计算机科学
COMPUTER SCIENCE

基于领域知识微调的缺陷报告严重性预测

陈冰婷, 邹卫琴, 蔡碧瑜, 刘文杰

引用本文

陈冰婷, 邹卫琴, 蔡碧瑜, 刘文杰. 基于领域知识微调的缺陷报告严重性预测[J]. 计算机科学, 2024, 51(6A): 230400068-7.

CHEN Bingting, ZOU Weiqin, CAI Biyu, LIU Wenjie. Bug Report Severity Prediction Based on Fine-tuned Embedding Model with Domain Knowledge [J]. Computer Science, 2024, 51(6A): 230400068-7.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.230400068

基于领域知识微调的缺陷报告严重性预测

陈冰婷　邹卫琴　蔡碧瑜　刘文杰
南京航空航天大学计算机科学与技术学院　南京 211106
(btchen@nuaa.edu.cn)

摘　要　有效预测缺陷报告的严重性,对快速、准确分派缺陷报告,帮助开发人员及时发现并处理软件中的缺陷至关重要。现有主流的基于传统信息检索或通用预训练模型的缺陷报告严重性预测方法,存在忽略上下文语义或缺陷报告特性导致预测效果受限的问题。对此,提出一种基于领域知识微调的缺陷报告严重性预测方法。利用能充分考虑文本上下文语义的BERT预训练模型,并使用缺陷报告数据对其进行模型微调使其学习到相关的领域知识。微调后的BERT模型用于抽取缺陷报告的语义特征,随后使用支持向量机进行严重性预测模型的构建。在Mozilla,Eclipse和Apache选取的共计15个项目上进行的实验表明,在准确率、召回率和F1值上,相较传统的信息检索方法,所提方法分别能提升4.5%～22.0%,3.0%～22.0%,4.0%～22.0%;相较通用BERT模型,微调后的BERT模型的准确率、召回率和F1值分别能够提高2.0%～5.1%,1.9%～5.1%,1.8%～5.0%。

关键词:词嵌入;BERT;预训练模型;缺陷报告;微调;严重性预测
中图分类号　TP311

Bug Report Severity Prediction Based on Fine-tuned Embedding Model with Domain Knowledge

CHEN Bingting,ZOU Weiqin,CAI Biyu and LIU Wenjie
College of Computer Science and Technology,Nanjing University of Aeronautics and Astronautics,Nanjing 211106,China

Abstract　Accurately predicting the severity of bug reports is crucial for efficiently assigning them and facilitating developers to timely detect and fix software bugs. However,existing severity prediction methods based on traditional information retrieval or general pre-training models have limitations in prediction accuracy due to the ignorance of context semantics or bug report characteristics. To address this problem,this paper proposes a severity prediction method based on domain knowledge fine-tuning. A BERT pre-trained model that can fully consider the semantic context of text is used,and the model is fine-tuned with bug report data to learn relevant domain knowledge. The fine-tuned BERT model is then used to extract semantic features of bug reports,and a support vector machine is employed to construct a severity prediction model. Experimental results on 15 projects,including Mozilla,Eclipse,and Apache,demonstrate that compared with traditional information retrieval methods,the proposed method can improve the accuracy,recall,and F1 score by 4.5% to 22.0%,3.0% to 22.0%,and 4.0% to 22.0%,respectively. Compared with the general BERT model,the fine-tuned BERT model can improve the accuracy,recall,and F1 score by 2.0%～5.1%,1.9%～5.1%,and 1.8%～5.0%,respectively.

Keywords　Word embedding,BERT,Pretrained model,Bug report,Fine-tuning,Severity prediction

1　引言

缺陷报告是软件开发中一种常见的文档类型,用于记录开发和测试过程中发现的缺陷和问题[1-2]。有效预测缺陷报告的严重性对于快速准确地分派缺陷报告具有重要意义,能够使开发人员更快速地响应和修复缺陷,从而提高软件开发的质量和效率[3-6]。过去的研究中,有的方法使用传统信息检索技术,通过对缺陷报告进行文本挖掘来预测缺陷报告的严重性,例如向量空间模型(Vector Space Model,VSM)和潜在狄利克雷分配(Latent Dirichlet Allocation,LDA)等。这些方法通过统计词频来提取有效的信息,但在处理复杂的语义关系及深入分析上下文信息上存在局限性[7-8];还有基于启发式规则或机器学习的方法来对缺陷报告进行分类,这类方法主要利用预定义的规则或人工设计的特征对缺陷报告进行分析,但需要耗费大量的人力和时间进行规则制定或特征工程[9-11]。

而随着深度学习的发展,对文本特征的表示也更进一步。词嵌入技术通过将每个单词表示为低维稠密向量来捕捉单词的语义和语法特征,进而还有在大量数据上进行预训练、学习通用语言表示的预训练模型。目前也已有相关研究使用该类技术对缺陷报告的严重性进行预测,如Ramay等[12]采用Word2Vec模型来抽取缺陷报告中的文本信息;Jia等[13]、Su等[14]则使用了FastText模型;Kumar等[15]通过实证研究的

