轻工学报 2016 年 7 月 第 31 卷 第 4 期
JOURNAL OF LIGHT INDUSTRY Vol. 31 No. 4 Jul. 2016

引用格式:方刚. 基于统计语言模型及动态规划算法的蛋白质表达载体的优化设计[J]. 轻工学报,2016,31(4):88 - 94.
中图分类号:TP319.9 **文献标识码**:A
DOI:10.3969/j.issn.2096 - 1553.2016.4.013
文章编号:2096 - 1553(2016)04 - 0088 - 07

基于统计语言模型及动态规划算法的蛋白质表达载体的优化设计

Protein expression vector optimization based on statistical language model and dynamic programming

方刚
FANG Gang

西安文理学院 生物与环境工程学院,陕西 西安 710065
School of Biological and Environmental Engineering, Xi'an University, Xi'an 710065, *China*

关键词:
统计语言模型;动态规划算法;蛋白质表达载体;合成生物学标准"零件"

Key words:
statistical language model (SLM); dynamic programming; protein expression vector; BioBrick

摘要:针对合成生物学基因片段组装中选择最优"零件"组装功能性蛋白质表达载体费时且易出错的问题,提出一种基于引入统计语言模型(SLM)与动态规划算法的蛋白质表达载体设计方法. 该方法通过统计合成生物学标准"零件"(BioBrick)的参数,将基础"零件"组装过程转化为 SLM,用动态规划算法找到最优路径,以实现蛋白质表达载体的设计. 实验结果证明该方法准确率高,可以减少真实装配过程的冗余操作,节省时间和费用,可用来优化其他合成生物学软件设计结果,也可独立使用来模拟装配合成生物学基因片段产生蛋白质表达载体,还可被迭代从而给出不同的优化结果供选择.

收稿日期:2016 - 04 - 13
基金项目:国家自然科学基金项目(61173113)
作者简介:方刚(1969—),男,陕西省西安市人,西安文理学院副教授,博士,主要研究方向为生物计算、生物信息学.

Abstract: In order to solve the problem of time consuming and error pronein selecting optimal "brick" to assemble functional protein expression vector, based on statistical language model (SLM), a dynamic programming algorithm of protein expression vector was carried out. By collecting the statistical parameters of BioBrick standard parts and transforming the assembling process into SLM, a dynamic programming algorithm could be performed to choose suitable parts to compose the final genetic construction. The result showed this method had high accuracy, redundant operations could be reduced and the time and cost required for conducting biological experiment could be minimized. The method could be not only used to optimize a design in a synthetic biological robotic platform, but also independently used to automate the DNA assembly process in synthetic biology. It could also be iterated and then give out different optimized results for consideration.

0 引言

随着合成生物学数据量的增加,十分需要开发一种用于蛋白质表达载体设计的工具软件,这类工具软件不仅可以设计蛋白质表达载体,还可以用于基因与代谢网络的研究[1-2]. 设计时,将每一个合成生物学标准"零件"(BioBrick)看作一个"词",用来设计所需的合成生物学构件[3-4]. 在表达载体设计中,把启动子、核糖体结合位点、基因及终止子都看作属于各自特性的"词类",然后依据特殊编制的语法来设计蛋白质表达载体[5-6]. 设计者们往往将已有的生物序列拆成"零件",并将其作为 BioBrick[7],而当设计者将这些属于不同词类的标准"零件"进行组装时,其过程通常耗时、费力而且容易发生错误. 实际的装配中,通常使用同一组限制性内切酶和连接酶,方可将同一组"零件"连接起来[8-9],在设计软件时,也会模拟使用这些酶进行组装. 这些 BioBrick 的模拟组装完全可以在计算机中自动实现[10],但是这样做只考虑了最基本的语法规则,而忽略了以往组装成功的案例. 因此,在进行蛋白质表达载体设计时,使用者可以根据自己的需要编制语法,再根据某些具有生物学意义的设计规则变换设计结构,最后选择"零件"完成设计[11]. 但是,当越来越多的"零件"被输入数据库后,在设计的最后一步,设计者往往不知道从词类中选择哪个"零件"更为合适. 为了解决这个问题,统计语言模型 SLM(Statistical Language Model)被引入设计中. SLM 最初用于自然语言识别[12],用来估算一组词串成为一个正确语句的概率. 它目前的主要应用是语音识别,除此之外还应用于机器翻译、智能输入及文本语音转换. 本文拟通过统计 BioBrick 的一些参数,将 BioBrick 的组装过程转化为 SLM,然后使用动态规划算法找出最优路径,即找出合适的"零件",组装成蛋白质表达载体,从而完成设计.

