面向软件缺陷报告的提取方法

林涛[1], 高建华[1], 伏雪[1], 马燕[1], 林艳[2]

（上海师范大学 计算机科学与工程系，上海，中国，200234）[1]

(奥克兰大学 信息系统系，奥克兰，新西兰，92019)[2]

摘要 软件工程中的软件缺陷报告在快速增长，开发者们越来越困惑于大量的缺陷报告。因此，为了缺陷修复和软件复用等目的，有必要研究软件缺陷报告的提取方法。该文提出一种提取方法，该方法首先合并缺陷报告中的同义词，然后建立空间向量模型，使用词频反文档频率以及信息增益等文本挖掘的方法来收集软件缺陷报告中单词的特征，同时设计算法确定句子复杂度以选择长句。最后将贝叶斯分类器引入该领域。该方法可以提高缺陷报告提取的命中率，降低虚警率。实验证明基于文本挖掘和贝叶斯分类器的软件缺陷报告提取方法在接受者工作特征曲线面积（0.71）、F-score（0.80）和 Kappa 值（0.75）方面有良好效果。

关键词 软件缺陷报告管理、文本挖掘、贝叶斯分类器、软件缺陷报告特征、空间向量模型、句子复杂度

中图法分类号 TP311.5　　**文献标识码** A

Extraction Approach for Software Bug Report

LIN Tao[1], GAO Jian-hua[1], FU Xue[1], MA Yan[1], LIN Yan[2]

（Department of Computer Science and Technology, Shanghai Normal University, Shanghai，200234, China）[1]

(Department of Information Systems, The University of Auckland, Auckland，92019，New Zealand) [2]

Abstract: There are increasingly bug reports in software engineering and developers bewildered by the rapid reports accumulation. Therefore, it is necessary to extract bug report for the task of bug fixing and software reuse, etc. The paper proposes a novel extraction approach. Synonyms were merged into one specific word firstly in the approach. Then, it sets up a vector space model, and some text mining methods, such as TF-IDF and information gain, are presented to collect word features for bug reports specifically, but also there is an algorithm for determining sentence complexity, so as to choose the sentence in long length. This work introduced Bayes classifier into bug report extraction. TPR is increased and FPR is decreased in this approach. The experiment proves that bug report extraction by using data mining and Bayes classifier is competitive through the evaluation of AUC(0.71), F-score（0.80） and Kappa value(0.75).

Keywords: Bug report management, Text mining, Bayes classifier, Bug feature, Vector space model, Sentence complexity

0 引言

过去十年见证了软件工程的快速发展，但是，越来越多的开发者陷入持续增长的软件缺陷报告困境，有如“焦油坑”。其重要原因是不能合理使用软件缺陷报告。当然，当代有种类繁多的集成开发环境，诸如 Eclipse 和 WingIDE；一些先进的软件工程管理工具，像 Microsoft Project, ProjectLibre 和 Openproj，可以有效帮助开发者处理管理事务，如表 1 所示。可是，另一方面，至今还没有关于软件缺陷报告提取方法的深入研究，以帮助开发者。

表 1 应用于软件工程各方面的一些工具

软件工程方面	工具
代码管理	Eclipse, WingIDE
工程计划	Microsoft Project, ProjectLibre, Openproj
软件缺陷报告提取工具	?

到稿日期：2015-04-29　返修日期：2015-07-14 本文受国家自然科学基金（61073163 和 61373004）和上海市企业自主创新专项资金项目（沪 CXY-2013-88）资助。

林涛（1988 —），男，甘肃兰州人，硕士研究生，CCF 学生会员（E200043943G），研究方向：软件测试、文本挖掘，Email：l.t@acm.org；高建华（1963 —），男，博士，教授，博士生导师，CCF 高级会员（E200005526S），主要研究方向为软件可靠性理论与设计、软件开发环境与开发技术、数据安全与计算机安全、网络测试、LSI/VLSI 测试等领域。

当然，有众多研究者以不同的文本挖掘方法研究软件缺陷报告提取方法。Pawan Goyal 等人提出了相对于传统的忽视上下文，而与之对应的基于上下文的通用文本提取方法[1]。另外一种基于图形的文本相似度和提取方法被 Michael T. Mills 等人提出[2]。

