ISSN 1673-9418 CODEN JKYTA8
Journal of Frontiers of Computer Science and Technology
1673-9418/2000/00(00)-0000-00
doi: 10.3778/j.issn.1673-9418.

E-mail: fcst@vip.163.com
http://www.ceaj.org
Tel: +86-10-89056056

基于缺陷报告的缺陷定位与代码变动预测方法*

席圣渠 [1,2+], 张晓飞 [1,2], 徐圣斌 [1,2], 徐 锋 [1,2]

1. 南京大学 计算机科学与技术系, 江苏 南京 210023
2. 南京大学 计算机软件新技术国家重点实验室, 江苏 南京 210023

Bug Report Based Bug Localization and Code Change Prediction*

XI Sheng-Qu[1,2+], ZHANG Xiao-Fei[1,2], XU Sheng-Bin[1,2], XU Feng[1,2]

1. Department of Computer Science and Technology, Nanjing University, Nanjing 210023, China
2. State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
+ Corresponding author: Phn: +86-152-0518-6100, E-mail: nju.cellzero@gmail.com

XI Sheng-Qu, ZHANG Xiao-Fei, XU Sheng-Bin, XU Feng.Bug report based bug localization and code change prediction. Journal of Frontiers of Computer Science and Technology, 2000, 0(0): 1-000.

Abstract: As software systems are often shipped with bugs, bug fix is an important part of software maintenance. However, bug localization which means to find the cause of a bug, is not an easy task. Besides, estimating the workload of fixing a bug is also difficult but essential, inappropriate estimation will lead to improper triage and delay the bug fix process. Recently, several information retrieval methods have been proposed to help developers locate related buggy source files for a given bug report. The basic assumption of these methods is that the bug description in a bug report should be textually similar to its buggy source files. However, code changes information (such as number of code changes) were ignored by these methods. In this paper, we first investigate code changes information in bug localization, then provided a supervised generative model which utilizes both the bug report and the related code changes. Experiment on three real datasets shows our bug localization approach outperforms rVSM by 7%~66%.

*The National Natural Science**** Foundation of China under Grant No.****61702252, ****61672274-, 61690204(基金中文完整名称国家自然科学基金); the **** Foundation of China under Grant No.****, **** (基金中文完整名称). [需中英文齐全].
Received 2000-00, Accepted 2000-00.

Key words: bug localization; bug report; code change level; supervised topic model

摘 要: 缺陷修复是软件维护的重要环节，然而缺陷定位和缺陷分派是缺陷修复过程中较为关键和困难的任务。一方面，从缺陷报告定位到源代码，需要开发人员具有相关知识和经验；另一方面，估算缺陷修复工作量也较为困难，错误的估计会导致不当的分派，从而延迟缺陷修复流程。当前，大部分研究工作基于信息检索方法进行缺陷定位，其基本假设是缺陷报告与关联源代码在文本上更为相似，但此类方法忽略了代码变动等信息的利用。通过研究代码变动信息在缺陷修复中的作用，提出了一种监督式的主题生成模型，建模了缺陷报告文本与关联代码的变动信息。在三个开源项目数据集上的实验结果表明，该方法在缺陷定位准确率上较 rVSM 方法有 7%~66%的提升。

关键词: 缺陷定位；缺陷报告；代码变动程度；监督主题模型

文献标志码：A **中图分类号：******

1 引言

随着软件规模的不断扩大，其复杂性也不断增加。虽然存在诸如软件审查、静态检测等软件质量保障机制，但在开发资源和开发周期的制约下，总会存在运行时的异常行为或者产生不合预期的结果。这样的情况被称为软件缺陷。

为了保证软件的质量，开发者通常会消耗大量的时间调试和修复缺陷[1]。这些缺陷通常来自项目的缺陷追踪系统，由用户或开发者在发现软件异常行为时，以缺陷报告的形式提交。缺陷报告软件的异常记录、引发缺陷的操作等有助于修复缺陷的信息。当开发者被指派修复某一缺陷报告，往往需要应用领域知识，或是复现缺陷场景[2]，或是了解程序状态，以定位到缺陷相关的代码片段，进而执行代码的修复工作。这一由缺陷信息定位到可疑代码片段的过程，被称为缺陷定位。然而缺陷定位较为困难，一方面，缺陷报告的质量参差不齐，且缺陷描述同引发代码存在理解差异；另一方面，软件项目往往存在大量缺陷报告（例如，Mozilla 收到超过 420000 份缺陷报告[3]），人工定位缺陷会花费大量的时间。研究表明，若能为开发者提供相关代码的排序列表，便可以减少定位到缺陷代码的时间[4]。

