基于LDA的软件代码主题摘要自动生成方法

李文鹏[1,2] 赵俊峰[1,2,3] 谢 冰[1,2,3]
(北京大学信息科学技术学院 北京 100871)[1] (高可信软件技术教育部重点实验室 北京 100871)[2]
(北京大学(天津滨海)新一代信息技术研究院 天津 300450)[3]

摘 要 理解软件代码的功能是软件复用的一个重要环节。基于主题建模技术的代码理解方法能够挖掘软件代码中潜在的主题,这些主题在一定程度上代表了软件代码所实现的功能。但是使用主题建模技术所挖掘出的代码主题有着语义模糊、难以理解的弊端。潜在狄利克雷分配(Latent Dirichlet Allocation,LDA)技术是一种比较常用的主题建模技术,其在软件代码主题挖掘领域已取得了较好的结果,但同样存在上述问题。为此,需要为主题生成解释性文本描述。基于LDA的软件代码主题摘要自动生成方法除了利用主题建模技术对源代码生成主题之外,还利用文档、问答信息等包含软件系统功能描述的各类软件资源挖掘出代码主题的描述文本并提取摘要,从而能够更好地帮助开发人员理解软件的功能。

关键词 软件代码,LDA,代码功能挖掘,软件文档,摘要
中图法分类号 TP301 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.04.008

Summary Extraction Method for Code Topic Based on LDA

LI Wen-peng[1,2] ZHAO Jun-feng[1,2,3] XIE Bing[1,2,3]
(School of Electronics Engineering and Computer Science,Peking University,Beijing 100871,China)[1]
(Key Laboratory of High Confidence Software Technologies,Ministry of Education,Beijing 100871,China)[2]
(Peking University Information Technology Institute(Tianjin Binhai),Tianjin 300450,China)[3]

Abstract Understanding the function of software code is important in software reuse. Topic modeling technologies can mine the latent topics from software code,which represent the software function. But these topics lack unambiguous explanation that make them hard to be understood by the developers. Latent Dirichlet allocation (LDA) technology is one of the popular topic modeling technology. There are studies which have used LDA to mine software code and get a good result,but there are also the problems in topic description. In this paper,in addition to the use of topic modeling technology to generate topics from source code,explanatory text descriptions were generated for code topics from software resource such as documents,pairs of question and answer,mailing lists and so on. It can help users to understand the function of software code. The experiments show that the approach proposed in this paper is effective.

Keywords Software code,LDA,Code function mining,Software document,Summarization

1 引言

随着开源项目的发展,可复用软件数量日益增加。软件复用可以提高软件开发的效率和质量[1],代码复用是一种重要的产品复用方式[2]。复用者在进行代码复用之前,需要理解软件代码的功能,但是很多软件都缺乏能够描述软件功能的解释性文档。

首先,软件代码中包含了所有的软件功能,从代码中挖掘软件功能是一个有效的解决方案。近年来,基于主题建模(Topic Modeling)技术的程序理解方法和基于自动摘要(Automatic Summarization)技术的代码摘要方法取得了一定的成果。但是现有的代码功能挖掘的方法都存在一个问题,即从代码文件中挖掘出的主题信息或者摘要信息都是词的集合。对于使用者而言,由于词集表述的信息模糊且难以理解,因此需要对主题进行描述。

其次,除了软件代码之外,例如用户手册、问答信息等软件文档中也会包含着软件功能的描述信息。为了克服代码主题不易理解的弊端,本文提出一种描述代码主题的方法——代码主题摘要(Code Function Summarization,CFS)。该方法包含3个过程:

1)挖掘主题。使用LDA技术挖掘出代码中存在的主题信息。

到稿日期:2015-11-30 返修日期:2016-02-28 本文受国家自然科学基金(61472007),质检公益性行业科研专项(201510209),国家重点专项(2016YFB1000801)资助。

李文鹏 男,硕士生,主要研究领域为软件工程;赵俊峰 女,博士,副教授,主要研究领域为软件工程、知识工程、服务计算,E-mail:zhaojf@pku.edu.cn(通信作者);谢 冰 男,教授,博士生导师,主要研究领域为软件工程。

2)建立关联。使用 LDA 技术得到代码文件和文档文件的主题概率分布,根据主题概率分布建立代码主题与软件文档之间的关联关系。

3)生成摘要。使用 LexRank 对每个主题的描述文本提取摘要,获得对代码功能主题的简要文本描述。

通过上述方法,最终得到代码的功能主题和主题的描述摘要。相比于传统的代码理解方法,结合代码功能主题信息与主题描述摘要能够更好地帮助软件开发和复用人员理解软件的功能。