正如前所述，研究软件缺陷报告提取方法有着至关重要的作用。首先，之前很少有研究注意到软件缺陷报告的管理。以软件工程中非常有名的瀑布模型为例，该模型将软件生命周期分为了五大部分，即交流、计划、建模、开发以及部署，其中开发部分包括编码和测试。虽然测试包括了单元测试、集成测试和系统测试，但是整个软件工程忽视了软件缺陷报告管理的问题。维护需花费高额成本的一个可能原因就是缺少测试。更确切的说，是软件工程中软件缺陷报告提取方法的短板问题。

在软件复用升级等方面，软件缺陷报告占有着越来越重要的位置。一些研究者认为必须考虑哪些软件缺陷报告对于整个软件工程更加重要，以此决定修复软件缺陷的先后顺序[3]。

无可否认的是，在特定的软件工程中代码确实有着重要地位，但是对于将来其它的软件工程，代码没有过多用途。与此相反的是，软件缺陷报告却深刻的影响将来软件开发过程，比方说作为参考和重审。问题在于：针对当代软件工程来说缺陷报告自身存在种种问题，主要由于有太多的缺陷报告需要归档，一方面，该软件工程的参与者也往往容易忽视或者忘记一些重要或者主要的缺陷报告；另一方面，其他非参与者去整理归纳缺陷报告，更是面临不熟悉领域知识和具体项目工程要求等重重困难。虽然在修补完成软件缺陷之后，一些软件公司要求开发者写简短的缺陷报告的总结。这种方法常常因为以下两个原因被置之不理。首先，诚然大多数开发者是优秀的软件缺陷修复者，但是这不等同于他们是同样出色的软件缺陷报告的提取者。第二个原因是，开发者通常没有兴趣和精力去完成缺陷报告的总结提炼，因为大多数公司更看重修复结果，而不是提取归档。

因此，基于以上分析，对于软件缺陷报告的自动提取是迫在眉睫的任务。将文本挖掘和模式识别的方法应用于这个领域是明智的。

本文主要展开了以下四个方面的工作,即软件缺陷报告提取方法的分类器选择、合并同义词、特征提取以及评价:接受者工作特征曲线面积和统计分析。

本文的一个主要工作是分类过程，本研究的总体思路包括合并同义词，建立空间向量模型，以及单词和句子的特征选择。

本文章节安排如下：

第一部分，简洁的介绍了朴素贝叶斯分类器，以及为什么选用此分类器作为软件缺陷报告的最优分类器。

第二部分，同义词总结。本部分的作用是辅助下一部分。

第三部分，由于本文主要研究软件缺陷报告的提取方法，需要用一些抽象方法对软件缺陷报告进行处理。因而，研究和设计了基于空间向量模型的特征选取方法。

论文最后，给出实验，并且计算接受者工作特征曲线所围面积 AUC 以及其它相关评价标准如精确值，返回值， F-score 以及 Kappa 分析等。

虽然有关于文本挖掘[4]以及基于支持向量机的软件缺陷报告总结的工作。但是，此论文的研究是首次将朴素贝叶斯分类器应用于软件缺陷报告总结提取。

1 分类器的选择

在机器学习领域，已经有很多性能不同的分类器，诸如支持向量机和逻辑回归。之前 Sarah Rastlar 等人使用支持向量机方法提取软件缺陷报告[5]。诚然，在很多场合支持向量机方法是一个有效的方法，如地理分析和人脸识别。但是，另一方面，针对软件缺陷报告来说，使用支持向量机不是明智的和可推荐的，主要有以下原因：使用支持向量机，必须计算软件缺陷报告中大量的“向量”（词语等要素的权重），存在着越来越依赖于高维向量的趋势。这样很容易在训练集中取得很好效果，却在测试集中效果不理想。与此相反，通过使用统计方法，朴素贝叶斯分类器却可以有效避免这种问题的发生[6]。

一个完整的软件缺陷报告里的每个句子可以被划分为两类，即“重要”和“非重要”，数学表示如公式(1)：

$$C_i = \begin{cases} C_1, i=1\text{为“重要类”} \\ C_2, \ i=2\text{为“非重要类”} \end{cases} \tag{1}$$

