中设置一定数量的卷积核。但是文章 [**?**] 中证明了，对于不同的数据集，卷积核数量和大小的选择是不同的，并且，对于每个通道使用相同大小的卷积核往往具有更好的表现。因此，我们通过实验得出了对于我们的每个数据集来说，最合适的选择，如表 4-1 所示。其中卷积核大小的选择空间设定为：$[3, 5, 7, 9, 10, 15, 20, 25]$，数量的选择空间设定为$[50, 100, 200, 300, 400, 500, 600]$。

4) *卷积方式:* 卷积方式有两种，宽卷积(Wide convolution)和窄卷积(Narrow convolution)，这两种方式的主要区别就是对于输入矩阵的边界值得处理。我们的模型中使用宽卷积，就是使用零填充(zero-padding)，这样就可以完全覆盖输入矩阵中所有的值，得到更大或者相同大小的输出特征图。

5) *激活函数:* 我们在不同的数据集上比较了几种激活函数方式，*Relu*，*tanh*，特殊的，我们也测试了不使用激活函数(*NoA*)时的表现，因为文章 [**?**] 中发现，在有些数据集中，不使用激活函数却得到了更好的结果。经过我们的试验发现，在引文数据集中，不使用激活函数取得了更好的实验结果。

6) *池化方式:* 在我们的模型中，池化层放在卷积层后面，池化的作用有两个，第一个作用是降低维度，第二个作用是统一输出形式。在我们的模型中，使用$1-max\ Pooling$，也是比较通用和有效的一种*Pooling* 方式，并且*Pooling* 窗口的大小就是输入文本的长度，这样每个经过卷积核最终只输出一个值。

7) *正则化:* 我们使用两种正则化策略，一个是*dropout* ，一个$l2$ 正则。我们使用了文章 [**?**]中相同的办法来寻找最佳的值，通过实验，我们得到图 4-1中的结果。其实，正则化对于模型的影响非常小，我们比较不同正则化值得区别也非常小。这可能是因为，我们的模型只是一个交浅层的网络，模型复杂度不

表 4-1 CNN模型在不同数据集上的超参选择

| 参数值 数据集 / 参数项 | 引文数据集 | 二手车 | 二手房 |
|---|---|---|---|
| 卷积方式 | 宽卷积 | 宽卷积 | 宽卷积 |
| 池化方式 | $1-max\ Pooling$ | $1-max\ Pooling$ | $1-max\ Pooling$ |
| 激活函数 | *NoA* | *Relu* | *Relu* |
| 卷积核大小和数量 | $(9,9)/200$ | $(10,10)/100$ | $(5,5)/400$ |
| 通道数 | 静态+动态 | 静态+动态 | 静态+动态 |
| 正则化 | $dropout: 0.5,\ l2: 20$ | $dropout: 0.4,\ l2: 40$ | $dropout: 0.4,\ l2: 20$ |

高，所以正则化的作用便没有那么明显。同时，我们也尝试在我们的模型中加入*batch normalization* 的技巧 [**?**]，发现并不会对模型的表现起到明显的帮助，只是稍微缩减了一点模型的训练时间。在$Tensorflow$ 中，我们可以直接调用高度抽象的函数*batch normalization* 来实现。在试验中，*dropout* 值的选择空间设定为：$[0.1, 0.2, 0.3, 0.4, 0.5]$，$l2$ 中的正则阈值选择空间设定为：$[3, 5, 10, 20, 30, 40, 50]$ 。

对于上面提到的参数，我们使用文章 [**?**]中相同的方法，通过较粗粒度的网格搜索(*grid search*)来给模型选择一套合适的参数。具体的过如表格 4-1所示。

## 4.3 系统架构及展示

为了方便执行抽取任务，方便移植到其他数据集，且可视化我们的抽取模型每一步的结果，我们做了一个自动化训练模型和抽取任务可视化的系统。整体来讲，这个系统主要分为下面三个部分：

1) 模型训练调参与训练集初始化

* 模型结构设计和参数设定：可以在网页上直接手动设计模型结构，制定超参的数值。
* 构造知识库（*Knowledge Base*），加载、处理和预览训练集，选择性构造特征：手动选择数据预处理相关参数，选择要构造的特征，并预览训练数据的初始化结果等。
* 深度学习模型训练过程可视化展现（TensorBoard）。

2) 输入文本执行抽取任务

* 输入一个文件，包含多条文本记录：对这个文件抽取结果的展示，即展示每条记录的属性值和相应属性名，BPSM的展示。
* 输入一条文本记录：展示模型每一步的中间结，（a）KB-based Blocking，展示每一块，标注出Anchor-block和Unknown-block。（b）CNN-labeling，展示归一化的CNN输出概率，展示目前分块和标注结果。（c）BPSM-amendment，展示BPSM模型以及最终的抽取结果。

