第 40 卷 第 12 期 计 算 机 工 程 2014 年 12 月
Vol. 40 No. 12 Computer Engineering December 2014

·人工智能及识别技术· 文章编号：1000-3428(2014)12-0126-06 文献标识码：A 中图分类号：TP391.1

基于自然语言处理的算法知识名称发现

朱国进，郑 宁

（东华大学计算机科学与技术学院，上海 201620）

摘 要： 网络中的很多程序资源在知识概念上有内在的联系，却没有超链接将它们连接在一起。将网络程序资源中的算法知识名称获取出来，组织成一个算法知识专家库文件，用于识别程序设计资源所含的知识点，即可将程序设计资源按知识点相互联系。为了自动获取程序资源中的算法知识名称，提出一种基于自然语言处理的算法知识名称发现方法。通过发现含有算法知识名称语句的字符串模式，从程序资源中提取可能含算法知识名称的字符串，从中找出最有可能出现在算法知识名称中的分词，并根据这些分词获取算法知识名称。实验结果表明，与原有人工整理出的算法知识名称集合相比，该方法新增了 11.2% 的算法知识点和 13.6% 的算法知识名称。

关键词： 知识发现；模式发现；自然语言处理；算法知识名称；中文分词；词性标注

中文引用格式： 朱国进，郑 宁. 基于自然语言处理的算法知识名称发现[J]. 计算机工程，2014，40(12)：126-131.

英文引用格式： Zhu Guojin，Zheng Ning. Discovering Terms of Algorithmic Knowledge Based on Natural Language Processing[J]. Computer Engineering，2014，40(12)：126-131.

Discovering Terms of Algorithmic Knowledge Based on Natural Language Processing

ZHU Guojin，ZHENG Ning

（School of Computer Science and Technology，Donghua University，Shanghai 201620，China）

【Abstract】 There are many programming resources on the Internet. Although these programming resources have internal relations，there are often no hyperlinks connecting them. Getting the terms of algorithmic knowledge，organizing the terms to an expert file，which is used for recognizing the knowledge in the programming resources，the programming resources can be connected by the knowledge. To get the terms of algorithmic knowledge，this paper proposes a method to discover terms of algorithmic knowledge based on natural language processing. This method consists of discovering the patterns of strings which contain terms of algorithmic knowledge，extracting from programming resources that probably contain terms of algorithmic knowledge according to the discovered patterns，finding the word segmentation most likely appearing in the terms of algorithmic knowledge，and fetching the terms of algorithmic knowledge according to the word segmentation. This method increases 11.2% algorithmic knowledge and 13.6% terms of algorithmic knowledge in comparison with the manual collection of terms of algorithmic knowledge which is obtained by previous work.

【Key words】 knowledge discovery；pattern discovery；natural language processing；terms of algorithmic knowledge；Chinese word segmentation；part-of-speech tagging

DOI： 10.3969/j.issn.1000-3428.2014.12.023

1 概述

在 Web 网中，许多知识资源虽然彼此之间在知识概念上存在着内在联系，但常常却不见有超链接将它们直接地连接在一起。在网络中存在的程序设计资源也是这样。例如，在国际大学生程序设计竞赛（简称 ACM 竞赛）中，很多爱好者会将自己在 ACM 竞赛过程中所取得的解题经验写成日志分享在网络上，这些日志是一些具体题目的解法，将它们称为解题报告。在之前的研究工作中，一些有经验的 ACM 竞赛队员建立了一个涉及 ACM 竞赛的算法知识的专家库[1]，并利用专家库识别 ACM 竞赛解题报告中所含的算法知识点[2]，然后将这些解题报告按照所含的算法知识点连接起来[1]。即使经过很

基金项目： 国家自然科学基金资助项目（60973121）。

作者简介： 朱国进（1958－），男，副教授，主研方向：知识发现，语义网络，自然语言处理；郑 宁，硕士研究生。

收稿日期： 2013-12-20 **修回日期：** 2014-02-27 **E-mail：** gjzhu.dhu@163.com

多有经验的人总结,所获得的算法知识名称也不一定全面。如果专家库中没有某些算法知识点的名称,系统就无法识别解题报告中的这些知识点,因而也就无法建立相应解题报告之间的知识联系。要从网络中发现算法知识名称来扩充专家库,使之包含的算法知识名称更加全面。

