软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2018,29(8):2210−2225 [doi: 10.13328/j.cnki.jos.005533]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于 StackOverflow 数据的软件功能特征挖掘组织方法*

朱子骁[1,2], 邹艳珍[1,2,3], 华晨彦[1,2], 沈 琦[1,2], 赵俊峰[1,2,3]

[1](高可信软件技术教育部重点实验室(北京大学),北京 100871)

[2](北京大学 信息科学技术学院 软件研究所,北京 100871)

[3](北京大学(天津滨海) 新一代信息技术研究院,天津 300450)

通讯作者: 邹艳珍, E-mail: zouyz@pku.edu.cn

摘 要: 软件的功能描述文档是开发人员了解软件的重要基础.现有的软件项目并不都是具备全面描述软件功能的文档,但软件项目开发和应用过程中的各种交流记录蕴含了讨论其功能的大量信息.为此,提出了一种基于StackOverflow 问答数据的软件功能特征挖掘组织方法.该方法提出以动宾短语形式描述软件功能特征,挖掘并组织蕴含在StackOverflow数据中的软件功能特征,自动生成一种以层次化方式展示的软件项目功能特征文档.在针对真实项目的实验中,该方法生成的软件功能文档可以覆盖官方文档中列举的 97.6%的软件常用功能.同时,该方法可以扩展从不同形式的项目交流记录中生成全面描述软件功能特征的文档.

关键词: 软件复用;功能特征;软件文档;StackOverflow;自然语言句法分析;频繁子图挖掘

中图法分类号: TP311

中文引用格式: 朱子骁,邹艳珍,华晨彦,沈琦,赵俊峰.基于 StackOverflow 数据的软件功能特征挖掘组织方法.软件学报,2018,29(8):2210−2225. http://www.jos.org.cn/1000-9825/5533.htm

英文引用格式: Zhu ZX, Zou YZ, Hua CY, Shen Q, Zhao JF. Mining and organizing software functional features based on StackOverflow data. Ruan Jian Xue Bao/Journal of Software, 2018,29(8):2210−2225 (in Chinese). http://www.jos.org.cn/1000-9825/5533.htm

Mining and Organizing Software Functional Features Based on StackOverflow Data

ZHU Zi-Xiao[1,2], ZOU Yan-Zhen[1,2,3], HUA Chen-Yan[1,2], SHEN Qi[1,2], ZHAO Jun-Feng[1,2,3]

[1](Key Laboratory of High Confidence Software Technologies (Peking University), Ministry of Education, Beijing 100871, China)

[2](Institute of Software, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China)

[3](Beida (Binhai) Information Research, Tianjin 300450, China)

Abstract: Functional specification documents are very important for the developers who want to understand and reuse unfamiliar software libraries. Due to high cost of human effort and time, lots of software do not provide the official functional documentation. However, some software communication records produced in software developing processes contain valuable information regarding software functions and usages. In this paper, an approach is proposed to automatically mining and organizing functional features for open source software based on StackOverflow data. By describing functional features in the form of verb phrases, this approach generates hierarchical list of software functional features as the supplement of software documentation. In the experimental evaluation on some real-world subjects, the automatically generated documents have covered 97.6% of the frequent-used functional features in the official

* 基金项目: 国家重点研发计划(2016YFB1000801); 国家杰出青年科学基金(61525201)

Foundation item: National Key Research and Development Program of China (2016YFB1000801); National Natural Science Fund for Distinguished Young Scholars (61525201)

本文由数据驱动的软件智能化开发方法与技术专题特约编辑谢冰教授、魏峻研究员、彭鑫教授、孙海龙副教授推荐.

收稿时间: 2017-07-19; 修改时间: 2017-09-28; 采用时间: 2017-12-22; jos 在线出版时间: 2018-03-13

CNKI 网络优先出版: 2018-03-13 17:30:53, http://kns.cnki.net/kcms/detail/11.2560.TP.20180313.1730.015.html

documents. At the same time, the proposed approach can be adapted to different types of software communication records, and applied to software in different domains.

Key words: software reuse; functional feature; software documentation; StackOverflow; natural language syntax parsing; frequent subgraph mining

当前,通过互联网可获取的可复用软件资源的数量日益增多,使用方式也更加便利.开发人员广泛复用已有软件所提供的功能(例如 API 等)完成软件开发.开发人员在初始了解要复用的软件时,通常需要阅读软件的功能描述文档,以全面了解软件所提供的功能特征[1].

由于人工编写软件功能描述文档需要大量的时间,现有的软件项目并不都是具备全面描述软件功能的文档[2,3].但在软件项目的各种交流记录(例如开发者/使用者交互的邮件列表、记录功能更新和缺陷修改的事务追踪系统、讨论软件项目的公开论坛或者开放领域的问答网站)中,蕴含了大量讨论该软件功能特征的信息[4−6].通过对不同交流渠道积累的信息进行分析,可以挖掘出软件的功能特征以及功能特征间的关系,进一步按照一定规律将这些软件功能特征组织生成软件项目的功能描述文档,将能够辅助开发人员快速了解和上手利用软件.StackOverflow 是软件开发领域最常用的技术问答网站,其上蕴含了各软件项目或技术主题的大量问答记录,这些数据组织结构清晰、包含较多有价值的结构化信息、覆盖大多数常用软件项目且易于获取.

本文以 StackOverflow 数据为例,研究一种从软件项目交流记录中挖掘软件功能特征信息并自动生成软件功能特征文档的方法.

目前,已有一些基于各类数据对软件文档进行改进和自动生成的研究工作.典型的,Treude 等人[7,8]提出了开发任务(development task)的概念,将开发任务定义为描述特定编程操作的动宾短语,并提出了一种基于开发任务索引和浏览软件文档的方法.通过该方法,开发人员可以更高效地浏览文档,定位自身复用需求.但该工作并未针对非官方文档的软件交流记录进行挖掘分析,它仅将开发任务用以索引官方文档,并未对开发任务之间的关联进行分析,也没有给出一种显式、可读的开发任务组织结构.Zhang 等人[9]提出了一种从 StackOverflow 数据中识别 API 功能缺陷的方法,该工作基于情感分析技术,从 StackOverflow 网站中提取有负面评价的 API 特性,并与官方文档中的描述进行对照.Binkley 等人[10]、Murphy 等人[11]也以开发任务为关注点进行分析,其主要目标面向开发和维护过程而非面向复用.Wong 等人[12]和 Panichella 等人[13]分别以 StackOverflow 问答网站和开发者交流渠道为数据来源,为 API 类或方法抽取描述信息,以丰富软件文档的内容.Rastkar 等人[14]提出了一种对邮件、缺陷报告等软件交流数据生成文本摘要的技术.Petrosyan 等人[15]对软件文档进行分析,提出了一种利用文本分类技术提取 API 类型解释性信息的方法.Shepherd 等人[16]、Haiduc 等人[17]则通过分析程序代码来获取软件功能的相关信息.可以看出,从软件项目的交流记录中获取信息以改进或丰富软件文档的工作已经得到了很多关注,但这些工作在为软件自动生成全面的功能描述方面尚不完善.

本文提出了一种从 StackOverflow 数据中挖掘软件功能特征及其关系的方法,并基于该方法自动生成软件项目的功能描述文档.该方法提出以动宾短语形式描述软件功能特征,基于自然语言处理技术,采用启发式规则过滤的方法从 StackOverflow 问答交流数据中提取软件功能特征;基于频繁子图挖掘算法识别关联密切的功能特征并进行聚类,进而以层次化列表的形式对功能特征进行组织;按照功能特征的讨论频率进行排序,生成软件项目的功能描述文档.

本文第 1 节介绍软件功能特征等定义.第 2 节概述本文方法框架.第 3 节介绍从 StackOverflow 数据中提取与过滤功能特征的方法.第 4 节介绍基于频繁子图挖掘对功能特征进行聚类和层次化组织方法.第 5 节展示一个具体应用案例.第 6 节通过实验评估验证方法效果.第 7 节总结全文.

