计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.200400041

基于迁移学习和过采样技术的跨项目克隆代码一致性维护需求预测

欧阳鹏[1]　陆　璐[1,2]　张凡龙[3]　邱少健[4]
1 华南理工大学计算机科学与工程学院　广州 510641
2 华南理工大学梅州技术研究院　广东 梅州 514021
3 广东工业大学计算机学院　广州 510006
4 华南农业大学数学与信息学院　广州 510642
(939956752@qq.com)

摘　要　近年来,随着软件需求的不断增加,开发人员通过复用已有的代码向项目中引入了大量的克隆代码。随着软件版本的迭代和更新,克隆代码会发生变化,而克隆代码变化会导致额外的维护代价,并逐渐成为软件维护的负担。研究人员尝试利用机器学习方法开展克隆代码一致性维护需求预测研究,通过预测克隆代码的变化是否会导致额外的维护代价,来帮助软件质量保障团队更有效地分配维护资源,从而提高工作效率并降低运维成本。然而,在软件开发的初期阶段,软件项目往往没有经过充分的演化,缺少历史数据用于构建有效的预测模型,因此跨项目克隆代码一致性维护需求预测方法被提出。文中以减少跨项目数据分布差异为切入点,提出了基于迁移学习和过采样技术的跨项目克隆代码一致性维护需求预测方法 CPCCP+,旨在将测试集与数据集映射到核空间中,通过迁移主成分分析方法减小跨项目数据的分布差异,并对数据集的类不平衡问题进行处理,从而提高跨项目预测模型的性能。在实验数据集方面,选取了 7 个开源数据集,合计形成 42 组跨项目克隆代码一致性维护需求预测任务。将提出的方法与使用基分类器的方法进行比较,评估指标包含 Precision,Recall 和 F-Measure。实验结果表明,CPCCP+能更有效地进行跨项目克隆代码一致性维护需求的预测。

关键词:克隆代码;跨项目预测;一致性变化;迁移学习;过采样技术
中图法分类号　TP311

Cross-project Clone Consistency Prediction via Transfer Learning and Oversampling Technology

OUYANG Peng[1],LU Lu[1,2],ZHANG Fan-long[3] and QIU Shao-jian[4]
1 School of Computer Science and Engineering,South China University of Technology,Guangzhou 510641,China
2 Technology Research Institute,South China University of Technology,Meizhou,Guangdong 514021,China
3 School of Computers,Guangdong University of Technology,Guangzhou 510006,China
4 School of Mathematics and Informatics,South China Agricultural University,Guangzhou 510642,China

Abstract　In recent years,as software requirements increase,developers have introduced a large amount of clone code into the project by reusing existing code. As the software version is updated,the clone code changes and it may become a burden on software maintenance. Researchers have attempted to use the machine learning to conduct research on the prediction of clone code consistency,and help the software quality assurance team to allocate maintenance resources more effectively by predicting whether changes to cloned code will cause additional maintenance costs,thereby improving work efficiency and reducing maintenance costs. However,in the early stage of software development,software projects are often not fully evolved,and historical data is lacking for constructing an effective predictive model. Therefore,cross-project clone code consistency prediction methods are proposed. In this paper,we propose a cross-project clone code consistency prediction method via transfer learning and oversampling technology (CPCCP+). This method aims to match test set and training set into kernel space,reduce the distribution discrepancy of cross-project data by transfer component analysis,and alleviate the class imbalance issue to improve the performance of cross-project prediction model. In terms of experimental datasets,this paper selects seven open source datasets,which can form 42 combinations of cross-project clone code consistency prediction tasks totally. In terms of model performance comparison,the

到稿日期:2020-04-09　返修日期:2020-07-29　本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家自然科学基金(61370103);广州产学研基金(201902020004);梅州产学研项目(2019A0101019)
This work was supported by the National Natural Science Foundation of China (61370103),Industry-University-Research Foundation of Guangzhou (201902020004) and Industry-University-Research Project of Meizhou(2019A0101019).
通信作者:陆璐(lul@scut.edu.cn)

CPCCP+ proposed in this paper is compared with the method only using base classifier. The evaluation metrics include precision, recall and F-measure. The experimental results show that CPCCP+ can more effectively perform cross-project clone code consistency prediction.

Keywords Code clone, Cross-project prediction, Consistent change, Transfer learning, Oversampling technology

