基于 Box-Cox 转换的集成跨项目软件缺陷预测方法*

王莉萍[1]，陈 翔[1,2†]，王秋萍[1]，赵英全[1]
(1.南通大学 计算机科学与技术学院，江苏 南通 226019; 2.南京大学 软件新技术国家重点实验室，南京 210093)

摘 要： 对跨项目缺陷预测问题展开了深入研究，在源项目实例选择时，考虑了三种不同的实例相似度计算方法，并发现这些方法的缺陷预测结果存在多样性，因此提出了一种基于 Box-Cox 转换的集成跨项目软件缺陷预测方法 BCEL。具体来说，基于不同的实例相似度计算方法，从候选集中选出不同的训练集；针对这些数据集，进行有针对性的 Box-Cox 转换，并借助特定分类方法构造出不同的基分类器，最后将这三个基分类器进行有效集成。基于实际项目的数据集，验证了 BCEL 方法的有效性，并深入分析了 BCEL 方法内的影响因素对缺陷预测性能的影响。

关键词： 软件缺陷预测；跨项目软件缺陷预测；集成学习；实证研究
中图分类号： TP311.5 **文献标志码：** A **文章编号：** 1001-3695(2017)07-2023-04
doi:10.3969/j.issn.1001-3695.2017.07.023

Box-Cox transformation based ensemble learning approach for cross-project software defect prediction

Wang Liping[1], Chen Xiang[1,2†], Wang Qiuping[1], Zhao Yingquan[1]
(1.*School of Computer Science & Technology, Nantong University, Nantong Jiangsu* 226019, *China*; 2.*State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing* 210093, *China*)

Abstract: This paper explored the issue of cross-project defect prediction in depth. In instance selection from source projects, it considered three different instance similarity calculation methods and found that the prediction results were diversity. Therefore this paper proposed a Box-Cox transformation based ensemble learning approach named BCEL. In particular, it selected different training sets based on different instance similarity calculation methods. Then it performed Box-Cox transformation based on these datasets and used a specific classifier to train these base classifiers. Finally it used an ensemble learning method to combine these base classifiers. Based on datasets from real software projects, it verifies the effectiveness of this proposed BCEL approach and analyzes the effect of different influencing factors in BCEL.

Key words: software defect prediction; cross-project software defect prediction; ensemble learning; empirical studies

0 引言

软件在人们的日常生活中无处不在，而隐含缺陷的软件在部署后可能会产生意料之外的结果，甚至有时会给企业带来巨大的损失。软件缺陷预测[1]通过预先识别出项目内的潜在缺陷程序模块，可针对这些模块投入足够的测试资源进行软件测试或代码审查，以确保软件产品的质量。

软件缺陷预测通过挖掘软件历史库(software historical repository)，从中抽取出程序模块并进行类型标记，程序模块的粒度可以设置为包、类或函数等，随后通过分析代码复杂度或软件开发过程，设计出与软件缺陷存在强相关性的度量元(metrics)，并借助这些度量元对已抽取的程序模块依次进行软件度量，旨在构建出缺陷预测数据集，最后基于特定的机器学习方法(如 Logistic 回归、决策树、支持向量机等)构建出缺陷预测模型，并用于对项目内新的程序模块进行预测，其预测目标可以是模块内是否含有缺陷、含有的缺陷数量或缺陷密度等。

但目前大部分研究工作都关注集中于同项目缺陷预测(within-project defect prediction，WPDP)，即选择来自同一项目的部分数据来构建模型，并用剩余未选择数据来评估模型的性能。但在实际的软件开发场景中，需要进行缺陷预测的目标项目可能已有的训练数据较为稀缺，或者可能是一个新启动项目。因此一种简单方法是使用其他项目已经搜集的高质量数据集来为目标项目构建缺陷预测模型。但不同项目间的特征(如开发流程、使用的编程语言或开发人员的经验等)并不相同，由此造成源项目与目标项目的数据集间存在较大的取值分布差异，因此如何从源项目中迁移出与目标项目相关的知识，吸引了国内外研究人员的关注，并称该问题为跨项目软件缺陷预测(cross-project defect prediction，CPDP)。一些研究人员对 CPDP 的可行性展开了调研，Zimmermann 等人[2]考虑了 12 个大规模项目，累计分析了 622 对 CPDP，他们发现仅有 21 对可以取得满意的缺陷预测性能。He 等人[3]对上述结论进行了确认。但 Rahman 等人[4]从成本收益角度出发，却发现 CPDP 的

