Journal of Computer Applications ISSN 1001-9081 2018-09-10
计算机应用,2018,38(9):2637-2643,2659 CODEN JYIIDU http://www.joca.cn

文章编号:1001-9081(2018)09-2637-07 DOI:10.11772/j.issn.1001-9081.2018020507

基于数据过采样和集成学习的软件缺陷数目预测方法

简艺恒[1],余 啸[2*]

(1.北京理工大学 信息与电子学院,北京 102488; 2.武汉大学 计算机学院,武汉 430072)

(* 通信作者电子邮箱 xiaoyu_whu@yahoo.com)

摘 要: 预测软件缺陷的数目有助于软件测试人员更多地关注缺陷数量多的模块,从而合理地分配有限的测试资源。针对软件缺陷数据集不平衡的问题,提出了一种基于数据过采样和集成学习的软件缺陷数目预测方法——SMOTENDEL。首先,对原始软件缺陷数据集进行 n 次过采样,得到 n 个平衡的数据集;然后基于这 n 个平衡的数据集利用回归算法训练出 n 个个体软件缺陷数目预测模型;最后对这 n 个个体模型进行结合得到一个组合软件缺陷数目预测模型,利用该组合预测模型对新的软件模块的缺陷数目进行预测。实验结果表明 SMOTENDEL 相比原始的预测方法在性能上有较大提升,当分别利用决策树回归(DTR)、贝叶斯岭回归(BRR)和线性回归(LR)作为个体预测模型时,提升率分别为 7.68%、3.31%和 3.38%。

关键词: 软件缺陷预测;数据不平衡;过采样;集成学习

中图分类号: TP181 **文献标志码:** A

Software defect number prediction method based on data oversampling and ensemble learning

JIAN Yiheng[1], YU Xiao[2*]

(1. *School of Information and Electronics, Beijing Institute of Technology, Beijing* 102488, *China*;
2. *School of Computer Science, Wuhan University, Wuhan Hubei* 430072, *China*)

Abstract: Predicting the number of the defects in software modules can help testers pay more attention to the modules with more defects, thus reasonably allocating limited testing resource. Focusing on the issue that software defect datasets are imbalanced, a method based on oversampling and ensemble learning (abbreviate as SMOTENDEL) for predicting the number of defects was proposed in this paper. Firstly, n balanced datasets were obtained by oversampling the original software defect dataset n times. Then, n individual models for predicting the number of defects were trained on the n balanced datasets using regression algorithms. Finally, the n individual models were combined to obtain an ensemble prediction model, and the ensemble prediction model was used to predict the number of defects in a new software module. The experimental results show that SMOTENDEL has better performance than the original prediction method. When using Decision Tree Regression (DTR), Bayesian Ridge Regression (BRR) and Linear Regression (LR) as the individual prediction model, the improvement is 7.68%, 3.31% and 3.38%, respectively.

Key words: software defect prediction; data imbalance; over sampling; ensemble learning

0 引言

软件在如今社会发挥着举足轻重的作用,复杂系统的可靠性高度依赖于软件的可靠性。软件缺陷是导致系统失效和崩溃的潜在根源[1],如果能对软件缺陷进行预测,就能在造成危害前对软件缺陷进行排查和修复,从而减少软件崩溃所带来的经济损失。

伴随着第一个软件的诞生并延续至今,软件缺陷预测技术已得到了长足的发展[2]。已有很多研究提出了很多软件缺陷预测方法。如文献[3]探索了传统的机器学习模型和半监督学习在软件缺陷预测中的应用,并在 PROMISE 数据集上进行测试,达到了工程的需求。文献[4]比较了包括决策树、贝叶斯、向量机、人工神经网络等传统机器学习模型在多机构数据集上的预测表现。

但是这些模型仅仅是基于分类问题,将软件模块分为由缺陷和无缺陷两类,不能预测出软件缺陷数目。如果能够预测软件缺陷数目,就能将有限的软件资源优先分配给软件缺陷数目多的模块,从而提升测试的效率。举例而言,假如预测出一个软件中有 40 个模块具有缺陷,而测试人员因测试资源有限,只能对其中 10 个软件模块进行测试。如果使用基于分类的预测方法预测出有 15 个软件模块有缺陷,测试人员只能随机抽取其中的 10 个模块进行测试,但如果能够预测出软件缺陷模块的具体数目,那么测试人员可以优先测试缺陷数目多的 10 个模块,大大提升了测试效率。

目前针对软件缺陷数目的预测亦有一些研究。如文献[5]构建了一个灵活的贝叶斯网络,利用贝叶斯网络进行软件缺陷数目的预测,取得了稳定的输出效果;文献[6]提出了一个基于缺陷状态转移模型的软件缺陷数目预测方法,并且在软件开发实践中得到了验证,但是这些研究却忽略了软件缺陷数目分布不平衡的问题,即软件中缺陷数目为 0 的软件模块的数量一般远多于缺陷数目大于 0 的软件模块数量。如果对软件模块训练数据不作任何处理,会导致训练出的软件缺陷数目预测

收稿日期:2018-03-12;修回日期:2018-05-17;录用日期:2018-06-04。

作者简介:简艺恒(1998—),男,湖北武汉人,主要研究方向:软件工程、数据挖掘; 余啸(1994—),男,湖北汉川人,博士研究生,主要研究方向:软件工程、数据挖掘。

模型对多数类样本(缺陷数目为0的软件模块)具有偏向性,从而对少数类样本(缺陷数目大于0的软件模块)的预测方面存在失真。而对缺陷模块具有的缺陷数目的准确预测正是软件缺陷预测技术的重点。故在进行软件缺陷数目预测的时候,有必要解决软件缺陷数据集的不平衡性问题。

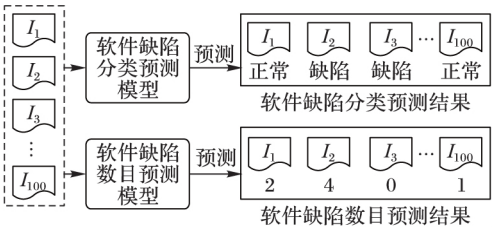

图1 软件缺陷分类预测与数目预测区别

Fig. 1 Difference between classification prediction and number prediction of software defects

针对不平衡数据集的处理通常包含代价敏感方法和采样方法。文献[5]指出虽然代价敏感方法能够有效地提高稀有类的识别率,但是仍存在若干局限,例如错误代价难以被精确估计,从而影响代价敏感的效果。采样方法分为欠采样方法和过采样方法,其中文献[6]表明欠采样通过删除部分多数类样本使数据分布重新平衡,但会造成信息的丢失。过采样方法通过增加少数类样本的数量使数据分布重新平衡,保留了所有的信息,相比欠采样方法有更好的效果。文献[7]提出了一种基于SMOTE(Synthetic Minority Over-sampling TEchnique)过采样的SMOTER(SMOTE for Regression)方法用于软件缺陷数目预测,具有普适性强和易于其他模型组合的优点,实验结果表明该方法具有良好的性能表现。

