硕士学位论文

基于感知器算法的高效中文分词与词性标注系统设计与实现

Design and Implementation of Efficient Chinese Word Segmentation and Pos-tagging System based on Perceptron Algorithm

邓知龙

哈尔滨工业大学

2013 年 6 月

内图书分类号：TP391.2　　学校代码：10213
国际图书分类号：681.37　　密级：公开

工程硕士学位论文

基于感知器算法的高效中文分词与词性标注系统设计与实现

硕 士 研 究 生：邓知龙
导　　师：刘挺教授
申 请 学 位：工程硕士
学　　科：计算机科学与技术
所 在 单 位：计算机科学与技术学院
答 辩 日 期：2013 年 6 月
授予学位单位：哈尔滨工业大学

Classified Index: TP391.2
U.D.C: 681.37

Dissertation for the Master Degree in Engineering

Design and Implementation of Efficient Chinese Word Segmentation and Pos-tagging System based on Perceptron Algorithm

Candidate: Deng Zhilong
Supervisor: Prof.Liu Ting
Academic Degree Applied for: Master of Engineering
Speciality: Computer Science and Technology
Affiliation: School of Computer Science and Technology
Date of Defence: June, 2013
Degree-Conferring-Institution: Harbin Institute of Technology

摘 要

分词、词性标注是自然语言处理的基础性课题，是很多其他自然语言处理任务的基础，同时在很大程度上影响着后续任务的最终性能。构建一个高性能、高效率的中文分词、词性标注系统具有重要的学术意义和应用价值。

本文着眼于构建一个性能优异、高效率的分词、词性标注系统。本文的研究内容主要包括三方面：词典和统计相结合的分词、词性标注方法，系统效率优化和性能提升以及基于感知器（Perceptron）算法的模型增量训练。

本文使用词典与统计相结合的分词、词性标注方法，不仅使分词、词性标注达到了一个较好的性能，而且通过将词典信息融入统计模型实现了中文分词领域自适应以及词性标注效率的提升。在此基础上本文实现了基于感知器的并行训练算法，在保证性能的前提下大幅度提高模型训练效率。此外，本文还通过对模型文件的压缩来提高速度以及减小内存需求。同时，本文使用半指导方法利用大规模未标注数据进一步提升词性标注准确率。然后我们利用感知器算法属于在线（Online）算法的优点，提出了基于感知器算法的模型增量训练方法，并通过实验验证了增量训练方法在相同领域的有效性。最后我们通过对跨领域中文分词增量训练结果不理想的原因进行深入分析，将 Stacked Learning 框架引入跨领域中文分词中。

实验结果表明，本文的分词、词性标注系统在性能上达到了目前分词、词性标注的最好性能，而且通过使用并行训练算法，可以大幅度的提高训练效率。实验结果也验证了本文提出的增量训练方法在相同领域数据中对于分词、词性标注任务的有效性。同时通过对比实验验证了 Stacked Learning 框架对跨领域中文分词的适用性。

关键词：分词；词性标注；感知器算法；并行训练；增量训练

Abstract

Word segmentation and Part-of-Speech tagging are fundamental research projects of natural language process and are the foundation of other nlp tasks. They can determine the final results of other tasks. A high-efficiency word segmentation and pos tagging system is not only important in terms of academic value, but also has important application value.

This work focuses on constructing an efficient word segmentation and pos tagging system with excellent performance. The major reaserch contents include the method of Chinese word segmentation and pos tagging combining statistic model and dictionary, the optimization of system efficiency, performance improvement and incremental training based on perceptron algorithm.

By using the method combining statistic model and dictionary, we achieved a comparable performance in word segmentation and pos tagging and by integrating dictionary information into statistic model we implement the domain adaption for Chinese word segmentation and efficiency improvemen for pos tagging. We then implement perceptron parallel training algorithm and improve triaing efficiency significantly without great loss of performance. We also decrease the memory requirement and accelerate test speed by compressing model file. Meanwhile, we also improve performance of pos tagging by using large-scale unlabeled data. Based on the advantages of online algorithm, three different incremental training methods based on perceptron algorithm are proposed and the validity of new methods have been affirmed through experiments. Finally, analyse the causation of fail in cross-domain Chinese word segmentation deeply and stacked learning frame is applied in cross-domain Chinese word segmentation.

Experimental results show that our system achieves the state-of-art performance of Chinese word segmentation and pos tagging and by using parallel training we can greatly improved training efficiency. The results also show that the increamental method proposed in this paper is valid in same domain dataset for Chinese word segmentation and pos tagging and stacked learning frame is effective for cross-domain Chinese word segmentation.

Keywords: Word segmentation, Part-of-Speech tagging, Perceptron algorithm, parallel training，Incremental Training

目 录

Contents

第1章 绪 论

1.1 课题来源及研究目的和意义

当今世界正处在一个信息极速鼓胀的时代。随着互联网的快速发展，信息的规模极速扩大，信息量急速增加，信息的来源也更加广泛，信息的内容涵盖范围越来越广，已经渗透到我们生活的方方面面。互联网的蓬勃发展对当今社会的发展产生了深刻影响，也在不断改变着人们生活的每一方面。在当今社会，语言信息处理能力已经成为国家科技实力的一个重要衡量标准。同时，信息的快速膨胀也对信息的自动处理带来了巨大的挑战，如何高效地处理海量文本，并从中抽取出有价值的语言信息，已经成为自然语言处理的重要课题。

随着我国科技实力的不断提升，中国已经成为世界互联网的重要一员。以中文为载体的信息已经成为互联网信息的一个重要组成部分。中文自然语言处理的相关研究起步时间落后于西方先进国家近 20 年，而且由于中文具有自身的特殊性，直接将英语等相关语言的处理技术应用到中文中并不能取得预期的理想效果。

中文是以字为最小单位的语言，由字构成词，词构成句子，句子构成段落，段落构成文本。和英文不用的是，在中文中词和词之间没有明显的分隔符，但是在文本的自动处理中，计算机处理的最小单元一般是词语。因此中文分词构成了中文自然语言处理的基础任务。中文分词结果的好坏将直接影响到后续相关任务的最终性能。其次，词性标注在自然语言处理中也具有举足轻重的地位，词性标注即根据上下文给每一个词语赋予一个正确的候选词性。词性标注是进行词法分析、语义分析以及语用分析等高层任务的基础和前提。根据词语的词性，能够消除词语歧义，减小查询模糊，同时词语词性还为更高层次的自然语言处理任务提供了非常有用的信息。词性标注的准确率在很大程度上决定了更高层次任务处理结果的性能好坏。

目前主流的中文分词、词性标注方法都使用到了大规模语料库。信息的极速膨胀给相关的语料的处理技术提出了新的挑战，如何在给定大规模的语料情况下保证相关处理任务的准确率而且能够高效率的完成相关任务，已经成为人们必须考虑解决的重要课题。同时随着可用语料规模的不断扩大扩大，

原有的单线程、顺序执行的语言模型在实际应用中面临很前所未有的挑战，许多常用的机器学习模型都提出了相应的分布式并行算法。

另外，语料的内容随着时间的推移在不断改进、增加，在使用原有语料训练出一个初始模型后，新的可用语料又接踵而至。大多数的统计机器学习模型一旦训练完成，直接使用新的训练语料改进原有模型将会非常困难，解决这一问题的方法就是使用所有语料重新训练模型。而在实际中，语料库资源往往不是对所有人进行免费开放的，因此这种方法将有可能会面临语料库知识产权的问题。此外使用所有语料完全重新训练模型将会浪费大量的资源和时间。而在线算法[1]恰恰可以很好的解决这一问题，在线算法训练模型进行更新时不需要使用所有的训练实例，每次只使用一个训练实例更新模型参数。使用在线算法完成模型训练后，当有新的可用训练语料时，在不需要使用原有训练语料的前提下，根据现有新增语料和原有模型进行增量训练，得到一个性能更好的模型，这样既可以避免对原有语料资源的需求同时可以大大的节约使用所有语料重新训练模型所需时间和资源。

感知器算法[2]是一种典型的在线算法，具有模型简单、速度快以及在线算法的优点。本文将实现一个基于 Average Perceptron 算法的中文分词、词性标注系统，在保证系统性能的前提下，实现模型的并行训练以及增量训练。

1.2 国内外研究现状

自然语言处理研究在国内起步较晚，和国外相比在时间上大约有近 20 年的差距。国外在 20 世纪 60 年代就已经出现了完整的自然语言处理系统，而国内最早的中文分词系统 CDWS[3]在 1983 年由北京航空航天大学实现。随着我国科技的不断发展，特别是近几年来我国互联网的快速发展，目前国内的自然语言处理技术已经取得了显著的进步，特别是在中文自然语言处理中，很多技术已经达到了世界领先水平并，很多技术已经应用到实际生活中并且取得了巨大的成功，创造了巨大的商业价值。

1.2.1 中文分词研究现状

中文和英文等语言不同，词语和词语之间没有明显的分隔符。而在大部分应用中，我们处理的最小单元是词语而不是构成词语的字，因此，中文分词构成了中文自然语言处理的基础。目前中文分词方法可大体分为四类：基

于词典的中文分词方法，基于规则的中文分词方法、基于统计的中文分词方法以及规则与统计相结合的分词方法。

1、 基于词典的中文分词的方法

基于词典的中文分词方法[4]首先需要构建一个大规模词典，词典中包含常用的词语。对文本分词时根据不同的匹配规则将文本与词典内容进行匹配，然后对文本进行切分完成分词操作。其中比较有代表性的基于词典的分词方法有正向最大匹配[5,6]、逆向最大匹配[7]、双向匹配算法[8]以及 N-最短路径分词算法[9]等。

2、 基于规则的中文分词方法

基于规则的方法通过让计算机模型人对句子的理解过程来实现句子的分词操作。这种方法从语言学角度出发，通过分析句子的语义来实现对文本的切分。

1991 年，何克抗等[10]首次将专家系统引进分词技术中，专家系统主要有两部分构成，一部分是独立的知识库，另一部分是一个推理机。系统将分词的过程转变为知识的推理过程，也就是所谓的句子“分词树”的生长过程。由于该方法首先需要构建一个规模巨大的知识库，而这时一个非常耗时、非常困难的任务。此外，推理的过程涉及到人工智能中的很多技术，要实现完全的自动推理面临很多目前无法解决的困难，因此该系统并未得到大规模的推广和使用。虽然该系统由于各种原因没能得到广泛的应用，但其理论分析和指导思想得到了普遍的关注，影响了很多后续系统的开发。

在原有的基于规则的中文分词方法基础上，1998 年，曹星明等人提出了基于多种知识源的汉语分词方法[11]。2004 年，王彩荣等设计了汉语自动分词的专家系统[12]。2005 年张茂元提出一种基于语境的分词方法[13]。

3、 基于统计的中文分词方法

随着大规模语料库的建立，统计机器学习算法在中文分词中得到了广泛的应用。目前基于大规模语料库的统计学习方法已经成为中文分词的主流方法。常用的统计机器模型有隐马尔可夫模型（Hidden Markov Model，HMM）[14]，最大熵模型（Maximum Entropy Model，ME）[15]，条件随机场模型（Conditional Random Fields ，CRF）[16]等。

2002 年中科院计算所基于层叠隐马尔科夫模型实现了分词系统 ICTCLAS[17]，该系统取得了较好的性能并在学术界和商业界得到了广泛的

推广和使用。

2003 年 Nianwen Xue 首次提出了将中文分词问题看做序列标注问题的思想[18]，通过为每一个字赋予一个候选符号来标记词语的开始、结束位置，常用的候选符号有 B-I-O、B-M-E-S 等。从而将很多序列标注模型应用到中文分词研究中，并取得了巨大的成功。

2004 年 FuChun Peng 等首次提出使用 CRF 模型进行中文分词的方法[19]；JK Low 等人将最大熵模型（ME）应用到中文分词中[20]。

4、 规则与统计相结合的分词方法

在规则与统计相结合的中文分词方法[21]中，往往根据语言学知识使用一些人工制定的规则处理统计模型无法学习到的语言特性从而在统计模型的基础上提高最终的分词性能。

1.2.2 词性标注研究现状

目前词性标注方法可分为三类：基于规则的词性标注方法、基于转换的错误驱动词性标注方法以及基于统计的词性标注方法。

1、 基于规则的词性标注方法

基于规则的词性标注方法首先由语言学专家制定相应的规则，在规则中使用大量的上下文信息来对词性进行判断。基于规则的方法首先面临的问题是规则的制定由具有语言学知识的专家手工完成，词性标注的性能与规则制定者的语言学知识具有很大的关系。其次要构造一套对语言的各方面特性都覆盖的规则是一个很艰难很耗时的工作，而且随着规则数量的增加，各规则之间往往会产生冲突。最具有代表性的基于规则的词性标注系统是 1971 年开发的 TAGGIT 标注系统[22]。

2、 基于转换的错误驱动词性标注方法

为了克服手工制定规则带来的问题，1995 年 Eric Brill 提出了基于转移的错误驱动的词性标注方法[23]。该方法从大规模词性标注语料库中通过模版采用错误驱动的方式学习到大规模的规则，最后使用这些规则对文本进行词性标注。该方法避免了人工制定规则所需要的人力、物力资源，而且解决了各规则之间冲突问题。

3、 基于统计的词性标注方法

基于统计的词性标注方法[24-26]将词性标注看作是一个序列标注问题，

为每一个词语赋予一个正确的候选词性。在基于统计的词性标注中，使用的模型和基于统计的分词模型基本一致。

近几年很多学者使用联合模型同时进行分词和词性标注任务，该方法首先在 2008 年有张跃提出[27]。通过使用联合模型，分词和词性标注之间可以互相进行帮助，能同时提升分词、词性标注的性能。当然除了必须同时进行分词、词性标注任务外，和 Pipeline 方法相比效率会有所降低。

1.3 基于感知器的序列标注算法

目前基于统计的方法是中文分词、词性标注中的主流方法。在基于统计的分词、词性标注方法中，问题抽象为统一的序列标注问题，即给定一个观测序列$\mathrm{X}=(x_1,x_2,\ldots,x_n)$，需要求解最优的标记序列$\mathrm{Y}=(y_1,y_2,\ldots,y_n)$。

其中一类方法从概率的角度来估计 X 和 Y 的概率分布，这类方法常用的统计模型有隐马尔科夫模型（Hidden Markov Model，HMM）[28,29]、最大熵模型(Maximum Entropy Model，ME)[30]以及条件随机场模型（Condition Random Fields Model，CRF)[16]。

在另一类序列标注算法中，定义观测序列$\mathrm{X}=(x_1,x_2,\ldots,x_n)$对应状态序列$\mathrm{Y}=(y_1,\mathrm{y}_2,\cdots y_n)$的分数（$score$）为公式（1-1）所示：

$$score(\mathrm{X},\mathrm{Y})=\sum_{k=1}^{K}w_k f_k(X,Y) \qquad (1\text{-}1)$$

其中$f_k(X,Y)$为特征函数，w_k为第 k 个特征对应的权重。

给定观测序列$\mathrm{X}=(x_1,x_2,\ldots,x_n)$，最好的状态序列 Y 为 $score$ 最大的状态序列，即如公式（1-2）所示：

$$\mathrm{Y}=argmax_{Y'}score(\mathrm{X},\mathrm{Y}') \qquad (1\text{-}2)$$

当通过训练得到每个特征对应的权重后，我们可以使用 Viterbi 算法快速的得到 $score$ 最大的状态序列。

在线算法[31]是一种常用的训练算法，在在线算法中，每次仅仅使用一个实例对参数进行更新，而不像梯度下降[32]之类的批处理训练算法，每次更新参数都需要用到所有的训练语料，导致对资源的巨大消耗。在线算法伪代码如图 1-1 所示：

Training data: $T=\{(x_t,y_t)\}_{t=1}^{T}$
1. $w_0=0;v=0;i=0$
2. for n:1..N
3. for t:1..T
4. $w^{(i+1)}$= update $w^{(i)}$ according to instance (x_t,y_t)
5. $v=v+w^{(i+1)}$
6. $i=i+1$
7. $w=v/(N*T)$

图 1-1 在线算法伪代码
Fig.1-1 pseudo code of Online Algorithm

在线算法第 4 步需要提供对参数的更新方式，不同的在线算法在更新参数时使用的方式不同。

感知器算法[33]是一种典型的在线算法。感知器算法每次使用一个训练实例对模型参数进行更新，在更新参数时每次将需要更新的参数权重加 1 或者减 1。为了防止模型对数据的过拟合，常对参数进行平均化操作，即 Average Perceptron 算法。Average Perceptron 算法伪代码如图 1-2 所示：

Training data: $T=\{(x_n,y_n)\}_{n=1}^{T}$
1. $w_0=0;v=0;i=0$
2. for t:1..T
3. for n:1..N
4. $y'_n=\text{argmax}_{y\in G(x_n)}\ w\bullet\Phi\ (x_n,y)$
5. $w^{(i+1)}=w^{i}+\Phi(x_n,y_n)-\Phi(x_n,y'_n)$
6. $v=v+w^{(i+1)}$
7. $i=i+1$
8. $w=v/NT$
9. *return* w

图 1-2 Average Perceptron 伪代码
Fig.1-2 pseudo code of Average Perceptron

Average Perceptron 算法不仅具有简单、速度快的特点，而且具有在线算法的优点，更新参数不需要一次使用所有训练语料，而只需要每次使用一个训练实例。

1.4 本文研究内容

本文研究内容主要包括三部分：词典和统计相结合的分词、词性标注方法；系统效率优化及性能提升；基于感知器算法的模型增量训练。

1.4.1 词典和统计相结合的分词、词性标注方法

在第三章我们使用词典与统计相结合的方法来实现中文分词、词性标注系统，训练算法使用在线算法中的 Average Perceptron 算法，使用 Viterbi 算法[34]作为解码算法。我们从训练语料中自动抽取分词、词性标注词典。在分词中，提取词典相关的特征，将词典信息融入到统计模型中，这样不仅能提高分词性能，同时还能够在跨领域进行分析时通过添加新领域专有名词对词典进行扩充实现中文分词的领域自适应。在词性标注中通过使用词典我们能够限制出现在词典中词语的候选词性，这样能够避免给一些常用词标注错误词性，提高词性标注准确率，同时能够减小算法解码的搜索空间，提高词性标注的效率。同时我们还对感知器算法的三种不同的参数处理方法进行了比较，验证不用处理方法对最终性能的影响。