1 模型与算法

使用链接 http://parts.igem.org/das/parts/entry_points/ 下载 BioBrick 合成生物学组件信息,2014 年 1 月的版本包含 7 242 个合成生物学组件. 编写一个 Perl 脚本,通过链接 http://parts.igem.org/das/parts/features/?segment=part# 分析并提取每个组件的信息. 将每个组件的信息分解排列成特有的结构,输入 MySQL 数据库. 输入数据库后共分解提取出 75 744个"零件",这些"零件"包括基础"零件"(启动子、核糖体结合位点、基因编码序列、终止子及质粒序列)和复合"零件",复合"零件"由基础"零件"组装而成. 通过查询 MySQL 数据库提取出这些基础"零件"并统计它们的使用频率,同时编写 Perl 脚本和一些 SQL 语句分析复合"零件",统计出相连续的 2 个、3 个和

4 个基础“零件”的使用频率. 通过查询数据库，共提取出 1 628 个符合 RFC 23 组装标准的基础“零件”[13]. 这意味着，在这些基础“零件”序列中(除了两侧的连接序列)不包含该组装标准使用的限制性酶切位点. 1 682 个基础“零件”共包含 405 个启动子、42 个核糖体结合位点、57 个终止子、1 178 个基因序列，将被用来设计蛋白质表达载体. 同时每一个基础“零件”的使用频率、每一对基础“零件”的使用频率，以及连续 3 个、4 个为一组的基础“零件”的使用频率都可以被计算出来.

1.1　语法模型

编制设计合成生物学组件的语法在文献中已有详细描述[14]，根据不同的要求可以编制不同的语法. 但是基本语法是一样的，即启动子—核糖体结合位点—基因—终止子(PRO (Promoter)—RBS (Ribosome-Binding Sites)—GEN (Genes)—TER(Terminator))，本文使用的语法与文献[14]使用的跟上下文无关的语法类似，但是增加了一些新的规则以表达融合蛋白，比如增加规则使一个表达盒变成一个启动子加一个阅读框，以便在一个表达盒里实现两个蛋白融合表达.

1.2　数学模型

在一些设计软件中，比如 GenoCAD，在设计的最后一步，设计者需要从大量的基础“零件”中选择合适的“零件”完成设计(见图 1)，这一过程比较困难. 为解决这一问题，本文引入了在语音识别、机器翻译、智能输入法中广泛应用的 SLM.

在这个模型中，一个句子 S(Sentence)是否有意义并且合理的判断依据是其出现的概率. 一个句子由一系列的词组成，本文中一个句子就是一个由基础“零件”构成的生物学组件，这些基础“零件”就是组成句子的词，一个基础“零件”*part* 就是一个词. 因此，$S = part_1, part_2, \cdots, part_n$，其发生概率

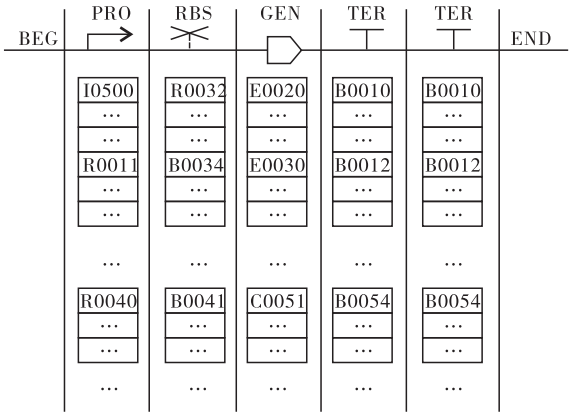

图 1　GenoCAD 设计最后一步的多个选项

Fig. 1　The last step of design in GenoCAD and options in every category

$$P(S) = P(part_1, part_2, \cdots, part_n) \qquad ①$$

根据公式①有

$$P(part_1, part_2, \cdots, part_n) = P(part_1) \cdot P(part_2 | part_1) \cdot P(part_3 | part_1, part_2) \cdots P(part_n | part_1, part_2, \cdots, part_{n-1}) \qquad ②$$