收稿日期：2016-09-21；修回日期：2016-10-31 基金项目：国家自然科学基金资助项目(61202006)；南京大学计算机软件新技术国家重点实验室开放课题(KFKT2016B18)；江苏省大学生创新训练计划项目(201610304090X)；江苏省高校自然科学研究项目(15KJB520030，16KJB520038)

作者简介：王莉萍(1992-)，女，硕士研究生，主要研究方向为软件缺陷预测；陈翔(1980-)，男(通信作者)，副教授，博士，主要研究方向为软件缺陷预测、软件缺陷定位和回归测试等(xchencs@ntu.edu.cn)；王秋萍(1993-)，女，硕士研究生，主要研究方向为软件缺陷预测；赵英全(1994-)，男，本科生，主要研究方向为软件缺陷预测.

性能并不一定比 WPDP 的性能差，并且要显著优于随机预测模型。

研究人员提出了多种方法来提高 CPDP 的性能。Turhan 等人[5] 提出的 Burak 过滤法和 Peters 等人[6] 提出的 Peters 过滤法尝试从源项目中选出相关实例。Ma 等人[7] 从为源项目的实例权重设置出发，提出了 TNB 方法。Chen 等人[8] 从识别并移除源项目内的负面实例(negative instances) 出发，提出了 DTB 方法。Nam 等人[9] 借助特征映射，提出 TCA + 方法。Panichella 等人[10] 深入分析了基于主成分分析对六种不同分类方法的等价性，发现不同分类方法预测出的有缺陷模块并不完全等同，提出了一种基于集成学习的方法 CODEP。Wang 等人[11] 考虑了表示学习算法，认为基于表示学习分析出的语义特征，可以更好地捕获缺陷的共有特征，并将该方法用于 CP-DP。Canfora 等人[12] 将 CPDP 问题建模为多目标优化问题，主要考虑了两个可能存在冲突的优化目标: 尽可能多地识别出被测项目内的缺陷数和尽可能少的代码审查量，提出了 MODEP 方法，该方法使用了一种经典的多目标遗传算法 NSGA-Ⅱ。

但在实际应用中，源项目与目标项目采用的度量元并不完全相同，研究人员将该问题称为异构跨项目缺陷预测[13] 并展开了研究。Nam 等人[13] 提出了 HDP 方法，该方法包括特征选择和特征映射两个阶段。Jing 等人[14] 针对该问题，提出一种 UMR 表示，随后使用了典型相关分析方法。

有的研究人员则尝试通过仅关注目标项目中的未标记数据，借助无监督学习方法来缓解 CPDP 问题。这类研究工作均基于如下重要假设[15,16]: 在软件缺陷预测问题中，有缺陷模块的度量元取值存在高于无缺陷模块的度量元取值的倾向。基于上述假设，Nam 等人[15] 提出了一种自动方法 CLA 和 CLA-MI。Zhang 等人[16] 则考虑了谱聚类(spectral clustering) 方法。

与上述研究工作不同，本文从源项目实例选择角度出发进行了深入研究，在 Burak 过滤法和 Peters 过滤法基础上，考虑了更多的实例相似度计算公式(即欧氏距离、余弦相似度和相关系数) ，并发现这三种不同的相似度计算公式在跨项目缺陷预测时具有多样性。基于上述观察，本文提出了一种集成跨项目缺陷预测方法 BCEL，并对该方法的内在影响因素(即是否使用 Box-Cox 转换、源项目实例选择方法以及分类方法) 进行了深入的分析。