其中 C_i 为句子类别，C_1 是被提取作为总结的句子类别。

缺陷报告里的每句话拥有 m 个属性，可以被表示为 a= $(a_1, a_2, a_3, \ldots, a_m)$。因此，提取总结缺陷报告的问题就是一个后验概率的问题。根据贝叶斯公式，软件缺陷报告每句话 a_x属于类别 C_i的概率可由公式(2)确定：

$$P(C_i|a_x) = P(C_i)\frac{P(a_x|C_i)}{P(a_x)} \quad (2)$$

$P(C_i)$ 是缺陷报告中句子属于特定类别的先验概率。

选用贝叶斯分类器的一个很重要原因，是因为该分类器容易避免在软件缺陷报告中主题相关的特定术语。换句话说，该分类器将来可以扩展到软件工程的其它文本总结提取中，例如需求规约、概要设计说明书和详细设计说明书等。

2 合并同义词

本文主要研究在国际上有影响的开源软件，其中软件缺陷报告全部为英语。众所周知，英语以及其它大部分自然语言存在二义不确定性的问题。为了得到更好效果，本文合并了软件缺陷报告中的常用同义词，缩减成一个单一关键词。

具体过程如下：

（1）统计本文所用软件缺陷报告数据库[7]中单词的词频数量，其中出现次数大于 100 次的单词如表 2 所示。

表 2 出现频率大于 100 次的单词

单 词	出 现 频 率
id	7584
sentence	3504
problem	1904
suggestion	1585
fix	730
the	599
links	593
meta	508
is	381
to	367
disagreement	347
a	304
agreement	276
in	213
labels	210
extractive	210
annotation	210
abstractive	210
and	204
it	183
of	175
this	153
be	130
for	129
was	125
that	125
not	101

（2）选择其中的主要实词（名词和动词）作为“关键词”，报告中其它同义词全部替换为该词，如表 3 所示。

表 3 合并同义词

关键词	可选词
problem	difficulty, drawback, issue
fix	tackle, arrange, solve
good	fine, ok, not bad
suggestion	proposal, proposition, submission, idea, recommendation
agreement	contract, arrangement, promise

3 软件缺陷报告的特征

在本节，首先引入空间向量模型，然后重点介绍基于文本挖掘理论的特征选择。

3.1 空间向量模型

目前，已有多种文本挖掘的方法，像语义分析、

概率潜在语义分析模型、狄利克雷划分、以及相关主题模型[8]。但是，使用的最广泛的，并且适合于缺陷报告提取的是由 G Salton 等人提出的空间向量模型[9]。

在空间向量模型中，软件缺陷报告被表示为向量，软件缺陷报告中的每句话被看成由二元特征组成的向量，如公式(3)，

$$\mathrm{b}=\{(t_1,w_1),(t_2,w_2),\ldots,(t_n,w_n)\} \tag{3}$$

t_i为特征项，w_i为相应的权重，n 为特征长度。在具体的缺陷报告中的特征空间的长度是确定的。因而，(3)式可以化简为式(4)，

$$\mathrm{b}=\{w_1,w_2,\ldots,w_n\} \tag{4}$$

软件缺陷报告中的每句话都可以用这个权重公式表示。根据这个公式决定软件缺陷报告中的每一个句子是否应该被提取。

在一个软件缺陷报告中，根据以下依据确定t_i：单词特征（词频反文档频率(TF-IDF)、 χ^2 统计量、信息增益（IG））和句子特征（句子复杂度）。

3.2 词频反文档频率(TF-IDF)

在软件缺陷报告中，首先需要考虑重要词汇。显然，第 2 部分合并同义词中的“关键词”应被选做特征。另一方面，本文尝试使用更为客观的方法，即计算词频反文档频率(TF-IDF)[10]，这是一个评价词语重要性的指标。这个指标包含两个独立的参数：特征频率（Term Frequency，TF）、文档频率（Document Frequency，DF）。

特征频率 TF 是单词出现的频率，如式(5)：

$$\text{特征频率}=\frac{\text{特定单词出现次数}}{\text{软件缺陷报告中总字数}} \tag{5}$$

文档频率 DF 定义如公式(6)，

$$\text{文档频率}=\log\frac{\text{含有特定单词的文档数}}{\text{软件缺陷报告中总文档数}} \tag{6}$$

反文档频率（IDF）就是对文档频率取反，定义如式(7)，

$$\text{反文档频率}=\frac{1}{\text{文档频率}} \tag{7}$$

一方面，比较高的文档频率说明该单词更加重要。但是，另一方面，低文档频率的单词有可能包含更多信息，从这方面讲，包含该单词的句子也应该被提取。因而，使用词频反文档频率(TF-IDF)这个参数作为折中，该参数是特征频率与反文档频率的乘积，如式(8)所示，