1 软件功能特征及其组织

软件功能是软件系统或部件的定义的目标或特征动作[18].传统上,文档中会用一段话来描述软件功能(包括功能的目标或动作、前提和后验条件、执行序列和输入输出等).在 Treude 等人[7,8]提出的“开发任务”概念以及 Zhang 等人[9]提出的“API 特征”概念中,都使用自然语言的动宾短语作为表述形式,描述程序或者功能接口的特

定行为.为此,本文提出以动宾短语的形式描述软件的功能特征(functional feature).软件功能特征通常由操作、对象、约束条件等部分组成,分别对应于动宾短语中的动词、作为宾语的名词短语和介词短语等成分.比如在开源软件项目 Apache POI 的 StackOverflow 问答记录中,出现了“set up print area”“iterate over cells”“merge cells”“convert a huge.csv file to excel”等动宾短语,都直观反映了该项目的功能特征.这样的功能特征短语简洁明确,易于开发人员理解,直接对应到复用该软件可以实现的编程目标或特征动作.

软件功能特征其实是一段软件功能描述文本的代称.软件项目的功能可以有两个层次.

- 一是从本项目代码角度看,所有本源 API 都是该项目对外提供的功能.我们将这一类称为“基本功能特征”.例如,“set up print area”功能对应着 Sheet 对象的 setPrintArea 方法.
- 另一些功能特征是开发人员编写一段客户端程序代码实现的复杂功能,例如实现“iterate over cells”功能需要使用循环控制结构,调用 Sheet 对象的 getRow 和 Row 对象的 getCell 方法协作完成;而“merge cells”功能需要 Sheet 对象的 addMergedRegion 和 CellRangeAddress 对象的构造方法协作完成.我们将这样的功能特征称为软件的复合功能特征.

通过分析软件源代码所提供的 API,我们就可以挖掘出该软件的基本功能特征,但复合功能特征反映了较复杂的软件复用方式,仅仅从代码角度难以获得.但在软件项目的讨论或问答等交流记录(比如 StackOverflow)中却有大量的相关信息,可以为挖掘复合功能特征提供充分的数据来源.

从软件项目交流记录中自动挖掘的功能特征会比较全面,数量也多,若是简单地列表展示,难以阅读.本文由此研究如何提供一种具有良好组织结构、易读、易懂的软件功能特征展示形式.在后续工作中,本文对挖掘出的软件功能特征以层次化的方式展示,并将存在语义关联的功能特征组织在一起.例如,前面举例的若干功能特征可以被组织成如图 1 所示的层次结构.同时,我们按照 StackOverflow 记录的讨论频率对功能特征进行排序,由此本文所生成的功能描述文档还可以起到类似软件项目 FAQ 或快速指南(quick guide)的作用.

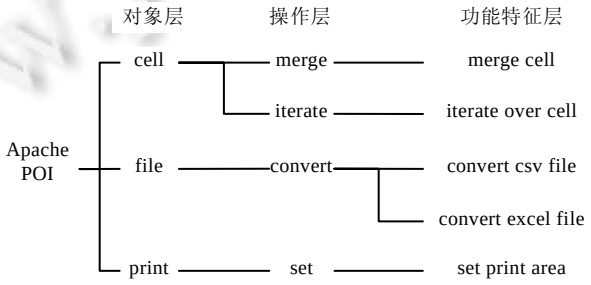

Fig.1 An example of hierarchical functional feature list

图 1 层次化的功能特征列表示例

2 方法概述

本文提出了一种从软件项目的交流记录中挖掘软件功能特征、分析功能特征间关联关系,并生成层次化的软件功能特征文档的方法.图 2 展示了该方法的整体框架图.

该方法可以分为两个主要阶段.

(1) 基于句法分析与启发式规则的功能特征提取与过滤

首先从软件项目的交流记录中提取并过滤功能特征,得到功能特征的候选集合.该阶段包括 3 个主要步骤:(a) 对原始数据进行预处理,解析出包含功能特征的文本片段;(b) 对文本片段中的语句进行句法分析,从句法树中提取动宾结构的短语作为候选短语;(c) 将候选动宾短语通过一组基于启发式规则的过滤器进行筛选,保留置信度评分高于一定阈值的动宾短语作为候选功能特征.

(2) 基于频繁子图挖掘的功能特征聚类与组织

在候选功能特征基础上,对功能特征进行聚类和层次化的组织,得到层次化的功能特征列表.该阶段包括 3

个主要步骤:(a) 基于频繁子图挖掘算法挖掘出最经常出现的功能特征,并将具有相同子结构的功能特征聚入同一类;(b) 对功能特征进行层次化的组织,并对每一层次的列表元素按照出现频率排序;(c) 建立功能特征与代码元素关联,形成完整的层次化功能特征文档.

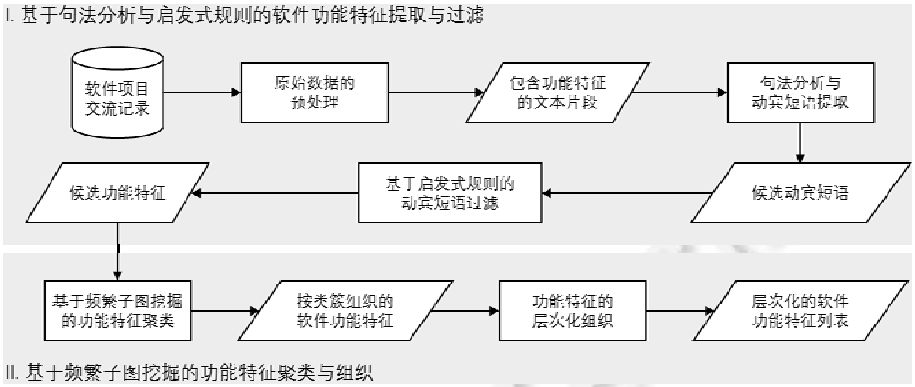

Fig.2 Approach framework

图 2 方法框架

3 基于句法分析与启发式规则的功能特征提取与过滤

StackOverflow 网站的帖子是自由格式的文本段落.从图 3 展示的一个 StackOverflow 帖子可以看出其中讨论软件功能特征的基本情况,这个帖子的内容是使用 Apache POI 软件编辑 Excel 文件,动宾短语“convert a huge .csv file to excel”是该软件的一项功能特征,其目标是将 csv 格式文件转化成 excel 格式的文件.但是每个 StackOverflow 帖子都可能包含大量的动宾短语,并不是每一个动宾短语都在描述一个软件功能特征.比如在图 3 中,只有实线框圈出的内容与功能特征相关,而虚线框中的动宾短语就与功能特征无关.因此,在从 StackOverflow 数据中提取功能特征时,如何准确地挖掘出功能特征,是主要的技术挑战.

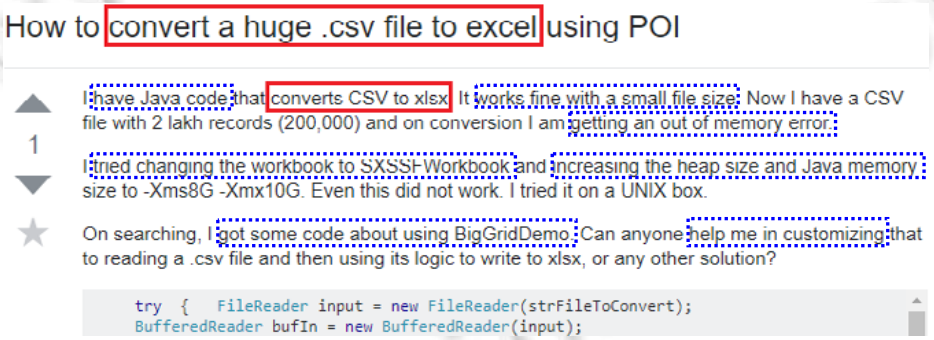

Fig.3 Functional features in software Q&A records: convert a huge .csv file to excel

图 3 软件问答记录中的功能特征:convert a huge .csv file to excel

本文提出了一种基于句法分析与启发式规则的功能特征提取与过滤方法,该方法包括原始数据预处理、句法分析与动宾短语提取、基于启发式规则的动宾短语过滤这 3 个主要过程.