1 BCEL 方法

BCEL 方法的整体框架如图 1 所示，其主要包括实例选择、度量元取值转换、基分类器构建和集成学习四个阶段。下面将对每个阶段的技术细节依次进行阐述。

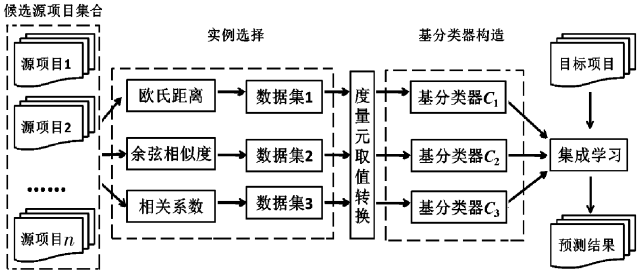

图 1 BCEL 方法的框架图

1.1 实例选择阶段

本文在实例选择阶段主要考虑的是多对一模式，即将所有的候选源项目的数据集汇总为一个候选训练集。但这种模式也会引入一些无关实例，并造成模型的预测性能不理想。Burak 过滤法[5] 和 Peters 过滤法[6] 是两种经典的实例选择方法。其中 Burak 过滤法首先计算出目标项目中实例与候选训练集中实例之间的相似度，随后依次为目标项目中的每一个实例，从候选训练集中选出距离最近的 k($k=10$) 个实例并添加到最终的训练集中。假设目标项目中含有 N 个实例，则最终会选出 $k\times N$ 个实例，但候选训练集中可能会存在一个实例被多次选中，则最终仅选择一次。

Turhan 等人从目标项目出发进行实例选择，该方法假设目标项目含有足够的实例。而 Peters 等人[6] 则认为候选训练集中包含的信息更多(即若考虑了很多候选源项目，则含有的实例数更多，因此与缺陷相关的信息也会更多) ，因此他们提出了 Peters 过滤法。具体来说: 首先针对候选训练集中的每个实例，从目标项目中识别出与之相似度最高的实例并进行标记; 随后对于目标项目中已标记的实例，从候选训练集中选出与之相似度最高的实例并添加到最终的训练集中。

但 Burak 过滤法和 Peters 过滤法仅采用了欧氏距离(Euclidean distance) 来计算实例的相似度，本文则进一步考虑了其他两种相似度计算方法: 余弦相似度(cosine similarity) 和相关系数(correlation coefficient) 。假设目标项目中的某个实例 x 用向量(x_1，x_2，…，x_n) 表示，目标项目中的某个实例 y 用向量(y_1，y_2，…，y_n) 表示，则这三种距离公式可依次定义如下:

a) 欧氏距离，其取值越小越好，对应的计算式为

$$\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2} \tag{1}$$

b) 余弦相似度，其取值介于 $-1\sim1$，取值越趋近于 1，则代表它们之间的相似度越高，对应的计算式为

$$\frac{\sum_{i=1}^{n}(x_i\times y_i)}{\sqrt{\sum_{i=1}^{n}x_i^2}\sqrt{\sum_{i=1}^{n}y_i^2}} \tag{2}$$

c) 相关系数，其取值也介于 $-1\sim1$，取值越趋近于 1，则代表它们之间的相似度越高，其对应的计算式为

$$\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}} \tag{3}$$

其中: $\bar{x}$ 和 $\bar{y}$ 分别为向量 x 和向量 y 的均值。

1.2 度量元取值转换阶段

已有研究表明，缺陷预测数据集中，大部分度量元的取值分布很难满足正态分布，更多时候是呈幂律分布，因此会降低缺陷预测方法的性能。目前一般借助取对数转换，来使得度量元的取值与正态分布更为接近。在 BCEL 方法中，本文则重点考虑了 Box-Cox 转换[17] 。假设度量元取值为 s，则 Box-Cox 转换借助式(4) 进行取值转换。

$$\bar{s}=\begin{cases}(s^{\lambda}-1)/\lambda & \lambda\neq0\\ \ln(s+1) & \lambda=0\end{cases} \tag{4}$$

其中: λ 是 Box-Cox 转换中的配置参数，不难看出 Box-Cox 转换代表一系列取值转换方法。例如，若 λ 取值为 1，则不进行取值转换; 若 λ 取值为 0，则进行取对数转换; 若 λ 取值为 0.5，则进行平方根转换; 若 λ 取值为 -1，则进行取倒数转换。其通过测试集和训练集估算出最优 λ 值。目前有两个指标可用于评估度量元的取值是否呈正态分布，分别是偏度(skewness) 和峰度(kurtosis) ，其中偏度是对取值分布的偏斜方向和程度的度

量，其正常取值为-0.8～0.8，而峰度则反映了峰部的尖度，其理想取值是 0。

给定源项目和目标项目，本文通过从候选值集合{-1，-0.9，…，0.9，1}中依次取值并赋值给 λ，随后同时对源项目和目标项目进行 Box-Cox 转换，并计算出源项目和目标项目中各个度量元的偏度均值，最后选出可以产生最优偏度的 λ 值。

