Computer Science and Application 计算机科学与应用, 2013, 3, 251-256
http://dx.doi.org/10.12677/csa.2013.35043 Published Online August 2013 (http://www.hanspub.org/journal/csa.html)

Feature Template-Based Parallel Computation Technique for Conditional Random Fields*

Shuangping Huang[1], Zhiliang Su[2], Xuejun Yue[1], Xiaoling Deng[1]

[1]College of Engineering, South China Agricultural University, Guangzhou
[2]School of Engineering and Applied Sciences, University at Buffalo, The State University of New York, New York
Email: huangshuangping@scau.edu.cn

Received: May 13th, 2013; revised: Jun. 7th, 2013; accepted: Jun. 21st, 2013

Abstract: Conditional Random Fields (CRFs) is a popular probabilistic graphical model, which has been applied in a wide range of areas, including Natural Language Process (NLP), Bioinformatics, Computer Vision, etc. However, contextual features-based methods usually lead to large-scale feature functions and result in high computational complexity and low model training efficiency. In this paper, a feature template-based parallel computation technique is proposed to parallelly process M matrix and reduce computational complexity through observing the main feature of contextual feature function created by the template. Experimental results show that our approach significantly outperforms traditional feature function approach on computation speed.

Keywords: Conditional Random Fields; Feature Templates; Parallel Computation

基于特征模板的条件随机场快速并行计算技术*

黄双萍[1]，苏志良[2]，岳学军[1]，邓小玲[1]

[1]华南农业大学工程学院，广州
[2]美国纽约州立大学水牛城分校工程与应用科学学院，纽约
Email: huangshuangping@scau.edu.cn

收稿日期：2013 年 5 月 13 日；修回日期：2013 年 6 月 7 日；录用日期：2013 年 6 月 21 日

摘　要：条件随机场(CRFs)是现今较为流行的一种概率图模型，已经被广泛地应用到自然语言处理、生物计算、计算机视觉等领域当中。在实际应用中，采用基于上下文特征的特征函数创建法往往会产生大规模的特征函数，造成因计算复杂度过大而建模困难，模型无法高效率地训练生成。本文提出一种基于特征模板的快速并行计算技术，通过观察特征模板创建的上下文特征函数主要特点，对 M 矩阵进行并行处理，降低计算量。实验结果表明，本文提出的快速计算方法较传统方法在速度上有非常大的优势。

关键词：模条件随机场(CRF)；特征模板；并行计算

1. 引言

条件随机场(CRFs，Conditional Random Fields)是 J. Lafferty 等人在 2001 年提出的概率图模型[1]，该模型由于克服了众多其它图模型存在的缺点，表现出良好的分类性能，因而被广泛应用于命名实体标注、DNA 剪接位点预测等序列数据标记任务当中。

条件随机场中常见的特征函数创建方法有基于先验知识创建和 S. Fei 等人提出的基于上下文特征创

*基金项目：国家自然科学基金(31201129)，广东省自然科学基金项目(S2012010009856)。

建两种方式[2]。在实际应用中基于上下文特征的特征函数创建方式取得了很好的效果，被广泛推广。但基于上下文特征创建的特征函数数量规模庞大，造成建模耗费时间过长。

本文针对上下文特征函数的特点，提出一种基于上下文特征模板的并行计算方法。应用此方法可以降低运算复杂度。在 CoNELL 2000(Computational Natural Language Learning 2000)[3]的浅层语义分析实验中应用本文提出的方法后，运算速度较未加速前提升明显。

2. 条件随机场(CRFs)

相比隐马尔科夫模型(HMM，Hidden Markov Model)与最大熵马尔科夫模型(MEMMs，Maximum Entropy Markov Models)两个流行的图模型[3,4]，CRFs 主要克服了过强的条件独立性假设、标记偏见等一系列问题。

