基于“大词”实例的中文分词研究*

修 驰[1]，宋 柔[1,2]

[1]北京工业大学 计算机学院，北京 100022

[2]北京语言大学 语言信息处理研究所，北京 100083

E-mail: xiuchi@blcu.edu.cn； songrou@blcu.edu.cn

摘 要：近几年的中文分词研究中，基于条件随机场(CRF)模型的中文分词方法得到了广泛的关注。但是这种分词方法在处理歧义切分方面存在一定的问题。CRF 虽然可以消除大部分原有的分词歧义，却会带来更多新的错误切分。本文尝试找到一种简单的、基于“大词”实例的机器学习方法解决分词歧义问题。实验结果表明，该方法可以简单有效的解决原有的分词歧义问题，并且不会产生更多新的歧义切分。

关键词：中文分词；CRF；大词；分词歧义；机器学习

Research on Big-word Example-based Chinese Word Segmentation

Xiu Chi[1], Song Rou[1,2]

[1] College of Computer Science, Beijing University of Technology, Beijing 100022

[2] Center of Language Information Processing, Beijing Language and Culture University, Beijing 100083

E-mail: xiuchi@blcu.edu.cn; songrou@blcu.edu.cn

Abstract: Chinese word segmentation based on CRF(Conditional Random Field) has attracted the most attention in recent research. But this method exists some problems in handling the ambiguity of word segmentation. Although most original ambiguity errors are solved, it will create more errors than it solved. In this paper, we attempt to find a simple and example-based machine learning method to deal with the problem of word segmentation ambiguity: the method based on big word. The experiment results indicate that big-word based method can solve the ambiguity simple and effective. And it will not introduce more new errors.

Keywords: Chinese Word Segmentation(CWS) ; CRF;big-word; ambiguity; machine learning

1 引言

解决中文分词问题的方法主要分为两类：基于词（或词典）的方法，例如基于规则的最大匹配方法[1]、基于统计的词的 N 元语法的方法[2]。基于字的方法，例如基于最大熵模型（ME）[3]、基于条件随机场模型（CRF）[4]的中文分词方法。

分词歧义和未登录词（OOV）一直是影响中文分词效果的两大因素。Bakeoff 2005 的语料库统计数据说明未登录词造成的分词精度失落比歧义切分造成的精度失落至少大 5 倍以上[5]。基于 CRF 模型的字标注分词方法由于可以较好的解决 OOV 问题，在近几年的 bakeoff 中都取得了很好的成绩。但是歧义切分错误仍然是汉语分词中不可忽视的问题，CRF 模型对歧义切分问题的解决并不够好，而且这一方法开销大、不灵活。有人采用规则加实例库的办法消除分词歧义[6]。但这种方法需要人工构建规则与实例库，只能解决有限语言现象，难以适用于各种不同的语料。因此，希望找到一种专用于中文分词的机器学习方法，既可以吸收基于字的分词方法的优点，充分挖掘训练语料中的分词信息，又不需要太长的训练时间得到较好的分词结果。

本文从 CRF 在分词歧义上存在的问题入手，提出了基于“大词”词表分词的方法，即一种基于实例的中文分词方法。利用大词可以简化机器学习过程，充分利用训练语料中的知识。本文组

* 本文得到国家自然科学基金（60872121）的资助。

织方式如下：第 2 节分析 CRF 中文分词方法存在的问题；第 3 节提出基于大词解决中文分词歧义的方法；第 4 节阐述实验设计并分析结果；第 5 节进行总结。

2 CRF

近几年流行的基于字序列标注的机器学习方法在分词方面取得了较好的结果。在各种机器学习方法中，CRF 模型方法效果最优。条件随机场 CRF(Conditional Random Filed)[4]，是一个无向图模型，在给定观察值的情况下计算状态值的条件概率。在基于该模型的分词方法中，通常是用字形和字类作为观察值的特征。测试结果[7]表明，这种方法可以较好的解决 OOV 问题。也可以在一定程度上解决已登录词的分词歧义（如表 1 所示）。但是我们分析分词结果发现，CRF 在解决 OOV 和分词歧义的同时产生了很多外加的错误。本文第 4 节给出的实验数据表明，CRF 同正向最大匹配方法相比，在后者的分词歧义错误中解决了 1511 例，占 87%，但是产生了 1840 例分词歧义错误。表 2、3、4 举例给出了 CRF 分词中存在的问题。