式中，$P(part_1)$ 指一个基础“零件”在一个设计中出现的概率，$P(part_2 | part_1)$ 指 $part_1$ 出现在 $part_2$ 之前 $part_2$ 出现的概率. 由此可知，$part_n$ 出现的概率由所有在它之前出现的基础“零件”确定. 其中 $P(part_1)$ 和 $P(part_2 | part_1)$ 容易计算，$P(part_3 | part_1, part_2)$ 计算比较复杂，而 $P(part_n | part_1, part_2, \cdots, part_{n-1})$ 计算将非常困难，因为牵扯的变量太多，导致条件过于复杂. 基于马尔科夫假设可以认为，一个“零件”在一个设计中出现的概率仅仅与相邻的前一个“零件”相关，即统计语言的二元模型(Bigram Model). 但是在蛋白质表达载体设计时，一个基因能否被有效表达不仅与其 RBS 相关，还与其 PRO 密切相关. 因此在实际设计中，一个“零件”在出现的概率与相邻的前两个“零件”相关. 由此，式②可以简化为

$$P(S)=P(part_1)\cdot P(part_2|part_1)\cdot P(part_3|part_1,part_2)\cdots P(part_i|part_i,part_{i-1})\cdots P(part_n|part_{n-2},part_{n-1}) \quad ③$$

则 $P(S)$,即一个句子发生的概率就可以计算出来了. 式③就是 SLM 中的三元模型,同理可以得出四元模型. 因此条件概率公式可以表示为

$$P(part_i|part_{i-2},part_{i-1})=\frac{P(part_{i-2},part_{i-1},part_i)}{P(part_{i-2},part_{i-1})} \quad ④$$

笔者将使用两个相邻基础“零件”出现的频率及连续 3 个基础“零件”出现的频率来估算④式中的 $P(part_{i-2},part_{i-1},part_i)$ 和 $P(part_{i-2},part_{i-1})$,即

$$P(part_{i-2},part_{i-1},part_i)\approx\frac{Count(part_{i-2},part_{i-1},part_i)}{Count(all\_parts)}$$

$$P(part_{i-2},part_{i-1})\approx\frac{Count(part_{i-2},part_{i-1})}{Count(all\_parts)}$$

可得

$$P(part_i|part_{i-2}-part_{i-1})\approx\frac{Count(part_{i-2},part_{i-1},part_i)}{Count(part_{i-2},part_{i-1})} \quad ⑤$$

根据式⑤,式③中任何一个部分都可以计算出来.

设计的最后一步,需要从很多个基础“零件”中选择最合理且最有意义的一种组合. 根据 SLM 理论,概率最大即为最佳选择. 在图 1 所示网格结构中,可以有很多个候选路径产生句子,一条路径产生一个句子,一个句子就是一个设计 $(path = S = part_1,part_2,\cdots,part_n)$. 最优路径

$$PATH=\arg\max_{all\_S}(P(S))$$

为避免计算时内存溢出,可以对 $P(S)$ 取对数值,即

$$PATH=\arg\max_{all\_S}(\log P(S))=\arg\max_{all\_S}(\log(P(part_1)\times P(part_2\mid part_1)\times\prod_{i=3}^{n}P(part_i\mid part_{i-2},part_{i-1})))=\arg\max_{all\_S}(\log(P(part_1)+\log P(part_2\mid part_1)+\sum_{i=3}^{n}\log P(part_i\mid part_{i-2},part_{i-1})) \quad ⑥$$

根据式⑤可以得到

$$P(part_i\mid part_{i-2},part_{i-1})=\frac{Count(part_{i-2},part_{i-1},part_i)}{Count(part_{i-2},part_{i-1})} \quad ⑦$$

$$P(part_2\mid part_1)=\frac{Count(part_1,part_2)}{Count(part_1)} \quad ⑧$$

$$P(part_1)=\frac{Count(part_1)}{Count(all\_parts)} \quad ⑨$$

因为我们从一个相对稀疏的语料库中提取参数,所以零概率问题不可避免. 为解决这一问题,我们采用线性插值法做数据平滑[15]. 式⑦⑧⑨用来计算式⑥中各个成分的值,从而得出最优路径,四元模型依据同样的原理计算. 最优路径 *PATH* 是所有路径中具有最大出现概率的那一个,本文使用动态规划算法进行查找.

