面向缺陷知识的多特征匹配搜索算法

李 斌[1,2],陈定山[1],孙小兵[1,2,3],薄莉莉[1,2,3]
(1. 扬州大学信息工程学院,江苏扬州 225127; 2. 江苏省知识管理与智能服务工程研究中心,江苏扬州 225127;
3. 南京大学计算机软件新技术国家重点实验室,江苏南京 210023)

摘 要: 缺陷数据分析正成为软件工程领域的热点,现有缺陷分析技术无法有效处理复杂和冗余的缺陷数据,以高效地辅助缺陷修复工作. 本文提出一种多特征匹配搜索算法——MMSBK (Multi-feature Matching Search Algorithm for Bug Knowledge). 首先对缺陷问题进行分析,抽取其包含的缺陷实体及关系;然后,基于实体和关系匹配将缺陷问题与缺陷知识图谱关联,通过知识图谱的关联性和可视化帮助软件开发搜索缺陷知识;最后,基于匹配算法生成的缺陷关系三元组生成搜索结果子图. 实验验证了 MMSBK 算法的有效性.

关键词: 缺陷知识图谱; 知识搜索; 缺陷实体; 缺陷关系
中图分类号: TP311.5 **文献标识码:** A **文章编号:** 0372-2112 (2021)04-0661-04
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.12263/DZXB.20200327

Multi-Feature Matching Search Algorithm for Bug Knowledge

LI Bin[1,2], CHEN Ding-shan[1], SUN Xiao-bing[1,2,3], BO Li-li[1,2,3]
(1. *School of Information Engineering, Yangzhou University, Yangzhou, Jiangsu* 225127, *China*;
2. *Jiangsu Engineering Research Center of Knowledge Management and Intelligent Service, Yangzhou, Jiangsu* 225127, *China*;
3. *State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, Jiangsu* 210023, *China*)

Abstract: Bug data analysis is becoming a hotspot in the software engineering domain. The accumulation of bug knowledge requires redefinition of a new search method to effectively process complex and redundant bug data to efficiently assist bug fixing. This paper proposes a multi-feature matching search algorithm for bug knowledge (MMSBK). First, we analyze the bug question and extract the bug entities and relations. Then, based on bug entity and relation matching, the bug question is associated with the bug knowledge graph. Finally, the search sub-graph is generated based on the bug triples generated by the matching algorithm. The experiment shows the effectiveness of MMSBK.

Key words: bug knowledge graph; knowledge search; bug entity; bug relation

1 引言

现有的缺陷分析工作主要是搜索相似缺陷以辅助缺陷修复[1]. 目前常用的搜索方法是基于关键词[2,3],这一搜索方式存在搜索结果范围广,以及拥有低质量和低相关性的缺点.

缺陷数据量巨大,且有着复杂的知识结构,传统的基于关键词的搜索方式不再适用于复杂且关联性强的缺陷数据. 智能化的搜索方法相比于传统的搜索方法可以快速、精准的定位相关搜索结果,并且能够返回层次更深、内容更广的搜索结果. 智能化知识搜索研究应运而生[4]. 也有一些缺陷数据分析技术[4,5]从实体和关系入手,抽取缺陷知识. 现阶段软件工程领域出现了针对软件知识的抽取工作[6,7]和具体应用[8]. 本文基于缺陷知识抽取工作[4,5]提出一种多特征的缺陷实体关系匹配方法. 首先构建缺陷领域知识图谱,增强缺陷问题搜索的关联性;再通过缺陷实体和关系的抽取,解决缺陷数据的歧义性问题,而且不需要制定特定的查询规则,便于用户针对复杂的缺陷问题进行搜索.

2 MMSBK 匹配算法

2.1 算法概述

MMBKS 算法包括两个部分:(1)基于自然语言处理的缺陷问题处理,并以缺陷三元组的形式存储;(2)缺陷三元组实体、关系匹配搜索,并展示当前缺陷问题的搜索子图.

收稿日期:2020-04-01;修回日期:2020-10-06;责任编辑:李勇锋
基金项目:国家自然科学基金(No. 61872312, No. 61972335, No. 62002309);扬州市校合作项目(No. YZU201803);南京大学计算机软件新技术国家重点实验室资助项目(No. KFKT2020B15, No. KFKT2020B16);扬州大学"高端人才支持计划"

第一阶段的任务是理解用户输入的缺陷问题,并进行自然语言处理、依存句法分析,抽取出缺陷问题三元组 $QTripe(NP_1,VP_1,NP_2)$,其中 NP_1 表示缺陷问题中的主语,NP_2 表示宾语,动词短语 VP_1 表示主语 NP_1 与宾语 NP_2 之间的语义关系.