1.4.2 效率优化及性能提升

虽然感知器算法在训练和测试上速度优于 ME、CRF 等复杂模型，但是随着语料规模的不断增加，模型的训练仍然是一个非常耗时的过程。因此在第四章中我们首先实现了基于感知器的并行训练算法，通过并行训练不仅充分利用了硬件资源，而且在保证性能的前提下大大的减少了训练所需要的时间，提高训练效率。然后我们在保证性能的前提下，通过模型权重的统计对特征进行裁剪实现对模型进行压缩，减小测试所需的内存以及加快测试速度。最后我们使用半指导的方法将大规模未标注语料应用到词性标注任务中，通过增加聚类特征来提高词性标注性能。

1.4.3 基于感知器算法的模型增量训练

随着时间的推移，语料库的内容在不断的扩充，规模也在不断的增大。当使用原有语料训练得到一个初始模型后，像 ME、CRF 这类使用批处理训练算法的统计模型将很难直接使用新增语料对初始模型进行改进，如果使用

所有语料重新训练模型，将会消耗大量的时间和资源，而且往往会面临语料库的知识产权问题。而在线算法的优点就是每次更新参数只使用一个训练实例，而不是所有训练数据，因此我们可以利用在线算法的这个优点实现模型增量训练。在第五章中我们提出了三种基于感知器的增量训练方法，并在分词、词性标注任务上进行对比实验，验证模型增量训练方法的有效性。通过分析跨领域中文分词中增量训练方法性能不佳的原因，我们将 Stacked Learning 框架应用到跨领域中文分词任务中，提高跨领域中文分词的性能。

第2章 词典与统计相结合的分词、词性标注方法

在中文分词和词性标注中，词典的使用对于性能具有重要的影响。在分词中，构建一个包含大规模词语的词典，在分词的过程中使用词典来提取相关特征将词典信息融合到统计模型中，不仅能够提高中文分词性能，而且通过使用新领域中的专业词语对词典进行扩充，还能够实现中文分词的领域自适应；在中文词性标注中，构建一个大规模的包含常用词语以及其可能词性的词典，通过提取词典相关的特征，不仅能够提高词性标注的性能，还能够减小模型解码时的搜索空间，提高解码的速度。

由于 Average Perceptron 算法具有速度快，性能高，同时具有在线算法的优点，因此本文使用 Average Perceptron 算法作为模型的训练算法。

2.1 词典构建

人工构建词典不仅需要具有丰富的语言学知识，而且是一个非常耗时、耗力的工作。因此，在本文中不直接手工构建相应的词典，而是从训练语料进行统计自动构建所需的分词、词性标注词典。

在中文分词中，统计训练语料中每个词语的词频，根据设定的阈值选取出现次数大于等于阈值的词语构成分词使用的词典；在词性标注中，首先统计每个词语的词频，然后根据设定的阈值获取词性标注词典中的词语，在从训练语料中获取每一个词性标注词典中的词语的所有出现的词性作为该词语的候选词性，构建词性标注词典。

在本文使用的方法中，设定分词阈值和词性标注阈值均为 3。即对在分词训练语料中出现 3 次或者 3 次以上的词语，将其包含在分词词典中。在词性标注中，选取出现次数大于等于 3 的词语以及词语在训练语料中出现的所有词性构建词性标注词典。

2.2 词典与统计相结合的中文分词方法

基于统计的中文分词方法目前达到了中文分词的 state-of-art 性能，但是完全基于统计的中文分词方法往往不具有良好的领域自适应性。外部词典能够很大程度的影响中文分词的最终性能，而且获取外部词典需要的时间、经

济代价远远小于标注一个新领域分词训练语料所需要的代价。如果中文分词方法能充分合理的利用外部词典，一方面可以提高中文分词的准确率，另一方面还可以通过词典的扩充实现中文分词的领域自适应性。当将一个分词模型应用到一个新的特定领域时，我们只需要将该领域的专有词语添加到原有词典中对词典进行扩充，便可大幅度的提高分词模型在新领域的分词性能。该方法的思想与张梅山等[35]的文章相似。

在本文中，使用词典与统计相结合的中文分词方法，通过将词典信息融入统计模型，不仅使分词性能到达当前分词性能的 state-of-art，而且通过词典的简单扩充实现了中文分词的领域自适应性。

2.2.1 分词使用的特征

在本文的基于词典和统计相结合的中文分词方法中，将使用的特征分为两类：基本特征以及外部词典特征。

（1） 基本特征

基本特征是基于文本本身的内容提取的特征。在本文的分词方法中，使用到的基本特征为：

- 字符 n-gram 特征
  1） unigram 特征：c_i，$i=-2$，$-1,0,1,2$
  2） bigram 特征：c_i, c_{i+1}，$i=-2,-1,0,1$ ；c_i, c_{i+2}，$i=-2,-1,0$
  3） trigram 特征：$c_i, c_{i+1}c_{i+2}$，$i=-1$

其中c表示的是待分词句子中的字，下标 i 表示的是相对于提取特征的字的相对位置。

- 字符类别特征
  1） unigram 特征：$charType(c_0)$
  2） trigram 特征：$charType(c_{-1}), charType(c_0), charType(c_1)$

其中$charType(c_i)$表示的是字符c_i的类别。在我们使用的字符类别特征中，共包含五类字符类别，如表 2-1 所示：

表 2-1 字符类别
Table 2-1 character type

字符类别	含义	示例
Punc	标点符号	，、。
Digit	阿拉伯数字	1、4、8
Chinese-Digit	中文数字	一、二
Letter	英文字母	A、b
Other	其他	练、行

- 字符叠字特征：$dup(c_0,c_1)$,$dup(c_0,c_2)$

字符叠字特征表示考虑的两个字是否是完全相同的两个字。其中$dup(c_0,c_1)$表示的是c_0、c_1是否是完全相同的字；$dup(c_0,c_2)$表示的是c_0、c_2是否是完全相同的两个字。

（2） 外部词典特征

外部词典特征时使用文本内容以及外部词典提取的特征。在根据外部词典提取特征的时候，我们根据基于词典的中文分词方法思想提取三种不同的外部词典特征。

给定句子$\mathrm{x}=c_1\cdots c_n$，以及外部词典D，考虑其中的第j个字符$c_j\ (1\le j\le n)$，定义三个外部词典特征如公式(2-1)、(2-2)及(2-3)所示：

$$f_B(\mathbf{x},j,\mathbf{D})=\max\ l,\ s.t.\ \begin{cases}\mathbf{w}=\mathrm{c_j}\cdots c_{j+l-1}\in\mathbf{D}\\ j+l-1\ \le n\end{cases} \tag{2-1}$$

$$\mathrm{f_M(x,j,D)}=\max\ \mathrm{l},\ \mathrm{s.t.}\begin{cases}\mathrm{w}=\mathrm{c_s}\cdots\mathrm{c_{s+l-1}}\in\mathrm{D}\\ \mathrm{j}<\mathrm{s}+\mathrm{l}-1\ \le\mathrm{n}\\ 1\le\mathrm{s}<\mathrm{j}\end{cases} \tag{2-2}$$

$$f_E(\mathbf{x},j,\mathbf{D})=\max\ l,\ \mathrm{s.t.}\ \begin{cases}\mathbf{w}=c_{j-l+1}\cdots c_j\in\mathbf{D}\\ 1\le\mathrm{j}-l-1\end{cases} \tag{2-3}$$

其中$\mathbf{w}$表示词语。$f_B(\mathbf{x},j,\mathbf{D})$表示以当前位置的字作为开始的出现在词典中的最长子串长度；$f_M(\mathbf{x},j,\mathbf{D})$表示的是以当前位置的字作为中间位置的出现在词典中的最长子串长度；$f_E(\mathbf{x},j,\mathbf{D})$表示以当前位置的字作为结束的出现在词典中的最长子串长度。

通过使用这三个外部词典特征，我们将基于词典的特征抽象为子串的长度融入到统计模型中，提高中文分词性能，在跨领域分词时，通过添加新词语对词典进行更新扩充实现中文分词的领域自适应。

2.2.2 实验结果及分析

在分词实验中，我们使用 the second International Chinese Word Segmentation Bakeoff[36]发布的分词语料进行实验。该数据共包括三个不同的分词语料 Microsoft Research Asia (MSR), City University of Hong Kong (CU), and Peking University (PKU)。语料中字和词的统计信息如表 2-2 所示：

表 2-2 语料字和词语统计信息

Table 2-2 statistic result of corpus

Corpus	Word Types	Words	Character Types	Characters
CU	69,085	1,455,629	4,923	2,403,355
PKU	55,303	1,109,947	4698	1,826,448
MSR	88,119	2,368,391	5167	4,050,469

由于三种语料中没有划分出开发集，因此从训练集中按照 8:1 将训练语料划分为训练集和开发集。根据开发集性能选择模型迭代次数，然后对测试集进行测试。

在分词中，使用准确率、召回率以及 F 值对性能进行评价。三个评价指标的定义如公式(2-4)、(2-5)、(2-6)所示：

$$\mathrm{P} = \frac{\text{分词结果中正确切分的词数}}{\text{分词结果中的词语数}} \tag{2-4}$$

$$\mathrm{R} = \frac{\text{分词结果中正确切分的词数}}{\text{测试语料中正确的词语数}} \tag{2-5}$$

$$\mathrm{F} = \frac{2\times P\times R}{(P+R)} \tag{2-6}$$

对于三种语料，均使用前面介绍的词典获取方法从训练语料中获取词典进行模型训练，然后根据模型在开发集上的分词性能确定迭代次数对测试集进行测试。为了验证本文方法的有效性，我们与相关论文中的实验结果进行了对比，对比结果如表 2-3 所示：

表 2-3 分词实验结果

Table 2-3 result of word segmentation

Corpus	Method	P(%)	R(%)	F(%)
MSR	Best05 (Tseng et al., 2005)[37]	96.2	96.6	96.4
	CRF + rule-system (Zhang et al., 2006)[38]	97.2	96.9	97.1
	Semi-Markov perceptron (Zhang and Clark, 2007)[39]	N/A		97.2
	Semi-Markov CRF (Gao et al., 2007)[40]	N/A		97.2
	Latent-variable CRF (Sun et al., 2009b)[41]	97.3	97.3	97.3
	CRF+ joint model(Sun 2012)[42]	97.6	97.2	**97.4**
	Our	96.7	**97.3**	97.0
CU	Best05 (Tseng et al., 2005)	94.1	94.6	94.3
	CRF + rule-system (Zhang et al., 2006)	95.2	94.9	**95.1**
	Semi-Markov perceptron (Zhang and Clark, 2007)	N/A	N/A	95.1
	Latent-variable CRF (Sun et al., 2009b)	94.7	94.4	94.6
	CRF+ joint model(Sun 2012)	94.8	94.7	94.8
	Our	94.5	**95.2**	94.9
PKU	Best05 (Tseng et al., 2005)	95.3	94.6	95.0
	CRF + rule-system (Zhang et al., 2006)	94.7	95.5	95.1
	Semi-Markov perceptron (Zhang and Clark, 2007)	N/A	N/A	94.5
	Latent-variable CRF (Sun et al., 2009b)	95.6	94.8	95.2
	CRF+ joint model(Sun 2012)	95.8	94.9	**95.4**
	Our	95.0	95.0	95.0

从表 2-3 的实验数据可以看出，当前系统的性能与目前分词研究的目前最好性能非常接近。但是分词性能优于本系统的方法基本都是使用如 CRF 这样的复杂模型，以及添加了一些复杂的规则或者是使用了联合模型。

2.2.3 分词领域自适应

在基于有指导的中文分词方法中，训练语料和测试语料的领域是否相同对于分词的性能有较大的影响。如对于一些特殊领域如娱乐、医疗、金融等

领域，在这些领域中，存在很多在其他领域不会出现的专有词语。在分词的时候，是否能够正确切分这些专有词语将决定性的影响分词的最终性能。

词典对于分词性能具有显著的影响，特别是对于特定领域，如果使用的词典包含有该领域的专有名词，那么将显著的提高分词性能。而且特定领域词典的获取要远远比标注该领域的大规模分词语料要容易得多。

在本文的词典和统计相结合的中文分词方法中，将外部词典特征抽象为子串的长度信息融入统计模型。这样当将现有分词模型应用到一个新领域时，只需要添加新领域的专有词语对词典进行扩充就可实现分词的领域自适应，不需要标注新领域的分词语料。这样不仅避免了标注新语料所需的人力、物力资源，而且避免了重新训练模型所需要的时间和资源。

为了验证本文分词方法的领域自适应能力，我们使用 PKU 语料作为训练语料，然后使用杭州同花顺公司提供的金融领域语料作为测试语料来进行对比实验。该金融语料共包括 51763 个句子，1326711 个词语。在训练模型时，从 PKU 语料中提取词典，使用 PKU 训练语料训练模型。在测试时，添加 1094 个金融领域的专有名词来扩充 PKU 词典，再使用 PKU 语料训练的模型对金融语料进行测试。实验结果如表 2-4 所示：

表 2-4 领域自适应实验结果

Table 2-4 result of domain adaption

词典	P(%)	R(%)	F(%)
PKU	86.20	89.61	87.88
PKU+金融	**90.63**	**92.16**	**91.39**

其中 PKU 表示直接使用 PKU 模型已经 PKU 训练语料中提取的词典进行测试，PKU+金融表示使用 PKU 模型进行测试，同时使用金融词语对 PKU 词典进行扩充。从表 2-4 实验结果可以看出，通过在 PKU 词典的基础上扩充金融领域专有词后，金融领域语料的分词性能得到了显著提升。这说明了本文使用的分词方法对于领域自适应具有显著效果。

2.3 词典与统计相结合的词性标注方法

在中文词性标注中，词典的内容为常用词以及词语对应的可能词性。在本文使用的中文词性标注方法中，仍然使用 2.1 节介绍的词典获取方法从训练语料中获取词性标注词典。

通过使用词典，将出现在词典中的词语的候选词性限制在词典中出现的

可能词性范围内。这样不仅避免了为词语分配不可能的词性，同时减小了解码时的搜索空间，加快解码速度。

2.3.1 词性标注使用的特征

在词典和统计相结合的词性标注方法中，将使用的特征分为基本特征和外部词典特征.

（1） 基本特征

- 词语 n-gram 特征
  1） unigram 特征： w_i，$i=-2$，-1，0，1，2
  2） bigram 特征： w_i，w_{i+1}，$i=-1$，0；w_{-1}，w_1

其中 w 表示需要进行词性标注的词语，下标 i 表示词语相对于当前考虑词语的相对位置。

- 字、词组合特征
  1） $lastChar(w_{-1})w_0$
  2） $firstChar(w_0)w_1$
  3） $firstChar(w_0)lastChar(w_0)$

其中$lastChar(w)$表示词语 w 的最后一个字，$firstChar(w)$表示词语 w 的第一个字。

- 词语前后缀特征
  1） 词语前缀特征：$prefix(w_0,i)$, $i=1$，2，3
  2） 词语后缀特征：$suffix(w_0,i)$，$i=1,2,3$

其中$prefix(w,i)$表示词语 w 长度为 i 的前缀子串；$suffix(w,i)$表示词语 w 长度为 i 的后缀子串。

- 词语长度特征：$len(w_0)$

在计算词语长度特征时，为了避免词语长度特征造成的数据稀疏，使用公式(2-7)计算词语长度：

$$len(w)=\begin{cases} length(w)\ if\ length(w)\le 4 \\ 5\ if\ length(w)\ge 5 \end{cases} \tag{2-7}$$

即如果词语长度小于五，则直接使用词语长度，如果词语长度大于等于五，则将词语长度特征设定为五。

（2） 外部词典特征：$lexicon(w_0,D)$

其中 D 表示词性标注使用的词典，$lexicon(w,D)$表示在词典中，词语 w

的可能词性。例如在人民日报 1998 年语料中，词语“规范”可能的词性有动词（v）、形容词（a）、区分词（b）以及名词（n），那么词语“规范”的外部词典特征为 lexicon=“word=规范/pos=v_a_b_n”。

2.3.2 实验结果及分析

在近几年的关于词性标注的研究中，已经没有学者单独将词性标注进行单独的研究，最近几年的论文中均是将分词和词性标注作为联合模型同时进行处理。在联合模型中，词性标注的评价标准与单独进行词性标注评价不一致，因此性能之间无法直接进行比较。

在调研前人的工作中发现单独进行词性标注研究中很多使用了人民日报 1998 年 1 到 6 月份的语料作为实验数据，但是各个方法对数据的划分又不一致。在此，列出了一些在 98 年人民日报语料上的中文词性标注实验结果，以便对该系统的整体性能有所认识。在本文实验中，使用 2-6 月的语料按照 8:1 划分为训练集和开发集，1 月份语料作为测试集。对比实验结果如表 2-5 所示：

表 2-5 词性标注性能比较

Table 2-5 performance comparison of POS Tagging

Method	使用语料	准确率(%)
基于改进的隐马尔科夫模型的汉语词性标注[43]	抽取 30 万词作为训练集，5 万作为测试集	96.20%
基于完全二阶隐马尔科夫模型的汉语词性标注[44]	抽取 30 万词作为训练集，5 万作为测试集	97.12%
基于条件随机场(CRFs)的中文词性标注方法[45]	1 月份数据分为训练集和测试集，训练集词数是测试集的 2.37 倍	96.60%
基于 SVMTool 的中文词性标注[46]	2-6 月作为训练语料，1 月份作为测试语料	97.84%
Our	2-6 月作为训练语料，1 月份作为测试语料	**97.98%**

从表 2-5 实验数据可以看出，虽然各个系统对实验数据的划分不完全一致，但是仍然可以看出，本文使用的方法在该数据集上，词性标注性能达到了 state-of-art。在相同数据划分对比实验中，本文的基于 Average Perceptron 的中文词性标注方法性能优于于 SVMTool 的中文词性标注方法。

为了与最新的相关研究结果进行比较，根据最新论文中使用的数据和评价标注，选取 CTB5.0 语料同时进行分词、词性标注实验，进行性能对比。CTB5.0 语料的划分和统计信息如表 2-6 所示：

表 2-6 CTB5.0 语料划分
Table 2-6 data split of CTB5.0

Data Set	CTB 文件序号	句子数	词语数
Train	1-270,400-931,1001-1151	18089	493939
Dev	301-325	250	6821
Test	271-300	348	8008

