计算机科学
COMPUTER SCIENCE

结合数据选择的多源跨项目缺陷预测

邓建华, 王炜

引用本文

邓建华, 王炜. 结合数据选择的多源跨项目缺陷预测[J]. 计算机科学, 2022, 49(11A): 210800160-7.

DENG Jian-hua, WANG Wei. Multi-source Cross-project Defect Prediction with Data Selection[J]. Computer Science, 2022, 49(11A): 210800160-7.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.210800160

结合数据选择的多源跨项目缺陷预测

邓建华　王　炜
云南大学软件学院　昆明 650091
(1765881146@qq.com)

摘　要　多源跨项目缺陷预测(Multi-sources Cross Project Defect Prediction,MCPDP)旨在使用多个来自其他项目(源项目)的历史数据来预测目标项目中软件模块出现缺陷的可能性。该研究解决了缺陷预测建模的冷启动问题,为新建软件或缺乏历史数据的软件系统建立缺陷预测模型提供了解决方案。对于进一步提高跨项目缺陷预测的准确性,源数据选择被认为是一条有效途径。因此,文中对数据选择的多源跨项目缺陷预测方法进行了研究,该方法包括两个步骤:1)源数据特征对齐;2)改进最大均值测度,实现源数据筛选。为了验证提出的方法的有效性,在 AEEEM,Relink,NASA,SOFTLAB 这 4 个公开数据集进行实验,结果表明所提方法在 *F*-measure 指标上比基线方法分别提高了 4%和 5%,证明该方法具有较好的性能。

关键词:多源域;跨项目;缺陷预测;数据选择;特征对齐
中图法分类号　TP311

Multi-source Cross-project Defect Prediction with Data Selection

DENG Jian-hua and WANG Wei
School of Software,Yunnan University,Kunming 650091,China

Abstract　Multi-sources cross project defect prediction(MCPDP) aims to use multiple historical data from other projects(source projects) to predict the likelihood of defects in software modules in the target project. The research solves the cold start problem of defect prediction modeling and provides a solution for establishing defect prediction model for new software or software system lacking historical data. Source data selection is considered to be an effective way to further improve the accuracy of cross-project defect prediction. Therefore,a multi-source cross-project defect prediction method for data selection is studied in this paper. The method includes two steps:1) feature alignment of source data;2) improve the maximum mean measure to realize source data screening. In order to verify the effectiveness of the proposed method,experiments are carried out on four public data sets,namely AEEEM,Relink,NASA and SOFTLAB. The results show that the proposed method improves the *F*-measure index by 4% and 5% respectively compared with the baseline method,which proves that the proposed method has good performance.

Keywords　Multi-source domain,Across projects,Defect prediction,Data selection,Feature alignment

1　引言

随着城市化进程的快速发展,缺陷预测结果对及时发现软件缺陷、提高软件质量、优化测试资源配置、节省维护成本具有重要意义[1]。为了解决缺陷预测建模过程中获取带标签的训练数据开销大和新开发软件缺乏历史数据,导致缺陷预测建模面临"冷启动"的问题[2-3],研究人员将迁移学习引入缺陷预测,提出了跨项目缺陷预测模型[4-6],该研究旨在使用一个或多个源项目的缺陷数据构建缺陷预测模型。由于开发语言、编程风格、设计模式、项目管理方式等的不同,源数据与目标数据间存在较大差异,导致异构性(Heterogeneity)广泛地存在于源数据与目标数据之间[7],这使得跨项目缺陷预测准确性低于项目内缺陷预测模型。

文献[8-9]指出目标数据和源数据之间的相似性极大地影响了缺陷预测模型的性能。使用与目标项目具有高相似性的软件缺陷数据可提升跨项目缺陷预测准确性,相反,若源域数据和目标数据存在较大差异,将导致预测模型准确率大幅下降。造成该现象的原因是迁移学习算法无法对存在较大分布差异的数据进行充分修正,使生成的缺陷预测模型性能不升反降[10]。数据选择通过度量源数据和目标数据之间的相似性,可进一步提高跨项目缺陷预测的准确性[11]。然而目前的数据选择方法大多依据欧氏距离、余弦距离等测度方法来度量源数据和目标数据之间的语义相似度,并基于此实现数据选择。

针对以上问题,本文提出了面向代码数据分布相似性测度的数据选择方法,并基于此建立了多源跨项目缺陷预测模型。对比以往方法,本文方法的创新之处在于:改进最大均值差异测度(Maximum Mean Discrepancy,MMD),使其能够从概率分布相似性的角度去度量源数据和目标数据之间的相似性,并基于此建立源项目选择机制,防止出现负迁移。

本文第 2 节回顾了缺陷预测相关工作;第 3 节介绍了本文的方法 MHDP(Multi-source Heterogeneity Defect Prediction);第 4 节描述本文的实验设置和结果;最后总结全文并展望未来。

2　相关工作

2.1　软件缺陷预测

软件缺陷预测的目的是在项目开发的早期阶段,预先识

基金项目:云南省中青年学术和技术带头人后备人选项目(2019HB104)
This work was supported by the Young and Middle-aged Academic and Technical Leader Candidate Project of Yunnan Province(2019HB104).
通信作者:王炜(wangwei@ynu.edu.cn)