随着网络信息的快速增加,仅靠人力已经不足以从大量的网络文本数据中发现知识,因此文本知识发现的技术也快速发展起来。模式发现作为数据发现的一种情况也有很多研究。文献[3]介绍了频繁模式挖掘。文献[4]介绍了建立视觉单词的语义来发现模式。但这些研究与本文研究的目的不同,它们的方法不能用于本文研究中。

算法知识名称发现方法是基于算法知识点名称的自然语言特征,这些特征包括算法知识点名称的词性特征和词长特征以及它们在自然语句中可能出现的概率特征。该方法通过发现含有算法知识名称语句的字符串模式,提取出可能含算法知识名称的字符串。然后将这些字符串进行分词得到分词串。接着对每个分词串求出它与其他各个分词串的公共分词串,从中选出最长的公共分词串作为该分词串的关键子串,简称关键串。从关键串中,根据算法知识点名称的词性特征以及它们出现在自然语句中的概率特征,以及利用不会在算法知识名称出现的分词组成的停用词表,寻找出可能出现在算法知识名称中的特征分词。含有同一个特征分词的所有关键串,构成一个关键串的集合。每一个特征分词都可以确定这样的一个关键串集合。最后根据算法知识点名称的词长特征,利用 N-gram[5] 算法从每个关键串集合中获取算法知识名称。

近年来,自然语言处理[6-7] 的研究也非常流行。虽然基于单词的解析还没有突破,但是构建在大型语料库之上的应用取得了很多的成果。本文使用的自然语言处理工具是 Stanford parser[8],它可以用于中文分词[9] 和标注词性。Stanford parser 是基于统计的分析程序,其解析的准确率较高[8]。

2　基本概念与术语

2.1　程序设计网络资源

解题报告:网络中对某个 ACM 竞赛题目进行分析和解答的网页。解题报告网页中包含超链接、评论、解题报告的正文内容和标题等信息。

正文标签:解题报告中正文内容被包含在一对 html 标签中,该标签称为正文标签。同一个网站上的解题报告其正文标签是相同的。

正文内容:解题报告的正文内容中通常含有分析内容和解题的代码。也有少部分解题报告正文内容中没有分析内容,只有解题的代码。

句子:对解题报告中分析内容进行断句后得到的短句。在本文研究中,断句的标志是句号、问号、感叹号、逗号,以及顿号。这是为了缩短句子的长度,减少干扰字符的数量。

2.2　含算法知识名称的字符串

算法知识名称:ACM 竞赛中涉及的知识名称和算法名称,如博弈论、二分法、最短路径等。

算法知识点:在本文研究中,算法知识点也是一个名称,与算法知识名称不同的是,算法知识点由它最常用的名称来表示,而且一个算法知识点可能有一个或多个算法知识名称。例如:算法知识点“冒泡排序”,它的名称有“冒泡排序”、“bubble sort”、“bubble-sort”。

字符串模式:含有算法知识名称句子的字符串特征,将其归结为正则表达式。例如:在许多含有算法知识名称的句子中都出现了“算法”这 2 个字,所以正则表达式“(s)*算法(s)*”就是这些句子的字符串特征,这里 s 指除了作为断句的标点符号之外的所有字符,* 指字符可以重复 0 次或多次。

句子集合 SENS:解题报告中所有包含系统中原始的算法知识名称的句子的集合。系统中原始的算法知识名称可以由人工整理得到。句子集合 SENS 将用于发现含有算法知识名称的字符串模式。

模式字符串集(*PS*):解题报告正文中与某个字符串模式匹配的句子称为模式字符串。例如:“然后想到匈牙利二分匹配算法”,这个句子与“(s)*算法(s)*”这个模式匹配,所以是一个模式字符串。所有解题报告中的所有模式字符串组成一个模式字符串集 *PS*。由于字符串模式反映了含有算法知识名称句子的字符串特征,因此模式字符串集 *PS* 中的每一个字符串都被认为可能含有算法知识名称。