3.1 原始数据预处理

本文以 StackOverflow 公开发布的数据存档文件(data dump)作为数据源.通过分析帖子的标签(tag)信息,形成一个软件项目相关所有帖子集合.对 StackOverflow 帖子的 HTML 格式文本,移除 HTML 标签后,将文本分段、分句,作为后续处理过程的输入.

StackOverflow 帖子包含了大量的代码内容,通过解析特定(如 pre,code 等标签)的 HTML 元素,我们将文本中单独成段的代码片段以及在文本段落中的代码表达式识别并移除,确保后续的句法分析正常进行.另外,许多语句中还嵌入了未被特定 HTML 标签管辖的代码元素,其中的代码符号和特殊名称会导致句法解析时产生错误.为此,本文使用了一组正则表达式规则对嵌入在语句中的代码元素进行识别.这组规则是基于 Dagenais 等人工作[19]中提出的在文本中识别代码表达式的规则修改而来.对于语句中识别出的代码元素,本文将其替换为一个特殊标记符号"CODE#",使其在句法解析时被标注为专有名词."#"表示该代码元素的编号,以便在句法解析后将其恢复为原本的代码元素名.

3.2 句法分析与动宾短语提取

在软件文档和交流记录中,功能特征通常被表示为动宾短语的形式.本文使用斯坦福自然语言句法解析器(Stanford NLP parser)[20,21]对自然语言语句进行句法分析,建立句法结构树.该解析器基于一个经过训练优化的概率上下文无关文法(PCFG)对语句的句法树进行分析构造.比如,语句"I'm trying to develop a complex report, and I need to set up the print areas for the excel file."在该文法下的解析结果如图 4 所示.该文法使用的语法标注和词性标注遵循 Penn Treebank 的约定[22−24].常用标注如:S,完整语句;VP,动词性短语;VB.*,不同形式的动词(VB.*表示以 VB 开头的所有标注);NP,名词性短语;NN.*,不同形式的名词;JJ,形容词;PP,介词短语;IN,介词.

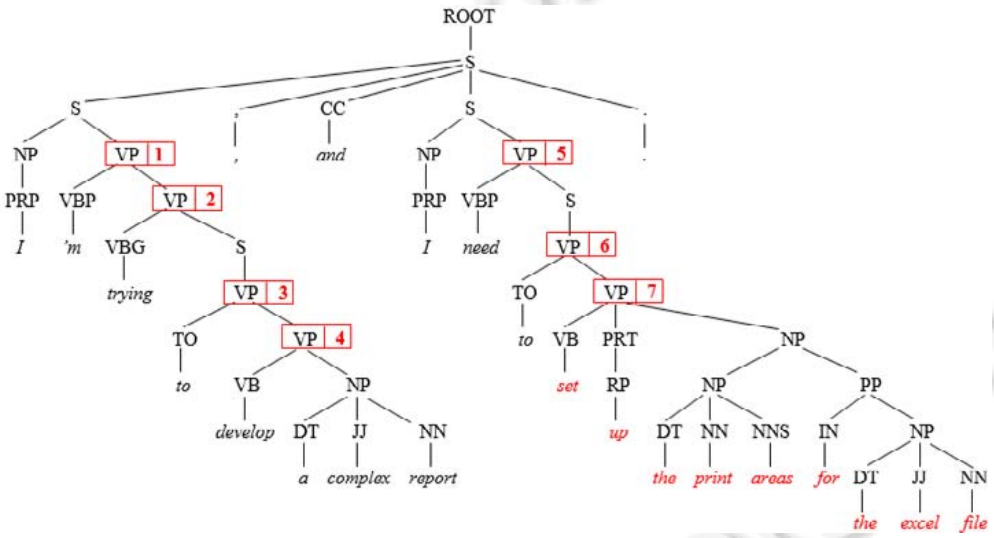

Fig.4 Parse tree of the sentence "I'm trying to develop a complex report, and I need to set up the print areas for the excel file."

图 4 语句"I'm trying to develop a complex report, and I need to set up the print areas for the excel file."的句法结构树

提取出句法结构树中标注为 VP 的子树所对应的句子成分,可以得到语句中全部的动宾短语.但是从图 4 例子中不难看出,仅依靠句法结构提取出的动宾短语中包含大量并不描述功能特征的无关短语.比如在该例中,只有编号第 4 个和第 7 个的 VP 标注下的动宾短语可以作为功能特征的候选,而只有第 7 个动宾短语是该语句中恰当的功能特征.这种现象在自由格式的自然语言文本数据中是很常见的,因此在提取出动宾短语后,需要研究一种方法过滤掉与功能特征无关的短语.

3.3 基于启发式规则的动宾短语过滤

自然语言文本中的动宾短语并不都与功能特征有关,需要进行过滤.现有的研究工作中,Zhang 等人[9]从 StackOverflow 数据中分析有缺陷的"API 特征",采用了基于官方文档的过滤方法:提前解析文档中的动宾短语

作为 API 特征的基准集合,将从 StackOverflow 提取的动宾短语与官方文档中提取出的特征进行匹配.Treude 等人[7]从软件官方文档中提取动宾短语作为开发任务,采用了白名单过滤法:事先分析实验项目的样本数据,人工构造一个白名单动词表,只有当动宾短语中的动词出现在白名单中时,才将其保留下来.白名单过滤法可以提高准确度,却降低了结果的召回率和覆盖率.因为该工作仅限于分析官方文档和特定的软件项目,其目标是便于官方文档的索引和浏览,所以该方法是可行的.本文工作希望应用到不同领域的软件项目、不同来源数据的文本分析中,并建立全面的软件功能特征文档,所以需要更完备的过滤方法.

首先,该方法对每个动宾短语设置了一个置信度属性,用来衡量该短语作为候选功能特征的可信程度.置信度包含一个得分和一组证据信息.每个动宾短语的置信度初值被设为 0,取值范围为[−∞,+∞].

其次,本文提出了一组启发式规则对所有候选动宾短语进行检验.当短语匹配到一条规则时,在置信度中添加一条证据信息.最终,根据每个动宾短语的置信度证据信息,对正面证据按一定规则加分,对负面证据按一定规则减分,最终得分低于所设定阈值的动宾短语将被过滤掉,得到功能特征的候选集合.本文所采用的过滤规则综合考虑了短语不同方面的特征,共包含 3 类启发式规则:基于短语结构特征的过滤规则、基于停用词表的过滤规则、基于上下文特征的过滤规则.

3.3.1 基于短语结构特征的过滤规则

根据动宾短语的句法结构特征,可以大幅度缩小候选短语的范围.在动宾短语的句法结构树中,如果树根结点的第 1 层子结点中不包含动词(VB.*)或者名词成分(NP),则可以将其直接判定为无效的动宾短语,将该短语的置信度评分设为最低分,直接结束过滤过程.例如,图 4 例子中的第 3 个、第 6 个动宾短语,其第 1 层子结构中不包含动词成分,第 1 个、第 2 个、第 5 个动宾短语中不包含名词成分,都属于无效的动宾短语.

3.3.2 基于停用词表的过滤规则

有些特殊单词不会出现在合法的功能特征中或者是功能特征的某些位置上,如果候选动宾短语中出现了这些“停用词”,可以直接将短语移除出候选列表,或者添加置信度的负面证据.本文定义了 3 类停用词.

- 特殊的语法成分

助动词(例如 be,do,have 等)和情态动词(例如 can,may,must 等)作为语句的组成结构,经常出现在动宾短语中,但功能特征中不需要这种成分.例如,图 4 例子中的第 1 个和第 5 个动宾短语,分别以助动词和情态动词开头.这类语法成分的过滤规则如下.

a) 当助动词或情态动词出现在候选动宾短语的动词位置时,将该短语的置信度评分设为最低分,直接结束过滤过程;
b) 当助动词或情态动词出现在候选动宾短语的其他位置(例如约束条件中)时,将该短语的置信度评分减 1.