别出项目内的潜在缺陷程序模块，并对这类程序模块分配足够的测试资源以确保可以进行充分的代码审查或单元测试，最终达到提高软件产品质量的目的。如图 1 所示，软件缺陷预测流程大致分为：1）通过挖掘与分析软件历史仓库，从中抽取程序模块并进行类型标记；2）随后通过分析软件代码的内在复杂度或者开发过程特征，设计出与软件缺陷存在强相关性的度量元（metrics），并借助这些度量元将已抽取的代码模块抽象为数值表达形式，构建出用于模型训练的缺陷预测数据集；3）基于机器学习方法构建出缺陷预测模型，用于预测模块内是否含缺陷、含有缺陷数或者缺陷密度等问题。

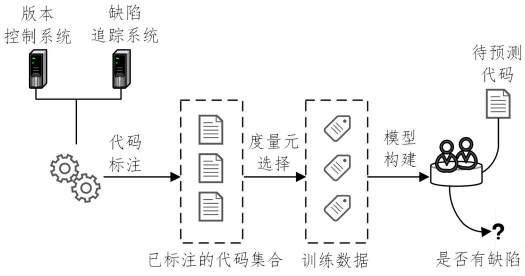


图 1 软件缺陷预测流程

Fig. 1 Software defect prediction process

2.2 跨项目缺陷预测

跨项目缺陷预测（Cross-Project Defect Prediction，CPDP）是使用其他软件项目的缺陷数据构建模型以实现对本项目是否存在缺陷的预测。CPDP 面临的主要挑战是源项目和目标项目之间的异构性，异构性体现在两个方面。1）不同的度量元。CPDP 面临的第一个挑战是怎样对齐源项目和目标项目中包含的不同度量元。现有的 CPDP 方法要求源项目和目标项目中的模块具有相同的度量元。然而，由于度量标准通常是由使用不同提取工具并有不同需求的组织提取的，因此源项目和目标项目可能会有不同的度量元标准。2）不同的数据分布。第二个挑战是源项目和目标项目数据分布的差异。虽然源项目和目标项目具有相同的度量元，但不同项目之间的本质区别也体现在数据分布的多样性上。只有当源项目和目标项目的数据分布尽可能相似时，源项目数据训练的预测模型才能更好地适应目标项目数据。

2.3 数据选择

数据选择通过选择与待预测软件代码具有相似性的训练数据，以确保跨项目缺陷预测的准确性。文献[12]提出 Burak 数据选择方法，对目标项目中的每个软件模块实例使用 KNN 算法从多个源项目数据中去寻找与其最接近的若干模块实例组成的目标项目数据。当目标项目数据远小于源项目数据时，源项目数据可以提供更多的信息，但特征不相似的数据也可能包含对模型训练有用的信息。针对该问题，文献[13]提出 Peters 数据选择方法，采用源项目数据寻找目标项目最近邻的源项目构成模型训练数据。这两种方法依然存在不足，当数据量增大时，算法运行时间呈指数增长，同时以上方法仅从数据个体的角度提出了数据选择，并没有考虑数据的整体特征分布。文献[14]提出在跨项目缺陷预测时，如果可以充分利用源项目和目标项目之间存在的分布特征相关性，从数据中获取先验知识指导源项目数据的选择，可以在很大程度上提高模型性能。文献[11]在实验中从项目数据定义其特征属性，然后通过聚类的方式寻找多个源项目数据作为训练集。但由于从项目数据进行数据选择会引入不相关的软件模块数据，因此导致了模型综合评价指标仍然较低。

He 等[15]使用欧氏距离直接度量源项目和目标项目之间的相似性，并利用 Peters 和 Burak 滤波器从候选项目中选择每个测试实例最接近的训练实例。Li 等[16]提出基于特征模块的数据选择方法，与已有的数据滤波器进行比较，并使用了朴素贝叶斯和支持向量机作为底层分类器。Liu 等[17]提出两阶段迁移学习模型，基于 F 值和成本效益提出源项目估算器 SPE，得到与目标项目分布相似度最高的两个源项目，然后根据 TCA[6]（Transfer Component Analysis）算法构建两个缺陷预测模型。以上实验结果表明，使用与待预测代码具有高相似性的其他项目代码作为源域数据，可以有效提高缺陷预测精度。

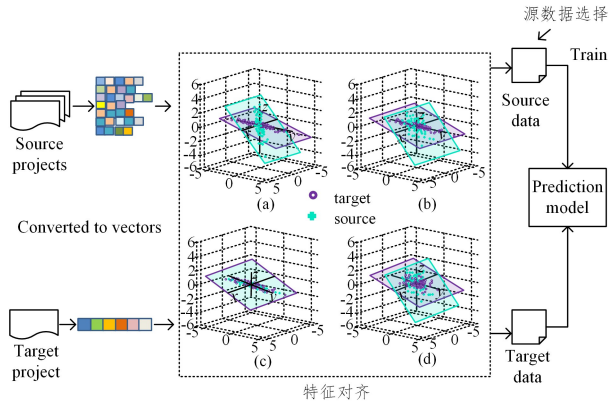


图 2 总体框架图

Fig. 2 Overall framework

然而这些方法大多关注于目标数据的线形相关性和目标数据点和源数据点之间的相似性，从数据集的角度使用数据分布的方法进行相似性测度的工作还比较少。因此，本文提出一种从数据集角度使用数据分布的改进的 MMD 数据选择方法。

3 本文方法

如图 1 所示，本文提出了一种基于数据选择的多源跨项目缺陷预测，该方法包含两个步骤：1）特征对齐，通过将源数据转换到目标数据的特征空间中来解决源数据、目标数据之间的异构性对缺陷预测建模造成的阻碍；2）改进 MMD 实现从概率分布相似性的角度去度量源数据和目标数据之间的相似性，并基于此实现源数据选择。