1 引言

克隆代码(Code Clone)是软件中彼此相似的代码片段。在软件开发过程中，软件开发人员通过复制和粘贴等操作向软件项目中引入了大量的克隆代码[1]。随着软件项目版本的迭代，克隆代码在其演化过程中往往会因开发人员修改而发生变化。由于克隆代码之间存在相似性，对某一克隆代码的修改可能会导致其他克隆代码发生变化，该现象被称为克隆代码一致性变化[2-3](Clone Consistent Change)。因此，对某一克隆代码进行修改后，可能需要对其他与之对应的克隆代码进行修改，以保障代码的正确性。克隆代码的一致性变化不仅会导致软件项目额外的维护代价，还会导致克隆代码的一致性违背缺陷[4-5](若开发人员遗忘了对其他需要一致性变化的克隆代码进行一致的修改)，从而影响软件质量。因此，克隆代码变化是影响软件质量、可理解性和可维护性的一个重要因素[6]。如果能够利用机器学习的方法对克隆代码一致性维护需求进行预测，给出需要维护的克隆代码模块，则能够帮助软件质量保障团队更有效地分配维护资源，从而提高工作效率和降低运维成本[7-8]。本文研究的克隆代码一致性维护需求预测有两种不同的结果(二分问题)，即满足一致性维护需求或不满足维护需求。其中，满足一致性维护需求的克隆代码片段在未来的演化过程中可能会引发一致性变化，建议开发人员采取相应的操作对其进行维护，如对其进行重构[9]。

在该技术的实际应用过程中，研究者们[10]发现，上述基于机器学习的软件克隆代码一致性维护需求预测技术容易遇到软件冷启动和标注数据稀缺的问题。为解决此问题，Zhang等[10]提出了采用跨项目数据预测克隆代码的一致性维护需求，认为软件开发流程的差异不大，且如果采用一致的属性特征表示克隆代码，则源项目的数据训练的克隆代码一致性维护需求预测模型可以直接用于目标项目中。然而，由于软件的规模、功能和编码规则不同，不同项目中的数据存在概率分布差异，如图1所示。因为不同项目中的数据存在分布差异，所以在源项目中学习的分类器不一定能很好地迁移到目标项目中，如果项目间数据形成的数据分布可以适配，则可增强跨项目克隆代码一致性维护需求预测方法的性能。

与很多二分类问题的数据集一样，克隆代码一致性维护需求数据集同样遭遇了类间不平衡问题，即在多个软件项目中，无一致性维护需求的克隆代码数量远超过有一致性维护需求的克隆代码数量。为了解决二分类问题中的类间不平衡问题，研究者们在数据层面和算法层面均提出了一些方法。数据层次上的方法包括一系列重采样技术，如随机上采样、随机下采样、SMOTE采样等[11-12]；算法层次上则有代价敏感学习技术等。这些方法均能有效改善预测模型的预测性能。

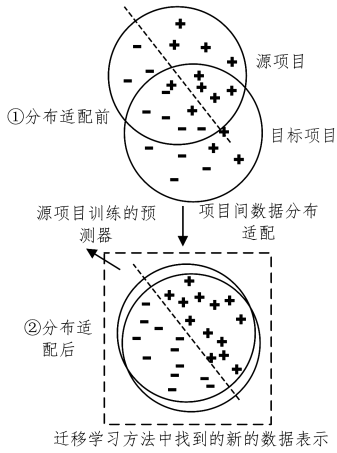


图1 分布适配可提高跨项目克隆代码一致性维护需求的预测性能

Fig. 1 Distribution adaptation can improve performance of cross-project clone consistency prediction

基于迁移学习[13]中数据分布适配的思想和类不平衡问题的普遍性，本文提出了基于迁移学习和过采样技术的跨项目克隆代码一致性维护需求预测方法(Cross-Project Clone Consistency Prediction via transfer Learning and Oversampling Technology，CPCCP+)，用于提高跨项目软件克隆代码一致性维护需求预测的性能。首先，CPCCP+分别从源项目和目标项目数据中收集训练数据集和测试数据集；其次，CPCCP+分别对数据集中的克隆创建实例和克隆变化实例进行特征表示；进一步地，对数据集进行随机过采样处理；CPCCP+将数据集映射到再生核希尔伯特空间中，以最大均值差异[14]作为分布之间差异的度量，通过特征空间映射的方式将不同项目的数据分布进行适配，以减小源项目数据和目标项目数据的分布差异，在此基础上得到映射空间中新的训练集和测试集；最后，使用映射空间中的训练集训练预测分类器，用以执行跨项目预测任务。

2 相关术语定义

2.1 克隆片段的一致性变化

设有两个克隆代码片段F_1和F_2，在后期分别被修改为F_1'和F_2'。对于一个接近于0的阈值ε，如果克隆代码F_1和F_2的变化满足以下两个条件，则称此变化为一致性变化(Consistent Change)；否则称其为不一致性变化(Inconsistent Change)。

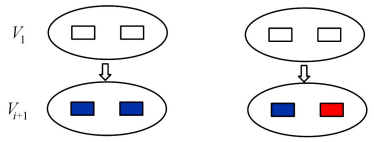
条件1：$TextSim(F_i, F_i') < 1, \forall i \in \{1,2\}$

条件2：$|TextSim(F_1, F_1') - TextSim(F_2, F_2')| < \varepsilon$

其中，$TextSim$表示代码之间的相似性度量方法，其计算

方式请参见文献[15]。

具有一致性变化特征的两个(或多个)克隆代码片段具体表现为:彼此十分相似,且随着版本的迭代,发生的变化也十分相似,如同时增加或减少一条相似的语句。若发生的变化不相似,例如一个克隆代码片段增加一条语句,而另外一个克隆代码片段减少一条语句,则称这两个克隆代码片段发生了不一致变化。如图 2 所示,变蓝代表增加一条语句,变红则代表减少一条语句。



