基于迁移学习的软件缺陷预测

程 铭[1,2],毋国庆[1,2],袁梦霆[1,2]
(1. 武汉大学计算机学院,湖北武汉 430072;2. 武汉大学软件工程国家重点实验室,湖北武汉 430072)

摘 要: 传统软件缺陷预测方法在解决跨项目缺陷预测过程中适应能力不足,主要是因为源项目和目标项目之间存在不同的特征分布.为了解决这个问题,提出一种新的加权贝叶斯迁移学习算法,算法首先收集训练数据和测试数据的特征信息,然后计算特征差异,将不同项目数据之间差异转化为训练数据权重,最后基于这些权重数据建立预测模型.在 8 个开源项目数据集上进行实验比较,实验结果表明与其他方法相比本文方法显著提高跨项目缺陷预测性能.

关键词: 软件缺陷预测; 迁移学习; 机器学习; 朴素贝叶斯
中图分类号: TP311 **文献标识码:** A **文章编号:** 0372-2112 (2016)01-0115-08
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2016.01.017

Transfer Learning for Software Defect Prediction

CHENG Ming[1,2], WU Guo-qing[1,2], YUAN Meng-ting[1,2]
(1. *School of Computer, Wuhan University, Wuhan, Hubei* 430072, *China*;
2. *State Key Lab of Software Engineering, Wuhan University, Wuhan, Hubei* 430072, *China*)

Abstract: The traditional software defect prediction methods have weak adaptive ability for cross-project defect prediction, largely because of feature distribution differences between the source and target projects. In order to resolve this problem, we propose a novel weighted naive Bayes transfer learning algorithm. Firstly, the feature information of the test data and training data are collected; next, our solution computes feature differences, and transfers cross-project data differences into the weights of the training data; finally, on these weighted data, the defect prediction model is built. Our experiments are conducted on eight open-source projects, and experimental results demonstrate that our method significantly improves cross-project defect prediction performance, compared to other methods.

Key words: software defect prediction; transfer learning; machine learning; naive Bayes

1 引言

随着软件系统规模不断扩大,为了提高软件质量,高效的软件缺陷预测技术越来越受人们的关注[1~4].软件缺陷预测能够在系统开发初期,及时准确地预测软件模块是否包含缺陷,合理分配测试资源,针对性的对缺陷模块进行分析提高产品质量[1].目前,关于软件缺陷预测研究主要集中于两个方面:一是提出新的预测模型;Jing[2] 提出一种基于字典学习的软件缺陷预测方法,能够高效地预测项目内缺陷分布.Wang[3] 结合斯皮尔曼秩相关系数,提出一种基于 C4.5 缺陷预测方法.二是不同方法组合提高预测性能.Jiang[4] 提出将支持向量机(Support Vector Machine,SVM)和蚁群算法相结合,使用蚁群算法优化求解 SVM 参数提高缺陷预测准确性和适用性.

但上述研究均基于项目内缺陷预测,即利用相同项目的历史数据构建预测模型.在实践中,跨项目缺陷预测是必要的.新项目缺少建立预测模型所必须的历史数据,因此使用其他项目数据建立预测模型,预测新项目缺陷分布已经成为发展趋势[5~10].Zimmermann[5] 使用 12 个项目构建了 622 个跨项目组合,评估跨项目缺陷预测模型的性能,发现当前预测模型并不能提供令人满意的预测效果,其主要原因是不同项目间存在不同的数据分布.而大部分机器学习分类器的设计是

收稿日期:2014-06-06;修回日期:2015-05-14;责任编辑:李勇锋
基金项目:国家自然科学基金(No.91118003,No.61003071);深圳战略性新兴产业发展专项资金(No.JCYJ20120616135936123)

假设训练数据和测试数据具有相同特征空间或数据分布[6],并不支持跨项目缺陷预测. 为了解决这个问题,Turhan[7]提出利用 K 近邻过滤器,选择不同项目中相似实例作为训练样本,丢弃差异较大的数据构建预测模型. 在一定程度上提高了预测性能,但是丢弃数据可能包含有用的训练信息. Canfora[8]提出基于遗传算法的多目标逻辑回归预测模型,充分考虑成本效益之间的权衡进行跨项目缺陷预测.

近年来基于迁移学习的跨项目缺陷预测方法被提出,Nam[9]利用迁移成分分析技术挖掘不同项目数据的共有特征空间,迁移有用信息消减数据差异,选择最优规范化策略进行跨项目缺陷预测. Ma[10]提出迁移贝叶斯模型(Transfer Naive Bayes,TNB)使用加权训练数据构造贝叶斯分类器,虽然该方法有效地提高了缺陷预测性能,但 TNB 算法在计算权值时只考虑目标数据每个属性的最大值和最小值,并不能完全反映目标数据集所有特征.