2.3　含算法知识名称的分词串

分词串集(*SP*):模式字符串集 *PS* 中的每一个字符串分词后就形成了分词串集 *SP*。例如:模式字符串“然后想到匈牙利二分匹配算法”分词后得到分词串“然后/想到/匈牙利/二分/匹配/算法”,其中,“/”表示分词之间的分隔符。在分词串集 *SP* 中,单个分词串记为 *sp*,除 *sp* 之外的其他分词串所构成的集合记为 *sp* ~。

公共分词串:2 个分词串共有的最长子分词串称为它们的公共分词串,例如 2 个分词串“用/了/线段/树”和“用/线段/树/实现”的公共分词串是“线段/树”。如果最长子分词串存在多个,为了计算方便,可以从它们中随机地选择一个作为这 2 个分词串的公共分词串。

关键串集(*KSP*):分词串集 *SP* 中的分词串 *sp*

与它的分词串集 $sp\sim$ 中每个分词串各自都有一个公共分词串,在这些公共分词串中取长度最长的一个作为该分词串 sp 的关键串。如果最长的公共分词串存在多个,为了计算方便,可以从它们中随机地选择一个作为该分词串 sp 的关键串,记为 ksp。所有关键串组成的集合称为关键串集,记为 KSP,关键串集中除 ksp 之外的其他关键串的集合记为 $ksp\sim$。

2.4 算法知识名称的自然语言特征

特征分词:很有可能在算法知识名称中出现的分词。

词性特征:词性特征值可以计算出哪种词性在算法知识名称中比较重要。可以利用词性特征值来为算法知识名称中的分词增加权值。分词 t 的词性特征值是该分词词性在算法知识名称中出现的频率与该词性在所有语句中出现的反频率的乘积,记为 $u(t)$,由式(1)计算得到:

$$u(t)=\frac{p(t)}{P_1}\ln\frac{P_2}{q(t)} \tag{1}$$

其中,$p(t)$是分词 t 的词性在所有算法知识名称中出现的次数;$q(t)$是分词 t 的词性在所有语句中出现的次数;P_1 是所有算法知识名称中分词的总数;P_2 是所有语句中分词的总数。

分词特征值:分词特征值用于选择特征分词。分词 t 的特征值 $v(t)$由式(2)计算:

$$v(t)=cf(t)/sf(t)\times u(t) \tag{2}$$

其中,t 是分词;$cf(t)$是分词 t 在关键串集 KSP 中出现的次数;$sf(t)$是分词 t 在分词串集 SP 中出现的次数;$u(t)$由式(1)计算得到。

3 算法知识名称发现方法

本文方法由 5 个步骤组成,图 1 中的数字标示了它们执行的先后顺序和每一步涉及到的主要内容。

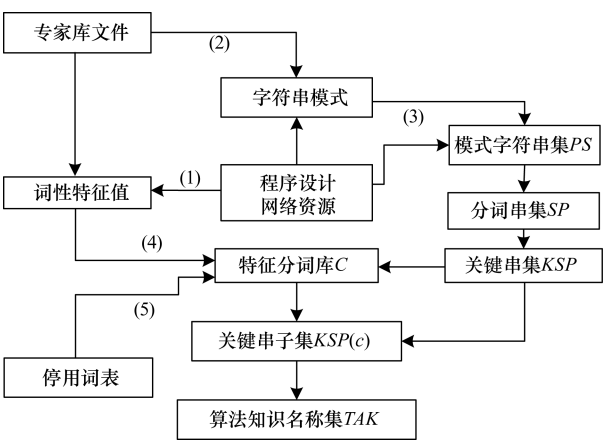

图 1 本文方法的逻辑结构

3.1 解题报告正文的提取

程序设计网络资源的解题报告网页中包含超链接、评论、解题报告的正文内容和标题等信息。要先提取解题报告的正文内容和标题,并将正文内容断句。仅处理正文和标题的内容可以降低提取的难度。