此外，缺陷报告需要分派给特定的开发者修复，不恰当的分派会导致重新分派给其他开发者修复（有 37%到 44%的缺陷报告需要重新分派[5]），进而拖慢缺陷修复进程。若能通过缺陷报告估算修复工作量，既能指导缺陷分派任务，又能辅助缺陷代码的定位，从而整体上提高缺陷的修复效率。

在近些年的研究中，出现了一系列基于信息检索（IR，information retrieval）的缺陷定位方法。其基本思想在于，将源代码文件集合视为文档，将缺陷报告视为查询，通过计算两者相似度，返回经排序的相关源代码文件列表。Lukins 等人[6]使用主题模型 LDA（Latent Dirichlet Allocation）[16]学习缺陷报告同源文件的隐式主题分布，并基于主题计算相似度。Zhou 等人[7]使用空间向量模型（VSM，vector space model），在缺陷报告同源代码的基础上进一步计算了缺陷报告之间的相似性，以提高缺陷定位效果。Lam 等人[8]和 Huo 等人[9]使用深度学习的方法提取缺陷报告和源代码的高层特征，并基于高层特征进行相似度比较。

综上，基于信息检索的方法，普遍只获取文本特征，再做相似性比较。然而，这种方案存在两个问题。首先，源代码格式要求更加严格，其变量命名形式、保留字的特殊含义，以及语法规范都区别于自然语言文本，不能简单的同等看待。其次，代码与文本的相似性不能等同于相关性，缺陷报告相关的代码也可能同其文本相似度较低。另外，基于信息检索的方法，对新缺陷报告，往往需要同已有源代码文件逐一比较，计算量较大，效率较低。

并且，当前的缺陷定位方法，都没有考虑代码的变动情况。这种变动，可以是代码的变动内容，也可以是代码的变动数量。这些真实修复时的代码信息，是同真实缺陷最为相关的信息，也是同缺陷报告关联更加紧密的信息。

综合上述两方面的考虑，本文基于 LLDA[18]提出了一种监督主题模型 sLLDA，建模了缺陷报告文本信息和关联代码变动信息，利用历史修复记录训练模型，进而预测新的缺陷报告对应的缺陷代码和代码变动情况。历史修复记录，包括缺陷报告的描述文本、为修复缺陷所变动的代码总行数，以及作为变动代码的文件名等。其中代码变动行数和代码文件名作为观察信息，用于模型的监督学习。在测试阶段，输入缺陷报告的描述文本，模型将预测相关的源代码排序，以及代码变动的行数。前者能帮助开发者找到引发缺陷的原因，后者能够辅助项目管理者分派缺陷报告。

为检验方法的效果，本文在 3 个开源项目数据集上进行了实验。并选取了 rVSM 方法作为基准方法进行对比。在预测命中率指标上，本文方法较基准方法有显著提升。

2 相关工作

基于缺陷报告的缺陷定位，目标在于输入缺陷描述信息，输出引发缺陷的位置。缺陷的位置有多种描述力度，如代码块级、方法级、源文件级等。基于动态收集程序运行信息的方法，往往定位到较细的力度，如代码行[11][12]，代码块等等。然而该类方法代价高昂，一方面需要编译运行程序，另一方面动态信息极为庞大；且其结果较为依赖测试用例的质量。因而，本文主要关注基于缺陷报告的定位，这种定位方法一般定位到源文件级别。

在基于缺陷报告的缺陷定位研究中，信息检索是得到广泛使用的一类方法。这些方法以相似度计算为核心，采用不同的方式提取缺陷报告和源代码文件的特征，并通过特征间的逐一比较，返回同缺陷报告相似度由高到低排序的源代码列表。不同方法的主要区别在于不同的特征提取方式，包括基于空间模型的方法，基于主题模型的方法，基于深度学习的方法。

在基于空间向量模型的方法中，主要计算文本相似度的方式。Zhou 等人[7]提出 rVSM，使用缺陷报告间的相似性来矫正 VSM 结果。Saha 等人[13]进一步考虑了代码的结构信息，如变量名和方法名等。Ye 等人[15]在使用 API 描述和自动生成的文本作为特征的基础上，考虑了 6 种结合了领域知识的特征，并使用学习排序（Learning-to-rank）的方式学习特征的权重。在其后续工作中，收集了更多种类的特征，使用 9 种特征并取得了更好的结果。

基于主题模型的方法，主要基于 LSI[19]和 LDA[16]方法，在语义层面比较相似性。Lukins 等人[6]以源代码文件训练 LDA 模型，并通过模型获取新缺陷报告和源代码文件的主题分布，利用主题分布进行相似性比较。Nguyen 等人[20]认为经常出错的文件和大文件更容易出错，并以此作为领域知识修改 LDA 模型。