为了进一步提高跨项目缺陷预测性能,本文充分考虑所有训练样本特征信息不丢弃任何样本,提出一种加权贝叶斯迁移学习算法(Weighted Naive Bayes,WNB). 将训练数据集和目标数据集之间的特征差异转换为训练实例权重;在加权训练数据上建立预测模型预测缺陷分布. 实验结果表明,WNB 算法简单适用、鲁棒性强显著提高跨项目缺陷预测性能优于其他比较算法.

2 基于迁移的缺陷预测

在基于迁移的软件缺陷预测中,每个特征为软件模块度量特征,所有度量值向量构成对应软件的特征空间. 假设软件特征空间为 χ,软件模块数据集为 $\boldsymbol{X}=\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n\}\in\chi$,其中 $\boldsymbol{x}_i$ 为相应软件的第 i 个模块的度量值向量. 通常情况下,不同的软件项目数据集,在相同度量特征的前提下,特征空间的特征值范围是不同的(即特征分布不同). 因此,本文提出一种加权贝叶斯迁移学习算法能够最大限度的利用跨项目数据的特征信息,提高预测性能.

2.1 朴素贝叶斯缺陷预测模型

假设训练样本集为 $\boldsymbol{L}=\{(\boldsymbol{x}_1,y_1),\cdots,(\boldsymbol{x}_n,y_n)\}$,其中 $\boldsymbol{x}_i$ 表示第 i 个样本,y_i 为样本 $\boldsymbol{x}_i$ 的类别属性,n 为训练样本个数. 在软件缺陷预测问题中 $y_i\in$('true','false'):缺陷模块被标记为('true'),无缺陷模块标记为('false'). 假定 $\boldsymbol{T}=\{\boldsymbol{t}_1,\cdots,\boldsymbol{t}_m\}$ 为测试数据集,m 为测试数据集样本个数. 朴素贝叶斯分类器可以使用下列公式对测试实例 $\boldsymbol{t}$ 进行分类:

$$y(\boldsymbol{t})=\arg\max_{y\in Y}P(y\mid\boldsymbol{t})=\arg\max_{y\in Y}\frac{P(y)\prod_{j=1}^{k}p(a_j\mid y)}{\sum_{y\in Y}P(y)\prod_{j=1}^{k}p(a_j\mid y)}\tag{1}$$

其中 $\boldsymbol{t}=\{a_1,\cdots,a_j,\cdots a_k\}$,$a_j$ 为测试实例第 j 个属性,k 为属性个数. $P(y)$、$P(a_j|y)$ 和 $P(y|\boldsymbol{t})$ 分别表示先验概率、条件概率和后验概率. 朴素贝叶斯分类器(Naive Bayes,NB)是一个经典的贝叶斯分类算法. 该算法假定属性值对给定类的影响独立于其他属性值. 本文所提 WNB 算法也是基于这个假设. 在软件缺陷预测中,每个软件模块使用软件度量提取特征,度量属性之间相互独立并不存在依赖关系,因此我们认为各个属性同样重要具有相同权重.

2.2 加权贝叶斯模型(Weighted Naive Bayes,WNB)

为了迁移测试数据信息,首先收集测试集数据,将每个训练实例与其进行比较,计算每个训练实例与测试集的相似度;然后基于数据引力方法将相似度转化为该训练实例权重,最后基于加权训练数据建立预测模型.

2.2.1 相似度计算

在 WNB 方法中,每个实例表示为 $\boldsymbol{x}_i=\{a_{i1},a_{i2},\cdots a_{ik}\}$,$a_{ij}$ 为 $\boldsymbol{x}_i$ 的第 j 个属性,k 为属性个数. 为了获取训练实例和测试集之间的相似度,首先计算测试集和训练集在对应属性上的特征差异. 定义属性特征向量(Attribute Characteristic Vector,ACV),其由 5 个元素组成,分别为数据集中该属性的最小值、最大值、平均值、中位值以及标准差,记作:$\mathrm{ACV}_j=\{att\_min_j,att\_max_j,att\_mean_j,att\_median_j,att\_std_j\}$,其中 $j=1,\cdots,k$,$\boldsymbol{ACV}_j$ 为第 j 个属性的属性特征向量. 计算测试集和训练集属性特征向量集合,即:$ACVSet\_Test=\{\boldsymbol{ACV}_1,\cdots,\boldsymbol{ACV}_k\}$,$ACVSet\_Train=\{\boldsymbol{ACV}_1,\cdots,\boldsymbol{ACV}_k\}$. 在本文,属性特征之间的差异,根据它们之间的距离度量,距离越大属性差异越大. 欧式距离是较常用的距离计算形式,由于软件缺陷数据类型为数值型,欧式距离适用于分析数值型数据之间的特征差异. 因此本文利用欧式距离计算测试集与训练集在对应属性上的特征差异,构造属性差异向量 $\boldsymbol{Dif}=\{d_1,\cdots,d_k\}$,其中 d_j 为第 j 个属性的差异.

