一种基于树核的汉语句法分析多重结果重排序技术[1]

郑晓东　陈 亮　常宝宝

北京大学计算语言学研究所 北京大学计算语言学教育部重点实验室，北京，100871

摘 要： 本文将基于概率上下文无关文法的汉语句法分析分为K-best基础模型和重排序两个阶段，主要研究为对多重结果的重排序技术。本文所介绍的重排序技术有两个特色：一是实现排序问题到分类问题的转化并通过改进的投票感知机算法实现重排序；二是引入树核方法到汉语句法分析中，并对树核做了区别对待产生式规则和引入聚合产生式规则的两个改进，有效地提高了重排序的效率。

关键词： 汉语句法分析 重排序 树核 投票感知机

A Tree Kernel Based Reranking Technique for Chinese Syntactic Parsing

Zheng Xiaodong　Chen Liang　Chang Baobao

Institute of Computational Linguistics, Peking University,　Key Laboratory of Computational Linguistics (Peking University), Ministry of Education,　Beijing 100871, China

Abstract: In this paper, we divide the PCFG based Chinese Syntactic Parsing into two stages, K-best basic model and Reranking, and the second stage reranking is the emphasis of the research. There are two novel points in this paper: the first one is using the improved voted perceptron algorithm to reranking with the transformation from ranking problems to a classification problem; the second one is introducing the tree kernel method, and improving tree kernel form two points rules weights distinction and similar rules converging, which increase reranking efficiency a lot.

Key Words: Chinese Syntactic Parsing　Reranking　Tree Kernel　Voted Perceptron

1　引言

句法分析是自然语言处理中的一个基本问题。许多自然语言处理中的任务，比如语义分析、机器翻译、信息抽取等，其完成的好坏依赖于句法分析的准确率。所以，对句法分析的研究有着重要的理论和实用价值。

汉语句法分析，相对于英语的句法分析起步较晚，加之训练树库的匮乏，其发展较英语句法分析也相对落后。本文把汉语句法分析分为K-best基础模型和重排序两个阶段，主要研究为对第一阶段产生的多重结果的重排序技术。本文中引入树核方法和投票感知机算法进行重排序的工作，Collins和Duffy已经在英语的句法分析中应用实现，并取得了很好的效果[3]。本文将树核和投票感知机算法应用于汉语的句法分析中，采用基于树核的投票感知机算法实现重排序，并且对树核方法做了引入产生式权重r_s以区别对待产生式规则和引入聚合产生式规则的两个改进。本文的实验以宾州中文树库(CTB)5.1 为基础，应用基于概率上下文无关文法的两阶段句法分析系统进行汉语句法分析试验。

[1] 本文工作得到国家自然科学基金项目(60303003)；国家社会科学基金项目(06BYY048)的支持。

本文第二节阐释了本文中分析树如何通过特征函数表示为特征向量，即分析树在机器学习方法中的形式化表示。第三节讲什么是树核，其中详细介绍了树核的计算方法。第四节内容包括引入机器学习方法进行重排序的相关算法和对树核方法的两个改进。第五节是本文中实验的设置和相关数据以及对实验结果的分析。最后一节是对本文中的重排序方法和相关实验的总结和讨论。

2 分析树的形式化表示

重排序的任务中，使用机器学习方法的一个首要问题是如何形式化表示这些分析树。对一个分析树T，定义一个函数$h(T)\in R^n$ 将分析树T表示为一个n维的特征向量，这个特征向量是分析树T的形式化表示。本文中函数h(T)取分析树T的内部结构(子树)作为T的特征来形式化表示。特征向量的维度n是树库中所有不重复的子树的个数，用下标1…m($1\leqslant m\leqslant n$)索引。定义$h_i(T)$为树库中标号为i的子树在分析树T中出现的次数；所以分析树T可以形式表示为$h(T)=(h_1(T),h_2(T),\cdots,h_n(T))$。

在句法分析领域中，子树通常有三种不同的定义：子树 STs(Vishwanathan and Smola 2002)，子集树 SSTs(Collins and Duffy 2002)， 和部分树 PTs(Moschitti 2006)。