表 1　CRF 解决歧义切分

FMM 切分结果	CRF 切分结果
稿件 / 上下 / 功夫 /	稿件 / 上 / 下功夫 /
不仅 / 不配 / 合 / ,	不仅 / 不 / 配合 / ,

表 2　训练语料中出现的词 CRF 切分错误

原词	CRF 切分
一支支	着 / 一 / 支 / 支动 / 听
请教 / 老者	记者 / 请 / 教老者 / 。

表 2 显示在训练语料中出现的、有且只有一种分词方式的字符串，利用 CRF 分词方法切分错误的词。字符串“一支支”在训练语料中出现过 1 次，“一 / 支 / ”出现 75 次。“支”共出现 1248 次，词首占 69%、词尾占 10%，单字占 15%。“动”共出现 4163 次，词尾占 71%。“一支”分开的概率更大，“支动”连接在一起概率更高。因此 CRF 产生错误的切分结果。

表 3　不应该成词的字符串被 CRF 合成一个词

原词	CRF 切分
分 /6/ 路	总工会 / 将 / 分 6 路 / 赴 / 全省 /
送电 /160 万	向 / 广东 / 送 / 电 160 万 / 千瓦 /

表 4　同种类型的字符串 CRF 分词结果不同

切分方式 1	切分方式 2
同比 / 增收 /11.1 亿 /	同 / 比 / 上升 /65% /
降水 / 概率 /20%	降 / 水概 / 率 0%

表 3 显示训练语料中没有出现，但也不应该识别为词的字符串。如果被测字符没有在训练语料中出现，前一个和后一个字符对它的影响十分重要。“6”这个字符在测试语料中是半角的，训练语料中没有半角字符，因此“6”没有在训练语料中出现过，但由于“分”经常做词首，“路”经常做词尾，取全局最优值后，“分 6 路”被划分为一个词。

表 4 显示测试语料中类型相同的两句话，由于上下文字形不一样，CRF 的分词方式不同。“同比”在训练语料中出现 15 次，其中 13 次为“同比 / 增长”，因此字符串“同比增收”可以正确切分。但“比 / 上”在训练语料中出现过 131 次，从全局出发，切分成“比 / 上” 比切分成“同比”可以得到更好的概率值。因此“同比上升”不能切分正确。

3 大词

3.1 大词定义

CRF 训练的目的是为了让模型尽可能的挖掘训练语料中的分词信息，拟合训练语料。OOV 可以按训练语料的分词知识进行构词。但对于在训练语料中已经存在的词所构成的词串，如果没有切分歧义，则不需要构词，可以不使用 CRF 分词。

对于测试训练语料中存在的词，字符串匹配是最简单的拟合语料的过程。假设测试语料中的

字符串 S_{test}，可以与训练语料中的字符串 S_{train} 完全匹配，且 S_{train} 在训练语料中只有一种分词标记方式，使用 S_{train} 的分词标记方式对 S_{test} 分词，也是一个拟合训练语料的过程。

如果 S_{train} 是由几个词组成的，S_{train} 串中间的字符携带了上下文信息，这样的字符在上下文环境的约束下，分词标记一定是确定的，可以避免分词歧义。而且不需要再计算概率和全局最优化，可以避免 CRF 中“一支 / 支动 / 听”“请 / 教老者”这样的错误发生。

因此我们提出用“大词”对训练语料中出现过的字符串分词。“大词”可以由多个小词组成，在训练语料中有且仅有一种切分方式。初步定义如下：

设每个字的分词信息使用标注符号表示：B 代表词首，M 代表词中，E 代表词尾，S 代表单字词（也可以加入 B2、B3 等标记表示词首第 2 个字、第 3 个字等，不影响下面的定义）。

定义 1：字符串 S，$S = w_1 w_2 \cdots w_n (n \geq 2)$，在训练语料中出现过 m 次 $(m > 1)$，每次出现时的分词标注集序列为 $T_j (j = 1, 2, \cdots m)$，$T_j = t_{j1} t_{j2} \cdots t_{jn}$，如果所有标注序列 T_j 完全相同，即分词信息一致，而且首尾都被切开，即 $\begin{cases} t_{j1} = S \quad or \quad t_{j1} = B \\ t_{jn} = S \quad or \quad t_{jn} = E \end{cases}$ 则 S 是大词，分词方式为 T_j。

