软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2023,34(6):2669−2689 [doi: 10.13328/j.cnki.jos.006495]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

一种基于同步语义对齐的异构缺陷预测方法*

李伟漳[1,2], 陈 翔[3], 张恒伟[4], 黄志球[1], 贾修一[4]

[1](南京航空航天大学 计算机科学与技术学院, 江苏 南京 211106)

[2](南京航空航天大学 航天学院, 江苏 南京 211106)

[3](南通大学 信息科学技术学院, 江苏 南通 226019)

[4](南京理工大学 计算机科学与工程学院, 江苏 南京 210094)

通信作者: 贾修一, E-mail: jiaxy@njust.edu.cn

摘 要: 异构缺陷预测 (heterogeneous defect prediction, HDP) 在具有异构特征的项目间进行缺陷预测, 可以有效解决源项目和目标项目使用了不同特征的问题. 当前大多数 HDP 方法都是通过学习域不变特征子空间以减少域之间的差异来解决异构特征问题. 但是, 源域和目标域通常呈现出巨大的异质性, 使得域对齐效果并不好. 究其原因, 这些方法都忽视了分类器对于两个域中的同一类别应产生相似的分类概率分布这一潜在知识, 没有挖掘数据中包含的内在语义信息. 另一方面, 由于在新启动项目或历史遗留项目中搜集训练数据依赖于专家知识, 费时费力且容易出错, 探究了基于目标项目内少数标记模块来进行异构缺陷预测的可能性. 鉴于此, 提出一种基于同步语义对齐的异构缺陷预测方法 (SHSSAN). 一方面, 探索从标记的源项目中学到的隐性知识, 从而在类别之间传递相关性, 达到隐式语义信息迁移. 另一方面, 为了学习未标记目标数据的语义表示, 通过目标伪标签进行质心匹配达到显式语义对齐. 同时, SHSSAN 可以有效解决异构缺陷数据集中常见的类不平衡和数据线性不可分问题, 并充分利用目标项目中的标签信息. 对包含 30 个不同项目的公共异构数据集进行的实验表明, 与目前表现优异的 CTKCCA、CLSUP、MSMDA、KSETE 和 CDAA 方法相比, 在 *F*-measure 和 AUC 上分别提升了 6.96%、19.68%、19.43%、13.55%、9.32% 和 2.02%、3.62%、2.96%、3.48%、2.47%.

关键词: 异构缺陷预测; 语义对齐; 少样本数据; 类不平衡; 线性不可分

中图法分类号: TP311

中文引用格式: 李伟漳, 陈翔, 张恒伟, 黄志球, 贾修一. 一种基于同步语义对齐的异构缺陷预测方法. 软件学报, 2023, 34(6): 2669–2689. http://www.jos.org.cn/1000-9825/6495.htm

英文引用格式: Li WW, Chen X, Zhang HW, Huang ZQ, Jia XY. Heterogeneous Defect Prediction Based on Simultaneous Semantic Alignment. Ruan Jian Xue Bao/Journal of Software, 2023, 34(6): 2669–2689 (in Chinese). http://www.jos.org.cn/1000-9825/6495.htm

Heterogeneous Defect Prediction Based on Simultaneous Semantic Alignment

LI Wei-Wei[1,2], CHEN Xiang[3], ZHANG Heng-Wei[4], HUANG Zhi-Qiu[1], JIA Xiu-Yi[4]

[1](College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China)

[2](College of Astronautics, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China)

[3](School of Information Science and Technology, Nantong University, Nantong 226019, China)

[4](School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China)

Abstract: Heterogeneous defect prediction (HDP) can effectively solve the problem that the source project and the target project use different features. It uses heterogeneous feature data from the source project to predict the defect tendency of the software module in the