$$\text{词频反文档频率}=\text{特征频率}*\text{反文档频率} \tag{8}$$

3.3 χ^2检测

除了词频反文档频率，本研究中使用 χ^2检测作为特征选择。更确定说，χ^2检测表明特定单词与软件缺陷报告的关系，如公式(9)，

$$\chi^2(\mathrm{m},\mathrm{e})=\frac{T(AD-BC)^2}{(A+C)(B+D)(A+B)(C+D)} \tag{9}$$

其中，m 表示单词，e 表示类别，即提取和不提取两类。T 表示软件缺陷报告中的总文档数。A 是包含特定单词 m 且属于缺陷报告提取的文档数。B 是包括特定单词但不属于缺陷报告提取的文档数。C 是不包含特定单词的应被提取的文档数。D 是不包括特定单词 m，同时也未被提取的文档数。

显然，当特征与提取的文档相互独立，$\chi^2(\mathrm{m},\mathrm{e})=0$。也可以说，$\chi^2(\mathrm{m},\mathrm{e})$越高，说明包含特定单词 m 的句子就越有可能作为缺陷报告的提取。

3.4 信息增益

决定是否选择特定单词作为特征的第三个参数是信息增益（IG）[11]，表明特定单词包括多少信息，表示为公式(10)。

$$\begin{aligned}\mathrm{IG}(\mathrm{t})=&-\sum_{i=1}^{2}p(c_i)*log_2p(c_i)+p(t)\sum_{i=1}^{2}p(c_i|t)*log_2p(c_i|t)\\&+p(\bar{t})\sum_{i=1}^{2}p(c_i|\bar{t})*log_2p(c_i|\bar{t})\end{aligned} \tag{10}$$

p(t)是包含词语 t 的文档数/缺陷报告总文档数。$p(c_i|t)$是条件概率。

信息增益决定了特定单词给缺陷报告带来多少信息。越多的信息，越表明该单词需要被作为特征。

3.5 句子复杂度

除了单词，缺陷报告的另一个重要特征是句子复杂度。本文基于以下三点考量：

（1）倾向长句。在基于 Unigram Language Model 方法的口语提取研究中，只包含个别单词的短句会造成结果的不确定性[12]。更高的复杂度表明

该句话含有更重要的信息，因而相对短句，本论文更倾向于提取长句。以本文实验的软件缺陷报告为例，诸如‘Good point.’, ‘But go ahead’, and ‘How does it sound?’，这类句子未包括有效信息，应排除在总结提取之外。

（2）句子长度大于全文所有句子平均长度的句子，可以作为长句的一个指标。

（2）相近位置的句子表达意义近似信息，因而不必全部选择提取。

具体过程由下列算法实现：

Input: <(e_1,length(e_1))...,(e_n, length (e_n))>

Input: average

```
1    T=ø
2        for i=1→n do
3            if ei<average then
4                if ei+1< average then
5                    ei=-1, ei+1=-1, i++
6                else
7                    ei=-1
8                end if
9                else
10                   if ei+1<average then
11                       ei=1
12                   else
13                       max(ex,ei+1)=1
14                   end if
15           end if
16       i++
17       end for
18
19       for i=1→n do
20           if ei=1 then
21               ei∈T
22           end if
23       end for
24   return T
```

输入是每个句子的编号 e 和包含的单词数量 length（e），以及软件缺陷报告中所有句子平均含有的单词数量 average。设立一个集合 T（第 1 行）。对于每句话，如果其长度小于平均长度（第 2、3 行），则检查下句话的长度，若长度也小于平均长度（第 4 行），则将这两句同时置为-1（第 5 行）；若第二句话长度大于或等于平均长度，则仅将第一句话值为-1（第 7 行）。如果第一句话长度大于或等于平均长度，并且第二句话长度小于平均长度（第 10 行），则将第一句话置为 1（第 11 行）。如果前后两句话长度均大于或等于平均长度，则将其中较长的一句话置为 1（第 13 行）。最后，将所有为 1 的句子放入集合 T 中（第 21 行），并且返回集合 T（第 24 行）。