3.1 特征对齐

本文将特征对齐问题抽象为将源数据投影到目标数据空间。给定源数据 $X_s \in \mathcal{R}^{m\times n}$，其中 $X_{s,i}=(x_{s,i}^1, x_{s,i}^2, \cdots, x_{s,i}^n)$ 表示第 i 个源数据，而 $x_{s,i}^j$ 表示源域的第 i 个数据的第 j 个度量元取值，给定目标域数据 $X_t \in \mathcal{R}^{k\times l}$，其中 $X_{t,j}=(x_{t,j}^1, x_{t,j}^2, \cdots, x_{t,j}^n)$ 表示目标域的第 j 个数据的第 m 个度量元取值。特征对齐可以形式化为：

$$\arg\min_{R} \| RC_s - C_t \|_2 \tag{1}$$

其中，C_s 和 C_t 分别表示源数据坐标系和目标数据坐标系。坐标系的生成可使用奇异值分解（SVD）对源数据、目标数据进行分解获得。即：

$$X_s = C_s \Sigma V^{\mathrm{T}} \tag{2}$$

$$X_t = C_t H P^{\mathrm{T}} \tag{3}$$

其中，Σ 和 H 分别表示源数据 X_s 和目标数据 X_t 的特征值对

角矩阵，C_s 和 C_t 的列组成了关于源数据 X_s 和目标数据 X_t 的基向量，而 V^{T} 和 P^{T} 的列组成了 SVD 输出数据基向量。

如图 3 所示，通过主成分分析和奇异值分解的方法，将异构源数据投影到目标数据空间，矩阵 R 表示投影矩阵。经过 R 变换的坐标系 RC_s 和坐标系 C_t 之间的距离刻画了源数据与目标数据之间异构性的大小。一个有效投影矩阵 R 可以使源数据与目标数据之间异构性最小，即可形式化为：

$$\min_{R} \| RC_s - C_t \|_2 \tag{4}$$

3.2 源数据选择

数据选择的核心是建立源数据、目标数据概率分布相似测度函数。

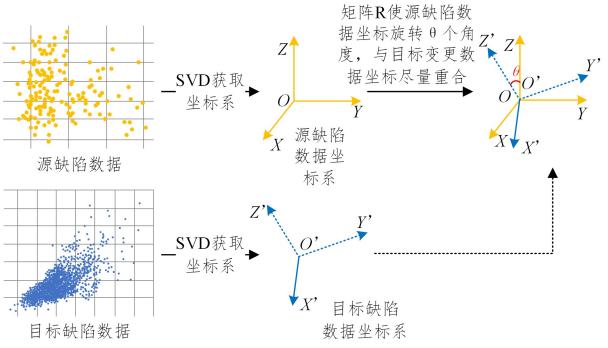


图 3 特征对齐过程

Fig. 3 Feature alignment process

本节通过对最大均值差异（Maximum Mean Discrepancy，MMD）[18-19] 进行修改，使其满足从数据分布概率的角度对源、目标数据相似性进行测度。如式(5)所示，MMD 是一种非参数相似性测度方法，该方法使用核函数 $\phi(\cdot)$ 把线性不可分数据映射到再生核希尔伯特空间（Reproducing Kernel Hilbert Space，RKHS），使用两个数据集均值向量间的欧氏距离刻画分布差异。其中，$x_{s,i}$ 和 $x_{t,i}$ 分别指源缺陷数据集和目标缺陷数据集的第 i 个数据，n_t 和 n_s 分别表示目标数据和源数据的数量。

$$MMD(X_t, X_s) = \left\| \frac{1}{n_t}\sum_{i=1}^{n_t}\phi(x_{t,i}) - \frac{1}{n_s}\sum_{i=1}^{n_s}\phi(x_{s,i}) \right\|_H^2 \tag{5}$$

但作者认为，MMD 仅从均值的角度刻画了两个数据集概率分布的差异，没有考虑数据分布的离散程度。数据分布的离散程度对刻画数据分布相似性具有重要意义，而方差能刻画数据分布离散程度。如图 4 所示，红色数据和蓝色数据的均值相同，但由于两个数据集的方差不同，导致了两个数据集的分布差异很大。

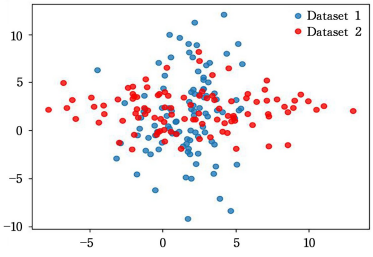


图 4 平均数相同而方差不同的两个数据集的例子

Fig. 4 Examples of two datasets with the same mean but different variances

基于以上认识，本节对 MMD 进行如下修改，修改后的最大均值差异（Maximum Mean Discrepancy based on Variance，MVMD）定义如下：

$$MVMD(X_s, X_t) = \mu Var + (1-\mu) MMD \tag{6}$$

其中，μ 是由用户定义的超参数，它反映了均值和方差两个因素的相对重要性。而 RKHS 中两个数据集之间的方差定义如下。

$$Var = \left\| \frac{1}{n_t}\sum_{i=1}^{n_t}(\phi(x_{t,i}) - \frac{1}{n_t}\sum_{i=1}^{n_t}\phi(x_{t,i}))^2 - \frac{1}{n_s}\sum_{i=1}^{n_s}(\phi(x_{s,i}) - \frac{1}{n_s}\sum_{i=1}^{n_s}\phi(x_{s,i}))^2 \right\|_H^2 \tag{7}$$