* 基金项目: 国家重点研发计划 (2018YFB1003900); 国家自然科学基金 (61906090, 62176123); 中央高校基本科研业务费专项资金 (30920021131)
收稿时间: 2021-04-12; 修改时间: 2021-07-18, 2021-09-14; 采用时间: 2021-09-18; jos 在线出版时间: 2022-10-28
CNKI 网络首发时间: 2022-11-15

target project. At present, HDP has made certain achievements, but its overall performance is not satisfactory. Most previous HDP methods solve this problem by learning domain invariant feature subspace to reduce the difference between domains. However, the source domain and the target domain usually show huge heterogeneity, which makes the domain alignment effect not satisfied. The reason is that these methods ignore the potential knowledge that the classifier should generate similar classification probability distributions for the same category in the two domains, and fail to mine the intrinsic semantic information contained in the data. In addition, because the collection of training data in newly launched projects or historical legacy projects relies on expert knowledge, is time-consuming, laborious, and error-prone, the possibility of heterogeneous defect prediction is explored based on a small number of labeled modules in the target project. Based on these, a heterogeneous defect prediction method is proposed based on simultaneous semantic alignment (SHSSAN). On the one hand, it explores the implicit knowledge learned from the labeled source projects, so as to transfer relevance between categories and achieve implicit semantic information transfer. On the other hand, in order to learn the semantic representation of unlabeled target data, centroid matching is performed through target pseudo-labels to achieve explicit semantic alignment. At the same time, SHSSAN can effectively solve the class imbalance problem and the data linearly inseparable problem, and make full use of the label information in the target project. Experiments on public heterogeneous data sets containing 30 different projects show that compared with the current excellent CTKCCA, CLSUP, MSMDA, KSETE, and CDAA methods, the *F*-measure and AUC are increased by 6.96%, 19.68%, 19.43%, 13.55%, 9.32% and 2.02%, 3.62%, 2.96%, 3.48%, 2.47%, respectively.

Key words: heterogeneous defect prediction (HDP); semantic alignment; few sample data; class imbalance; linearly inseparable

传统的软件缺陷预测 (software defect prediction, SDP) 方法基于项目中足够多的已标记数据来训练一个稳定的分类模型, 然后基于该模型来预测同一项目中新的模块或实例中的缺陷倾向性, 这称为项目内缺陷预测 (within-project defect prediction, WPDP)[1−3]. 通常, 只有当足够多的历史缺陷数据用于训练模型时, WPDP 才能发挥出其高效的模型性能. 然而, 从软件项目中获取足够多的已标记数据并不现实, 特别是对于新启动项目和历史遗留项目, 历史缺陷数据可能非常有限甚至会没有[4−6], 这严重制约了 WPDP 的应用.

为了解决 WPDP 中已标记数据不足的问题, 研究人员提出了跨项目缺陷预测 (CPDP), 即使用其他项目 (称为源项目) 中有足够多历史数据的项目构建预测模型, 并使用该模型对另一个项目 (称为目标项目) 进行预测[7,8]. 软件缺陷预测中常用的 PROMISE 数据集中有许多可用的开源缺陷项目, 可以利用这些公共数据集来预测没有历史数据的项目中的缺陷. 研究发现, 从跨项目数据构建的预测模型与从项目内部数据构建的预测模型一样有效[9,10]. 但是传统的同构 CPDP 方法通常假设源项目和目标项目的数据样本具有完全相同的度量标准集, 即源项目和目标项目的特征要完全相同[2,11]. 然而, 在绝大部分情况下, 源项目和目标项目中只有少量相同的特征指标, 如表 1 所示 (相同特征指的是两个数据集中定义相同的特征). 在这种情况下, 仅使用少量的通用特征指标构建同构 CPDP 预测模型, 因损失了一些特征信息, 会造成模型性能较差, 难以达到预期的结果.