然后使用 Pipeline 的方法进行分词、词性标注。具体实验方法为首先分别使用 CTB5.0 语料的训练集训练单独的分词和词性标注模型，然后使用分词模型对测试集进行分词测试，然后使用词性标注模型对测试集的分词结果进行词性标注。实验对比结果如表 2-7 所示：

表 2-7 与 joint 方法性能对比
Table 2-7 performance comparison with joint methods

Method	F-SEG(%)	F-SEG&POS(%)
(Jiang et al., 2008a)[47]	97.85	93.41
(Jiang et al., 2008b)[48]	97.74	93.37
(Kruengkrai et al., 2009)[49]	97.87	93.67
(Zhang and Clark, 2010)[50]	97.78	93.67
(Sun 2012)[51]	**98.17**	**94.02**
Our	97.41	92.12

从表 3-7 对比实验数据可以看出，使用 pipeline 的方法在分词以及词性标注上性能均低于联合模型，但是 pipeline 的方法在实际应用中仍然具有自身的优势。除了具体的需求以外，在联合模型中，需要使用的训练语料必须同时完成分词及词性标注，这对于语料库的构建往往具有挑战性。而在 pipeline 方法中，可以使用两个不同的分词语料和词性标注语料来分别完成分词和词性标注训练。

2.4 感知器算法参数平均化时间比较

2.4.1 感知器算法参数平均时间

在使用感知器算法训练模型的过程中，很容易导致模型对训练数据的过拟合，因此往往在最后求解模型参数时通过对参数进行平均化操作来避免模型对训练语料的过拟合。

在原始的 Structure Perceptron 算法中没有使用参数平均化步骤来防止过拟合。即 Structured Perceptron 算法伪代码如图 2-1 所示：

Training data: $T=\{(x_n,y_n)\}_{n=1}^{T}$
1. $w_0=0;v=0;i=0$
2. for $t:1..T$
3. for $n:1..N$
4. $y'_n=\text{argmax}_{y\in G(x_n)}\ w\bullet\Phi\ (x_n,y)$
5. $w^{(i+1)}=w^i+\Phi(x_n,y_n)-\Phi(x_n,y'_n)$
6. $v=v+w^{(i+1)}$
7. $i=i+1$
8. *return* w

图 2-1 Structured Perceptron 算法伪代码
Fig.2-1 pseudo code of Structured Perceptron

为了防止过拟合，在文献 2 中，在最后对参数权重进行平均化操作，即所谓的 Average Perceptron 算法，伪代码如图 2-2 所示：

Training data: $T=\{(x_n,y_n)\}_{n=1}^{T}$
1. $w_0=0;v=0;i=0$
2. for $t:1..T$
3. for $n:1..N$
4. $y'_n=\text{argmax}_{y\in G(x_n)}\ w\bullet\Phi\ (x_n,y)$
5. $w^{(i+1)}=w^i+\Phi(x_n,y_n)-\Phi(x_n,y'_n)$
6. $v=v+w^{(i+1)}$
7. $i=i+1$
8. $w=v/NT$
9. *return* w

图 2-2 Average Perceptron 伪代码
Fig.2-2 pseudo code of Average Perceptron

在图 2-2 所示的伪代码中，在所有迭代次数结束的时候，对参数进行一次平均化操作。因为在最后一次迭代结束前，模型有可能已经对训练语料过拟合，因此还有一种选择就是在每完成一轮迭代就对参数进行一次平均化操作，然后使用平均化操作得到的模型参数进行后续的迭代。伪代码如图 2-3 所示：

Training data: $T=\{(x_n,y_n)\}_{n=1}^{T}$
1. $w_0=0;v=0;i=0$
2. for $t:1..T$
3. for $n:1..N$
4. $y'_n=\text{argmax}_{y\in G(x_n)}\ w\bullet\Phi\ (x_n,y)$
5. $w^{(i+1)}=w^i+\Phi(x_n,y_n)-\Phi(x_n,y'_n)$
6. $v=v+w^{(i+1)}$
7. $i=i+1$
8. $w=v/Nt$
8. *return* w

图 2-3 Average per-iterator Perceptron 算法伪代码
Fig.2-3 pseudo code of Average per-iterator Perceptron

2.4.2 实验结果

为了验证三种不同的参数平均化时间对模型性能的影响，我们使用 PKU 语料作为分词语料，进行对比实验。开发集上准确率、召回率和 F 值的变化趋势如图 2-4、2-5、2-6 所示：

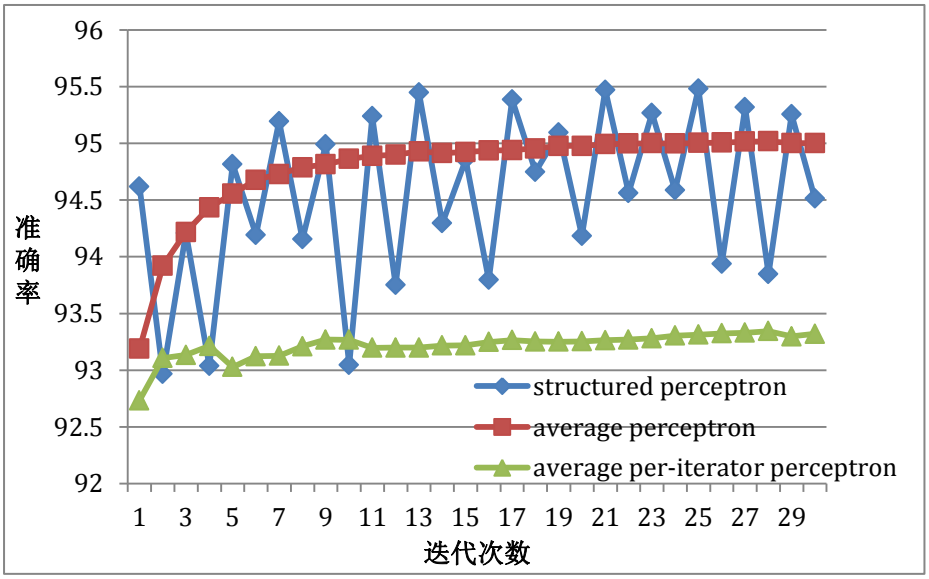


图 2-4 PKU 语料开发集准确率实验结果
Fig.2-4 Development set accuracy of PKU corpus

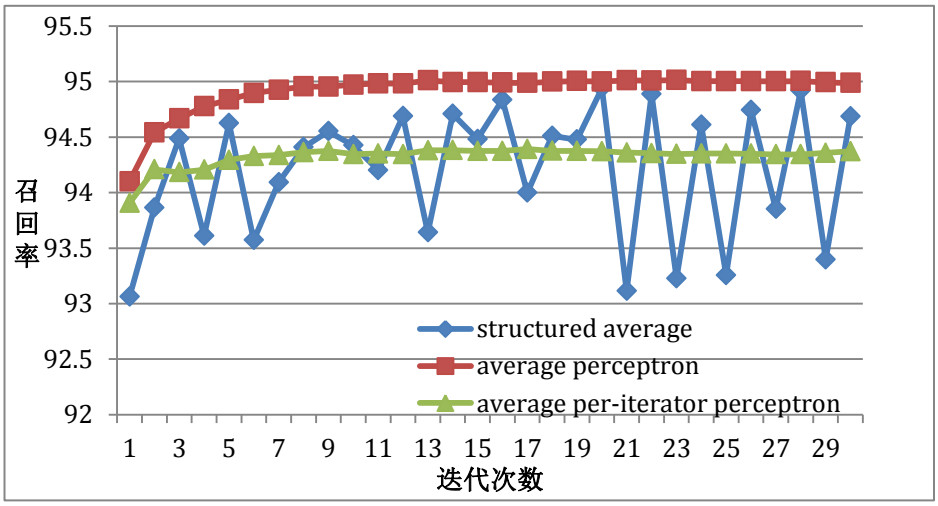


图 2-5 PKU 语料开发集召回率实验结果

Fig.2-5 Development set recall of PKU corpus

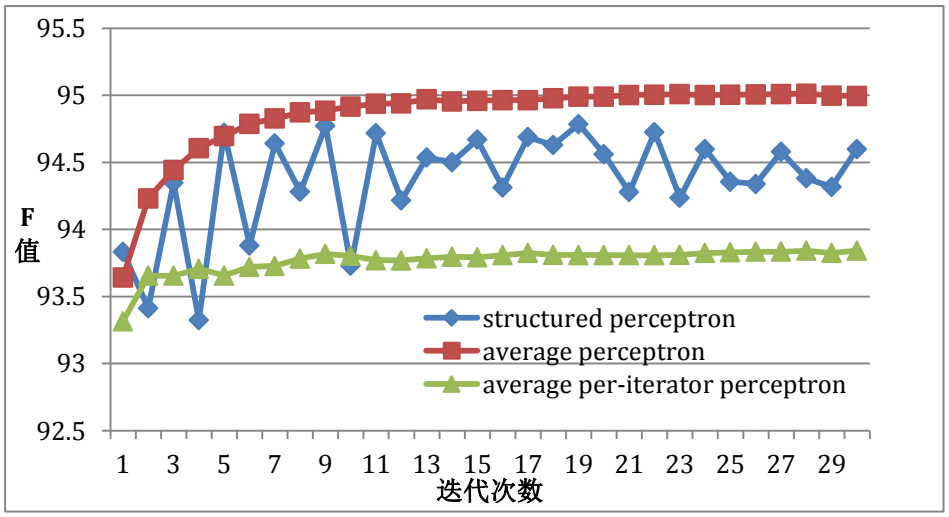


图 2-6 PKU 语料开发集 F 值实验结果

Fig.2-6 Development set F1 of PKU corpus

根据开发集性能选择模型对测试集语料进行测试，实验结果如表 2-8 所示：

表 2-8 三种不同参数方法分词实验结果

Table 2-8 performance comparison using three different methods

Methods	P(%)	R(%)	F(%)
Structured Perceptron	95.10	94.48	94.78
Average Perceptron	**95.02**	**95.00**	**95.01**
Average per-iterator Perceptron	93.31	94.37	93.84

由于人民日报语料规模巨大，进行实验非常耗时，而 CTB5 语料规模太小，实验不具有较大的说服力，因此我们在后续的实验中，使用 CONLL06 语料作为词性标注语料。CONLL06 语料信息如表 2-9 所示：

表 2-9 CONLL06 语料统计信息

Table 2-9 statistic result of CONLL06

Data Set	Sentences	Word Types	Words Number
Train	16090	34576	497913
Dev	803	4537	20454
Test	1910	8577	50319

在 CONLL06 语料上进行词性标注对比实验，三种不同参数处理方法在开发集数据上词性标注准确率对比曲线如图 2-7 所示：

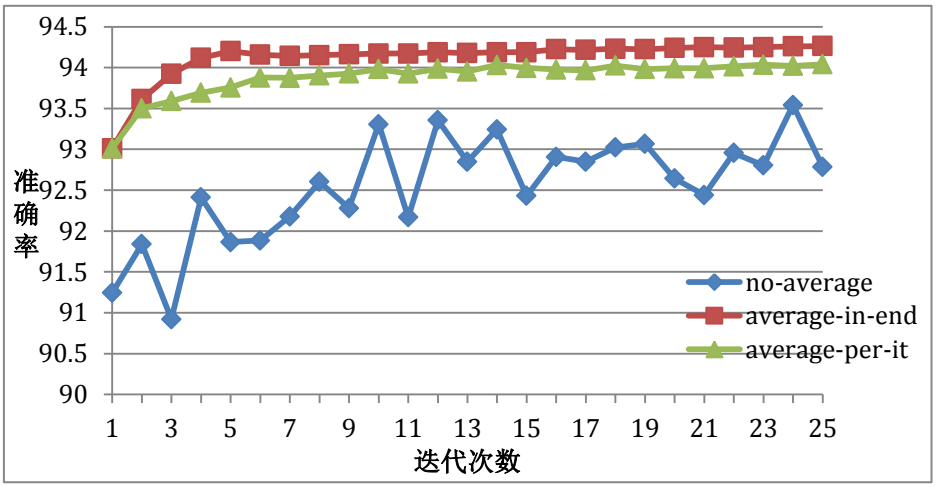


图 2-7 CONLL06 语料开发集准确率实验结果

Fig.2-7 Development set accuracy of CONLL06 corpus

根据开发集性能选择模型对测试集语料进行测试，实验结果如表 2-10 所示：

表 2-10 三种参数处理方法在 CONLL06 测试集性能

Table 2-10 performance comparison using three different methods

Methods	Precision(%)
Structured Perceptron	93.30
Average Perceptron	**93.74**
Average per-iterator Perceptron	93.35

从实验结果可以看出，在分词和词性标注实验中，Structure Perceptron 由于容易对训练语料过拟合，性能波动较大。完成所有迭代次数后对参数进行一次平均化操作性能最优。因此在本文系统中，训练算法使用 Average

Perceptron 算法。

2.5 本章小结

在本章中，我们详细介绍了词典与统计相结合的中文分词、词性标注方法。在中文分词中，通过使用词典，我们不仅提高了分词准确率，而且通过对词典的扩充实现了中文分词的领域自适应。该方法在实际中具有重要的使用价值。通过使用词典，我们提高了中文词性标注的准确率，同时加快了解码速度，提高了效率。

同时我们还比较了感知器算法中参数的三种不同处理方法对性能的影响。Structure perceptron 算法由于容易造成过拟合，因此性能波动较大；完成所有迭代次数后对参数进行一次平均化操作性能在三种方法中最优。

第3章 效率优化及性能提升

虽然基于感知器的序列标注算法在速度上已经优于 ME、CRF 这些复杂模型。但是和所有的结构预测模型一样，即使是在强独立性假设前提下，推导过程的复杂度仍然与序列长度是非线性关系。当训练集规模较大时，感知器算法的训练过程仍然是非常耗时的。

本章，我们首先介绍感知器的分布式并行训练算法。实验结果表明在不显著降低模型性能的前提下，并行训练能大幅度的提高训练效率。考虑到模型特征空间巨大，我们对模型进行了压缩，在不损失性能的前提下，压缩模型文件大小，减小运行时内存空间的需求。然后将我们的系统与现有的主要开源分词、词性标注系统进行比较。最后我们通过对大规模未标注语料进行自动分词然后进行聚类，在词性标注中引入聚类特征来进一步提高词性标注的准确率。

3.1 感知器并行训练算法

Structured Perceptron 分布式并行训练算法[52]由 Ryan McDonald 在 2009 年提出。算法的基本思想是当训练数据规模较大时，将训练数据划分为 S 个不相交的子集，然后在这 S 个不相交子集上并行训练多个子模型，对多个子模型进行融合得到最终的模型。

在文献 52 中提到了两种不同的参数融合方式：直接参数融合以及迭代参数融合。

3.1.1 直接参数融合算法

在直接参数融合方法中，模型参数的融合使用的是非常直接的方式。基本思想是将训练集 T 划分为 S 个不相交的子集，即$T = \{T_1, T_2, \dots, T_s\}$，然后在每一个子集上使用感知器算法训练一个模型，最后对 S 个模型参数权重根据融合系数$\boldsymbol{\mu} = \{\mu_1, \mu_2, \dots, \mu_S\}$进行加权求和得到最终的模型。我们发现直接参数融合方法每一个子模型都在一个不想交的子集上进行训练，可以很容易的在集群系统上实现分布式并行训练或者使用多线程实现并行训练。

直接参数融合算法伪代码如图 3-1 所示：

Training data: $\mu = \{\mu_1,...,\mu_S\}$, $T = \{(x_t, y_t)\}_{t=1}^{|T|}$
1. *Shard* T into S *pieces* $T = \{T_1,...,T_s\}$
2. $w^{(i)} = Perceptron(T_i)$
3. $w = \sum_i \mu_i w^{(i)}$
4. *return* w

图 3-1 感知器直接参数融合并行训练算法伪代码
Fig.3-1 Distributed perceptron using parameter mixing

其中 $\boldsymbol{\mu} = \{\mu_1, \mu_2, \dots, \mu_S\}$ 表示模型融合系数，满足 $\forall \mu_i \in \boldsymbol{\mu}: \mu_i \geq 0 \; and \; \sum_{i=1}^{s} \mu_i = 1$。

有学者对最大熵模型并行训练[53]、CRF 模型并行训练[54]进行了研究。对于最大熵模型，Mann 在 2009 年的文献 53 中证明了使用这种参数融合方式并行训练的模型和直接使用所有训练数据串行训练的模型在性能差异上具有一个确定的界。但是该结论成立的前提是最大熵模型中参数的正则化因子必须具有一个确定的界。而这个条件在感知器算法中是不能满足的。

直接参数融合方法使用并行训练虽然解决训练效率问题，但是并不能保证模型对数据的可分性(separability)不变。即对于一个可分的训练数据，使用直接参数融合并行训练算法并不能保证得到一个对训练数据可分的模型。

定理 1：对于任意一个边缘γ可分的训练数据 T，使用感知器直接参数融合并行训练算法不能保证得到一个对 T 仍然可分的模型。

证明：假设对已一个二元分类问题，类别为$Y = \{0,1\}$。训练集 T 包含 4 个训练实例，将训练集划分为两个不相交的训练子集$T_1 = \{(x_{1,1}, y_{1,1}), (x_{1,2}, y_{1,2})\}$，$T_2 = \{(x_{2,1}, y_{2,1}), (x_{2,2}, y_{2,2})\}$。其中$y_{1,1} = y_{2,1} = 0$，$y_{1,2} = y_{2,2} = 1$，假设特征向量$\boldsymbol{f} \in R^6$，每一个实例的特征向量如图 3-2 所示：

$f(x_{1,1}, 0) = [1,1,0,0,0,0]$	$f(x_{1,1}, 1) = [0,0,0,1,1,0]$
$f(x_{1,2}, 0) = [0,0,1,0,0,0]$	$f(x_{1,2}, 1) = [0,0,0,0,0,1]$
$f(x_{2,1}, 0) = [0,1,1,0,0,0]$	$f(x_{2,1}, 1) = [0,0,0,0,1,1]$
$f(x_{2,2}, 0) = [1,0,0,0,0,0]$	$f(x_{2,2}, 1) = [0,0,0,1,0,0]$

图 3-2 训练实例特征向量
Fig.3-2 feature vector of training instances

假设在分类过程中是类别 1 平局决胜（label 1 tie-break），即当两个类别的分数相同时，认为类别 1 是正确分类结果。

使用感知器直接参数融合并行训练算法进行训练，迭代两次以后参数将保持不变，得到$w^{(1)} = [1,1,0,-1,-1,0]$，$w^{(2)} = [0,1,1,0,-1,-1]$。此时如果设