第二阶段的任务是基于缺陷问题三元组与缺陷领域知识图谱中的三元组进行匹配,并通过子图的形式将相似的缺陷数据关联起来,帮助用户理解与搜索相关缺陷.

2.2 缺陷领域知识图谱构建

基于现有缺陷知识抽取工作[3],抽取出缺陷知识的关系三元组,并基于图形数据库 Neo4j (https://neo4j.com/)实现缺陷实体和关系的存储以构建图谱.

首先,抽取的缺陷知识三元组可批量导入到图形数据库中;其次,通过 Neo4j 自带的 Cypher 查询语句进行实体查询,查看节点和关系,以获取当前节点的概况;最后,搜索实体和关系信息以获取缺陷知识. 当前的缺陷领域知识图谱除了 16 种缺陷实体类型[2]的节点和 8 种缺陷关系[3]外还包括:(1) BugReport,用以存储缺陷报告对应的缺陷 ID;(2) SDNewnode,用以存储缺陷实体的同义词、反义词;(3) Describe 边,用以描述缺陷实体节点与对应缺陷报告的包含关系.

2.3 缺陷问题三元组抽取

首先对用户输入的搜索问句 S 进行自然语言处理,并通过依存句法分析形成缺陷问题的语法依赖树 T,然后根据 T 分析词与词之间的依存关系即语法关系. 再将生成的语法依赖树 T 作为缺陷实体和关系联合抽取模型[3]的输入,抽取出问题语义三元组集合 $QTripe$. 用户输入的问题语句可抽取出多个三元组,因此抽取结果为集合. $QTripe$ 存储所有经过语法依赖树、抽取模型抽取出的三元组 qt. 每一个三元组 qt 包括三个元素:主语、关系与宾语,表示成 $qt(NP_1,VP_1,NP_2)$.

2.4 双特征缺陷实体匹配

2.4.1 基于字符层面的实体匹配

用户输入的缺陷问题是自然语言语句,虽然对这些问题进行了自然语言处理,但是很难将所有处理过的单词、短语与知识库中的实体进行完全、严格的匹配. 因此,本节提出基于字符层面的实体匹配,让搜索结果更丰富,获取初步实体匹配三元组集合 $CTripe$. 具体步骤为:(1)求取问题语义三元组 qt 中头实体 q_h、尾实体 q_t 与缺陷知识图谱三元组库中每个三元组 gt 中头实体 g_h、尾实体 g_t 的字符相似度 p_h 与 p_t. 具体公式如下:

$$P(q_h,g_h)=1-\frac{n(q_h,g_h)}{\max(l(q_h),l(g_h))} \tag{1}$$

其中 q_h 和 g_h 分别表示问题语义三元组的头实体与缺陷知识图谱三元组库中三元组的头实体,$n(q_h,g_h)$ 表示实体 q_h、g_h 不同字符的个数,$l(q_h)$ 表示实体 q_h 的长度,$l(g_h)$ 表示实体 g_h 的长度. 通过上式计算输入问句形成的三元组与缺陷知识图谱中三元组头实体的字符相似度 p_h. 再计算当前两个三元组尾实体的字符相似度 p_t;(2)判断实体相似度 p_h、p_t 与设定阈值 f_1 之间的关系. 若 p_h、p_t 均大于设定阈值 f_1,则该实体对应的缺陷知识图谱三元组库中三元组与该问题语义三元组在字符层面匹配. 默认情况下,将阈值 f_1 设置为 0.9,并将目标实体与知识库中所有实体标签进行匹配,如果相似度满足 f_1,那么对应的三元组就是候选三元组集合;(3)软件缺陷知识图谱三元组库中所有与问题语义三元组匹配的三元组 t 构成初步实体匹配三元组集合 $CTripe$.