定义 1 限制所有的 T_j 必须完全相等，目的是为了找到内部和边界在训练语料中都没有歧义切分的字符串。由于没有考虑上下文，当语料中同时存在“基本 / 生存”和“基本 / 生存权”时，字符串“基本生存”因为“存”有两种标记方式，则不是大词。当语料不断增加，类似于“基本生存”的现象也会不断增加。另一方面，这种限制相当于对训练语料的过度拟合，测试语料中字符串上下文的变化，会产生部分新的切分形式，这种情况下利用现有的大词切分，会带来新的切分歧义。针对以上问题，我们对定义 1 进行的了修改，对“大词”添加了上下文的限制。

定义 2：字符串 $\alpha S \beta$，（$S = w_1 w_2 \cdots w_n$，$n \geq 2$；α，$\beta \in$ 词 or ϕ），在训练语料中出现过 m 次 $(m \geq 1)$，每次出现时 S 的分词标注序列为 $T_j (j = 1, 2, \cdots m)$，$T_j = t_{j1} t_{j2} \cdots t_{jn}$，如果所有标注序列 T_j 完全相同，而且 t_{j1} t_{jn} 满足：$\begin{cases} t_{j1} = S \quad or \quad t_{j1} = B \\ t_{jn} = S \quad or \quad t_{jn} = E \end{cases}$，那么 S 是大词，且在上下文为 α、β 的约束条件下分词方式为 T_j。

更改后的定义 2，在实验中的解决切分歧义效果好于定义 1。虽然也会产生部分新的切分歧义，但相对定义 1 已经大幅度减少。

3.2 固结词

实验中，如果测试字符串在大词词表中找不到匹配项，则在普通词表中查找是否有匹配项。使用普通词表会出现如下问题。例如字符串“新世纪”在训练语料中以词的形式出现过，那么“新世纪”是普通词表中的一个词。但是这个字符串在训练语料中存在两种分词方式“新 / 世纪”和“新世纪”。大部分情况下的切分为“新 / 世纪”，只有在特定的上下文环境下才会被切分为“新世纪”（“新世纪 / 出版社”）。由于切分方式不唯一，这个字符串不是大词。当测试语料出现字符串“新世纪”时，则会按照普通词词表“新世纪”的方式切分，造成大部分错误。实验中这样的问题十分明显，因此将实验中使用的“普通词”更改为“固结词”。

固结词是指，普通词表中的词，如果在训练语料中总是整体出现从不被切开，或者虽然有被切开的情况但以整体一个词的形式出现的次数最多，就将这个词作为固结词。因此，“新世纪”不在固结词词表中。虽然固结词和大词词表中都不存在“新世纪”这个词，但是当测试语料拥有足够的上下文，则可以找到包含“新世纪”的大词，将“新世纪”正确切分出来。

3.3 分词策略

分词策略是将基于大词实例的方法与基于普通词表的方法相结合。原则是以大词为主，普通词为辅。具体方法如下，以定义 2 为例说明：

词表包含：携带上下文信息的大词表、不携带上下文信息的大词表、普通词表。从左至右前向最大匹配（FMM）对测试语料进行分词。设测试语句为 *Sen*。

Step1 取 *Sen* 已经切分出的最后一个小词（如果还未开始切分，则为 ϕ ）为上文 α ，在携带上下文信息的大词表中查找最长匹配字符串 $\alpha S \beta$ 。如果找到这个串，并且这个串的 α 与 S 之间是被切开的关系，则将匹配的字符串按照大词 S 的切分标记标注。转 Step1。如果没有匹配到，转 Step2。

Step2 *Sen* 中等待被切分的字符串，在没有上文信息的大词表中进行最大字符串匹配 $S\beta$ 。如果匹配成功，将匹配的字符串按照大词 S 的切分标记标注。转 Step1。如果没有匹配到，转 Step3。

Step3 *Sen* 中等待被切分的字符串，在普通词表中进行最大字符串匹配。如果匹配成功，将这个词切分出来转 Step4。如果没有匹配到，将首字符字切分出来，此字符为单字。转 Step1。

Step4 对于利用普通词表切分出的字符串，回退一个字。如果以回退的这个字为首字，与后面的字符串可以组成大词，并且，原字符串去掉这个字仍然是个普通词。则更改利用普通词表切分的结果，使用回退一个字得到的切分结果，转 Step1。否则直接转 Step1。