本文采用子集树(SSTs)作为子树的定义方式。

3 树核

有了分析树的特征向量表示之后，使用两棵分析树特征向量的点积来定义这两棵分析树的相似度，用$Sim(T_1,T_2)$表示。

$$Sim(T_1,T_2)=h(T_1)\cdot h(T_2)=\sum_{i=1}^{n}h_i(T_1)h_i(T_2) \tag{1}$$

在实际计算$Sim(T_1,T_2)$过程中，由于树库中的不重复的子树的数目n是非常庞大的，常规的计算点积的方法，时间代价很大，所以引入树核方法[2]。

T1、T2 中的结点集合分别用N_1、N_2表示，我们定义指标函数$I_i(n)$如下：

$$I_i(n)=\begin{cases}1,\text{如果第}i\text{个子树以结点}n\text{为根}\\0,\text{如果第i个子树不以结点}n\text{为根}\end{cases}$$

有了上述$I_i(n)$的定义，有：

$$Sim(T_1,T_2)=\sum_{i=1}^{n}h_i(T_1)h_i(T_2)=\sum_{i=1}^{n}\sum_{n1\in N1}\sum_{n2\in N2}I_i(n_1)I_i(n_2)=\sum_{n1\in N1}\sum_{n2\in N2}\sum_{i=1}^{n}I_i(n_1)I_i(n_2) \tag{2}$$

把$\sum_{i=1}^{n}I_i(n_1)I_i(n_2)$部分重写为$Common(n_1,n_2)$。

$Common(n_1,n_2)$是计算树核的关键，它表示在T_1和T_2中，同时以n_1和n_2为根的公共子树的数目，

可以通过递归的方式快速计算，具体计算方法如下：

- 如果在T_1，T_2中，n_1和n_2处的产生式规则不同，则Common(n_1, n_2)=0
- 如果在T_1，T_2中，n_1和n_2处的产生式规则相同，且n_1、n_2都为预终结符(pre-terminal)，则Common(n_1, n_2)=1
- 如果在T_1，T_2中，n_1和n_2处的产生式规则相同，且n_1、n_2都不为预终结符，则Common(n_1, n_2)可以通过下述形式计算出来

$$Common(n1,n2)=\prod_{i=1}^{nc(n1)}(1+Common(ch(n1,i),ch(n2,i))) \tag{3}$$

其中nc(n_1)表示结点n_1的孩子数目，ch(n_1, i)表示结点n_1的第i个孩子，因为n_1和n_2处的产生式规则相同，所以有nc(n_1)=nc(n_2)，ch(n_1, i)=ch(n_2, i)。

有了上述的计算方法，Common(n_1, n_2)可以通过动态规划的方法来计算，所以计算Sim(T_1, T_2)的时间复杂度为O(|N_1||N_2|)。

4 算法

4.1 使用树核的投票感知机算法

重排序的任务是如何从第一阶段产生的侯选分析树中选取正确的侯选分析树。在引入投票感知机算法之前，先介绍本论文中的重排序问题：

- 训练集中句子数为n，用x_{ij}表示训练树库中第i个句子的第j个分析结果，第i个句子有n_i个分析结果，$1\leqslant i\leqslant n$和$1\leqslant j\leqslant n_i$
- 不失一般性，假设x_{i1}是第i个句子的最好分析结果
- $Q(x_{ij})$是第一阶段K-Best程序获得的x_{ij}的概率，这个概率的对数形式用$L(x_{ij})$表示
- 有一个独立的测试集，第一阶段K-Best有输出y_{ij}，对y_{ij}也有类似的定义$Q(y_{ij})$和$L(y_{ij})$

现在我们需要定义一个排序函数(ranking function)：$F(x_{ij})$，它能对y_{ij}，$1\leqslant j\leqslant n_i$重新排序，并输出：$\arg\max_{j=1...n_i} F(y_{ij})$

重排序问题实质上是一个分类问题，在文献[6]中提出了实现最大边界分类的感知机算法。本文采用了基于树核的投票感知机算法来解决重排序问题。

用向量$\mathbf{h}(x_{ij})$表示特征集合$\langle h_0(x_{ij}), h_1(x_{ij}), \cdots, h_n(x_{ij})\rangle$，感知机的分类向量w为$\langle \alpha_0, \alpha_1, \cdots, \alpha_n\rangle$，排序函数为：$F(x_{ij}, w)=\mathbf{w}\cdot\mathbf{h}(x_{ij})$