1.3 算法

寻找最优路径的过程就是求解式⑥的过程,三元模型的算法源于维特比算法[16],二元、四元模型的算法与此相同,均由 3 个步骤组成.

步骤 1 建立候选网格.

每一类基础“零件”对应一列,而每列中的每一个节点对应一个基础“零件”. 在网格的开始位置和结束位置添加 BEG 和 END 列,在这两列中添加虚拟节点 *B* 和 *E*. 网格的建立、填充与回溯过程如图 2 所示. 图 2 中每一个节点是一个三元组 $<name,V,P>$,三元组的第一元素 *name* 是基础“零件”的序列号,这一序列号在数据库中是唯一的.

步骤 2 填充网格.

从左至右填充网格,对于每一个三元组,*V* 和 *P* 需要被计算出来并填充相应的值. *V* 由连续三列所有节点进行组合运算的最大值进行填

充,P 将存储与当前节点组合运算产生最大值的前一个节点的地址信息.

1)对第一列,节点 B 使 $V = 0$ 且 $P = NULL$.

2)对第二列,每一节点三元组 $< name, V, P >$ ($name \in \{\text{I0500}, \text{R0011}, \cdots, \text{R0040}, \cdots\}$)将与 B 节点组合计算其 V 值和 P.

$$V = V_B + \log P(part) = \log P(part)$$

$$P = address\_of\_B$$

3)对第三列,每一节点三元组 $< name, V, P >$ ($name \in \{\text{R0032}, \text{B0034}, \cdots, \text{B0041}, \cdots\}$)将与第二列中的所有节点组合并计算其 V 值和 P.

$$V = \max\{\log P(part \mid part_{\text{prior}})\}$$

$$P = address\_where\_V\_comes\_from$$

4)对第四列,每一节点三元组 $< name, V, P >$ 将与前两列中的所有节点组合并计算其 V 值和 P.

$$V_{\text{Current}} = \max\{V_{\text{Before}} + V_{\text{Previous}} + \log P(part \mid part_{\text{Before}}, part_{\text{Previous}})\}$$

$$P_{\text{Current}} = Address\_of\_V_{\text{Previous}}\_belong$$

$$P_{\text{Previous}} = Address\_of\_V_{\text{Before}}\_belong$$

5)重复 4),前列的每一个节点都与前两列的每一个节点组合计算其 V 值和 P.

6)在 END 列,E 节点的 V 将由前一列的最大值填充,P 将存储前一列最大值对应节点的地址信息.

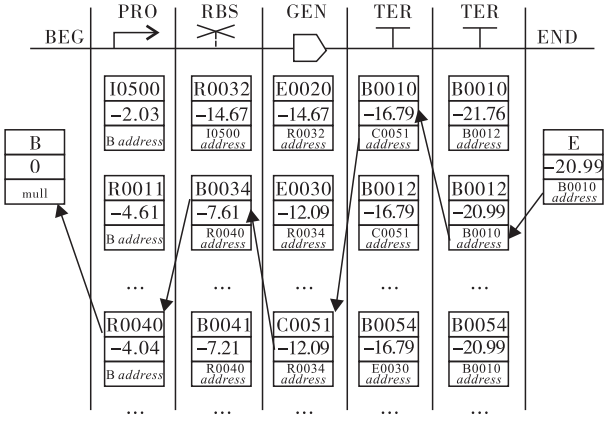

图 2　网格的建立、填充与回溯过程示意图

Fig. 2　The diagram of building lattice, filling lattice and recalling process

步骤 3　回溯找出最优路径 *PATH*.

从节点 E 开始不断找出前面节点的 P,具有最大概率的 P 即为最优路径 *PATH*,其产生的句子 S 就是设计完成的具有生物学意义的分子生物学组件. 如果 S 的长度是 L,而一列中节点的个数最多是 D,则三元模型算法复杂度是 $O(L \cdot D^3)$,二元模型算法复杂度是 $O(L \cdot D^2)$,四元模型算法复杂度是 $O(L \cdot D^4)$,穷举算法复杂度是 $O(D^L)$. 用二元模型进行计算速度最快,但理论上准确率要差一些,三元模型、四元模型准确率较高,但计算速度较慢,穷举算法速度最慢,不适用于解决规模较大的问题.