(a)一致性变化 (b)不一致性变化

图 2 一致性变化与不一致性变化(电子版为彩色)

Fig. 2 Consistent change and inconsistent change

2.2 克隆组与克隆家系

一个软件项目通常会有多个彼此相似的代码片段,这些代码片段的集合即为一个克隆组(Clone Group,CG)。而一个软件项目中通常也会有多个不同的克隆组。克隆家系(Clone Genealogy,CGE)模型则描述了一个克隆组随着软件系统进行演化的过程,即不同软件版本中同一克隆组的演化序列,该模型由 Kim 等[9]提出,是当前用于描述克隆代码演化情况的最好模型。演化模式主要包括静态、增加、减少、分离、合并、一致和不一致等。

2.3 克隆实例

在一个克隆组的克隆家系中,克隆家系的根节点克隆组为克隆创建实例,发生变化的克隆组为克隆变化实例。本文将克隆创建实例和克隆变化实例统称为克隆实例。

2.4 克隆一致性维护需求

给定软件项目版本 v 中的一个克隆实例 CG,如果在其版本 k 中存在一个克隆实例 $CG'(k>v)$ 满足以下条件,则称该克隆实例满足克隆一致性维护需求;反之则称该克隆实例不满足克隆一致性维护需求。

(1)在 CG' 中至少存在两个克隆片段在其克隆家系中可以映射到克隆实例 CG 中;

(2)在 CG' 具有克隆片段一致性变化。

本文开展的克隆代码一致性维护需求预测研究,就是给定一个克隆实例,预测该克隆实例是否满足克隆一致性维护需求。

3 基于迁移学习和过采样技术的跨项目克隆代码一致性维护需求预测方法

为解决不同项目中数据的分布差异问题和数据集的类间不平衡问题,本文提出了基于迁移学习和过采样技术的跨项目克隆代码一致性维护需求预测方法 CPCCP+。图 3 给出了该方法的框架,其主要包含 5 个步骤:1)收集训练数据集和测试数据集;2)表示克隆实例;3)采用随机过采样技术对数据集进行处理;4)采用迁移主成分分析方法 TCA[16]适配跨项目软件数据的分布;5)执行跨项目克隆代码一致性维护需求预测。

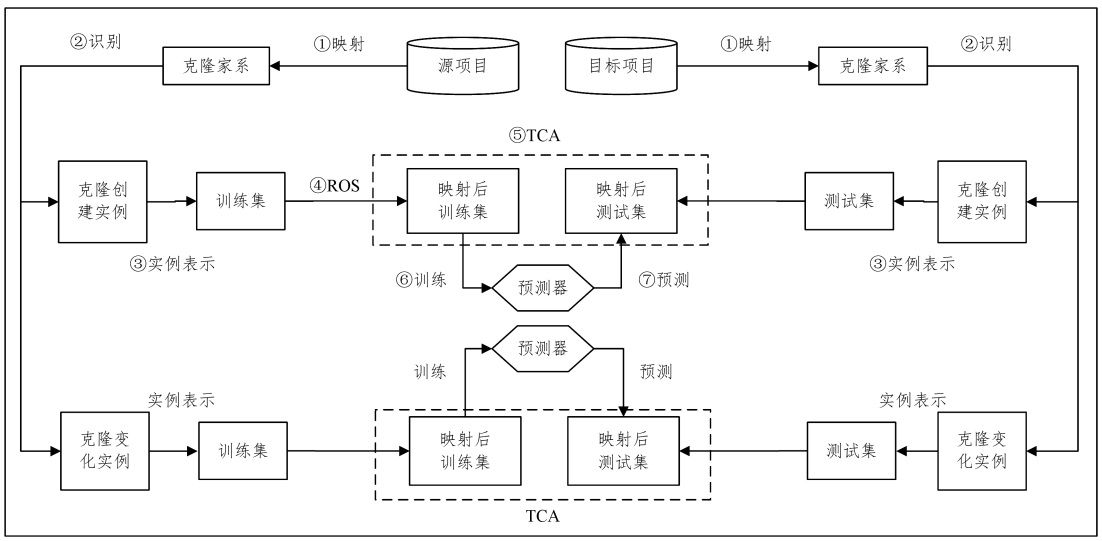


图 3 基于迁移学习和过采样技术的跨项目克隆代码一致性维护需求预测方法

Fig. 3 Cross-project clone consistency prediction method via transfer learning and oversampling technology