2.4.2 基于语义层面的实体匹配

软件缺陷中有大量的词不相似却拥有相似的含义. 比如:词“error”、“bug”与“fault”等都是常见的表示缺陷的词,若通过字符层面进行相似度匹配,很难找到三者的共同性,而挖掘这些词可以丰富搜索结果. 因此,本节提出基于语义相似度的实体匹配方法,获得最终实体匹配三元组集合. 目前,计算词与词之间语义相似度采用的主流方法是 word2vec[9]模型. 通过 word2vec 模型学习文本,并利用词向量来表示词的语义信息,使得语义相似的单词在空间内距离更近. 本节针对三元组之间的头实体与尾实体进行语义相似度度量,具体步骤为:(1)采用基于 Skip-Gram 模型的 word2vec,将问题语义三元组中的实体与初步实体匹配三元组集合 $CTripe$ 中每个三元组 t 中的头实体和尾实体均映射为空间向量;(2)头实体、尾实体的词向量表示隐含了具体实体词的结构特征,可通过求取空间向量之间的余弦距离作为实体相似度,具体公式如下:

$$Q_h(\boldsymbol{q}_h,\boldsymbol{g}_h)=1-\frac{\boldsymbol{q}_h*\boldsymbol{g}_h}{\|\boldsymbol{q}_h\|\times\|\boldsymbol{g}_h\|} \tag{2}$$

$\boldsymbol{q}_h$, $\boldsymbol{g}_h$分别表示问题语义三元组 qt 和知识图谱三元组 gt 中头实体经过 word2vec 模型生成的空间向量,$\|\boldsymbol{q}_h\|$ 与 $\|\boldsymbol{g}_h\|$ 是头实体向量的二次范式. 每个三元组均获得两个实体相似度 Q_h、Q_t,判断实体语义相似度 Q_h、Q_t 与设定阈值 f_2 之间的关系. 默认情况下,将阈值 f_2 设置为 0.85,若 Q_h、Q_t 均大于设定阈值 f_2,则知识图谱三元组 gt 与问题语义三元组 qt 在语义层面相匹配. 更新初步实体匹配三元组集合 $CTripe$.

2.4.3 缺陷关系匹配

文献[4]定义的 8 种缺陷关系,每一种关系都有多种表达方式. 将缺陷问句与知识图谱对应的关系映射也需要从字符以及语义两个层面展开. 首先,针对某条问题语义三元组 qt 的缺陷关系 q_r 与缺陷知识图谱中的关系依次进行字符层面和语义层面的双特征匹配,以

获取实体匹配三元组集合中满足关系 q_r 的对应三元组集合 $RTripe(r)$,具体公式为:

$$RTripe(q_r)=\{(e_i,e_j):(e_i,q_r,e_j)\in GTripe\} \quad (3)$$

其中 $1\leqslant i\leqslant j\leqslant n$,$e_i$、$e_j$ 表示缺陷实体节点,下标 r 表示对应的关系,n 表示软件缺陷知识图谱中缺陷实体总数,$GTripe$ 表示缺陷知识图谱实体匹配三元组集合,q_r 表示所有与关系 q_r 字符相似、语义相似的关系. 其次,将实体匹配的三元组集合 $CTripe$ 与关系匹配的三元组集合 $RTripe$ 进行实体对齐,并返回问答三元组 $ATripe$. 整体算法如下.

算法 1　MMSBK 算法

输入　缺陷问句 S、匹配三元组集合 $CTripe$、缺陷知识图谱三元组库 $GTripe$

输出　三元组集合 $ATripe$

1. for $w_i\in S$ do
2.　分词、词性标注、词性还原等处理操作 $l(w_i)$
3. end for
4. 依存句法分析,生成语法依赖树 T
5. 作为实体、关系联合抽取模型输入,生成问题语义三元组 $QTripe(NP_1,VP_1,NP_2)$
6. for $qt\in QTripe$ do
7.　for $gt\in GTripe$ do
8.　　计算 qt、gt 中头和尾实体的字符相似度 p_h 和 p_t
9.　　计算 qt、gt 中头和尾实体的语义相似度 Q_h 和 Q_t
10.　　if ($P_h\geqslant f_1$ and $P_t\geqslant f_1$) or ($Q_h\geqslant f_2$ and $Q_t\geqslant f_2$) then
11.　　更新三元组集合 $CTripe$, $CTripe=CTripe+t$
12.　　end if
13.　end for
14. end for
15. for $qt\in QTripe$ do
16.　for $gt\in GTripe$ do
17.　　计算 qt、gt 中关系 r_c 和 g_c 字符相似度 P_r
18.　　计算 qt、gt 中关系 r_c 和 g_c 语义相似度 Q_r
19.　　if ($P_r\geqslant f_1$ and $Q_r\geqslant f_2$) then
20.　　更新三元组集合 $RTripe$
21.　　$RTripe=RTripe+gt$
　　end if
22.　end for
23. end for
24. 将三元组集合 $CTripe$、$RTripe$ 实体对齐,合并成最终问题三元组集合 $ATripe$
25. return $ATripe$