3) 将结果存入到结构化文件中输出

## 4.4　本章小结

# 第五章 总结与展望

## 5.1 全文总结

随着互联网的高速发展和社交网络的持续增长，文本类型数据的量级呈现出爆炸式的增长，形式也变得越来越复杂和多样。文本类型数据中包含了大量的知识和有用信息，但因为文本类型数据的高度抽象化和形式多样化，自动的、准确的从文本数据中抽取有用信息变得更困难也更重要。人工智能的发展离不开知识的支撑，特别是从感知智能到认知智能的进化，必须具备大量的知识积累，并具有知识推理的能力。散落在互联网上的文本信息中包含了大量的有价值的知识，我们需要一种更准确和高效的方法，从这些海量文本中抽取出有用信息，构建我们的知识图谱，来支撑人工智能的发展。本文主要研究信息抽取领域的一个重要的任务——分割式信息抽取，即对一段语义丰富的半结构化文本首先按各属性的不同准确分割文本，然后给予每个文本段正确的标签。对于这种特殊的文本形式，因为其内部属性值之间没有关联，各属性值顺序不固定等根本原因，使得传统的方法的抽取质量不能够满足现实的要求，适应场景也收到限制。因此，本文提出了一种基于深度学习的贪婪式概率标注算法，并实现了一个自动训练和执行抽取的系统，可以更方便的训练模型，更方便的移植到其他数据集，更方便的执行抽取任务，并且可以更客观的观察我们提出的方法各步骤结果的具体展示。本文的具体研究内容如下：

（1）对深度学习在文本上的应用做了讨论和分析，重点介绍了词向量的训练思路和实用性，列举了深度学习在自然语言处理中若干个领域的解决思路。介绍了分割式信息抽取的概念，说明了分割式信息抽取的现实意义，深入研究了现有方法的解决思路，并对这些方法的优缺点做了详细的分析。

（2）提出了一种基于深度学习的贪婪式概率标注算法。对于传统方法中的匹配函数表现能力不足，应用场景受限等劣势，我们提出了一种适用场景更多，抽取质量更好的方法，即利用深度学习强大的数据表征能力，并结合概率模型来构建我们的模型，非常明显得提升了抽取质量。简要来讲，借助于事先构建的知识库，训练一个卷积神经网络模型（CNN 模型），并执行初始的分割，为了充分使用知识库，另外从知识库中求得了文本段在不同属性中词频先关的信息，用来识别锚点块，锚点

块在我们的模型中的具有非常重要的作用。对于我们的输入文本类型，各属性值虽然语法上独立，但是语义上存在着某种隐藏联系，本文提出的算法就利用了这种结构相关的信息。尽管CNN模型能够直接给出文本段属于各属性的概率，但是这种标注方式只是单独的考虑了这一个文本段，没有综合考虑这个文本段与输入文本中其他文本段的关系，因此这种方式可能不能找到最佳的分割和标注方式，因此本文提出了一种贪婪式概率标注算法，可以从全局考虑，找到最佳方案。我们的模型至此就已经可以取得非常好的效果了，但是为了进一步修正错标注和漏标注的情形，本文提出了一种双向位置与序列模型，这个模型可以捕获关于某一领域数据集中属性值分布的一些规律，并从前向和后向两个不同的方向来刻画这种信息，本文将充分利用这个重要的信息来进一步强化标注的结果。

（3）对于上面的模型，本文详细介绍了构建卷积神经网络模型的各个细节，并分析了一些超参的选择策略。另外，我们还实现了一个系统，用于更方便地训练模型、执行抽取任务并查看本文提出模型各步骤的中间结果，在文章中详细介绍了这个系统的各个模块内容。

## 5.2　工作展望

本文对分割式信息抽取任务进行了研究，分析了传统方法的不足之处，提出了基于深度学习的贪婪式概率标注算法，通过与以往方法对比，在几个真实数据集上验证了本文提出算法的效果，实验结果表明，所提出的算法在抽取质量上取得了非常明显的提升，并且在抽取效率上也要优于其他算法。然后，其中仍有可以改进的地方：

（1）本文提出的基于卷积神经网络的算法中，分割与标注是分开进行的，使用知识库做分割，然后将分割的结果再使用卷积神经网络来处理，然后基于CNN模型的输出结果设计一系列概率相关的标注算法。之所以这样处理，是因为对于文本来说，一个单词，或者中文中的一个词语，粒度太大了，我们必须根据知识库先做一步分割。然而，图像中处理思路却不是这样的，在做图像的物体识别任务时，定位和分类是放在一个模型中做的，比如在经典的物体识别模型Fast R-CNN [**?**]中，除了提取proposal，它将CNN特征提取、物体分类、物体框的定位回归放在同一个网络