最近，提出了一些深度学习方法。Lam 等人[8]在缺陷报告和源代码文件上使用深度神经网络学习特征表示。Huo 等人[9]使用卷积神经网络，进一步提取了代码的结构信息。

上述的方法，一方面采用了信息检索的方法，需要逐一比较缺陷报告同源代码文件的相似性，计算开销较大，且同缺陷报告特征相似的源代码不一定是目标位置，即相似性并不一定等于相关性，故方法在缺陷定位任务上效果较差；另一方面，这些方法都没有考虑代码变动信息，如代码变动行数、代码变动内容等，这些信息能更好的反映代码的变更，且修复某一缺陷报告的代码变更，能更好的描述缺陷产生的根本原因。

因此，本文提出了一种新的结合了代码变动信息的监督方法。为刻画相关性，本文采用监督方法学习历史数据中缺陷报告同源代码的关联，Kim 等人[21]同样采用了监督方法，其将问题转化为分类问题，并使用朴素贝叶斯分类器预测缺陷报告最可能

的源代码。而本文认为，基于语义的主题模型能够对文档进行更好的建模，故基于监督主题模型 LLDA[18]进行修改。在描述代码变动信息上，Rao 等人[14]结合了代码变动信息，认为近期修改过的源代码元素更可能是引发缺陷，并将其作为候选。而本文更关注修复某一缺陷报告的代码变动信息，并首先添加了代码变动行数，以检验其对缺陷定位任务的影响，未来工作则会考虑对代码变动内容信息的建模。

3 预备知识

3.1 缺陷报告

缺陷追踪系统中，用户或开发者可以通过填写描述缺陷相关信息的自然文本提交缺陷报告，而后开发者会通过缺陷报告修复缺陷。一份简化的缺陷报告如图 1 上半部分所示。最上方的灰色框中的文字是缺陷报告的标题，是对缺陷的简单描述。白色区域的部分被称为描述（description），是更加详细的关于缺陷如何引发、缺陷发生时的运行效果等有助于开发者修复缺陷的自然语言文本。这两部分内容，是缺陷报告最为重要的文本信息，故本工作使用缺陷报告标题和描述作为方法的文本输入。

3.2 代码变动

开发者通过提交代码到版本控制程序，以修复缺陷。通过版本控制程序的提交描述，可以找到对应修复的缺陷编号，并且能够获取为修复缺陷变动的源代码名称以及位置。如图 1 红色框所示，通过检索版本控制系统的提交记录，能够找到同缺陷报告编号相同的提交文本记录。且如蓝色方框所示，还能够得知两个版本间代码文件变动的数量，以及增添删除的代码行数。本文主要考察代码变动行数对缺陷定位方法的影响。

4 基于缺陷报告的缺陷定位与代码变动预测方法

4.1 问题定义

本文采用监督的方式，解决以缺陷报告定位到

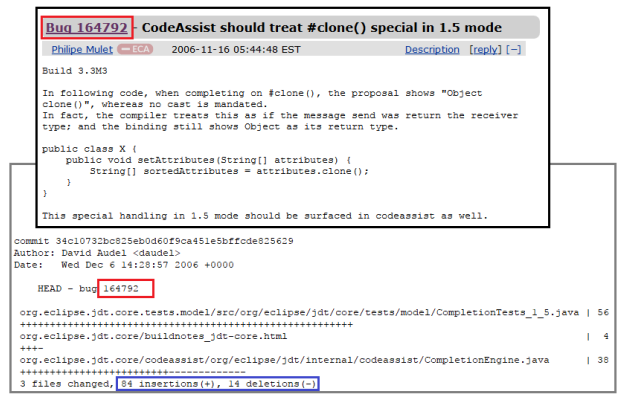

Fig.1 Bug Report and Code Change Log

图 1 缺陷报告与代码变化记录

相关源代码的任务。具体来说，方法的核心为一个监督主题模型 sLLDA，模型需要从历史缺陷报告和其源代码的对应关系进行学习，进而对新缺陷报告做出预测。

下定义本文使用的变量。本文以 D 代表缺陷报告集合，以 V 代表词汇表的集合，以 S 代表源代码的集合。对于每份缺陷报告，以 d 表示其文本信息、w 表示文本中的单词，N_d表示其单词数量。修复该

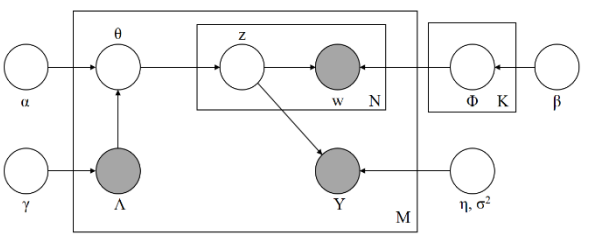

Fig.2 Structure of sLLDA

图 2 sLLDA 概率图

缺陷的源代码以Λ_d表示，其元素皆属于集合 S 且数量可以为一份或多份。代码修改的总行数以 Y 表示。

故本文问题可以描述为：给定历史缺陷报告集合 D，其中每份缺陷报告包含N_d个单词的文本 d，并已知修复该缺陷的源代码文件Λ_d和代码变动程度 Y。对于一份新缺陷报告，如何预测其相关的源代码文件，以及可能的代码修改行数。