1.3　基分类器构造阶段

给定目标项目，在第一阶段通过考虑三种不同的实例相似度计算方法，可以从候选训练集中选出三个不同的训练集；随后针对不同的目标项目数据集和训练集，进行有针对性的 Box-Cox 转换；最后借助指定的分类方法（如 Logistic 回归）可以构造出三个不同的基分类器 $C=\{C_1,C_2,C_3\}$。其中 C_1 是基于欧氏距离构建出的基分类器；C_2 是基于余弦相似度构建出的基分类器；而 C_3 是基于相关系数构建出的基分类器。

1.4　集成学习阶段

若第三阶段训练出的三个基分类器，其缺陷预测结果具有多样性（即他们预测出的缺陷模块并不完全相同），则可以考虑借助集成学习方法来提升 CPDP 性能。本文考虑了如下的集成学习方法。给定需要预测的程序模块 m，假设第 i 个基分类器 C_i 预测其内部含有缺陷的概率为 $p(C_i,m)$，若取值大于 0.5，则将该模块预测为有缺陷模块，否则预测为无缺陷模块。若在 C 中至少存在一个基分类器将 m 预测为有缺陷模块，则集成分类器 E 借助如下公式计算出最终的预测概率：

$$p(E,m)=\min\{1,\frac{\sum_{C_i\in C\wedge p(C_i,m)>0.5} w_i\times p(C_i,m)}{|\{C_i\mid C_i\in C\wedge p(C_i,m)>0.5\}|}\} \tag{5}$$

否则借助如下公式计算出最终的预测概率：

$$p(E,m)=\max\{0,1-\frac{\sum_{C_i\in C} w_i\times(1-p(C_i,m))}{3}\} \tag{6}$$

其中：w_i 根据基分类器在各自训练集上的准确率来确定，即为准确率更高的分类器设置更高的权重。假设 C_i 在对应的训练集上的准确率为 a_i，则 $w_i=a_i/\min_a$，其中 $\min_a=\min\{a_1,a_2,a_3\}$。

式（5）中 min 函数的作用是将最终预测出的概率值的最大值限定为 1，式（6）中 max 函数的作用是将最终预测出的概率值的最小值限定为 0。

2　实证研究

本章对 BCEL 方法的有效性进行实证研究，下面是实证研究中需要回答的两个实验问题。

RQ1　分析本文考虑的三种不同的实例相似度计算方法在缺陷预测时，是否具有多样性？若具有多样性，则本文提出的 BCEL 方法是否能够提高 CPDP 的性能？

RQ2　分析 BCEL 方法内的影响因素对预测结果是否存在影响？其中重点分析的影响因素包括实例选择方法的设定、是否进行 Box-Cox 转换以及分类方法的设定。

2.1　评测数据集

本文在实证研究中考虑了 AEEEM 数据集[18]，该数据集由 D'Ambros 等人搜集，并经常被用于 CPDP 的研究中[9,13,16]。该评测数据集考虑的度量元主要基于代码修改特征、历史缺陷检测信息、模块复杂度等。表 1 列出了数据集中的度量元简称和具体含义。AEEEM 评测数据集的统计特征如表 2 所示，包括项目名称、模块数以及缺陷模块数及所占比例。

表 1　AEEEM 数据集的度量元简称及含义

简　称	含　义
CBO	coupling between objects
FanOut	number of other classes referenced by the class
NOA	number of attributes
NOAI	number of attributes inherited
NOLOC	number of lines of code
NOPRM	number of private attributes
NOPRM	number of private methods
RFC	response for class
WMC	weighted method count
LAU	lines added until
AAU	avgLines added until
NOBFU	number of bugs found until

表 2　AEEEM 数据集的统计特征

项目名称	模块数	缺陷模块数（比例/%）	项目名称	模块数	缺陷模块数（比例/%）
Eclipse	997	206（20.7）	Mylyn	1 862	245（13.2）
Equinox	324	129（39.8）	PDE	1 497	209（14.0）
Lucene	691	64（9.3）			

