统计语言模型综述*)

邢永康　马少平
(清华大学计算机系智能技术与系统国家重点实验室　北京100084)

A Survey on Statistical Language Models

Xing Yong-Kang　Ma Shao-Ping
(The State Key Lab. of Intelligent Tech. and System, Tsinghua University, Beijing 100084, China)

Abstract The Statistical Language Model (SLM) is a distribution to capture the generation rule in natural language. Since the first model was proposed in 1980, the SLM was used in many application such as speech recognition, optic character recognition, machine translation and etc. Recently the information retrieval based-on SLM is becoming a new direction in IR community. In this paper, we introduce the primary theory and current method about the SLM, and point out several promising research direction.

Keywords Statistical language model, Smoothing method, n-Gram model, Decisio tree model, Maximum entropy model

1 引言

统计语言模型产生于基于统计方法的自然语言处理系统的研究中:如语音识别系统、字符识别系统以及机器自动翻译系统等。对于一个语音识别系统,给定语音信号 a 和语言的句子集合 S,则系统需要解决的问题可以表示为:

$$\arg\max_{i\in S}(P(s|a)) \tag{1}$$

即确定概率值最大的句子 s(由单词构成的序列)作为识别结果。根据 Bayes 公式:

$$\arg\max_{i\in S}P(s|a)=\arg\max_{i\in S}P(a|s)P(s)/P(a) \tag{2}$$

其中,信号波形的先验概率 $P(a)$ 与 s 的选择无关,可以不计算;$P(a|s)$表示句子与信号的对应关系(如在英语中,每个句子和它的声音波形的对应关系),它由信号处理过程决定,称为采样模型;$P(s)$称为语言模型,在这里它表示语言中句子的分布概率。

因此,统计语言模型就是表示语言基本单位(词、词组、句子等,我们一般将句子看作语言的基本单位)的分布函数,它描述了该语言的基于统计的生成规则。一般来说,任何一个自然语言的单词量都非常大,而且句法复杂,构成的句子数目也将非常巨大,要表示出所有句子的概率就空间复杂性来说是不可能的。所以,一般语言模型的实质都是将句子的概率分解为各个单词的条件概率的乘积,即:

$$P(s)=P(w_1,w_2,\cdots,w_{n-1},w_n)=\prod_{i=1}^{n}P(w_i|h_i) \tag{3}$$

其中:n--句子 s 的长度;$h_i=(w_1,w_2,\cdots,w_{i-1})$--句子中单词 w_n 前面 $i-1$个单词构成的序列,称为单词 w_n 的上下文。

因此,我们设语言的单词集合为 $V=\{w_1,w_2,\cdots,w_n\}$,所有上下文构成的集合为 $X=\{(w_1,w_2,\cdots)|w_i\in V\}$,则一个语言模型 M 就是要给出语言中的每个单词对于各种上下文组合的条件概率,即:$P(y|x)$,其中 $x\in X$,$y\in V$。本文中,我们称 x 为上下文变量,y 为预测变量。

评价统计语言模型性能的最直观的指标是在测试语料集上计算的交叉熵函数:

$$H(P_T;P_M)=-\sum_{s\in T}P_T(s)\log P_M(s) \tag{4}$$

其中:T--测试语料集;$P_T(s)$--测试语料集 T 的真实语言模型;$P_M(s)$--从学习语料集上学习建立的语言模型 M。

交叉熵的值越大,则学习模型与真实模型的差别也越大,表明模型的效果越差。由于测试语料集的真实语言模型实际上无法获得,因此也常采用以下的公式评价语言模型的性能:

$$H(P_T;P_M)=(-\sum_{s\in T}\log P_M(s))/|T| \tag{5}$$

另一个最常用的评价指标是模型 M 的分支均值(perplexity):

$$Perplexity(P_T,P_M,T)=2^{H(P_T;P_M)} \tag{6}$$

该指标可以简单理解为,在模型 M 所表示的语言中,等概率出现在一个单词后面的单词的平均数目。因此它既可以用来评价语言模型的质量(该值越小,模型越好),也可以用来表示语言的复杂性(该值越大,该语言越复杂)。