4.2 缺陷报告与代码变动建模

本文使用信息为缺陷报告的文本信息，代码变动的行数信息。

缺陷报告的文本，包括其标题和描述，经过文本预处理，这些信息被拆分为单词，并以词袋（bag of words）的方式存储。

代码变动的行数信息，首先需要搜索版本控制程序，找到缺陷报告对应的提交；其次对修改的文件，比较其前后两个版本的差异，统计该提交变动的代码总行数；最后，通过统计将其划分为高、中、低三种类型，代表了本次修复的代码变动程度。

4.3 图模型与生成过程

如图 2 所示，为本文提出的监督主题模型 sLLDA 的概率图。图中灰色的圆形代表观察值，其中Λ代表了修复缺陷的源代码，Y 代表了修复缺陷的代码变动程度，w 代表了文档中的单词。M 代表了缺陷报告文档的总数目，对于每份缺陷报告，其主题受到α和Λ共同影响，在训练阶段，由于Λ作为观察值已知，故限定文档的主题只能在Λ_{d}中产生。主题的总数目 K 同源代码的总数量相等。N 表示文档长度，即单词的数目，对于特定的文档 d，其文档长度为$\mathrm{N_d}$。根据文档的主题分布θ，为文档的每一个单词采样一个主题 z，不同的主题下单词的分布ϕ_{z}不同，故采样出单词 w，并且主题对代码变动程度有所影响，集合正态分布采样出代码变动程度 Y。

模型 sLLDA 的生成过程如下所示：

1）采样主题下单词的分布
   a) 对每一个主题$\mathrm{k} \in \mathrm{K}$，
      i. 采样各主题 K 上的词分布参数ϕ_{k}，$\phi_{\mathrm{k}} \sim Dir(\beta)$
2）采样文档$\mathrm{d} \in \mathrm{D}$的标签
   a) 对每一个主题$\mathrm{k} \in \mathrm{K}$，
      i. 采样文档 d 中主题 k 是否出现的概率$\Lambda_{\mathrm{d,k}}$，$\Lambda_{\mathrm{d,k}} \sim Bernoulli(\gamma)$
3）采样文档 d 的主题分布参数θ_{d}，$\theta_{\mathrm{d}} \sim Dir(\alpha)$
   a) 对每一个单词$\mathrm{i} \in [1, \mathrm{N_d}]$，
      i. 采样主题$\mathrm{z_i}$，$\mathrm{z_i} \sim Multi(\theta_d)$
      ii. 根据主题$\mathrm{z_i}$，采样单词$\mathrm{w_{i,\phi}}$，$\mathrm{w_{i,\phi}} \sim Multi(\phi_{z_i})$
4）采样响应变量Y
   a) 对每一个文档$\mathrm{d} \in \mathrm{D}$
      i. 采样$\mathrm{Y_d}$，$\mathrm{Y_d} \sim Norm(\eta^T \bar{z}, \sigma^2)$

其中，$Dir(\cdot)$表示狄利克雷分布；$Bernoulli(\cdot)$表示伯努利分布；$Multi(\cdot)$表示多项分布。$Norm(\cdot,\cdot)$代表正态分布，其参数分别表示其均值和方差。

4.4 训练与预测

模型 sLLDA 分为训练和预测两个阶段。训练阶段，三种观察值均已知，可以利用源代码文件名和代码变动程度作为监督信息调整模型参数；测试阶段，代码变动程度和可疑源文件未知，故利用缺陷报告的文本信息，使用概率方式预测其主题分布，进而预测源代码文件和代码变动程度。以下分别对两个阶段详细说明。

在训练阶段，一方面要不断更新单词的采样，进而更新不同主题下单词的分布ϕ_{k}；另一方面，在更新全部文档后，需要更新正态分布的参数η和σ^2。

对于文档 d 中的第 i 个单词，主题z_i的更新公式如下：

$$
\begin{aligned}
&P(z_i = j | z_{-i}, j \in \Lambda_d) \\
&\propto \frac{n_{-i,j}^{(d)} + \alpha_j}{\sum_{k \in \Lambda_d} \left(n_{-i,k}^{(d)} + \alpha_k \right)} \frac{n_{-i,j}^{w_i} + \beta_{w_i}}{\sum_{v \in V} \left(n_{-i,j}^{v} + \beta_v \right)} \\
&\times \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(Y - \eta^T \bar{Z})^2}{2\sigma^2} \right)
\end{aligned}
\tag{1}
$$