基金项目:国家自然科学基金(62002161);南京航空航天大学前瞻布局科研专项资金;南京航空航天大学人才科研启动基金
This work was supported by the National Natural Science Foundation of China(62002161),Fund of Prospective Layout of Scientific Research for NUAA(Nanjing University of Aeronautics and Astronautics) and Scientific Research Foundation for the Introduction of Talent for NUAA.
通信作者:邹卫琴(weiqin@nuaa.edu.cn)

方式对比了不同的词嵌入技术即 CBOW, Skip-gram, GloVe, Word2Vec, FastText, BERT 和 GPT 在缺陷报告严重性预测上的表现。然而,现有的研究虽然使用了预训练模型,但并未考虑到缺陷报告作为一种特定领域的文本,具有特定的语言特征和上下文关系。如果预训练语言模型未经过领域知识的学习微调,它将不能更好地理解该领域语言的特征和上下文关系,从而可能导致其表征语义的性能下降。

因此,本文提出了一种基于领域知识微调的缺陷报告严重性预测方法。该方法使用缺陷报告领域知识对预训练模型进行微调,使该预训练模型能够更好地理解缺陷报告的文本特征和上下文关系。BERT(Bidirectional Encoder Representations from Transformers)是目前主流使用的一种先进预训练模型,它是基于 Transformer 架构并使用双向学习,因此能够更好地理解上下文信息并表征文本内容,也在许多自然语言理解任务上具有优越性能,所以我们选择 BERT 作为预训练模型进行探究。本研究首先使用大量经过预处理的缺陷报告构建缺陷报告专用语料库,并使用该专用语料库对 BERT 预训练模型进行领域知识微调,随后使用微调的 BERT 预训练模型抽取缺陷报告的文本特征,并将其输入到支持向量机(Support Vector Machine, SVM)中,实现缺陷报告的严重性分类预测。实验结果表明,缺陷报告专用语料库确实能够提高 BERT 预训练模型在缺陷严重性预测任务上的表现;与传统的信息检索方法 VSM 相比,使用 BERT 词嵌入技术能提高 4.5%～22.0%的准确率、3.0%～22.0%的召回率和 4.0%～22.0%的 F1 值;而微调的 BERT 预训练模型较通用 BERT 预训练模型能提高 2.0%～5.1%的准确率、1.9%～5.1%的召回率和 1.8%～5.0%的 F1 值。

本文第 2 章介绍相关工作与研究背景;第 3 章介绍基于领域知识微调的缺陷严重性预测方法;第 4 章介绍具体实验并对实验结果进行分析;第 5 节讨论了本研究的局限性;最后总结全文。

2 相关工作与研究背景

2.1 缺陷报告

缺陷报告是软件从业人员发现、修复缺陷和确保软件质量的重要信息载体。图 1 给出了一个典型的、由缺陷追踪系统(Bug Tracking System, BTS)收集的缺陷报告。如图 1 所示,一份缺陷报告通常包含 BugID、状态(Status)、产品(Product)、组件(Component)、重要性(Importance)、报告者及时间(Reported)、问题概要描述(Summary)、问题详细描述(Description)、评论(Comment)等字段。BugID(如图中显示为 550200)是一个数字,用于唯一地识别一个项目的错误。问题概要描述(Summary)通常是一句话,简要地概括了一个缺陷。而问题详细描述(Description)提供了关于该缺陷的细节,其中的细节一般包括重现该缺陷的步骤、预期行为或观察到的行为等。状态(Status)描述了一个错误报告的当前状态。严重性(Severity)字段即表明该缺陷报告的严重性(如图中的"normal")。因本文仅探究缺陷报告语义特征,不考虑其他特征,故仅使用缺陷报告的问题概要描述、问题详细描述、状态和严重性字段。

图 1 Eclipse 缺陷报告片段示例

Fig. 1 Example of bug report in Eclipse

2.2 缺陷严重性预测

在软件测试或维护时,遇到的有些缺陷较为严重,如数据损坏,需要立即修复;而有些较为轻微,可以推迟到资源可用时再修复。缺陷严重性预测旨在预测缺陷报告的严重性。缺陷报告的严重性揭示了软件中缺陷的严重程度。因此,缺陷严重性预测可以帮助分派者更好地分配修复缺陷的资源,并尽可能确保软件正常工作。为了能够在有限的资源下分派并解决尽可能多的更严重的缺陷报告,研究人员提出了许多关于缺陷严重性预测的方法和技术。

基于信息检索的方法是预测缺陷报告严重性的一大类方法。而在信息检索领域,TF-IDF 算法被广泛用于文本分类、文本聚类、相似度计算等任务。TF-IDF 算法可以通过计算单词在文档中的出现频率(Term Frequency, TF)以及在语料库中的逆文档频率(Inverse Document Frequency, IDF)来确定单词的重要性。具体来说,一个单词的 TF-IDF 值等于它的 TF 与 IDF 之积。因此,可以说 TF-IDF 算法是信息检索方法中最常用的算法之一。而 VSM 作为使用 TF-IDF 算法的经典方法,被广泛使用。