中，实现了End-to-End的训练，这样会大大减少训练模型的时间，以及使用模型的时间，将分类和定位共同训练，从某种角度来说也能增加模型的精度。在我们提出的模型中，训练模型和执行抽取时各部分是隔离的，虽然我们提出了相关措施来避免独立考虑文本块，但这种模型仍然有很多弊端。因此，在我们的未来工作中，我们会考虑将文本块的定位与文本块的分类放在一个模型中，完全改变现有模型各步骤串行工作的方式，这会大大提高抽取效率。

（2）深度学习模型已经广泛的应用于自然语言处理的各个领域中，事实证明在一些领域深度学习模型非常适合。因此，我们将在未来的工作中，探索使用更多的深度学习模型来解决信息抽取任务，包括使用善于解决序列标注问题的LSTM，基于Attention机制的网络，对抗网络，强化学习等。另外，在构建CNN模型的输入矩阵时，本文使用了6种特征（word2vec也算做了一种），我们可以继续探索更多的、更有效的特征加入到输入矩阵中。在序列化数据的建模中，比如用RNN，LSTM这样的循环神经网络虽然可以学习到潜在的数据间依赖，但是其表达力依然有限，越来越多的研究工作开始关注引入外部知识库或者额外信息来丰富当前数据的语义表达，从而辅助数据理解。因此，我们认为对知识库的使用还不够完全，我们将在未来的工作中，探索如何在建模时将更多的外部知识融入到模型之中。

（3）在我们的模型中有两处阈值的设定，一个是定义锚点块的$\theta$，一个是控制低质量标注的$\lambda$，阈值的设定对模型的影响非常大。本文中提出的算法中所使用的两个阈值都是通过大量实验获得的，虽然我们在这三个数据集上得到了相同的阈值，但是目前还不能证明在所有的数据集上这个阈值也是适应的，我们还是认为阈值是对数据集敏感的。这就意味着，对于每个数据集我们都要通过做实验来找到最佳的阈值，这在实际应用中将是一个很繁琐的事情。对此，可以使用一些机器学习的方法或众包的方式去自动调整阈值。通过这些方法可以使算法更好的适应其它数据集，进而获得更好的抽取效果。

# 攻读硕士学位期间发表的论文

**会议论文:**

[1] **Qiang Yang**, Zhixu Li, Binbin Gu, An Liu, Guanfeng Liu, Pengpeng Zhao and Lei Zhao. CTextEM: Using Consolidated Textual Data for Entity Matching [A]. International Conference on Database Systems for Advanced Applications [C]. Springer International Publishing, 2016: 117-132. (**EI, CCF B 类**)

**申请发明专利:**

[1] 李直旭, **杨强**等. “一种用于实体匹配的方法及系统”. 申请日期: 2015年07月14日, 专利申请号: 201510407893.6

**软件著作权:**

[1] **杨强**, 李直旭等. “基于非主属性的概率决策树实体匹配软件”. 申请日期: 2015年01月23日, 登记号: 2015SR077638

**参与的科研项目:**

[1] 国家自然科学基金青年项目(61402313) “基于互联网海量信息的数据库文本类型数据清洗研究”, 2015/01-2017/12

# 致 谢

时光荏苒，岁月如梭，转眼间就要挥手告别研究生生活了。回顾这三年来的点点滴滴，我获得丰富的专业知识，更获得了许多宝贵的经验与回忆，同时也收获了许多感动。此时此刻，我要对这三年以来给予我帮助的亲人、老师、同学们表示衷心的感谢!

首先，我要感谢我的家人，感谢你们对我的支持与付出。是你们用辛勤的双手给了我良好的生活环境，也给了我克服困难的勇气。正是有你们的陪伴和鼓励，才使我不断突破自我，克服重重困难险阻，找到自己的人生定位。在今后的人生路上，有你们的陪伴也是我最大的幸福。

其次，我要衷心地感谢我的导师李直旭副教授。在我的研究生这三年中，李直旭老师无论在学术上还是生活中都给了我极大的帮助。在学术方面，李老师用渊博的学术知识、严谨的治学精神和正直的生活态度深深地感染了我，让我受益匪浅，在学术上对我进行悉心的教导，纠正我学术上错误；在生活方面，李老师教导我如何提升自信心，鼓励我多锻炼自己，这些方方面面对我我以后的生活产生了都将产生极大的影响。此外，我能够顺利地完成学业、发表多篇科研论文都离不开李老师无微不至的关心与爱护。在此，我要再次衷心地感谢李直旭老师，您是我今后学习的榜样，您的谆谆教导让我终生难忘。

接着，我要感谢实验室的所有同学们，感谢你们在学习和生活上无微不至的关怀和帮助。在此，我要特别感谢我的师兄蒋俊和师妹何莹，正是你们的帮助才让我顺利完成学术研究。同时，感谢顾斌斌、林天巧、旷晓鹏和周剑等同学在学术方面对我的帮助。接下来，我要感谢我的舍友们，与你们的朝夕相处，使我们建立起了深厚的友谊。

最后，感谢各位老师在百忙中对我的论文进行评审。

杨强

二〇一七年三月二十五日