其中，$n_{-i,j}^{(d)}$表示除去第 i 个单词后，文档 d 中属于主题 j 的单词的计数；$n_{-i,j}^{w_i}$表示除去文档 d 中的第 i 个单词后，属于主题 j 的单词w_i的计数。需要注意，单词新主题只会落在Λ_d之中，即采样只需要考虑$j \in \Lambda_d$的主题即可。

公式第一部分代表了文档 d 在主题 j 上分布，即$\theta_{d,j}$；第二部分代表了单词主题 j 上单词w_i的分布，即Φ_{j,w_i}；第三部分代表了代码变动程度的影响，由其服从正态分布，故以其概率相乘。

正态分布的参数更新，在每次更新全部文档的主题后进行。令 A 为$(\mathrm{D} \times \mathrm{K})$的矩阵，表示所有文档的主题分布；令 Y 为$(\mathrm{D} \times 1)$的矩阵，表示文档的代

码变动观测值。采用最大似然估计，可以得出正态分布参数的更新公式：

$$\hat{\eta} = (A^T A)^{-1} A^T Y \tag{2}$$

$$\widehat{\sigma^2} = \frac{1}{D}(Y - A(A^T A)^{-1} A^T Y)^T (Y - A(A^T A)^{-1} A^T Y) \tag{3}$$

模型 sLLDA 可以采用多种方式进行训练，如最大似然估计、吉布斯采样等方法。本文训练采用 CVB0 的方法[22],因其收敛更为迅速且相较于另外两种方法结果更为稳定[23]。CVB0 的更新规则可有公式(1)推导，不复累述。

在预测阶段，模型需要完成两项任务，第一预测同缺陷报告相关的源代码文件，第二预测修复缺陷所需的代码变动程度，以下详细阐述预测方式。

给定新的缺陷报告d_{new}，假定其可属于任何主题，则预测可疑源代码文件可转化为计算概率$p(z|d_{new})$，其中概率更大的主题具有更高的可疑程度，具体预测公式如下：

$$\mathrm{p(z|d_{new})} = \sum_{w \in d_{new}} \frac{p(w|z)p(z)}{p(w)} p(w|d_{new}) \tag{4}$$

公式(4)中，$p(w|z)$表示主题 z 中单词 w 出现的概率，其中 w 皆为新文档d_{new}中出现的单词，其值可由训练好的模型参数$\Phi_{z,w}$近似替代。

$p(w|d_{new})$表示新文档d_{new}中单词 w 出现的概率，可以单词 w 占文档总单词数的比例近似替代，对应公式如下：

$$\mathrm{p(w|d_{new})} = \frac{count(w)}{N_{d_{new}}} \tag{5}$$

其中，$count(w)$表示新文档$\mathrm{d_{new}}$中单词 w 的数量，$N_{d_{new}}$表示新文档$\mathrm{d_{new}}$包含的单词总数。

对于p(z)，可以利用公式(6)进行近似估计：

$$\mathrm{p(z)} = \sum_{d \in D} p(z|d)p(d) \tag{6}$$

其中，$p(z|d)$表示训练文档的主题分布，具体而言，对于特定的文档 d 和主题 z，其概率可由训练好的模型参数$\theta_{d,z}$近似替代。对于$p(d)$，本文假设每篇文档出现机会是均等的，故等式的整体含义为计算训练文档中主题的均值。

每个单词 w 都是从某个特定主题 z 中生成的，故p(w)可用公式(7)计算：

$$\mathrm{p(w)} = \sum_z p(w|z)p(z) \tag{7}$$

其中，$p(w|z)$可由$\Phi_{z,w}$近似替代，$p(z)$可以利用公式(6)进行计算。

其次，由已计算概率$p(z|d_{new})$，可以计算新缺陷报告修复所需的代码变动程度，其值为当前主题分布的情况下，正态分布的均值，公式如下：

$$Y_{new} = \eta^T p(z|d_{new}) \tag{8}$$

其中，η为模型参数，其值在训练过程中确定。由训练过程描述，$\eta^T \bar{z}$代表了正态分布的均值，对于新文档d_{new}，$\bar{z}$由概率$p(z|d_{new})$近似替代。