表 1 两个数据集之间具有的相同特征数

数据集A∩数据集B	N∩S	N∩R	N∩A	N∩P	S∩R	S∩A	S∩P	R∩A	R∩P	A∩P
相同特征数量	28	4	1	1	4	1	1	1	1	8

注: N表示NASA, S表示SOFTLAB, R表示ReLink, A 表示AEEEM, P表示PROMISE

近几年, 为了克服源项目和目标项目特征异构的问题, 研究人员提出了异构缺陷预测 (HDP). HDP 不仅不要求源项目和目标项目具有相同的特征, 而且当源项目和目标项目的特征完全不相同时也适用. 根据处理异构特征的方法, 按照已处理特征是否保持其原始含义的标准可将现有的 HDP 方法分为两类: 特征选择和特征转换[7]. 基于特征选择的 HDP 方法从源项目中选择指标, 并将其与目标项目中的指标进行匹配, 以统一指标结构, 同时保留指标的原始含义. 基于特征转换的 HDP 方法为两个项目构建了一个公共度量空间, 或将数据从一个域映射到另一个域, 以消除源项目与目标项目之间的异质性 (即 HDP 中源域项目和目标项目的特征呈现差异性), 如图 1 所示.

随着研究人员考虑问题的深入, 目前大部分 HDP 方法都有着不错的性能表现, 特别是与一些同构 CPDP 方法相比, HDP 已经在大部分评价指标上占据优势. 但与传统的 WPDP 方法相比, 其模型整体的稳定性较差. 究其原因, 主要是目前的 HDP 方法考虑的问题还不够全面, 在进行特征转换或特征选择时, 数据中包含的内在语义信息或没有得到充分的挖掘或直接被丢弃. 另一方面, 由于目标项目中含有标记的数据有限且人工标记数据会有误差

又耗时, 我们考虑只在少量目标项目上进行异构缺陷预测. 鉴于此, 本文提出了一种基于同步语义对齐的异构缺陷预测方法 (SHSSAN). 针对异构缺陷项目中存在的类不平衡问题, 利用基于 GAN 网络的过采样方法可以有效解决. 借鉴迁移学习中的异构域适应 (heterogeneous domain adaptation, HDA) 技术可以消除源项目和目标项目中数据的异构性, 从而使得域之间的差异达到最小化. 在将源项目和目标项目对齐的过程中, 不同类别的实例会混杂在一起, 会无法产生区分性特征. 因此我们考虑利用数据基础的语义属性, 利用语义对齐方式能够更加精准的对齐源域和目标域[12]. 更确切地说, 要处理呈现出异质性的数据, 我们首先使用神经网络 (每个域一个) 构建两个非线性特征编码器, 这样能够生成一个公共特征子空间. 此后, 我们利用所有标记的源数据以及少量已标记的目标数据来训练具有标准监督分类损失的共享分类器. 同时, 在观察到分类器在标记的源数据和目标数据之间共享同一类别的相似预测分布的情况下, 我们提出了一个隐式语义相关性损失, 由此可以实现跨域类别之间语义相关性的对齐. 但是, 当优化上述两个目标以导出两个域中受监管数据的判别结构时, 无法完全挖掘未标记的目标数据. 为此, 我们通过神经网络 (共享分类器) 和几何相似性机制达成共识的整体预测, 为未标记的目标实例分配伪标记. 鉴于此, SHSSAN 将通过使用此伪标签修饰过程来自动增强伪标签的置信度. 同时, 通过利用目标伪标签, 显式对齐每个类别的特征表示.