一阶链式 CRFs 是最为常用的一种结构，输入一条随机变量序列 x，对应的标记序列 y 的后验概率定义为：

$$p(y|x)=\frac{\prod_{c\in C}\phi_c(y,x)}{Z(x)} \tag{1}$$

其中 ϕ_c 是无向图结构中的连通团块(Clique)的势函数(Potential function)。J. Lafferty 等将势函数定义如下：

$$\prod_{c\in C}\phi_c(y,x)= \exp\left\{\sum_i\sum_{k\in K}\lambda_k f_k(y_{i-1},y_i,x)+\sum_i\sum_{k\in K}\mu_k g_k(y_i,x)\right\} \tag{2}$$

式中 f_k 与 g_k 为特征函数(Feature function)，f_k 考虑当前位置 i 对应的标签值 y_i 与前一位置对应的标签值 y_{i-1} 对(pair)后验概率的影响，因此被称为边特征函数(Edge function)；g_k 只考虑当前位置 i 对应的标签值对后验概率的影响，因此被称为节点特征函数(Node function)。λ_k 与 μ_k 为特征函数对应的权值。z 是归一化函数(Partition function)，用于将计算得到的“势”值归一化为概率形式。

3. 特征函数定义方法

特征函数的定义方法可分为两种，一种是基于先验知识的定义方法，另一种是基于上下文内容的定义方法。

3.1. 基于先验知识的定义方法

基于先验知识的定义方法是指基于先验专业知识定义特征函数。以人名提取任务为例，规定一条特征函数为：

If y_{i-1} = ‘姓’ & y_i = ‘名’ & x_{i-1} = ‘白’ & x_i ≠ ‘色’ ***output*** = 1

因为“白”字既可以作为中国人常见姓氏出现，也可以作为颜色来出现，根据先验知识，若一个“白”字后面是“色”字，则此“白”字通常不是姓氏。

不难看出，根据先验知识定义特征函数具有较强烈的专业针对性，可以非常直观地表现数据特征。然而使用先验知识定义特征函数也有其不足之处。首先，我们未必能完全观察到各种先验知识，比如在 DNA 剪接位点预测任务中，剪接点位置的特征未必能完全被模型使用者观察到，因此不能建立起完善的特征函数表；另外，当一个领域内的先验知识系统十分庞大时，模型使用者很难把所有先验知识都转换为特征函数表达出来。比如在词性标注任务中，中文语法规则库非常庞大，若要对每一条规则都转换为特征函数表达出来，非常耗时。

3.2. 基于上下文特征的定义方法

由于基于先验知识的特征函数定义方法存在的缺陷，Fei S 等提出另一种特征函数的定义方法[2]。这种方法忽略当前任务的专业领域知识，只根据序列当前位置的上下文特征创建特征函数。

CRF++[5]是实现 CRFs 的一个主流开源工具，也是目前综合性能最佳的 CRFs 著名实现，其程序沿用了 Fei S 等人提出的这种特征函数定义方法的思想，并将创建特征函数的模式用一系列模板(Template)直观地表现出来。以某个浅层语法分析任务(Chunking)中的模板为例(表 1 所示)：

每个模板有三部分组成：模板类型、编号、上下文特征提取方式。模板类型分为 Unigram 和 Bigram 两类，写在模板的最前面，分别用缩写 U 和 B 来表示。Unigram 表示由此生成出来的特征函数是节点特征函数，Bigram 表示此模板产生的是边特征函数。编号用以区分不同模板产生的同一个上下文特征，每个模板

Table 1. Feature template paradigm
表 1. 特征模板范例

模板	模板
U00:%x[−2,0]	U14:%x[2,1]
U01:%x[−1,0]	U15:%x[−2,1]/%x[−1,1]
U02:%x[0,0]	U16:%x[−1,1]/%x[0,1]
U03:%x[1,0]	U17:%x[0,1]/%x[1,1]
U04:%x[2,0]	U18:%x[1,1]/%x[2,1]
U05:%x[−1,0]/%x[0,0]	U20:%x[−2,1]/%x[−1,1]/%x[0,1]
U10:%x[−2,1]	U21:%x[−1,1]/%x[0,1]/%x[1,1]
U11:%x[−1,1]	U22:%x[0,1]/%x[1,1]/%x[2,1]
U12:%x[0,1]	B
U13:%x[1,1]	