5 实验

5.1 数据处理

在实验中，本文选取了 Eclipse 中的三个有代表性的开源项目：JDT、Platform、PDE 进行实验。表 1 表述了这三个项目的统计信息。

Table1 The statistics of projects

表 1 sLLDA 的命中率指标结果

项目	源文件数	报告数
JDT	3440	4930
Platform	4506	3707
PDE	2956	3286

所有的缺陷报告都是从 Eclipse 网站的 BugZilla 系统中获取，本文从所有缺陷报告中筛选出状态为 Fix 的缺陷报告作为实验的数据集。缺陷报告所对应的待修复文件则是从项目的 git 历史中获得。同时也可以从 git commit 记录中获取每个缺陷对应改动行数，并以此将代码变动程度划分为三类：当改动行数大于 50 时，认为代码变动程度高；代码变动小于 10 时，认为代码变动程度低；代码变动行数在 10 到 50 之间时，认为代码变动程度为中。

对缺陷报告文本部分，本文只使用原始的报告文本，而没有考虑报告后续的讨论，以还原真实的使用场景。在文本处理上，首先把文本处理成单词

序列，再将所有单词进行词干化、去停词处理，最后删去词频小于 10 的词语。

5.2 评估方法

为了衡量 sLLDA 模型的进行缺陷定位的效果以及与其他方法进行比较，本文引入 Hit@k 指标进行度量。Hit@k 表示取排序前 k 个文件就能命中需要修改的源文件的缺陷所占的比例。其计算公式对应如下：

$$Hit@k = \frac{N_{hit}}{N_{all}} \tag{9}$$

其中N_{hit}表示当选取排序前 k 个源文件时能命中缺陷所在文件的缺陷数目，N_{all}表示所有缺陷的数目。

同时本文直接利用代码变动程度的准确率来表示我们预测代码变动的准确程度。

5.3 参数设置

sLLDA 包含两个超参数 α 和 β，在实验中取$\alpha = 50.0$，$\beta = 0.01$。

在对比模型 rVSM 中包含一个参数 alpha，本文选取效果最好的参数设置即$\text{alpha} = 0.2$。

5.4 实验结果

本文在 JDT、Platform、PDE 这三个数据集上对 sLLDA 模型和现有方法 rVSM 都选取十折交叉验证的方法进行了对比实验。对比实验结果如图 3 至图 5 和表 2、表 3 所示。图中横坐标代表排序前 k 个文件的 k 的取值。

Table2 The hit result of sLLDA
表 2 sLLDA 的命中率指标结果

项目	Top1	Top10	Top20
JDT	0.260	0.652	0.752
Platform	0.225	0.594	0.687
PDE	0.163	0.555	0.657

Table3 The hit result of rVSM
表 3 rVSM 的命中率指标结果

项目	Top1	Top10	Top20
JDT	0.087	0.333	0.453
Platform	0.232	0.534	0.642
PDE	0.134	0.459	0.581

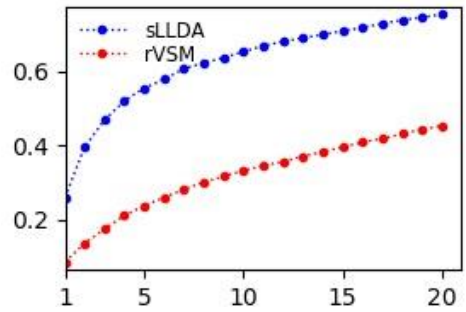

Fig.3 The comparison of hit result of JDT between sLLDA and rVSM
图 3 sLLDA 和 rVSM 在 JDT 数据集命中率比较

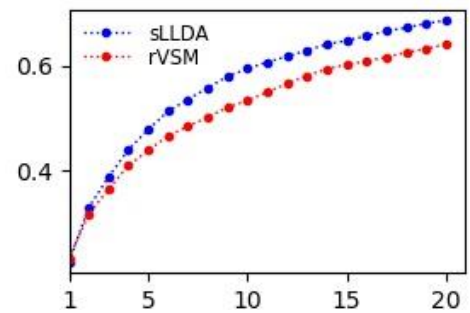

Fig.4 The comparison of hit result of Platform between sLLDA and rVSM
图 4 sLLDA 和 rVSM 在 Platform 数据集命中率比较

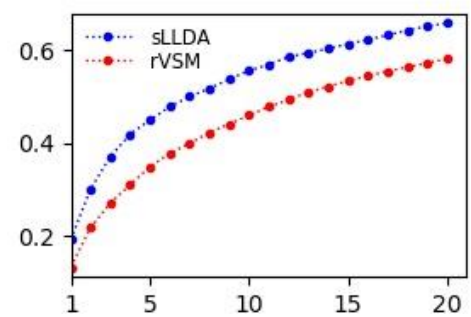

Fig.5 The comparison of hit result of PDE between sLLDA and rVSM
图 5 sLLDA 和 rVSM 在 PDE 数据集命中率比较