本论文中，使用分类方法去解决排序问题的关键是把要排序的实例做差后产生的“差向量”作为分类的元素。在训练w时，使用训练集中的向量之差$\mathbf{h}(x_{i1})-\mathbf{h}(x_{ij})$作为训练点。

根据感知机的算法思想，有如下等式：

$$\mathbf{w}=\sum_{ij}\alpha_{ij}(\mathbf{h}(x_{i1})-\mathbf{h}(x_{ij}))$$

其中，α_{ij}为感知机在训练过程中，$\mathbf{h}(x_{i1})-\mathbf{h}(x_{ij})$ 被错误分类的次数。

从而有$F(x)=w\cdot h(x)=\sum_{ii_m}\alpha_{ij}[h(x_{i1})\cdot h(x)-h(x_{ij})\cdot h(x)]$，其中$h(x_{i1})\cdot h(x)$的是分析树$x_{i1}$和x的相似度值。$h(x_{ij})\cdot h(x)$的计算用树核方法来实现。

在训练过程中，如果投票感知机迭代次数为t次，则产生t个感知机分类向量 w_i $(1\leqslant i\leqslant t)$。最后根据每次迭代出错的总数计算权重来组合t个感知机分类器，得到最终结果。算法详细描述见图1

定义： $F(x)=\sum_{(i,j)}\alpha_{ij}(h(x_{i1})\cdot h(x)-h(x_{ij})\cdot h(x))$， $1\leqslant i\leqslant n$， $2\leqslant j\leqslant n_i$
训练
初始化：
$\alpha_{ij}=0$
err = 0
do
for i = 1 … n
if $F(x_{i1})<F(x_{ij})$
$\alpha_{ij}=\alpha_{ij}+1$
err = err + 1
Until (err = = 0)
测试
定义：
$F^t(x)=\sum_{(i,j)}\alpha_{ij}(h(x_{i1})\cdot h(x)-h(x_{ij})\cdot h(x))$， $1\leqslant i\leqslant t$， $2\leqslant j\leqslant n_i$， $1\leqslant t\leqslant n$
$V^t(s)=\mathrm{argmax}_{x\in C(s)}F^t(x)$
对一个句子s，输出$\{V^1(s),V^2(s),...,V^n(s)\}$中出现次数最多的下标对应的分析树

图1　本文实验的投票感知机算法框架

4.2 改进树核方法

4.2.1 区别对待产生式规则

在标准的树核方法中，对所有的规则，它们在计算两棵分析树相似性时所作的贡献都是一样的，都为1。但实际上，不同的规则，它们应该被区别对待。出现频度较高的规则，它们对两棵分析树相似性的贡献应该较小；而对于出现频度较低的规则，对两棵分析树相似性的贡献应该较大，如果在两棵分析树中都出现了频度较低的规则，则应该给予这两棵分析树一个较大的相似度。所以，我们引入产生式规则权重r_w来作为新的贡献值计算方式，这种改进启发于逆向文档频率idf(inverted document frequency)，通过产生式规则在树库中频率来计算出产生式规则的权重。对于产生式规则r的产生式规则权重r_w，我们定义如下：

$$r_w=\log\frac{\text{训练树库中的树个数}}{\text{训练树库中包含}r\text{的树个数}}$$

使用产生式规则权重r_w方法区分不同规则的权重也产生了新的问题：有一部分规则，由于包

含它们的分析树的个数特别小，导致这些规则的r_w值非正常的偏大，从而导致在计算两棵分析树相似性中给予了这些规则过分的权重。为了缓解这个问题，我们采取了如下一些措施：