2 相关研究与技术

2.1 LDA 主题模型

潜在狄利克雷分布(LDA)[5]是最简单的主题模型,其基础是文章是由多个主题构成的,而每个主题都是词集的一个概率分布。

在文本生成过程中可以得到可见变量和隐含变量的联合概率分布:

$$p(\beta_{1:K},\theta_{1:D},z_{1:D},\omega_{1:D})=\prod_{i=1}^{K}p(\beta_i)\prod_{d=1}^{D}p(\theta_d)\left(\prod_{n=1}^{N}p(z_{d,n}\mid\theta_d)p(\omega_{d,n}\mid\beta_{1:K},z_{d,n})\right) \quad (1)$$

而实际上能够观察到的只有文档的词汇信息,因此 LDA 通过采样算法(或变分法)的迭代过程逼近后验概率,找到文档的主题内容:

$$p(\beta_{1:K},\theta_{1:D},z_{1:D}\mid\omega_{1:D})=\frac{p(\beta_{1:K},\theta_{1:D},z_{1:D},\omega_{1:D})}{p(\omega_{1:D})} \quad (2)$$

2.2 LDA 技术挖掘代码主题的相关研究

Kuhn 等人[3]第一次提出了挖掘代码主题的研究思路。经典通用的主题建模技术包括 PLSI[4],LDA[5]等。Xing Wei 等人[6]研究发现将 LDA 技术用于特殊数据的检索效果优于 PLSI 技术。Pierre F. Baldi 等人[7]的研究验证了使用 LDA 技术挖掘代码功能主题的有效性。Girish Maskeri 等人[8]优化了使用 LDA 技术挖掘代码主题的细节。Bing Xie 等人[9]的研究根据主题之间的紧密关系凝聚原始主题。

2.3 软件代码摘要的相关研究

Sonia Haiduc 等人[10]提出了从代码中提取摘要来支持程序理解的方法。Sonia Haiduc 等人的研究使用了文本检索(Text Retrieval,TR)技术结合单词位置信息来自动生成代码摘要。Brian P. Eddy 等人[11]延续了 Sonia Haiduc 的研究,提出了使用 hPAM 模型进行主题建模来抽取代码摘要。

2.4 LexRank 自动摘要技术

GunesErkan 和 Dragomir R. Radev 提出的 LexRank 算法[13]是一种基于图形表示的计算语句权重的方法。

3 问题分析与解决方案

3.1 问题分析

为了得到软件的功能主题与主题的描述摘要,本文从 3 个方面进行研究:挖掘主题、建立关联、生成摘要。

1)挖掘主题:使用"代码"作为数据源比使用"代码加文档"作为数据源进行主题挖掘得到的主题的混乱程度更低、噪音更小。使用软件代码进行 LDA 主题挖掘需要考虑代码的预处理,选取能够体现语义信息的代码元素作为 LDA 的输入候选集。另外,根据前人对代码注释质量的研究[16-17],不同项目的注释质量参差不齐,而且可能没有注释,因此本文不考虑注释。

2)建立关联:找到与 LDA 生成的主题相关的软件文档,能够有效地度量软件文档与主题的关联度。

3)生成摘要:过滤软件文档文本中出现的噪音信息。还需要根据文本信息的大小,选择摘要占原文的比例,使得生成的主题摘要内容简短精要。

3.2 总体解决方案

图 1 示出 CFS 方法的框架,包括 3 个部分。

1)挖掘主题:挖掘主题部分包括数据源预处理和主题建模。Java 代码中包含有语义信息的代码元素(如函数名、方法名),也有常见的不能表述代码语义信息的内容(如 Java 代码关键字和保留字),在挖掘主题时应该删去后者。而对于标示符函数名等这样人为创造的语义元素需要进行分词,得到其中包含的语义信息。进行 LDA 主题建模时,需要设置 LDA 的参数(包括 LDA 主题建模得到的主题数以及 LDA 的 alpha,beta 值等)。

2)建立关联:使用 LDA 技术对软件代码和软件文档进行主题建模,得到每个文件的主题概率分布向量,通过计算主题概率分布向量之间的余弦相似度,得到文件之间的相关度。根据软件文档与代码文件的相关度以及代码文件与主题的相关度来计算主题与软件文档的相关度,以此得到与主题相关度较高的软件文档,从而生成主题的描述。