定μ_1、μ_2非零，则所有的实例将会被分类为 0.如果$\mu_1 = 1$，$\mu_2 = 0$则实例$(x_{2,2}, y_{2,2})$将被错误的分类为 0。如果$\mu_1 = 0$，$\mu_2 = 1$则实例$(x_{1,2}, y_{1,2})$将被错误分类为 0。因此，不论如何设定融合系数$\boldsymbol{\mu}$都无法得到一个对训练集可分的模型。但是使用所有训练数据训练可得到模型$\boldsymbol{w} = [-1,2,-1,1,-2,1]$能够将训练集进行正确划分。

因此感知器直接参数融合并行训练算法不能保证最终模型的性能。Ryan McDonald 对算法进行了相应的改进，得到感知器迭代参数融合并行训练算法。

3.1.2 迭代参数融合算法

在直接参数融合算法中，在参数融合前在多个不相交子集数据上训练得到多个子模型。在迭代参数融合算法中，每一轮迭代结束时，对参数进行融合，然后每个子模型使用融合后更新的参数进行下一次迭代。感知器迭代参数融合并行训练算法伪代码如图 3-3 所示：

PerceptronIterParamMix

Training data: $T = \{(x_t, y_t)\}_{t=1}^{|T|}$

1. *Shard* T into S *pieces* $T = \{T_1, ..., T_s\}$
2. $w = \vec{0}$
3. *for* n : 1...N
4. $\quad w^{(i,n)} = OneEpochPerceptron(T_i, w)$
5. $\quad w = \sum_i \mu_{i,n} w^{(i,n)}$
6. *return* w

图 3-3 感知器迭代参数融合并行训练算法伪代码

Fig.3-3 Distributed perceptron using iterative parameter mixing

其中$w^{(i,n)}$表示第 n 次迭代中在第 i 个子集T_i上训练得到的模型参数，每一个 $w^{(i,n)}$ 之间的训练是并行的。$\boldsymbol{\mu_n} = \{\mu_{1,n}, \dots, \mu_{s,n}\}$，$\forall \mu_{i,n} \in \boldsymbol{\mu_n}: \mu_{i,n} \geq 0 \; and \; \sum_i \mu_{i,n} = 1$ 为模型融合系数。OneEpochPerceptron 表示在训练子集上串行训练模型，OneEpochPerceptron 伪代码如图 3-4 所示：

OneEpoch*Perceptron*(T, w^*)
1. $w^{(0)} = w^*; k = 0$
2. *for* t : 1...$|T|$
3. *let* $y' = \text{argmax}_{y'} w^{(k)} * f(x_t, y')$
4. *if* $y' \neq y_t$
5. $w^{(k+1)} = w^{(k)} + f(x_t, y_t) - f(x_t, y')$
6. $k = k + 1$
7. *return* w^k

图 3-4OneEpochPerceptron 算法伪代码

Fig.3-4 pseudo-code of OneEpochPerceptron

在设置$\boldsymbol{\mu_n} = \{\mu_{1,n}, \ldots, \mu_{s,n}\}$时，一种最简单的方法就是将每一个子模型的权重看作是相等的，即设定$\mu_{i,n} = 1/S$。另一种方式是根据每个模型在迭代过程中预测错误数来设定模型融合权重，基本思想是如果一个模型在当前完成的迭代中预测错误数较多，那么该模型的性能应该提升较大，对应的该模型的融合权重应该较大。即$\mu_{i,n} = k_{i,n}/k_n$，其中$k_{i,n}$表示第 *i* 个模型在第 *n* 次迭代中的预测错误数，$k_n = \sum_i k_{i,n}$。实验结果发现两个不同的权重设定方式性能基本一致，因为在每一轮迭代中，每个子模型预测错误数相差不大。因此在本文的实现中，设定$\mu_{i,n} = 1/S$。

在 OneEpochPerceptron 算法中，有两种参数的处理方法。一种直接使用 Structured Perceptron，即不进行参数的平均化操作；一种是使用 Average Perceptron，即在训练子集上训练模型迭代结束时对参数进行平均化操作。在本文中，同时实现了两个参数处理方式进行对比实验。

3.1.3 实验结果及分析

我们使用 1998 年人民日报 1 月份至 6 月份的语料进行词性标注实验来验证并行训练算法的性能和效率。将人民日报 1 月份语料作为测试集，2 到 6 月份语料按照 8:1 比率划分为训练集和开发集。

在实验过程中，设定训练集划分数目为 10，即将训练语料划分为相同大小不想交的 10 个子集。然后进行四组对比实验：

- Average Perceptron Serial （all data）：所有训练语料作为训练集，使用 Average Perceptron 算法进行串行训练
- Average Perceptron Serial（Sub Sampling）：从训练集中随即抽取十分之一的语料作为训练数据，使用 Average Perceptron 算法进行串行训练。

- Structured Perceptron Parallel（all data）：使用所有训练数据作为训练语料，在每个训练数据子集上训练子模型时使用 Structured Perceptron 算法，不对参数进行平均化操作。
- Average Perceptron Parallel（all data）：使用所用训练数据作为训练语料，在每个训练数据子集上训练子模型时使用 Average Perceptron 算法。

在开发集上实验结果如图 3-5 所示：

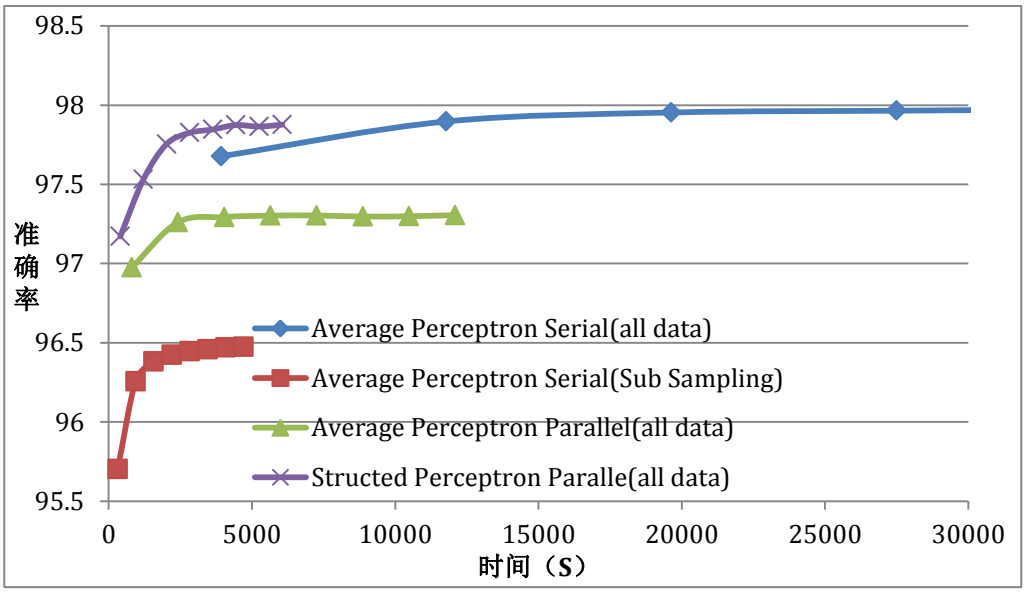


图 3-5 开发集实验结果

Fig.3-5 experimental result of development set

根据开发集性能选择模型对 1 月份语料进行测试，测试结果如表 3-1 所示：

表 3-1 测试集词性标注性能

Table 3-1 pos tagging performance of test set

Training method	Precision(%)
Average Perceptron Serial (all data)	**97.98**
Average Perceptron Serial (Sub Sampling)	96.47
Average Perceptron Parallel（all data）	97.31
Structured Perceptron Parallel（all data）	**97.79**

从图 3-5 以及表 3-1 可以看出，使用迭代参数融合并行训练算法对模型进行训练，大大的减少了模型训练收敛所需要的时间。在迭代训练中，Structured Perceptron 性能不低于 Average Perceptron 性能，其原因是参数融合其实也是一种参数平均化操作，能够防止模型对训练数据的过拟合。因此在后续实验中并行训练直接使用 Structured Percetron 算法。

同时我们发现，在人民日报语料上的实验结果表明随机抽取十分之一的语料使用 Average Perceptron 进行训练，模型的性能很接近使用所有数据训练的

模型性能，其原因是人民日报语料规模很大，即使是使用十分之一的数据仍然能够训练一个足够好的词性标注模型。

为了验证迭代参数融合并行训练算法在小规模语料上的性能，我们在 PKU 语料和 CONLL06 语料分别进行了分词和词性标注对比实验。语料的划分与第三章中的划分一致，参数设置和人民日报实验一致。

在 PKU 语料上进行分词实验，开发集实验结果 F 值如图 3-6 所示：

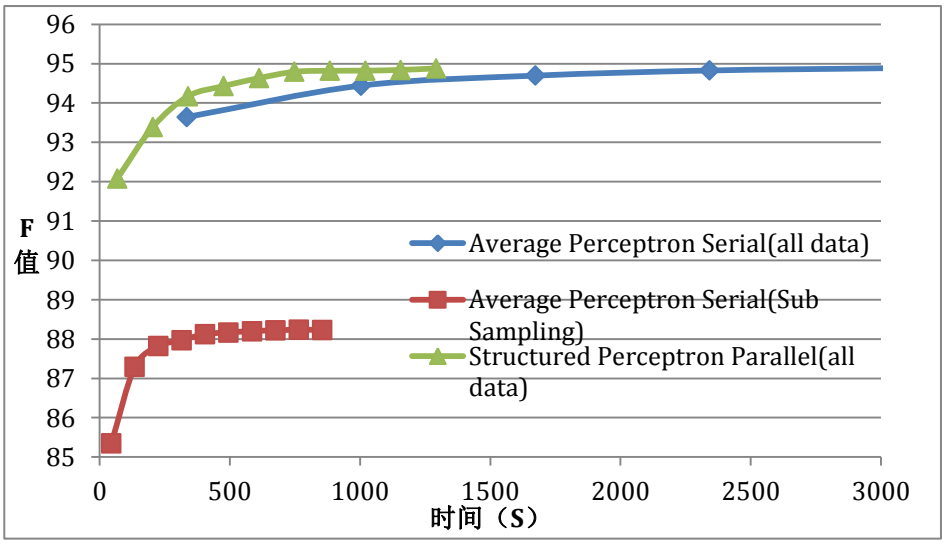


图 3-6 PKU 开发集实验结果

Fig.3-6 development set result of PKU

根据开发集性能选择模型对测试集语料进行测试，实验结果如表 3-2 所示：

表 3-2 PKU 测试集分词性能

Table 3-2 word segmentation performance of PKU development set

Training method	P(%)	R(%)	F(%)
Average Perceptron Serial (All Data)	**95.02**	**95.00**	**95.01**
Average Perceptron Serial((Sub Sampling)	87.62	88.89	88.25
Structured Perceptron Parallel（all data）	94.80	94.83	94.81

在 CONLL06 语料上进行词性标注实验，开发集上实验结果对比如图 3-7 所示：

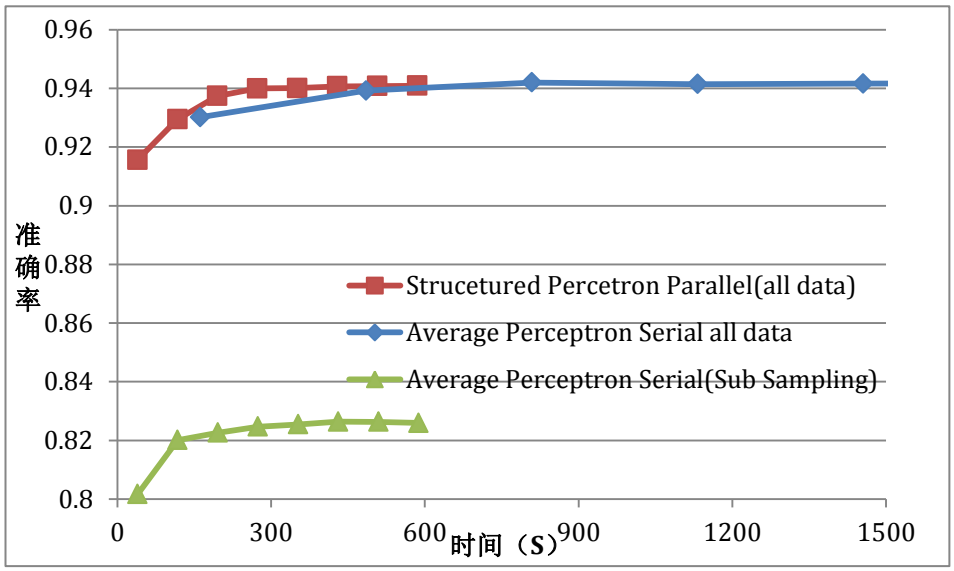


图 3-7 CONLL06 词性标注开发集实验结果

Fig.3-7 development set result of CONLL06

根据开发集性能选择模型对测试集语料进行测试，测试结果如表 3-3 所示：

表 3-3 CONLL06 词性标注测试集结果

Table 3-3 pos tagging performance of CONLL06 development set

Training method	Precision(%)
Average Perceptron Serial (All Data)	93.74
Average Perceptron Serial (Sub Sampling)	84.92
Structured Perceptron Parallel（all data）	93.57

从实验结果可以看出，不论是在人民日报的大规模数据集上，还是在 PKU、CONLL06 这样的小规模数据集上，感知器迭代参数融合并行训练算法不仅能显著的提高训练效率，节约训练所需时间。而且能保证模型的性能不出现显著下降。

3.2 模型压缩

虽然和 CRF、ME 等复杂模型相比较，基于感知器算法的序列标注模型已经相对比较简单。但是在实际应用中，即使训练语料规模不是特别大，根据模版提取的特征数量仍然会到达百万级甚至是千万级之多，消耗大量内存。实际上，模型中存在很大一部分特征的权重很小，对于计算状态序列的分数影响微乎其微，因此可以通过统计特征的权重对模型进行压缩，将对计算分数结果影响特别小的特征从模型中删除。这样在不显著影响性能的前提下既可以减小模型文件的大小还可以降低对内存的需求。定义特征压缩比例为删除的特征数目占总的特征数目的百分比。模型压缩方法为：首先设定压缩比例，然后统计模型的特征权重，根据设定的压缩比例得到阈值，将权重绝对值小于阈值的特征从模型中删除。在模型压缩实验中，以 5%为步长，不断增大压缩比例。分别

使用 PKU 语料和 CONLL06 语料进行分词和词性标注模型压缩实验。在 PKU 语料测试集进行模型压缩实验，实验结果如图 3-8 所示：

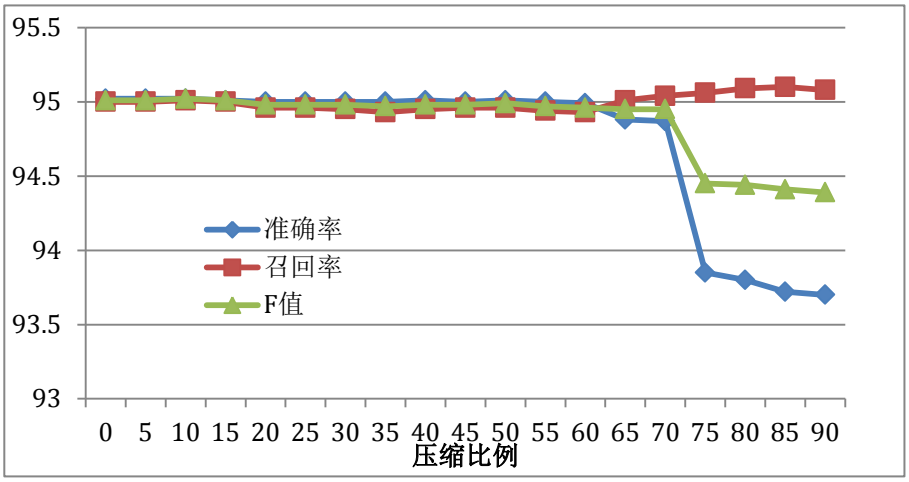


图 3-8 PKU 分词模型压缩实验结果

Fig.3-8 model compression result of PKU word segmentation

分词压缩实验内存和时间结果如表 3-4 所示：

表 3-4 分词内存和时间变化结果

Table 3-4 result of memory and test time of word segmentation

特征压缩比例（%）	最大内存（M）	测试时间（S）
0	610	24
15	572	21
30	531	18
45	502	15
60	478	12
75	451	11
90	442	10

使用 CONLL06 语料进行词性标注模型压缩实验，实验结果如表 3-9 所示：

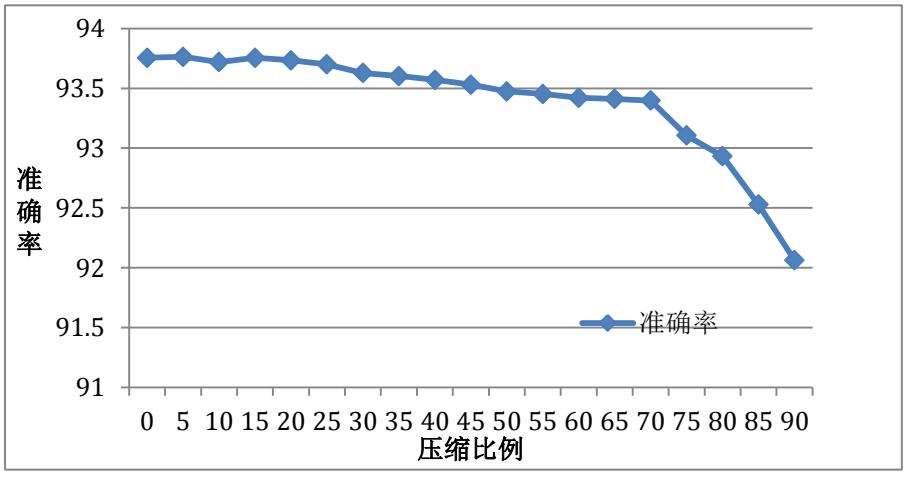


图 3-9 词性标注模型压缩实验结果

Fig. 3-9 model compression result of pos tagging

词性标注模型压缩实验内存和时间变化结果如表 3-5 所示：

表 3-5 词性标注模型压缩内存和时间变化

Table 3-5 result of memory and test time of pos tagging

特征压缩比例（%）	最大内存（M）	测试时间（S）
0	533	18
15	503	17
30	454	16
45	412	15
60	388	14
75	365	13
90	332	13