都有自己的独立编号。上下文特征提取方式包含有两个参数：Row 和 Col，Row 指示取输入序列 x 中相对于当前位置 Row 距离的位置特征，Col 指示取输入序列 x 的第几维属性，例如，在中文分词中，输入序列 x 既可以是原始的字，也可以是每个字对应的 POS(Part of Speech)标签。假如 x 序列有只有 1 种属性，则规定 Col 为 0。

规定对每一对训练预料的每一个位置都套用所有模板规则，每个模板都能得到一批按照自身规则提取出来的上下文特征，然后去除每一批特征中重复的项，这样，每个模板得到一个上下文特征集。假设输出的标签集合大小是$|\mathbb{Y}|$，对于 Unigram 模板得到的上下文特征集，集合里面的每个上下文特征都枚举产生$|\mathbb{Y}|$个特征函数。以模板 U00：%x[−2, 0]里的上下文特征“Vision”为例(见表 2)，若标签集合为$\{\text{noun}, \text{adj}, \text{adv}, \text{verb}\}$，将产生表 2 所示的特征函数。

Table 2. Context feature function examples
表 2. 上下文特征函数示例

行	特征函数
1	if y_i = ‘noun’ & ***context feature*** = ‘Vision’ then ***output*** = 1
2	if y_i = ‘adj’ & ***context feature*** = ‘Vision’ then ***output*** = 1
3	if y_i = ‘adv’ & ***context feature*** = ‘Vision’ then ***output*** = 1
4	if y_i = ‘verb’ & ***context feature*** = ‘Vision’ then ***output*** = 1

4. 基于上下文特征的快速计算技术

J Lafferty 等人发现一阶链式结构 CRFs 模型中，在标签序列 y 前后分别加入两个特殊标签 start 与 stop 后，标签序列 y 的后验概率 $p(y|x)$ 可以矩阵形式表达：

$$p(y|x)=\frac{\prod_{i=1:n+1}M_i(y_{i-1},y_i|x)}{\left[\prod_{i=1:n+1}M_i(x)\right]_{\text{start,stop}}} \tag{3}$$

将输出标签集合记为 Y，该集合中元素记为 y，矩阵 M_i 为一个$|\mathbb{Y}|\times|\mathbb{Y}|$的矩阵，$M_i$ 矩阵中每个元素为：

$$M_i(y',y|x)=\exp\{\Lambda_i(y',y|x)\} \tag{4}$$

式中

$$\begin{aligned}\Lambda_i(y',y|x)&=\sum_k\lambda_k f_k(y_{i-1}=y',y_i=y,x)\\&+\sum_k\mu_k g_k(y_i=y,x)\end{aligned} \tag{5}$$

显然计算 M_i 矩阵要求对 Y 集合中元素 y 进行枚举，并且枚举的每次组合都需要对所有特征函数进行运算并连加。假设训练样本总共 N 条，特征函数数目为 k，每条样本序列长度为$(n+1)$，那么计算所有 M_i 矩阵的算法复杂度为$O\left(knN|\mathbb{Y}|^2\right)$。若使用迭代方法更新参数集$\theta=\{\lambda_{1:k},\mu_{1:k}\}$，则每次参数集更新后都需要重新计算所有 M_i 矩阵的值。另外，无论对训练还是测试过程，都需要计算 $p(y|x)$ 的值。由此可知，必须采用更快速的计算技巧才能使 CRFs 在可容忍的时间内实现。

上下文特征函数主要特点有三个：第一个是二值化，即特征函数的输出为 0、1；第二个是枚举性，即特征函数由枚举产生；第三个特征是模板区别性，由不同特征模板提取出的上下文特征各不相同。

得益于上下文特征函数的三个特点，我们可以设计出适用于计算 M_i 矩阵的快速并行运算。此种算法将数据表达为矩阵形式，同时计算某一条训练序列的在所有特征函数上的输出值，因此能快速地计算 M_i 矩阵。