本文介绍了应用很广的 n-grams 模型;给出了几种常用的平滑化方法;介绍了决策树模型;分析了最大熵模型;最后指出了统计语言模型中几个有潜力的研究方向。

2 n-Gram 统计语言模型

n-gram 模型于1980年提出来,是一种应用很广的统计语言模型。它采用了 Markov 假设,即认为每个预测变量只与长度为 $n-1$的上下文有关,即:

$$P(w_m|w_1,w_2,\cdots,w_{m-1})=P(w_m|w_{m-(n-1)},w_{m-(n-2)},\cdots,w_{m-1}) \tag{7}$$

如果用 w_i^j 表示单词串 $w_i,w_{i+1},\cdots,w_{j-1},w_j$,则上式可以简化表示为:

$$P(w_m|w_1^{m-1})=P(w_m|w_{m-(n-1)}^{m-1}) \tag{8}$$

*)本项目受到国家重点基础研究(973)(G1998030509)、自然科学基金项目(60223004)以及863高科技项目(No. 2001AA114082)资助。邢永康　博士,主要研究领域为机器学习,模式识别,网络数据挖掘。马少平　博士,教授,博士导师,主要研究领域为模式识别,信息检索,网络数据挖掘。

(8)式中参数 n 称为模型的阶数，其取值决定了模型的精度和复杂性。试验表明，n 值越大，则对单词之间的依赖关系的描述越准确，即模型的精度越高，但模型的复杂性也越高。因此合适的 n 值是在模型的精度和复杂性之间的一种折衷，一般为 $1 \leqslant n \leqslant 7$。其中 n＝1，2，3，分别称为 Unigram、Bigram 及 Trigram 模型。

可以看出，n-grams 模型描述的语言是一种有限状态的正则文法产生的语言。它与灵活多变的自然语言之间存在着明显的差别。然而该模型在实际应用中却取得了较大的成功，原因就在于它成功地捕捉到了自然语言中存在的局部约束(Local constraint)性质。如对包含139，000，000个单词的语料库[1]的统计中，$P(\text{“}gates\text{”})=0.00001$，而 $P(\text{“}gates\text{”}|\text{“}bill\text{”})=0.004$，$P(\text{“}gates\text{”}|\text{“}chairman\text{”},\text{“}bill\text{”})=0.47$。可以看出，每增加一个限定词，其概率值就要提高两个数量级。也正因为这个原因，该模型无法表示语言中存在的远程约束(Remote constraint)，这些约束大都源于句子结构、语义关系等。

基于训练语料集建立 n-grams 模型，一般采用最大似然法 (Maximize Likehood)，即：

$$P_{ML}(w_m|w_{m-(n-1)}^{m-1})=c(w_{m-(n-1)}^{m-1},w_m)/c(w_{m-(n-1)}^{m-1}) \quad (9)$$

(9)式中 $c(w_{m-(n-1)}^{m-1})$ 表示语料集中单词串 $w_{m-(n-1)}^{m-1}$ 的出现次数。

然而，该方法存在一个问题，即可能存在某个 n-gram(连续 n 个单词构成的串)，它在学习语料集中没有出现，而可能出现在测试语料集中。如在文[2]的实验中，对一个包含242，000，000单词的语料库，采用最大似然法建立了一个基于60，000个单词的 trigram 模型。当将该模型用于实际的测试语料集时，发现测试集中只有69%的 n-gram 在学习集中出现的次数大于1次。该问题称为数据的稀疏问题。而且，阶数 n 的值越大，该问题也越突出。如果简单地根据最大似然法，我们会得到该 n-gram 的概率值为0。这种判定明显过于武断，对模型的精度影响很大。一般来说，单纯地增加训练语料集的规模，无法解决该问题。因此采用各种方法，对这些没有出现在学习语料中的 n-gram 估计一个不为0的值，就成为 n-gram 语言模型研究中的一个主要问题。

3 几种平滑处理方法

处理数据稀疏问题的技术统称为平滑化(Smoothing)方法。这些方法可以分为两类：一类是直接对最大似然法的估计结果进行修整，包括插值法、折扣法以及回退法等；另一类是通过对单词的聚类减小模型空间来解决数据的稀疏问题，包括各种聚类算法。

Good-Turing 方法，又称为折扣最大似然法，是一种应用比较广泛的折扣法。它通过对最大似然法的结果进行调整，可以在保证满足概率归一性质的条件下，估计出在训练语料中没有出现的 n-gram 的概率值。它分为两个步骤来完成。