结果显示 sLLDA 的预测效果强于 rVSM 模型。

总体而言，sLLDA 预测能力更强的原因在于，该方法在文本的处理上采用了主题模型，考虑了单词的共现关系，能够表示文本中单词相同但含义不同的情况。另一方面，sLLDA 考虑了代码的变动程度作为观察信息，加入模型的训练，从侧面建模了

源代码文件的修复难度，有助于新缺陷报告的定位。

另一方面，sLLDA 在 JDT 数据集上表现远超 rVSM 方法。经过观察，发现 JDT 数据中出错的源代码较为集中，即曾经出错的源代码文件，其后出错的可能性依旧很高。而本文的方法采用监督的方式，更多的关注曾经出现过错误的源代码文件，rVSM 方法则平等的考虑所有源代码文件，其候选集合过大，因而效果较差。

同时，本文基于 sLLDA 模型对代码变动程度也进行了相应的预测，预测结果如表 4 所示：

Table4 The accuracy result of the extent of the change

表 4 代码变动程度预测准确率结果

项目	准确率
JDT	0.447
Platform	0.400
PDE	0.414

由预测结果可知,在给出缺陷所在文件的同时，sLLDA 同时也可以为预测代码变动的程度，为项目管理者和程序开发人员了解修复任务的难易程度提供辅助信息。

本文方法能够进行代码修改规模的预测，其原因在于训练过程中，将代码变动规模作为观察信息，辅助主题模型生成更好的词汇对主题的概率分布，并且主题中蕴含了代码变动规模的强弱信息。在测试阶段，根据单词预测主题分布时，也隐式的揭示了代码变动的程度。

5.5 有效性分析

本文方法在实验结果上优于 rVSM，但仍然存在一些局限。首先，sLLDA 是监督方法，在训练过程中，以曾经出错过的源代码文件作为标签（类别），因而预测阶段无法定位到从未出错过的源代码文件。其次，本文提及的代码变动程度预测，是缺陷定位过程中的一个附属产品，在实验结果上有一定的效果，但其准确率并不是很高，如何更准确的预测代码变动程度，还需要进一步的研究。

6 结束语

本文提出了一种新的监督主题模型 sLLDA，在 LLDA 的基础上，添加了代码变动信息，通过将代码变动行数划分为高、中、低三种变动程度，以可观察随机变量的形式加入到概率图模型中。一方面，代码变动程度作为监督信息能使模型在训练时得到更符合缺陷定位任务的参数，进而提高了预测准确率；另一方面，可以为新缺陷报告预测修复所需的代码变动程度，为评估修复的难易程度提供了一个参考指标，实验结果显示其效果优于随机猜测。

目前，仅考虑基于代码变动行数来度量缺陷报告修复的难易程度，度量结果较为粗糙，今后的工作中，可尝试基于缺陷修复所花费时间、代码的语义变动信息等获得更精确的修复难易程度度量，以获得更好的定位准确率和缺陷修复工作量预测精度。

References:

[1] Nguyen A T, Nguyen T T, Al-Kofahi J, et al. A topic-based approach for narrowing the search space of buggy files from a bug report[C]// Ieee/acm International Conference on Automated Software Engineering. IEEE, 2011:263-272.

[2] Latoza T D, Myers B A. Hard-to-answer questions about code[C]// Evaluation and Usability of Programming Languages and Tools. ACM, 2010:1-6.

[3] Bettenburg N, Premraj R, Zimmermann T. Duplicate bug reports considered harmful … really?[C]// IEEE International Conference on Software Maintenance. IEEE, 2009:337-345.

[4] Fischer M, Pinzger M, Gall H. Analyzing and Relating Bug Report Data for Feature Tracking[C]// Reverse Engineering, 2003. Wcre 2003. Proceedings. Working Conference on. IEEE, 2003:90-99.

[5] Jeong G, Kim S, Zimmermann T. Improving bug triage with bug tossing graphs[C]// The, Joint Meeting of the European Software Engineering Conference and the ACM Sigsoft Symposium on the Foundations of Software Engineering. ACM, 2009:111-120.

[6] Lukins S K, Kraft N A, Etzkorn L H. Source Code Retrieval for Bug Localization Using Latent Dirichlet Allocation[C]// Working Conference on Reverse Engineering. IEEE Computer Society, 2008:155-164.

[7] Zhou J, Zhang H, Lo D. Where should the bugs be fixed? More accurate information retrieval-based bug localization based on bug reports[J]. 2012, 8543(1):14-24.