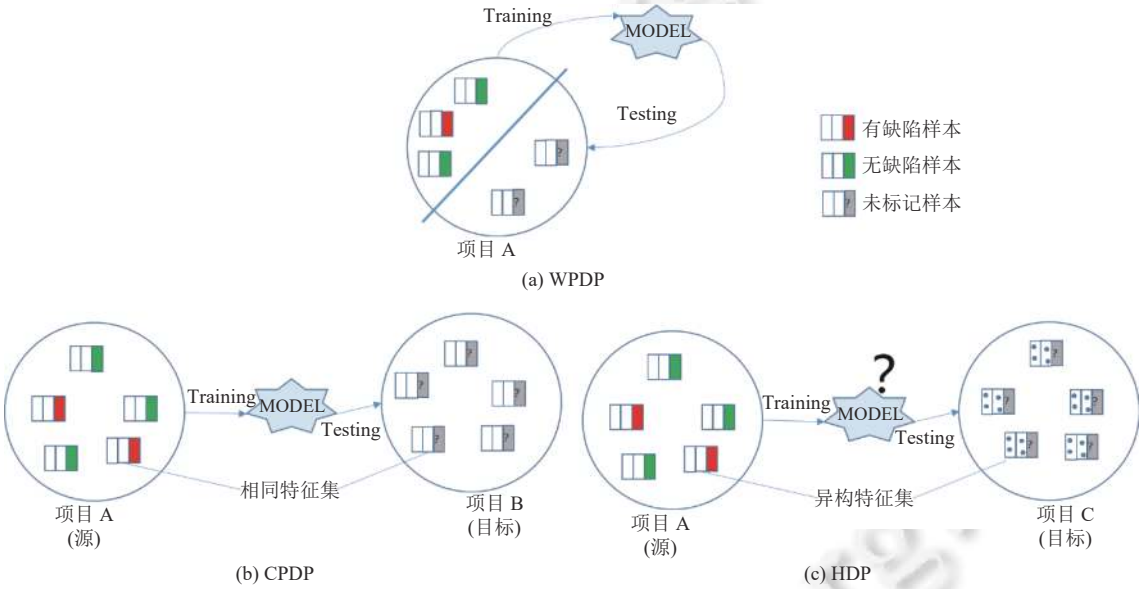

图 1 不同缺陷预测模型

为了说明方法的有效性, 我们进行了大量实验, 与 5 种当前表现最优的 HDP 方法: CTKCCA、CLSUP、MSMDA、KSETE 和 CDAA 进行性能比较, 使用了来自 NASA、AEEEM、ReLink、SOFTLAB 和 PROMISE 这 5 个数据集共 30 个项目进行异构缺陷预测实验. 实验结果表明, 在 *F*-measure 和 AUC 上分别提升了 6.96%、19.68%、19.43%、13.55%、9.32% 和 2.02%、3.62%、2.96%、3.48%、2.47%.

本文第 1 节介绍异构缺陷预测相关方法和研究现状. 第 2 节详细介绍了我们的方法. 第 3 节给出了实验方法. 第 4 节介绍了实验结果并对结果进行了统计性分析. 第 5 节讨论了我们的研究目前存在的不足之处. 最后, 第 6 节是工作总结以及后续潜在研究方向.

1 相关工作

本节简要回顾了异构缺陷预测 (HDP) 和迁移学习中的异构域适应 (HDA) 方法.

1.1 异构缺陷预测

异构缺陷预测是软件缺陷预测领域的研究人员最近几年才提出来的研究方向, 虽然发展时间不长, 但得到的

关注越来越多, 是软件缺陷预测领域未来的重要研究方向. 目前, 关于 HDP 的方法主要分为 3 类: 基于分布的方法、基于特征选择的方法和基于特征转换的方法, 表 2 是近几年部分 HDP 相关方法汇总.