但是不平衡数据再平衡后,使用单一的回归模型,容易出现过拟合降低了模型的稳定性。集成学习通过构建并合并多个个体学习器来完成学习任务,常可获得比单一学习器更显著优越的泛化性能。目前已有的研究普遍采用了集成学习的方法,在实验中取得了较高的准确性和稳定性[8-10]。

目前的研究针对软件缺陷数目预测模型的评价指标一般为预测缺陷数量与实际缺陷数量的绝对误差和相对误差,如文献[11-12],但是这些评价指标同样没有考虑到软件缺陷不平衡的问题,由于软件缺陷数目分布不平衡问题,这类评价指标容易导致评价过于乐观。本文采用专门用于软件缺陷数目预测领域的FPA(Fault-Percentile-Average)指标[13],能够有效地表现出预测模型的准确性。

本文在现有研究的基础上,结合过采样方法和集成学习的方法,提出了一种基于数据过采样和集成学习的软件缺陷数目预测方法——SMOTENDEL(SMOTE for predicting the Number of Defect using Ensemble Learning)。该方法首先通过数据不平衡率确定过采样比例,执行n次过采样方法,得到n个不同的平衡的软件缺陷数据集;然后基于这n个平衡的软件缺陷数据集训练出n个个体学习器;最后,对这n个个体学习器通过集成得到一个组合学习器,并利用这个组合学习器来预测待预测的软件模块的缺陷数目。

1 相关工作

目前对软件缺陷的预测已有不少研究,其中可分为两大类:一种是预测软件模块是否有缺陷,一种是预测软件模块具体的缺陷数目。

1.1 预测软件模块是否有缺陷

文献[14]针对目前软件缺陷预测方法大多是针对特定的数据集建立的模型的现状,研究了跨项目软件缺陷预测的方法,发现将针对特定数据集的模型与跨项目模型的预测结果相比,只存在微小差异,在此基础上建立了一种跨项目软件缺陷预测模型,取得了更高的性能表现。文献[15]提出了一种跨项目软件模块缺陷预测方法,建立了基于元分类逻辑回归预测模型,使用成本效益和F-measure作为评价标准,实验结果表明该模型相较于常规的逻辑回归预测模型在F-measure上和平均F-measure上提升了36.88%;文献[16]比较了基于信息增益特征选择算法(GIS)和基于近邻的遗传算法(NN-filter)的预测性能,实验结果表明GIS方法在多个指标上取得了最优的表现;文献[17]提出了一种多目标优化的JIT-SDP(Just-In-Time Software Defect Prediction)方法,使用逻辑回归方法建立模型,实验结果表明该模型的预测能力强于目前最先进的监督学习和非监督学习模型,且在跨项目、跨时间预测上同样有良好的表现。上述方法没有进行软件缺陷数目的预测,虽然可以预测出软件缺陷模块,但是在测试资源有限的情况下无法合理分配资源,优先测试具有更多缺陷的软件模块,会造成测试资源的浪费。

软件缺陷数据集一般存在数据不平衡问题,即在数据集中有缺陷的模块数量很少,无缺陷模块数量很多。针对数据不平衡的问题,目前已有不少研究,主要包含代价敏感和采样方法。其中代价敏感方法[18]指的是将软件缺陷分为两种类型,两种类型的错误分类成本不同,其中代价敏感方法将软件缺陷分为两种错误分类代价不同的类型,从而使错误分类代价最小。文献[19-20]探索了基于代价敏感的不平衡数据再平衡方法,取得了良好的表现效果,但是存在错误分类成本难以准确制定的不足,对预测精度有较大的影响。采样方法分为欠采样和过采样方法,欠采样方法通过删除多数类样本使得不同类型样本数量基本保持一致。文献[21]首先使用清除正常模块中特征相重叠的模型,对整个数据集进行多次欠采样使数据集再平衡,再使用AdaBoost方法建立预测模型,实验结果表明该方法在AUC(Area Under Curve)和G-mean的评价指标上取得了良好的结果。文献[22]提出了一种基于RUS(Random Under-Sampling)欠采样方法的数据再平衡方法,该模型具有建立简单和高效的优点,但是它的预测效果很大程度上取决于数据集自身的特性和后续数据处理时训练算法的选择。文献[23]提出了一种在软件缺陷数目预测中的针对不平衡现象的方法。文献[7]在SMOTE过采样的基础上对其进行改进提出了SMOTER方法,用于预测目标变量的极值点;该方法普适性强,可与多种的回归模型结合,是一种高效通用的数据平衡化方法。文献[24]考虑了缺陷的软件模块分布不平衡问题,提出了基于SDAEs(Stacked Denoising AutoEncoders)深度学习和集成学习的软件缺陷预测模型,先对软件模块数据集进行深度学习,然后使用集成学习的方法处理软件的不平衡问题,该模型在NASA的软件模块数据集上取得了良好的表现。

1.2 预测软件模块的缺陷数目

文献[25]构建了一个灵活的贝叶斯网络,利用贝叶斯网络进行软件缺陷数目的预测,取得了稳定的输出效果;文献[26]提出了一个基于缺陷状态转移模型的软件缺陷数目预测方法,并且在软件开发实践中得到了验证;文献[27]探索了决策树回归(Decision Tree Regression, DTR)算法在版本内缺陷数目预测和跨版本缺陷数目预测能力,在PROMISE的5个数据集上的实验表明决策树回归算法在平均绝对误差和平均相对误差的评价指标上取得了良好的效果。文献[28]在探索了遗传算法、多层感知器算法、线性回归(Linear

Regression，LR)、决策树回归、泊松回归和负二项回归在软件缺陷数目预测上的应用，实验结果表明决策树回归、遗传算法、多层感知器算法和线性回归在平均绝对误差和平均相对误差上取得了更良好的表现，而负二项回归和泊松回归的表现最差。文献[12]探究了在项目内和跨项目的软件缺陷数据集上使用决策树回归、贝叶斯岭回归(Bayes Ridge Regression，BRR)、支持向量回归、线性回归、近邻回归、梯度下降算法对软件缺陷数目进行预测，并使用精确率和均方根误差作为评价指标，结果表明决策树算法在以上的 6 个算法中的表现最为良好，同时 6 种回归方法在项目内和跨项目中取得了相似且良好的结果。文献[11,29]提出了一种基于线性、非线性的软件模块缺陷数目预测方法，采用集成学习的方法，将线性方法(线性回归)和非线性方法(决策树回归、支持向量机等方法)的结果进行集成，实验结果表明该方法在软件模块缺陷数目预测的绝对误差和相对误差上取得了称为 level I 的良好表现。相比而言，本文首先将不平衡的训练集利用过采样方法进行平衡，然后再利用集成学习方法构建组合预测模型。文献[30]建立了基于负二项式回归的软件模块缺陷数目预测模型，并将其应用到两个大型工业系统中，实验结果表明，该模型预测出了两个工业系统软件模块中，分别包含总缺陷数目 71% 和 92% 的 20% 软件模块，即成功预测出了具有最多软件缺陷数目、最应该被测试的软件模块，但是上述研究没有考虑到软件缺陷预测中，软件缺陷模块分布不平衡问题，预测结果存在偏向性从而不准确。文献[31]提出了一个结合数据过采样、欠采样与 AdaBoost. R2 算法的软件缺陷数目预测方法。相比文献[31]的方法，本文方法利用平均法的组合策略对个体回归模型进行组合。