第一步是估计没有在学习语料中出现的 n-gram 的概率值。首先假设学习语料的单词数目为 N，且其中出现 r 次的 n-gram 的数目为 N_r，则存在下面的等式：

$$\sum_r rN_r=N \quad (10)$$

对于没有出现在学习语料中的 n-gram，采用如下的公式估计它们的概率值：

$$\sum_{w_1^n:c(w_1^n)=0} P_{GT}(w_1^n)=N_1/N \quad (11)$$

该估计的基本思想是：在所有 n-gram 的中，存在一个由大量的特殊的 n-gram 构成的集合。该集合中的每一个 n-gram 在学习集中要么不出现，要么只出现一次。因此，可以用学习集中出现的这部分 n-gram 占有的比例来估计该特殊集合在所有 n-gram 中的比例。

根据最大似然估计法，对于所有出现次数为 r 的 n-gram，它们的概率和为：

$$\sum_{w_1^n:c(w_1^n)=r} P_{ML}(w_1^n)=\frac{r}{N}+\frac{r}{N}+\cdots+\frac{r}{N}=\frac{rN_r}{N} \quad (12)$$

因此，对于所有的 n-gram(包括出现次数 r＝0)，其概率和为：

$$\sum_{w_1^n:c(w_1^n)=0}^{n} P(w_1^n)+\sum_{w_1^n:c(w_1^n)=1}^{n} P_{ML}(w_1^n)+\cdots+\sum_{w_1^n:c(w_1^n)=r} P_{ML}(w_1^n)+\cdots \quad (13)$$

将式(11)和(12)代入(13)得：

$$=\frac{N_1}{N}+\frac{N_1}{N}+\frac{2N_2}{N}+\cdots+\frac{rN_r}{N}+\cdots=\frac{N_1}{N}+\frac{\sum_r rN_r}{N}$$

$$=1+\frac{N_1}{N} \quad (14)$$

这明显违背了概率的归一化性质。产生的原因是我们为那些没有出现的 n-gram 估计出了一个不为0的概率值(见公式(11))。要解决该问题，只有减少那些出现了的 n-gram 的概率值，即：

$$P_{GT}(w_1^n)=\sigma P_{ML}(w_1^n) \quad 0<\sigma<1 \quad (15)$$

其中 σ 称为折扣系数，这就是该方法称为折扣法的原因。简单的方法是对所有的概率做归一化处理，这样会平均减小所有的 n-gram 的概率值。而 Good-Turing 方法则是根据各个 n-gram 的出现次数 r 来确定折扣系数 σ_r，即：

$$r^{new}=(r+1)N_{r+1}/N_r \quad (16)$$

$$\sigma_r=r^{new}/r \quad (17)$$

可以证明，经过调整后，所有 n-gram 的概率和将满足归一化性质：

$$\sum_{w_1^n:c(w_1^n)>0}^{n} P_{GT}(w_1^n)=\sum_{w_1^n:c(w_1^n)>0}^{n} \sigma_r P_{GT}(w_1^n)=\sum_{w_1^n:c(w_1^n)>0} \frac{(c(w_1^n)+1)\frac{n_{c(w_1^n)+1}}{n_{c(w_1^n)}}}{c(w_1^n)}\times\frac{c(w_1^n)}{N}=\sum_{r>0}^{n}\frac{(r+1)\frac{N_{r+1}}{N_r}}{r}\cdot$$

$$\frac{rN_r}{N}=\sum_{r>0}\frac{(r+1)N_{r+1}}{N}=1-\frac{N_1}{N}$$

Good-Turing 方法的优点是它可以对训练语料中没有出现的 m-gram 直接估计出一个概率值。因此在平滑化处理中被广泛使用。可以看出，随着模型阶数 n 的增加，数据稀疏问题会越来越严重。因此，利用低阶的模型来近似计算高阶模型也是一类经常使用且比较直观的平滑化方法。

插值法[4]就是通过插值技术，将一个 n-grams 模型表示为由1阶直到 n 阶模型的线性组合，即：

$$P_{INT}(w_1^n)=\alpha_1 P_{ML}(w_1^n)+\alpha_2 P_{ML}(w_2^n)+\cdots+\alpha_{n-1} P_{ML}(w_{n-1}^n)+\alpha_n P_{ML}(w_n) \quad (18)$$