早在 2008 年,Menzies 和 Marcus 等[16]就率先提出一种名为"SEVERIS"的自动化方法,通过使用文本挖掘技术处理缺陷报告中的文本信息来提取特征,建立机器学习模型来预测缺陷的严重性。之后,Lamkanfi 等[17]在三大开源项目 Eclipse, Mozilla 和 GNOME 上也同样通过对文本信息进行处理,使用朴素贝叶斯(Naive Bayes)对缺陷报告的严重性进行粗粒度的分类。Sari 和 Siahaan[18]也采用了类似的方法,并使用 InfoGain 来提取更多的相关特征,最后使用 SVM 来开发缺陷严重度预测模型。2013 年,Ruchika 等[10]使用 TF-IDF 对报告中的信息进行选择获取,然后通过 SVM 进行报告严重性的分类。Jindal 等[19]也做了类似的工作,从缺陷描述中提取特征,使用 TF-IDF 来提取特征,然后使用径向基函数网络来构建用于开发缺陷严重度预测的模型。2014 年,Yang 等[20]利用主题模型(LDA)并结合缺陷报告的组件、优先级、产品、严重性等特征,通过 K 近邻算法

(*K*-Nearest Neighbor,KNN)来对缺陷的严重性进行分类预测。2017 年,Jindal 等[21]在公开领域数据集 PITS 上使用向量空间模型(Vector Space Model,VSM)进行文本挖掘,再比较多元逻辑回归(Multinomial Logistics Regression,MMLR)、多层感知器(Multilayer Perceptron,MLP)和决策树(Decision Tree,DT)建立模型的预测效果。2021 年,Kumar 等[15]通过实证研究的方式对比了不同的词嵌入技术(CBOW,Skip-gram,GloVe,Word2Vec,FastText,BERT,GPT)和多种分类方法在缺陷报告严重性预测上的表现,发现 GloVe 和 Word2Vec 在一众词嵌入技术里表现突出,而 SVM 这种分类方法最好。同年,Jia 等[13]提出使用 FastText 模型来抽取缺陷报告中文本信息的 EKD-BSP 方法,并与传统经典分类器——朴素贝叶斯(Naive Bayes,NB)、KNN、逻辑回归(Logistic Regression,LR)和深度学习方法——长短期记忆网络(Long Short-Term Memory,LSTM)进行比较,发现能够提升缺陷严重性预测的准确性。在这基础上,Su 等[14]于 2022 年提出了同样使用 FastText 来提取文本特征的 CIL-BSP 方法,并进一步考虑了类不平衡方法来预测缺陷报告的严重性。

2.3 词嵌入技术——BERT

词嵌入技术能够更好地提取缺陷报告的语义信息,而随着预训练模型的兴起,基于预训练模型的方法在自然语言处理领域取得了极大的成功。其中,BERT 由于其强大的表征能力和在多项 NLP 任务上的卓越表现,成为了自然语言处理领域的研究热点。BERT 是一种预训练语言模型[22],使用了大量的文本数据进行预训练,包括经典的维基百科数据集、大型书籍文本数据集(BookCorpus)等等。该预训练模型使用 Transformer 架构,包含了多个编码器(Encoder)模块,通过无监督的方式从海量未标注文本中双向深度学习、预训练出通用的语义表示模型,从而能够捕捉单词之间的复杂语义关系,同时还可以理解语法和上下文信息。同时,BERT 在多项自然语言处理任务上较其他预训练模型达到了相对出色的效果,例如文本分类、问答、文本生成等。