4 实验

为了验证所提方法的有效性和先进性，本文通过与基线方法进行对比试图回答如下两个问题：

RQ1 如何体现本文方法的有效性和先进性？

TCA＋[6]、CCA＋（Canonical Correlation Analysis，CCA+）[20] 和多源异构缺陷预测方法（HDPM）[21] 是跨项目缺陷预测的代表性成果，本节将其设为基线方法。通过准确率、召回率和调和平均数 3 个维度进行对比，全面反应本文方法的有效性和先进性。

RQ2 基于分布相似性的数据选择方法对跨项目缺陷预测准确率有怎样的影响？

基于特征相似性的数据选择方法是当前研究的主流方法。本文从数据分布角度提出的数据选择研究成果还较少。通过对比实验试图回答两种数据选择算法对缺陷预测建模结果的影响。

4.1 基准数据集

为了保证研究成果的客观性，本节使用了开源数据集[1)]，其中收集了公开可用的数据集。表 1 列出了本文实验中使用的所有数据集组。每个数据集的简要描述如表 1 所列。

表 1 实验所用数据集的详细信息

Table 1 Details of data set used in experiment

Group	Dataset	of instances: All	of instances: Buggy	of metric
AEEEM	EQ	324	129(39.81%)	61
	JDT	997	206(20.66%)	
	LC	691	64(9.26%)	
	ML	1 862	245(13.16%)	
	PDE	492	209(14.01%)	
Relink	apache	194	98(50.52%)	26
	safe	56	22(39.29%)	
	zxing	399	118(29.57%)	
NASA	cm1	344	42(12.21%)	37
	mw1	264	27(10.23%)	
	PC3	1 125	140(12.44%)	
	PC4	1 399	178(12.72%)	
SOFTLAB	ar1	121	9(7.44%)	29
	ar3	63	8(12.70%)	
	ar4	107	20(18.69%)	
	ar5	36	8(22.22%)	
	ar6	101	15(14.85%)	

如表 1 所列，实验数据来自 AEEEM[22]，Relink[23]，NASA[24] 和 SOFTLAB 这 4 个数据集的 17 个缺陷数据集。上述数据在多篇缺陷预测相关论文中使用过[6,23,25-26]。通过统计可以看出以上数据具有以下特点：

(1)AEEEM 数据集包含 61 个度量元，包括面向对象

1) https://lifove.github.io/hdp/#cm

(OO)度量、先前缺陷度量、更改和代码的熵度量以及源代码流失等等。而 Relink 数据集仅包含 26 个度量元。

(2)具有较强的非平衡性。从表 1 中的 Buggy 指标可以看出缺陷数据和非缺陷数据的比值大多低于 50%,具有非平衡性。非平衡性最强的数据来自 SOFTLAB 数据集中的 ar1 项目,缺陷数据仅占数据总量的 7.44%。

上述属性符合本文研究问题的语境。

4.2 评价指标

本文使用了 2 个广泛使用的度量标准:F 度量和 AUC(ROC 曲线下面积)。F 度量是精度和查全率的谐波平均值。根据如表 2 所列的混淆矩阵,它们定义如下。

表 2 混淆矩阵

Table 2 Confusion matrix

	有缺陷模块	无缺陷模块
有缺陷模块	TP	FN
无缺陷模块	FP	TN

精度(查准率):真实有缺陷部分与预测为有缺陷部分之比。

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

召回率(查全率):预测为有缺陷部分与所有缺陷部分之比。

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

F-measure 值:查全率和查准率的调和平均数。

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

准确率(ACC):正确预测模块与所有模块之比。

$$accuracy=\frac{TP+TN}{TP+FN+FP+TN}$$

由于查全率和查准率之间存在互逆关系,无法全面反映预测模型的性能,因此本文选择查全率和查准率的调和平均值 F-measure 作为度量标准。

4.3 实验设置

本文实验所用平台是 PyCharm2019.1.2 版本,编程语言是 Python。为了全面反映预测模型的性能,本文对实验进行了如下设置。

模型参数信息:学习次数为 200,数据集划分为 5,卷积核数量为 16。

基线方法设置:为了验证所提方法的有效性,本文将 TCA+,CCA+和 HDPM[21]设为基线方法,选取这 3 个方法作为基线有两个原因:1)TCA+和 CCA+方法是比较具有代表性的方法,而 HDPM 是 19 年发表在 MBE 上的方法;2)其实验设置与本文相同,故而能形成有效对比。

TCA+:TCA 是指在保证数据差异的约束条件下获得源项目和目标项目数据特征的线性映射函数,通过该函数将跨项目数据映射到潜在的特征空间以实现模型迁移。在实验中为避免数据属性取值范围不同对模型性能的影响,提出 TCA+自动正规化算法。

CCA+:典型相关分析(Canonical Correlation Analysis,CCA+)方法。CCA+通过为源项目和目标项目寻找一个共有空间,使得投影到该空间的的两个数据集间的相关性最大化。

HDPM:首先,从项目中提取特征,并对来自源项目的数据进行预处理;然后特征对齐;最后,基于投影矩阵获得项目的归一化向量,并训练分类器预测目标项目的缺陷倾向性。

源数据集数量设置:基于 MVMD 分别计算源数据集 $X_s=\{X_{s,1},X_{s,2},\cdots,X_{s,n}\}$ 与目标数据之间的相似性,并根据相似性排序,选取前 K 个源数据集作为预测器的训练数据。根据 He 等[27]提出的三步法,即,从其他项目中测试所有不超过 3 个数据集的组,K 值确定为 3。本文选择与目标域相似性最高的几个数据模块作为异构源项目,按照这个选择标准选择,在表 3-表 6 中分别选择 20 组实验数据。本文实验使用五次交叉验证方法来评估 MHDP 方法,即将 80%的数据用作源项目,而其他 20%用作目标项目来评估性能。