其中，参数 α_i 满足条件 $\sum_{i=1}^{n}\alpha_i=1$。关于参数值的确定，一种是理论计算。做法是在测试语料集上计算模型的分支均值 Perplexity(M)，并取使该值最小的参数值；另外一种方法是试验调整，对于具体的应用系统(如语音识别系统、字符识别系统等)，可以通过对测试集的反复测试，确定使错误率最小的参

数值。

Kaze 的回退法(Backoff)[3]也是一种常用的平滑化方法。与插值法不同,它将每一个 *n-gram* 模型表示为 *m-gram* 的非线性组合,其中 $m=1,2,\cdots,n$。对于每一个 *m-gram* 模型,由一个回退概率 β_m 表示由 *m-gram* 模型回退到 $(m-1)$-*gram* 模型的概率,也可以理解为在学习语料集中,给定上下文($m-1$)-*gram*,*m-gram* 不出现的概率。因此存在如下的递推公式:

$$P_K(w_n|w_1^{n-1})=P_{GT}(w_n|w_1^{n-1})+\beta_n P_K(w_n|w_2^{n-1})$$
$$P_K(w_n|w_2^{n-1})=P_{GT}(w_n|w_2^{n-1})+\beta_{n-1} P_K(w_n|w_3^{n-1})$$
$$\cdots \qquad (19)$$

其中 $P_{GT}()$的表示采用 Good-Turing 法确定的概率值,给定学习语料库,它可以被预先计算出来。因此,确定回退概率是该方法的主要任务。根据对回退概率的理解,包含两种情况:

1. 如果一个 *m-gram* 出现在学习语料库中,即 $c(w_1^m)>0$,则回退概率应该为 $\beta_m=0$;2. 如果一个 *m-gram* 没有出现在学习语料库中,即 $c(w_1^m)=0$,则需要按照下式计算回退概率:

$$\beta_m=\upsilon(w_1^m)/[1-\omega(w_1^m)] \qquad (20)$$

其中:$\upsilon(w_1^m)=\sum_{w_1^m:c(w_1^m)>0} P_{GT}(w_m|w_1^{m-1})$;

$$\omega(w_1^m)=\sum_{w_1^m:c(w_1^m)>0} P_{GT}(w_m|w_2^{m-1})$$

4 决策树语言模型(Decision Tree Modeling)

分析可以看出,*n-gram* 模型中存在两个相互矛盾的问题:一方面由于模型复杂性的制约,实际中一般只采用很短的上下文,长度 n 的值一般为2～7,因此用于预测的上下文信息太少,我们可以称其为上下文有限问题。另一方面,在 *n-gram* 模型中,对于一个预测变量,需要考虑长度为 $n-1$ 的所有上下文的组合,然而在实际应用中,可能上下文中的某些单词对预测变量并没有关系。这些上下文的存在,一来会增加模型的存储复杂性,更重要的是可能会对预测变量的计算产生干扰。我们将此称为上下文过多问题。解决这两个问题的一个直接想法就是,如何对上下文信息进行简化,使其满足两个条件:1)尽可能地保留那些与预测变量相关的单词,而不管该单词到预测变量距离的远近;2)尽可能地从上下文中删除那些与预测量无关的单词,以减少造成的干扰。这就是决策树语言模型的基本思想。

给定训练语料集 T(可以整理成表示为一系列有序对(x,y)的出现次数,其中 $x\in X, y\in V$),建立在它上面的决策树模型是一个树形结构,它包括两类结点:

1)内结点:含有子结点的结点。每一个内结点 in_i 包含有两部分(q_i,T_i),q_i 表示该结点对应问题;T_i 表示该结点对应的训练子集。根结点是一个特别的内结点。它的训练子集就是训练集 T。根据根结点对应问题的 m 个回答形成该结点的 m 个子结点,并将训练集 T 划分到 m 个结点。以此类推,可以获得每个结点对应的训练子集。

2)叶结点:是没有子结点的结点。在每个叶结点 ln_i 包含两部分$(P_i(y|x),T_i)$。T_i 表示该结点对应的训练子集,$P(y|x)$为该结点对应的概率函数,它是通过对该结点的训练 T_i 学习得到的。