2　SMOTENDEL 方法

本文针对软件缺陷预测中存在的数据不平衡现象以及单一学习模型导致的过拟合问题，结合过采样和集成学习提出一种软件缺陷数目预测方法——SMOTENDEL。该方法包含过采样、回归和集成 3 个阶段。首先通过对原始缺陷数据集进行多次过采样形成多个平衡的缺陷数据集，然后利用当前经典的回归算法在各平衡数据集上进行训练，得到若干个个体软件缺陷数目预测模型，将多个个体预测模型进行集成得到一个组合软件缺陷数目预测模型，最后利用该组合预测模型对新的软件模块进行预测。SMOTENDEL 方法的基本框架如图 2 所示。

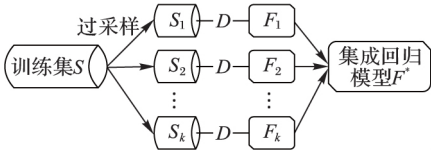

图 2　SMOTENDEL 方法过程示意图

Fig. 2　Schematic diagram of SMOTENDEL

其中，S_i 表示第 i 次过采样之后的平衡数据集，D 代表一种回归算法，F_i 表示基于平衡数据集 S_i 训练出的个体软件缺陷数目预测模型，F^* 表示组合预测模型。

2.1　SMOTEND 采样

SMOTE 是一种经典的过采样方法，由 Chawla 等[13]在 2002 年提出，在不平衡数据的平衡化方面有着不俗的表现，但 SMOTE 算法只能处理分类领域的数据不平衡问题。文献[7]提出了一种 SMOTE 的改进算法——SMOTER，用于预测目标变量的极值点。考虑到本文是数值预测问题以及软件缺陷模块在整体软件模块中占比较小的事实，本文改进 SMOTER 方法，用以解决软件缺陷数据集数据不平衡的问题，称改进的 SMOTER 方法为 SMOTEND(SMOTE for predicting the Number of Defects)。为便于后面的描述，本文给出下列定义：

定义 1　正常模块。缺陷数目为零的软件模块。

定义 2　缺陷模块。缺陷数目大于零的软件模块。

SMOTEND 方法包含两个关键步骤：

步骤 1　确定原始数据集中哪些缺陷模块被用于合成新的缺陷模块；

步骤 2　计算新的缺陷模块的特征向量和缺陷个数。

SMOTEND 首先依据正常模块和缺陷模块的期望比例，确定需要合成的新的缺陷模块的个数。然后，根据需要合成的新的缺陷模块的个数和原始缺陷模块个数确定每个原始缺陷模块需要选择的近邻个数，随后根据软件缺陷模块和近邻的软件缺陷模块构造新的软件缺陷模块。具体步骤如下：

针对步骤 1，设正常模块的数量为 n，缺陷模块的数量为 m，期望得到的缺陷模块与正常模块的比例为 $ratio$。当 $ratio \leqslant [2 \times m/n]$，即需要合成的缺陷模块个数小于原始数据集中缺陷模块个数时，参与合成的原始缺陷模块数目 $p = ratio \times n - m$，每个参与合成的原始缺陷模块的选取的近邻个数 $t = 1$；当 $ratio > [2 \times m/n]$ 时，即需要合成的缺陷模块个数大于原始数据集中缺陷模块个数时，参与合成的原始缺陷模块数目 $p = m$，每个参与合成的原始缺陷模块的选取的近邻个数 $t = [(ratio \times n/m)]$。

针对步骤 2，用 $X = (\boldsymbol{x}, y)$ 表示一个软件缺陷模块，其中 $\boldsymbol{x}$ 是缺陷模块 X 的特征向量，y 是缺陷模块 X 的软件缺陷数目。对每个参与合成新模块的原始缺陷模块 X_i，在其最近邻的 k 个缺陷模块中(根据文献[7]，本文实验中 k 取 5)，有放回抽样出 t 个最近邻模块 $\{X_{ij} \mid j \in (0,t]\}$，根据模块 X_i 与抽出模块 X_{ij} 构造新的软件缺陷模块 X_{new}，X_{new} 的特征向量 $\boldsymbol{x}_{\text{new}}$ 为：

$$\boldsymbol{x}_{\text{new}} = \boldsymbol{x}_i + \text{rand}(0,1) \times (\boldsymbol{x}_{ij} - \boldsymbol{x}_i) \tag{1}$$

记新合成的缺陷模块特征向量 $\boldsymbol{x}_{\text{new}}$ 与合成新缺陷模块的两个母体模块特征向量 $(\boldsymbol{x}_i, \boldsymbol{x}_{ij})$ 之间的距离分别为 d_1 和 d_2，则新合成的缺陷模块 X_{new} 的缺陷数目为：

$$y_{\text{new}} = \frac{d_2 \times y_i + d_1 \times y_{ij}}{d_1 + d_2} \tag{2}$$

其中 y_i 和 y_{ij} 分别是缺陷模块 X_i 和 X_i 的近邻模块 X_{ij} 的缺陷数目。

举例说明：如图 3 所示，根据点(23，9) 和(5，14) 代表的这两个缺陷模块构造新的软件缺陷模块，假设 rand(0，1) 取 0.2，则新合成的缺陷模块为：(5，14) + 【(23，9) - (5，14)】 × 0.2 = (8.6，13)

设(5，14) 点的缺陷数目为 4，(23，9) 点的缺陷数目为 1，求得新合成的点和合成该点的两个母体点之间的欧氏距离分别为 $d_1 = 3.73$ 和 $d_2 = 14.94$，则新合成的点的缺陷数目为

$$\frac{d_1}{d_1 + d_2} \times 4 + \frac{d_2}{d_1 + d_2} \times 1 = 1.59 \tag{3}$$

四舍五入得点(8.6，13) 代表的模块的缺陷数目为 2。

SMOTEND 伪代码如算法 1 所示。

算法 1　SMOTEND。

输入：软件模块数据集 $S = 【(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_K, y_K)】$；采样后缺陷模块和正常模块数量的期望比例 $ratio$。