当代词(例如 it,that,me,them 等)出现在候选短语中时,通常会因为指代不明使短语不能准确表达其原有含义.对于这种情况,将其过滤掉或者调低置信度评分.

a) 当代词出现在候选动宾短语的名词位置,并且是最后的核心名词时,将该短语的置信度评分设为最低分,直接结束过滤过程;
b) 当代词出现在候选动宾短语的名词位置,但不是核心名词时,将该短语的置信度评分减 2;
c) 当代词出现在候选动宾短语的其他位置时,将该短语的置信度评分减 1.

- 表示讨论、求助或问答语义的词或词组

在 StackOverflow 上讨论功能特征时,通常有许多表示讨论、求助或者提问、回答的词或词组,这些词可能预示着后文将有重要的讨论内容,但包含这些词本身的短语通常不是有效的功能特征,图 4 例子中的第 2 个和第 5 个动宾短语都属于这类情形.这种类型的部分动词如 ask,answer,help,hope,try,want,plan 等.这种类型的部分名词包括 requirement,solution,issue,suggestion,advice 等.

a) 当该类型的动词、名词出现在候选动宾短语的约束条件(介词短语或从句等)中时,将该短语的置信度评分减 1;

b) 当该类型的动词、名词出现在候选动宾短语的其他位置时,将该短语的置信度评分设为最低分,直接结束过滤过程.

- 软件开发术语

StackOverflow 的文本中经常出现软件开发的专业技术术语或者程序的关键词等,例如 extend、implement、include、object、method 等.出现这类词的短语通常只是描述某种具体的编程行为,并不描述软件的功能特征,图 4 例子中的第 4 个动宾短语就属于这类情形.

a) 该类型的动词、名词出现在候选动宾短语的动词位置或名词短语中的核心名词位置时,将该短语的置信度评分减 2;

b) 当该类型的动词、名词出现在候选动宾短语的其他位置时,将该短语的置信度评分减 1.

3.3.3 基于上下文特征的过滤规则

短语的上下文特征对于判断其是否更可能是功能特征有着很显著的作用.本文将短语的上下文特征也分为几类进行处理.

- 短语位置

我们发现,从文档、网页的标题中提取的候选短语通常与该文讨论的软件功能密切相关.另外,出现在一篇文档、一个网页、一条帖子中开头或结尾的文字,也更倾向于描述真正的功能特征.因此,对出现在特殊位置的短语,添加置信度的正面证据.

a) 当短语出现在标题位置时,将该短语的置信度评分加 2;

b) 当短语出现在文段的第 1 句或者最后一句话时,将该短语的置信度评分加 1.

- 表示讨论、求助或问答的先导词或词串

在基于停用词表的过滤规则中,表示讨论、求助或者问答的词本身所在的动宾短语通常不是有效的功能特征,但可能预示着后文紧邻的内容会与功能特征或发帖人意图密切相关.例如,图 4 例子中的第 5 个动宾短语“need to set up...”并不是一个有效的功能特征短语,但“need to”引导的动宾短语“set up...”就是一个有效的功能特征.此外,还有一些可能引导后文功能特征的标志性词串,例如 how to,how can (I),have problem to 等.

a) 当先导词或词串出现在紧邻候选短语之前的位置时,将该短语的置信度评分加 2;

b) 当先导词或词串出现在候选短语所在语句中之前但不紧邻的位置时,将该短语的置信度评分加 1.

- 否定性语句或短语

很多 StackOverflow 的帖子会提出开发中遇到的问题,或描述已经尝试过的错误解决方案,这类内容通常是描述复用某软件时错误或者易错的情况[9],应该从候选短语中移除.当短语的上下文中出现否定词(例如 not, never 等)时,降低其置信度评分.

a) 当否定词出现在紧邻候选短语前的位置时,将该短语的置信度评分设为最低分,直接结束过滤过程;

b) 当否定词出现在候选短语所在语句中之前但不紧邻的位置时,将该短语的置信度评分减 2.

4 基于频繁子图挖掘的功能特征聚类与组织

功能特征通常由操作、对象、约束条件等部分组成,按照功能特征的对象和操作,可以将功能特征进行分类.但因为功能特征中还存在内容相似约束条件不同的情形,同时,StackOverflow 中关于相同或相似功能特征的讨论也较多,具有相同对象和操作的功能特征数量依旧很多,全部展示出来会导致功能特征文档可读性差,因此,有必要将相似功能特征聚类在一起[25],着重展示其中具有代表性的功能特征.

本文提出了一种基于频繁子图挖掘对相似或相关的功能特征进行聚类的方法,进而以层次化列表的形式对功能特征进行组织,按照功能特征的讨论频率进行排序,生成软件项目的功能描述文档.

4.1 基于频繁子图挖掘的功能特征聚类

首先对功能特征的表示结构进行归一化调整.例如,图 5 中的句法树(b)展示了图 4 例子中第 7 个动宾短语“set up the print area for the excel files”的句法解析树;图 5 中的句法树(a)展示了另一个功能特征“set the print

area”的句法树.可以看到:这两个功能特征存在共同成分“set the print area”,但是两个句法树之间的最大公共子图是(VP (VB set) (NP))和(NP (DT the) (NN print)).这是因为两棵句法树的上层结构不同,存在名词短语(NP)的嵌套结构,无法提取出共同成分所在的子句法树.

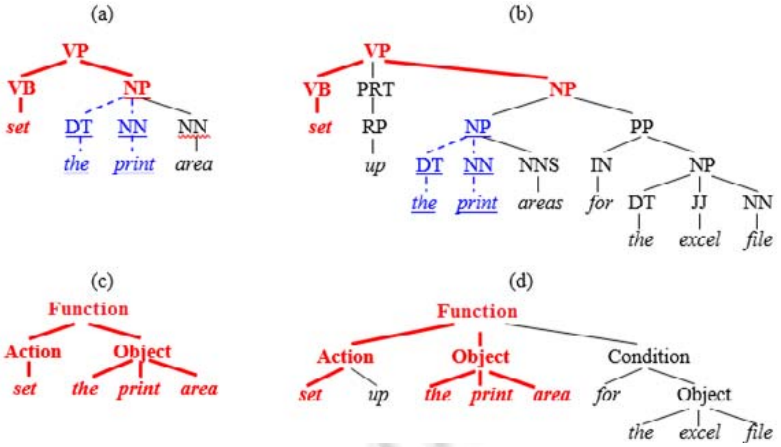

Fig.5 Comparison between parse tree and normalized tree in mining frequent subtrees

图 5 功能特征的句法树表示和统一文法表示在挖掘频繁子树时的效果比较

短语之间多层嵌套的情形在基于 Stanford NLP Parser 解析得到的自然语言句法树中常见,为了能使这类句法解析树结构不同、实质上语义相似的短语在应用频繁子图挖掘算法时产生预期的结果,本文给出了一种功能特征的统一文法,见表 1.

Table 1 Normalized grammar of functional features

表 1 功能特征的统一文法

Function	::=	*Action Object* {*Condition*}
Action	::=	*verb* [*particle*]
Object	::=	{*dt*} {*adj*} {*noun*} *noun*
Condition	::=	*prep* [*verb*] *Object*

该文法采用 BNF 范式进行定义,大写字母开头的符号表示文法中的非终结符,全小写字母的符号表示文法中的终结符,对应于自然语言中的单词.花括号(“{}”)表示括号中的符号串可以重复 0 到多次,方括号(“[]”)表示括号中的符号串是可选的(可以出现 0 或 1 次).本文法的非终结符 Function,Action,Object,Condition 分别表示功能特征、功能特征中的操作、功能特征中的对象和功能特征中的约束条件.终结符 verb,particle,dt,adj,noun,prep 分别对应于自然语言中的小品词、冠词、形容词、名词和介词.可以看出,非终结符 Function,Object,Condition 基本对应于自然语言中可能嵌套了多层子短语的动词短语(VP)、名词短语(NP)、介词短语(PP).基于此文法,将功能特征的原句法结构树转换成符合统一文法的树结构,可以将原短语中的各部分单词准确对应到功能特征的各组成要素,并有效地解决了频繁子图挖掘时可能遇到的问题.