参数设置:

(1)数据分组:数据分为训练集和测试集。将上一节数据选择得到的 K 个源项目作为输入,与目标项目中有标签的部分数据合并起来,构成训练集。将目标项目中设定无标签的部分数据作为测试集。

(2)定义输入:输入层输入的是长度不固定、宽度为 20 的数据,新增一维,使输入数据变为三维数据(K,20,1),其中 K 为不确定的数据样本长度。

(3)特征选择部分实现:用两层一样的卷积神经网络对特征进行选择。

(4)张量平面化操作:第 3 层输出为二维张量,对张量进行“拉平”操作,变成一维向量。

(5)预测部分实现:用 sigmoid 函数作为分类器进行分类,用 adam 优化器定义损失函数,用准确率衡量预测的准确性。

4.4 数据选择结果

为了选择出最佳的多源异构缺陷预测组合,提高多源跨项目缺陷预测的预测性能,本文使用改进后的 MVMD 方法对源数据进行选择,各组数据之间的相似性如表 3 所列。

表 3 AEEEM 与 NASA,Relink,SOFTLAB 间的相似性

Table 3 Similarities between AEEEM and NASA, Relink, SOFTLAB

(单位:%)

	cm1	mw1	PC3	PC4	apache	safe	zxing	ar1	ar3	ar4	ar5	ar6
EQ	23.3	70.2	5.0	18.2	0.3	**74.7**	49.0	49.0	37.1	34.3	51.4	69.3
JDT	23.2	68.2	5.0	18.2	0.4	14.5	6.2	43.5	37.0	33.8	51.0	57.6
LC	23.3	70.4	5.0	18.4	0.2	68.8	37.3	51.5	37.1	34.4	51.4	70.6
ML	23.3	70.3	5.0	18.4	0.2	78.2	49.2	51.5	37.1	34.3	51.4	70.8
PDE	23.3	70.4	5.0	18.4	0.2	**74.7**	63.3	51.9	37.1	34.3	51.4	**71.4**

表 4 NASA 与 AEEEM,Relink,SOFTLAB 间的相似性

Table 4 Similarities between NASA and AEEEM, Relink, SOFTLAB

(单位:%)

	EQ	JDT	LC	ML	PDE	apache	safe	zxing	ar1	ar3	ar4	ar5	ar6
cm1	25.7	25.7	23.4	21.7	22.7	0.3	22.0	21.6	22.4	27.4	22.3	30.7	25.0
mw1	68.7	67.2	70.1	71.3	**70.4**	0.2	**72.0**	68.4	66.9	39.7	44.3	54.0	68.2
PC3	2.4	3.4	3.6	3.0	4.2	0.2	2.3	4.0	4.5	7.6	4.9	8.2	3.4
PC4	19.7	17.9	20.1	20.8	20.1	0.2	22.6	21.8	18.6	27.6	22.8	37.3	22.2

表 5　Relink 与 AEEEM，NASA，SOFTLAB 间的相似性

Table 5　Similarities between Relink and AEEEM，NASA，SOFTLAB

（单位：%）

	EQ	JDT	LC	ML	PDE	cm1	mw1	PC3	PC4	ar1	ar3	ar4	ar5	ar6
apache	0.2	0.2	0.3	0.3	0.3	0.3	0.2	0.3	0.2	0.3	0.2	0.2	0.2	0.2
safe	62.0	7.0	67.7	55.7	**82.7**	23.3	**70.6**	3.7	19.3	53.7	36.8	32.6	53.4	**75.1**
zxing	38.1	4.8	41.2	24.4	67.1	23.3	70.9	3.7	19.4	55.3	36.9	32.6	53.5	**75.0**

表 6　SOFTLAB 与 AEEEM，NASA，Relink 间的相似性

Table 6　Similarities between SOFTLAB and AEEEM，NASA，Relink

（单位：%）

	EQ	JDT	LC	ML	PDE	cm1	mw1	PC3	PC4	apache	safe	zxing
ar1	52.6	45.4	53.2	52.8	52.4	25.8	68.0	2.5	21.0	0.2	51.7	54.2
ar3	36.0	36.0	36.4	39.4	35.9	29.6	41.1	6.7	27.7	0.3	35.0	36.0
ar4	32.8	29.0	31.9	34.0	32.2	25.6	45.8	3.4	24.9	0.2	34.6	34.5
ar5	52.6	52.0	53.9	53.0	54.1	32.7	55.0	6.1	38.4	0.2	50.8	53.2
ar6	68.1	58.0	69.6	69.8	**73.2**	25.9	69.5	2.6	22.6	0.3	**71.9**	73.9

为了使结果看起来更加直观，本文在取结果时选择小数点后 3 位，并且乘一个百分比和四舍五入。从表 3－表 6 中可以看出各个数据集的数据的相似性，从表中的数据的相似性可以看出 ar6 不管是作为源数据还是目标数据，它与每个数据的相似性基本都超过了 0.5，相反，apache 数据与每个数据的相似性都是最差的，甚至未超过 0.01。从整体来看，SOFTLAB 数据集中的数据不管作为源数据或者目标数据，与其他数据集的数据的相似性都是不错的。当 NASA 和 Relink 数据集作为源数据时与其他数据集的数据的相似性很差，这说明这两个数据集不适合作为源项目数据。因此，本文在选择源项目和目标项目数据时只需根据上表按图索骥即可以选择出最优组合，因为当源项目与目标项目的相似性越高时，缺陷预测模型的性能就越好。