例如,通过在每个内结点设置问题:"前面的一个单词是什么?",就可以将一个 *n-gram* 模型对应地表示为一个决策树模型,如图1所示。该决策树模型的每个内结点包含 $|V|$ 个子结点,且决策树的高度为 $n-1$。最后一层全部为叶结点,其上包含有每个预测变量在对应的上下文条件下的概率。当然,该决策树是对 *n-gram* 模型的直接表示,它表示预测变量的出现概率与且仅与所有长度为 $n-1$ 的上下文相关,因此它并没有利用到决策树的优点。如果我们能够在每个节点设置一个"有效"的问题,那么就可以从所有的上下文中,只选出那些与预测变量相关的上下文,并且包括长度大于 $n-1$ 的上下文。这样就可以解决 *n-gram* 模型存在的上下文有限和过多的问题。

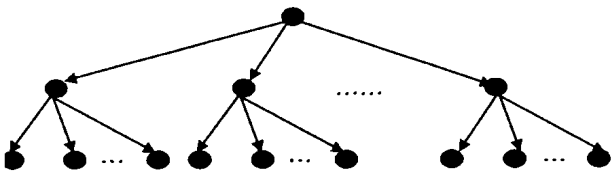

图1 *n-gram* 模型可以表示为一个特殊的决策树

建立决策树语言模型的关键在于:1)如何选择适当的问题集合 $Q=\{q_1,q_2,\cdots,q_i,\cdots\}$。2)给定了问题集合,如何确定"合适"的问题序列来建立决策树。由于决策树的目的是尽可能从上下文集合中只找出那些与预测变量 y 相关的上下文,因此一种最直观的思想就是,可以将建立决策树的过程看作是通过选择问题序列来降低变量 y 的熵过程。基于这种思想,我们给出如下的决策树模型建立算法[5,6]。

算法1:决策树语言模型生成算法

输入:训练语料 T 及问题集合 $Q=\{q_1,q_2,\cdots,q_i,\cdots\}$

输出:决策树语言模型

过程:

根据对树递归定义的特点,我们对每个结点 n_i(开始时为根结点)作如下的处理:

1 在结点的训练语料 T_i 中(对于根结点,训练语料为 T),如果所有的序对都满足如下的条件:

$$\exists\, y'\in Y:\forall\,(x,y)\in T_i, y=y'$$

则定义该结点为一个叶结点。并计算出所有预测变量的概率值。算法结束。

2 对于每一个问题 $q_i\in Q$,基于该结点的训练语料 T_i,计算变量 y 的条件熵:即 $H(Y|q_i)$。并从中选出条件熵最小的问题,假设为 q_i。

3 如果条件熵 $H(Y|q_i)$ 没有降低,则该结点是一个叶结点,并基于该结点的训练语料 T_i 估计该结点的概率分布。算法结束;否则利用根据对问题 q_i 的 m 个答案,为该结点生成 m 个子结点。并根据对问题 q_i 的答案,将训练语料 T_i 划分到各个子结点的训练语料。

4 分别对每个子结点重复以上1～3的过程。

确定问题集合 Q 对于建立决策树模型也非常重要。为了降低模型的复杂性,要求尽可能选择那些答案数目比较少的问题。一般要求答案为"是与否"两种情况,从而建立的决策树是一个二叉树结构。

决策树模型尽管部分克服了 *n-gram* 模型存在的上下文限制和上下文过多的问题,但在决策树模型的建立过程中存在一个问题:随着决策树高度的增加,每个结点的训练语料的规模将越来越小。而训练语料越小,参数估计的精度就越低。该问题称为数据碎化(Data Fragmentation)问题。可以看出,该问题在 *n-gram* 模型中也存在。只要增加上下文的长度,就等于对数据增加了限制条件,必然会减少数据,因此该问题很

难彻底解决，但可以通过前面介绍的平滑化方法减少它对参数精度的影响。

5 最大熵模型(Maximum Entropy Modeling)