在句子复杂度方面，集合T中的句子符合要求。

4 实验

本节主要对基于文本挖掘和贝叶斯分类器的软件缺陷报告提取方法进行验证。

本实验，重点关注以下两个问题：

（1）软件缺陷报告能否自动总结提取？

（2）该提取方法能否达到理想效果？并且是否优于其他方法？

以此，计算真阳性率（true positive rate， TPR）和假阳性率（false positive rate， FPR）。之后，描绘接受者工作特征曲线（ROC），计算其所围面积（AUC），以及相关统计分析。

4.1 实验准备

实验环境如下：操作系统为 Windows Server 2012 64bit，处理器为 Intel Xeon E3 3.10Ghz，内存为 8GB。开发语言为 Python，开发工具为 IDLE。同时，使用了 Sarah Rastkar 等人的软件缺陷报告数据库[7]。一个选用该数据库的重要原因是因为该数据库已有一个人工完成的总结提取。然后，使用了基于 Python Textblob[13]的朴素贝叶斯分类器。

所选数据库包含36个软件缺陷报告，共计2361个句子。人工选出了其中465个句子作为总结提取。由于该数据库较小，使用模式识别中的“留一交叉验证法”（使用大部分句子作为训练集，仅留下一个句子作为测试集，循环迭代）作为贝叶斯分类器训练集与测试集选取的准则。实验过程如图 1 所示。

使用 Sarah 等人提出的 BRC 方法作为对照组[5]。BRC 方法借鉴了会议和邮件内容提取方法，使用支持向量机作为分类器。

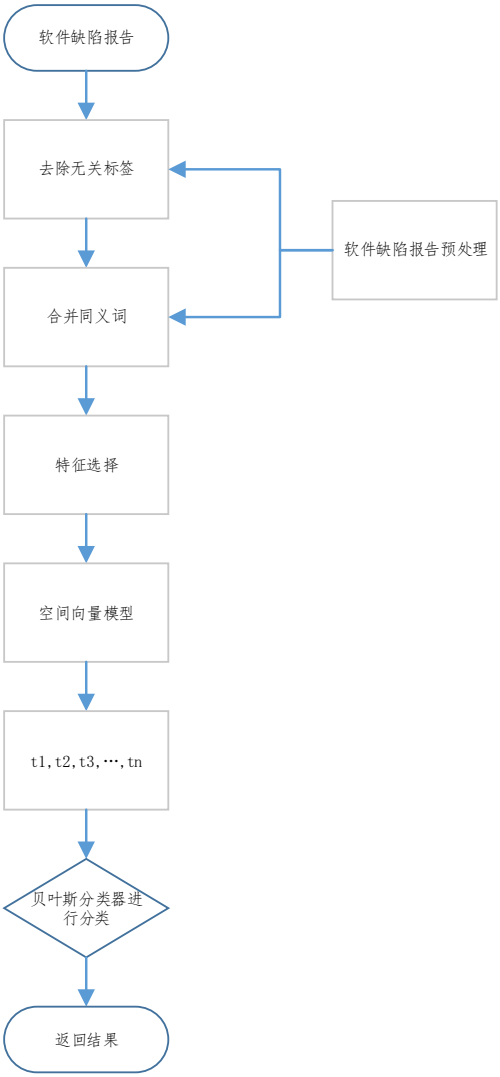

图 1 软件缺陷报告提取的实验过程

4.2 结果与分析

实验结果如表 4 所示。

表 4 实验结果

		手工提取		
		提取	未提取	
本方法	提取	388	112	500
	未提取	77	1784	
		465		
		手工提取		
		提取	未提取	
BRC 方法	提取	214	286	500
	未提取	251	1610	
		465		

以此描绘接受者工作特征曲线。首先，需要计算真阳性率，如公式(11)，

$$\text{真阳性率（TPR）}=\frac{\text{分类器选出的句子中是人工选出的数量}}{\text{人工选出的总量}} \tag{11}$$

真阳性率也称命中率，表示贝叶斯分类器的灵敏度。

然后，计算假阳性率，如公式(12)，

$$\text{假阳性率（FPR）}=\frac{\text{分类器选出的句子中不是人工选出的数量}}{\text{缺陷报告中不是人工选出的句子的总量}} \tag{12}$$