4.5　实验结果

为了体现数据选择的有效性，本文对比了 TCA＋和 CCA＋在数据选择前后 *F*-measure 值的变化。表 7 和表 8 中是数据选择前后 TCA＋和 CCA＋的 F-measure 值。最佳结果用黑体突出显示。

表 7　基线方法数据选择前的 *F*-measure 值

Table 7　*F*-measure values before data selection in baseline method

	Without data selcet	
Source Target	TCA＋	CCA＋
cm1→ar4	0.40	0.80
ar4→cm1	0.28	0.78
PC3→ar4	0.37	0.75
ar4→PC3	0.23	0.74
mw1→ar4	0.38	0.70
PC3→ar3	0.46	**0.80**
PC3→ar5	0.51	0.76
cm1→zxing	0.53	0.54
zxing→cm1	0.40	0.39
mw1→apache	0.33	0.69
apache→mw1	0.43	0.27
PC3→safe	0.22	0.73
safe→PC3	0.19	0.56
cm1→apache	**0.62**	0.76
ar3→apache	0.28	0.32
apache→ar3	0.31	0.38
ar4→zxing	0.40	0.47
zxing→ar4	0.22	0.52
ar5→safe	0.31	0.47
safe→ar5	0.32	0.55
Average	0.36	0.60

表 8　基线方法数据选择后的 *F*-measure 值

Table 8　*F*-measure values after data selection in baseline method

	Data selcet	
Source Target	TCA＋	CCA＋
EQ→safe	0.55	0.77
safe→EQ	0.35	0.60
JDT→mw1	0.51	0.76
PDE→mw1	0.42	0.68
mw1→ar4	0.32	0.58
mw1→apache	0.23	0.47
ML→ar6	0.37	0.60
ar6→ML	0.41	0.63
zxing→cm1	0.32	0.57
ML→ar5	0.26	0.50
safe→mw1	0.27	0.56
PDE→safe	0.36	0.60
safe→PDE	**0.66**	**0.81**
LC→ar6	0.35	0.60
zxing→mw1	0.26	0.53
ar6→safe	0.44	0.72
safe→ar6	0.57	0.78
ar6→zxing	0.44	0.70
zxing→ar6	0.44	0.70
PDE→ar6	0.47	0.76
Average	0.40	0.65

表 7 和表 8 列出了数据选择前后 TCA＋和 CCA＋方法的 *F*-measure 值，可以看出经过数据选择后 TCA 和 CCA＋的 *F*-measure 的平均值分别增长了 4%和 5%，因此可以得出结论，实验中选取与目标项目相似性较高的数据作为源数据时，得出的实验结果有所提升，虽然数据选择对实验结果的提升有限，且对于不同的算法的提升也不同，但是这也验证了进行源数据选择的有效性。

对于本文提出的多源域的方法，本文在 SOFTLAB，NASA，AEEEM 和 Relink 这 4 个数据集中选择一个作为目标项目，其它 3 个项目作为异构源项目。在选择源数据和目标数据时，本文根据表 3 中的各组数据的相似性进行选择，选择源数据与目标数据相似性高的数据进行多源跨项目缺陷预测。

将本文方法与没有进行数据选择的基线方法 HDPM 进行对比，表 10 中 *F*-measure 值是 100 次重复计算的平均值，以黑体显示的数字为最佳结果。本文方法的 *F*-measure 平均值为 0.72，而基线方法的 *F*-measure 平均值为 0.71，本文方法的 *F*-measure 的平均值比基线方法的 *F*-measure 的平均值高出 1%，从表 9 中可以看到，本文方法 AUC 的最高值是 0.85，与基线方法相比不管是在部分还是整体上都取得了一定的领先。实验结果说明数据选择对于模型的预测有所提升，这为以后提升软件缺陷预测的预测结果提供了一个可行的办法。

表 9 每个目标的 AUC 比较结果

Table 9 Comparison results of AUC for each target

Target	TCA+	CCA+	MHDP
EQ	0.732	0.678	0.735
JDT	0.735	0.645	0.723
LC	0.633	0.619	0.708
ML	0.659	0.599	0.718
PDE	0.655	0.632	0.669
apache	0.697	0.730	0.608
safe	0.721	0.779	0.659
zxing	0.617	0.608	0.63
cm1	0.642	0.657	0.609
mw1	0.685	0.691	0.709
PC3	0.627	0.651	0.748
PC4	0.687	0.730	0.690
ar1	0.624	0.669	0.688
ar3	0.733	0.768	**0.803**
ar4	0.751	0.762	0.771
ar5	**0.819**	**0.828**	0.782
ar6	0.582	0.633	**0.852**

4.6 预设问题分析

RQ1 如何体现本文方法的有效性和先进性？

对比表 7 和表 8 可以看出，TCA+和 CCA+算法在数据选择后，*F*-measure 在整体上分别提升了 4%和 5%，因此可以得出结论：数据选择对模型的预测性能有所提升，这也说明了本文方法的有效性。从表 10 中可以看出本文提出的方法的 *F*-measure 平均值为 0.72，而 HDPM 方法的 *F*-measure 平均值为 0.71。而本文提出的方法比 HDPM 高出 1%。HDPM 方法是对数据进行预处理而没有进行数据选择，本文方法选择与目标项目相似性高的数据作为源数据，对比实验结果证明了本文方法的先进性。