设训练实例 $\boldsymbol{x}_i=\{a_{i1},\cdots,a_{ik}\}$,计算其每个属性与测试数据对应属性的差异,计算公式如下:

$$difference_{ij}=\sqrt{\frac{(a_{ij}-a'_{1j})^2+\cdots+(a_{ij}-a'_{mj})^2}{n}}\tag{2}$$

其中,$difference_{ij}$ 为训练实例 $\boldsymbol{x}_i$ 第 j 个属性与测试集中对应属性差异,m 为测试样本个数,a'_{mj} 表示第 m 个测试样

本的第 j 个属性. 每个属性在测试样本中贡献相同,通过比较 $difference_{ij}$ 和 ***Dif*** 对应分量 d_j 的值计算训练实例 $\boldsymbol{x}_i$ 与测试集的相似度,计算相似属性个数为:

$$sim_i = \sum_{j=1}^{k} s(a_{ij}) \tag{3}$$

如果 $difference_{ij} \leqslant d_j$,则 $s(a_{ij}) = 1$,否则 $s(a_{ij}) = 0$. 在此给一个例子,设 3 个训练实例:$\boldsymbol{x}_1 = \{2,6,3,'false'\}$,$\boldsymbol{x}_2 = \{7,8,2,'false'\}$,$\boldsymbol{x}_3 = \{3,5,6,'true'\}$. 3 个测试实例:$\boldsymbol{t}_1 = \{5,2,4\}$,$\boldsymbol{t}_2 = \{2,6,3\}$,$\boldsymbol{t}_3 = \{1,3,2\}$. 通过上述公式,计算训练集和测试集属性特征向量集合 $ACVSet\_Train = \{\{2,7,4,3,2.6\},\{5,8,6.3,6,1.5\},\{2,6,3.7,3,2\}\}$、$ACVSet\_Test = \{\{1,5,3,3,2\},\{2,6,3.7,3,2\},\{2,4,3,3,1\}\}$,二者之间的属性差异向量为 ***Dif*** = $\{2.5,5.4,2.4\}$. 然后,根据式(2)、(3)计算每个训练实例的相似度 sim_i. 因为训练样本 $\boldsymbol{x}_1$ 的 3 个属性所对应属性差异 $difference_{1j}$ 分别为 $\{1.9,2.9,0.8\}$,$1.9 < 2.5$,$2.9 < 5.4$,$0.8 < 2.4$,所以 $sim_1 = 3$. 同理 $sim_2 = 2$,$sim_3 = 2$.

2.2.2　训练数据权重

为了迁移测试数据集信息本文引入数据引力概念,数据引力是指在数据分析中模拟万有引力,目前有许多研究将数据引力方法应用到机器学习领域[11,12]. 本文通过模拟引力定律,计算训练数据权重.

万有引力定律:自然界中任何两个物体都是相互吸引的,引力大小与两个物体质量乘积成正比,它们之间距离平方成反比,即 $F = G\frac{m_1 m_2}{r^2}$. G 是引力常数,m_1 和 m_2 为研究对象的质量,r 是两者之间距离.

在训练数据和测试数据上模拟一个"力",假设样本数据中每个属性的质量为 M,那么测试样本质量之和为 mkM,每个训练样本质量为 sim_iM(所有相似属性质量之和). 因此训练实例 $\boldsymbol{x}_i$ 的权重与 $mksim_iM^2$ 成正比,与 $(k - sim_i + 1)^2$ 成反比(分别对应万有引力中两个物体质量的乘积和距离的平方). 因此,训练数据权重公式为:

$$w_i = \frac{mksim_iM^2}{(k - sim_i + 1)^2} \approx \frac{sim_i}{(k - sim_i + 1)^2} \tag{4}$$

根据上述公式,实例 $\boldsymbol{x}_i$ 与测试集相似度越高,赋予权重 w_i 越大. 根据数据加权计算先验概率,先验概率计算公式主要反映测试数据类分布. 如果训练实例与测试数据集相似,则这个训练实例应具有更多权重,那么该训练实例所在类也应具有更多权重,因为这个类可以认为更多存在于测试数据集中. 根据文献[13],先验概率加权计算公式为:

$$P(y) = \frac{1 + \sum_{i=1}^{n} \theta(y_i, y) w_i}{n_y + \sum_{i=1}^{n} w_i} \tag{5}$$