3.2 字符串模式的发现

假设 1 绝大多数含算法知识名称的语句均具有相同特征的字符串模式。

解题报告的正文内容经过断句后有很多的句子,这些句子不是每个都含算法知识名称。但是有些句子会含有系统中原始的含算法知识名称。根据假设 1,可以通过这些句子找到包含算法知识名称的句子的特点。然后根据这些特点获取正文中所有可能包含算法知识名称的句子。

首先从所有句子中获取包含系统中原始算法知识名称的句子,得到句子集合 SENS,并对这些句子进行分词。以这些句子中比较重要的分词作为字符串模式中的分词。根据 TF/ISF(Term Frequency/Inverse Statement Frequency)选择重要的分词,其中,TF 指分词在集合 SENS 的句子中出现的频率;ISF 指分词的反句子频率。TF/ISF 是对 TF/IDF[11] 方法的应用型改变。分词 t 的 TF/ISF 的计算如下:

$$\frac{IT}{ISF}=\frac{f(t)}{T_1}\cdot\ln\frac{T_2}{s(t)} \tag{3}$$

其中,$f(t)$是分词 t 在集合 SENS 的句子中出现的频率;$s(t)$是分词 t 在所有解题报告的所有句子中出现的频率;T_1 是集合 SENS 中句子的总数;T_2 是所有解题报告中所有句子的总数。

选取 TF/ISF 值最大的 N 个分词,使集合 SENS 中的每个句子都包含 N 个分词中的某一个。模式的形式是(s)*WS(s)*,其中,s 是除了断句的标点之外的所有字符;* 指该字符可以重复零次或多次;WS 是 Word Segmentation(分词)的缩写,表示某个分词。选取字符串模式中重要分词的算法描述如下:

输入 解题报告中的所有句子

输出 字符串模式中的分词

(1) 获取含专家库中算法知识名称的句子集合;

(2) 计算句子集合中每个分词的 TF/ISF 值;

(3) 选取 TF/ISF 值较大的若干个分词,使得句子集合中每个句子都包含这些分词中的一个。

3.3 关键串集的获取

假设 2 公共分词串的长度越大,其包含算法知识名称的可能性越大。

根据包含专家库中算法知识名称的句子提取字符串模式,并根据模式提取可能含算法知识名称的字符串,即模式字符串集 PS。要提取的对象是代表算法知识的术语,从可能含这些术语的模式字符串

中提取,其准确率要比从全文提取的更精确。

按照字符串模式提取模式字符串集 *PS*,下载的解题报告有些是完全相同的,因此,要删除模式字符串集中的重复项。利用分词器对模式字符串集 *PS* 中的每个字符串分词,就得到了分词串集 *SP*。

分词串是从模式字符串拆分得到的。拆分后,得到的多个分词串中可能只有某一部分分词串包含算法知识名称。将分词串去除重复项之后,对每个分词串 *sp* 与分词串集 *sp* ~ 中每个分词串作比较,按照假设 2,在获取的多个公共分词串中取最长的一个,如果有几个公共分词串长度相同且最长时,任意选择一个公共分词串,这个公共分词串作为分词串 *sp* 的关键串。

取关键串可以大量减少与算法知识名称无关的分词数量,但是在分词串中出现 2 次及以上的算法知识名称数量基本不会减少。

获取关键串集的算法描述如下:

输入　模式字符串集 *PS*

输出　关键串集 *KSP*

(1) 对每个模式字符串 *ps* 进行分词获取分词串 *sp*;

(2) 对每个分词串 *sp* 与 *sp* ~ 进行比较获取多个公共分词串,选取其中最长的公共分词串作为 *ps* 的关键串。如果最长的公共分词串有多个,则从多个最长的公共分词串中随机选取一个作为关键串。

3.4　特征分词的获取

将分词串 *sp* 中的每个分词标注词性,然后将其对应的关键串中的每个分词标注上相同的词性。关键串是分词串的一部分,但是对两者分别标注词性时相同分词的词性标注也不相同,因此,要标注分词串中分词的词性,然后将关键串中相同的分词标注上相同的词性。