3.1 收集训练数据集和测试数据集

本文选定一个软件项目作为目标项目,提取其中的克隆实例作为测试数据集,同时选用另一个非目标项目的软件项目克隆实例作为训练数据集。在本步骤中,首先下载软件项目所有版本的源码,并使用 NiCad[17]来检测软件版本中的所有克隆,通过在相邻版本的克隆组之间进行映射来构建克隆家系;其次,通过 2.3 节的内容识别软件项目中所有的克隆实例;最后,通过遍历实例所在克隆家系的演化情况,标识克隆实例的一致性维护需求,如果克隆实例在演化过程中发生了一致性变化,则将其标注为满足一致性维护需求,否则为不满足维护需求。

3.2 克隆实例表示

一个克隆实例中包含两种不同类型的克隆代码片段,即被复制的克隆代码片段和被粘贴的克隆代码片段。针对被复

制的克隆代码片段，开发人员往往不会对其进行修改；而对于被粘贴的克隆代码片段，由于两类片段所处的代码上下文环境可能不同，开发人员可能会对其进行一定程度的修改。因此，对于克隆实例的表示，除了提取反映克隆代码本身的代码属性外，还需提取反映克隆代码片段之间区别和联系的上下文属性。

本文针对克隆创建实例，提取其代码和上下文两组属性：1）代码属性特征包括克隆代码粒度、Halstead 属性[18]、结构属性、参数访问数量、总函数调用次数、本地函数调用次数、库函数调用次数和其他调用次数等；2）上下文属性包括代码相似度、局部克隆标识、文件名相似度、文件名相似度标识、方法名相似度、总参数名相似度、最大参数名相似度、总参数类型相似度、块信息标识等。

针对克隆变化实例，由于克隆代码在演化中是不断发展变化的，仅根据单一版本的特征来反映重构克隆代码是不够的[19]，因此除了提取代码、上下文属性外，还需提取演化属性：1）代码属性特征，包括克隆粒度、代码平均行数、Halstead 属性平均值、结构属性平均值、总函数调用平均次数、本次函数调用平均次数、库函数调用平均次数和其他调用平均次数；2）上下文属性，包括代码相似度平均、最大参数名相似度平均、总参数类型相似度平均和块信息标识；3）演化属性，包括变化实例寿命、历史演化模式统计、当前演化模式和历史变化统计等。

3.3 采用随机过采样技术处理训练数据

通过 3.2 节所述步骤，可分别得到两类数据集：1）克隆创建实例组成的训练集和测试集；2）由克隆变化实例组成的训练集和测试集。由于在克隆变化实例组成的数据集中需进行一致性维护的实例数量和无需进行一致性维护的实例数量较为平衡，而在克隆创建实例组成的数据集中两者数量相差较大，因此此步骤只针对克隆创建数据集。采用随机过采样（Random Over-Sampling，ROS）对克隆创建数据集进行处理，该方法从少数类的样本中进行随机采样来增加新的样本，使得数据集中需进行一致性维护和无需进行一致性维护两类实例的数量达到平衡。根据 Khoshgoftaar 等[20]的建议，将该平衡比设置为 35%/65%。

3.4 采用迁移主成分分析方法适配数据分布

通过以上步骤，可得到由源项目形成的类间较为平衡的训练数据集和由目标项目所形成的测试数据集。然而，由于不同项目中的数据存在概率分布差异，训练集中学习的预测器往往不能直接运用于测试集的预测任务中。因此，本文引入迁移主成分分析方法，将测试集与数据集映射到再生核希尔伯特空间中，通过迁移主成分分析方法 TCA，借助最大均值差异（Maximum Mean Discrepancy，MMD）作为分布之间差异的度量，通过特征空间映射的方式将不同项目的数据分布进行适配，以减小源项目数据和目标项目数据的分布差异；在此基础上得到映射空间中新的训练集和测试集，并将其作为跨项目克隆代码一致性维护需求预测的数据集。

具体地，迁移主成分分析方法 TCA 尝试使用最大均值差异在再生核希尔伯特空间中跨领域学习可迁移主成分，在这些跨领域主成分形成的子空间 ϕ 中，跨项目的数据属性得以保留。基于这个子空间 ϕ 中数据表示得到的跨项目数据，可以采用标准的机器学习方法训练出源项目中的一致性维护需求预测模型，并将其运用到目标项目中。

TCA 方法的主要目的是找到合适的 ϕ，以减小源项目数据和目标项目数据的分布差异。由于 KL 散度等度量分布差异的方法需要估算分布的概率密度，是一种参数化的方法，增加了求解的难度，因此 TCA 选用了一种非参数化的方法（MMD）来度量分布的差异，其用于计算映射后源域和目标域的均值之差。MMD 的计算方式如下：

$$MMD(X_{\mathrm{src}},X_{\mathrm{tar}})=\left\|\frac{1}{n_1}\sum_{i=1}^{n_1}\phi(x_{src_i})-\frac{1}{n_2}\sum_{i=1}^{n_2}\phi(x_{tar_i})\right\|_{\mathcal{H}}$$