算法的输入包括用户输入的缺陷问句 S,其中 1 ~ 5 行是对用户输入的问句进行自然语言处理和三元组抽取工作,并生成问题语义三元组 $QTripe$. 6 ~ 14 行将 $QTripe$ 与缺陷知识图谱中的三元组 $GTripe$ 进行双特征实体匹配. 15 ~ 23 行将语义三元组 $QTripe$ 与知识图谱中的三元组 $GTripe$ 进行双特征关系匹配. 最终,返回经过实体匹配和关系匹配生成的问题三元组 $ATripe$.

2.4.4　知识子图搜索

MMBKS 算法围绕上述的缺陷问题三元组匹配结果进行搜索,返回最终的搜索子图. 首先,针对匹配三元组集合 $ATripe$ 中的每个三元组 at,$at=(e_s,r_t,e_m)$,式中 $1\leqslant s\leqslant m\leqslant n$,$e_s$、$e_m$ 表示缺陷实体节点,r_t 表示缺陷实体对应的关系. 其次,将每个三元组 at 表示为一条边,若两个边共享某一个实体节点,则将三元组对应的实体节点合并,重复该过程直至遍历完整个匹配三元组集合,以此将匹配三元组集合 $ATripe$ 合并为一个结构化的搜索结果子图,并返回给用户.

3　实验结果与分析

3.1　数据集

实验基于 Mozilla 项目构建了缺陷数据集,包含文献[3]中的 1000 条 Mozilla 项目的缺陷报告. 重点分析 150 个缺陷报告,并构建其对应的缺陷问题和答案,共抽取到缺陷问题三元组 210 条. 三元组中包含实体 378 个、关系 162 条. 抽取其中的 100 条三元组作为训练集以训练算法中涉及的阈值,其他 50 条样本当作测试数据集.

3.2　评价指标

从准确率 P、召回率 R 和 $F1$ 值评估 MMBKS.

$$P=\frac{TP}{ET}$$

$$R=\frac{TP}{GT}$$

$$F1=\frac{2\times P\times R}{P+R} \quad (4)$$

其中 TP 表示正确抽取到的缺陷实体和关系,ET 表示抽取到的总的缺陷实体和关系,GT 表示缺陷领域知识图谱中的缺陷实体和关系.

3.3　与基于关键词搜索策略的对比实验

实验从时间、召回率、准确度三个维度与基于关键词的搜索方式进行对比. 从 150 条样本问题中随机选取 100 条缺陷问题,并以 10 个为一组,共分成 10 组进行验证. 如果一个缺陷问题包含多个关键词,我们将这些关键词以“和”、“或”的关系进行连接以统一进行搜索.

结果如图 1 所示,基于实体、关系的匹配搜索方式相比于基于关键词的搜索方式用时较少,平均每组时间约为 100s.

图 2 从 $F1$ 值比较了两种搜索算法. 基于实体、关系匹配的搜索方式平均 $F1$ 值达到了 0.805,相比于关键词策略的 $F1$ 值 0.695 有了较大提升.