最大熵模型(又称指数模型)可以解决数据碎化问题，是目前研究的一个热点模型。其基本思想源于将统计语言问题看作是一个求解受限的概率分布问题[8]。

给定训练语料 T，建立统计语言模型就是要确定概率分布 $P(y|x)$，其中 $x\in X, y\in V$。我们可以通过描述(x,y)中的特征，在(X,Y)空间上定义了一个等价类。比如要求上下文 x 的最后两个单词是"Chainman Bill"，并且 y="Gates"，就可以得到一个满足该条件的(x,y)的集合。该集合可以用一个定义在(X,Y)上的标志函数(0-1 函数)来表示，称为特征函数：

$$f_i(x,y)=\begin{cases}1 & (x,y)\in R\\ 0 & (x,y)\notin R\end{cases} \tag{21}$$

利用给定的训练语料 T，可以计算出特征函数 $f_i(x,y)$ 的经验期望值(用 E_i 表示)为：

$$E_i=\sum_{(x,y)}\tilde{P}(x,y)f_i(x,y) \tag{22}$$

其中，$\tilde{P}(x,y)$--由训练语料 T 计算出的经验分布。

根据该特征函数及其经验期望，我们可以对概率分布 $P(y|x)$引入如下的约束条件：

$$\sum_{(x,y)}P(x,y)f_i(x,y)=E_i \tag{23}$$

即要求模型给出的特征函数的期望一定要等于经验期望值。由于我们只需要条件概率，而不是连和概率，因此该约束式可以重写为：

$$\sum_{(x,y)}\tilde{P}(x)P(y|x)f_i(x,y)=E_i \tag{24}$$

其中，$\tilde{P}(x)$--根据训练语料 T 建立的经验分布。

以此类推，可以设定多个特征函数来建立对概率分布 $P(y|x)$的约束，从而将问题转化为给定多个约束条件下求解概率分布 $P(y|x)$。根据最大熵理论，满足该约束条件的分布必须尽可能地接近一个均匀分布，也就是说，该分布的熵必须最大。可以证明[8]：以下的指数分布满足该条件：

$$P(y|x)=\left(\prod_i u_i^{f_i(x,y)}\right)/z_\lambda(x) \tag{25}$$

其中：$u_i=e^{\lambda_i}$，$z_\lambda(x)=\sum_y\prod_i u_i^{f_i(x,y)}$

该结果也可以用最大似然原理推导出来，即可以证明：在满足约束条件的所有概率分布中，指数分布式(25)对于训练语料集 T 的似然函数值最大。

算法2：最大熵模型学习算法[7,8]

输入：训练语料 T；一批特征函数—经验期望值对$(f_i(x,y),K_i)$。

输出：统计语言模型 $P(y|x)$，$x\in X, y\in V$。

过程：

1. 初始化：首先为模型指定参数，可以采用均匀分布。
2. 反复进行如下的操作，直到模型收敛为止：

1) 根据当前的参数值，计算每一个特征函数的期望值，即：

$$K_i^{(new)}=\sum_{(x,y)}\tilde{P}(x)P^{(old)}(y|x)f_i(x,y) \tag{26}$$

2) 将新的期望值与经验期望值进行比较并修正参数，即：

$$u_i^{(new)}=u_i^{(old)}K_i/K_i^{(new)} \tag{27}$$

3) 基于新的参数，重新估计概率函数的值，即：

$$P^{(new)}(y|x)=\frac{\prod_i(u_i^{(new)})^{f_i(x,y)}}{z_\lambda^{(new)}(x)} \tag{28}$$

从算法2可以看出，最大熵模型的计算是在整个训练语料 T 上展开的，因此有效地避免了以前模型中存在的数据碎化问题；而且，特征函数的定义非常灵活，它可以任意建立上下文 x 与预测变量 y 之间的关系，因此，可以通过引入一些揭示语义关系的特征函数，部分地实现基于语义的语言模型。可以看出，建立该模型的主要任务在于定义特征函数，因此如何自动定义特征函数是一个主要研究任务；另外，该模型学习算法是循环算法，其计算量相当大，因此，研究高效的收敛算法也是当前的一个主要研究任务。

总结 分析统计语言模型的研究现状，可以看出，统计语言模型的研究目前正在、而且还将继续从以下几个方向展开[10]：

1. 结合语言知识的统计语言模型研究。从前面介绍的几个模型来看，几乎都没有涉及语言本身的知识，只是通过对单词之间的统计关系的挖掘，来表示语言模型。尽管这些模型在某些应用领域表现不错，但这种先天不足最终会表现出来。因此研究基于语言知识的统计模型将是一个必然的方向。这方面的模型如概率上下文无关文法模型，它基于传统的上下文无关文法，为每一个非终结符扩展式加上一个统计概率，来使其具有统计的特性。