3 基于领域知识微调的缺陷报告严重性预测方法

本节主要介绍基于领域知识微调的缺陷严重性预测方法,首先介绍整体框架,随后详细介绍专用语料库的构建、BERT 预训练模型的微调和分类预测。

3.1 整体框架

图 2 给出了基于领域知识微调的缺陷严重性预测方法的整体架构。

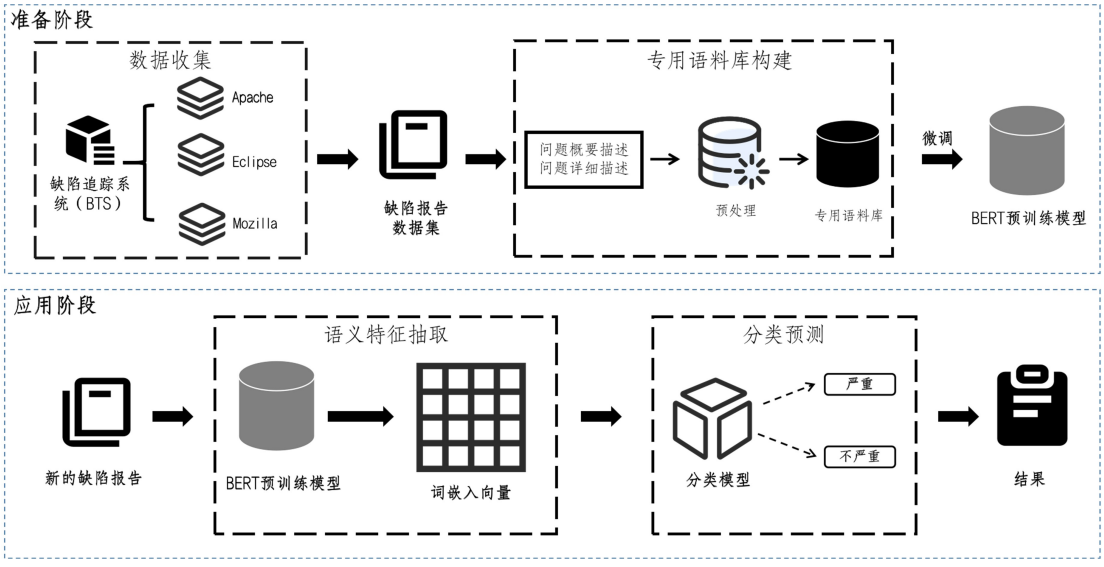


图 2 基于领域知识微调的缺陷报告严重性预测方法的整体框架

Fig. 2 Overall framework of bug report severity prediction method based on fine-tuned domain knowledge

在训练阶段,首先使用爬虫从缺陷追踪系统中爬取三大开源软件(Eclipse,Mozilla,Apache)的缺陷报告数据;并对获得的缺陷报告数据集进行预处理,包括数据清洗、去重、分词等操作,然后构建专用语料库(详见 3.2 节),用于微调通用 BERT 预训练模型。3.3 节详细介绍了微调过程,其中包括模型参数和训练模型的过程。经过微调后,得到了在缺陷严重性预测任务上效果最好的模型。在测试阶段,使用训练好的模型对缺陷报告进行语义抽取,将缺陷报告转换为词向量表示,最后通过分类模型对缺陷报告进行严重性预测。整个过程中,我们使用了十则交叉验证等方法来确保模型的可靠性和鲁棒性。我们将于下文详细阐述每个步骤的细节。

3.2 专用语料库构建

将缺陷报告的问题概要描述(Summary)和问题详细描述(Description)抽取出来作为构建缺陷报告专用语料库的基础。因此,过滤掉问题概要描述或问题详细描述为空的缺陷报告。另外,因为本研究仅针对由英文撰写的缺陷报告进行实验,所以对使用其他语言撰写的报告,如阿拉伯文、韩文、日文等,予以删除。然后,对筛选后数据进行预处理。先将问题概要描述和问题详细描述字段分词,而后利用自然语言工具包(NLTK 库)对其中包含的 178 个停用单词(如“the”“and”“a”等)进行删除。这些词经常出现在语料库中,但通常对文本内容的理解贡献很少。接着,删除特殊的符号和数字,包括 HTML 标签。在此之后,进行词形还原操作,将单个单词转化成它们的根形式(例如,better→good,running→run)。最后,将这些处理好的问题概要描述和问题详细描述合并构建成缺陷报告的专用语料库。具体地,选用了 11 万多条

Eclipse 缺陷报告数据、30 万多条 Mozilla 缺陷报告数据和 8 万多条 Apache 缺陷报告数据构成了缺陷报告专用语料库。

3.3 微调 BERT 预训练模型

构建完缺陷报告专用语料库后，使用 NSP 预训练任务来微调通用 BERT 预训练模型，从而在应用阶段抽取语义特征。Masked Language Modeling（MLM）和 Next Sentence Prediction（NSP）是 BERT 的两种常用的预训练任务。对于 MLM 任务来说，其做法是随机掩盖掉输入序列中的 Token（即用“[MASK]”替换掉原有的 Token），然后预测对应位置的真实值。但由于语料库的特性，在微调时，输入序列中并不存在“[MASK]”这样的 Token，这将导致预训练和微调之间存在不匹配的问题。因此，在对 BERT 预训练模型进行微调时，选取了 NSP 作为预训练任务。在 NSP 任务里，是将每个样本视为由 A 和 B 两句话构成，通过预测句子 B 是否为句子 A 的下一句话来训练模型，从而提高模型对句子之间关系的理解能力。

使用 Pytorch 下载并调用 HuggingFace 上的通用 BERT 预训练模型和分词器。将专用语料库作为输入的训练文本，通过自定义函数将句子分解为独立的词元，处理过长的词元，并将句子划分为最大长度的块，然后获取每一行数据并将其处理为 BERT 模型需要的输入形式。这些输入将会通过 BERT 模型的内部逐步受到训练。首先是输入层（Embedding Layer），即嵌入层，它同时考虑了词位嵌入（Position Embeddings）、词形嵌入（WordPiece Embeddings）和段落嵌入（Segment Embeddings），将每个输入词元转换为一个维度为 768 的向量。接着是自注意力层（Self-Attention Layer），该层有 12 层，在每一层中，模型都会通过自注意力机制来理解文本中词元之间的依赖关系。自注意力可以让模型看到输入序列的全部内容，并根据需要关注序列中的任何部分。然后是前馈神经网络，该层由两个全连接层和一个 ReLU 激活函数组成。至于最后的输出层，因为我们使用的是 BERT 的 MLM 模型，所以最后一层是一个预测每个词元的层。模型会输出一个概率分布，表示每个可能的词元是被遮掩的词元的概率。

在 BERT 的训练参数上设定训练参数，包括学习率设为 1×10^{-7}、训练步数为 70000、批量大小为 8、权重衰减为 0.01 等。在训练后，选取效果最好的 BERT 预训练模型对缺陷报告抽取语义特征。