表 2 近几年部分 HDP 方法汇总

方法	方法组成					实验设置
	度量转换	度量选择	度量匹配	处理类不平衡	数据集	评价指标
CPDP-IFS[13]	Yes	—	—	—	11	*F*-measure
FMT[14]	—	Yes	Yes	—	16	AUC
CCA+[15]	Yes	—	—	—	14	*Recall*、Pd、Pf、*F*-measure
HDP-KS[11]	—	Yes	Yes	—	28	AUC
CCT-SVM[16]	Yes	—	—	Yes	14	*F*-measure、AUC
EMKCA[17]	Yes	—	—	Yes	30	AUC
CTKCCA[2]	Yes	—	—	Yes	28	Pd、Pf、*F*-measure、G-mean、AUC
CLSUP[18]	Yes	—	—	Yes	30	G-mean、AUC
HAD[6]	Yes	—	—	—	12	*F*-measure、AUC、Bal
MSMDA[22]	Yes	—	—	—	28	IPR (针对隐私)、G-mean、AUC
TSEL[19]	Yes	—	—	Yes	30	AUC、*Precision*、*Recall*、*F*-measure、Bal
MVSE[20]	Yes	—	—	—	12	PofD20 (effort-aware)
KSETE[21]	Yes	—	—	Yes	3	Pd、Pf、 AUC、G-measure、MCC、IFA、Popt (20%)
CDAA[23]	Yes	—	—	—	28	AUC、MCC、Popt (effort-aware)
FSLBDA[24]	Yes	—	—	Yes	2+15	*F*-measure、AUC、G-mean

(1) 基于分布的方法

2014 年, He 等人[13]首次引出异构缺陷预测思想, 其考虑当跨项目缺陷预测中源项目和目标项目特征不平衡时, 同构 CPDP 是否依然运作良好. 为了解决这个问题, He 等人基于分布特征的实例映射提出了一种简单的方法, 验证了将带有不平衡特征的方法与传统的 CPDP 方法相结合能够在一定程度上提高常规 CPDP 的预测性能.

(2) 基于特征选择的方法

2015 年, Nam 等人[11]首次提出异构缺陷预测概念, 他们在具有异构特征的项目上进行特征选择和特征匹配并建立预测模型. 实证研究表明, 他们的方法约有 68% 的结果优于 WPDP 或与 WPDP 相当.

2017 年, 为了解决 HDP 中的异类特征, Yu 等人[14]提出了一种特征匹配和传递 (FMT) 方法. 首先, 他们对源项目进行特征选择, 并获得选定特征的分布曲线. 同样, 还获得目标项目中所有要素的分布曲线. 其次, 根据不同分布曲线的距离, 设计一种特征匹配算法, 将异构特征转换为匹配特征. 最后, 可以实现从源项目到目标项目的特征转移. 实验结果表明 FMT 方法有效.

(3) 基于特征转换的方法

2015 年, Jing 等人[15]针对异构项目提出统一的指标表示 (UMR). 在统一度量表示的基础上, 首次将一种有效的迁移学习方法典型相关分析 (CCA) 引入 HDP 中, 以使源项目和目标项目的数据分布相似.

2016 年, Cheng 等人[16]在 Jing 等人[15]的基础上首次考虑了异构缺陷预测项目中的类不平衡问题. 其考虑了不同的误分类成本, 使分类趋向于将模块分类为有缺陷的模块, 从而减轻了数据不平衡的影响. 不过该方法只考虑了在 SVM 分类器上的性能表现, 在其他分类器上的表现并不占优.

2017 年, Li 等人[17]重点考虑了异构缺陷预测中的两个问题: 数据线性不可分和数据不平衡问题. 提出了一种基于集成多核相关对齐 (EMKCA) 的新方法. 具体来说, 首先通过多个内核倾斜将源和目标项目数据映射到高维内核空间, 以便可以更好地分离有缺陷和无缺陷的模块. 然后, 设计了一种内核相关性对齐方法, 以使源项目和目标项目的数据分布在内核空间中相似. 最后, 将多个核分类器与集成学习相结合, 以减轻类不平衡问题

的影响, 从而可以提高缺陷预测模型的准确性. 在 30 个公共项目上进行的大量实验表明, EMKCA 优于相关的竞争方法.