3)生成摘要:将文本中出现的代码片段过滤掉,然后整合相关软件文档的文本内容,使用 LexRank 文本摘要技术得到主题的摘要信息。

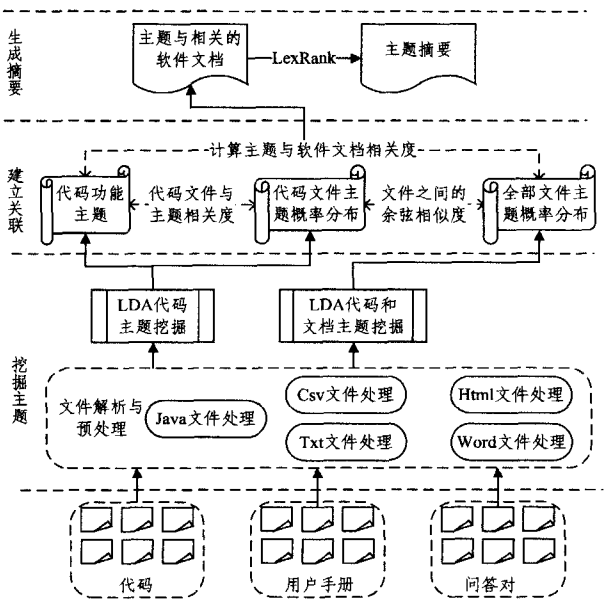

图 1 CFS 层次图

3.3 挖掘主题

挖掘主题主要包括两个步骤:数据预处理,LDA 参数的选择与代码主题建模。

3.3.1 数据预处理

图 2 示出了代码预处理的流程,图 3 示出了软件文档预处理的流程。

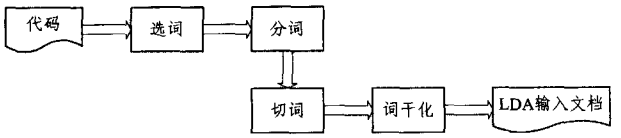

图 2　代码预处理流程

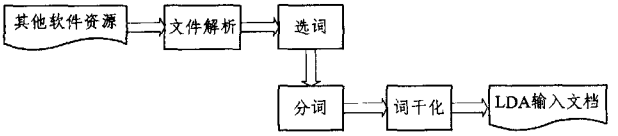

图 3　软件文档预处理流程

各部分的内容说明如下：

1)文件解析。将软件文档文件上面的文本内容保存成 txt 格式。

2)选词。选择能够作为 LDA 输入的单词，将其余的单词删去，包括 Java 保留字和关键字、长度小于 3 的单词(通常是临时变量名)以及注释内容。

3)分词。本文使用 OpenNLP 进行分词(tokenization)处理，将代码文本拆分成单词集合。

4)切词。将驼峰命名法的元素按照大写字母拆分，同时能够加强代码文件之间的关联。

5)词干化。本文使用 Snowball Stemmer 算法对单词集进行提取词干的处理。

3.3.2　代码主题建模

本文使用 Mallet(Machine Learning for Language Toolkit)来进行 LDA 主题建模。参数设置包括：α 值、β 值、迭代次数和构造的主题数目。相关研究表明[15]，当 α 初始值设为 $50/K$(K 为要构造的主题数量)，β 初始值设为 0.1 时，LDA 算法挖掘出的 Topic 的效果较好。

perplexity 是一种被广泛使用的评测主题模型质量的指标，本文使用 TMT(Stanford Topic Modeling Toolbox)计算 perplexity 来评估主题建模的好坏，以确定设置 LDA 的迭代和构造主题数。

本文对两个开源项目 Lucene 和 Tomcat 的代码进行了实验。实验中发现迭代 1000 次后 pereplexity 不再大幅变化，因此迭代次数设置为 1000。在确定主题数目时，本文参考了 Girish 等人的工作[8]，通过遍历主题数目 1 到 N，找到第一个使得 perplexity 值最小的主题数目。Lucene 实验结果如图 4 所示。因此，本文选择拐点处的主题数 50，在此主题数下，既能得到较好的主题模型，又不会产生太多的主题。Tomcat 以同样的方式选择主题数为 50。

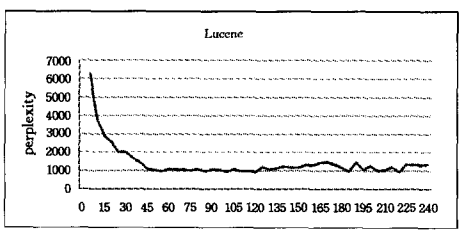

图 4　主题数对代码 LDA 主题模型的影响

3.4　建立关联

主要是将 LDA 建模得到的主题与软件文档建立关联，这一部分分成两个步骤：代码和文档主题建模，建立主题文档关联。

3.4.1　代码和文档主题建模

与 3.3.2 节对软件代码和文档(Java 代码、用户手册、问答对)进行 LDA 主题建模类似。选择迭代次数为 1000，Lucene 的 LDA 迭代的主题数为 $n_2=130$，Tomcat 的 LDA 迭代的主题数为 $n_3=170$。