4.1. 基本步骤

我们先对标签集 $\mathbb{Y}$ 中元素以及各个特征模板得到的所有唯一上下文特征作为编号，然后对所有特征函数建立一个索引表 3。

设一个类型为 Unigram 的特征模板有{1,2,3,4}这些唯一的上下文特征，标签集 $\mathbb{Y}$ 有标号分别为

{1,2,3}，综合将产生 12 个上下文特征函数，排列顺序如表 3 所示。将所有模板产生的上下文特征函数串接，得到一个完整的索引表(表 3)。

出于内存占用量考虑，我们分别处理每个特征模板。假设待处理的 x 序列长度为 n，当前特征模板是模板 t，抽取索引表中由模板 t 产生的特征函数对应部分，并复制 $n+1$ 份后串联拼接。将由模板 t 在所有训练样本中提取到的唯一上下文特征集合记为$\left|\mathbb{X}^t\right|$，集合中元素个数(上下文特征数目)记为$\left|\mathbb{X}^t\right|$，上述过程用图 1 表示。

Table 3. Feature function index paradigm
表 3. 特征函数索引表示例

上下文特征函数索引表							
编号	1	2	3	4	5	6	7
y_{i-1} 编号	N/A	N/A	N/A	N/A	N/A	N/A	N/A
y_i 编号	1	1	1	1	2	2	2
特征编号	1	2	3	4	1	2	3
编号	8	9	10	11	12	…	…
y_{i-1} 编号	N/A	N/A	N/A	N/A	N/A	…	…
y_i 编号	2	3	3	3	3	…	…
特征编号	4	1	2	3	4	…	…

将序列 x 中，通过模板 t 规则提取到的所有上下文特征对照模板 t 中的唯一上下文特征集合进行标记，得到输入序列 x 对于模板 t 上下文特征的索引表，长度为 $n+1$。将这个索引表同样复制$\left|\mathbb{X}^t\right|*\left|\mathbb{Y}\right|*\left|y\right|$份。其中$\left|\mathbb{X}^t\right|$上标表示由模板 t 规则提取。若模板 t 类型为 Unigram，$\left|y\right|=1$；若类型为 Bigram，$\left|y\right|=\left|\mathbb{Y}\right|$。此时特征函数索引的复制、串联过程简化为图 2 所示。

将产生的两条长度为$(n+1)*\left|\mathbb{X}^t\right|*\left|\mathbb{Y}\right|*\left|y\right|$的序列作对比，得到上下文特征编号相同项对应的特征函数索引号。由此索引号查找相对应的模型参数，将提取出来的模型参数按顺序填入一个长度为$\left|\mathbb{Y}\right|*\left|y\right|*(n+1)$的矩阵中。若从输入序列 x 中提取到的上下文特征不被包含在模板的独立特征库里面，则规定用$\left|\mathbb{Y}\right|$个 0 值填入矩阵中。具体过程如图 3 所示。

若当前处理模板为 Unigram 类型，按照上述步骤将获得$\left|\mathbb{Y}\right|*(n+1)$个参数，将这$\left|\mathbb{Y}\right|*(n+1)$个参数值向下复制$\left|\mathbb{Y}\right|$份，变成一个$\left(\left|\mathbb{Y}\right|*(n+1)\right)\times\left|\mathbb{Y}\right|$的矩阵，如图 3 所示。

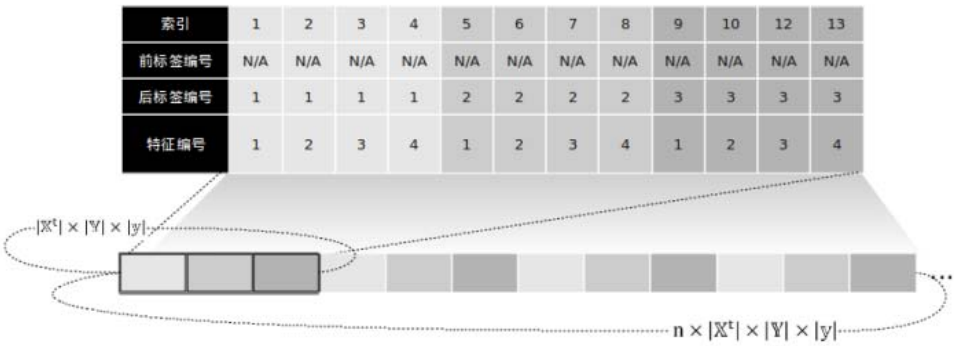

Figure 1. Replication and cascading schematic of Unigram partial feature function index
图 1. Unigram 特征函数索引复制、串联示意图