其中,w_i 为训练样本 $\boldsymbol{x}_i$ 权重,y_i 为其所属类属性值,n 为训练实例个数,n_y 为类别个数. $\theta(y_i, y)$ 是一个指示函数,如果 $y_i = y$ 则为 1,否则为 0. 相同类训练数据越多,该类的先验概率越大. 对于测试实例 $\boldsymbol{x}$,第 j 个属性 a_j 的条件概率为:

$$P(a_j \mid y) = \frac{1 + \sum_{i=1}^{n} \theta(a_{ij}, a_j)\theta(y_i, y) w_i}{n_j + \sum_{i=1}^{n} \theta(y_i, y) w_i} \tag{6}$$

其中,a_{ij} 为第 i 个训练实例中第 j 个属性值,n_j 为第 j 个属性不同值数量.

由于软件缺陷数据的属性均为数字型,需要对其进行离散化处理. 结合以上公式,测试数据可以根据预测模型进行分类. 对于上面的例子,假设对测试数据 $\boldsymbol{t}_2 = \{2,6,3\}$ 进行分类,根据式(4),得 $w_1 = 3$,$w_2 = 0.5$,$w_3 = 0.5$.

根据式(5)计算 $P(y)$,其中 $n_y = 2$,$n = 3$,所以:

$P('false') = \frac{1 + w_1 \times 1 + w_2 \times 1}{n_y + w_1 + w_2 + w_3} = 0.75$;同理,$P('true') = 0.25$.

根据式(6)计算 $P(a_j \mid y)$,依据上例有,$n_1 = 3$,$n_2 = 3$,$n_3 = 3$,所以:

$P(a_1 = 2 \mid 'false') = \frac{1 + w_1 \times 1 \times 1}{n_1 + w_1 + w_2} = \frac{8}{13}$,同理

$P(a_2 = 6 \mid 'false') = \frac{3}{13}$,$P(a_3 = 3 \mid 'false') = \frac{3}{13}$,

$P(a_1 = 2 \mid 'true') = \frac{2}{7}$,$P(a_2 = 6 \mid 'true') = \frac{2}{7}$,

$P(a_3 = 3 \mid 'true') = \frac{2}{7}$,因此对于测试数据 $\boldsymbol{t}_2$,根据式(1)得:$P('false' \mid \boldsymbol{t}_2) = 0.969$,$P('true' \mid \boldsymbol{t}_2) = 0.031$.

因为 $0.969 > 0.031$,$\boldsymbol{t}_2$ 被预测为'false'.

2.2.3　WNB 算法分析

算法 1　Weighted Naive Bayes (WNB)

Input: The set of labeled samples $\boldsymbol{L}$;
　　The set of unlabeled samples $\boldsymbol{T}$;
Output: WNB classifier M;
1: Compute $ACVSet\_Train$ of $\boldsymbol{L}$ and $ACVSet\_Test$ of $\boldsymbol{T}$;
2: Compute ***Dif*** between ACVSet_Train and $ACVSet\_Test$;
3: for each instance $\boldsymbol{x}_i \in \boldsymbol{L}$ do
4:　　According to Eqs(2,3), set sim_i to $\boldsymbol{x}_i$;
5: end for
6: for each instance $\boldsymbol{x}_i \in \boldsymbol{L}$ do
7:　　According to Eq(4), set w_i to $\boldsymbol{x}_i$;
8: end for
9: According to Eqs(5,6,1),
　Build Weighted Naive Bayes;
10: for each instance $\boldsymbol{t}_i \in \boldsymbol{T}$ do

11: Use Eq(1) to predict t_i;
12: end for
13: return M

算法 1 给出了 WNB 分类器伪代码. 假设训练数据个数为 n,测试数据个数为 m,k 为属性数量. WNB 分类器主要包括 3 个部分:相似度计算,权值计算以及构造 WNB 分类器. 理论上,计算训练数据和测试数据的属性特征向量需要运行时间分别为 $O(kn)$ 和 $O(km)$,训练实例和测试数据的相似度计算需要运行时间为 $O(kmn)$,权重赋值需要运行时间为 $O(kn)$. 最后 WNB 分类器构建的运行时间为 $O(kn)$. 因此 WNB 分类器理论运行时间为 $O(kmn)$. 它与 NN 过滤器方法的理论运行时间相当,高于 TNB 方法运行时间.