2.2　评测指标

跨项目缺陷预测问题可视为二分类问题。若将有缺陷模块设为正例，无缺陷模块设置为反例，则可以将程序模块根据其真实类型与模型的预测类型的组合划分为真正例（true positive）、假正例（false positive）、真反例（true negative）和假反例（false negative）这四种情形。令 TP、FP、TN、FN 分别表示对应的模块数。其中查准率指标返回的是预测为有缺陷的模块中，真实缺陷模块所占的比例，其计算式为

$$\text{precision}=\frac{\text{TP}}{\text{TP}+\text{FP}} \tag{7}$$

查全率返回的是所有缺陷模块中，被预测为有缺陷模块所占的比例，其计算式为

$$\text{recall}=\frac{\text{TP}}{\text{TP}+\text{FN}} \tag{8}$$

F-measure 指标是查准率和查全率的调和平均数，可以对查准率和查全率这两个指标进行有效的平衡，因此本文在评估模型性能的时候选择采用该指标，其计算式为

$$F\text{-measure}=\frac{2\times\text{precision}\times\text{recall}}{\text{precision}+\text{recall}} \tag{9}$$

目前该评测指标也被常用于 CPDP 的研究中[3,4,7,9,11,15,16]。

2.3　实验细节及结果分析与讨论

本文基于 Weka 软件包和 R 软件包综合实现了 Burak 过滤法、Peters 过滤法以及本文提出的 BCEL 方法。在跨项目缺陷预测时，若选择 AEEEM 数据集中的某一项目为目标项目时，则其他四个项目为候选源项目集。针对 RQ1，本文首先将分类方法设置为 Logistic 回归，该方法是目前 CPDP 研究中使用最多的一种方法[3,4,9,10,12,13]，因此具有一定的代表性。本文将仅基于欧氏距离的实例选择方法称为 ED 方法，将仅基于余弦相似度的实例选择方法称为 CS 方法，将仅基于相关系数的方法称为 CC 方法。BCEL、ED、CS 和 CC 方法在实例选择时均基于 Burak 过滤法，并均进行 Box-Cox 转换。

a）借助 McNemar 检验来分析不同基分类器的预测结果是否具有多样性。McNemar 属于非参数统计检验方法，其可以用于分析两种不同的分类器的预测结果是否具有多样性。将可信度设为 95%（即 $p<0.05$），并将原假设设定为不同基分类器在目标项目上预测出的缺陷模块相似。若 McNemar 检验的 p 值小于 0.05，则拒绝原假设。最终结果如表 3 所示，并对小于 0.05 的 p 值进行了加粗。

表 3 McNemar 检验的 p 值

目标项目	C_1 vs C_2	C_1 vs C_3	C_2 vs C_3
Eclipse	2.47e-05	4.05e-06	0.11
Equinox	1	0.40	0.22
Lucene	0.06	0.02	0.51
Mylyn	2.25e-04	2.45e-07	0.16
PDE	2.18e-13	2.24e-07	1.47e-06

结果表明,大部分情况下(即 8/15),不同基分类器预测出的缺陷程序模块并不相同,尤其是 C_1 与 C_2,C_1 与 C_3 之间。

b) 分析本文提出的 BCEL 方法与其他方法(即 ED、CS 和 CC 方法)相比,是否能够提高 CPDP 性能,最终结果如表 4 所示,并对其中的最优值进行了加粗。

表 4 不同方法之间的性能比较

目标项目	ED	CS	CC	BCEL
Eclipse	0.442	0.428	**0.461**	0.457
Equinox	0.196	0.235	0.179	**0.444**
Lucene	0.260	0.270	0.277	**0.297**
Mylyn	0.143	0.203	0.182	**0.225**
PDE	0.216	0.281	0.276	**0.283**
均值	0.251	0.283	0.275	**0.341**

从表 4 中不难看出,大部分情况下,本文提出的 BCEL 方法可以取得最好的 CPDP 性能,虽然在预测 Eclipse 项目时并未取得最好的性能,但仍排名第二,且与排名最好的 CC 方法的预测性能非常接近。总体来说,BCEL 方法相对于 ED 方法有 35.9%的提升;相对于 CS 方法有 20.5%的提升;相对于 CC 方法有 24%的提升。针对 RQ2,本文重点分析 BCEL 方法内的影响因素对预测性能的影响。本文重点考虑的影响因素包括是否进行 Box-Cox 转换、实例选择方法的设定以及分类方法的设定。

a) 分析在 BCEL 方法中,借助 Box-Cox 转换是否可以有效提高 BCEL 方法的性能。在 BCEL 方法的执行时,以 Burak 过滤法作为实例选择方法,以 Logistic 回归作为分类方法,将分别进行 Box-Cox 转换和未进行 Box-Cox 转换的预测结果,用盒图进行表示,最终结果如图 2 所示。结果表明 Box-Cox 转换可以提高 BCEL 方法的性能。

b) 分析不同实例选择方法(即 Burak 过滤法和 Peters 过滤法),对 BCEL 方法的影响。同样 BCEL 方法在执行时,基于 Logistic 回归,采用 Box-Cox 转换,将分别基于 Burak 过滤法和 Peters 过滤法的预测结果,用盒图进行了表示,最终结果如图 3 所示。结果表明 Burak 过滤法的性能要优于 Peters 过滤法。