定义 1 与定义 2 相比，大词不携带上下文信息，因此在操作时没有 Step1（在携带上下文信息大词表中查找匹配的过程），其他步骤相同。

目前，我们仅选用了这种最简单的分词策略，已经有效的降低了分词歧义。如果改进策略，相信将得到更好的分词效果。

4 实验

4.1 实验语料基本情况

本实验采用第二季国际分词竞赛（bakeoff-2005）中 PKU 的语料对分词方法进行测试。语料中数字、英文、全半角符号进行了统一的预处理。

表 5 bakeoff-2005 PKU 语料词形词数

PKU	词形（type）	词数（token）
训练语料	55303	1109947
测试语料	13150	104377

PKU 训练语料中数字、英文、符号，使用全角符号（比如 1.1%在训练语料中为 1.1%）。

PKU 测试语料中全半角符号混合使用，大部分数字使用半角符号。未登录词 6006 个（token），占测试语料总词数（token）的 5.7%。在不包含单字词的情况下共有 5592 个未登录词（token），其中数字、英文、日期为 2370 个，占未登录词的 42.3%。

4.2 Baseline 和 Topline

实验的主要目的是测试大词分词方法对歧义的影响。实验结果采用准确率 P、召回率 R 和调和平均值：$F = 2RP/(R+P)$ 来进行评估。

表 6 Baseline

	Precision	Recall	F1-score
Baseline	0.9057	0.9284	0.9169

表 7 Topline

	Precision	Recall	F1-score
Topline	0.9315	0.9694	0.95007

利用训练语料生成普通词词表，使用FMM的方法进行分词，得到分词歧义的baseline。为了进一步确定歧义切分对分词结果影响，实验设置了 topline。topline 表示测试语料中所有分词歧义问题都被解决，只存在OOV错误的分词结果上限。实验中，将标准答案中的OOV用训练语料生成的普通词表进行FMM分词，其他不改变，得到topline。如果所有的分词歧义都被正确解决，那么F值已经可以达到0.95。由此可见，歧义切分问题的解决对分词结果有着重要的影响。由于测试语料、训练语料中存在分词不一致现象，表7显示的是的topline理想值。

4.3 实验设计与结果

CRF的实验使用6-tag[7]标注集：B　B2　B3　M　E　S。本实验对5字窗口和3字窗口特征的结果进行对比，采用3字窗口（表8所示）可以得到更好的分词结果。工具为CRF++0.54[1]。

表8　CRF实验使用的特征集

特征类型	特征模板
一元语法	$C_n(n=-1, 0, 1)$
二元语法	$C_nC_{n+1}(n=-1, 0)$
跳跃特征	$C_{-1}C_1$
标点特征	$Pun(C_0)$
中文数字、阿拉伯数字、英文	T_0

大词实验中，每个实验都考虑定义1 定义2 两种情况。大词的字符串不包含标点符号，长度限制在50个字符以内。表9显示两种定义分词结果对比，大词的F值明显高于baseline。

表9实验1的结果显示，定义2的效果好于定义1。首先，定义1排除的不一致字符串较多，使得大词个数比定义2少19721个。

其次，离开了上下文的约束，大词也会产生新的歧义。例如，“中央歌剧”在训练语料中出现多次，切分方法都是“中央 / 歌剧”，测试语料遇到“中央歌剧院”，按照定义1的大词理解“剧”则被切成词尾，实际应该是“中央 / 歌剧院”，由此产生了新的歧义切分。定义2需要考虑上下文信息，使用的大词则与定义1不同。例如，“为了看清楚”，首先找到携带上下文的字符串，其分词方式为“ϕ / 为了 / 看 / 清”，“ϕ”和“清”分别相当于定义2中的“α”和“β”，“为了看”是“α”和“β”中间的大词，可以确定其分词方式为“为了 / 看 / ”。然后再顺序查找以“看”为上文的携带上下文环境的大词，当查找到“看清楚...”这种字符串，“清楚”被正确的切分出来。因此定义2的大词由于携带了上下文信息，避免了定义1的错误。

表9　大词定义1与定义2的分词结果对比

	实验		Precision	Recall	F1score		Precision	Recall	F1score
1	大词+普通词	定义1	0.9016	0.9372	0.9190	定义2	0.9155	0.9434	0.9292
2	大词+普通词+回退	定义1	0.9019	0.9376	0.9194	定义2	0.9157	0.9441	0.9296
3	大词+固结词	定义1	0.9026	0.9422	0.9220	定义2	0.9166	0.9486	0.9323
4	大词+固结词+回退	定义1	0.9029	0.9425	0.9223	定义2	0.9176	0.9496	0.9334