由于实际的训练集与测试集之间存在的分布差异,因此 WNB 分类器具有很大的优势. 文献[7],通过计算它们之间的距离,利用最近邻 k 个训练实例来训练预测模型,距离较大实例被丢弃. 文献[9],挖掘训练集和测试集之间的共有特征空间,将二者映射到该空间消减二者的差异. 但本文认为任何丢弃训练数据以及消减差异都可能包含有用信息. 因此我们为所有训练样本赋予不同权重,在此基础上构建预测模型. 基于这个策略预测模型可以最大限度地利用跨项目数据信息,从而避免损失训练数据的问题. 实验结果表明 WNB 方法性能更优.

3 实验结果与分析

3.1 数据集

本文在两个数据集 AEEEM[14] 和 ReLink[15] 上构造了 26 组跨项目缺陷预测任务. 自动的缺陷信息提取技术会产生标记偏差,因此本文采用手工方式进行缺陷信息提取[16],如表 1 和表 2 所示. 其中 AEEEM 包含 61 个度量属性,ReLink 包含 26 个度量属性,表 3 表 4 分别列出部分度量信息描述.

表 1 AEEEM 数据集

数据集	版本号	时间周期	文件数	缺陷数	属性数
Eclipse JDT Core www. eclipse. org/jdt/core/	3. 4	1. 1. 2005 – 6. 17. 2008	997	206(20. 66%)	61
Eclipse PDE UI www. eclipse. org/pde/pde-ui/	3. 4. 1	1. 1. 2005 -9. 11. 2008	1492	209(14. 01%)	61
Equinox framework www. eclipse. org/equinox/	3. 4	1. 1. 2005 – 6. 25. 2008	325	129(39. 69%)	61
Mylyn www. eclipse. org/mylyn/	3. 1	1. 17. 2005 – 3. 17. 2009	1862	245(13. 16%)	61
Apache Lucene lucene. apache. org	2. 4. 0	1. 1. 2005 – 10. 8. 2008	691	91(13. 16%)	61

表 2 ReLink 数据集

数据集	版本号	时间周期	文件数	缺陷数	属性数
Apache HTTP Server httpd. apache. org/	2. 0	11. 2004 – 4. 2008	194	98(50. 52%)	26
OpenIntents Safe www. openintents. org/	Revision 1088 – 2073	12. 2007 – 12. 2010	56	22(39. 29%)	26
ZXing code. google. com/p/zxing/	1. 6	11. 2007 – 12. 2010	399	118(29. 57%)	26

表 3 AEEEM 数据集部分度量信息描述

度量描述	度量	描述
Previous defects	numberOfBugsFoundUntil	Number of all bugs
Source code	ck _ oo _ wmc	Weighted Method Count
	ck _ oo _ cbo	Coupling Between Objects
Entropy	CvsEntropy	Entropy of code changes
	CvsLogEntropy	Logarithmically decayed CvsEntropy
Entropy of source code	LDHH _ cbo	Linearly decayed entropy of ck _ oo _ cbo
Churn of source code	WCHU _ cbo	Weighted churn of ck _ oo _ cbo

表 4　ReLink 数据集部分度量信息描述

度量描述	描述
AvgCyclomatic	Average cyclomatic complexity for all nested functions of methods.
AvgLine	Average number of lines for all nested functions or methods.
CountLine	Number of all lines for all nested functions or methods.
CountStmt	Number of statements.
MaxCyclomatic	Maximum cyclomatic complexity of all nested functions or methods.
RatioCommentToCode	Ratio of comment lines to code lines.
SumCyclomatic	Sum of cyclomatic complexity of all nested functions or methods.

3.2　实验结果分析

本文采用查全率、查准率和 *F*1 值来评估模型的预测能力. 这些度量基于表 5 所示的混淆矩阵.

表 5　预测结果

	预测为有缺陷	预测为无缺陷
真实有缺陷	*A*	*B*
真实无缺陷	*C*	*D*

查全率(*recall*),正确预测缺陷模块数与真实有缺陷模块数比值,计算公式如下:

$$recall = \frac{A}{A+B}$$

查准率(*precision*),正确预测缺陷模块数与预测缺陷模块数比值,计算公式如下:

$$precision = \frac{A}{A+C}$$

*F*1 为查全率和查准率的调和平均数,值越高性能越好,计算公式如下:

$$F1 = \frac{2 \times recall \times precision}{recall + precision}$$

(1)跨项目缺陷预测实验结果比较分析

构造了 26 组跨项目缺陷预测任务,AEEEM 数据集中包含 20 组跨项目组合:EQ- > PDE、JDT- > EQ、ML- > JDT 等. 数据集 ReLink 中包含 6 组跨项目组合. 由于数据集 AEEEM 和 ReLink 之间存在不同的度量属性,因此不能交叉组合. 将 WNB 和朴素贝叶斯分类器 NB,NN(相似训练数据选择 K = 10)[7]、TCA[9] 以及 TNB[10] 等方法进行比较.