3.4 分类预测

根据 Eclipse，Mozilla 和 Apache 的缺陷追踪系统的配置，缺陷报告的严重性字段值包括 Blocker，Critical，Major，Minor，Trival 和 Enhancement。根据文献[17，23-24]，将缺陷报告分为两类，即严重和非严重。严重级别为 Blocker，Critical 和 Major 的缺陷报告被归入“严重”类别。严重性级别为 Minor 和 Trival 的缺陷报告则被认为是“不严重”类别。Enhancement 级别的缺陷报告被忽略，因为这些报告是对新功能的请求，所以不予考虑。在分类预测时，我们选取了 SVM 作为机器学习分类模型，因为在多项研究中[10-15]它都表现出色。使用微调的 BERT 预训练模型将缺陷报告抽取到的语义特征，即 768 维的向量作为支持向量机的输入，进行分类预测。值得注意的是，在进行数据处理时发现数据存在类别不平衡的问题。为了避免这种情况对分类结果产生影响，采用随机下采样的方法来处理数据。这种处理方法能够缓解数据不平衡所带来的问题，同时保持数据集的代表性和有效性，从而为分类预测任务提供更加可靠的数据基础，并使用十则交叉验证来保证实验结果的准确性和可靠性。

4 实验

4.1 实验数据集

本文在 3 个著名的开源软件上进行实验，即 Eclipse，Mozilla 和 Apache。使用 Python 的 Scrapy 库编写爬虫代码，并根据缺陷报告在缺陷追踪系统上的地址和相关参数构建对应的爬虫程序。在广泛使用并公开的缺陷追踪系统 Bugzilla 和 Jira 上爬取至 2022 年为止 Apache，Eclipse，Mozilla 的缺陷报告数据，爬取的字段包括不限于 Bug ID，Summary，Description，Status 等。实验数据集的具体信息如表 1 所列。

表 1 实验数据集

Table 1 Experimental datasets

Project	Product	Number of Bug Reports
Eclipse	Platform	121878
	JDT	62920
	Community	24591
	CDT	22205
	Mylyn	8863
Mozilla	Firefox	262038
	Firefox OS	73598
	Graveyard	
	DevTools	42091
	Testing	38122
	Thunderbird	32407
Apache	Tomcat	13494
	POI	4789
	JMeter	4776
	Fop	2086
	Log4j	1335

从每个开源软件中各选择具有代表性的 5 个项目，共计 15（5 * 3）个项目，其中一些经常在现有研究中使用[15，24-25]，它们来自不同的开源软件，具有不同的规模，这在一定程度上保证了我们的发现在实际中的可推广性。

4.2 研究问题

本文的实验希望解决以下几个研究问题。

问题 1 使用词嵌入技术是否能够捕获更多的语义信息？

为了回答这个问题，将通用 BERT 预训练模型与基于信息检索的方法 VSM 进行对比，比较两者在缺陷严重性预测任务上的效果。

问题 2 使用缺陷报告数据微调的 BERT 预训练模型是否更适用于该领域？

为了回答这个问题，将专用语料库微调的 BERT 预训练模型与通用 BERT 预训练模型进行比较，观察其在缺陷严重性预测任务上的效果。

问题 3 用专用语料库训练的 BERT 预训练模型效果如何？

使用特定领域数据对 BERT 预训练模型进行微调可以使模型更适用于该领域，那么完全用特定领域的数据训练一个 BERT 预训练模型，它在该领域上的效果会如何？由此，使用构建好的专用语料库重新训练了一个 BERT 预训练模型，并观察该模型在缺陷严重性预测任务上的效果。

4.3 评价指标

本文采用广泛使用的精确率(Precision)、召回率(Recall)和F1分数来衡量我们的模型在缺陷报告严重性预测任务中的表现，这3个值越大，说明缺陷严重性预测的效果越好。这3个指标的计算方法如下所示，其中S代表预测的类标签。

$$Precision=\frac{\text{被正确检测为 } S \text{ 类的缺陷报告数}}{\text{被检测为 } S \text{ 类的缺陷报告数}} \tag{1}$$

$$Recall=\frac{\text{被正确检测为 } S \text{ 类的缺陷报告数}}{S \text{ 类的缺陷报告数}} \tag{2}$$

$$F1=\frac{2\times Precision\times Recall}{Precision+Recall} \tag{3}$$

4.4 实验结果及分析

为了探究词嵌入技术BERT和缺陷报告专用语料库在抽取语义上的有效性，将其与以下方法进行实验比较。

(1)VSM：向量空间模型，传统基于信息检索的主流方法，通过TF-IDF算法实现。

(2)BERT：通用BERT预训练模型。

(3)BERTFT：使用缺陷报告专用语料库微调后的BERT预训练模型。

(4)BERTOwn：使用缺陷报告专用语料库重新训练后的BERT预训练模型。

在不改变参数和项目的情况下，本节使用4.2节中的评价指标展示实验结果，具体如下。

4.4.1 词嵌入BERT与传统方法VSM结果对比

如表2所列，使用词嵌入技术BERT比传统基于信息检索的技术VSM相比有明显提升。与VSM相比，BERT在Eclipse项目上的*Precision*平均提升了9.9%，*Recall*平均提升了9.8%，*F1*平均提升了10.0%；在Mozilla项目上，*Precision*平均提升了12.7%，*Recall*平均提升了12.7%，*F1*平均提升了12.9%；在Apache项目上，*Precision*平均提升了4.5%，*Recall*平均提升了4.4%，*F1*平均提升了4.5%。这些实验结果表明，相比传统的基于信息检索的方法，BERT能够更好地理解缺陷报告的语义和上下文关系，从而提高了预测性能。