为了进一步解决普通词产生的歧义，实验2将普通词切分的字符串回退一个字。例如“正面 / 对 / 世界”中的“正面”是由普通词切分得到的，回退 “面”。以“面”为首字，可以在大词表中找到“面对 / 世界”，而且去掉“面”字，“正”是一个单字词，则修改分词结果。

[1] http://crfpp.sourceforge.net

实验 3、4 采用固结词替代普通词。实验表明，将普通词表换成固结词词表，结果有所提高。

表 10 显示的是大词的最好成绩与 CRF 的分词成绩对比，在没有识别 OOV 的情况下，大词 Recall 的成绩已经好于 CRF 的成绩。表 11 显示大词和 CRF 在相同机器上的训练、分词时间。

表 10　大词与 CRF 结果对比

	Precision	Recall	F1-score
大词	0.9176	0.9496	0.9334
CRF	0.9547	0.9449	0.9498

表 11　大词和 CRF 训练时间对比

秒	训练时间	分词时间	
大词	11. 641 秒	Load/3.5 秒	Seg/0.383 秒
CRF	9684 秒	1.5 秒	

表 12 显示的是大词和 CRF 分词结果各类错误的数据统计，其中分词不一致是指训练测试语料中都存在的词，但是分词方式不同，并且不是组合歧义。由于大词携带上下文信息，对于这种分词不一致情况的适应能力好于 baseline。

表 12　大词和 CRF 各类错误数量对比

Type/token	歧义切分	分词不一致	OOV
大词	248/630	223/562	2050/7300
CRF	928/2057	165/474	648/1907

表 13　大词和 CRF 解决分词歧义对比

Type/token	Baseline 歧义	解决	新歧义
大词	445/1728	283/1262	54/164
CRF	445/1728	392/1511	850/1840

从表 12、13 可以发现，CRF 主要解决的是 OOV 识别的问题，CRF 分词所产生的新切分歧义要大于其解决的分词歧义，对于已经在训练语料中的词，可以选用其他的方法解决。利用大词这种简单的机器学习方法，不仅可以解决分词歧义，带来的新分词歧义相对较少。而且不需要大量训练时间，不仅可以提高速度也可以提高分词效果。

5　结语

本文分析了 CRF 在解决分词歧义时存在的问题，指出 CRF 在切分训练语料中出现过的字符串时会产生更多新的分词歧义。提出了基于大词的分词方法，将基于大词实例和基于普通词表的分词方法相结合，利用简单的机器学习拟合训练语料，解决测试语料中分词歧义的问题。实验表明，这种方法可以在一定程度上解决分词歧义问题，并且不会产生太多的副作用。

大词虽然可以解决部分歧义，但仍需要改进分词策略才能取得更好的效果。对于大词、普通词的切分边界，可以吸取 CRF 的优点，引入字在词中位置的概率，进一步解决歧义问题。对于 OOV 的识别，借鉴 CRF 模型的优点找到一种专用的分词方法，也是我们下一步的工作。

参 考 文 献

[1] 骆正清，陈增武，胡上序. 一种改进的 MM 分词方法的算法设计[J]. 中文信息学报，1996, 10(3): 30-36.

[2] 吴春颖，王士同. 基于二元语法的 N-最大概率中文粗分模型[J]. 计算机应用，2007, 27(12): 332-339.

[3] N. Xue. Chinese Word Segmentation as Character Tagging[J]. Computational Linguistics and Chinese Language Processing, 2003, 8(1), 29-48.

[4] J.Lafferty, A. McCallum, and F.Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data.[C]In Proceedings of the 18th International Conf. On Machine Learning, Page 282-289, 2001.

[5] 黄昌宁，赵海. 中文分词十年回顾[J]. 中文信息学报，2007, 21(3): 8-19.

[6] 罗智勇，宋柔. 现代汉语通用分词系统中歧义切分的实用技术[J].计算机研究与发展, 2006, 43(6): 1122-1128.

[7] Hai Zhao and Chunyu Kit. Unsupervised Segmentaion Helps Supervised Learning of Character Tagging for Word Segmentation and Named Entity Recognition[C]//The Sixth SIGHAN Workshop on Chinese Language Processing (SIGHAN-6), Hyderabad, India, 2008: 106-111.