2018 年, Li 等人[2]为 HDP 提出了一种新的成本敏感型迁移内核典型相关分析 (CTKCCA) 方法. CTKCCA 不仅可以使非线性特征空间中的源项目和目标项目的数据分布更加相似, 在这种非线性特征空间中学习的特征具有良好的可分离性, 而且可以利用缺陷和无缺陷类别的不同误分类成本来减轻类别不平衡问题. 同年, Li 等人[18]首次考虑了混合项目数据来预测目标项目, 提出了一种新颖的成本敏感标签和结构一致的单边投影 (CLSUP) 方法. CLSUP 不仅可以更好地利用项目内和跨项目的数据, 而且还可以通过为易错样本和不易错样本设置不同的误分类成本来减轻类不平衡问题. Xu 等人[6]将异构域适应 (HDA) 引入异构缺陷预测中, HDA 将跨项目数据视为来自具有异构功能集的两个不同域. 它采用域适应方法将来自两个域的数据嵌入到一个具有较低维的可比较特征空间中, 然后使用字典学习技术来测量两个映射的数据域之间的差异.

2019 年, Li 等人[19]在 EMKCA 的基础上, 提出了一种新颖的用于 HDP 的两阶段集成学习 (TSEL) 方法, 该方法包含两个阶段: 集成多核域自适应 (EMDA) 阶段和集成数据采样 (EDS) 阶段. 在 30 个公共项目上的大量实验表明, 所提出的 TSEL 方法优于一系列竞争方法. Xu 等人[20]首次将 effort-aware 评价指标应用于异构缺陷预测, 考虑了 HDP 场景下的预测成本, 提出了一种称为多视图光谱嵌入 (MVSE) 的新颖方法来解决异构缺陷预测问题. MVSE 将跨项目数据视为两个不同的视图, 并利用频谱嵌入方法将异构特征集映射到两个映射特征集具有最大相似性的一致空间中. Tong 等人[21]提出了一种新的用于 HDP 的内核频谱嵌入传输集成 (KSETE) 方法. KSETE 首先解决源数据的类不平衡问题, 然后尝试通过结合内核频谱嵌入, 迁移学习和集成学习为源和目标数据集找到潜在的公共特征空间. 在 HDP 和 CPDP-CM 场景下, 针对 22 个公共项目进行了实验并得出结论, KSETE 在 HDP 和 CPDP-CM 方案中都是非常有效的. Li 等人[22]首次考虑了 HDP 中的数据隐私问题, 提出了一种基于多源选择的流型判别对齐 (MSMDA) 方法. 为了保护数据所有者的隐私, 设计了一种基于稀疏表示的双重模糊算法并将其应用于 HDP. 通过对 28 个项目的案例研究, 表明 MSMDA 可以实现比一系列基准方法更好的性能.

2020 年, Gong 等人[23]首次考虑数据的标签信息对 HDP 的影响, 提出了一种新的条件域对抗适应 (CDAA) 方法来解决 SDP 中的异构问题, 该方法受生成对抗网络 (GAN) 的驱动. CDAA 的体系结构中有 3 个网络, 包括一个生成器, 一个鉴别器和一个分类器. 生成器学习如何将源样本空间转移到目标样本空间, 鉴别器学习如何识别生成器生成的伪造样本, 分类器学习如何正确分类样本的标签. 实验结果表明, CDAA 方法可以利用标签信息有效地将源项目映射到目标项目并提高预测性能. Wang 等人[24]首次将小样本学习应用到 HDP 中, 提出了一种针对异构缺陷预测的小样本学习平衡分布自适应 (FSLBDA) 方法. 首先使用极端梯度增强功能来删除项目的冗余指标. 然后, 通过平衡分布自适应, 减少源域和目标域之间的数据差异, 考虑了边际分布和条件分布差异的可能性, 并为它们自适应地分配了不同的权重. 最后, 采用自适应提升来减轻训练数据规模较小所带来的影响. 实验结果表明, 与 3 种经典方法相比, FSLBDA 可以有效地提高预测性能.