表2 词嵌入BERT与传统方法VSM结果的对比

Table 2 Comparison of results between BERT and VSM

Project	Product	*Precision*		*Recall*		*F1_score*	
		VSM	BERT	VSM	BERT	VSM	BERT
Eclipse	CDT	0.62	**0.67**	0.62	**0.67**	0.62	**0.67**
	Community	0.48	**0.70**	0.48	**0.70**	0.48	**0.70**
	Mylyn	0.63	**0.68**	0.63	**0.68**	0.63	**0.68**
	PDE	0.63	**0.72**	0.63	**0.72**	0.63	**0.72**
	Platform	0.63	**0.73**	0.64	**0.72**	0.64	**0.73**
Mozilla	DevTools	0.64	**0.85**	0.64	**0.84**	0.63	**0.83**
	Firefox	0.60	**0.75**	0.59	**0.75**	0.57	**0.75**
	Firefox OS	0.53	**0.66**	0.53	**0.66**	0.53	**0.66**
	Graveyard	0.84	**0.94**	0.83	**0.94**	0.83	**0.93**
	Testing	0.69	**0.73**	0.69	**0.73**	0.69	**0.73**
	Thunderbird						
Apache	Fop	0.58	**0.58**	0.58	**0.58**	0.58	**0.58**
	JMeter	0.59	**0.65**	0.59	**0.65**	0.59	**0.65**
	Log4j	0.58	**0.61**	0.58	**0.61**	0.57	**0.61**
	POI	0.59	**0.66**	0.59	**0.66**	0.59	**0.66**
	Tomcat	0.60	**0.66**	0.60	**0.66**	0.60	**0.66**
	Fop	0.58	**0.58**	0.58	**0.58**	0.58	**0.58**

4.4.2 专用语料库微调对BERT的影响

如表3所列，使用缺陷报告专用语料库微调的BERT预训练模型(BERTFT)与通用BERT预训练模型(BERT)在缺陷严重性预测任务上进行比较。

表3 微调BERT预训练模型与通用BERT预训练模型的结果对比

Table 3 Performance comparison between fine-tuned BERT(BERTFT) and BERT pretrained model

Project	Product	*Precision*		*Recall*		*F1_score*	
		BERT	BERTFT	BERT	BERTFT	BERT	BERTFT
Eclipse	CDT	0.67	**0.70**	0.67	**0.69**	0.67	**0.70**
	Community	0.70	**0.72**	0.70	**0.72**	0.70	**0.72**
	Mylyn	0.68	**0.71**	0.68	**0.71**	0.68	**0.71**
	PDE	0.72	**0.74**	0.72	**0.74**	0.72	**0.73**
	Platform	0.73	0.73	0.72	**0.73**	0.73	0.73
Mozilla	DevTools	0.85	**0.86**	0.84	0.83	0.83	0.83
	Firefox	0.75	**0.77**	0.75	**0.77**	0.75	**0.77**
	Firefox OS	0.66	**0.71**	0.66	**0.70**	0.66	**0.71**
	Graveyard	0.94	**0.95**	0.94	**0.95**	0.93	**0.95**
	Testing	0.73	**0.76**	0.73	**0.76**	0.73	**0.76**
	Thunderbird						
Apache	Fop	0.58	**0.67**	0.58	**0.66**	0.58	**0.66**
	JMeter	0.65	**0.68**	0.65	**0.68**	0.65	**0.67**
	Log4j	0.61	0.66	0.61	**0.66**	0.61	**0.66**
	POI	0.66	**0.71**	0.66	**0.71**	0.66	**0.71**
	Tomcat	0.66	**0.71**	0.66	**0.71**	0.66	**0.71**
	Fop	0.58	**0.67**	0.58	**0.66**	0.58	**0.66**