从实验数据可以看出，分词模型中，当设置特征压缩比率为 50%的时候，分词性能的 F 值变化范围在 0.04%内，当压缩比率设定为 90%时，F 值下降 0.65%；对于词性标注来说，当压缩比率范围在 20%以内，模型的性能变化在 0.02%范围内，当增大压缩比率至 60%时，性能出现了 0.3%的下降，当压缩比率增大到 90%时，性能下降 1.7%，变化显著。

从实验数据可以看出，在设置相同的压缩比率时，分词和词性标注性能的变化差别较大。其主要原因是分词模型不进行压缩时模型特征数为 1939925，而词性标注模型特征数为 1008188。可以看出分词模型的特征数量远远多于词性标注模型的特征数量。因此在设置相同压缩比率时，对词性标注模型的性能影响比分词的模型性能影响要大。

实验结果表明，通过对模型进行压缩，不仅可以减小模型文件本身的大小，还可以在一定程度上减小运行时的内存需求和时间。因此，在训练语料规模、特征数量、内存需求较大时，可以对模型进行压缩，而不会显著的降低模型性能。

3.3 多线程并行测试

随着计算机制造技术的迅速发展，多核处理器已经成为计算机的标准配置。即使是个人计算机，目前也大部分是双核甚至是 4 核的。而在进行分词、词性标注测试时，只需要共享同一个模型即可，完全可以实现对文件中的多个句子的多线程并行解码。

在本系统中，实现了对文件进行分词、词性标注的多线程解码，这样可以大大加快测试速度。

在测试多线程对测试速度影响时，仍然使用 PKU 语料作为分词语料，使用 Conll06 语料作为词性标注语料。

分词多线程对比实验结果如表 3-6 所示：

表 3-6 分词多线程实验结果

Table3-6 Multithreading experimental word segmentation results

线程设置	资源装载时间（S）	文本分词时间（S）
单线程	5	30
5 线程	5	10
10 线程	5	5

词性标注多线程对比实验结果如表 3-7 所示：

表 3-7 词性标注多线程实验结果

Table3-7 Multithreading experimental pos tagging results

线程设置	资源装载时间（S）	词性标注时间（S）
单线程	3	15
5 线程	3	6
10 线程	3	3

从实验数据可以看出，增加多线程功能以后，对文本的处理时间大大的缩短了，加快了处理的速度

3.4 与主要开源系统比较

目前使用比较广泛的开源中文分词、词性标注系统主要有 FudanNLP 以及 Stanford NLPTool。FudanNLP 基于 CRF 模型进行分词、词性标注，直接使用系统提供的默认特征模版。Stanford NLPTool 分词训练比较复杂，而且目前提供的训练工具只适用于 CTB 格式的语料，直接使用相关论文中的实验结果进行分词性能比较。因此对于分词不进行训练效率的比较。

使用 the second International Chinese Word Segmentation Bakeoff 语料以及 CONLL06 语料作为分词和词性标注语料对三个系统进行性能和效率比较。

在中文分词中，对比实验结果如表 3-8 所示：

表 3-8 分词性能对比结果

Table 3-8 performance comparison of word segmentation

Corpus	Method	P(%)	R(%)	F(%)
MSR	FudanNLP	95.42	95.68	95.55
	Stanford NLPTool	96.60	96.20	96.40
	Our	**96.70**	**97.30**	**97.00**
CU	FudanNLP	93.68	93.47	93.57
	Stanford NLPTool	**94.60**	94.10	94.30
	Our	94.50	**95.20**	**94.90**
PKU	FudanNLP	88.67	89.35	89.01
	Stanford NLPTool	95.40	94.60	95.00
	Our	95.02	95.00	95.01

使用 CONLL06 语料进行词性标注对比实验，实验结果如表 3-9 所示：

表 3-9 词性标注性能对比结果

Table 3-9 performance comparison of pos tagging

Corpus	Method	Precision(%)
Conll6 Dev	FudanNLP	91.61
	Stanford NLPTool	93.90
	Our	**94.32**
Conll6 Test	FudanNLP	90.87
	Stanford NLPTool	93.53
	Our	**93.74**

在 CONLL06 语料上训练时间、测试时间及最大内存对比结果如表 3-10 所示：

表 3-10 时间和内存比较

Table 3-10 comparison of time and memory

Method	**Training time(S)**	**Test time(S)**	**Memory(M)**
FudanNLP	198	5	425
Stanford NLPTool	854	7	500
Our	12565	18	553

其中我们的系统设定迭代次数为 30 次，Average Perceptron 算法在实际应用中迭代次数一般超过 12 次以后性能就趋于稳定，本系统还可使用多线程进行并行训练，可以大大的减少训练所需时间。从实验数据也可以看出，另外两个系统在测试速度和内存消耗上优于本系统，特别是 FundanNLP 训练非常快。其主要原因在于 FundanNLP 只使用了简单的特征，模型特征空间较小，解码时特征提取所需时间也较小。同时本系统实现了多线程解码，可以大幅度的加快文本测试速度，在一定程度上克服了本系统速度上的缺陷。

3.5 未标注数据在词性标注中的使用

在有指导的方法中，训练语料规模的大小和语料内容的覆盖面往往会影响到模型的最终性能。目前，利用大规模的未标注语料的半指导方法已经成为自然语言处理领域的一个研究热点。

在本文中，我们首先对大规模未标注数据进行自动分词，然后使用聚类工具对分词结果进行自动聚类，最后提取词语的聚类信息作为词性标注的特征，来提升词性标注性能。

3.5.1 未标注语料

在本文中使用的未标注语料为新华日报 2000 年至 2010 年的数据，使用 Stanford NLPTool 工具进行自动分词，然后使用 Brown 聚类算法[55]进行聚类。聚类信息如表 3-11 所示：

表 3-11 新华日报语料聚类信息

Table 3-11 statistic result of Xinhua corpus

语料	Words Number	Cluster Number
新华日报	1414685	1000

3.5.2 聚类特征

在进行词性标注时，增加的聚类特征为：

- bigram 聚类特征：$c_{-1}c_0$；c_0c_1；$c_{-1}c_1$
- trigram 聚类特征：$c_{-1}c_0c_1$；$sc_{-1}sc_0sc_1$

其中c表示词语的聚类标号（如 001010），下标表示相对于当前考虑词语的相对位置。sc表示词语聚类标号的长度为 6 的前缀。对于未出现在聚类文件中的词语，聚类标号为“NULL”。

3.5.3 实验结果

我们使用 CONLL06 语料进行实验，验证聚类特征对词性标注性能的影响。在 CONLL06 语料上，开发集对比实验结果如图 3-10 所示：

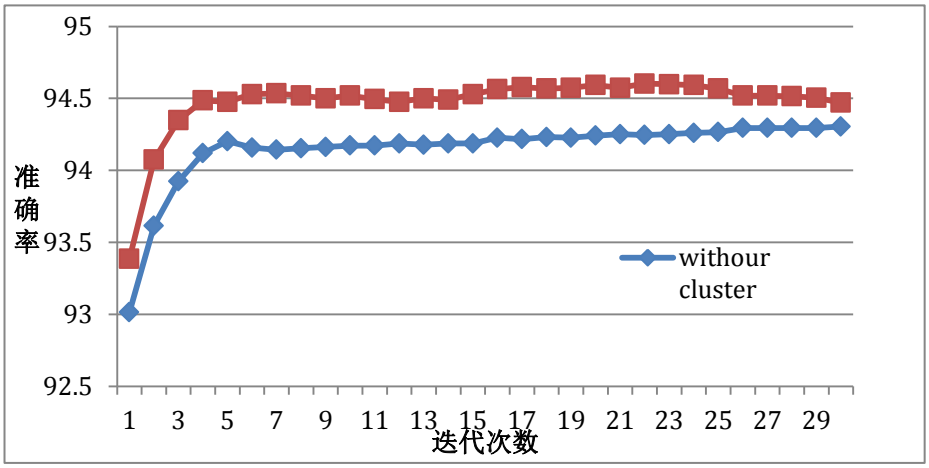


图 3-10 CONLL06 开发集实验对比结果

Fig.3-10 development set result of CONLL06

根据开发集性能选择模型对测试集进行测试，实验结果如表 3-12 所示：

表 3-12 CONLL06 实验结果

Table 3-12 perfprmance of CONLL06

Corpus	P without cluster(%)	P with cluster(%)
Dev	94.32	**94.60**
Test	93.74	**94.13**

从实验结果结果可以看出，使用聚类特征以后，开发集和测试集上性能分别提升了 0.28%和 0.39%。但是由于需要建立一个大规模的聚类词表，频繁的进行查询，导致效率的下降。因此在实际使用中，需要在性能和效率之间进

行平衡。

3.6 本章小结

本章首先详细的介绍了两种感知器并行训练算法的参数融合方式，并与串行训练方法进行了效率和性能的比较。实验结果表明不论是在大规模数据集上，还是现在小规模数据集上，迭代参数并行训练算法在不损失较大性能的前提下能显著的提高训练速度。然后我们介绍了模型压缩方法，通过对模型中的特征数量进行压缩，在不损失性能前提下能够减小内存需求以及提高速度。然后我们将本系统与现有的主要开源系统进行比较，实验结果表明我们的系统在所有测试数据上性能均优于其他系统。最后我们介绍了大规模未标注数据在词性标注中的使用，通过对大规模未标注数据进行自动分词、聚类操作，然后在词性标注中添加词语的聚类特征能够提高中文词性标注性能。

第4章 基于感知器算法的模型增量训练

感知器算法属于在线算法，在线算法的特点是更新参数时不需要一次使用所有训练数据，而是每次使用一个训练实例对参数进行更新，因此在面对大规模训练数据时在线算法有巨大的优势。

在实际应用中我们往往面临以下两种情形：

（1）初始时，可用训练语料规模比较小，只能使用小规模语料训练模型。后期得到更多的可用训练语料。由于版权等原因我们无法得到初始训练语料，而只能使用初始语料训练得到的原始模型。此时如何使用这些新增训练语料和原始模型重新进行训练来得到一个性能更好的模型。

（2）领域移植问题是中文分词的一个难点。当初始训练语料和需要处理的目标文本属于两个不同领域时，使用初始训练语料训练的模型在目标文本上分词性能往往差强人意。在标注一部分目标文本领域分词语料后，如何使用初始模型和目标领域分词语料重新训练得到一个对目标领域性能更好的模型。

在情形一中，如果能够直接使用原始模型和新增语料进行增量训练得到一个性能更优的模型，那么不仅可以避免语料版权相关问题，还可以避免使用所有语料重新训练模型所需要的时间。在情形二中，如果能够通过使用原始领域的模型结合目标领域的语料进行增量训练得到一个对于目标领域分词性能较好的模型，不仅可以避免大规模的重新标注目标领域的分词文本，还能够在一定程度上节约训练所需要的时间。

本章我们首先介绍在相同领域的文本中基于感知器的模型增量训练方法，通过实验验证该方法的有效性。然后根据实验中跨领域分词模型增量训练出现的问题，我们将 Stacked Learning 方法应用到跨领域分词中，来提高分词性能。

4.1 基于感知器的模型增量训练方法

在增量训练中，首先使用初始训练语料训练一个初始模型，然后结合初始模型以及增量语料进行增量训练得到一个增量模型，使增量模型的性能优于初始模型。同时避免了对初始语料的需求以及使用全部语料训练模型所需要的时间。

模型增量训练流程图如图 4-1 所示：

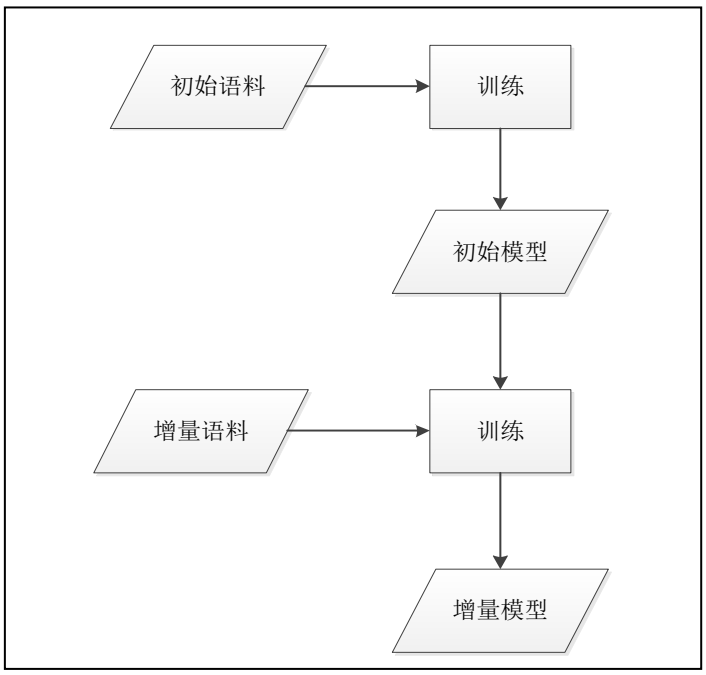


图 4-1 模型增量训练流程图

Fig.4-1 procedureof incremental training

4.1.1 模型增量训练方法

根据感知器并行训练的思想，我们首先使用增量语料训练一个子模型M_1，然后将子模型和初始模型M_2进行融合得到最终模型M。在对子模型和初始模型进行融合时，我们使用了三种不同的参数融合方法：

- method1：参数融合方法如公式（4-1）所示：

$$w_i = \frac{w_{1,i}+w_{2,i}}{2} \tag{4-1}$$

其中w_i是模型M中第 i 个特征f_i的权重，$w_{1,i}$是特征f_i在模型M_1中的权重，$w_{2,i}$是特征f_i在模型M_2中的权重。

- method2：参数融合方法如公式（4-2）所示：

$$w_i = \begin{cases} \frac{w_{1,i}+w_{2,i}}{2} \ \ if\ f_i\ in\ M_1\ and\ f_i\ in\ M_2 \\ w_{1,i} \ \ if\ f_i\ in\ M_1\ and\ f_i\ \ not\ in\ \ M_2 \\ w_{2,i} \ \ if\ f_i\ in\ M_2\ and\ f_i\ \ not\ in\ \ M_1 \end{cases} \tag{4-2}$$

- method3：参数融合方法如公式（4-3）所示：

$$w_i = \begin{cases} w_{1,i} \times p_1 + w_{2,i} \times p_2 \ \ if\ f_i\ in\ M_1\ and\ f_i\ in\ M_2 \\ w_{1,i} \ \ if\ f_i\ in\ M_1\ and\ f_i\ \ not\ in\ \ M_2 \\ w_{2,i} \ \ if\ f_i\ in\ M_2\ and\ f_i\ \ not\ in\ \ M_1 \end{cases} \tag{4-3}$$

其中p_1表示模型M_1在开发集上的性能，p_2表示模型M_2在开发集上的性能。

4.1.2 实验结果及分析

为了验证增量训练方法的有效性，我们分别使用 CONLL06 语料和 PKU 语料在词性标注、分词两个不用的任务上进行相同领域的增量训练。使用 PKU 语料和金融语料在分词任务上进行跨领域增量训练。

（1） 词性标注实验

CONLL06 语料训练集共有 16000 句，我们抽取前 8000 句作为初始训练语料训练初始模型，然后以 2000 句为步长不断增加增量训练语料的规模进行模型增量训练。同时使用 CONLL06 语料开发集数据选择模型对测试集数据进行词性标注性能评价。

当增量训练数据规模为 2000 句时，开发集实验结果对比如图 4-2 所示：

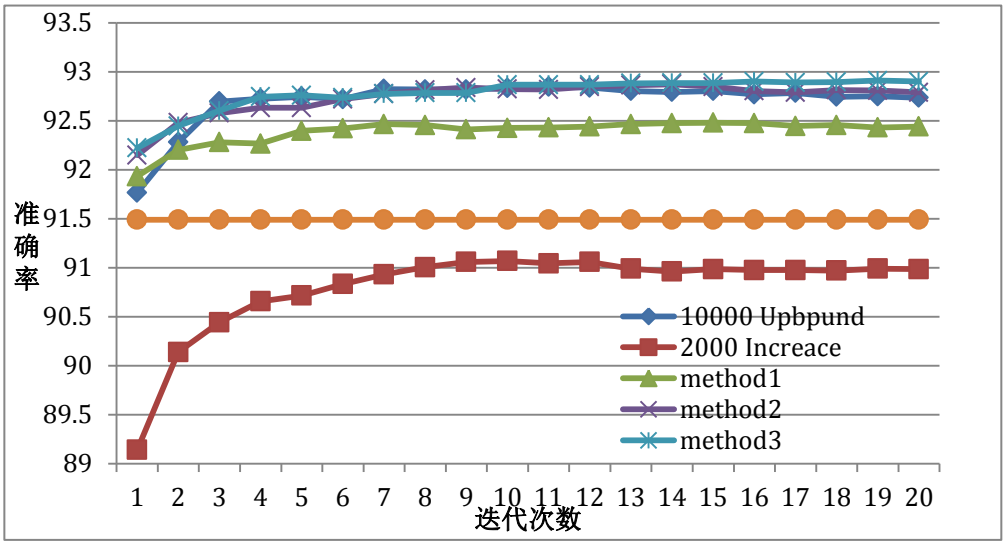


图 4-2 2000 句增量训练开发集实验结果

Fig.4-2 development set result with 2000 incremental instances

其中 10000 Upbound 曲线表示的是使用初始语料以及增量语料直接训练的模型在开发集上的性能；2000 Increace 曲线表示使用 2000 句增量语料训练的模型在开发集上的性能，训练和测试时使用的词典为 8000 句初始语料中抽取的词典和 2000 句增量语料中抽取的词典之和；8000 BaseModel 曲线表示使用 8000 句初始语料训练的初始模型在开发集上的性能。method1、method2、method3 三条曲线分别表示使用三种不用的增量训练方法得到的模型在开发集上的性能。

根据开发集性能选择模型对测试集语料进行测试，实验结果如表 4-1 所示：

表 4-1 2000 句增量训练测试集实验结果

Table 4-1 perfprmance of test set with 2000 incremental instances

Method	Precision(%)
8000 BaseModel（8000 dic）	90.87
8000 BaseModel（10000 dic）	91.01
10000 Upbound	**92.18**
method1	91.73
method2	92.05
method3	**92.15**