- 对于preterminal->word形式的规则，统计word在训练树库中的频度，对低于某个指定阈值（threshold）的，修改word为“UNKNOWN”，即把上述规则改为preterminal->UNKNOWN，用r'表示修改后的规则，则r_w为：

$$r_w = \sqrt{\log \frac{\text{训练树库中的树个数}}{\text{训练树库中包含r'的树个数}}}$$

- 对于X->λYμ形式的规则，r_w的计算修改为如下形式：

$$r_w = \sqrt{\log \frac{\text{训练树库中的树个数}}{\text{训练树库中包含}r\text{的树个数}}}$$

我们需要相应修改树核的计算方法如下：

- 如果在T_1，T_2中，n_1和n_2处的产生式规则不同，则$Common(n_1, n_2)=0$
- 如果在T_1，T_2中，n_1和n_2处的产生式规则相同，为r，且n_1、n_2都为预终结符（pre-terminal），则$Common(n_1, n_2)= r_w$
- 如果在T_1，T_2中，n_1和n_2处的产生式规则相同，为r，且n_1、n_2都不为预终结符，则$Common(n_1, n_2)$可以通过下述形式计算出来

$$Common(n_1, n_2) = \prod_{i=1}^{nc(n_1)} (r_w + Common(ch(n_1, i), ch(n_2, i)))$$

4.2.2 聚合相似产生式规则

如图2中的两棵分析树，从分析树结构上看，它们是完全相同的，但却因为它们的子树“NN->苹果”和“NN->橘子”的不同，使得在树核方法中，这两棵分析树的相似性被低估；在树核方法中，这两棵分析树有两棵以VP为根的公共子树，它们的相似性为2，但实际上，它们的相似性为4更为合理，因为在这两棵分析树中，子树“NN->苹果”和子树“NN->橘子”应该计算为相同的分析树。所以，基于这一点考虑，我们对树核进行了聚合相似产生式规则修改。

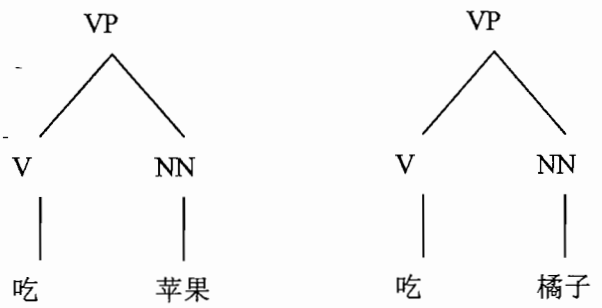

图2　两棵结构相同的分析树

5 实验

本文的实验以宾州中文树库(CTB)5.1 为基础，应用基于概率上下文无关文法的两阶段句法分析系统进行汉语句法分析试验。为了减少样本的选择对结果的影响，我们对 CTB 树库进行了 10 次分割，每次随机从树库中抽取 1/50 的句子作为测试集，剩下的作为训练集，按照同样的实验设置做 10 轮实验，把 10 轮的平均结果作为最终结果。

为了比较第二阶段重排序的效果，设计了第一组实验：

- 实验 1：应用 K-Best CYK 算法进行 1-Best 分析；
- 实验 2：应用 K-Best CYK 算法进行 20-Best 分析得前 20 个分析结果，使用基于标准的树核方法的投票感知机算法进行重排序；

如表 5-1，给出了第一组实验的测试结果：

实验	准确率	召回率	F 值
1: 1-Best CYK 算法	70.4%	74.1%	72.2%
2: 投票感知机算法重排序	74.9%	81.0%	77.8%

表 5-1 第一组实验测试结果

从第一组实验结果中可以看到，把机器学习方法运用到第二阶段的重排序中，采用基于树核的投票感知机算法来实现重排序，对第一阶段的 K-best 句法分析结果在 F 值上有 5 个百分点以上的提升。分析提升的原因是概率上下文无关文法 PCFG 缺少结构信息，基于 PCFG 的句法分析得到的分析树的概率是这个分析树中产生式概率的乘积，缺少对分析树结构上的判别能力。引入基于树核的投票感知机算法后，加入了分析树的结构特征，从而分析效果有了较大的提升。

为了比较修改树核方法后带来的效果，设计了第二组实验：

- 实验 1：第一阶段得到 top20 分析结果，使用基于标准的树核方法的投票感知机算法进行重排序；（同第一组实验 2）
- 实验 2：第一阶段得到top20 分析结果，引入产生式权重r_w区别对待产生式规则计算树核，使用基于引入产生式权重r_w的树核的投票感知机算法进行重排序；
- 实验 3：第一阶段得到 top20 分析结果，引入聚合产生式规则计算树核，使用基于引入聚合产生式规则的树核的投票感知机算法进行重排序；
- 实验 4：第一阶段得到top20 分析结果，引入产生式权重r_w和聚合产生式计算树核，使用基于同时引入产生式权重r_w和聚合产生式的树核的投票感知机算法进行重排序；