专家库中算法知识名称作为原始算法知识名称,以词性在原始算法知识名称中出现的频率和在所有句子中出现的反频率计算词性特征,公式见式(1)。

计算关键串中每个分词的特征值。利用分词在分词串集与关键串集中出现频率的比值及分词的词性特征作为该分词的特征值,公式见式(2)。获取分词串中的关键串,减少了分词串中分词的数量,而算法知识名称中的分词基本不会减少。因此,不是算法知识名称中的分词,其在分词串集中出现的频率与在关键串集出现的频率比值会减小。

从每个关键串 *ksp* 中提取特征分词的条件如下:

(1) 在组成关键串的一个或多个分词中特征值 *V* 最大。如果一个关键串中 2 个分词特征值 *V* 相同,则都作为待选分词。

(2) 分词的特征值大于 0.1。

从关键串中提取的特征分词,还需要利用停用词表来过滤一些与算法知识名称无关的分词。停用词表一部分是从网络下载停用词表,另一部分是经过人工观察和算法知识名称无关的分词,例如:“完完全全”,“喜欢”,“有感”等。

从关键串集 *KSP* 中获取特征分词库 *C* 中的算法知识名称如下:

输入　关键串集 *KSP*

输出　特征词库 *C*

(1) 对于关键串集 *KSP* 中的每个关键串 *ksp*;

(2) 利用式(2)计算关键串 *ksp* 每个分词 *t* 的特征值 $v(t)$;

(3) 获取关键串 *ksp* 中特征值最大且值大于0.1的分词作为特征分词,存入特征词库 *C*;

(4) 利用停用词表过滤无关分词。

3.5　算法知识名称的获取

将关键串集 *KSP* 中每个含特征分词 *c* 的关键串分成一个子集,记为关键串子集 $KSP(c)$。关键串子集 $KSP(c)$ 中的单个关键串记为 $ksp(c)$。如果特征分词库中有 *n* 个分词,则关键串集 *KSP* 会被分成 *n* 个子集。将关键串集按特征分词分成多个子集,不会影响结果的准确性,但是可以大大减少运行的时间。

在关键串子集 $KSP(c)$ 中,利用每个关键串 $ksp(c)$ 中分词建立一元和多元的组合词。将包含特征分词 *c* 的组合词,以及特征分词 *c* 作为候选词。

统计每个候选词在关键串中出现的次数,选出出现次数最大和第二大的候选词作为算法知识名称。

从关键串集 *KSP* 中获取算法知识名称的算法描述如下:

输入　关键串集 *KSP*,特征分词库 *C*

输出　算法知识名称

(1) 对于每个特征分词 *c*,获取关键串子集 $KSP(c)$;

(2) 对于关键串子集中每个 $ksp(c)$,用组成它的分词构建组合词;将特征分词 *c*,以及包含特征分词 *c* 的组合词作为候选词;

(3) 将出现次数最大和第二大的候选词作为算法知识名称。

4　实验结果与分析

4.1　实验数据

在实验中,根据 POJ[2] 这个在线判题系统上的

7 181 个ACM 竞赛题目的信息,从网络中下载了 25 107 个解题报告,删除相同的解题报告后共有 17 178 个页面。

解题报告的正文内容是根据网页标签来提取的。同一网站的解题报告中正文内容所在的标签是相同的。可以统计有多少种网站的类型,然后查找同一类型较多的解题报告的标签,这样可以提取到大部分解题报告的正文内容。根据 8 个知名大网站的解题报告正文标签,从下载的解题报告中提取了 15 425 个解题报告的正文内容,这些解题报告占到了总数的 90% 以上。在 15 425 个解题报告正文中,有分析内容的解题报告数量是 14 181 个,其占总数的比例是 91.9% ,从 14 181 个有分析内容的解题报告中提取到的句子总数是 149 256。

4.2 字符串模式

本文实验从 149 256 个正文句子(删除重复句子之后的个数)中提取到的包含专家库中算法知识名称的句子总数是 24 720,从中发现了它们的字符串模式"(s)*WS(s)*",如表 1 所示。表 1 中分词 WS 的 TF/ISF 值都大于 0.2。在所有包含专家库中算法知识名称的 24 720 条句子中,包含这 7 个分词中之一的句子所占的比例是 62.3% 。还有一些分词,例如"通常"和"求得",它们的 TF/ISF 值分别为 0.37 和 0.36,在所有句子中出现的次数并不多,所以没有将它们作为发现的字符串模式。