其中 8000 BaseModel（8000 dic）表示使用 8000 句初始语料训练模型，测试时仍然使用 8000 句初始语料中提取的词典。8000 BaseModel（10000 dic）表示使用 8000 句初始语料训练模型，测试时词典为 8000 句初始语料中提取的词典以及 2000 句增量语料中提取的词典之和。

将增量语料规模增加到 4000 句，开发集数据上实验结果对比如图 4-3 所示：

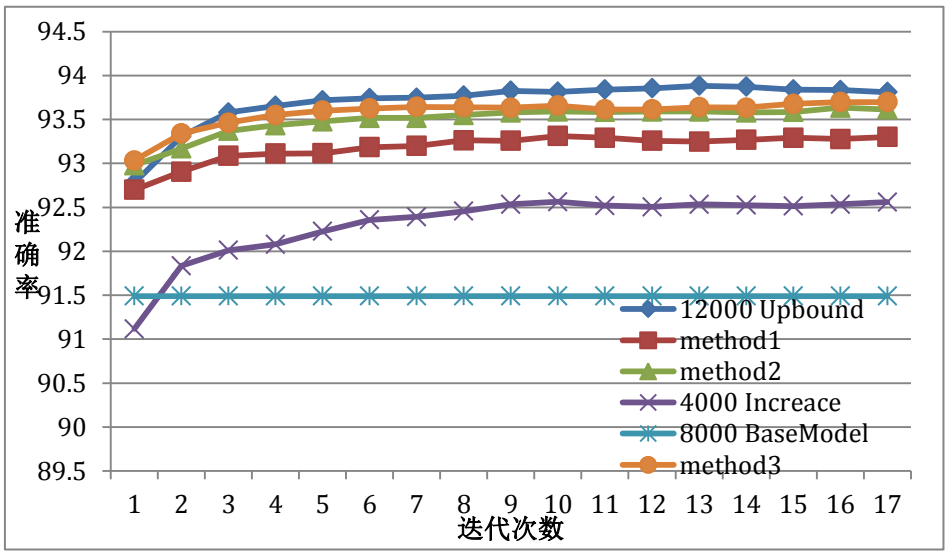


图 4-3 4000 句增量训练开发集实验结果对比图

Fig.4-3 development set result with 4000 incremental instances

根据开发集性能选择模型对测试集语料进行测试，实验结果如表 4-2 所示：

表 4-2 4000 句增量训练测试集实验结果

Table 4-2 perfprmance of test set with 4000 incremental instances

Method	Precision(%)
8000 BaseModel（8000 dic）	90.87
8000 Basemodel（12000 dic）	90.92
12000 Upbound	**93.04**
Method1	92.44
Method2	92.79
Method3	**92.83**

其中各方法表示的含义与表 4-1 中的方法相似。

将增量语料规模扩大到 8000 句，开发集数据上实验结果对比如图 4-4 所示：

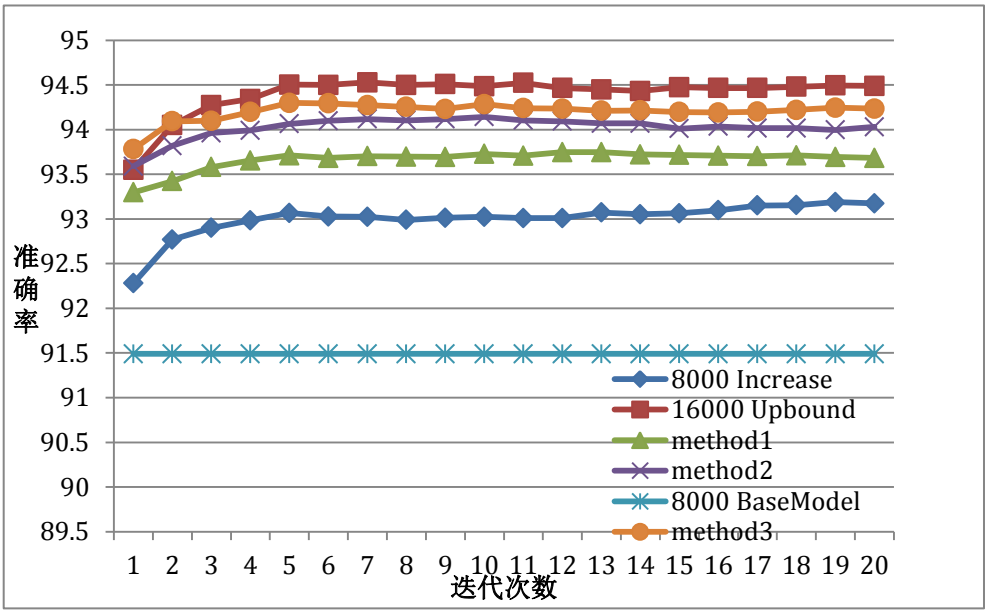


图 4-4 8000 句增量训练开发集实验结果对比图

Fig.4-4 development set result with 8000 incremental instances

根据开发集性能选择模型对测试集语料进行测试，实验结果如表 4-3 所示：

表 4-3 8000 句增量训练测试集实验结果

Table 4-3 perfprmance of test set with 8000 incremental instances

Method	Precision(%)
8000 BaseModel（8000 dic）	90.87
8000 Basemodel（16000 dic）	90.72
16000 Upbound	**94.09**
Method1	93.24
Method2	93.76
Method3	**93.89**

其中各方法表示的含义与表 4-1 中的方法相似。

根据实验结果我们发现通过使用增量训练方法能显著的提高模型的词性标注性能。在三种增量训练方法中，method2 和 method3 性能优于 method1，method2 和 method3 性能非常接近，原因在于初始模型和使用增量语料训练的子模型在开发集上的性能差别不大，因此 method2 和 method3 两种模型融合方法没有显著的差别。同时我们发现使用初始模型，同时仅仅通过增量语料对词典进行扩充并不能显著提高模型性能。因此增量训练方法是有意义而且效果显著的。

（2） 相同领域分词实验

由于 PKU 语料没有划分开发集，因此从 19056 句训练语料中选取前 10000 句训练初始模型，选取 10001 至 17000 句作为增量训练语料，17001 至

19056 句作为开发集，测试集为 PKU 语料的测试集。

分别使用三种不同的增量训练方法，在开发集数据上准确率、召回率以及 F 值实验结果对比如图 4-5、4-6、4-7 所示：

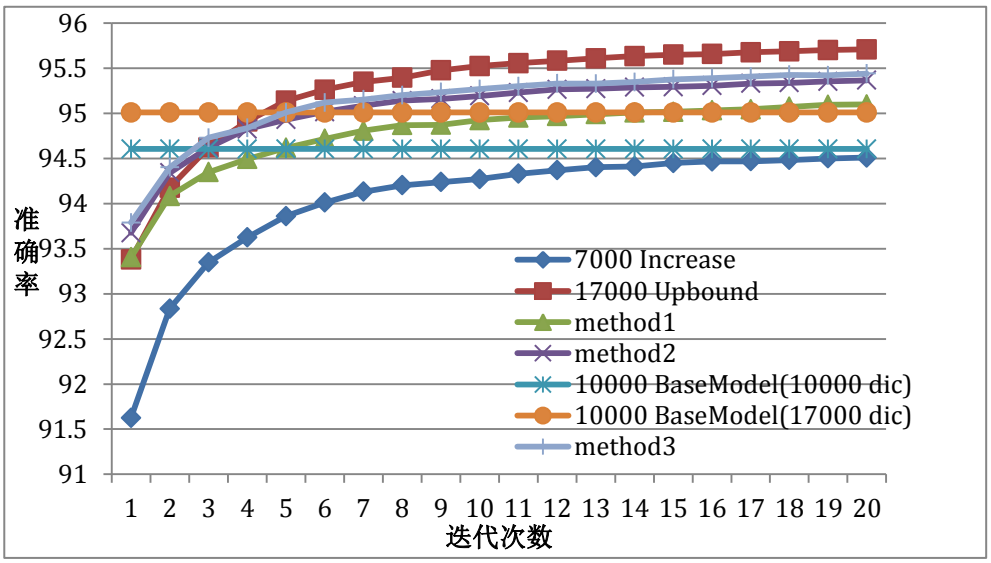


图 4-5 PKU 开发集准确率实验结果

Fig.4-5 Development set accuracy of PKU corpus

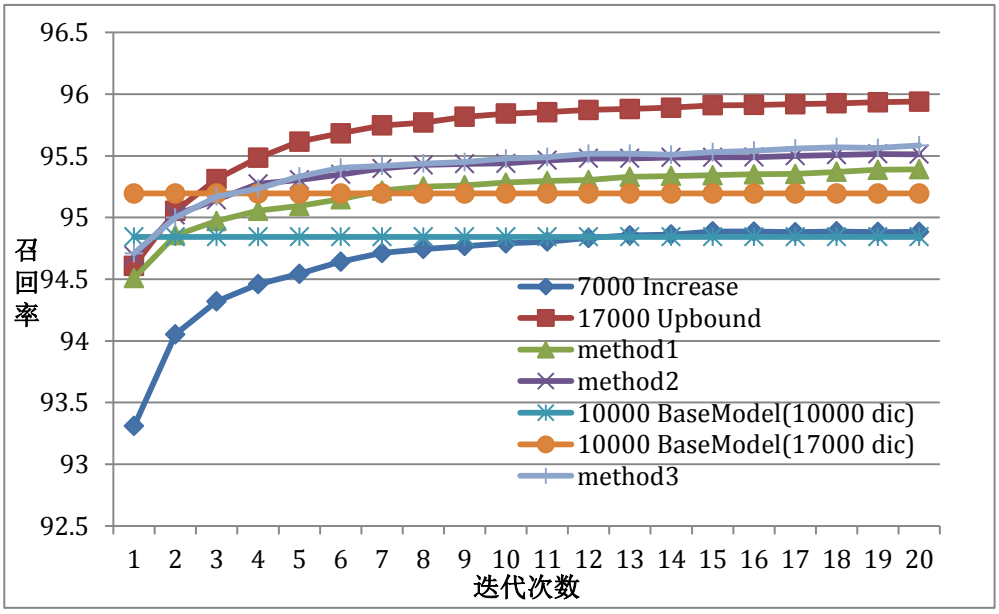


图 4-6 PKU 开发集召回率实验结果

Fig.4-6 Development set recall of PKU corpus

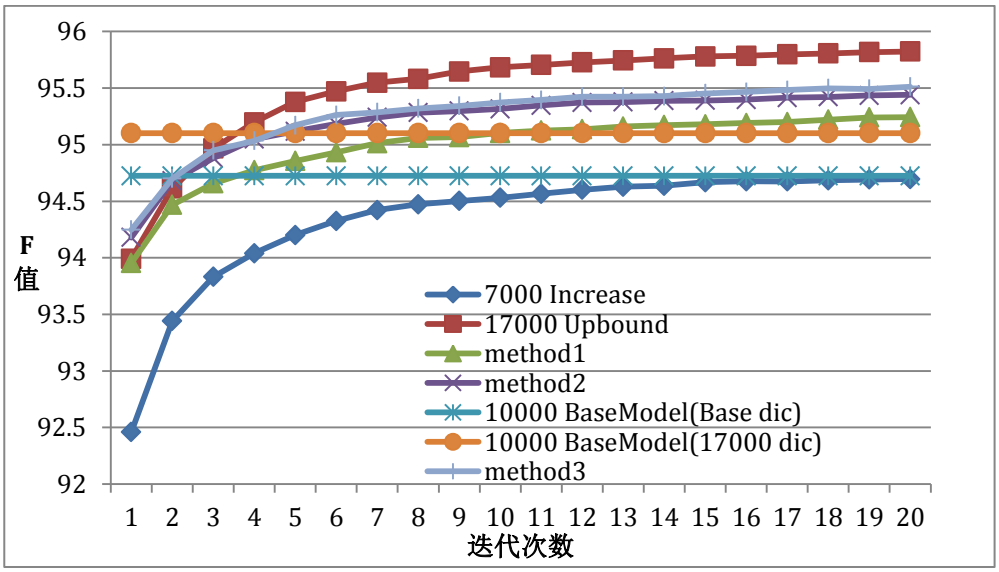


图 4-7 PKU 开发集 F 值实验结果

Fig.4-7 Development set F1 of PKU corpus

根据开发集性能选择模型对测试集数据进行测试，实验结果如表 4-4 所示：

表 4-4 PKU 测试集实验结果

Table 4-4 perfprmance in test set of PKU

Method	P(%)	R(%)	F(%)
10000 BaseModel（10000 dic）	94.15	94.16	94.15
10000 BaseModel（17000 dic）	94.42	94.46	94.44
17000 Upbound	**95.00**	**94.98**	**94.99**
Method1	94.26	94.42	94.36
Method2	94.53	94.62	94.57
Method3	**94.59**	**94.63**	**94.61**

从实验结果可以看出增量训练对于提升模型性能还是比较显著的，method3 和 method2 方法性能接近，优于 method1 方法。而且仅仅使用初始模型，通过增量语料来扩充词典得到的性能仍然低于增量训练的性能。所以增量训练在中文分词上也是必要且有效的。

（3） 跨领域分词实验

领域自适应问题仍然是中文分词的一个难点和研究热点。前面我们通过实验验证了增量训练方法在相同领域数据上的有效性，接下来我们将通过实验验证该方法在跨领域数据上是否适用。

在跨领域的增量训练中，初始训练语料和增量训练语料属于不同的领域，增量训练语料和待分词的目标文本属于相同领域，直接使用初始语料训练的模型对目标文本进行处理将会得到非常差的分词结果。此时我们有两种不同的处理方法：

- 直接使用增量语料训练一个目标领域的分词模型对目标文本进行分词。
- 在初始模型基础上，使用增量训练方法以及增量语料训练得到一个新

模型，然后使用新模型对目标领域文本进行分词。

本文使用 PKU 训练语料中的 10000 句作为初始语料训练初始模型，然后使用金融语料作为目标领域语料，验证两种不同方法在跨领域分词中的性能。金融领域共 58343 句，将 10001 至 40000 句作为测试语料，40001 至 48343 作为增量训练的开发集语料。剩下的前 10000 句作为增量数据集，以 5000 句为步长，进行两组对比实验。

当增量训练语料规模为 5000 句时，在金融语料开发集上准确率、召回率以及 F 值性能如图 4-8、4-9、4-10 所示：

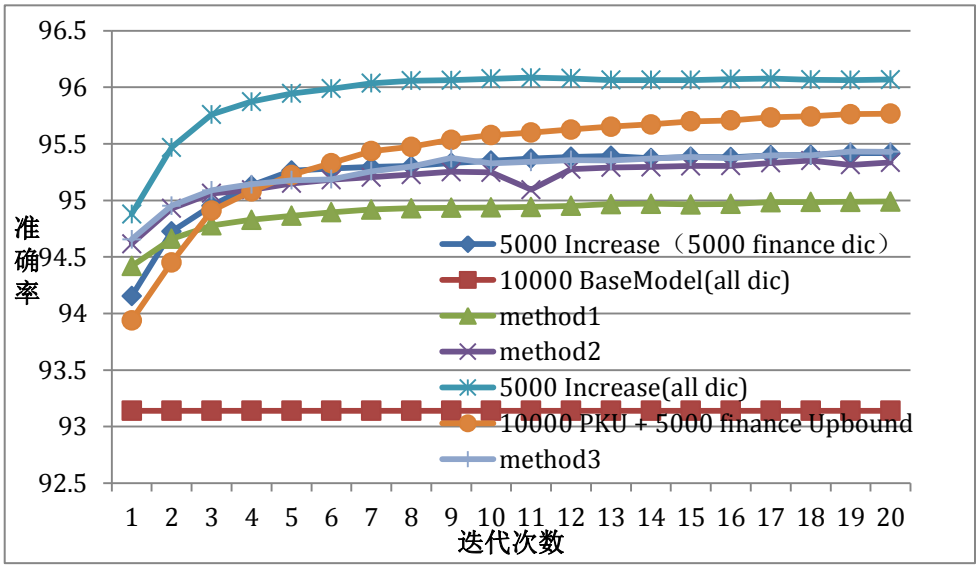


图 4-8 5000 增量语料开发集准确率实验结果

Fig.4-8 Development set accuracy with 5000 incremental instances

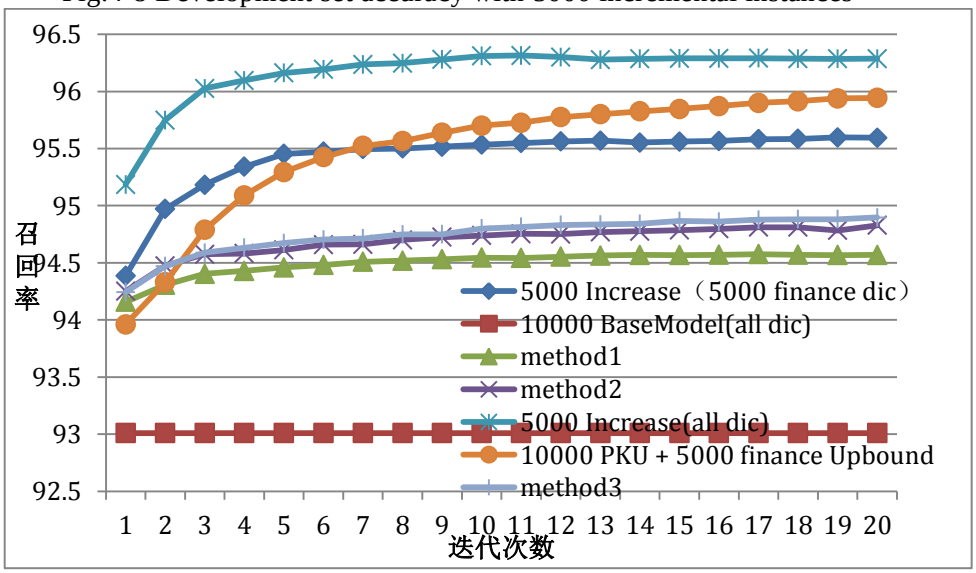


图 4-9 5000 增量语料开发集召回率实验结果

Fig.4-9 Development set recall with 5000 incremental instances

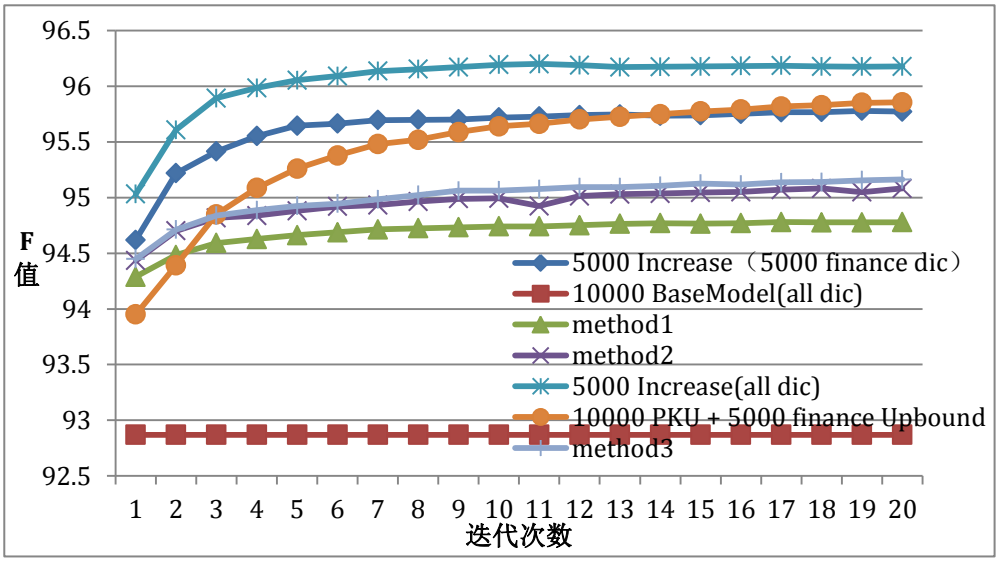


图 4-10 5000 增量语料开发集 F 值实验结果

Fig.4-10 Development set F1 with 5000 incremental instances

其中 5000 Increase（5000 finance dic）表示直接使用 5000 句增量语料训练的模型，训练和测试时词典为增量语料中提取的词典；10000 BaseModel（all dic）表示直接使用 PKU 初始模型对开发集语料进行分词，使用的词典为 PKU 词典以及增量语料中提取词典之和；5000 Increase（all dic）表示直接使用 5000 句增量语料训练的模型，训练和测试时使用的词典为 PKU 词典以及增量语料中提取词典之和；10000PKU+5000 finance Upbound 表示直接使用 10000 句 PKU 初始语料和 5000 金融语料直接训练的模型。