如表 5-2，给出了第二组实验的测试结果：

实验	准确率	召回率	F 值
1: 基于标准的树核的投票感知机算法	74.9%	81.0%	77.8%
2: 引入产生式权重r_w计算树核	76.9%	81.4%	79.1%
3: 引入聚合产生式规则计算树核	76.5%	80.6%	78.5%
4: 引入产生式权重r_w和聚合产生式计算树核	76.6%	80.8%	78.7%

表 5-2 第二组实验测试结果

实验 1 和实验 2 结果对比发现，引入产生式权重r_w区别对待产生式规则计算树核后，效果有较大的提升。提升的主要原因是区别对待频率不同的产生式规则，计算出来的树核值更能合理地表示两颗分析树的相似度，应用于感知机模型的学习和测试过程后，效果得到了提升。

实验 1 和实验 3 结果对比，引入聚合产生式规则计算树核的方法，效果好于标准的树核计算方法。前者在后者的基础上，考虑了更多的结构相似性，只要结构相似，子树就能有更多的贡献值，这显然比后者必须要结构完全相同、覆盖词完全相同才能有贡献值的限定要合理，并且比后者更好的解决了结构稀疏的问题。

实验 2，实验 3 和实验 4 结果对比，发现同时引入产生式权重r_s和聚合产生式，效果不如只引用产生式权重r_s的树核计算方法，但是比引入聚合产生式的树核计算方法效果好。比较实验 2 和实验 4，之所以引入聚合产生式后，比只是引入产生式权重r_s树核效果下降了，是因为引入聚合产生式计算树核，计算出的树核值会比之前的树核值大几倍，这样就会减小引入产生式权重r_s区别对待产生式在树核计算中的判别作用。

6 结论和小结

对汉语句法分析的研究主要集中在对中心驱动模型的改进和语义知识的引入上，而对于研究如何引入更多特征来提高句法分析结果则较少。本文从如何引入更多特征出发，研究多重结果的重排序技术，一方面把重排序问题转化为分类问题，引入投票感知机算法；另一方面则把基于树核的重排序技术应用到汉语的句法分析上。实验证明，应用这种重排序技术，句法分析的最终效果比第一阶段的基于PCFG的k-best句法分析模型有较大提升。本文对树核方法有两个创新改进，第一个是引入产生式权重r_s区别对待产生式规则；另一个改进是引入聚合产生式规则。理论和实际数据都显示这两种改进有一定的效果。本文中用于重排序的侯选分析树均来自基于PCFG的k-best模型，如果使用基于词汇化PCFG的k-best模型生成侯选分析树，重排序后得到F值应该会比本文中提供的F值高一些。

参考文献

[1] M.Collins. Discriminative Reranking for Natural Language Parsing. In Proceedings of the 7th International Conference on Machine Learning, Stanford, CA, 2000: 175-182

[2] M.Collins and N.Duffy. Convolution Kernels for Natural Language.In Proceedings of Neural Information Processing Systems (NIPS 14). 2001.

[3] M.Collins and N.Duffy. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In ACL,2002, pages 263-270, Philadelphia, PA.

[4] Alessandro Moschitti, Daniele Pighin, Roberto Basili. Tree Kernels for Semantic Role Labeling. In Proceedings of ACL, 2008, HLT.

[5] M.Collins. Ranking Algorithms for Named-Entity Extraction: Boosting and the Voted Perceptron. In Proceedings of ACL,2002.

[6] Freund, Y. and Schapire, R. Large Margin Classification using the Perceptron Algroithm. In Machine Learning, 1999, 37(3):277-296.

[7] Vishwanathan, S.V.N. and A.J.Smola. Fast kernels on strings and trees. In Preceedings of Neural Information processing Systems, 2002, pages 569-576, Vancouver, British Columbia.

[8] Moschitti, Alessandro. Efficient convolution kernels for dependency and constituent syntactic trees. In Proceedings of The 17th European Conference on Machine Learning, 2006,pages 318-329, Berlin, Germany.