3.4.2　建立主题与软件文档之间的关联

在计算主题与软件文档的相关度时需要使用 3.4.1 节和 3.3.2 节中 LDA 主题建模得到的数据。

3.4.1 节中 LDA 主题建模得到软件代码和文档中每个文件的主题概率分布 V_i^1：

$$V_i^1=(P_{topic_0^1},P_{topic_1^1},P_{topic_2^1},\cdots,P_{topic_{n_2}^1})$$

其中，$topic$ 的上标“1”表示这是对软件代码和文档进行 LDA 主题建模得到主题概率，下标“n_2”表示 Lucene 主题建模的主题总数是 n_2，$P_{topic_0^1}$ 表示文件 D_i 中包含主题 $topic_0$ 的概率。

对于任意两个文件 D_i 和 D_j，可以通过计算这两个文件的主题概率分布 V_i 和 V_j 之间的余弦相似度得到两个文件的相关程度 $similarity_{i,j}$：

$$similarity_{i,j}=\frac{V_i\cdot V_j}{\|V_i\|*\|V_i\|}$$

这样每个文件 D_i 都可以得到一个向量 S_i，以表示文件 D_i 与任意 Java 代码文件 D_j 的相似度：

$$S_i=(similarity_{i,0},similarity_{i,1},\cdots,similarity_{i,N})$$

其中，N 表示 Java 代码文件的数目。

3.3.2 节中可以得到每个代码文件的主题概率分布 V_i^0：

$$V_i^0=(P_{topic_0^0},P_{topic_1^0},P_{topic_2^0},\cdots,P_{topic_{n_0}^0})$$

其中，$topic$ 的上标“0”表示这是代码文件资源进行 LDA 主题建模得到主题概率，下标“n_0”表示 Lucene 代码文件资源主题建模的主题总数是 n_0，$P_{topic_0^0}$ 表示代码文件 D_i 中包含主题 $topic_0$ 的概率。

对于得到的每个主题 $topic_i^0$，可以通过 $V_0^0,V_1^0,\cdots,V_N^0$ 得到与该主题相关的代码文件的概率分布 R_i^0：

$$R_i^0=(P_{D_0},P_{D_1},P_{D_2},\cdots,P_{D_N})$$

通过 S_j 和 R_i^0 可以计算主题 $topic_i^0$ 与软件文档文件 D_j 的相关度 $e_{i,j}$：

$$e_{i,j}=R_i^0\cdot S_j$$

最终得到的 $e_{i,j}$ 就是判断主题与软件文档相关程度的指标，其值越大表示两者之间越相关。

3.5　生成摘要

本文使用 sumy 完成摘要工作。对于每个主题相关度最高的前 5 篇文档，过滤代码片段，只保留其文本内容，然后使用 LexRank 技术提取的 15%作为摘要。其中“5”，“15%”的选择是根据软件文档切分粒度选择的较为合适的数值。

4　实验分析

4.1　分析主题相关软件文档的内聚度

为了验证 CFS 工具产生的主题与软件文档的关联效果，本文对 Lucene 项目进行了主题摘要的提取，选择了《Lucene in Action》[14]的章节片段作为软件文档，然后对其进行了

CFS方法的处理，得到了主题以及主题相关的文本片段。观察每个主题相关度最高的前5个文本片段，如果这5个片段都属于《Lucene in Action》目录下面的同一个章节，就认为这个主题的相关资源的内聚度最高，得分为5分；如果没有同一章节的文本片段，就认为这个主题的相关资源的内聚度很差，得分为1；以此类推。

实验结果见表1。实验结果表明，建立关联模块算法能够得到较好的内聚度水平，验证了该算法的有效性。

表1 内聚度分析表

项目名称 \ 主题数	5分	4分	3分	2分	1分	总计	平均分
Lucene	1	11	24	12	2	50	2.94
Lucene 含注释	2	28	64	35	1	130	2.96

4.2 分析主题摘要的描述效果

为了评测主题摘要的描述效果，本文对Lucene和Tomcat两个项目进行主题摘要的提取，然后通过Web网页进行展示，邀请了5名对Lucene和Tomcat项目比较了解并具备3年以上编程经验的人员来对该描述效果进行评分。评分结果如表2所列(满分为5分)。