其中，X_{src} 和 X_{tar} 分别表示源项目和目标项目的数据，n_1 和 n_2 分别表示源项目和目标项目的数据数量，$\phi(\cdot)$ 表示再生核希尔伯特空间上的映射。在解析该距离度量的过程中，由于寻找合适的映射函数 $\phi(\cdot)$ 比较困难，且不同源项目和目标项目组合的映射函数也可能不同，因此 TCA 引入了核矩阵 $\boldsymbol{K}$，使得寻找合适映射函数 $\phi(\cdot)$ 的问题转变为寻找合适的核函数的问题。核矩阵 $\boldsymbol{K}$ 的形式如下：

$$\boldsymbol{K}=\begin{pmatrix}K_{\mathrm{src},\mathrm{src}},K_{\mathrm{src},\mathrm{tar}}\\K_{\mathrm{tar},\mathrm{src}},K_{tar,tar}\end{pmatrix}$$

同时引入半正定矩阵 $\boldsymbol{L}$：

$$L_{ij}=\begin{cases}\dfrac{1}{n_1^2}, & x_i,x_j\in X_{\mathrm{src}}\\[2ex]\dfrac{1}{n_2^2}, & x_i,x_j\in X_{\mathrm{tar}}\\[2ex]-\dfrac{1}{n_1n_2}, & \text{otherwise}\end{cases}$$

MMD 的计算可变化为如下形式：

$$MMD(X_{\mathrm{src}},X_{\mathrm{tar}})=\mathrm{tr}(\boldsymbol{KL})-\lambda\mathrm{tr}(\boldsymbol{K})$$

其中，$\mathrm{tr}(\cdot)$ 是求矩阵的迹，即矩阵对角线元素的和；λ 是可调的正则化参数。

此时，MMD 的最小化计算是一个半定规划（Semi-definite Programming）问题，求解效率较低。TCA 方法进一步地通过降维的方式构造结果，最终 TCA 的优化目标转化为了：

$$\min_{\boldsymbol{W}}\ \mathrm{tr}(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{KLKW})+\lambda\mathrm{tr}(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{W})$$

$$\text{s.t. } \boldsymbol{W}^{\mathrm{T}}\boldsymbol{KHKW}=\boldsymbol{I}_m$$

其中，λ 为控制参数矩阵 $\boldsymbol{W}$ 的复杂度的权重参数，$\boldsymbol{H}=\boldsymbol{I}_{n_1+n_2}-1/(n_1+n_2)\mathbf{1}\mathbf{1}^{\mathrm{T}}$ 为中心矩阵，$\boldsymbol{I}_m\in\mathrm{R}^{m\times m}$ 为单位矩阵。最小化过程中的约束条件用于维持各自的数据特征。

求解得到的 $\boldsymbol{W}$ 即可表示为子空间 ϕ，将源项目和目标项目数据映射到该空间后，可得到子空间 ϕ 中分布差异较小的源项目和目标项目数据集。

3.5 跨项目克隆代码一致性维护需求预测

本节使用过采样技术和 TCA 方法处理过的源项目克隆实例（训练集）训练机器学习预测模型，并预测目标项目中克

隆实例(测试集)的一致性维护需求。本文将分别对两种不同时刻的克隆代码(克隆创建实例和克隆变化实例)的一致性维护需求进行预测。

预测结果包含两种,即需要一致性维护和不需要一致性维护。如果预测结果为"需要",开发人员须考虑克隆代码的一致性变化,慎重执行相应的操作;若预测结果为"不需要",开发人员则可较为自由地引入或者更改克隆代码[21]。

4 实验设置与结果分析

4.1 数据集

为了评估 CPCCP+模型的性能,本文在 7 个 Java 开源软件上进行实验。表 1 列出了软件项目的基本信息,包含类型、测试软件、实例数和需维护率。类型分别有克隆创建实例和克隆变化实例两种。需维护率则是需要进行一致性维护的克隆实例占所有克隆实例的比率。为了验证 CPCCP+方法的通用有效性,数据集由不同大小和需维护率的软件项目组成(克隆创建实例数据集的规模为 34～3 666 个,需维护率为 11.53%～47%;克隆变化实例的数量范围为 10～1 040 个,需维护率为 20%～76.19%)。7 个软件项目可组成 42 对跨项目克隆创建实例的代码一致性维护需求预测任务和 42 对跨项目克隆变化实例的代码一致性维护需求预测任务(若选定 ArgoUML 作为目标项目提取测试集,则分别使用其余 6 个项目的数据作为源项目,用于提取训练集)。

表 1 实验数据集

Table 1 Datasets

类型	测试软件	实例数	需维护率/%
克隆创建实例	ArgoUML	3 340	22.93
	carol	100	47.00
	dnsjava	34	20.59
	jEdit	633	11.53
	jFreeChart	3 366	40.20
	jabref	960	20.63
	Tuxguitar	1 429	28.90
克隆变化实例	ArgoUML	427	32.55
	carol	42	76.19
	dnsjava	10	20.00
	jEdit	159	50.94
	jFreeChart	1 040	56.54
	jabref	171	48.54
	Tuxguitar	354	74.29