假阳性率也称虚警率，是分类器的特异性。

最后，描绘接受者工作特征曲线（ROC），接受者工作特征曲线表示缺陷报告提取的代价和收益之间的关系，可以平衡真阳性率和假阳性率之间的制约关系，能够均衡度量本分类器的性能，如图 2 所示。

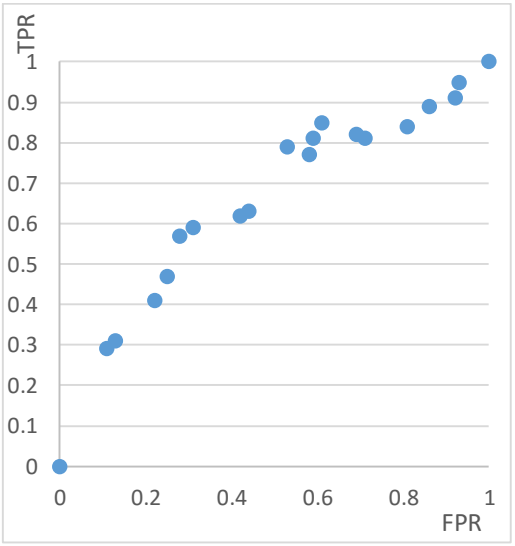

图 2 贝叶斯分类器的接受者工作特征曲线

根据图 2 的曲线，接受者工作特征曲线所围面积 AUC 为 0.73，优于基于支持向量机的 BRC 方法所得出的结果（0.71），而且贝叶斯分类器使用了更低的维数。

可以更进一步分析贝叶斯分类器的有效性通过计算一些标准的数值，即精确值、返回值、F-score 以及 Kappa 分析。

当设定分类器选出 500 个句子时，结果如表 4 所示。

计算精确值和返回值，分别如公式(13)和(14)，

$$\text{精确值}=\frac{\text{分类器提取的句子中是人工提取的数量}}{\text{分类器提取的句子数量}} \tag{13}$$

$$\text{返回值} = \frac{\text{分类器提取的句子中是人工提取的数量}}{\text{人工提取的句子数量}} \quad (14)$$

事实上，返回值与真阳性率是相等的。在多数情况下，不能保证精确值与返回值都取得较高值，通常评价分类器需要使用 F-score，如公式(15)所示。F-score 是精确值与返回值的调和平均值，F-score 值介于 0 到 1 之间，0 为最差，1 为最优。

$$\text{F}-\text{score} = 2*\frac{\text{精确值}*\text{返回值}}{\text{精确值}+\text{返回值}} \quad (15)$$

计算结果如表 5 所示。

表 5 提取方法的精确值、返回值和 F-score 分析

	精确值	返回值	F-score
本研究方法	0.78	0.83	0.80
BRC 方法	0.43	0.46	0.44

两种方法分别与人工提取结果的相符合程度可以使用 Cohen's Kappa 分析。Kappa 分析是比较两组数据相符一致性的一种统计方法，其值属于 0（完全不同）到 1（完全相同）之间，值越高，表示自动提取方式的结果与人工结果越吻合。本方法和 BRC 方法的 Kappa 值分别为 0.75 和 0.30。

综上所述，本研究中的方法优于其他方法，图 3 是本研究方法与 BRC 方法的有效性对比。

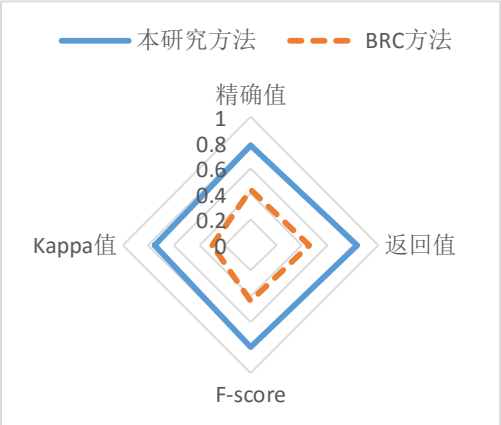

图 3 提取方法的有效性比较

因为自然语言本身的原因，所以存在着一些问题使得很难提高缺陷报告自动提取的有效性，比如自然语言中的修改，撤销。例如，一位开发者说："Once I started messing around with what is ticked, the problem went away."（一旦我开始点击这个，问题就解决了），但是这位开发者紧接着说："Now I am also unable to reproduce it."（现在我也不能重做这个了）。这两个句子是典型的前后矛盾，但是不论是基于支持向量机的分类器还是贝叶斯分类器，都很难做出正确的决定。