根据开发集性能选择模型对测试集进行测试，实验结果如表 4-5 所示：

表 4-5 5000 增量语料测试集实验结果

Table 4-5 perfprmance in test set with 5000 incremental instance

Method	P(%)	R(%)	F(%)
10000 BaseModel(Base dic)	86.56	88.95	87.74
10000 BaseModel(all dic)	93.14	92.87	93.01
10000 PKU+5000 finance Upbound	95.61	95.78	95.69
5000 Increase(all dic)	**96.01**	**96.18**	**96.09**
5000 Increase (5000 finance dic)	95.12	95.25	95.18
Method1	94.90	94.54	94.72
Method2	95.25	94.81	95.03
Method3	**95.29**	**94.88**	**95.08**

从实验结果可以看出，当初始语料和增量语料属于不同领域时，使用增量训练方法得到的模型在目标领域上并不能得到好的分词结果，分词性能甚至低于直接使用增量语料训练的模型。同时我们也发现，通过扩充词典，可以提高模型的性能。当增量语料规模增加到 10000 句时，开发集上准确率、召回率和 F 值性能如图 4-11、4-12、4-13 所示：

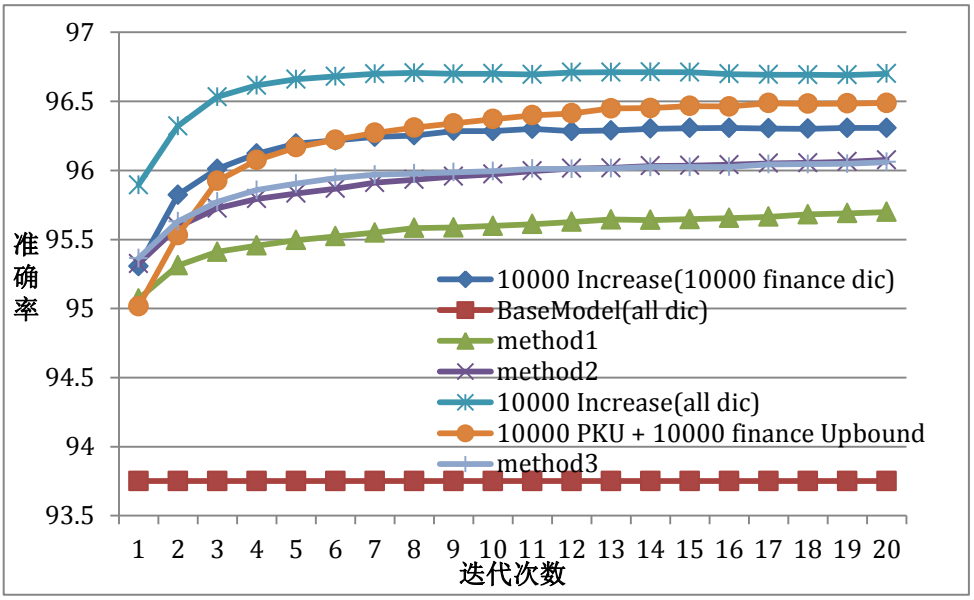


图 4-11 10000 增量语料开发集准确率实验结果

Fig.4-11 Development set accuracy with 10000 incremental instances

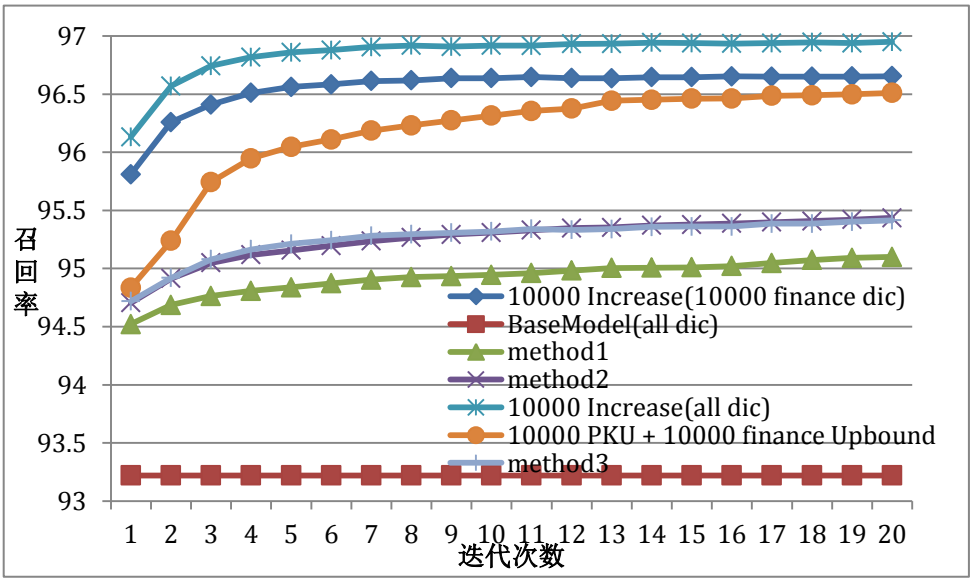


图 4-12 10000 增量语料开发集召回率实验结果

Fig.4-12 Development set recall with 10000 incremental instances

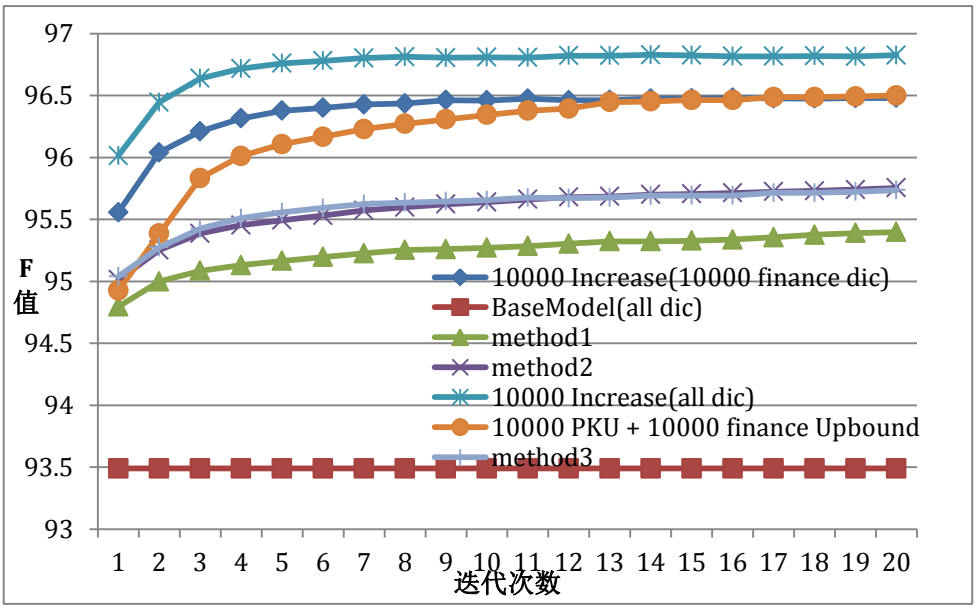


图 4-13 10000 增量语料开发集 F 值实验结果

Fig.4-13 Development set F1 with 10000 incremental instances

根据开发集性能选择模型对测试集语料进行测试，实验结果如表 4-6 所示：

表 4-6 10000 增量语料测试集实验结果

Table 4-6 perfprmance in test set with 10000 incremental instances

Method	P(%)	R(%)	F(%)
10000 BaseModel(Base dic)	86.56	88.95	87.74
10000 BaseModel(all dic)	93.75	93.23	93.49
10000 PKU+10000 finance Upbound	96.48	96.48	96.48
10000 Increase(all dic)	**96.70**	**96.90**	**96.80**
Method1	95.69	95.13	95.41
Method2	96.06	95.46	95.76
Method3	**96.08**	**95.49**	**95.78**

从实验结果可以看出，当增量语料规模增加到 10000 句和初始语料规模相等时，实验结论仍然和增量语料规模为 5000 一致。这说明在跨领域分词中，增量训练方法并不能得到较好的实验结果。

在跨领域分词实验中，增量训练没有得到较好的实验结果，其原因在于初始 PKU 语料和增量金融语料属于完全不同的两个领域，两个数据之间的分词风格差异较大，这样对两个不同领域、不同风格的模型进行融合得到的模型是两者的折中，因此在性能上并不能达到最好。

4.2 基于 Stacked Learning 框架的跨领域中文分词方法

在跨领域分词中增量训练并不能得到较好的性能，一个原因是初始语料和增量语料分词风格上存在较大的差异。比如在初始语料中“农业银行”分为“农业”和“银行”两个词语，而在增量语料中则作为一个词语。对于分词风格存在

差异的情形，如果能使用机器学习算法学习到两种不同风格之间的对应关系，则可以根据对应关系来提高最终分词结果。因此我们将 Stacked Learning 框架引入到跨领域分词中。

4.2.1 基于 Stacked Learning 的跨领域分词方法

Stacked Learning 是一个学习框架，2005 年 Cohen 首次将其应用到序列标注任务[56]中并取得了较好的性能。Stacked Learning 已经被成功的应用在如命名实体识别[57]和依存句法分析[58]等多种 NLP 任务中。

通常来说，Stacked Learning 框架由两层组成。第一层由一个或多个模型$g_1, g_2, \ldots, g_k$组成。对于每一个输入 x，每个模型给出一个输出$g_i(x)$。输入和第一层的各个预测函数的输出组成一个向量$< x, g_1(x), \ldots, g_k(x) >$，作为第二层的预测函数的输入。第二层由一个预测函数 h 组成，h 接受第一层的输出并预测 x 的输出。

在基于 Stacked Learning 框架的跨领域中文分词方法中，第一层由初始语料训练的初始模型组成。对于目标领域的每一个句子，首先使用第一层的初始模型进行分词，然后将分词结果以及句子作为第二层分词模型的输入。通过训练以后，第二层的分词模型将能够学习到初始领域语料与目标领域语料分词风格之间的差异。在测试时，首先使用初始模型对句子进行分词，分词结果和句子作为目标模型的输入，目标模型对句子进行分词操作，得到最终的分词结果。

在基于 Stacked Learning 框架的跨领域分词方法中使用的新特征有：

- $c_{i-1}c_i sl_0 \ \ i = 0,1$
- $c_{-1}c_1 sl_0$
- $c_{-1}c_0c_1 sl_0$
- $c_{i-1}c_i sl_{i-1}sl_i \ \ i = 0,1$
- $c_{-1}c_0c_1 sl_{-1}sl_0sl_1$

其中c表示处理的字，sl表示使用第一层模型给字赋予的分词标签，下标表示相对于当前考虑的字的相对位置。

4.2.2 实验结果及分析

在实验中使用 PKU 训练语料中的 10000 句训练第一层初始模型。金融语料训练集、开发集和测试划分方式和 4.1.2 节跨领域分词增量训练语料为 5000 句时一致。开发集上准确率、召回率以及 F 值实验结果如图 4-14、4-15、4-16 所示：

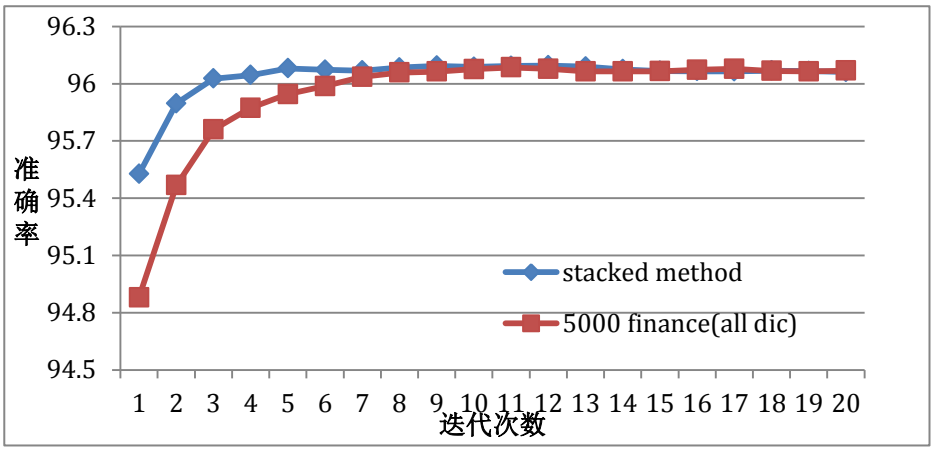


图 4-14 开发集准确率对比结果

Fig.4-14 result of development set accuracy

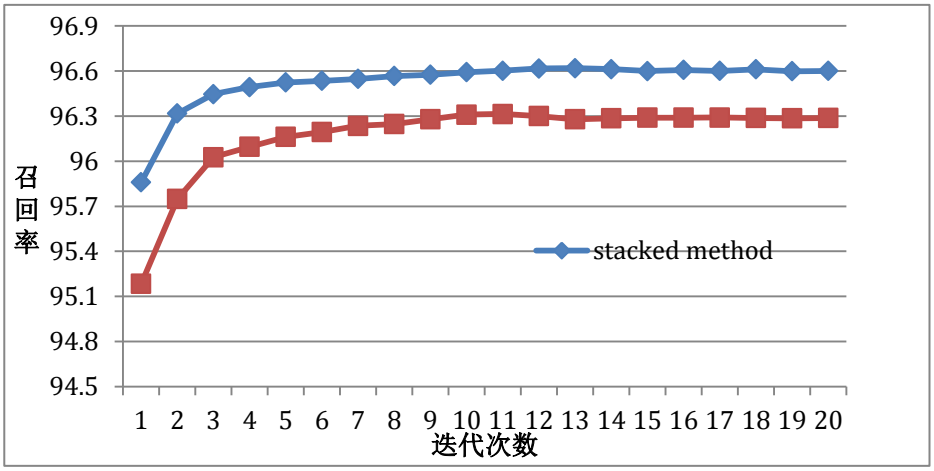


图 4-15 开发集召回率对比结果

Fig.4-15 result of development set recall

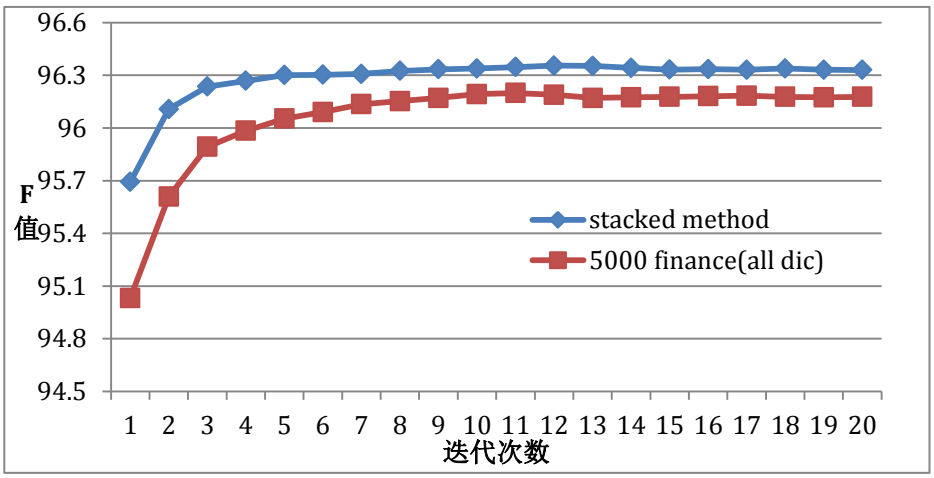


图 4-16 开发集 F 值对比结果

Fig.4-16 result of development set F1

根据开发集性能选择模型对测试集语料进行测试，实验结果如表 4-7 所示：

表 4-7 测试集实验结果

Table 4-7 perfprmance in test set

Method	P(%)	R(%)	F(%)
5000 finance(all dic)	96.01	96.18	96.09
Stacked method	**96.05**	**96.73**	**96.39**

从实验结果可以看出使用基于 Stacked Learning 框架的跨领域分词方法在原有最好性能基础上提高了 0.3%。提高并不是特别显著。同时我们发现，基于 Stacked Learning 的方法主要是提高召回率，对于准确率基本没有影响。