4.2 评估方法

本文实验采用 Precision, Recall 和 F-measure[22] 3 个指标来评估方法的性能。

在本文二分类的跨项目克隆代码一致性维护需求预测任务中,可以获得 4 个测试数据结果:将真正需维护的实例分类为了需维护的(真阳性,TP);将真正无需维护的实例分类为了需维护(假阳性,FP);将真正需维护的实例分类为了无需维护的(假阴性,FN);将真正无需维护的实例分类为了无需维护的(真阴性,TN)。基于这 4 个结果,可以定义精确率(Precision)和召回率(Recall)。

精确率为模型正确标注为需维护的实例数量与真正需维护的实例数量的比例。精确率也称查准率,其值越大越好。

$$Precision=\frac{TP}{TP+FP}$$

召回率定义为用于评估模型正确标注缺陷实例的比例。召回率又称为查全率,其值越大越好。

$$Recall=\frac{TP}{TP+FN}$$

实际上,在精确率和召回率之间存在折衷,仅使用精确率或召回率来比较预测模型的性能存在困难。二分类预测通常使用 F-Measure 作为模型评价的指标,它是精确率和召回率的调和平均值。F-Measure 值为 1 时,达到最佳的精确率和召回率,在 0 处则为最差值。

$$F\text{-}Measure=\frac{2\times Precision\times Recall}{Precision+Recall}$$

4.3 实验设置

实验将仅采用基分类器的跨项目克隆代码一致性维护需求预测方法[10]作为 Baseline,并将本文提出的 CPCCP+方法与其进行比较。本文分别选用 K 最近邻(K-Nearest Neighbor,KNN)、逻辑回归(Logistic Regression,LR)、支持向量机(Support Vector Machine,SVM)和随机森林(Random Forest,RF)4 种基分类器进行实验,通过调用 Matlab 中对应的机器学习算法来实现各基分类器算法。当实现 CPCCP+算法时,本文首先采用 TCA 对数据进行映射处理,然后将结果放入 Matlab 中开展跨项目克隆代码一致性维护需求的预测。

4.4 实验结果及分析

本文采用 Precision, Recall 和 F-Measure 3 个指标对 Baseline 和 CPCCP+两种方法的预测性能进行度量。表 2 和表 3 分别列出了克隆创建实例的跨项目预测效果和克隆变化实例的跨项目预测效果。平均值(AVG)的比较中,如果 Baseline 比 CPCCP+有更好的效果,则在较好的结果后追加星号;如果 CPCCP+效果好于 Baseline,则较好的结果加粗显示。

从表 2 中可以观察到:1)CPCCP+在多数预测任务中可以获得更好的预测性能;2)在最后一列平均性能的比较中,当 CPCCP+性能弱于 Baseline 时,性能差异是极小的(仅有 0.001 的差距),而在 CPCCP+性能高于 Baseline 时,其性能明显优于 Baseline;3)在 4 个基分类器的比较中,SVM 和 RF 具有较好的预测性能。表 2 所呈现的结果一定程度上说明,在克隆创建实例的跨项目预测任务中,结合了迁移主成分分析的方法和过采样技术后所得到的 CPCCP+,在跨项目克隆代码一致性维护需求预测任务中具有更好的性能。

从表 3 中可以观察到:1)CPCCP+在多数预测任务中可以获得更好的预测性能;2)在最后一列平均性能的比较中,CPCCP+性能大多高于 Baseline 方法(仅在使用 LR 作为基分类器时的 *F-Measure* 比较中略低 0.003);3)在 4 个基分类器的比较中,LR 具备较好的预测性能。表 3 中的结果一定程度上说明,在克隆变化实例的跨项目预测任务中,CPCCP+在跨项目预测任务中具备更好的性能。