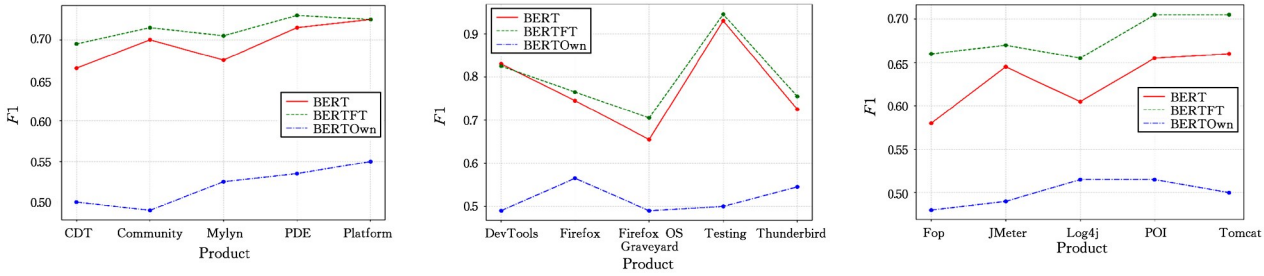
可以发现，微调的 BERT 预训练模型比通用 BERT 预训练模型效果有所提升。具体地，微调的 BERT 预训练模型在 Eclipse 项目上 *Precision* 提升了 1.5%～3.0%，*Recall* 提升了 0.5%～3.0%，*F*1 提升了 1.5%～3.0%；在 Mozilla 项目上，*Precision* 提升了 1.0%～5.0%，*Recall* 提升了 1.0%～4.5%，*F*1 提升了 1.5%～5.0%；在 Apache 项目上，*Precision* 提升了 3.0%～8.5%，*Recall* 提升了 3.0%～8.0%，*F*1 提升了 2.5%～8.0%。

从实验结果来看，在缺陷报告这一领域使用缺陷报告专用语料库对通用 BERT 预训练模型进行微调，能够提升模型对该领域的适应性，从而捕获缺陷报告更丰富的语义。但其提升效果有限，究其原因，有可能是因为：(1)在进行微调时，使用的仍旧是 BERT 通用分词器，如果使用专用语料库先训练出一个专用的分词器再进行微调，效果可能会有所提升；(2)超参设置，因为未对微调时的超参如学习率、损失函数、batch size 等进行调整，所以仍旧存在一个最佳超参设置使得微调的效果更好。

4.4.3 专用语料库训练 BERT 的效果

从问题 2 的实验结果可以发现缺陷报告专用语料库确实有一定的价值，于是我们使用通用 BERT 预训练模型的框架，在保证参数不变的情况下，以该专用语料库为训练数据训练出了一个专用 BERT 预训练模型(BERTOwn)。F1 值是同时考虑 Precision 和 Recall 的一种综合指标，可以用来评估模型的整体性能，所以图 3 仅在 F1 指标下直观展示了该专用 BERT 预训练模型与通用 BERT 预训练模型、微调的 BERT 预训练模型在缺陷报告严重性预测任务上的效果。可以发现，从 F1 来看，使用专用 BERT 预训练模型比通用 BERT 预训练模型略差，差距为 12.9%～25.9%；与微调的 BERT 预训练模型相比也有 17.9%～28.1%的差距。

换言之，仅使用缺陷报告专用语料库训练的专用 BERT 预训练模型效果较为不佳。经过研究分析，我们认为相对于通用 BERT 预训练模型的训练数据——BERT 预训练模型在 Book Corpus 和维基百科数据上进行预训练。Book Corpus 是一个由 11038 本未出版的书籍和英文维基百科(不包括列表、表格和标题)组成的数据集来说，我们专用数据集相对较小，这可能是效果相差甚远的原因。因此，可以加强缺陷报告数据的收集，扩大训练数据集，或者提高训练的深度，增强其学习能力。



(a)Eclipse (b)Mozilla (c)Apache

图 3 专用 BERT、微调的 BERT 和通用 BERT 预训练模型的结果对比

Fig. 3 Performance comparison between domain-specific BERT (BERTOwn), fine-tuned BERT(BERTFT) and BERT pretrained Model

5 效度威胁讨论

本节将从如下 2 个方面讨论影响本文实验结果的有效性和潜在威胁。

(1)本文的对比实验仅与传统信息检索方法 VSM 对比。不能保证我们得出的结论可以适用于所有领域。但是，考虑到当前研究领域中对缺陷报告抽取语义的主流方式仍是信息检索和词嵌入技术，我们的发现仍然有一定的价值。我们鼓励未来的研究可以与其他领域的方法进行对比，如使用深度学习方法探究。

(2)本文实验只在大型开源软件的缺陷报告上进行了实验，我们不能保证得出的结论可以适用于其他小项目、非开源项目或商业项目。但是，考虑到所有选定的实验项目都是正在积极开发的典型软件产品，我们的发现仍然有一定的意义。我们鼓励未来的研究来复制我们的研究对来自其他项目的缺陷报告进行研究，以进一步提高结论的普遍性。

结束语 本文提出了一种基于领域知识微调的缺陷报告严重性预测方法，增强了对缺陷报告的语义信息捕获，提高了预测准确性。本文研究证明使用 BERT 词嵌入方法比基于信息检索的方法表现更优，使用缺陷报告专用语料库微调的 BERT 预训练模型与通用 BERT 预训练模型相比效果也有所提高，使用缺陷报告专用语料库的领域知识对 BERT 预训练模型进行微调能够提高缺陷报告严重性预测的效果，未来可探究参数设置并考虑更多数据集实验。

参 考 文 献

[1] BETTENBURGN, JUST S, SCHRÖTER A, et al. What makes a good bug report? [C]// Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering. 2008: 308-318.

[2] ANVIK J, HIEW L, MURPHY G C. Who should fix thisbug? [C]// Proceedings of the 28th international conference on Software engineering. 2006: 361-370.