2. 提升语言模型的基本单位(模型粒度)。目前的模型一般都是通过分解，将一个句子的概率分解为单词的条件概率，因此，模型表示的最小单位是单词。我们知道，语言中的许多结构：如人称与动词的数的一致，语义相关等很难基于单词之间的概率关系来表示。因此，建立基于句子，甚至段落的语言模型，将是一个研究方向。

3. 降维技术及高效降维算法的研究。在一般的语言模型中，由于没有考虑各个语言元素之间的归类关系，所以造成变量的维数很大。直觉告诉我们，这些语言元素之间完全可以进行归类。因此，如何引入聚类技术，来降低整个模型的维度，但又不降低模型精度，就是一个有意义的研究方向。试验表明，绝对聚类(即每个语言元素属于一个且只有一个特定的类)往往是以降低模型的精度为代价的。因此，我们认为研究基于概率或模糊逻辑的聚类算法，是可望取得突破的一个方向。

4. 在模型中加入人的知识的研究。目前的模型大都是基于训练数据的分析来确定模型参数，可以称之为数据驱动的模型，因此不可避免地会出现数据不足的问题，而加入人的知识是克服该问题的一个可行方法。但人的知识往往表现得不够严谨，经常会出现一些例外情况，因此基于 Bayes 理论的修正系统是一个值得研究的方向。通过将人的知识表示为一个先验的概率，既可以避免数据不足造成的错误估计，也可以通过数据学习，对不够精确的人的知识进行修正。

参考文献

1 Graff D. The 1996 broadcast news speech and language-model corpus. Slides from lecture at the 1997 DARPA Speech Recognition Workshop, Feb. 1997

2 Rosenfeld R. A maximum entropy approach to adaptive statistical language modeling. Computer Speech and Language, 1996, 10: 187～228

3 Katz S M. Estimation of probabilities from sparse data for the language model component of speech recognizer. IEEE Transactions on Acoustics, Speech and Signal Processing, 1987, ASSP-

35：400～401
4 Jelinek F，Mercer R L. Interpolated estimation of Markov source parameters from sparse data. In：Proc. of the Workshop on Pattern Recognition in Practice，Amsterdam，The Netherlands：North-Holland，May 1980，381～397
5 Magerman D M. Natural Language Parrsing as Statistical Pattern Recognition：[PhD Thesis]. Stanford University，1994
6 Bahl L R，Brown P F，De Souza P V，Mercer R L. A tree-based statistical language model for natural language speech recognition. IEEE Transactions on Acoustics，Speech，and Signal Processing. 1989，37(7)：1001～1008
7 Rosenfeld R. Adaptive Statistical Language Modeling：A Maximum Entropy Approach：[PhD thesis]. Carnegie Mellon University，1994. CMU Technical Report CMU-CS-94-138
8 Darroch J，Ratcliff D. Generalized iterative scaling for log-linear models. The annals of Mathematical statistics 1972，43：1470～1480
9 Berger A L，Della Pietra S A，Della Pietra V J. A maximum entropy approach to natural language processing. Computational Linguistics 1996，22(1)：39～71
10 Rosenfeld R. Two decades of Statistical Language Modeling：Where Do We Go From Here? Proceedings of the IEEE，2000，88(8)