1) for $i = 1$ to p do:
2) 对第 i 个缺陷模块选择 k 个最近邻缺陷模块;
3) for $j = 1$ to t do:
4) 从 k 个最近邻中随机抽取第 j 个最近邻缺陷模块,记作 X_{ij};
5) 根据软件模块 X_i,X_{ij} 和式(1)构造新的模块的特征向量 $\boldsymbol{x}_{ij}$;
6) 根据式(2)得到新的模块的缺陷数目 y_{ij};
7) end for
8) end for
9) 把上述新生成的缺陷模块集与原缺陷模块集合并,得到一个平衡的数据集 S';

原始不平衡数据通过上述过采样,即根据缺陷模块和最近邻的几个缺陷模块构造新的软件缺陷模块,得到 $p \times t$ 个新的缺陷模块。这种方式能够极大地改善数据的不平衡性。

2.2 回归算法

为了预测软件模块的缺陷数目,本文采用回归算法对平衡化后的数据集进行学习,得到一个学习模型 F,根据训练出的模型对待检测的软件缺陷数据集进行预测,可得到软件缺陷数据集的缺陷数目预测结果。已有研究[8-9]表明决策树回归(DTR)模型,贝叶斯岭回归(BRR)和线性回归(LR)这三种回归模型在软件缺陷数目预测中被广泛使用并取得了较好的预测效果,同时这三种模型分属于三种不同类型的回归模型,其中,DTR 属于决策树模型,BRR 属于概率框架模型,LR 属于统计学模型。这三个回归模型的详细信息如下:

决策树回归(DTR) 依据决策树原理,曲线逼近给定的训练集,然后使用训练出的预测模型对测试集进行预测。

线性回归(LR) 对存在线性关系的一个或多个自变量和一个因变量进行线性拟合,使用拟合的模型对测试集进行预测。

贝叶斯岭回归(BRR) 是一种基于贝叶斯算法的回归预测方法,可利用正则参数的对贝叶斯的参数进行先验,规避了主观性的矛盾,结果更具备说服力。

2.3 集成学习

集成学习(ensemble learning)通过构建并结合多个学习器,来完成学习任务。通过集成学习,将预测能力较弱的个体学习器进行结合,可以得到预测能力较强的组合学习器,显著提高泛化性能。集成学习的一般结构为:先产生一组“个体学习器”,再使用某种策略将它们结合起来。集成学习的策略分为两种,一种是同质集成方法,即使用同一种学习方案对数据集的不同子集进行学习,将学习的结果集成在一起;另一种是异质集成方法,使用多种不同的学习方法对同一数据集进行学习,将学习的结果进行集成。本文采用第一种同质集成方法。具体步骤如下:

首先利用 2.1 节中提出的 SMOTEND 对原始软件缺陷数据集进行过采样使数据集平衡化,得到一个平衡的软件缺陷数据集,然后利用 2.2 节中的经典回归算法对平衡数据集进行学习得到一个预测模型。为了避免在单一数据集上学习而导致结果偏差过大,本文提出对多次过采样得到的软件缺陷数据集进行多次回归,得到多个个体软件缺陷数目预测模型,然后,将这多个个体预测模型集成得到一个组合预测模型。本文选择的组合策略为平均法,即将这多个个体模型对待预测软件模块的缺数目的预测值取平均。本文把这种方法称为 SMOTENEDEL。SMOTENDEL 方法步骤如算法 2 所示。

算法 2 SMOTENDEL。

输入:平衡数据集的个数:n;软件缺陷数据集 S;一种回归算法,如决策树回归算法。

输出:组合预测模型 F^*。

Begin
1) for $i = 1$ to n do
2) 对 S 进行 SMOTEND 过采样得到平衡数据集 S_i;
3) 基于 S_i 利用回归算法学习出一个个体软件缺陷数目预测模型 F_i;
4) end for
5) $$F^*(\boldsymbol{x}) = \frac{1}{n}\left(\sum_{i=1}^{n} F_i(\boldsymbol{x})\right)$$
End

经过上述 3 个阶段,得到一个组合软件缺陷数目回归预测模型 F^*。利用该组合回归预测模型对新的软件模块进行缺陷数目预测得到新的软件模块的缺陷数目。

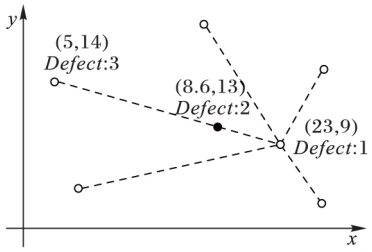

图 3 SMOTENDEL 示意图
Fig. 3 Schematic diagram of SMOTENDEL

3 实验设置

3.1 实验数据集

本实验根据文献[33],从开源的数据集 PROMISE 中选取 5 种实验数据集,其中,软件特征采用 CK metrics 数据集提供的 20 个软件缺陷特征。实验数据集的详细信息如表 1 所示。通过表 1 可以看出软件缺陷数据集中软件缺陷数目分布有极大的不平衡性,缺陷模块的数量明显远少于正常模块的数量。

表 1 实验中使用的 PROMISE 数据集
Tab. 1 PROMISE datasets used in the experiment

数据集	版本号	模块数	缺陷模块数	缺陷比率/%	缺陷总比率/%
Ant	1.3	125	20	16.00	20.68
	1.4	178	40	22.50	
	1.5	293	32	10.90	
	1.6	351	92	26.20	
	1.7	745	166	22.30	
Ivy	1.1	111	63	56.80	16.90
	1.4	241	16	6.60	
	2.0	352	40	11.40	
Jedit	3.2	272	90	33.10	17.32
	4.0	306	75	24.50	
	4.1	312	79	25.30	
	4.2	367	48	13.10	
	4.3	492	11	2.20	
Synapse	1.0	157	16	10.20	25.51
	1.1	222	60	27.00	
	1.2	256	86	33.60	
Xalan	2.4	723	110	15.20	32.56
	2.5	803	387	48.20	

3.2 评价指标

现有的软件缺陷数目预测的文献采用的大多是传统的评价指标。如文献[12]采用了平均绝对误差(Average Absolute

Error，AAE)、平均相对误差(Average Relative Error，ARE)、均方根误差(Root Mean Square Error，RMSE)等指标，但这些指标没有考虑到缺陷数目的不平衡性，在本文所研究的问题中具有较大的局限性。比如 AAE 的定义如下:

$$AAE = \frac{1}{n}\sum_{i=1}^{n} | x_{i,\text{actual}} - x_{i,\text{predict}} | \quad (4)$$

其中:$x_{i,\text{actual}}$ 代表数据集中第 i 个软件模块的实际软件缺陷数目，$x_{i,\text{predict}}$ 代表数据集中第 i 个软件模块的软件缺陷数目的预测值，n 代表数据集中软件模块的个数。