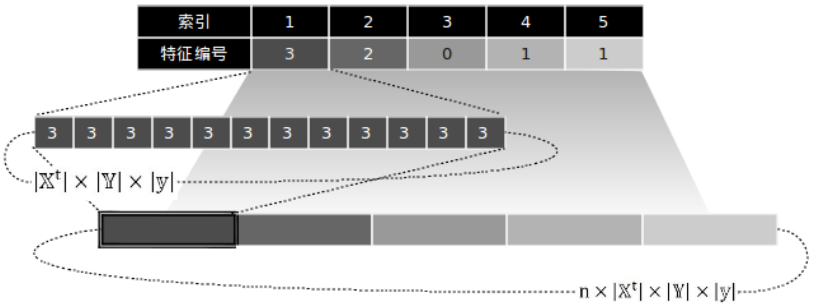

Figure 2. Replication and cascading schematic of Bigram partial feature sequence index
图 2. Bigram 特征序列索引复制、串联示意图

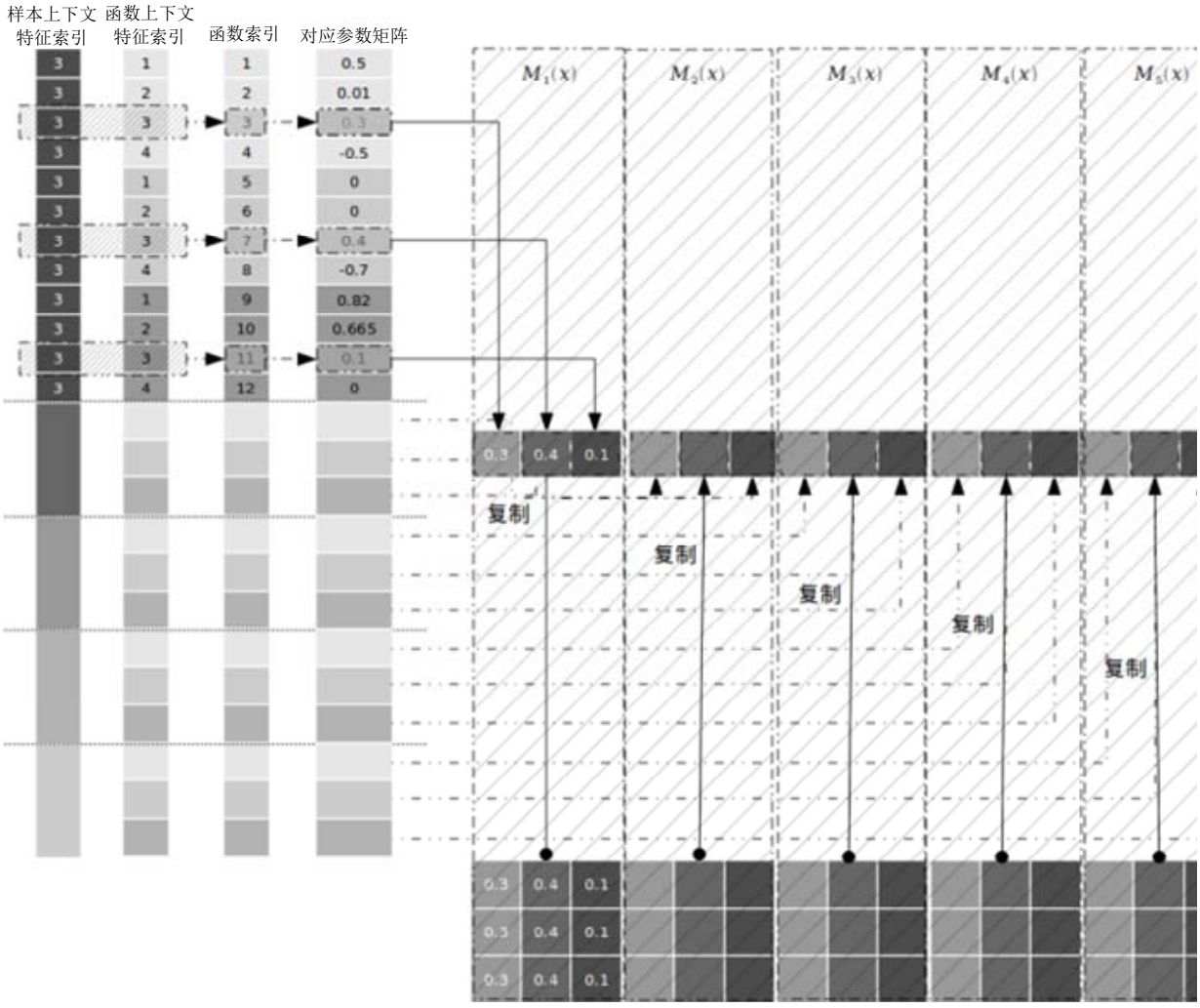

Figure 3. Matrix M producing schematic
图 3. M 矩阵生成示意图

若当前处理的模板类型为 Bigram，将获得 $|\mathbb{Y}|*|\mathbb{Y}|*(n+1)$ 个参数，此时与处理 Unigram 类型模板方法不同，将这条序列每隔 $|\mathbb{Y}|*|\mathbb{Y}|$ 的长度分割成子向量，将每一个子向量按顺序划分成 $1*|\mathbb{Y}|$ 长度的块，再将这些块由上至下排列，使得原来的 $|\mathbb{Y}|*|\mathbb{Y}|$ 向量变形为一个 $|\mathbb{Y}|*|\mathbb{Y}|$ 的矩阵。如此处理，原来的 $1*|\mathbb{Y}|*|\mathbb{Y}|*(n+1)$ 参数向量将可以变成一个 $(|\mathbb{Y}|*(n+1))*|\mathbb{Y}|$ 的矩阵，形式与 Unigram 类型模板作处理后得到的矩阵形式一样。