再回到图 5 中的例子,如果将句法树归一化为符合表 1 统一文法的表示结构后,句法树(a)和句法树(b)所对应的短语分别被表示为句法树(c)和句法树(d)的格式.此时,功能特征的表示结构树已经统一,在此基础上应用频繁子图挖掘算法,可以有效提取功能特征中的频繁子结构,即最经常出现的功能特征.本文使用的频繁子图挖掘算法是该领域的经典算法 gSpan 算法[26].为了满足对功能特征进行聚类的目标,根据以下规则对该算法进行扩展和设定.

a) 频繁子图中必须包含动词、核心名词两个成分.

b) 算法中频繁子图的最小支持度(即出现频度)被设为 0.01%.

c) 移除被其他频繁子图完全覆盖的频繁子图.当一个频繁子图是另一个频繁子图的子图,并且两个子图的出现频度一样时,可以认为较小的频繁子图并不会以其他搭配形式出现在其他功能特征结构中,此时只保留最大的频繁子图.

d) 挖掘出所有频繁子图后,将所有的候选功能特征归入与其最相似的频繁子图类簇下,以其共同的频繁子图所对应的功能特征短语作为该类簇名称.

4.2 软件功能特征的层次化组织

在完成对功能特征的聚类之后,对功能特征按照如下定义的层次化结构进行组织,并对每一层元素的频率进行统计.

(1) 对象层:所有功能特征可以按照所涉及的问题域中的概念或对象来进行组织,这些概念或对象通常对应于功能特征的操作对象,也就是功能特征中的名词成分.

(2) 操作层:对于处在对象层同一对象类别下的所有功能特征,可以按照功能特征中施加在对象之上的操作进行第 2 层组织,也就是功能特征中的动词成分.

(3) 聚类功能特征层:对于具有相同对象和操作的功能特征,按照其中相似或相同功能特征的频繁子图聚类结果,形成功能特征列表的第 3 层.

(4) 关联代码层:在功能特征层之下,将与该功能特征相关的代码元素(例如 API 类或者方法)与功能特征关联起来,作为功能特征列表的最底层.

本文还对层次化列表中每一层的每一项功能特征或列表元素给出了在 StackOverflow 上讨论的频率,并按照讨论频率的高低进行了排序.频率统计的方式为:

a) 对于聚类功能特征层的元素,即每一个具有代表性的功能特征频繁子结构而言,其频率为具有该公共子结构的所有功能特征对应短语的出现频率之和;

b) 对于操作层的元素,其频率为对相同对象执行相同操作的所有聚类功能特征层元素的频率之和;

c) 对于对象层的元素,其频率对同一对象施加操作的所有操作层元素的频率之和.

4.3 建立功能特征与代码元素之间的关联

在层次化的软件功能文档最底层,还需建立功能特征和与其相关的代码元素之间的关联,以定位到相关的 API 类或者方法.Dagenais 等人[19]、Gao 等人[27]提出了一种将文档中的代码词与代码中的实际代码元素建立关联,进而建立文档与代码元素之间关联的方法.本文在其方法基础上建立功能特征与代码元素之间的关联,以下给出了该过程的伪代码.

```
Input: feature cluster list clusterList in clustered functional feature layer
Output: feature clusters with related code elements ranking list
Procedure:
1.  for each feature cluster c in clusterList
2.      for each feature f in c.features
3.          List⟨TextFragment⟩tfList=findTextFragementContainingFeature(f);
4.          for each text fragment t in tfList
5.              List⟨CodeElement⟩ceList=findRelatedCodeElements(t);
6.              for each code element ce in ceList
7.                  if (c.relatedCodeElementList,contains(ce))
8.                      ce.count++;
9.                  else
10.                     ce.count=1;
11.                     c.relatedCodeElementList.add(ce);
12.     rankByCECount(c.relatedCodeElementList);
```

对于聚类功能特征层的每一个元素,处理该类簇下的所有功能特征,通过之前从文档中挖掘功能特征时建立的关联索引,获取包含该功能特征的原始文本片段(*findTextFragementContainingFeature* 方法),如第 1 行~第 3 行所示;然后,使用 Dagenais 等人提出的方法[19]对上一步中获取到的所有片段进行分析,找出与这些文本片段关联的代码元素(*findRelatedCodeElements* 方法),对每个关联的代码元素,统计其出现频次(第 6 行~第 11 行);最终将这些代码元素按照发现关联的频次进行排序,该结果作为功能特征与代码元素之间的关联.

5 应用实例

本文以 JFreeChart 项目为例,介绍方法所生成的功能特征文档.JFreeChart 项目是一个绘制专业数据图表的 API 代码库.JFreeChart 项目的 StackOverflow 数据包含 2 065 个问答主题、4 440 条帖子、6 374 条评论信息.将原始数据经过预处理和句法分析后提取出 87 838 个动宾短语,经过启发式规则过滤后,保留了 10 126 个候选功能特征.经过功能特征聚类和组织,最终生成的层次化功能特征列表中共包含 37 个对象层元素、67 个操作层元素以及聚类后得到的 239 个功能特征.本方法为该项目所生成的层次化功能特征列表如图 6 所示.

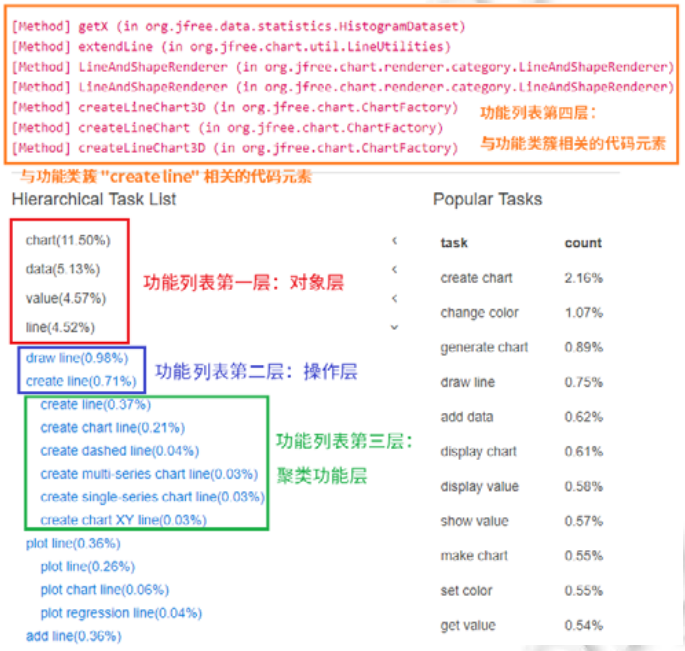

Fig.6 Hierarchical functional feature list of JFreeChart

图 6 JFreeChart 项目的层次化功能特征列表

(1) 对象层中展示了 JFreeChart 项目的分层功能特征列表中对象层频率最高的部分元素.可以看到:作为一个绘制图表的 API 库,JFreeChart 项目中最常出现的对象是 chart.其他经常出现的对象涉及到绘制图表时的数据源(data,value)、绘图元素(line,color,label 等)、图表类型(bar,plot 等)等.

(2) 操作层中选取图 6 所示的对象“line”下的子层次进行展示.可以看到:在对象层“line”类目下的操作层中,出现频度最高的操作是“draw”,其他经常出现的操作还包括 create,plot,add 等.

(3) 聚类功能特征层中展开显示了“line+create”和“line+plot”等两个操作层类目下的聚类功能特征层元素.可以看到:在“line+create”类目下的功能特征中,具体包括了创建虚线、多数据系列图表线、单数据系列图表线、创建 XY 图表线等功能特征.

(4) 关联代码层中展示了与“create line”这个功能特征类簇相关的代码元素,可以看到,包括了 *getX*(), *extendLine*(),*createLineChart3D*()等 API 方法.

从该例还可以看到,功能列表的右栏展示了整个软件中讨论热度最高、最频繁使用的功能特征.