鉴于数据集的极不平衡性，类似于 AAE 的这类评价指标可能会导致评价过于乐观。软件缺陷数目预测技术的重点是正确地预测出软件缺陷数目多的模块，而这些模块往往处于极少数的地位，它们的预测错误在类似 AAE 的评价指标中除以庞大的基数，错误会被掩盖。

例如针对 Ant. 1. 3 数据集，其有 125 个模块，正常模块有 105 个，缺陷数目为 1、2、3 的模块分别有 11、5、4 个。假设预测模型预测这 20 个缺陷模块的缺陷数目都为 1，则 AAE 计算得 0. 104；假设预测模型预测缺陷数目为 3 的预测正确了 2 个，缺陷数目为 2 的预测正确了 3 个，缺陷数目为 1 的预测正确了 8 个，其他的全预测为正常模块，则 AAE 计算得 0. 104，与前者结果相同，但显然后者的预测结果对测试人员帮助更大，故不能使用类似于 AAE 的这类评价指标作为本研究的评价指标。

文献[13]提出 FPA(Fault-Percentile-Average)这个评价指标。FPA 是一种专门用于软件缺陷预测领域的评价指标，广泛用于验证预测模型的预测性能，该指标以软件缺陷数目为权重，如果一个模块的软件缺陷数目预测值越大，则该模块的权重就高，对评价指标的影响大。因此在软件缺陷数目多的模块上的预测准确度越高，评价指标表现就越良好。

FPA 的计算公式如下:

$$FPA = M/(n \times Y) \quad (5)$$

其中:n 代表数据集中软件模块个数，Y 代表数据集中所有软件模块所具有的缺陷的总数，M 是将软件模块按照预测缺陷数目降序排列得到的实际缺陷数目的累加和，具体计算如下。

考虑一个含有 n 个软件模块的数据集，按预测出的缺陷数目升序排列 $S = \{S_1, S_2, \cdots, S_n\}$，即软件模块 S_n 被预测为拥有最多的软件缺陷。记 $\{y_1, y_2, \cdots, y_n\}$ 为对应软件模块实际的缺陷数目。对第 $m(0 < m \leqslant n)$ 个模块，计算软件缺陷数目比该模型的缺陷数目多的所有软件模块 $\{S_m, S_{m+1}, \cdots, S_n\}$ 的缺陷数目和:

$$M = \sum_{m=1}^{n} P_m = \sum_{m=1}^{n}\sum_{k=n-m+1}^{n} y_k = \sum_{k=1}^{n} k \times y_k \quad (6)$$

FPA 值越大，表明软件缺陷数目预测模型的预测效果越好。本实验中，选取 FPA 作为评价指标评价软件缺陷数目预测模型效果的评价指标。

3.3　十折交叉验证法

为了验证 SMOTENDEL 预测方法导出的模型在软件缺陷数目预测上的预测性能，需要对模型进行验证。对模型的验证通常有交叉验证法(Cross-Validation)和自助采样法(Bootstrap)两种方法。文献[13]分别探究了 k 折交叉验证法中，k 的取值对验证效果的影响和自助采样法中采样的数目对验证效果的影响，实验结果表明十折交叉验证法在对模型准确性的验证中取得了最优的表现。因此本文采用十折交叉验证法对 SMOTENDEL 预测方法进行验证。

本文在进行十折交叉检验时，将表 1 中同一项目不同版本的数据集合并为一个数据集。然后将数据集等分为 10 组，从中取 1 组作为测试集，剩下的 9 组作为训练集，利用 SMOTENDEL 训练得出软件缺陷数目预测模型，用预测模型对测试集的软件模块进行预测，得到测试集的软件缺陷数目的预测值。依次选择 10 组中的每 1 组作为测试集，重复上述步骤可以得到整个软件缺陷数据集的缺陷数目预测值，根据软件缺陷数据集的实际缺陷数目和预测缺陷数目计算 FPA 值。

3.4　研究问题

为了验证 SMOTENDEL 方法在软件缺陷数目预测方面的性能，本文提出了以下两个研究问题:

问题 1　SMOTENDEL 方法中，应该如何设置个体缺陷数目预测模型个数 n?

问题 2　SMOTENDEL 是否能提升软件缺陷数目预测模型的预测性能?

4　实验结果分析

4.1　研究问题 1

针对 3.4 节中的问题 1，本文分别选择个体缺陷数目预测模型个数 n 为 1，2，3，…，9，10，15，20，…，40 和 45，对数据集进行 SMOTENDEL 方法的软件缺陷预测，记录 SMOTENDEL 方法在不同 n 值下的表现。

根据实验结果数据绘出如图 4 所示的 FPA-n 折线图。由 FPA-n 折线图可以得出，在使用决策树回归算法的 SMOTENDEL 方法中，当 n 取 1 ~ 5 的时候，预测模型的 FPA 值随着 n 的增加而增大；当 n 处于 5 ~ 15 时，模型的稳定性较差，FPA 值具有较大的起伏，但整体趋势随着 n 的增加而增大；当 n 大于 15 时，模型的 FPA 值基本保持高水平和稳定。

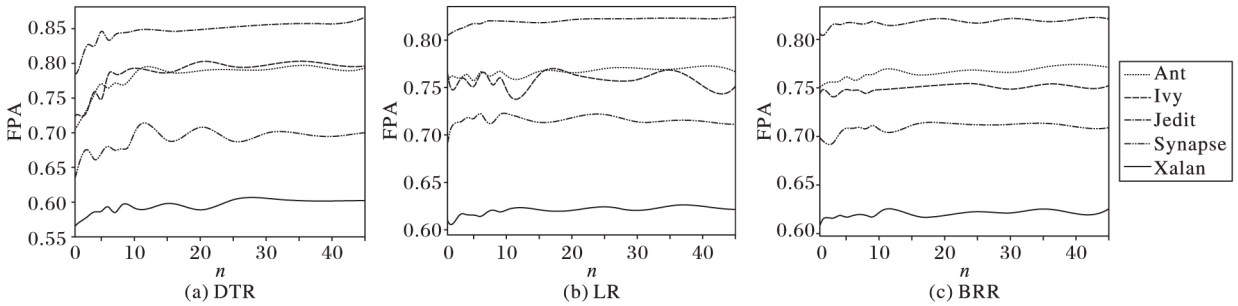

图 4　采用 3 种回归模型的 SMOTENDEL 在 5 种数据集上的预测性能随个体模型个数 n 的变化

Fig. 4　Prediction performance of SMOTENDEL method using three regression models on five datasets with different number n of individual models