综上所述,在解决源项目和目标项目数据分布差异的基础上,在本文关注的克隆创建实例的跨项目预测和克隆变化

实例的跨项目预测任务的实验中,CPCCP+相比 Baseline 方法具有更好的预测性能。

表 2 克隆创建实例的跨项目预测性能

Table 2 Performance of cross-project prediction for clone createing instances

评估指标	分类器	方法	To ArgoUML	To carol	To dnsjava	To jEdit	To jFreechart	To jabref	To Tuxguitar	AVG
平均 *Precision*	KNN	Baseline	0.670	0.485	0.706	0.696	0.550	0.715	0.638	0.637*
		CPCCP+	0.673	0.552	0.735	0.737	0.564	0.609	0.581	0.636
	LR	Baseline	0.685	0.517	0.706	0.770	0.556	0.723	0.674	0.661*
		CPCCP+	0.689	0.538	0.750	0.745	0.569	0.717	0.611	0.660
	SVM	Baseline	0.717	0.522	0.755	0.812	0.594	0.735	0.667	0.686
		CPCCP+	0.739	0.567	0.765	0.806	0.589	0.723	0.668	**0.694**
	RF	Baseline	0.712	0.480	0.730	0.760	0.567	0.728	0.673	0.664
		CPCCP+	0.727	0.558	0.770	0.835	0.583	0.737	0.660	**0.696**
平均 *Recall*	KNN	Baseline	0.642	0.421	0.704	0.799	0.497	0.676	0.602	0.620
		CPCCP+	0.676	0.526	0.742	0.801	0.501	0.628	0.568	**0.635**
	LR	Baseline	0.634	0.479	0.660	0.809	0.470	0.695	0.635	0.626*
		CPCCP+	0.650	0.439	0.678	0.832	0.491	0.694	0.594	0.625
	SVM	Baseline	0.626	0.343	0.647	0.795	0.404	0.673	0.548	0.577
		CPCCP+	0.647	0.480	0.675	0.820	0.418	0.634	0.542	**0.602**
	RF	Baseline	0.651	0.337	0.712	0.815	0.497	0.677	0.612	0.614
		CPCCP+	0.672	0.454	0.665	0.829	0.492	0.647	0.569	**0.618**
平均 *F-measure*	KNN	Baseline	0.649	0.406	0.694	0.734	0.493	0.688	0.607	0.610
		CPCCP+	0.666	0.485	0.721	0.762	0.491	0.609	0.561	**0.613**
	LR	Baseline	0.644	0.437	0.674	0.781	0.465	0.692	0.623	0.617*
		CPCCP+	0.653	0.443	0.703	0.764	0.488	0.686	0.572	0.616
	SVM	Baseline	0.654	0.392	0.695	0.795	0.450	0.671	0.580	0.605
		CPCCP+	0.676	0.477	0.706	0.785	0.450	0.666	0.589	**0.621**
	RF	Baseline	0.667	0.360	0.704	0.773	0.468	0.688	0.613	0.610
		CPCCP+	0.680	0.431	0.704	0.815	0.459	0.682	0.596	**0.624**

表 3 克隆变化实例的跨项目预测性能

Table 3 Performance of cross-project prediction for clonechanging instances

评估指标	分类器	方法	To ArgoUML	To carol	To dnsjava	To jEdit	To jFreechart	To jabref	To Tuxguitar	AVG
平均 *Precision*	KNN	Baseline	0.526	0.417	0.550	0.492	0.478	0.486	0.430	0.483
		CPCCP+	0.460	0.595	0.400	0.526	0.513	0.514	0.529	**0.505**
	LR	Baseline	0.441	0.480	0.517	0.509	0.518	0.562	0.500	0.504
		CPCCP+	0.459	0.524	0.517	0.524	0.524	0.540	0.504	**0.513**
	SVM	Baseline	0.481	0.480	0.533	0.513	0.498	0.505	0.483	0.499
		CPCCP+	0.463	0.504	0.500	0.517	0.507	0.517	0.498	**0.501**
	RF	Baseline	0.501	0.405	0.483	0.494	0.488	0.488	0.446	0.472
		CPCCP+	0.502	0.512	0.567	0.526	0.514	0.514	0.462	**0.514**
平均 *Recell*	KNN	Baseline	0.601	0.598	0.711	0.500	0.394	0.507	0.575	0.555*
		CPCCP+	0.443	0.625	0.541	0.533	0.533	0.556	0.583	0.545
	LR	Baseline	0.532	0.622	0.733	0.517	0.530	0.582	0.612	0.590
		CPCCP+	0.566	0.637	0.685	0.553	0.541	0.590	0.610	**0.597**
	SVM	Baseline	0.604	0.457	0.519	0.416	0.406	0.439	0.435	0.468
		CPCCP+	0.502	0.524	0.557	0.519	0.461	0.528	0.502	**0.513**
	RF	Baseline	0.587	0.593	0.620	0.478	0.488	0.484	0.602	0.550
		CPCCP+	0.576	0.545	0.717	0.536	0.468	0.519	0.605	**0.567**
平均 *F-measure*	KNN	Baseline	0.478	0.399	0.569	0.438	0.379	0.423	0.380	0.438
		CPCCP+	0.386	0.585	0.357	0.484	0.425	0.448	0.498	**0.455**
	LR	Baseline	0.411	0.499	0.557	0.472	0.487	0.528	0.495	0.493*
		CPCCP+	0.433	0.518	0.509	0.481	0.478	0.499	0.512	0.490
	SVM	Baseline	0.393	0.423	0.484	0.399	0.391	0.378	0.411	0.411
		CPCCP+	0.381	0.468	0.441	0.416	0.402	0.418	0.442	**0.424**
	RF	Baseline	0.461	0.393	0.502	0.439	0.413	0.422	0.416	0.435
		CPCCP+	0.471	0.490	0.552	0.464	0.444	0.468	0.447	**0.477**

结束语 为了预测跨项目的克隆代码的一致性变化,本文引入了迁移学习的方法,提出了基于迁移学习和过采样技术的跨项目克隆代码一致性维护需求预测方法 CPCCP+,旨在通过对跨项目数据分布差异的适配和对类间不平衡问题的处理,来提高跨项目克隆代码一致性维护需求预测的性能。本文方法首先从源项目和目标项目数据中收集训练数据集和测试数据集;其次对数据集进行过采样处理;然后将测试集与训练集映射到再生核希尔伯特空间中,通过迁移主成分分析 TCA 方法来减小源项目数据和目标项目数据的分布差异;最后使用映射空间中的训练集学习预测器,用以执行跨项目克