基于 Stacked Learning 的方法适用的场景应该是两种领域的文本分词风格存在一定的对应模式。通过使用 Stacked Learning 方法学习到这种对应模式之间的映射关系$f: L_1 \rightarrow L_2$。

通过对 PKU 初始训练语料和金融训练语料中的词语进行统计发现，PKU 语料中包含 38459 个词，但是有 31255 个词语完全没有出现在金融文本中。金融语料中包含 12107 个词语，有 6032 个词语未出现在 PKU 语料中（未出现是指未作为整个句子子串出现，而不仅仅是未作为句子中的某个词语）。通过统计数据表明，两种语料不仅在分词风格上存在不一致，更在内容上存在显著差异。模型在两个不同领域性能出现显著差异的另一个原因是两个模型的特征空间差别较大，而不仅仅是由分词风格的不一致导致的。因此基于 Stacked Learning 框架的方法对于性能的提高并不是特别显著。但是基于 Stacked Learning 框架的分词方法仍然可作为跨领域分词的一个重要方法，特别是两个领域特征空间差别不存在显著差异的情况。

4.3 本章小结

在本章首先提出三种基于感知器的模型增量训练方法，然后在不同的数据集上进行分词、词性标注对比实验。通过对比实验验证了模型增量训练方法在相同领域数据集上中文分词、词性标注任务中的必要性和有效性。

同时通过对增量训练方法在跨领域分词中性能不佳的原因进行分析，我们将 Stacked Learning 框架应用到跨领域中文分词中，通过对比实验，在金融语料上 F 值取得了 0.3%的提升。

结 论

中文分词、词性标注作为中文自然语言处理的基础任务，对后续其他任务的处理结果具有决定性的影响。中文分词、词性标注可统一为序列标注任务。在序列标注任务中，基于统计的机器学习方法目前在性能上取得了较好的结果。一类统计模型从概率的角度刻画序列标注任务，另一类统计模型直接对序列标注任务进行打分。本文着眼于构建一个高效的中文分词、词性标注系统，由于感知器算法在性能、效率方面的优势，我们使用感知器算法作为分词、词性标注训练算法。

在中文分词、词性标注中，词典对性能具有重要的影响。在分词中，首先通过词典提取特征可以提高分词性能，而且在本文使用的方法中，通过使用新领域词语对词典进行扩充还能实现分词的领域自适应性。通过实验证明，通过添加 1049 个金融词语扩充 PKU 词典后，PKU 分词模型在金融领域的分词性能取得了 3.5%的提升，证明了分词方法具有较好的领域自适应性。在词性标注中，通过使用词典不仅能提高词性标注性能，还能够提高词性标注的效率。我们从训练语料中自动获取词典，避免了手工构建词典所需人力、物力资源的消耗。我们使用词典与统计相结合的分词、词性标注方法，在分词、词性标注任务上取得了较好的性能。

虽然感知器算法在训练效率上已经优于 ME、CRF 等较复杂的模型，但是当训练语料规模较大的时候，感知器算法的训练仍然是一个非常耗时的过程。为了加快训练效率，我们实现了基于感知器的并行训练算法。实验结果表明，在不显著降低性能的前提下，并行训练能够大幅度提高模型训练效率。同时我们对模型文件进行压缩，不仅能够加快测试速度而且还能够减小运行时的内存需求。我们基于无指导的方法，对大规模未标注数据进行分词、聚类，提取聚类特征是词性标注取得了 0.3%的性能提升。

随着时间的推移，语料库的规模和内容不断得到扩充。在 ME、CRF 这些统计模型中，一旦训练结束，使用新增语料对模型进行更改将会非常困难。但是感知器算法却能利用在线算法的优点，在原有模型基础上使用新增语料进行增量训练，提高模型性能。在本文中，我们提出了三种不同的基于感知器的模型增量训练方法，通过实验验证了该方法在相同领域数据的有效性。在跨领域分词中，在使用词典实现分词领域自适应的基础上，我们通过使用 Stacked Learning 框架使跨领域分词在原有最好性能基础上取了了 0.3%的提升，验证了该方法的有效性。

综上所述，本文首先以感知器作为模型训练算法，使用词典与统计相结合的分词、词性标注方法，实现了一个性能较好的分词、词性标注系统。同时实现了基于感知器的并行训练算法，在保证性能的前提下，大幅提高训练效率；然后通过使用大规模未标注语料提高词性标注性能。最后，利用感知器作为在线算法的优点，实现了基于感知器的模型增量训练。最后，在跨领域分词中引入 Stacked Learning 框架，实现了跨领域分词的性能提升。

参考文献

[1] Crammer K, Singer Y. Ultraconservative online algorithms for multiclass problems [J]. The Journal of Machine Learning Research, 2003, 3: 951-991.

[2] Collins M. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms[C]. Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10. Association for Computational Linguistics, 2002: 1-8.

[3] 梁南元. 书面汉语自动分词系统-CDWS[J]. 中文信息学报, 1987, 2(2): 44-52.

[4] 张恒, 杨文昭, 屈景辉, 等. 基于词典和词频的中文分词方法[J]. 微计算机信息, 2008, 24(1): 232-240.

[5] 王瑞雷, 栾静, 潘晓花, 等. 一种改进的中文分词正向最大匹配算法[J]. 计算机应用与软件, 2011, 28(3): 195-197.

[6] 张彩琴, 袁健. 改进的正向最大匹配分词算法[J]. 计算机工程与设计, 2010 (011): 2595-2597.

[7] 丁振国, 张卓, 黎靖. 基于 Hash 结构的逆向最大匹配分词算法的改进 [J]. 计算机工程与设计, 2008, 29(12): 3208-3211.

[8] 麦范金, 李东普, 岳晓光. 基于双向匹配法和特征选择算法的中文分词技术研究[J]. 昆明理工大学学报 (自然科学版), 2011, 36(1): 47-51.

[9] 张华平, 刘群. 基于 N 一最短路径方法的中文词语粗分模型[J]. 中文信息学报, 2002, 5: 1-7.

[10] 何克抗, 徐辉, 孙波. 书面汉语自动分词专家系统设计原理[J]. 中文信息学报, 1991, 5(2): 1-14.

[11] 曹星明, 鲁汉榕, 李玉珍. 基于多种知识源的汉语自动分词[J]. 计算机工程与设计, 1998, 2.

[12] 王彩荣.汉语自动分词专家系统的设计与实现[J]. 微处理机,2004,25(3):56-57

[13] 张茂元, 卢正鼎, and 邹春燕. 一种基于语境的中文分词方法研究. 小型微型计算机系统 26.1 (2005).

[14] Rabiner, Lawrence, and B. Juang. An introduction to hidden Markov models."ASSP Magazine, IEEE 3.1 (1986): 4-16.

[15] Berger, Adam L., Vincent J. Della Pietra, and Stephen A. Della Pietra. A maximum entropy approach to natural language processing. Computational

linguistics 22.1 (1996): 39-71.

[16] Lafferty, John, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. ICML 18(2001):45-54.

[17] Zhang, Hua-Ping, et al. HHMM-based Chinese lexical analyzer ICTCLAS.Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 17. Association for Computational Linguistics, 2003.

[18] Xue, Nianwen. Chinese word segmentation as character tagging. Computational Linguistics and Chinese Language Processing 8.1 (2003): 29-48.

[19] Peng, Fuchun, Fangfang Feng, and Andrew McCallum. Chinese segmentation and new word detection using conditional random fields. Proceedings of the 20th international conference on Computational Linguistics. Association for Computational Linguistics, 2004.

[20] Low, Jin Kiat, Hwee Tou Ng, and Wenyuan Guo. A maximum entropy approach to Chinese word segmentation. Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing. Vol. 1612164. 2005.

[21] 赵伟, et al. 一种规则与统计相结合的汉语分词方法. 计算机应用研究 21.3 (2004): 23-25.

[22] Màrquez, Lluís, Lluis Padro, and Horacio Rodriguez. A machine learning approach to POS tagging. Machine Learning 39.1 (2000): 59-91.

[23] Brill, Eric. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. Computational linguistics21.4 (1995): 543-565.

[24] 洪铭材, et al. 基于条件随机场 (CRFs) 的中文词性标注方法. 计算机科学 33.10 (2006): 148-155.

[25] Merialdo, Bernard. Tagging English text with a probabilistic model. Computational linguistics 20.2 (1994): 155-171.

[26] Giménez, Jesús, and Lluis Marquez. SVMTool: A general POS tagger generator based on Support Vector Machines. In Proceedings of the 4th International Conference on Language Resources and Evaluation. 2004.

[27] Zhang, Yue, and Stephen Clark. Joint word segmentation and POS tagging using a single perceptron. Proceedings of ACL-08: HLT (2008): 888-896.

[28] Eddy S R. Hidden markov models [J]. Current opinion in structural biology, 1996, 6(3): 361-365.

[29] Dobrushin R L. Central limit theorem for nonstationary Markov chains. I[J]. Theory of Probability & Its Applications, 1956, 1(1): 65-80.

[30] Ratnaparkhi A. A maximum entropy model for part-of-speech tagging[C]. Proceedings of the conference on empirical methods in natural language processing. 1996, 1: 133-142.

[31] Manshadi V H, Gharan S O, Saberi A. Online stochastic matching: Online actions based on offline statistics [J]. Mathematics of Operations Research, 2012, 37(4): 559-573.

[32] Barzilai J, Borwein J M. Two-point step size gradient methods [J]. IMA Journal of Numerical Analysis, 1988, 8(1): 141-148.

[33] Freund Y, Schapire R E. Large margin classification using the perceptron algorithm [J]. Machine learning, 1999, 37(3): 277-296.

[34] Forney Jr G D. The viterbi algorithm [J]. Proceedings of the IEEE, 1973, 61(3): 268-278.

[35] 张梅山, 邓知龙, 车万翔, 等. 统计与词典相结合的领域自适应中文分词 [J]. 中国计算语言学研究前沿进展 (2009-2011), 2011.

[36] Emerson T. The second international Chinese word segmentation bakeoff[C]. Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing. 2005, 133.

[37] Tseng H, Chang P, Andrew G, et al. A conditional random field word segmenter for sighan bakeoff 2005[C]. Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing. Jeju Island, Korea, 2005, 171.

[38] Zhang R, Kikui G, Sumita E. Subword-based tagging by conditional random fields for Chinese word segmentation[C]. Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers. Association for Computational Linguistics, 2006: 193-196.

[39] Zhang Y, Clark S. Chinese segmentation with a word-based perceptron algorithm[C]. ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. 2007, 45(1): 840.

[40] Gao J, Andrew G, Johnson M, et al. A comparative study of parameter estimation methods for statistical natural language processing[C]. ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. 2007, 45(1): 824.

[41] Sun X, Zhang Y, Matsuzaki T, et al. A discriminative latent variable chinese segmenter with hybrid word/character information[C]. Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, 2009: 56-64.

[42] Sun X, Wang H, Li W. Fast online training with frequency-adaptive learning

rates for chinese word segmentation and new word detection[C]. Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1. Association for Computational Linguistics, 2012: 253-262.

[43] 王敏, 郑家恒. 基于改进的隐马尔科夫模型的汉语词性标注[J]. 计算机应用, 2006, 26(12): 197-198.

[44] 梁以敏, 黄德根. 基于完全二阶隐马尔可夫模型的汉语词性标注[J]. 计算机工程, 2005, 31(10): 177-179.

[45] 洪铭材, 张阔, 唐杰, 等. 基于条件随机场 (CRFs) 的中文词性标注方法[J].

[46] 王丽杰, 车万翔, 刘挺. 基于 SVMTool 的中文词性标注[J]. 中文信息学报, 2009, 23(4): 16-21.

[47] Jiang W, Huang L, Liu Q, et al. A cascaded linear model for joint chinese word segmentation and part-of-speech tagging[C]. In Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics. 2008.

[48] Jiang W, Mi H, Liu Q. Word lattice reranking for Chinese word segmentation and part-of-speech tagging[C]. Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1. Association for Computational Linguistics, 2008: 385-392.

[49] Kruengkrai C, Uchimoto K, Kazama J, et al. An error-driven word-character hybrid model for joint Chinese word segmentation and POS tagging[C]. Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1. Association for Computational Linguistics, 2009: 513-521.

[50] Zhang Y, Clark S. A fast decoder for joint word segmentation and POS-tagging using a single discriminative model[C]. Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2010: 843-852.

[51] Sun W. A stacked sub-word model for joint chinese word segmentation and part-of-speech tagging[C]. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. 2011: 1385-1394.

[52] McDonald R, Hall K, Mann G. Distributed training strategies for the structured perceptron[C]. Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for

Computational Linguistics. Association for Computational Linguistics, 2010: 456-464.

[53] Mann G, McDonald R, Mohri M, et al. Efficient large-scale distributed training of conditional maximum entropy models [J]. Advances in Neural Information Processing Systems, 2009, 22: 1231-1239.

[54] Lin X, Zhao L, Yu D, et al. Distributed Training for Conditional Random Fields[C]. Natural Language Processing and Knowledge Engineering (NLP-KE), 2010 International Conference on. IEEE, 2010: 1-6.

[55] Brown, Peter F., et al. Class-based n-gram models of natural language. Computational linguistics 18.4 (1992): 467-479.

[56] Cohen, William W. Stacked sequential learning. CARNEGIE-MELLON UNIV PITTSBURGH PA SCHOOL OF COMPUTER SCIENCE, 2005.

[57] Wu, Dekai, Grace Ngai, and Marine Carpuat. A stacked, voted, stacked model for named entity recognition. Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4. Association for Computational Linguistics, 2003.

[58] Nivre, Joakim, and Ryan McDonald. Integrating graph-based and transition-based dependency parsers. Proceedings of ACL-08: HLT (2008): 950-958.

哈尔滨工业大学学位论文原创性声明及使用授权说明

学位论文原创性声明

本人郑重声明：此处所提交的学位论文《基于感知器算法的高效中文分词与词性标注系统设计与实现》，是本人在导师指导下，在哈尔滨工业大学攻读学位期间独立进行研究工作所取得的成果，且学位论文中除已标注引用文献的部分外不包含他人完成或已发表的研究成果。对本学位论文的研究工作做出重要贡献的个人和集体，均已在文中以明确方式注明。

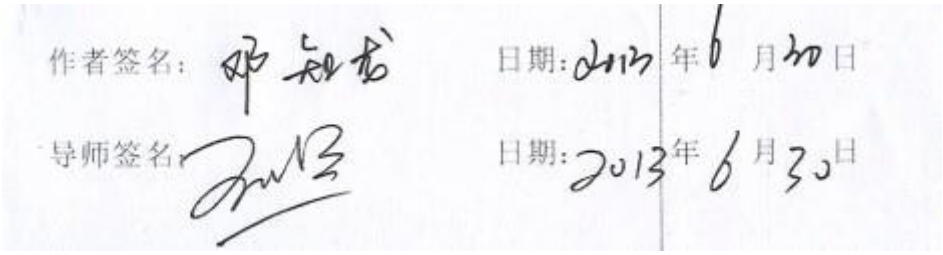
学位论文使用权限

学位论文是研究生在哈尔滨工业大学攻读学位期间完成的成果，知识产权归属哈尔滨工业大学。学位论文的使用权限如下：

（1）学校可以采用影印、缩印或其他复制手段保存研究生上交的学位论文，并向国家图书馆报送学位论文；（2）学校可以将学位论文部分或全部内容编入有关数据库进行检索和提供相应阅览服务；（3）研究生毕业后发表与此学位论文研究成果相关的学术论文和其他成果时，应征得导师同意，且第一署名单位为哈尔滨工业大学。

保密论文在保密期内遵守有关保密规定，解密后适用于此使用权限规定。

本人知悉学位论文的使用权限，并将遵守有关规定。



致 谢

时间匆匆而过，研究生的生涯眼看就要结束了，回首进入 SCIR 的两年，百感交集，满怀感恩。这两年的时间在我的研究生生活中留下了如此多的美好的回忆，研究中心的老师同学伴随着我成长，给予了我无数的帮助和指导。

首先要感谢我的导师刘挺教授。感谢您给我这个弥足珍贵的机会，让我能够加入 SCIR 这个友爱、优秀的大家庭，感谢您在我成长中对我的点拨以及对我的包容和理解。如果当初没能进入 SCIR 这个优秀的大家庭，我无法取得这么大的进步，如果没能得到这么多优秀的老师指点，同学们的帮助，我不能获得这么大的动力和启发。感谢刘老师为我们构造了这么优异的成长环境。

感谢我的指导老师车万翔老师，从进入实验室开始到研究生毕业两年半的时间，我就一直在车老师带领的 LA 组学习，感谢您为 LA 组营造了宽松、充满活力的学习氛围，在您去斯坦福大学访问的一年里，您仍然时刻牵挂着实验室的工作和同学。您在工作中的兢兢业业，在研究时的细致入微，讨论时的平和从容，思考时的开阔视野与敏捷思维，生活中的豁达乐观，都使我受益匪浅。感谢您对我的包容，在我失落、消极的时候给我的支持和鼓励。

感谢 LA 组的大师兄正华师兄，大师兄在车老师出差以及去斯坦福访问的时间里，担当起了 LA 组的负责人，不仅将 LA 组的工作安排的井井有序，而且经常将自己做研究的心得跟大家分享、交流，使我们获益颇多。正华师兄严谨、踏实的精神值得我们每一个人学习。

感谢 LA 组的二师兄梅山师兄，从我进入实验室开始，就一直在梅山师兄的带领下学习，从最初的分词到后来的语义依存分析，所有这些工作都是在梅山师兄的细心指导下完成的。在找工作的时候梅山师兄给我了很大的鼓励和支持。同时感谢梅山师兄对我的包容，由于自己的倔强和无知，给师兄的工作造成了很多不必要的麻烦。师兄还多次将自己的工作心得教授给我，让我少走了很多弯路。

感谢郭江师兄，在每一次遇到问题的时候总向郭江师兄请教，每一次师兄都会细致、耐心的给我进行讲解，每一次与师兄的讨论都获益匪浅。师兄踏实、专研的精神永远是我学习的榜样。

感谢一佳、丁宇、任斌、少磊以及 LA 组的其他同学，非常幸运在人生最美好的时刻遇到你们，和你们在一起的两年是我一生中最美好的回忆，希望你们在以后的日子里学习、生活一帆风顺。感谢 SCIR 的其他老师和同学，谢谢你们对我生活的关心和学习的帮助。我会珍惜这两年多在实验室的美好回忆。

感谢赵江江、陆子龙、张健、宋原、刘安安、慕福楠、骄阳、王沛、刘飞。