6 实验评估

为了进一步验证文本工作,我们选取若干软件的 StackOverflow 数据进行实验验证,选择的软件项目包括 Apache Nutch,Apache Jena,JFreeChart,Apache POI 和 Apache Lucene 等.这 5 个项目分别来自不同的应用领域:网络爬虫、语义网构造、图表绘制、Office 文档处理和搜索引擎.选择这几个项目的原因是:(1) 这几个项目都比较成熟,使用比较广泛,可以获取到更多用于实验评估的数据和官方文档;(2) 覆盖不同类型的应用领域;(3) 实验评估人员对这些项目比较熟悉,便于在实验中人工构造基准数据集.部分软件项目具有一定的功能描述文档,例如 Apache POI 项目的文档比较详尽,其官网提供了“Busy Developers’ Guide”(https://poi.apache.org/spreadsheet/quick-guide.html),以列表方式展示了最常见的功能特征;Apache Lucene 有《Lucene in Action》一书,按章节组织介绍了 Lucene 的各类功能.而另一些项目缺少相应完善的功能描述文档,如 Apache Nutch,Apache Jena,JFreeChart 等.

本文挖掘组织上述项目的功能特征并进行了实验评估.在实验中,分析以下两个问题.

(1) 本方法生成的功能特征是否覆盖软件项目已有的官方功能描述?对于 POI 和 Lucene 项目来说,已有的官方描述成为客观的对比目标.

(2) 对缺少完善的功能描述文档的软件项目,通过开发人员的主观评估来确定本方法的性能,具体包括准确率和召回率等.

6.1 问题1:与官方文档的客观对比

POI 的“Busy Developers’ Guide”文档中列举了 49 条常见功能.Lucene 的《Lucene in Action》一书的第 1 部分,Lucene 核心技术,描述的是 Lucene 项目的主要功能,共有 216 个各级章节.我们逐一检视这些功能描述,判断其是否出现在本文方法生成的功能特征列表中.其中,对 Lucene 项目,我们分析章节标题中涉及的功能特征.本文算法生成的功能特征描述和人工编制文档中的描述可能用词不完全一致,本文关注意义相同的功能特征覆盖问题.对于不一致的字词描述给出了如下的一种评价打分方法(见表 2).

Table 2 Grading standards of comparing the generated functional feature list against the official documents

表 2 生成的功能特征与官方文档进行对照时的评分标准

评分	生成的功能特征对官方文档中功能的覆盖情况
4	包含完全相同的功能特征
3	包含该功能特征,但描述方式不同
2	包含对象与操作相同或相似,但约束条件不同的功能特征
1	仅出现了该功能特征中的相似对象或操作
0	未能覆盖

(1) POI 项目分析

表 3 展示了本文方法生成的 POI 项目功能特征列表与 POI 的“Busy Developers’ Guide”文档中的 49 项功能特征的对比情况.表格后两列分别展示了每一档评分的功能特征数量在所有官方文档功能特征数量中的占比和累积占比.

Table 3 Functional feature coverage of our generated list on the Busy Developers’ Guide of POI

表 3 本文生成的功能特征列表对 POI “Busy Developers’ Guide”中功能特征的覆盖情况

评分	官方文档中功能特征的评分分布	占比(%)	累积占比(%)
4	27	55.10	55.10
3	12	24.49	79.59
2	7	14.29	93.88
1	1	2.04	95.92
0	2	4.08	100.00
总计	**49**	平均分	**3.245**

可以看到:评判为包含相同功能特征(4 分)或包含内容相同但描述不同的功能特征(3 分)的情形占比达到了 79.59%,包含相似对象或操作以及更加匹配的情形(1 分及以上)的覆盖率达到了 95.92%.POI 文档中,全部功能特征的平均评分达到 3.245(满分 4 分).有 2 项完全没有被包含在生成的功能特征列表中.通过分析发现:这两个功能特征都是因为在 StackOverflow 网站上没有任何讨论记录,因此无法挖掘出来.

(2) Lucene 项目分析

本文与《Lucene in Action》一书的对比分两个部分进行.该书的前 4 章介绍了 Lucene 软件的基础功能特征,包括索引、检索、解析文本等,第 5 章、第 6 章介绍了 Lucene 软件的高阶功能特征,分别是高阶搜索技术(advanced search techniques)和在 Lucene 核心用法上进行扩展与自定义的内容(extending search).本文将基础功能和高阶功能分开与本文方法进行对比,进一步分析本文方法所生成的功能列表的覆盖特点.

我们从《Lucene in Action》基础功能部分的 142 个不同级别的章节标题中分析出了 76 项功能特征,对比情况见表 4.可以看到:评判为包含相同功能特征(4 分)或包含内容相同但描述不同的功能特征(3 分)的情形占比达到了 86.84%,包含相似对象或操作以及更加匹配的情形(1 分及以上)的覆盖率达到了 98.68%.76 项功能特征中仅有 1 项因为 StackOverflow 网站上没有讨论记录而缺失.功能特征的平均覆盖评分达到 3.513(满分 4 分).

Table 4 Functional feature coverage of our generated list on the basic features in the book 《Lucene in Action》

表 4 本文生成的功能特征列表对《Lucene in Action》中基础功能的覆盖情况

评分	《Lucene in Action》基础功能特征		
	评分分布	占比(%)	累积占比(%)
4	52	68.42	68.42
3	14	18.42	86.84
2	8	10.53	97.37
1	1	1.32	98.68
0	1	1.32	100.00
总计	**76**	平均分	**3.513**

我们从《Lucene in Action》高阶功能部分的 74 个不同级别的章节标题中分析出了 47 项功能特征,对比情况见表 5.评判为包含相同功能特征(4 分)或包含内容相同但描述不同的功能特征(3 分)的情形占比只有 48.94%,包含相似对象或操作以及更加匹配的情形(1 分及以上)的覆盖率为 82.98%.文档中 17.02%的功能特征完全没有被包含在生成的功能特征列表中.这 47 条功能特征的平均评分只有 2.404(满分 4 分).

Table 5 Functional feature coverage of our generated list on the advanced features in the book 《Lucene in Action》

表 5 本文生成的功能特征列表对《Lucene in Action》中高阶功能的覆盖情况

评分	《Lucene in Action》高阶技术功能特征		
	评分分布	占比(%)	累积占比(%)
4	17	36.17	36.17
3	6	12.77	48.94
2	11	23.40	72.34
1	5	10.64	82.98
0	8	17.02	100.00
总计	**47**	平均分	**2.404**

可以看出,本文方法生成的功能特征列表对于基础功能的覆盖情况明显优于对于高阶功能的覆盖情况.基础功能介绍的是 Lucene 项目最常见、最常用的技术,高阶技术内容则介绍了 Lucene 项目中一些比较少被使用和讨论的“冷门”功能点.本文方法对常用功能的覆盖度优于不常用功能,这和本文使用的 StackOverflow 数据密切相关:(1) 部分不常用的功能特征可能尚未被讨论过或讨论较少;(2) 基于频繁子图挖掘的功能特征聚类方法在将高频出现的相似功能特征进行合并的同时,还会将出现频率低于阈值、缺少相似功能的功能特征删除,某些不常用功能特征可能虽然在原始数据中被讨论过,但因为频率较低而被过滤掉.

POI 文档和《Lucene in Action》一书的章节标题并不涵盖项目最全面的功能特征,但是也可以检视本文方

法自动生成功能特征的有效性.

综合两个项目,本文方法对官方文档中列举的常用功能特征(POI 文档列举的全部功能和《Lucene in Action》一书中的基础功能)达到了 97.6%的覆盖率(相似或相同),对官方文档中 84%的功能特征可以生成完全一致或者同样内容不同描述方式的功能特征,总体平均评分为 3.408(最高 4 分).

6.2 问题2:缺失官方文档项目的主观对比分析

对于 Apache Nutch,Apache Jena,JFreeChart 等缺少完善的功能描述文档的开源软件项目,我们通过人工阅读文档和代码、构造基准数据集的方式,进行本文方法的准确率和召回率评估.