Bigram 模板处理过程如图 4 所示。待当前模板处理完成后，将得到的 $(|\mathbb{Y}|*(n+1))*|\mathbb{Y}|$ 矩阵加上前一次处理模板得到的 $(|\mathbb{Y}|*n)*|\mathbb{Y}|$ 矩阵，直到所有模板处理完成。

将经过上述步骤得到的 $(|\mathbb{Y}|*(n+1))*|\mathbb{Y}|$ 矩阵，横向分割成 $(n+1)$ 个 $|\mathbb{Y}|*|\mathbb{Y}|$ 的矩阵，这 $(n+1)$ 个矩阵即为当前输入序列 x 所对应的 M 矩阵 $M_1, M_2, \cdots, M_{n+1}$。

4.2. 算法复杂度分析

在上述过程中，我们的每一次运算都并行处理某一条 x_{seq} 序列全部节点处的特征，并且每一次都与由某个模板 t 产生的所有特征函数作对比。假设训练样本一共为 N 条，特征模板数目为 T，每条样本序列长度为 $(n+1)$，那么计算所有 M_i 矩阵的算法复杂度为 $O(TNn|\mathbb{Y}|)$。对比在章节 3 开始处提到的传统方法算法复杂度 $O(knN|\mathbb{Y}|^2)$，若在标签集合大小 $|\mathbb{Y}|$ 较小的情况下，两种计算方法的主要区别在于特征函数数目 k 与特征模板数目 T。从章节 3 开始处提到的特征函数生成方法得知，当训练样本较多时，会从一个特征模板中产生出大规模的特征函数，因此 T 值往往比 k