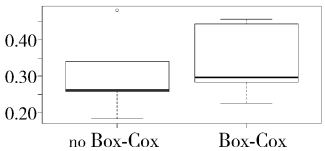

图 2 Box-Cox 转换对 BCEL 方法性能的影响

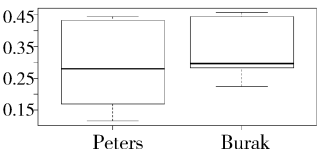

图 3 实例选择方法对 BCEL 方法性能的影响

最后分析不同的分类方法对 BCEL 方法的影响。其中 BCEL 方法在执行时考虑了 Box-Cox 转换,在实例选择时考虑的是 Burak 过滤法。本文额外考虑了朴素贝叶斯(naïve Bayes)、K 最近邻(K nearest neighbor)、决策树(decision tree)、随机森林(random forest)、多层感知器(multi-layer perception)和随机树(random tree)等分类方法。这些分类方法覆盖了不同类型的方法,均基于 Weka 软件包实现,并采用了分类方法的默认参数取值。最终结果如表 5 所示,并对其中的最优值进行了加粗。不难看出大部分情况下,本文提出的 BCEL 方法均可以有效提高 CPDP 的性能。总体来说,本文提出的 BCEL 方法相对于 ED 方法有 22.4%的提升,相对于 CS 方法有 15.6%的提升,相对于 CC 方法有 10.6%的提升。

表 5 不同分类方法下的性能比较

分类方法	ED	CS	CC	BCEL
Logistic 回归	0.251	0.283	0.275	**0.341**
朴素贝叶斯	**0.421**	0.403	0.397	0.402
K 最近邻	0.301	0.295	0.303	**0.332**
决策树	0.213	0.296	0.308	**0.348**
随机森林	0.258	0.315	0.326	**0.349**
多层感知器	0.270	0.243	0.294	**0.337**
随机树	0.282	0.280	0.306	**0.335**
均值	0.285	0.302	0.316	**0.349**

2.4 有效性影响因素分析

本节主要讨论可能影响实证研究有效性的一些影响因素。外部有效性主要涉及到实证研究结论是否具有一般性。本文选用了 CPDP 研究中经常使用的 AEEEM 数据集[9,13,16],该数据集可以在 PROMISE 库(http://openscience.us/repo/)中下载,因此可以有效保证研究结论的代表性。内部有效性主要涉及到可能影响到实验结果正确性的内部因素。本文编写的代码主要基于 Weka 软件包和 R 软件包,因此可以保证分类器实现的正确性,除此之外,本文还通过一些简单实例对方法实现是否正确进行了验证。结论有效性是指本文采用的评测指标是否合理。本文主要考虑了 F-measure[3,4,7,9,11,15,16] 指标,该指标是当前 CPDP 研究中的一个重要指标并被研究人员广泛使用。

3 结束语

本文通过考虑更多实例相似度计算方法,提出了一种基于 Box-Cox 转换的集成跨项目软件缺陷预测方法 BCEL,并基于 AEEEM 数据集对该方法的有效性进行了验证,随后深入探讨了方法内的影响因素对 CPDP 预测性能的影响。该方法仍然存在一些后续研究工作,具体来说:a) 将尝试基于其他数据集,对本文结论的一般性进行验证;b) 尝试其他集成学习方法,来对 BCEL 方法进行优化;c) 尝试考虑特征选择方法[19~23],通过移除数据集中的冗余特征和无关特征,来进一步提升 CPDP 性能。

参考文献:

[1] 陈翔,顾庆,刘望舒,等.静态软件缺陷预测方法研究[J].软件学报,2016,27(1):1-25.

[2] Zimmermann T, Nagappan N, Gall H, *et al*. Cross-project defect prediction: a large scale experiment on data vs. domain vs. process [C]//Proc of Joint Meeting of the European Software Engineering Conference and International Symposium on the Foundations of Software Engineering. 2009: 91-100.