在使用贝叶斯岭回归算法的 SMOTENDEL 方法中，当 n 取 1 ~ 10 的时候，预测模型的 FPA 值随着 n 的增加而增大，

但在 Synapse 数据集上有较大起伏;当 n 处于 10 ~ 15 时,预测模型的 FPA 值在 Jedit 和 Synapse 数据集上随着 n 的增大先下降后提升,在 Ant 和 Xalan 数据集上随着 n 的增大先提升后下降,在 Ivy 数据集上随着 n 的增大而提升;当 n 大于 15 时,预测模型的 FPA 值基本保持高水平和稳定。

在使用线性回归算法的 SMOTENDEL 方法中,当 n 取1 ~ 5 的时候,预测模型的 FPA 值随着 n 的增加而增大,但在 Ivy 和 Synapse 数据集上有较大起伏;当 n 处于 5 ~ 10 时,FPA 值有小幅度的振荡;当 n 大于 15 时,预测模型的 FPA 值基本保持高水平和稳定。

根据实验结果得出结论:为了使 SMOTENDEL 方法得到的模型具有良好的性能表现,应该将 n 设定为 15 或 15 以上。考虑到集成学习次数多对计算资源的占用较大,依实验结果,将 n 设定为 15 可以在测试资源消耗与性能之间取得较好的平衡。

4.2 研究问题 2

根据 4.1 节的结论,实验将预测模型个数 n 设定为 15,使用 SMOTENDEL 方法对软件缺陷数据集进行学习,导出软件缺陷数目预测模型,使用十折交叉验证法对预测模型进行验证,使用 FPA 指标进行评价。FPA 的最高值,即所有软件模块缺陷数目全部预测正确,分别为 0.931 6,0.957 1,0.957 6,0.911,0.874 8。对各个数据集分别采用 SMOTENDEL 预测方法和原始预测(RAW)方法得到的 FPA 值以及平均值(Average)如表 2 所示,其中 ELEVATION RATIO(ER)为 SMOTENDEL 方法相对于直接进行回归预测方法的 FPA 提升率。实验结果表明,在 FPA 绝对值提升率上,基于 DTR 决策树回归的 SMOTENDEL 方法相对于直接回归的传统方法有着平均 7.68% 的提升,基于线性回归(LR)和贝叶斯岭回归(BRR)的 SMOTENDEL 方法相对于传统直接回归的方法有着平均 3.31% 和 3.38% 的提升。

表 2 5 个数据集在 3 个回归模型下的 FPA 值以及 FPA 提升率

Tab. 2 Performance of 5 datasets with three regression models on FPA and elevation ratio

回归模型	数据集	SMOTENDEL	RAW	Ds/%	Dr/%	EDR/%	ER/%
DTR	Ant	0.777 5	0.687 4	16.54	26.21	36.89	13.11
	Ivy	0.784 3	0.691 1	18.05	27.79	35.05	13.49
	Jedit	0.848 1	0.724 4	11.43	24.35	53.06	17.08
	Synapse	0.690 7	0.629 4	24.18	30.91	21.77	9.74
	Xalan	0.594 0	0.561 1	32.10	35.86	10.49	5.86
	平均	0.738 9	0.658 7	20.46	29.02	31.45	11.86
LR	Ant	0.770 5	0.716 5	17.29	23.09	25.12	7.54
	Ivy	0.753 9	0.742 2	21.23	22.45	5.43	1.58
	Jedit	0.819 4	0.807 9	14.43	15.63	7.68	1.42
	Synapse	0.717 7	0.659 4	21.22	27.62	23.17	8.84
	Xalan	0.620 5	0.579 3	29.07	33.78	13.94	7.11
	平均	0.737 5	0.664 5	20.61	28.40	29.10	10.74
BRR	Ant	0.773 0	0.709 3	17.02	23.86	28.67	8.98
	Ivy	0.748 0	0.734 6	21.85	23.25	6.02	1.82
	Jedit	0.821 8	0.800 3	14.18	16.43	13.69	2.69
	Synapse	0.710 1	0.665 1	22.05	26.99	18.30	6.77
	Xalan	0.623 6	0.584 1	28.72	33.23	13.57	6.76
	平均	0.731 4	0.674 7	21.25	27.33	23.17	8.36

此外,本文还将 SMOTENDEL 和 RAW 方法与最优模型(即将所有软件模块缺陷数目全部预测正确)进行了比较,SMOTENDEL_DIFFERENCE(Ds)、RAW_DIFFERENCE(Dr)表示 SMOTENDEL、RAW 与最优模型 FPA 值相差的百分比,ELEVATION RATIO OF DIFFERENECE (ERD) 为 SMOTENDEL 与最优模型 FPA 值相差的百分比相较于 RAW 与最优模型 FPA 值相差的百分比的提升率。

在与完全预测正确的结果的 FPA 值比较中,基于决策树回归的 SMOTENDEL 方法与完全预测正确的 FPA 值相差 20.46%,相比传统直接回归方法有着近 31.45% 的提升,基于 LR 线性回归和 BRR 贝叶斯岭回归的 SMOTENDEL 方法与完全预测正确的 FPA 值相差分别为 20.65% 和 20.67%,相比传统直接回归方法也有着 15.07% 和 16.05% 的提升。

图 5 为使用盒须图对 SMOTENDEL 方法在各个数据集上的 FPA 表现进行的描述,盒须图包含最大值、上四分位点、中位数、平均数、下四分位点和最小值,能够有效地对比 SMOTENDEL 方法和传统预测方法在 FPA 上的表现。

根据图 5 可以发现,在分别使用决策树、线性回归、贝叶斯岭回归的情况下,本文提出的 SMOTENDEL 软件缺陷预测方法较传统、不作数据处理的回归预测方法在四分位点上等多个指标上均取得了明显的提升。

图 6 对基于三种回归方法的 SMOTENDEL 软件缺陷数目预测方法得到的 FPA 提升率进行对比分析。

由图 6 可知,决策树回归(DTR)下的 SMOTENDEL 方法得到的 FPA 值的提升率最为明显。线性回归(LR)和贝叶斯岭回归(BRR)在中位数、最大最小值上没有明显的差异,贝叶斯岭回归较线性回归在四分位点上有着略微的优势。

根据实验结果得出结论:SMOTENDEL 方法能够有效地提升软件缺陷数目预测的效果。基于决策树算法的 SMOTENDEL 方法具有最良好的性能表现。

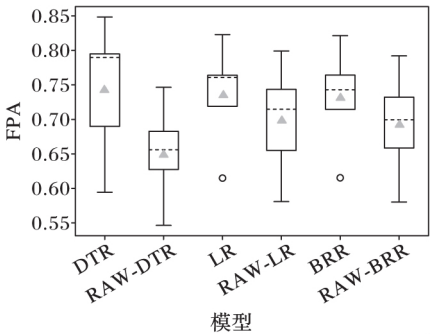

图 5 对 5 个数据集分别采用和不采用 SMOTENDEL 方法进行预测得到的 FPA 值

Fig. 5 Average FPA of three regression models on five datasets by using SMOTENDEL method or not