2　结果与讨论

为验证本方法的准确性,以一个可以产生香蕉气味的质粒(http://parts.igem.org/Part:BBa_J45900)为研究对象,展示如何将 BioBrick 组装成功能性合成生物学组件. 该质粒由麻省理工学院参加 2006 年 iGEM(International Genetically Engineered Machine Competition)竞赛的参赛队设计并实施. 该组件包含两个表达盒:一个表达盒包含 BAT_2 和 THI_3 基因,另一个表达盒的产物通过催化前面基因的产物而使大肠杆菌发出香蕉的气味. 确定要表达的基因之后,装配算法需由一个 Perl 脚本执行,因此应首先编制相应的语法,然后根据语法设计合成生物学组件,用于设计表达融合蛋白的语法见表 1.

依据规则 1,一个表达盒 Cass 可以变成两个表达盒 Cass-Cass. 对第一、第二个表达盒分别使用规则 4 和规则 3,该设计变成 PRO-Cis-PRO-Cis-TER. 对第一个表达框 Cis 使用规则 7,对第二个表达框 Cis 使用规则 5,该设计变为 PRO-Cis-Cis-PRO-RBS-GEN-TER. 最后一步对两个 Cis 使用规则 5,对 TER 使用规则 6,此时

表 1 用于设计表达融合蛋白的语法

Table 1 The grammar for fusion protein expression

规则	Comments	Left term	Right term
1	Transform a cassette (Cass) into two cassettes (Cass)	Cass	Cass-Cass
2	Reverse the sequence orientation of a cassette (Cass)	Cass	[Cass]
3	Transform a cassette (Cass) into a promoter (PRO), a cistron (Cis), and a terminator (TER)	Cass	PRO-Cis-TER
4	Transform a cassette (Cass) into promoter and cistrons (PRO-Cis)	Cass	PRO-Cis
5	Transform a cistron (Cis) into a rbs (RBS) and a gene (GEN)	Cis	RBS-GEN
6	Transform a terminator (TER) into two terminators (TER)	TER	TER-TER
7	Transform a cistron (Cis) into two cistron (Cis)	Cis	Cis-Cis

的设计就是 PRO-RBS-GEN-RBS-GEN-PRO-RBS-GEN-TER-TER. 在输入该设计之前,先确定需要表达的基因,因此输入的设计就是

PRO. RBS. GEN(J45008). RBS. GEN(J45009). PRO. RBS. GEN(J45014). TER. TER

装配算法由一个 Perl 脚本执行,则

二元模型给出序列为

R0040-B0034-J45008-B0030-J45009-R0040-B0034-J45014-B0010-B0012

三元模型给出序列为

R0040-B0030-J45008-B0030-J45009-R0040-B0030-J45014-B0010-B0012

四元模型给出序列为

R0011-B0030-J45008-B0030-J45009-R0040-B0030-J45014-B0010-B0012

四元模型给出的序列正是产生香蕉气味这一合成生物学组件的真实组成. 由此可见,四元模型能够准确地给出优化结果,三元模型与真实组成只有一个"零件"不同. 如果进行其他的设计并执行算法,该方法将给出一个优化的结果,这一结果采用了以往设计的经验. 而且我们采用模型考虑的词串越长,优化结果越好,这一点与自然语言处理的理论和经验也是相符的.

本文研究的优化方法采用三元、四元 SLM,这意味着一个"零件"只与它前面两三个相邻的"零件"有联系. 然而在真实的分子生物学环境中,一个基因能否高效表达不仅与其核糖体结合位点、启动子有关,还与其他调控序列有关. 若考虑 N 元模型,则意味着每个"零件"与它前面 $N-1$ 个"零件"有关系,但是这种情况其条件概率是难以计算的. 当 $N=5$ 或更多时,尽管在其他 SLM 应用范例中(如机器翻译、分词、智能输入法)准确率会大大提高,但是计算量也大大增加,需要功能强大的计算机才能实现[12]. 下一步,我们将开发一个五元模型并且将其他质粒序列考虑进来,从而更真实地模拟合成生物学基因片段的组装过程. 计算条件概率时,本文采用了线性插值法做数据平滑,以解决零概率问题. 但是由于目前 SLM 模型在合成生物学中鲜有应用,尚无资料显示哪一种数据平滑技术更有效,在下一步工作中,我们将扩大语料库,并综合比较拉普拉斯法、Good-Turing 估计、卡茨退避法等[17-18]数据平滑技术,以进一步提高准确率.