1.2 异构域适应

域适应是迁移学习的研究内容之一, 其旨在将知识从源域转移到不同但相关的目标领域, 关键是要为这两个领域学习良好的特征表示[6]. 域适应方法已成功应用于不同的研究领域, 例如计算机视觉、图像分类、语音识别、面部识别、机器人建模和推荐系统等.

但是, 传统的域适应方法假定来自两个域的数据由具有相同维的相同特征表示. 因此, 对于源域和目标域数据特征不同的情况并不能直接应用. 这种情况称为异构域适应. 解决此问题的基本思想是使用两个映射的矩阵将数据从两个域转换为一个低维特征空间, 在该空间中可以测量两个映射数据的相似性. 这与解决 HDP 异质性的主要思想相对应. 实际上, 可以将 HDP 视为异构域适应的一种特殊情况. 源项目和目标项目可以被视为 2 个不同的域, 而它们的指标也可以被视为要素[18]. 例如, 来自 NASA 数据集的 CM1 项目和来自 ReLink 数据集的 Apache 项目是 2 个不同的项目, 并且具有不同的指标. 具体来说, CM1 是从用 C 语言编写的航天器项目中提取的, 具有 38 个

度量, 而 Apache 是从以 Java 语言编写的现代操作系统的开源 HTTP 服务器中提取的, 具有 26 个度量. 如果我们使用 CM1 来预测 Apache 或反向预测, 则可以将其视为异构转移学习的一种情况. 异构域适应是一种先进的迁移学习技术, 它可以使用源域中的数据构建分类器, 从而在目标域中表现良好[23].

近年来, 异构域适应 (HDA) 引起了越来越多的关注. 本文提议的 SHSSAN 在两个方面与现有的 HDA 方法有所区别.

(1) 隐式语义相关迁移: 当弥合跨领域的差距时, 现有方法仅利用从特征级别提取的信息. Wang 等人[25]提出了一种流形对齐方法 (DAMA), 以在对齐过程中保留标签信息. Li 等人[26]通过在训练过程中利用未标记的目标数据将异构特征对齐 (HFA) 扩展为半监督版本 (SHFA). 受深度学习最新进展的启发, Chen 等人[27]提出了带有随机修剪的“迁移神经树” (TNT), 以解决特征映射并促进适应性. 但是, 这些方法都没有利用预测中包含的语义相关性来有效地指导对齐过程并促进更好的可迁移性.

(2) 显式语义对齐: 存在几种方法, 它们应用未标记目标实例的伪标记来强制跨域进行语义对齐. 为了学习代表性的跨域界标, 以得出消除域差异的适当特征子空间, Tsai 等人[28]提出了跨域界标选择 (CDLS). 最近, 为了避免传统的硬标签分配所引入的伪标签, Yao 等人[29]提出了一种软传输网络 (STN), 以在类级别对齐过程中采用软标签策略. 相反, 我们引入了几何相似度伪标签细化机制, 该机制可以利用原始数据的几何属性来提高伪标签作为真实标签的准确性.

2 本文方法

针对标记的源域和只有少量标记的目标域, 我们可以将问题转化为具有半监督设置的异构缺陷预测. 具体来说, 让 $D_S=\{X_S,Y_S\}=\{(x_{s_i},y_{s_i})\}_{i=1}^{n_s}$ 表示源域的一组训练样本, $x_{s_i}\in R^{d_s}$ 表示第 i 个具有 d_s 维特征的样本, $y_{s_i}\in\{0,1\}$ 代表对应的样本标签, 其中 0 代表样本无缺陷, 1 代表样本有缺陷. 相似地, 让 $D_L=\{X_L,Y_L\}=\{(x_{l_i},y_{l_i})\}_{i=1}^{n_l}$ 和 $D_U=\{X_U\}=\{(x_{u_i})\}_{i=1}^{n_u}$ 分别表示目标域的标记样本和未标记样本, 其中 $x_{li},x_{u_i}\in R^{d_t}$ 、 $y_{li}\in\{0,1\}$. 注意, $d_s\neq d_t$ 且 $n_l<<n_u$.