[3] He Zhimin, Shu Fengdi, Yang Ye, *et al*. An investigation on the feasibility of cross-project defect prediction [J]. Automated Software Engineering, 2012, 19(2): 167-199.

[4] Rahman F, Posnett D, Devanbu P. Recalling the "imprecision" of cross-project defect prediction [C]//Proc of International Symposium on the Foundations of Software Engineering. 2012: 61:1-61:11.

[5] Turhan B, Menzies T, Bener A B, *et al*. On the relative value of cross-company and within-company data for defect prediction [J]. Empirical Software Engineering, 2009, 14(5): 540-578.

(下转第 2031 页)

其中分类器预测性能评价指标采用 F-measure 以及 G-mean。从实验结果可以看出，本文提出的基于分布的过抽样+随机向下欠抽样处理数据不平衡效果更好，尤其是 PC1 数据集，G-mean 值达到了 90%以上。

参考文献：

[1] Catal C. Performance evaluation metrics for software fault prediction studies [J]. Acta Polytechnica Hungarica, 2012, 9(4): 193-206.

[2] 杨晓杏. 基于度量元的软件缺陷预测技术 [D]. 合肥：中国科学技术大学，2014.

[3] 李雄飞，李军，董元方，等. 一种新的不平衡数据学习算法 PCBoost [J]. 计算机学报，2012，35(2)：202-209.

[4] Tomek I. Two modifications of CNN [J]. IEEE Trans on Systems Man and Communications, 1976, 6(11): 769-772.

[5] 李元菊. 数据不平衡分类研究综述 [J]. 现代计算机，2016(4)：30-33.

[6] Chawla N V, Bowyer K W, Hall L O, *et al*. SMOTE: synthetic minority over-sampling technique [J]. Journal of Artificial Intelligence Research, 2011, 16(1): 321-357.

[7] 董燕杰. 不平衡数据集分类的 Random-SMOTE 方法研究 [D]. 大连：大连理工大学，2009.

[8] Han Hui, Wang Wenyuan, Mao Binghuan. Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning [C] //Proc of International Conference on Advances in Intelligent Computing. Berlin: Springer-Verlag, 2005: 878-887.

[9] 刘胥影，吴建鑫，周志华. 一种基于级联模型的类别不平衡数据分类方法 [J]. 南京大学学报：自然科学版，2006，42(2)：148-155.

[10] 李想. 基于均衡采样方法的数据不平衡问题研究 [D]. 大连：大连理工大学，2014.

[11] Hart P E. The condensed nearest neighbor rule [J]. IEEE Trans on Information Theory, 1968, 14(3): 515-516.

[12] Tomek I. Two modification of CNN [J]. IEEE Trans on Systems, Man and Communications, 1976, 6(11): 769-772.

[13] Laurikkala J. Improving identifieation of difficult small classes by balancing class distribution [C] //Proc of the 8th Conference on AI in Medicine. Berlin: Springer, 2001: 63-66.

[14] Batista G E A P A, Prati R C, Monard M C. A study of the behavior of several methods for balancing machine learning training data [J]. ACM SIGKDD Explorations Newsletter, 2004, 6(1): 20-29.

[15] Fan Wei, Stolfo S J, Zhang Junxin, *et al*. AdaCost: misclassification cost-sensitive boosting [C] //Proc of the 16th International Conference on Machine Learning. San Francisco: Morgan Kaufmann, 1999: 97-105.

[16] Joshi M V, Kumar V, Agarwal R C. Evaluating [C] //Proc of IEEE International Conference on Data Mining. Washington DC: IEEE Computer Society, 2001: 257-264.

[17] 张银蜂，郭华平，职为梅，等. 一种面向不平衡数据分类的组合剪枝方法 [J]. 计算机工程，2014，40(6)：157-161.

[18] 徐丽丽，闫德勤. 不平衡数据加权集成学习算法 [J]. 微型机与应用，2015，34(23)：7-10.

[19] López V, Fernández A, García S, *et al*. An insight into classification with imbalanced data: empirical results and current trends on using data intrinsic characteristics [J]. Information Sciences, 2013, 250(11): 113-141.

[20] 肖坚. 基于随机森林的不平衡数据分类方法研究 [D]. 哈尔滨：哈尔滨工业大学，2013.