3 结论

本文将 SLM 引入合成生物学基因片段组装,通过统计 BioBrick 的参数将基础"零件"的装配过程转化为 SLM,然后执行动态规划算法找出最优结果,实现蛋白质表达载体的设计. 该算法依据以往经验自动选择合适的"零件"装配成合成生物学组件,可以减少真实装配过程的冗余操作,节省时间和费用. 这一方法可用来优化其他合成生物学软件设计结果,也可独立使用来模拟装配合成生物学基因片段产生蛋白质表达载体,还可被迭代从而给出不同的优化

结果供选择.

本文从 iGEM 网站下载的语料库相对稀疏,下一步考虑将语料库扩展到常用的、商业化的蛋白质表达载体上统计相应的参数,这样 SLM 可以更广泛地应用于合成生物学并得到检验. 但是对这些合成生物学片段进行描述的术语还没有完全统一,因此发展合成生物学开放语言 SBOL(Synthetic Biology Open Language)并开发与之相应的数据挖掘技术十分必要.

参考文献:

[1] GOLER J A, BRAMLETT B W, PECCOUD J. Genetic design: rising above the sequence[J]. Trends Biotechnol, 2008, 26: 538.

[2] GRASLUND S, NORDLUND P, WEIGELT J, et al. Protein production and purification[J]. Nat Methods, 2008(5): 135.

[3] CZAR M J, CAI Y, PECCOUD J. Writing DNA with GenoCAD[J]. Nucleic Acids Res, 2009, 37: W40.

[4] CAI Y, WILSON M L, PECCOUD J. GenoCAD for iGEM: a grammatical approach to the design of standard-compliant constructs[J]. Nucleic Acids Res, 2010, 38: 2637.

[5] ISAACS F J, DWYER D J, DING C, et al. Engineered riboregulators enable posttranscriptional control of gene expression[J]. Nat Biotechnol, 2004, 22: 841.

[6] GARDNER T S, CANTOR C R, COLLINS J J. Construction of a genetic toggle switch in Escherichia coli[J]. Nature, 2000, 403: 339.

[7] ADAMES N R, WILSON M L, FANG G, et al. GenoLIB: a database of biological parts derived from a library of common plasmid features[J]. Nucleic Acids Res, 2015, 43: 4823.

[8] ARKIN A. Setting the standard in synthetic biology[J]. Nat Biotechnol, 2008, 26: 771.

[9] CANTON B, LABNO A, ENDY D. Refinement and standardization of synthetic biological parts and devices[J]. Nat Biotechnol, 2008, 26: 787.

[10] DENSMORE D, HSIAU T H C, BATTEN C, et al. Algorithms for automated DNA assembly[J]. Nucleic Acids Res, 2010, 38: 2607.

[11] COLL A, WILSON M L, GRUDEN K, et al. Rule-based design of plant expression vectors using GenoCAD[J]. PLoS ONE, 2015, 10(7): e0132502.

[12] JELINEK F. Statistical Methods for Speech Recognition (Language, Speech, and Communication)[M]. Cambridge: MIT Press, 1998.

[13] PHILLIPS I E, SLIVER P A. A new biobrick assembly strategy designed for facile protein engineering[EB/OL]. (2016 - 02 - 15)[2016 - 11 - 15]. http://dspace.mit.edu/handle/1721.1/32535, 2006.

[14] CAI Y, HARTNETT B, GUSTAFSSON C, et al. A syntactic model to design and verify synthetic genetic constructs derived from standard biological parts[J]. Bioinformatics, 2007, 23: 2760.

[15] CHEN S F, GOODMAN G. An empirical study of smoothing techniques for language modeling[J]. Computer Speech and Language, 1999(13): 359.

[16] VITERBI A J. A personal history of the viterbi algorithm[J]. IEEE Signal Processing Magazine, 2006, 23: 120.

[17] HUANG F L, YU M S, HWANG C Y. An empirical study of good-turing smoothing for language models on different size corpora of Chinese[J]. Journal of Computer and Communications, 2013(1): 14.

[18] KATZ S M. Estimation of probabilities from sparse data for the language model component of a speech recogniser[J]. IEEE Transactions on Acoustics (Speech and Signal Processing), 1987, 35: 400.