(1) 基准数据集的构造

为研究问题(2),首先需要知道一个软件包含的全部功能特征都有哪些,实验中选择为软件项目构造功能特征的基准数据集(benchmark).实验邀请了 6 位北京大学计算机专业研究生,都具有至少 4 年的 Java 编程经验,并熟悉实验项目中的至少 1 项.由他们人工构造每个实验项目的基准数据集,并进行评分.方法如下.

a) 为每个实验项目分配 2 名标注人员;
b) 将每个实验项目的官方文档(包括可以获取到的教程、开发者指南、FAQ 等)提供给标注人员,要求他们提取出项目文档中的功能特征;
c) 标注者根据经验判断是否将 2 个包含相同或相似的动词、名词短语的功能特征合并为 1 个;
d) 2 位评判者分别列出功能特征清单后,进行 2 人标注结果的比对和综合,形成该项目功能特征的基准数据集.

(2) 准确率的评估分析

请构造基准数据集的 2 位开发者作为评判人员,浏览本文方法生成的功能特征列表,并对其中每个条目的准确性进行打分.评判者被要求对所生成的功能特征列表中每一层的每一个元素(功能特征或者动词/名词)进行评分,评分标准分为 3 档,见表 6.

Table 6 Grading standards of evaluating precision of the functional features

表 6 评估功能特征准确率的评分标准

评分	功能特征在基准数据集中的情况
2	被评分项可以在基准数据集中找到描述相同内容的相同功能特征
1	被评分项可以在基准数据集中找到相似的功能特征,例如操作或对象相同或相似,但可能存在约束条件不同或不完整等问题
0	被评分项不存在于基准数据集中,或者无意义、与功能特征无关

对于每一个评分项,收集 2 位评判者的评分,取其平均值作为该项得分,计算每个实验项目的功能特征列表中每一层元素的平均得分,并按照每一个元素在原始数据集中的出现频率进行加权,计算加权平均得分.

表 7 显示了本文方法的准确率评估结果.表中第 3 列显示的是通过功能特征提取和过滤得到的候选功能特征数量,第 4 列显示的是经过聚类得到的每一层的列表元素数量,第 5 列~第 7 列显示的是每一档评分的功能特征计数,最后两列显示的是功能特征评分的平均值和按频率加权的平均值.

可以看出,在跨项目的整体统计数据上,所生成的功能特征列表的第 1 层对象层~第 3 层聚类功能特征层分别有 58.1%,41.9%,44.9%的数据项被评为 2 分,也就是功能特征列表中的条目准确地对应于基准数据集中一个条目的情形.3 层中分别有 29.7%,38.5%,33.5%的条目可以在基准数据集中找到相似的功能特征.在这 3 层中分别有 12.2%,19.6%,21.6%的功能特征被评为 0 分,也就是错误识别的功能特征项.总体来看,本文生成的功能特征列表中 79.9%的数据项可以在基准数据集中找到相同或相似的功能特征,在这一项上表现最好的 Jena 项目可以达到 89.8%的相同/相似比例.

从全部实验项目的平均评分来看,本文方法所生成的功能特征的平均得分是 1.249 分(最高 2 分),说明所生成的功能特征有较好的准确率.在加权计算的平均分中,3 层功能特征的平均得分分别比不加权的情形高了 8.9%,7.4%和 7.8%,这说明本文方法生成的功能特征列表中,出现频率越高的功能特征项,生成得越准确.

JFreeChart 项目的功能特征列表的准确率最高,而 Nutch 项目的功能特征列表的准确率最低.Nutch 项目的文档主要在讨论软件的部署、命令行运行,因此可能该软件的官方文档本身对于基于 API 的复用方式的介绍就不够全面,导致基于官方文档构造的基准数据集不够完整,影响了功能特征列表的准确率评估结果.

Table 7 Precision scores of the generated functional feature list

表 7 生成的软件功能特征列表的准确率评估结果

实验对象		候选功能特征数量	功能列表各层元素数	准确性评分分布			相同或相似功能的比例(%)	平均得分	频率加权平均得分
				2	1	0			
Nutch	对象层	6 199	19	9	5	5	73.68	1.211	1.336
	操作层		134	47	43	44	67.16	1.022	1.167
	聚类功能层		172	81	40	51	70.35	1.174	1.231
Jena	对象层	9 139	18	7	9	2	88.89	1.278	1.350
	操作层		69	25	39	5	92.75	1.290	1.307
	聚类功能层		99	46	41	12	87.88	1.343	1.247
JFreeChart	对象层	10 126	37	27	8	2	94.59	1.676	1.837
	操作层		67	41	22	4	94.03	1.552	1.608
	聚类功能层		239	102	90	47	80.33	1.230	1.435
总计	对象层	**25 464**	**74**	**43**	**22**	**9**	**87.84**	**1.459**	**1.590**
	操作层		**270**	**113**	**104**	**53**	**80.37**	**1.222**	**1.312**
	聚类功能层		**510**	**229**	**171**	**110**	**78.43**	**1.233**	**1.330**

(3) 召回率的评估分析

评判实验项目的基准数据集与本文方法生成结果,计算本文方法的召回率.根据字词的匹配情况,评分标准见表 8.同样,收集 2 位评判者的评分,取其平均值作为该项得分.

Table 8 Grading standards of evaluating recall of the functional features

表 8 评估功能特征召回率的评分标准

评分	基准数据集中的功能特征在自动生成列表中的情况
2	生成的功能特征列表中包含描述相同内容的相同功能特征
1	生成的功能特征列表中包含相似的功能特征,但功能特征内容不完全一致,例如操作或对象相同或相似,但可能存在约束条件不同或不完整等问题
0	生成的功能特征列表中未能包含基准数据集中的该项目

表 9 显示了本文方法的召回率评估结果.可以发现:本文方法准确覆盖了基准数据集中 40.3%的功能特征,还有 25.9%可以找到相似的功能特征;基准集中有大约 33.8%的功能特征并没有被挖掘出来;平均 66.21%的基准集功能特征可以找到相同或相近的功能特征.使用基准集对本文方法所生成的功能特征列表进行评判的平均得分为 1.065 分(最高 2 分).

Table 9 Recall scores of the generated functional feature list on the benchmark

表 9 本文生成的功能特征列表在人工标注数据集上的召回率评估结果

实验对象	基准数据集中的功能特征数量	基准数据集中功能特征的评分分布			平均得分	相同或相似功能特征的比例(%)
		2	1	0		
Nutch	85	30	22	33	0.965	61.18
Jena	90	39	19	32	1.078	64.44
JFreeChart	118	49	35	34	1.127	71.19
总计	**293**	**118**	**76**	**99**	**1.065**	**66.21**

与人工构造的基准数据集比对的召回率低于之前与官方文档比对时的覆盖率,有两个可能的影响因素.(1) 人工构造的功能特征集要比项目文档中列举的常见功能特征覆盖面更广;(2) 标注人员熟悉项目情况,会抽取出一些更加细节化的描述,本文方法基于的 StackOverflow 数据未必涉及这些细节.例如在 JFreeChart 项目中,基准数据集中的功能特征相对更加简短,该项目的覆盖率得分水平就高于其他两个项目.

7 结束语

软件的功能描述文档对于开发人员初始了解软件的功能特征具有重要意义.很多软件项目并不具备全面描述软件功能的文档,但软件项目的各种交流记录,如邮件、缺陷报告、问答记录等,蕴含了讨论该软件功能的大量信息.本文提出了一种基于功能特征的功能描述文档自动生成方法,以动宾短语形式描述软件功能特征,挖掘并组织软件功能特征,生成一种以层次化方式展示的功能特征文档.

本文以真实 StackOverflow 数据为基础进行了实验分析.本文方法能够从成熟、被广泛使用而具有丰富数据的软件项目的 StackOverflow 数据中挖掘出该软件的功能特征.本文方法按照讨论频率对功能特征进行排序,也更有益于开发人员发现和理解项目的常用功能.根据与人工编纂的官方文档对比可以看出,本文方法生成的软件功能文档对官方文档中的常用功能特征达到了 97.6%的覆盖率(相似或相同),对官方文档中 84%的功能特征可以生成完全一致或者同样内容不同描述方式的功能特征,总体平均评分为 3.408(最高 4 分).