[21] Moser R, Pedrycz W, Succi G. A comparative analysis of the efficiency of change metrics and static code attributes for defect prediction [C] //Proc of International Conference on Software Engineering. 2008: 181-190.

[22] 常瑞花，慕晓冬，宋国军，等. 不平衡数据的软件缺陷预测方法 [J]. 火力与指挥控制，2012，37(5)：56-59.

[23] Wang Shuo, Yao Xin. Using class imbalance learning for software defect prediction [J]. IEEE Trans on Reliability, 2013, 62(2): 434-443.

[24] Seiffert C, Khoshgoftaar T M, Hulse J V, *et al*. An empirical study of the classification performance of learners on imbalanced and noisy software quality data [J]. Information Sciences, 2007, 259: 571-595.

（上接第 2026 页）

[6] Peters F, Menzies T, Marcus A. Better cross company defect prediction [C] //Proc of Working Conference on Mining Software Repositories. 2013: 409-418.

[7] Ma Ying, Luo Guangchun, Zeng Xue, *et al*. Transfer learning for cross-company software defect prediction [J]. Information and Software Technology, 2012, 54(3): 248-256.

[8] Chen Lin, Fang Bin, Shang Zhaowei, *et al*. Negative samples reduction in cross-company software defects prediction [J]. Information and Software Technology, 2015, 62(1): 67-77.

[9] Nam J, Pan S J, Kim S. Transfer defect learning [C] //Proc of International Conference on Software Engineering. 2013: 382-391.

[10] Panichella A, Oliveto R, De Lucia A. Cross-project defect prediction models: L'Union fait la force [C] //Proc of Conference on Software Maintenance, Reengineering and Reverse Engineering. 2014: 164-173.

[11] Wang Song, Liu Taiyue, Tan Lin. Automatically learning semantic features for defect prediction [C] //Proc of the 38th International Conference on Software Engineering. New York: ACM Press, 2016: 297-308.

[12] Canfora G, Lucia A D, Penta M D, *et al*. Defect prediction as a multiobjective optimization problem [J]. Software Testing, Verification and Reliability, 2015, 25(4): 426-459.

[13] Nam J, Kim S. Heterogeneous defect prediction [C] //Proc of Joint Meeting of the European Software Engineering Conference and International Symposium on the Foundations of Software Engineering. 2015: 508-519.

[14] Jing Xiaoyuan, Wu Fei, Dong Xiwei, *et al*. Heterogeneous cross-company defect prediction by unified metric representation and CCA-based transfer learning [C] //Proc of the 10th Joint Meeting of the European Software Engineering Conference and the International Symposium on the Foundations of Software Engineering. New York: ACM Press, 2015: 496-507.

[15] Nam J, Kim S. CLAMI: defect prediction on unlabeled datasets [C] //Proc of International Conference on Automated Software Engineering. 2015: 452-463.

[16] Zhang Feng, Zheng Quan, Zou Ying, *et al*. Cross-project defect prediction using a connectivity-based unsupervised classifier [C] //Proc of the 38th International Conference on Software Engineering. New York: ACM Press, 2016: 309-320.

[17] Box G E P, Cox D R. An analysis of transformations [J]. Journal of the Royal Statistical Society, Series B: Methodological, 1964, 26(2): 211-252.

[18] D'Ambros M, Lanza M, Robbes R. An extensive comparison of bug prediction approaches [C] //Proc of Working Conference on Mining Software Repositories. 2010: 31-41.

[19] Liu Wangshu, Liu Shulong, Gu Qing, *et al*. Empirical studies of a two-stage data preprocessing approach for software fault prediction [J]. IEEE Trans on Reliability, 2016, 65(1): 38-53.

[20] 刘望舒，陈翔，顾庆，等. 软件缺陷预测中基于聚类分析的特征选择方法 [J]. 中国科学：信息科学，2016，46(9)：1298-1320.

[21] 刘望舒，陈翔，顾庆，等. 一种面向软件缺陷预测的可容忍噪声的特征选择框架 [J]. 计算机学报，2016，39：No.33.

[22] 陈翔，贺成，王宇，等. HFS：一种面向软件缺陷预测的混合特征选择方法 [J]. 计算机应用研究，2016，33(6)：1758-1761.

[23] 陈翔，陆凌姣，吉人，等. SBFS：基于搜索的软件缺陷预测特征选择框架 [J]. 计算机应用研究，2017，34(4)：1105-1108，1119.