隆代码一致性维护需求预测任务。与使用4种基分类器的Baseline方法的对比实验说明，CPCCP+能提高跨项目克隆代码一致性维护需求预测的准确性。未来，将探索更多的迁移学习方法，并进行对比研究；同时，在本文CPCCP+方法的基础上，进一步研究TCA中各参数调节和设置的方法；此外，将针对跨项目克隆代码一致性维护需求预测中分类不均衡的问题进行进一步的调研，并提出更加合理的解决方案。

参 考 文 献

[1] SAJNANI H, SAINI V, SVAJLENKO J, et al. SourcererCC: Scaling code clone detection to big-code[C]// 2016 IEEE/ACM 38th International Conference on Software Engineering. 2016: 1157-1168.

[2] KRINKE J. A study of consistent and inconsistent changes to code clones[C]// 14th working Conference on Reverse Engineering. 2007: 170-178.

[3] BETTENBURG N, SHANG W, IBRAHIM W M, et al. An empirical study on inconsistent changes to code clones at the release level[J]. Science of Computer Programming, 2012, 77(6): 760-776.

[4] WAGNER S, ABDULKHALEQ A, KAYA K, et al. On the relationship of inconsistent software clones and faults: an empirical study[C]// 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering. 2016: 79-89.

[5] JUERGENS E, DEISSENBOECK F, HUMMEL B, et al. Do code clones matter? [C] // Proceedings of the 31st International Conference on Software Engineering. 2009: 485-495.

[6] WHITE M, TUFANO M, VENDOME C, et al. Deep learning code fragments for code clone detection[C]// Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering. 2016: 87-98.

[7] ZHANG F, KHOO S C, SU X. Predicting consistent clone change[C]// 2016 IEEE 27th International Symposium on Software Reliability Engineering. 2016: 353-364.

[8] ZHANG F, KHOO S C, SU X. Machine-Learning Aided Analysis of Clone Evolution[J]. Chinese Journal of Electronics, 2017, 26(6): 1132-1138.

[9] KIM M, SAZAWAL V, NOTKIN D, et al. An empirical study of code clone genealogies [J]. Acmfigsoft Software Engineering Notes, 2005, 30(5): 187-196.

[10] ZHANG F. Research on analysis and consistency maintenance of code clone based on software evolution[D]. Harbin: Harbin Institute of Technology, 2017.

[11] KAMEIY, MONDEN A, MATSUMOTO S, et al. The effects of over and under sampling on fault- prone module detection[C]// Proceedings of the First International Symposium on Empirical Software Engineering and Measurement. IEEE, 2007: 196-204.

[12] SEIFFERT C, KHOSHGOFTAAR T M, VAN HULSE J. Improving software-quality predictions with data sampling and boosting[J]. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 2009, 39(6): 1283-1294.

[13] PAN S J, YANG Q. A survey on transfer learning[J]. IEEE Transactions on Knowledge and Data Engineering, 2010, 22(10): 1345-1359.

[14] BORGWARDT K M, GRETTON A, RASCG M J, et al. Integrating structured biological data by kernel maximum mean discrepancy[J]. Bioinformatics, 2006, 22(14): e49-e57.

[15] ZHANG F, KHOO S, SU X. Predicting change consistency in a clone group[J]. Journal of Systems and Software, 2017, 134: 105-119.

[16] PAN S J, TSANG I W, KWOK J T, et al. Domain adaptation via transfer component analysis[J]. IEEE Transactions on Neural Networks, 2011, 22(2): 199-210.

[17] ROY C K, CORDY J R. NICAD: Accurate detection of near-miss intentional clones using flexible pretty-printing and code normalization[C]// Proceedings of IEEE International Conference on Program Comprehension. 2008: 172-181.

[18] HALSTEAD M H. Elements of software science[M]. New York: Elsevier, 1977.

[19] SHE R, ZHANG L. Method for Identifying and Recommending Reconstructed Clones Based on Software Evolution History[J]. Computer Science, 2019, 46(8): 224-232.

[20] KHOSHGOFTAAR T M, SEIFFERT C, VAN HULSE J, et al. Learning with limited minority class data[C]// Proceedings of the International Conference on Machine Learning and Applications. IEEE, 2007: 348-353.

[21] SU X, ZHANG F. A Survey for Management-Oriented Code Clone Research [J]. Chinese Journal of Computers, 2018, 41(3): 628-651.

[22] HHAN J, PEI J, KAMBER M. Data mining: concepts and techniques[M]. NewYork: Elsevier, 2011.

OUYANG Peng, born in 1996, postgraduate. His main research interests include software reliability maintenance and transfer learning.

LU Lu, born in 1971, Ph.D, professor, is a member of China Computer Federation. His main research interests include software engineering, software testing and software architecture design.