我们也看到,这类基于数据方法的成效也严重依赖于数据集的丰富程度.本文方法是对文本类型的交流记录的处理.在未来工作中,我们考虑将软件项目中的其他讨论记录,如邮件列表、版本更新、事务追踪系统或论坛、博客等信息增加进来,将能够取得更好的效果.在现有功能文档的基础上,我们还可以为每项功能特征补充文本描述信息,进一步挖掘功能特征之间的关联关系,并提供更加准确的功能与 API 之间的关联关系,为开发者在软件复用时提供更多帮助.

References:

[1] Robillard MP, Deline R. A field study of API learning obstacles. Empirical Software Engineering, 2011,16(6):703−732.

[2] Robillard MP. What makes APIs hard to learn? Answers from developers. IEEE Software, 2009,26(6):27−34.

[3] Scaffidi C. Why are APIs difficult to learn and use? Crossroads, 2006,12(4):No.4.

[4] Parnin C, Treude C, Grammel L, Storey MA. Crowd documentation: Exploring the coverage and the dynamics of API discussions on Stack Overflow. Technical Report, Georgia Institute of Technology, 2012.

[5] Wang H, Peng X, Yu H, Zhao WY. Research on interaction process of question and answer in social software development. Computer Applications and Software, 2017,34(5):1−11 (in Chinese with English abstract).

[6] Liu CM, Guo Y, Yu XM, Zhao L, Liu Y, Cheng XQ. Information extraction research aimed at open source Web pages. Journal of Frontiers of Computer Science and Technology, 2017,11(1):114−123 (in Chinese with English abstract).

[7] Treude C, Robillard MP, Dagenais B. Extracting development tasks to navigate software documentation. IEEE Trans. on Software Engineering, 2015,41(6):565−581.

[8] Treude C, Sicard M, Klocke M, Robillard M. TaskNav: Task-Based navigation of software documentation. In: Proc. of the Int'l Conf. on Software Engineering (ICSE). IEEE, 2015. 649−652.

[9] Zhang Y, Hou D. Extracting problematic API features from forum discussions. In: Proc. of the Int'l Conf. on Program Comprehension (ICPC). IEEE, 2013. 142−151.

[10] Binkley D, Lawrie D, Hill E, Burge J, Harris I, Hebig R, Keszocze O, Reed K, Slankas J. Task-Driven software summarization. In: Proc. of the Int'l Conf. on Software Maintenance (ICSM). IEEE, 2013. 432−435.

[11] Murphy GC, Kersten M, Robillard MP, Cubranic D. The emergent structure of development tasks. In: Proc. of the ECOOP. 2005. 33−48.

[12] Wong E, Yang J, Tan L. Autocomment: Mining question and answer sites for automatic comment generation. In: Proc. of the Int'l Conf. on Automated Software Engineering (ASE). IEEE, 2013. 562−567.

[13] Panichella S, Aponte J, Di Penta M, Marcus A, Canfora G. Mining source code descriptions from developer communications. In: Proc. of the Int'l Conf. on Program Comprehension (ICPC). IEEE, 2012. 63−72.

[14] Rastkar S, Murphy GC, Murray G. Summarizing software artifacts: A case study of bug reports. In: Proc. of the Int'l Conf. on Software Engineering (ICSE). ACM Press, 2010. 505−514.

[15] Petrosyan G, Robillard MP, De Mori R. Discovering information explaining API types using text classification. In: Proc. of the Int'l Conf. on Software Engineering (ICSE). IEEE, 2015. 869−879.

[16] Shepherd D, Fry ZP, Hill E, Pollock L, Vijay-Shanker K. Using natural language program analysis to locate and understand action-oriented concerns. In: Proc. of the Int'l Conf. on Aspect-Oriented Software Development. ACM Press, 2007. 212−224.

[17] Haiduc S, Marcus A. On the use of domain terms in source code. In: Proc. of the Int'l Conf. on Program Comprehension (ICPC). IEEE, 2008. 113−122.

[18] General Administration of Quality Supervision, Inspection and Quarantine of the P.R.C, Standardization administration of the P.R.C. GB/T 11457-2006: Information Technology Software Engineering Terminology. 2006 (in Chinese).

[19] Dagenais B, Robillard MP. Recovering traceability links between an API and its learning resources. In: Proc. of the Int'l Conf. on Software Engineering (ICSE). IEEE, 2012. 47−57.

[20] Manning CD, Surdeanu M, Bauer J, Finkel JR, Bethard S, McClosky D. The Stanford CoreNLP natural language processing toolkit. In: Proc. of the ACL (System Demonstrations). 2014. 55−60.

[21] Klein D, Manning CD. Accurate unlexicalized parsing. In: Proc. of the Annual Meeting on Association for Computational Linguistics, Vol.1. Association for Computational Linguistics, 2003. 423−430.

[22] Levy R, Andrew G. Tregex and tsurgeon: Tools for querying and manipulating tree data structures. In: Proc. of the Int'l Conf. on Language Resources and Evaluation. 2006. 2231−2234.

[23] Santorini B. Part of speech tagging guidelines for the Penn Treebank Project (3rd Revision). Department of Computer and Information Science, University of Pennsylvania, 2009.

[24] Bies A, Ferguson M, Katz K, MacIntyre R, Tredinnick V, Kim G, Marcinkiewicz MA, Schasberger B. Bracketing guidelines for Treebank II style Penn Treebank project. University of Pennsylvania, 1995. 97−100.

[25] Zhao W, Zhang L, Mei H, Sun JS. A functional requirement based hierarchical agglomerative approach to program clustering. Ruan Jian Xue Bao/Journal of Software, 2006,17(8):1661−1668 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/17/1661.htm [doi: 10.1360/jos171661]

[26] Yan X, Han J. gSpan: Graph-Based substructure pattern mining. In: Proc. of the Int'l Conf. on Data Mining. IEEE, 2002. 721−724.

[27] Gao Y, Liu H, Fan XZ, Niu ZD. Method name recommendation based on source code depository and feature matching. Ruan Jian Xue Bao/Journal of Software, 2015,26(12):3062−3074 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4817.htm [doi: 10.13328/j.cnki.jos.004817]

附中文参考文献:

[5] 王海,彭鑫,于涵,赵文耘.社交化软件开发问答中的交互过程研究.计算机应用与软件,2017,34(5):1−11.

[6] 刘春梅,郭岩,俞晓明,赵岭,刘悦,程学旗.针对开源论坛网页的信息抽取研究.计算机科学与探索,2017,11(1):114−123.

[18] 中华人民共和国国家质量监督检验检疫总局,中国国家标准化管理委员会.GB/T 11457-2006:信息技术 软件工程术语.2006.

[25] 赵伟,张路,梅宏,孙家骕.一种基于功能需求层次凝聚的程序聚类方法.软件学报,2006,17(8):1661−1668. http://www.jos.org.cn/1000-9825/17/1661.htm [doi: 10.1360/jos171661]

[27] 高原,刘辉,樊孝忠,牛振东.基于代码库和特征匹配的函数名称推荐方法.软件学报,2015,26(12):3062−3074. http://www.jos.org.cn/1000-9825/4817.htm [doi: 10.13328/j.cnki.jos.004817]

朱子骁(1990－),男,湖南郴州人,博士,主要研究领域为软件复用,软件资源理解,知识挖掘.

沈琦(1995－),男,博士生,主要研究领域为软件工程,软件资源管理与复用.

邹艳珍(1976－),女,博士,副教授,CCF 专业会员,主要研究领域为软件复用,软件资源管理,软件数据挖掘,知识图谱.

赵俊峰(1974－),女,博士,副教授,CCF 高级会员,主要研究领域为软件复用与构件技术,智慧城市数据分析与挖掘,知识工程.

华晨彦(1994－),男,硕士生,主要研究领域为软件工程,软件资源管理与复用.