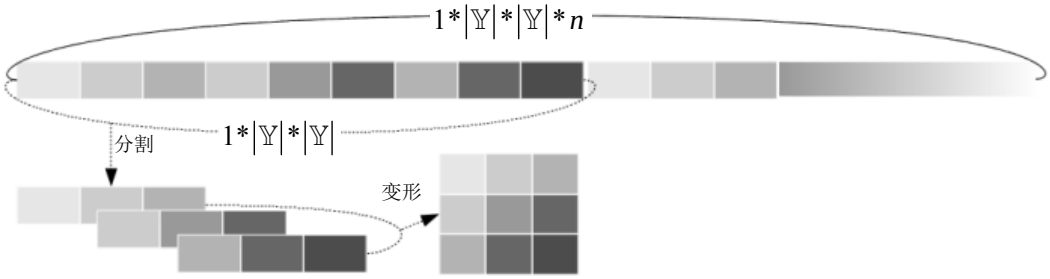

Figure 4. Bigram template processing schematic
图 4. Bigram 模板处理示意图

值要少 3 到 4 个数量级。

5. 测试结果比较及分析

本文采用 MATLAB 编程实现本文提出的快速并行计算方法来计算 M 矩阵的值，并将其嵌入 IIS (Improved Iterative Scaling，改进迭代缩放)[6]程序中。测试数据采用 CoNLL 2000 的浅层语义分析实验数据[7]，这套数据包含 78 条训练样本序列与 823 条测试样本序列。为观察在不同特征函数数目的情况下，本文方法对于速度的提升，我们在实验中使用表 2.1 中不同特征模板的组合。本节实验在普通个人电脑上运行 (I7-2600, 16 GB RAM, Linux)。

5.1. 测试结果比较

实验使用表 1 中不同特征模板组合，具体安排如表 4。

表 5 为不通特征模板组合的条件下，在每一次迭代中，计算全部训练样本的 M 矩阵的平均耗时结果。

从表 5 的实验结果可以看出，在计算 M 矩阵时，本文提出的方法比传统方法在速度上具有明显优势，消耗时间下降倍数达到 1000 的数量级。

6. 总结

通过本文方法可以解决在条件随机场中使用规模上下文特征函数的建模问题，降低运算成本，使得条件随机场可以更广泛地应用到各个领域当中，具有非常大的实用价值。

在下一步工作中，我们将探讨如何选择质量更高的特征函数，使得条件随机场建模的运算量进一步降低。

Table 4. Combinations of feature template
表 4. 特征模板组合

编号	模板组合
1	U15-U18
2	U20-U22
3	U15-U18, U20-U22
4	U00-U6,U10-U18,U20-U22

Table 5. Average time cost with different feature template combinations
表 5. 不同特征模板组合的平均耗时测试结果

模板组合编号	特征函数数目	本文方法耗时(s)	普通方法耗时(s)
1	21024	1.245 ± 0.660	1260.544 ± 1.522
2	42640	2.147 ± 0.117	2543.067 ± 0.422
3	63664	3.173 ± 0.842	4011.712 ± 0.165
4	175120	30.346 ± 0.082	N/A

参考文献 (References)

[1] J. Lafferty, A. McCallum and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. Proceedings of the Eighteenth International Conference on Machine Learning, 2001: 282-289.
[2] F. Sha, F. Pereira. Shallow parsing with conditional random fields. Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, 2003, (1): 134-141. http://www.cnts.ua.ac.be/conll2000/
[3] C. M. Bishop. Pattern recognition and machine learning. New York: Springer, 2006: 359-418.
[4] A. McCallum, D. Freitag and F. Pereira. Maximum entropy markov models, 2000.
[5] TakuKudo, CRF++ toolkit, 2005. http://crfpp.sourceforge.net/
[6] S. Della Pietra, V. Della Pietra and J. Lafferty. Inducing features of random fields. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1997, 19(4): 380-393.
[7] F. Erik, T. K. Sang and S. Buchholz. Introduction to the CoNLL-2000 Shared Task: Chunking. Proceedings of CoNLL-2000 and LLL-2000, 2000.