表 1 可能含算法知识名称的句子字符串模式

模式编号	WS	TF/ISF	模式的正则表达式
1	算法	0.47	(s)*算法(s)*
2	思路	0.22	(s)*思路(s)*
3	分析	0.23	(s)*分析(s)*
4	问题	0.42	(s)*问题(s)*
5	模板	0.22	(s)*模板(s)*
6	好题	0.31	(s)*好题(s)*
7	用	0.38	(s)*用(s)*

4.3 算法知识名称

16 097 个正文模式字符串,分词串个数和关键串个数也是 16 097。利用分词的特征值从关键串集 *KSP* 中获取的特征分词利用停用词表过滤之后的个数是 229 个。其中,180 个特征分词是专家库中算法知识名称包含的分词。对于特征词库中 49 个专家库中算法知识名称不包含的分词,它们可能会产生新的算法知识点。

将包含同一个特征分词的分词串分为一组,对其中的每个分词串利用一元、二元、三元、四元、五元分词建立组合词,选取其中包含该组对应的特征分词的组合词作为候选词。利用 229 个特征分词,将关键串分成了 229 个关键串子集,从关键串子集中共获得了 372 个算法知识名称,其中,准确的个数为 220。从候选词中获取 2 个出现次数最多的作为算法知识名称,出现不完整的算法知识名称的概率减小了,但是也增加了不准确的算法知识名称的个数。表 2 为部分专家库中没有的算法知识点。

表 2 部分专家库中没有的算法知识点

序号	算法知识点	序号	算法知识点
1	polya 定理	7	树的同构
2	stirling 公式	8	余弦定理
3	tarjan 算法	9	约瑟夫环
4	顶点覆盖问题	10	最大权闭合图
5	进位制问题	11	最大子段和问题
6	旅行商问题	12	最长不降子序列

4.4 解题报告标题中的算法知识名称

因为有些解题报告中没有分析内容,所以无法从正文中提取算法知识名称。本实验从解题报告标题中提取算法知识名称,尝试能否弥补这个不足。

(1) 网页标题标签:网页标题标签即 html 中的 < title > 标签。

(2) 解题报告网页标题:解题报告网页标题标签中的内容,即解题报告网页标题。

(3) 解题报告正文标题:解题报告正文标题包含在 html 页面的 headings 中。但是 headings 的内容包含了 < h1 > ~ < h6 > 6 个标签的内容。无法分清哪个标签中是解题报告正文的标题。解题报告正文标题也是网页标题的一个子串。一般来说 < title > 标签中"-",或"_"前的是解题报告的正文标题。

从网页标题截取的正文标题,删除正文标题中具体题目的名称和编号信息减少干扰字符串个数,之后得到的字符串个数是 16 261 个,删除重复项之后的个数是 6 680。分词串个数是 6 680,关键串个数是 6 680。从关键串获取的特征分词个数是129 个。其中,有 21 个是专家库中算法知识名称不包含的分词。最终提取的算法知识名称个数是 192,其中准确的个数是 124 个。

4.5 结果分析

目前还没有从文本中获取算法知识名称的研究。通过与专家库中原有的算法知识名称比较来判断本文方法,即算法知识发现方法(Method of Discovering Terms of Algorithmic Knowledge,MDTAK)的有效性。

MDTAK 获取的算法知识名称,人工地判断其中的准确性,计算出其准确率(P)和召回率(R),以及新发现的算法知识名称的个数。

$$P = AR/AT, R = NR/TR$$

其中,AR 为提取到的准确的算法知识名称个数;AT 为提取到的算法知识名称个数;NR 为包含提取到的

准确的算法知识名称的解题报告个数;*TR* 为获取到正文的解题报告数量。

对专家库中的文件也可以计算其召回率。MDTAK 获取的算法知识名称与专家库中的算法知识名称对比情况如表 3 所示。

表 3　TAK 和专家库中算法知识名称的对比情况

集合	总数	准确数	新发现数	准确率/%	召回率/%
TAK(1)	372	220	42	59.1	49.3
TAK(2)	192	124	23	64.6	40.5
专家库	455	455	–	–	70.2
扩充库	517	517	62	–	82.2