（上接第13页）

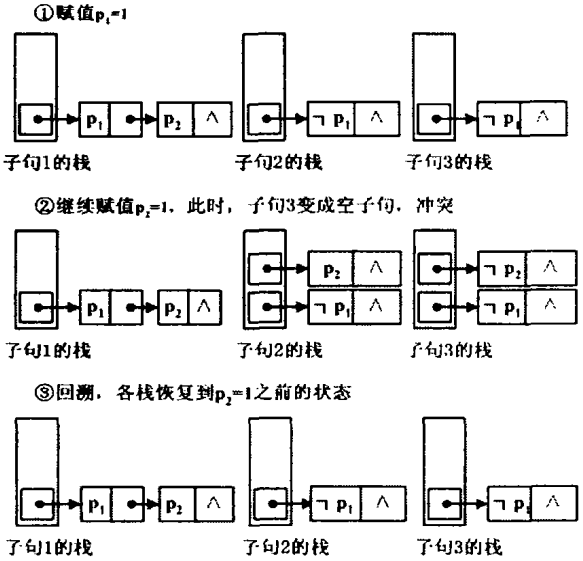

图4 子句栈的变化

总结 实验结果表明，我们开发的 QuickSAT 接近了国际先进水平。虽然 QuickSAT 还不是世界上最高效的，但我们相信这主要是因为我们有意地放弃了许多先进技术的使用，而这些技术已经被证明了能有效提高 SAT 算法的运行性能。我们的本意只是要通过这个程序的开发，来例证算法工程的原则和方法的有效性。这一点，我们相信已经做到了。

当然，算法工程的理论与方法的发展，还需要研究者们不断地努力，要提出更多行之有效的方法，要对各种方法进行进一步的论证，还要朝着系统化、形式化的方向发展。我们在这里的讨论，只是起到“抛砖引玉”的作用罢了。

参 考 文 献

1 Aho A V，Johnson D S，Karp R M，et al. Theory of Computation：Goals and Directions. STOC 1996，March 15，1996
2 Cohen P，Gent I P，Walsh T. Empirical Methods for CS and AI. given at AAAI，ECAI and Tableaux Conf. in 2000
3 Davis M，Putnam H. A Computing Procedure for Quantification Theory. Journal of the ACM，1960，7(3)：201～215
4 Davis M，Logemann G，Loveland D. A Machine Program for Theorem Proving. Communications of the ACM，1962，5(7)：394～397
5 Marques-Silva J P，Sakallah K A. GRASP：A Search Algorithm for Propositional Satisfiability. IEEE Transactions on Computers，1999，48(5)：506～521
6 Moskewicz M W，Madigan C F，Zhao Y，et al. Chaff：Engineering an Efficient SAT Solver. In：Proc of the 38th Design Automation Conference (DAC'01)，June 2001，530～535

附表：

基准测试集	测试实例组	组中实例数目	GRASP[1]		zChaff[2]		QuickSAT	
			超时次数	总运行时间	超时次数	总运行时间	超时次数	总运行时间
DIMAC 基准测试集[3]	ii8	14	0	1.4	0	0.1	0	0.1
	ii16	10	1	326.8[6]	0	7.6	0	12
	ii32	17	0	2.7[6]	0	0.6	0	0.6
	aim100	24	0	0.7[6]	0	0.1	0	0.1
	aim200	24	0	7.9	0	0.3	0	0.3
	pret	8	0	7.2[6]	0	0.8[5]	0	0.6
	par8	10	0	0.1[6]	0	0.1	0	0.1
	par16	10	7	1195.7[6]	0	54.9	0	91.8
	ssa	8	0	3.2[6]	0	0.3	0	1.1
	jnh	50	0	6.9[6]	0	0.7	0	0.7
	dubois	13	0	0.3	0	0.2	0	0.3
	hole	5	2	299.9[6]	0	128.7	0	134.4
CMU 基准测试集[4]	SSS 1.0	48	5	1084	0	62	0	85
	SSS 1.0a	8	6	9190	0	25	0	37
	SSS-SAT 1.0	100	100	-	0	632	7	8931
	PVP-UNSAT 1.0	4	2	2944	0	1033	0	1477
	VLIW-SAT 1.0	100	100	-	0	4665	4	9617

注：(1)GRASP 使用的是2000年2月的版本，其源程序和 Linux 二进制版本可以从 http://sat.inesc.pt/～jpms/grasp/下载；(2)zChaff 使用的是2001年的版本，源程序或 Linux 二进制代码可以从 http://ee.princeton.edu/～chaff/zchaff.php 下载；(3)超时时限为100s；(4)超时时限为1000s；(5)对于本组测试实例，将 zChaff 配置中的 maxLitsInClaseForConfDriven 由默认的10000改成10，除此之外，其他情况 zChaff 都是用 cherry.smj 中的默认配置；(6) GRASP 运行使用如下开关参数：+ T100 +B10000000 +C10000000 +S10000 +g20 +rt4 +dDLIS，其他情况下 GRASP 都使用以下开关参数：+T100 +B10000000 +C10000000 +S10000 +V0 +g40 +rt4 +dMSMH +dr5。