表 6 ~ 7 分别为本文方法与其他方法在 26 组跨项目组合上的预测结果对比. 基于 NB 建立的模型预测结果较差,这是由于该方法并没有考虑不同项目之间数据差异;基于 NN 和 TCA 建立模型的预测性能明显优于基于 NB 方法,但由于它们在构建过程中丢弃了部分差异较大的训练样本而这些训练中可能包含有用的信息,因此这两种方法预测性能并不十分理想;基于 TNB 建立模型预测性能结果较好,但是该模型只考虑了目标样本属性的极大极小值,并不能完全反映目标数据集所有特征,只有在属性特征差异较小时才能取得较好的预测效果;基于 WNB 建立的预测模型不丢弃任何训练样本,并充分考虑目标样本的所有属性值,使得预测性能优于其他比较模型. 在表 6 和表 7 中也展示了项目内部缺陷预测性能(Test- > Test,十折交叉验证). 通常情况下,项目内部缺陷预测的性能优于跨项目缺陷预测. 但是 WNB 的预测结果较接近甚至高于项目内部缺陷预测性能. 此外使用 Wilcoxon 符号秩检测方法对实验结果 *F*1 值进行统计分析,显著性水平设置为 5%,即 *P* 值小于 0.05 时认为两种比较方法的性能差异具有统计学意义,结果表明所有数据集中 *P* 值均小于 0.05,因此 WNB 与其他比较方法在统计学意义上存在显著差异.

表 6　ReLink 数据集跨项目缺陷预测 *F*1 结果对比

Train- > Test	NB	NN-filter	TCA + NB	TNB	WNB	Within(NB) Test- > Test
ZXing- > Safe	0.3128	0.4666	0.4038	0.5263	0.5263	0.5831
Apache- > Safe	0.5272	0.6692	0.4118	0.6923	0.7273	
Safe- > ZXing	0.3019	0.3285	0.2957	0.2972	0.3357	0.4711
Apache- > ZXing	0.3312	0.3601	0.3006	0.4090	0.4090	
ZXing- > Apache	0.4339	0.5111	0.5077	0.5449	0.5549	0.5983
Safe- > Apache	0.4644	0.4771	0.5051	0.5575	0.5797	
Average	0.3952	0.4687	0.4041	0.5045	0.5222	0.5508

表 7 AEEEM 数据集跨项目缺陷预测 *F*1 结果对比

Train- > Test	NB	NN-filter	TCA + NB	TNB	WNB	Within(NB) Test- > Test
JDT- > EQ	0. 6515	0. 6667	0. 4706	0. 6513	0. 6942	0. 6396
LC- > EQ	0. 6171	0. 5732	0. 3981	0. 6727	0. 6727	
ML- > EQ	0. 6031	0. 6001	0. 3976	0. 6391	0. 6585	
PDE- > EQ	0. 6213	0. 5532	0. 4449	0. 6495	0. 6634	
EQ- > JDT	0. 3074	0. 2562	0. 2182	0. 3672	0. 3939	0. 4748
LC- > JDT	0. 2438	0. 3333	0. 2068	0. 3714	0. 3680	
PDE- > JDT	0. 2637	0. 3909	0. 2539	0. 3840	0. 4030	
ML- > JDT	0. 2444	0. 3409	0. 2533	0. 3593	0. 3727	
EQ- > LC	0. 2000	0. 1394	0. 1081	0. 2913	0. 2701	0. 3029
JDT- > LC	0. 2292	0. 1667	0. 1295	0. 3225	0. 3452	
PDE- > LC	0. 1582	0. 2545	0. 1051	0. 2700	0. 2941	
ML- > LC	0. 1262	0. 0645	0. 1179	0. 2349	0. 2517	
EQ- > ML	0. 2070	0. 1788	0. 1256	0. 2770	0. 3017	0. 3709
JDT- > ML	0. 2373	0. 3076	0. 1731	0. 3309	0. 3408	
LC- > ML	0. 1515	0. 1764	0. 1316	0. 2708	0. 2588	
PDE- > ML	0. 1702	0. 2314	0. 1389	0. 2808	0. 2926	
EQ- > PDE	0. 2119	0. 1802	0. 1423	0. 2382	0. 2575	0. 3693
JDT- > PDE	0. 2172	0. 2771	0. 1853	0. 2548	0. 2806	
LC- > PDE	0. 1773	0. 2308	0. 1396	0. 2291	0. 2979	
ML- > PDE	0. 1715	0. 2021	0. 1625	0. 2283	0. 2425	
Average	0. 2905	0. 3063	0. 2151	0. 3661	0. 3830	0. 4315