表 3 中集合指算法知识名称的集合,TAK(1)是从正文中发现的算法知识名称,TAK(2)是从标题中发现的算法知识名称,“专家库”指专家库中的算法知识名称,“扩充库”是将新发现的算法知识名称扩充进“专家库”之后的算法知识名称集合。“总数”是集合中算法知识名称的个数。“准确数”是集合中准确的个数。“新发现数”是 MDTAK 发现的专家库中没有的算法知识名称数。MDTAK 新发现了 62 个算法知识名称,“扩充库”的召回率相比原专家库提高了 10 个百分点以上。如果用专家库中的算法知识名称来识别解题报告中的算法知识点,就会有 10% 以上含算法知识点的解题报告中的知识点无法被识别,这些解题报告也无法被连接起来。

表 4 对本文方法发现的算法知识点和算法知识点名称进行了统计。“原个数”是从专家库中原有的知识点和知识名称个数。“新增个数”是发现的新的知识点或知识名称个数。

表 4　发现情况的统计

类型	原个数	新增个数	新增比例/%
算法知识点	214	24	11.2
算法知识名称	455	62	13.6

5　结束语

从大量的解题报告中获取算法知识名称,不仅需要耗费大量人力,而且可能出错。本文提出了一种基于自然语言处理的算法知识名称发现方法,弥补了传统方法的不足。随着网络中解题报告的更新,利用本文方法可以得到一个更加全面而且不断扩充算法知识点的专家库文件。该专家库文件用于识别解题报告中的算法知识点,然后将解题报告按知识结构联系在一起,可以为教学和学习提供全面而丰富的算法知识。

参考文献

[1] Zhu Guojin, Fu Lichao. Automatic Organization of Programming Resources on the Web [M]//Jin D, Lin S. Advances in Computer Science and Information Engineering. Berlin, Germany: Springer, 2012.

[2] Zhu Guojin. Automatic Discovery of Solutions to Programming Problems on the Web [C]//Proceedings of IEEE International Conference on Information Science and Technology. Phuket, Thailand: [s. n.], 2013.

[3] Han Jiawei, Cheng Hong, Xin Dong, et al. Frequent Pattern Mining: Current Status and Future Directions [J]. Data Mining and Knowledge Discovery, 2007, 15(1): 55-86.

[4] Yuan Junsong, Wu Ying, Yang Ming. Discovery of Collocation Patterns: From Visual Words to Visual Phrases [C]//Proceedings of the 25th IEEE Conference on Computer Vision and Pattern Recognition. [S. l.]: IEEE Press, 2007: 1-8.

[5] Jiao Hui, Liu Qian, Jia Huibo. Chinese Keyword Extraction Based on N-gram and Word Cooccurrence [C]//Proceedings of International Conference on Computational Intelligence and Security Workshops. [S. l.]: IEEE Press, 2007: 152-155.

[6] Resnik P. Semantic Similarity in a Taxonomy: An Information-based Measure and Its Application to Problems of Ambiguity in Natural Language [J]. Journal of Artificial Intelligence Research, 1998, 11: 95-130.

[7] Collobert R, Weston J, Bottou L, et al. Natural Language Processing (Almost) from Scratch [J]. Journal of Machine Learning Research, 2011, 12: 2493-2537.

[8] Maccartney B. Stanford Parser [EB/OL]. (2013-06-20). http://nlp. stanford. edu /software/lex-parser. shtml.

[9] Sproat R, Emerson T. The First International Chinese Word Segmentation Bakeoff [C]//Proceedings of the 2nd SIGHAN Workshop on Chinese Language Processing. Sapporo, Japan: [s. n.], 2003: 133-143.

[10] Li Juanzi, Fan Qi' na, Zhang Kuo. Keyword Extraction Based on TF/IDF for Chinese News Document [J]. Wuhan University Journal of Natural Sciences: English Vision, 2007, 12(5): 917-921.

[11] Rajaraman A, Ullman J D. Mining of Massive Datasets [M]. Cambridge, UK: Cambridge University Press, 2011.

编辑　顾逸斐