表2 主题摘要效果分析表

项目名称 \ 主题数	5分	4分	3分	2分	1分	0分	总计	平均分
Lucene	2	14	20	7	6	1	50	2.92
Tomcat	1	11	22	9	7	0	50	2.80

从实验结果来看，最后得到的主题摘要的描述效果平均为3分，表示CFS得到的主题摘要能够较好地帮助使用者理解代码主题的功能。

结束语 本文提出了一个代码主题摘要自动生成方法，其能够从软件代码中提取功能主题，并从软件文档中提取对代码主题的描述摘要。包括3个部分：

1)挖掘主题。使用LDA主题模型对代码进行主题挖掘，从中得到能够描述代码功能的主题，包括数据预处理的方式以及主题参数的选择等。

2)建立关联。本文提出了一种基于LDA的主题概率分布计算代码主题与软件文档的相关度的方法。

3)生成摘要。过滤主题相关文档的内容，并使用LexRank算法对主题相关文档的文本内容提取摘要，以生成简短的主题摘要。

未来的工作可以从3个方面进行：1)研究改进主题建模方法；2)使用traceability等技术加强建立关联；3)评测软件文档的质量重排名。

参 考 文 献

[1] YANG F Q, MEI H, LI K Q. Software Reuse and Software Component Technology[J]. Chinese Journal of Electronics, 1999, 27(2): 68-75. (in Chinese)
杨芙清，梅宏，李克勤. 软件复用与软件构件技术[J]. 电子学报，1999，27(2)：68-75.

[2] ABRAN A, MOORE J, BOURQUE P, et al. Guide to the software engineering body of knowledge[M]// SWEBOK. IEEE Computer Society, 2004.

[3] KUHN A, DUCASSE S, GLRBA. Semantic clustering: Identifying topics in source code[J]. Information and Software Technology, 2007, 49(3): 230-243

[4] HOFMANN, THOMAS. Probabilistic Latent Semantic Indexing [C]// Proceedings of the Twenty-Second Annual International SIGIR Conference on Research and Development in Information Retrieval. 1999: 50-57.

[5] BIEID, NG A, JORDAN M. Latent dirichlet allocation[J]. The Journal of Machine Learning Research, 2003, 3: 993-1022.

[6] WEI X, Croft W B. LDA-based document models for ad-hoc retrieval[C]// Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval(SIGIR'06). 2006: 178-185.

[7] BALDI P F, LOPES C V, LINSTEAD E J, et al. A Theory of Aspects as Latent Topics[C]// Proceedings of the 23rd ACM SIGPLAN Conference on Object-oriented Programming Systems Languages and Applications(OOPSLA'08). 2008: 543-562.

[8] MASKERI G, SARKAR S, HEAFIELD K. Mining Business Topics in Source Code using Latent Dirichlet Allocation[C]// Proceedings of the 1st India software engineering conference (ISEC'08). 2008: 113-120

[9] XIE B, LI M, JIN J, et al. Mining Cohesive Domain Topics from Source Code[M]// Safe and Secure Software Reuse: ICSR 2013. LNCS 7925, 2013: 239-254.

[10] HAIDUC S, APONTE J, MORENO L, et al. On the Use of Automated Text Summarization Techniques for Summarizing Source Code[C]// 2010 17th Working Conference on Reverse Engineering (WCRE). IEEE, 2010: 35-44.

[11] EDDY B P, ROBINSON J A, KRAFT N A, et al. Evaluating source code summarization techniques: Replication and expansion[C]// 2013 IEEE 21st International Conference on Program Comprehension (ICPC). IEEE, 2013: 13-22.

[12] CHANG J, BLEIl D M. Hierarchical relational models for document networks[J]. The Annals of Applied Statistics, 2010, 4 (1): 124-150.

[13] ERKAN G, RADEV D R. LexRank: graph-based lexical centrality as salience in text summarization[J]. Journal of Artificial Intelligence Research, 2011, 22(1): 457-479.

[14] MCCANDLESS M, HATCHER E, GOSPODNETIC O. Lucene in Action(Second Edition)[M]. The United States of America: Manning Publications Co., 2010: 532.

[15] GRIFFITHS T L, STEYVERS M. Finding scientific topics[J]. PNAS, 2004, 101: 5228-5235.

[16] ARAFAT O, RIEHLE D. The comment density of open source software code[C]// 31st International Conference on Software Engineering-Companion (ICSE-Companion 2009). IEEE, 2009: 195-198.

[17] FLURI B, WÜRSCH M, GALL H C. Do code and comments co-evolve? on the relation between source code and comment changes[C]// 14th Working Conference on Reverse Engineering, 2007(WCRE 2007). IEEE, 2007: 70-79.