(2)基于不同训练数据规模实验

探讨训练数据规模对预测模型的影响,在数据集 ReLink 和 AEEEM 中构造 8 组实验数据,选择单个项目作为测试数据合并其他项目数据作为训练数据构造跨项目组合. 数据集 ReLink 可以构造 3 组实验数据对:Apache + ZXing- > Safe、Apache + Safe- > Zxing、Safe + Zxing- > Apache. 同理,AEEEM 数据集可构造 5 组实验数据对. 训练规模依次从 10% 到 100%,观察预测模型的性能.

图 1 ~8 分别展示了各种方法在不同数据规模上的 *F*1 值. 从图中可以看出,相对于其他方法,WNB 始终有较好的 *F*1 性能. 虽然 TNB 的 *F*1 值较接近 WNB 方法,但它并不是一个合理的模型,因为其只考虑属性的最大值和最小值来捕获不同项目数据之间的相似权重,很可能会丢失有用信息. 当训练样本规模增加时,WNB 可以使用更多的数据信息. 因此训练数据量的不同显著影响预测模型的性能,当训练数据和测试数据相似时,随着训练数据量的增加预测模型性能提高;当训练数据和测试数据之间存在较大差异,随着训练样本数量逐渐增大,预测模型性能下降.

4 结论

本文针对跨项目缺陷预测问题提出了一种新的基于加权贝叶斯模型的迁移学习算法 WNB. 该算法能够最大限度的利用跨项目数据信息,避免现有预测模型丢弃大量训练数据的问题. 在 8 个开源项目数据集上进行实验,实验结果表明 WNB 算法显著提高了跨项目缺陷预测性能. 在后续工作中,我们将收集更多的开源项目来验证 WNB 的通用性,同时进一步考虑如何获取目标数据更多的信息提高预测性能.