因而，一方面要更深入研究分类器的使用。另一方面，在不限制开发者创造性思维以及有效团队合作的前提下，建立一个基本的书写软件缺陷报告的标准是适当的。

5 总结

开发者需要优良的缺陷报告以提高软件工程的效率。但是，到目前为止，除了使用基于支持向量机的复杂方法以外，很少有研究聚焦于软件缺陷报告的提取方法。

本文提出了一种简单，但有效的基于文本挖掘和贝叶斯分类器的提取方法。

本文研究结果的一个重要考量是选用的数据库。详细说，这个数据库不是非常庞大，人工提取也不是客观的。因而，或许不能从严格客观的角度评价分类器的有效性。在将来的工作中，需要建立一个大规模的软件缺陷报告总结提取的数据库。

其次，数据库中的软件缺陷报告都来源于四个国际开源软件，即 Eclipse Platform, Gnome, Mozilla 和 KDE。研究者不容易得出此研究在商用软件环境中的使用优势。同时，这个问题很难解决，因为几乎所有的商用软件缺陷报告都是其所属公司的机密。

至于将来的工作，有两个方向。首先，在短期，我们试图设计一个面向软件缺陷报告的消除同义词的分类器。其次，在更长远时间，如果组合多种分类器，比如贝叶斯分类器和决策树分类器进行软件缺陷报告的提取，是否能取得更好的效果？因而，这也是一个方向。

参考文献

[1] Goyal P, Behera L, Mcginnity T M. A Context-Based Word Indexing Model for Document Summarization[J]. IEEE Transactions on Knowledge and Data Engineering. 2013, 25(8): 1693-1705.

[2] Mills M T, Bourbakis N G. Graph-Based Methods for Natural Language Processing and Understanding—A Survey and Analysis [J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems. 2014, 44(1): 59-71.

[3] Alenezi M, Banitaan S. Bug Reports Prioritization: Which Features and Classifier to Use?[C]. Miami, FL: 12th International Conference on Machine Learning and

Applications (ICMLA). 2013.

[4] Kastner C, Dreiling A, Ostermann K. Variability Mining: Consistent Semi-automatic Detection of Product-Line Features[J]. IEEE Transactions on Software Engineering. 2014, 40(1): 67-82.

[5] Rastkar S, Murphy G C, Murray G. Automatic Summarization of Bug Reports[J]. IEEE Transactions on Software Engineering. 2014, 40(4): 366-380.

[6] 陈旋，刘健，冯新淇，等. 基于朴素贝叶斯的差分隐私合成数据集发布算法[J]. 计算机科学. 2015, 42(1): 236-238. Chen Xuan, Liu Jian, Feng Xin-qi, et al. Differential Private Synthesis Dataset Releasing Algorithm Based on Navie Bayes[J]. Computer Science, 2015,42(1):236-238

[7] Rastkar S, Murphy G C. Summarizing Software Artifacts[EB/OL].[2015/4/16]. https://www.cs.ubc.ca/cs-research/software-practices-lab/projects/summarizing-software-artifacts

[8] Sangno L, Baker J, Song J, et al. An Empirical Comparison of Four Text Mining Methods[C]. Honolulu, HI: 43rd Hawaii International Conference on System Sciences (HICSS). 2010.

[9] Saari P, Eerola T. Semantic Computing of Moods Based on Tags in Social Media of Music[J]. IEEE Transactions on Knowledge and Data Engineering. 2014, 26(10): 2548-2560.

[10] Mishra A, Singh G. Improving keyphrase extraction by using document topic information[C]. Kaohsiung: IEEE International Conference on Granular Computing (GrC). 2011.

[11] Wijayasekara D, Manic M, Mcqueen M. Information gain based dimensionality selection for classifying text documents[C]. Cancun: IEEE Congress on Evolutionary Computation (CEC). 2013.

[12] Kuan-Yu C, Shih-Hung L, Chen B, et al. A recurrent neural network language modeling framework for extractive speech summarization[C]. Chengdu: IEEE International Conference on Multimedia and Expo (ICME). 2014.

[13] Loria S. TextBlob[EB/OL].[2015/4/16]. http://textblob.readthedocs.org/en/dev/

联系人：林涛

联系方式：l.t@acm.org

联系电话：13916704339