RQ2 基于分布相似性的数据选择方法对跨项目缺陷预测准确率有怎样的影响？

对比表 3－表 6 和表 8 可以看出，当源数据与目标数据的相似性越高，*F*-measure 值也越高，反之，则越低。例如：safe→PDE 的相似性是 0.82，safe→PDE 的 *F*-measure 值是 0.81，safe→ar6 的相似性是 0.75，而 safe→ar6 的 *F*-measure 值是 0.78，mw1→apache 的相似性是 0.002，而 mw1→apache 的 *F*-measure 也是 20 组数据中最低的 0.47。对比表 8 和表 10 可以看出，综合考虑多源数据之间的相似性与多源数据与目标数据上的相似后，多源数据与目标数据的相似性越高，*F*-measure 值也越高，多源数据之间的相似性越高，*F*-measure 值也越高。例如：{mw1，safe，ar6}→PDE 的 *F*-measure 值是 0.83，而 safe→PDE 的相似性是所有数据里相似性最高的0.83，ar6→PDE 的相似性是 0.73，mw1→PDE 的相似性是 0.70，mw1→safe 的相似性是 0.72，mw1→ar6 的相似性是 0.68，safe→ar6 的相似性是 0.75，safe→mw1 的相似性是 0.71，ar6→mw1 的相似是 0.70，ar6→safe 的相似是 0.72，从表 9 中可以看出本文方法几乎在所有目标域上的 AUC 都比基线方法领先。以上的结果都表明了本文提出的数据选择的有效性，也说明了数据选择对于多源跨项目缺陷预测的准确性是有一定的提升。

表 10 多源域的 MHDP 与 HDPM 的 *F*-measure

Table 10 Multi-source domain MHDP and HDPM *F*-measure

Source→Target	MHDP	Source→Target	HDPM
{EQ,cm1,apache}→ar1	0.62	{CM,Apache,LC}→AR3	**0.81**
{PDE,mw1,safe}→ar6	**0.83**	{MW1,ZXing,EQ}→AR3	0.66
{ML,PC4,apache}→ar6	0.71	{MW1,Apache,PDE}→AR4	0.69
{PDE,ar1,safe}→PC3	0.63	{PC1,ZXing,EQ}→AR4	0.79
{ar4,mw1,zxing}→PDE	0.68	{CM1,ZXing,ML}→AR5	0.57
{JDT,mw1,ar5}→apache	0.57	{PC1,Apache,PDE}→AR5	0.64
{mw1,safe,ar6}→PDE	**0.83**	{AR3,Apache,JDT}→CM1	**0.88**
{PDE,mw1,ar6}→zxing	**0.81**	{AR4,ZXing,PDE}→CM1	0.62
{PDE,PC4,ar6}→safe	0.64	{AR3,Safe,JDT}→MW1	0.69
{ar5,PC3,apache}→ML	0.70	{AR5,ZXing,ML}→MW1	0.65
{ML,PC3,apache}→ar4	0.76	{AR4,Safe,JDT}→PC1	0.80
{PDE,mw1,zxing}→ar6	0.80	{AR5,Apache,JDT}→PC1	0.73
{EQ,ar1,zxing}→ar6	0.78	{CM1,AR3,JDT}→Apache	0.75
{JDT,ar6,apache}→mw1	0.67	{PC1,AR4,PDE}→Apache	0.68
{ML,PC4,apache}→ar1	0.67	{CM1,AR4,ML}→Safe	0.61
{EQ,PDE,ar6}→safe	**0.82**	{MW1,AR3,EQ}→Safe	0.75
{LC,ar6,safe}→cm1	0.62	{MW1,AR4,LC}→ZXing	0.54
{EQ,safe,ar6}→mw1	**0.83**	{PC1,AR3,EQ}→ZXing	0.69
{ML,cm1,safe}→ar5	0.71	{AR3,MW1,ZXing}→EQ	**0.87**
{PC3,safe,ar4}→PDE	0.63	{AR4,CM1,Safe}→EQ	**0.82**
{LC,zxing,ar6}→mw1	**0.83**	{AR4,CM1,ZXing}→JDT	0.64
{mw1,zxing,ar6}→PDE	**0.81**	{AR5,MW1,Apache}→JDT	0.78
{EQ,PC3,ar3}→safe	0.70	{AR3,PC1,Safe}→LC	0.67
{PC3,safe,ar5}→ML	0.65	{AR4,MW1,Apache}→LC	0.70
{PC4,safe,ar1}→PDE	0.78	{AR3,PC1,ZXing}→ML	**0.83**
{JDT,PC4,ar1}→ar1	0.67	{AR4,MW1,Safe}→ML	0.66
{PDE,PC4,zxing}→ar1	0.67	{AR3,MW1,Safe}→PDE	0.62
{EQ,PDE,safe}→ ar6	**0.82**	{AR4,CM1,Apache}→PDE	**0.84**
Average	0.72	Average	0.71

结束语 本文提出使用 MVMD 方法来进行数据选择，选择与目标数据相似性高的数据作为源数据，然后构造投影矩阵进行特征对齐，它可以将源项目迁移到目标项目空间，实现源项目的分布与目标项目的相似分布。实验结果表明，与

最新的 HDPM 方法相比,本文方法不仅能够取得略微的领先,而且还为提升缺陷预测的性能提供了一条有效途径。在未来,将使用更多的数据集评估本文方法,以验证本文方法的泛化,并尝试其他迁移学习方法和分类器,以进一步提高缺陷预测的性能。

虽然实验取得了良好的结果,但是本文的结果可能具有一定的普遍性,由于实验中的数据集的规模仍然很小,因此受到限制。在未来,计划分析更多的缺陷数据来减少这种威胁,特别是来自商业软件项目的数据。本文中使用 F-measure 值和 AUC 来评估本文的方法,这是缺陷预测中常用的评估指标。然而,在缺陷预测中还可以使用其它度量,如预测精度(Acc)以及 G-means 等,这些度量也是综合度量。

参 考 文 献

[1] TIAN J. Software Quality Engineering: Testing, Quality Assurance, and Quantifiable Improvement[M]. Wiley-Interscience, 2005.