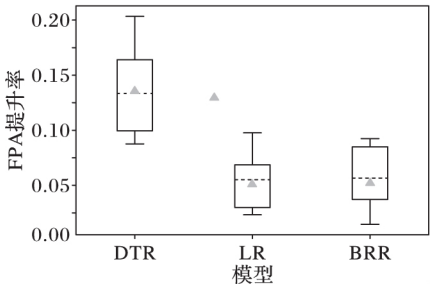

图 6 对 5 个数据集采用 3 种回归模型的 SMOTENDEL 方法的 FPA 提升率

Fig. 6 FPA elevation ratio of SMOTENDEL method using three regression models on five datasets

5　结语

针对软件缺陷数目预测问题,本文提出了一种基于过采样和集成学习的 SMOTENDEL 方法。SMOTENDEL 方法首先对不平衡的软件缺陷数据集进行多次过采样得到多个平衡数据集,然后对这多个平衡数据集进行回归预测得到对应的个体预测模型,随后利用集成学习的方法将这多个个体预测模型集成,得到一个组合预测模型,最后利用该组合预测模型对待预测的软件模块进行预测,得到其软件缺陷数目。本文使用 FPA 作为评价指标,利用十折交叉验证法对该方法进行检验。实验结果表明:1)SMOTENDEL 方法能够有效提高软件缺陷数目预测模型的性能,其中基于决策树回归的预测模型取得了最优的性能表现;2)集成学习次数 n 选择为 15 时能够同时占有较少的计算资源而获得良好的性能表现。

本文所选择的数据集特征全部来源于同一类 Chidamber and Kemerer (CK) metrics,这种类型的软件缺陷数据主要来自使用面对对象语言编写的软件系统,在 1994 年创建并经历了时间的考验,在软件缺陷预测技术上被普遍使用并取得了良好的表现,但是由于创建时间过早,创建时里面只有 SmallTalk 和C ++ 两种语言,在现在计算机语言飞速发展的如今,使用 CK 软件缺陷数据得到的预测模型可能并不完全满足于现代工业界的软件缺陷预测需求。在未来会使用多个不同种类的数据集来验证 SMOTENDEL 方法的普遍性,使其更好地服务于现代工业界。

参考文献 (References)

[1] 王青,伍书剑,李明树. 软件缺陷预测技术 [J]. 软件学报,2008,19(7):1565 - 1580. (WANG Q, WU S J, LI M S. Software defect prediction [J]. Journal of Software, 2008, 19(7): 1565 - 1580.)

[2] MALHOTRA R. A systematic review of machine learning techniques for software fault prediction [J]. Applied Soft Computing Journal, 2015, 27(C): 504 - 518.

[3] LI M, ZHANG H, WU R, et al. Sample-based software defect prediction with active and semi-supervised learning [J]. Automated Software Engineering, 2012, 19(2): 201 - 230.

[4] SHEPPERD M, BOWES D, HALL T. Researcher bias: the use of machine learning in software defect prediction [J]. IEEE Transactions on Software Engineering, 2014, 42(11): 1092 - 1094.

[5] 蒋盛益,谢照青,余雯. 基于代价敏感的朴素贝叶斯不平衡数据分类研究 [J]. 计算机研究与发展,2011,48(S1):387 - 390. (JIANG S Y, XIE Z Q, YU W. Naïve Bayes classification algorithm based on cost sensitive for imbalanced data distribution [J]. Journal of Computer Research and Development, 2011, 48(S1): 387 - 390.)

[6] BACH M, WERNER A, ZYWIEC J, et al. The study of under-and over-sampling methods' utility in analysis of highly imbalanced data on osteoporosis [J]. Information Sciences, 2017, 384: 174 - 190.

[7] TORGO L, BRANCO P, RIBEIRO R P. Resampling strategies for regression [J]. Expert Systems, 2015, 32(3): 465 - 476.

[8] 戴翔,毛宇光. 跨机构的软件缺陷集成采样预测研究 [J]. 小型微型计算机系统,2015,36(8):1700 - 1705. (DAI X, MAO Y G. Research on cross-company software defect prediction based on integrated sampling and ensemble learning [J]. Journal of Chinese Computer Systems, 2015, 36(8): 1700 - 1705.)

[9] 戴翔,毛宇光. 基于集成混合采样的软件缺陷预测研究 [J]. 计算机工程与科学,2015,37(5):930 - 936. (DAI X, MAO Y G. Research on software defect prediction based on integrated sampling and ensemble learning [J]. Computer Engineering and Science, 2015, 37(5): 930 - 936.)

[10] 李勇. 结合欠抽样与集成的软件缺陷预测 [J]. 计算机应用,2014,34(8):2291 - 2294. (LI Y. Software defects prediction based on under-sampling and ensemble algorithm [J]. Journal of Computer Applications, 2014, 34(8): 2291 - 2294.)

[11] RATHORE S S, KUMAR S. Linear and non-linear heterogeneous ensemble methods to predict the number of faults in software systems [J]. Knowledge-Based Systems, 2017, 119: 232 - 256.

[12] CHEN M, MA Y. An empirical study on predicting defect numbers [EB/OL]. [2018-01-21]. http://pdfs.semanticscholar.org/43b5/eb8026719fe47338684060b843979981a0c7.pdf.

[13] WEYUKER E J, OSTRAND T J, BELL R M. Comparing the effectiveness of several modeling methods for fault prediction [J]. Empirical Software Engineering, 2010, 15(3): 277 - 295.

[14] HERBOLD S, TRAUTSCH A, GRABOWSKI J. Global vs. local models for cross-project defect prediction [J]. Empirical Software Engineering, 2016, 22(4): 1 - 37.

[15] ZHANG Y, LO D, XIA X, et al. Combined classifier for cross-project defect prediction: an extended empirical study [J]. Frontiers of Computer Science, 2018, 12(2): 280 - 296.

[16] HOSSEINI S, TURHAN B, MÄNTYLÄ M. A benchmark study on the effectiveness of search-based data selection and feature selection for cross project defect prediction [J]. Information and Software Technology, 2018, 95: 296 - 312.

[17] CHEN X, ZHAO Y, WANG Q, et al. MULTI: multi-objective effort-aware just-in-time software defect prediction [J]. Information and Software Technology, 2018, 93: 1 - 13.

[18] KAI M T. An instance-weighting method to induce cost-sensitive trees [J]. IEEE Transactions on Knowledge and Data Engineering, 2002, 14(3): 659 - 665.

[19] ZHOU Z H, LIU X Y. Training cost-sensitive neural networks with methods addressing the class imbalance problem [J]. IEEE Transactions on Knowledge and Data Engineering, 2006, 18(1): 63 - 77.