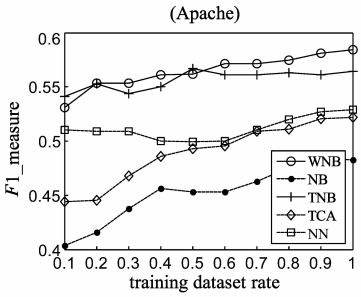

图1　预测Apache项目*F*1性能对比

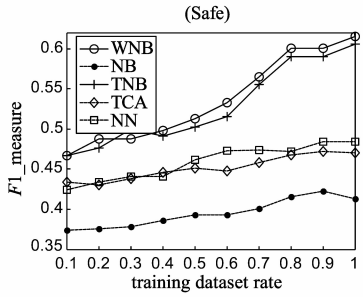

图2　预测Safe项目*F*1性能对比

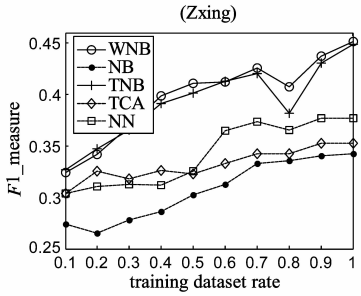

图3　预测Zxing项目*F*1性能对比

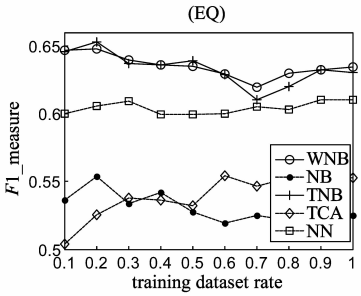

图4　预测EQ项目*F*1性能对比

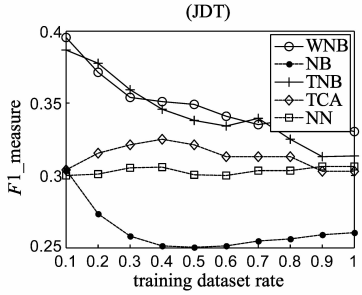

图5　预测JDT项目*F*1性能对比

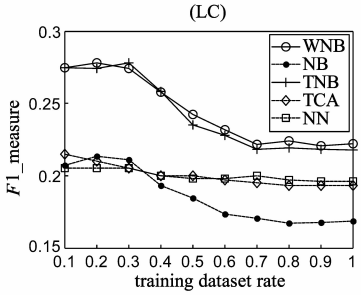

图6　预测LC项目*F*1性能对比

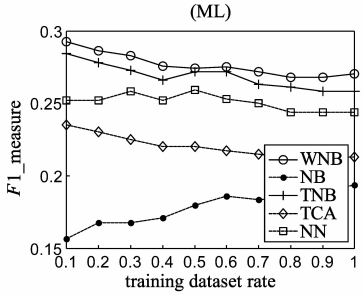

图7　预测ML项目*F*1性能对比

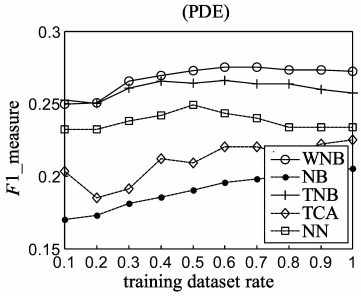

图8　预测PDE项目*F*1性能对比

参考文献

[1] Pizzi N J. A fuzzy classifier approach to estimating software quality[J]. Information Sciences,2013,241:1 – 11.

[2] Jing X Y, Ying S, et al. Dictionary learning based software defect prediction[A]. Proceedings of the 36th International Conference on Software Engineering [C]. Hyderabad: ACM,2014. 414 – 423.

[3] Wang J, Shen B J, Chen Y T. Compressed C4. 5 models for software defect prediction[A]. Proceedings of the 12th International Conference on Quality Software [C]. Xi′an: IEEE,2012. 13 – 16.

[4] 姜慧研,宗茂,刘相莹. 基于 ACO-SVM 的软件缺陷预测模型的研究[J]. 计算机学报,2011,34(6):1148 – 1154.
Jiang Hui-yan, Zong Mao, Liu Xiang-ying. Research of software defect prediction model based on ACO-SVM[J]. Chinese Journal of Computers,2011,34(6):1148 – 1154. (in Chinese)

[5] Zimmermann T, Nagappan N, et al. Cross-project defect prediction[A]. Proceedings of the 7th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on Foundations of Software Engineering[C]. Amsterdam: ACM,2009. 91 – 100.

[6] Pan S J, Yang Q. A survey on transfer learning[J]. IEEE Transactions on Knowledge and Data Engineering,2010,22(10):1345 – 1359.

[7] Turhan B, Menzies T, 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.

[8] Canfora G, Lucia A D, et al. Multi-objective cross-project defect prediction[A]. Proceedings of the 6th IEEE International Conference on Software Testing, Verification and Validation[C]. Luxembourg: IEEE,2013. 252 – 261.

[9] Nam J, Pan S J, et al. Transfer defect learning[A]. Proceedings of the 35th International Conference on Software Engineering[C]. San Francisco: ACM/IEEE, 2013. 382 – 391.

[10] Ma Y, Luo G C, et al. Transfer learning for crosscompany software defect prediction[J]. Information and Software Technology,2012,54(3):248 – 256.

[11] Peng L Z, Yang B, et al. Data gravitation based classification[J]. Information Sciences, 2009, 179(6): 809 – 819.
[12] Wang C, Chen Y Q. Improving nearest neighbor classification with simulated gravitational collapse[A]. Proceedings of the First International Conference on Natural Computation [C]. Changsha: Springer-Verlag, 2005. 845 – 854.
[13] Frank E, Hall M, et al. Locally weighted naive Bayes [A]. Proceedings of the 9th International Conference on Uncertainty in Artificial Intelligence[C]. San Francisco: Morgan Kaufmann, 2003. 249 – 256.
[14] D'Ambros M, Lanza M, et al. An extensive comparison of bug prediction approaches[A]. Proceedings of the 7th IEEE Working Conference on Mining Software Repositories[C]. Cape Town: IEEE, 2010. 31 – 41.
[15] Wu R X, Zhang H Y, et al. Relink: recovering links between bugs and changes[A]. Proceedings of the 19th ACM SIGSOFT Symposium and the Thirteenth European Conference on Foundations of Software Engineering[C]. Szeged: ACM, 2011. 15 – 25.
[16] Bird C, Bachmann A, et al. Fair and balanced?: bias in bug-fix datasets[A]. Proceedings of the 7th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on Foundations of Software Engineering[C]. Amsterdam: ACM, 2009. 121 – 130.

作者简介

程 铭 男,1985 年生于河南郑州,武汉大学计算机学院博士研究生,研究方向:软件工程、缺陷预测、机器学习.
E-mail: chengming@ whu. edu. cn

毋国庆 男,1954 年生,教授,博士生导师,研究方向:软件工程、软件演化.
E-mail: wgq@ whu. edu. cn