[2] CATAL C, DIRI B. Investigating the effect of dataset size, metrics sets, and feature selection techniques on software fault prediction problem[J]. Information Sciences, 2009, 179(8): 1040-1058.

[3] MENZIES T, TURHAN B, BENER A, et al. Implications of ceiling effects in defect predictors[C]// Proceedings of the 4th International Workshop on Predictor Models in Software Engineering. 2008: 47-54.

[4] 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.

[5] MA Y, LUO G, ZENG X, et al. Transfer learning for cross-company software defect prediction[J]. Information and Software Technology, 2012, 54(3): 248-256.

[6] NAM J, PAN S J, KIM S. Transfer defect learning[C]// 2013 35th International Conference on Software Engineering (ICSE). IEEE, 2013: 382-391.

[7] MARTINEZ-FERNANDEZ S, JOVANOVIC P, FRANCH X, et al. Towards automated data integration in software analytics [C]// Proceedings of the International Workshop on Real-Time Business Intelligence and Analytics. 2018: 1-5.

[8] KAMEI Y, FUKUSHIMA T, MCINTOSH S, et al. Studying just-in-time defect prediction using cross-project models[J]. Empirical Software Engineering, 2016, 21(5): 2072-2106.

[9] HALL T, BEECHAM S, BOWES D, et al. A systematic literature review on fault prediction performance in software engineering[J]. IEEE Transactions on Software Engineering, 2011, 38(6): 1276-1304.

[10] LIN D, AN X, ZHANG J. Double-bootstrapping source data selection for instance-based transfer learning[J]. Pattern Recognition Letters, 2013, 34(11): 1279-1285.

[11] HERBOLD S. Training data selection for cross-project defect prediction[C]// Proceedings of the 9th International Conference on Predictive Models in Software Engineering. 2013: 1-10.

[12] 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.

[13] PETERS F, MENZIES T, MARCUS A. Better cross company defect prediction[C]// 2013 10th Working Conference on Mining Software Repositories (MSR). IEEE, 2013: 409-418.

[14] HE Z, SHU F, YANG Y, et al. An investigation on the feasibility of cross-project defect prediction[J]. Automated Software Engineering, 2012, 19(2): 167-199.

[15] HE P, LI B, ZHANG D, et al. Simplification of training data for cross-project defect prediction[J]. arXiv: 1405.0773, 2014.

[16] LI Y, HUANG Z, WANG Y, et al. Evaluating data filter on cross-project defect prediction: Comparison and improvements [J]. IEEE Access, 2017, 5: 25646-25656.

[17] LIU C, YANG D, XIA X, et al. A two-phase transfer learning model for cross-project defect prediction[J]. Information and Software Technology, 2019, 107: 125-136.

[18] GRETTON A, BORGWARDT K M, RASCH M J, et al. A kernel two-sample test[J]. The Journal of Machine Learning Research, 2012, 13(1): 723-773.

[19] SMOLA A, GRETTON A, SONG L, et al. A Hilbert space embedding for distributions[C]// International Conference on Algorithmic Learning Theory. Berlin: Springer, 2007: 13-31.

[20] JING X, WU F, DONG X, et al. Heterogeneous cross-company defect prediction by unified metric representation and CCA-based transfer learning[C]// Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering. 2015: 496-507.

[21] YIN X, LIU L, LIU H, et al. Heterogeneous cross-project defect prediction with multiple source projects based on transfer learning [J]. Mathematical Biosciences and Engineering, 2020, 17(2): 1020-1040.

[22] D'AMBORS M, LANZA M, ROBBES R. An extensive comparison of bug prediction approaches[C]// 2010 7th IEEE Working Conference on Mining Software Repositories (MSR 2010). IEEE, 2010: 31-41.

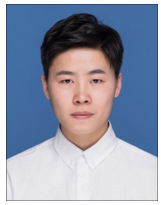
[23] WU R, ZHANG H, KIM S, et al. Relink: recovering links between bugs and changes[C]// Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering. 2011: 15-25.

[24] MENZIES T, GREENWALD J, FRANK A. Data mining static code attributes to learn defect predictors[J]. IEEE Transactions on Software Engineering, 2006, 33(1): 2-13.

[25] D'AMBORS M, LANZA M, ROBBES R. Evaluating defect prediction approaches: a benchmark and an extensive comparison [J]. Empirical Software Engineering, 2012, 17(4): 531-577.

[26] PETERS F, MENZIES T. Privacy and utility for defect prediction: Experiments with morph[C]// 2012 34th International Conference on Software Engineering (ICSE). IEEE, 2012: 189-199.

[27] HE Z, SHU F, YANG Y, et al. An investigation on the feasibility of cross-project defect prediction[J]. Automated Software Engineering, 2012, 19(2): 167-199.



DENG Jian-hua, born in 1997, master candidate, is a member of China Computer Federation. His main research interest is software defect prediction in software engineering.

WANG Wei, born in 1979, Ph.D, associate professor, is a member of China Computer Federation. His main research interests include software engineering, machine learning and formal methods.