[20] KHOSHGOFTAAR T M, GELEYN E, NGUYEN L, et al. Cost-sensitive boosting in software quality modeling [C] // HASE '02: Proceedings of the 7th IEEE International Symposium on High Assurance Systems Engineering. Washington, DC: IEEE Computer Society, 2002: 51.

[21] CHEN L, FANG B, SHANG Z, et al. Tackling class overlap and imbalance problems in software defect prediction [J]. Software Quality Journal, 2018, 26(1): 97 - 125.

[22] ESTABROOKS A, JO T, JAPKOWICZ N. A multiple resampling method for learning from imbalanced data sets [J]. Computational Intelligence, 2010, 20(1): 18 - 36.

[23] BENNIN K E, KEUNG J, PHANNACHITTA P, et al. MAHAKIL: diversity based oversampling approach to alleviate the class imbalance issue in software defect prediction [J]. IEEE Transactions on Software Engineering, 2018, 44(6): 534 - 550.

(下转第 2659 页)

[6] ZHANG Z. Flexible camera calibration by viewing a plane from unknown orientations [C] // Proceedings of the 7th IEEE International Conference on Computer Vision. Washington, DC: IEEE Computer Society, 1999, 1: 666 – 673.

[7] TSAI R Y, LENZ R K. A new technique for fully autonomous and efficient 3D robotics hand/eye calibration [J]. IEEE Transactions on Robotics and Automation, 1989, 5(3): 345 – 358.

[8] KUMAR A, AHUJA N. Generalized radial alignment constraint for camera calibration [C] // ICPR '14: Proceedings of the 2014 22nd International Conference on Pattern Recognition. Washington, DC: IEEE Computer Society, 2014: 184 – 189.

[9] ANTUNES M, BARRETO J P, AOUADA D, et al. Unsupervised vanishing point detection and camera calibration from a single manhattan image with radial distortion [C] // Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition. Washington, DC: IEEE Computer Society, 2017: 6691 – 6699.

[10] PERS J, KOVACIC S. Nonparametric, model-based radial lens distortion correction using tilted camera assumption [J]. Kropatsch Proceedings of the Computer Vision Winter Workshop, 2002: 286 – 295.

[11] MARIOTTINI G L, PRATTICHIZZO D. EGT for multiple view geometry and visual servoing: robotics vision with pinhole and panoramic cameras [J]. IEEE Robotics and Automation Magazine, 2005, 12(4): 26 – 39.

[12] GONZALEZ-AGUILERA D, GOMEZ-LAHOZ J, RODRIGUEZ-GONZALVEZ P. An automatic approach for radial lens distortion correction from a single image [J]. IEEE Sensors Journal, 2011, 11(4): 956 – 965.

[13] BRADLEY D, HEIDRICH W. Binocular camera calibration using rectification error [C] // CRV '10: Proceedings of the 2010 Canadian Conference on Computer and Robot Vision. Washington, DC: IEEE Computer Society, 2010: 183 – 190.

[14] HUANG F S, CHEN L. CCD camera calibration technology based on the translation of coordinate measuring machine [J]. Applied Mechanics and Materials, 2014, 568/569/570: 320 – 325.

[15] POULIN-GIRARD A S, THIBAULT S, LAURENDEAU D. Influence of camera calibration conditions on the accuracy of 3D reconstruction [J]. Optics Express, 2016, 24(3): 2678.

[16] FATHI H, BRILAKIS I. A multi-step explicit stereo camera calibration approach to improve euclidean accuracy of large-scale 3D reconstruction [J]. Journal of Computing in Civil Engineering, 2016, 30(1): 1 – 33.

[17] ZHAO L, KONG L, WANG Y. Error analysis of binocular active hand-eye visual system on parallel mechanisms [C] // Proceedings of the 2008 International Conference on Information and Automation. Piscataway, NJ: IEEE, 2008: 95 – 100.

[18] DIMA E, SJOSTROM M, OLSSON R. Assessment of multi-camera calibration algorithms for two-dimensional camera arrays relative to ground truth position and direction [C] // Proceedings of the 2016 3DTV-Conference: the True Vision — Capture, Transmission and Display of 3D Video. Piscataway, NJ: IEEE, 2016:1 – 4.

This work is partially supported by the National Natural Science Foundation of China (51675324).

YANG Shangkun, born in 1992, M. S. candidate. His research interests include machine vision, 3D reconstruction.

WANG Yansong, born in 1971, Ph. D., professor. His research interests include vehicle system dynamics.

GUO Hui, born in 1981, Ph. D., associate professor. His research interests include vehicle vibration, noise control, design of intelligent piezoelectric mechanism.

WANG Xiaolan, born in 1985, Ph. D., lecturer. Her research interests include vehicle engineering, automatic navigation.

LIU Ningning, born in 1988, M. S., assistant experimentalist. His research interests include vehicle vibration, noise control.

(上接第 2643 页)

[24] TONG H, LIU B, WANG S. Software defect prediction using stacked denoising autoencoders and two-stage ensemble learning [J]. Information and Software Technology, 2017,96: 94 – 111.

[25] OKUTAN A, YILDIZ O T. Software defect prediction using Bayesian networks [J]. Empirical Software Engineering, 2014, 19(1): 154 – 181.

[26] WANG J, ZHANG H. Predicting defect numbers based on defect state transition models [C] // ESEM '12: Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement. New York: ACM, 2012: 191 – 200.

[27] RATHORE S S, KUMAR S. A decision tree regression based approach for the number of software faults prediction [J]. ACM SIGSOFT Software Engineering Notes, 2016, 41(1): 1 – 6.

[28] RATHORE S S, KUMAR S. An empirical study of some software fault prediction techniques for the number of faults prediction [J]. Soft Computing, 2017, 21(24): 7417 – 7434.

[29] RATHORE S S, KUMAR S. Towards an ensemble based system for predicting the number of software faults [J]. Expert Systems with Applications, 2017, 82: 357 – 382.

[30] OSTRAND T J, WEYUKER E J, BELL R M. Predicting the location and number of faults in large software systems [J]. IEEE Transactions on Software Engineering, 2005, 31(4): 340 – 355.

[31] YU X, LIU J, YANG Z, et al. Learning from imbalanced data for predicting the number of software defects [C] // ISSRE '17: Proceedings of the 2017 IEEE 28th International Symposium on Software Reliability Engineering. Washington, DC: IEEE Computer Society, 2017: 78 – 89.

[32] CHAWLA N V, BOWYER K W, HALL L O, et al. SMOTE: synthetic minority over-sampling technique [J]. Journal of Artificial Intelligence Research, 2002, 16(1): 321 – 357.

[33] YANG X, TANG K, YAO X. A learning-to-rank approach to software defect prediction [J]. IEEE Transactions on Reliability, 2015, 64(1): 234 – 246.

JIAN Yiheng, born in 1998. His research interests include software engineering, data mining.

YU Xiao, born in 1994, Ph. D. candidate. His research interests include software engineering, data mining.