[8] Huo X, Li M, Zhou Z H. Learning unified features from natural and programming languages for locating buggy

source code[C]// International Joint Conference on Artificial Intelligence. AAAI Press, 2016:1606-1612.
[9] Ye X, Bunescu R, Liu C. Mapping Bug Reports to Relevant Files: A Ranking Model, a Fine-grained Benchmark, and Feature Evaluation[J]. IEEE Transactions on Software Engineering, 2016, 42(4):379-402.
[10] Ramage D, Hall D, Nallapati R, et al. Labeled LDA: a supervised topic model for credit attribution in multi-labeled corpora[C]// Conference on Empirical Methods in Natural Language Processing: Volume. Association for Computational Linguistics, 2009:248-256.
[11] Jones J A, Harrold M J, Stasko J. Visualization of Test Information to Assist Fault Localization[C]// International Conference on Software Engineering. IEEE, 2002:467-477.
[12] Jones J A, Harrold M J. Empirical evaluation of the tarantula automatic fault-localization technique[C] //Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering. ACM, 2005: 273-282.
[13] Saha R K, Lease M, Khurshid S, et al. Improving bug localization using structured information retrieval[C]//Automated Software Engineering (ASE), 2013 IEEE/ACM 28th International Conference on. IEEE, 2013: 345-355.
[14] Rao S, Kak A. Retrieval from software libraries for bug localization: a comparative study of generic and composite text models[C]//Proceedings of the 8th Working Conference on Mining Software Repositories. ACM, 2011: 43-52.
[15] Ye X, Bunescu R, Liu C. Learning to rank relevant files for bug reports using domain knowledge[C]//Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering. ACM, 2014: 689-699.
[16] Blei D M, Ng A Y, Jordan M I. Latent dirichlet allocation[J]. Journal of machine Learning research, 2003, 3(Jan): 993-1022.
[17] Yu Z, Tan C H, Kang D. 10708 Probabilistic Graphical Models Final Project Presentation[J]. 2012.
[18] Ramage D, Hall D, Nallapati R, et al. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora[C]//Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1. Association for Computational Linguistics, 2009: 248-256.
[19] Landauer T K. Latent semantic analysis[M]. John Wiley & Sons, Ltd, 2006.
[20] Nguyen A T, Nguyen T T, Al-Kofahi J, et al. A topic-based approach for narrowing the search space of buggy files from a bug report[C]//Automated Software Engineering (ASE), 2011 26th IEEE/ACM International Conference on. IEEE, 2011: 263-272.
[21] Kim D, Tao Y, Kim S, et al. Where should we fix this bug? a two-phase recommendation model[J]. IEEE transactions on software Engineering, 2013, 39(11): 1597-1610.
[22] Asuncion A, Welling M, Smyth P, et al. On smoothing and inference for topic models[C]// Conference on Uncertainty in Artificial Intelligence. AUAI Press, 2009:27-34.
[23] Porteous I, Newman D, Ihler A, et al. Fast collapsed gibbs sampling for latent dirichlet allocation[C]// ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, Usa, August. DBLP, 2008:569-577.

1 寸数字照片
须为证件照，不能提供生活照，不能低于 300 像素

Name was born in which year. She/He received the Ph.D./M.S. degree in which area from which university in which year (or She/He is a Ph.D. candidate at which university). She/He is a professor/lecturer and Ph.D. supervisor at which university. Her/His research interests include *****, ****, etc.
作者名(出生年–)，性别，****(籍贯,具体到市、县或地区)人，何年何单位获何学位(或目前学历)，目前何单位何职称，主要研究领域为*****, ****。发表论文情况，主持或承担过何项目等。

1 寸数字照片
须为证件照，不能提供生活照，不能低于 300 像素

Name was born in which year. She/He received the Ph.D./M.S. degree in which area from which university in which year (or She/He is a Ph.D. candidate at which university). She/He is a professor/lecturer and Ph.D. supervisor at which university. Her/His research interests include *****, ****, etc.
作者名(出生年–)，性别，****(籍贯,具体到市、县或地区)人，何年何单位获何学位(或目前学历)，目前何单位何职称，主要研究领域为*****, ****。发表论文情况，主持或承担过何项目等。

Name was born in which year. She/He received the Ph.D./M.S. degree in which area from which university in which year (or She/He is a Ph.D. candidate at which university). She/He is a professor/lecturer and Ph.D. supervisor at which university. Her/His research interests include *****, ****, etc.

作者名(出生年–)，性别，****(籍贯,具体到市、县或地区)人，何年何单位获何学位(或目前学历)，目前何单位何职称，主要研究领域为*****, ****。发表论文情况，主持或承担过何项目等。

Name was born in which year. She/He received the Ph.D./M.S. degree in which area from which university in which year (or She/He is a Ph.D. candidate at which university). She/He is a professor/lecturer and Ph.D. supervisor at which university. Her/His research interests include *****, ****, etc.

作者名(出生年□)，性别，****(籍贯,具体到市、县或地区)人，何年何单位获何学位(或目前学历)，目前何单位何职称，主要研究领域为*****, ****。发表论文情况，主持或承担过何项目等。