[3] ZOU W, LO D, CHEN Z, et al. How practitioners perceive automated bug report management techniques[J]. IEEE Transactions on Software Engineering, 2018, 46(8): 836-862.

[4] TAN Y, XU S, WANG Z, et al. Bug severity prediction using question-and-answer pairs from Stack Overflow[J]. Journal of Systems and Software, 2020, 165: 110567.

[5] SANTOS K, DIAS J P, AMADO C. A literature review of machine learning algorithms for crash injury severity prediction [J]. Journal of Safety Research, 2022, 80: 254-269.

[6] LUAPHOL B, POLPINIJ J, KAENAMPORNPAN M. Text Mining Approaches for Dependent Bug Report Assembly and

Severity Prediction[J]. Internarional Arab Journal of Information Technology,2022,19(6):915-924.

[7] TIAN Y,LO D,SUN C. Information retrieval based nearest neighbor classification for fine-grained bug severity prediction [C]// 2012 19th Working Conference on Reverse Engineering. IEEE,2012:215-224.

[8] YANG G,ZHANG T,LEE B. Towards semi-automatic bug triage and severity prediction based on topic model and multi-feature of bug reports[C]// 2014 IEEE 38th Annual Computer Software and Applications Conference. IEEE,2014:97-106.

[9] ROY N K S,ROSSI B. Towards an improvement of bug severity classification[C]// 2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications. IEEE,2014:269-276.

[10] MALHOTRA R,KAPOOR N,JAIN R,et al. Severity assessment of softwarebug reports using text classification[J]. International Journal of Computer Applications,2013,83(11):13-16.

[11] YANG C Z,HOU C C,KAO W C,et al. An empirical study on improving severity prediction of defect reports using feature selection[C]// 2012 19th Asia-Pacific Software Engineering Conference. IEEE,2012,1:240-249.

[12] RAMAY W Y,UMER Q,YIN X C,et al. Deep neural network-based severity prediction of bug reports[J]. IEEE Access,2019,7:46846-46857.

[13] JIA Y,CHEN X,XU S,et al. EKD-BSP:bug report severity prediction by extracting keywords from description[C]// 2021 8th International Conference on Dependable Systems and Their Applications. IEEE,2021:42-53.

[14] SU Y,HU X,CHEN X,et al. CIL-BSP:Bug Report Severity Prediction based on Class Imbalanced Learning[C]// 2022 IEEE 22nd International Conference on Software Quality,Reliability,and Security Companion. IEEE,2022:298-306.

[15] KUMAR L,KUMAR M,MURTHY L B,et al. An empirical study on application of word embedding techniques for prediction of software defect severity level[C]// 2021 16th Conference on Computer Science and Intelligence Systems. IEEE,2021:477-484.

[16] MENZIES T,MARCUS A. Automated severity assessment of software defect reports[C]// 2008 IEEE International Conference on Software Maintenance. IEEE,2008:346-355.

[17] LAMKANFI A,DEMEYER S,GIGER E,et al. Predicting the severity of a reported bug[C]// 2010 7th IEEE Working Conference on Mining Software Repositories. IEEE,2010:1-10.

[18] SARI G I P,SIAHAAN D O. An attribute selection for severity level determination according to the support vector machine classification result[C]// Proceedings of the 1st International Conference on Information Systems for Business Competitiveness. 2011.

[19] JINDAL R,MALHOTRA R,JAIN A. Software defect prediction using neural networks[C]// Proceedings of 3rd International Conference on Reliability,Infocom Technologies and Optimization. IEEE,2014:1-6.

[20] YANG G,ZHANG T,LEE B. Towards semi-automatic bug triage and severity prediction based on topic model and multi-feature of bug reports[C]// 2014 IEEE 38th Annual Computer Software and Applications Conference. IEEE,2014:97-106.

[21] JINDAL R,MALHOTRA R,JAIN A. Prediction of defect severity by mining software project reports[J]. International Journal of System Assurance Engineering and Management,2017,8:334-351.

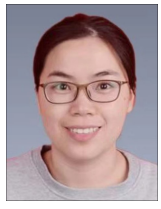
[22] DEVLIN J,CHANG M W,LEE K,et al. Bert:Pre-training of deep bidirectional transformers for language understanding[J]. arXiv:1810.04805,2018.

[23] LAMKANFI A,DEMEYER S,SOETENS Q D,et al. Comparing mining algorithms for predicting the severity of a reported bug[C]// 2011 15th European Conference on Software Maintenance and Reengineering. IEEE,2011:249-258.

[24] TIAN Y,LO D,XIA X,et al. Automated prediction of bug report priority using multi-factor analysis[J]. Empirical Software Engineering,2015,20:1354-1383.

[25] VAN NGUYEN T,NGUYEN A T,PHAN H D,et al. Combining word2vec with revised vector space model for better code retrieval[C]// 2017 IEEE/ACM 39th International Conference on Software Engineering Companion. IEEE,2017:183-185.

CHEN Bingting, born in 1998, postgraduate. Her main research interests include bug repository mining and so on.



ZOU Weiqin, born in 1988, Ph.D, professor, is a member of CCF (No. D3300M). Her main research interests include bug localization and software repository mining.