3.4　不同特征引入下算法的整体表现

为进一步分析 MMBKS 匹配算法的有效性,分别对基于字符、语义、实体和关系的多特征匹配进行实验. 评估结果如表 1,仅基于字符层面的实体匹配时得到的准

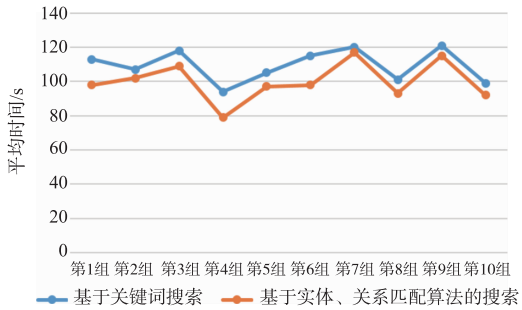

图1 两种搜索方式时间比较

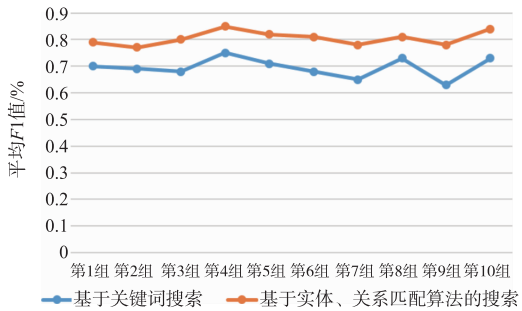

图2 两种搜索方式性能比较

确率虽然较高,但召回率最差,只有 61% 的精度. 仅基于语义层面的实体匹配相对于字符匹配来说召回率上升到 69%,但准确率下降到 79%. 当加入了关系进行匹配后,虽然准确率提升不大,但是召回率有明显提升,达到 78%. 综合考虑准确率和召回率,完整的实体、关系匹配算法最好的 $F1$ 值达到了 83%.

表 1 MMBKS 算法的性能表现

	P	R	F
基于字符层面相似度	0.87	0.61	0.72
基于语义层面相似度	0.79	0.69	0.74
基于字符 + 语义相似度	0.86	0.71	0.78
基于字符 + 语义 + 关系相似度	0.89	0.78	0.83

4 结束语

本文提出一种面向缺陷知识的多特征匹配搜索算法的缺陷知识搜索算法(MMBKS),从缺陷实体、关系的角度理解和分析缺陷问题. 实验验证了 MMBKS 相比于基于关键词方法的优势.

参考文献

[1] GU Z, BARR E T, SCHLECK D, et al. Reusing debugging knowledge via trace-based bug search[J]. ACM SIGPLAN Notices, 2012, 47(10): 927 – 942.

[2] 马慧芳,刘芳,夏琴,等. 基于加权超图随机游走的文献关键词提取算法[J]. 电子学报,2018,46(6):1410 – 1414.
MA Hui-fang, LIU Fang, XIA Qin, et al. Keywords extraction algorithm based on weighted hypergraph random walk [J]. Acta Electronica Sinica, 2018, 46(6): 1410 – 1414. (in Chinese)

[3] 赵京胜,朱巧明,周国栋,等. 自动关键词抽取研究综述[J]. 软件学报,2017,028(009):2431 – 2449.
ZHAO J, ZHU Q, ZHOU G, et al. Review of research in automatic keyword extraction [J]. Journal of Software, 2017, 028(009): 2431 – 2449. (in Chinese)

[4] ZHOU C, LI B, SUN X, et al. Recognizing software bug-specific named entity in software bug repository[A]. Proceedings of the 26th Conference on Program Comprehension[C]. IEEE, 2018. 108 – 119.

[5] CHEN D, LI B, ZHOU C, et al. Automatically identifying bug entities and relations for bug analysis[A]. 2019 IEEE 1st International Workshop on Intelligent Bug Fixing (IBF) [C]. IEEE, 2019. 39 – 43.

[6] ZHAO X, XING Z, KABIR M A, et al. HDSKG: Harvesting domain specific knowledge graph from content of webpages [A]. IEEE International Conference on Software Analysis Evolution and Reengineering[C]. IEEE, 2017. 56 – 67.

[7] YE D, XING Z, FOO C Y, et al. Software-specific named entity recognition in software engineering social content [A]. IEEE International Conference on Software Analysis Evolution and Reengineering[C]. IEEE, 2016. 90 – 101.

[8] WANG L, SUN X, WANG J, et al. Construct bug knowledge graph for bug resolution[A]. 2017 IEEE/ACM 39th International Conference on Software Engineering Companion[C]. IEEE, 2017. 189 – 191.

[9] MIKOLOV T, Sutskever I, CHEN K, et al. Distributed representations of words and phrases and their compositionality [A]. Advances in Neural Information Processing Systems [C]. Curran Associates, Inc, 2013. 3111 – 3119.

作者简介

李 斌 男,1965 年 12 月出生,江苏靖江人,扬州大学信息工程学院教授、博士,主要研究方向为智能软件工程.
E-mail: lb@yzu.edu.cn

陈定山 男,1995 年 1 月出生,江苏连云港人,扬州大学信息工程学院硕士,主要研究方向为缺陷分析.
E-mail: MX120170402@yzu.edu.cn