本文提出的 SHSSAN 方法包括数据预处理、特征子空间生成、隐式语义相关迁移和显示语义对齐共 4 个部分, 其方法整体架构如图 2 所示, SHSSAN 算法伪代码如算法 1 所示, 下文将详细描述每个过程的细节.

算法 1. SHSSAN 算法伪代码.

输入: 已标记的源项目 D_S, 目标项目 D_T(少量有标签、大量无标签);
输出: D_U 对应的标签 Y.

```
利用过采样方法使得 D_S 中的数据达到类平衡, 记为 D_BS
  For sd in D_BS:
    For td in D_T:
      For rd in Rounds (迭代轮次):
        从 D_T 中获取少量已标记的目标项目 D_L 和大量未标记的项目 D_U
        For it in Iteration (训练次数):
          针对源 (D_BS + D_L) 和目标 (D_U) 项目分别构建特征编码器 E_S 和 E_T, 以产生域不变且可区分特征
          子空间, 得到源监督分类损失 L_s, 建立相关模型 model
          model.train()
          计算隐式语义对齐损失 L_ISC
          计算显式语义对齐损失 L_ESA
        得到目标损失函数: min_{C,E_S,E_T} L_S + αL_ISC + βL_ESA
      得到未标记目标项目 D_U 的标签 Y
```

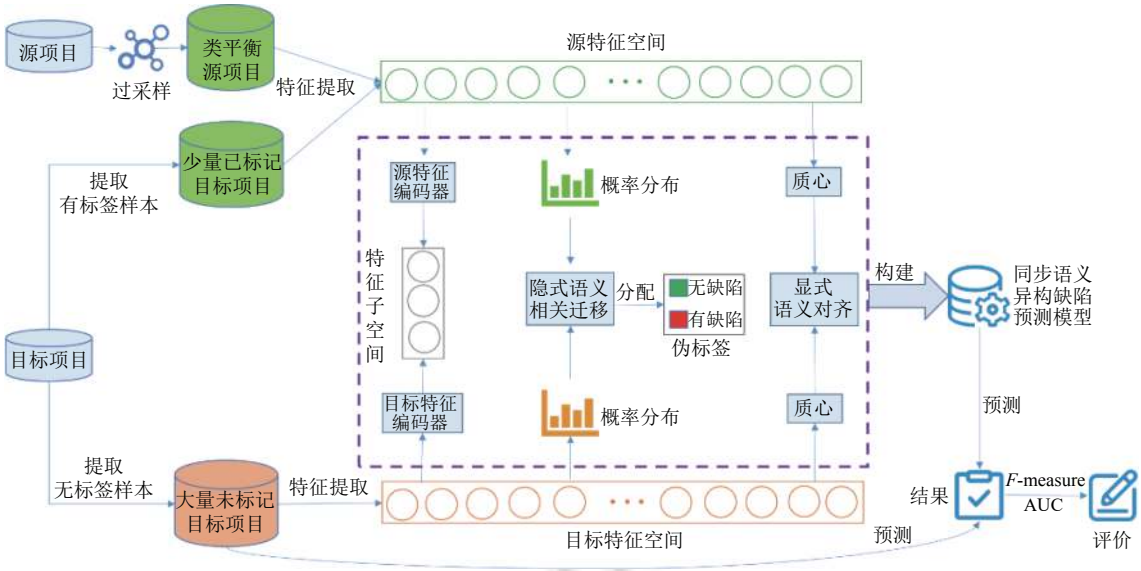

图 2 本文提出的 SHSSAN 方法整体架构

2.1 数据预处理

(1) 处理源项目中数据类不平衡问题

类不平衡问题一直是缺陷预测中需要解决的问题, 在异构缺陷预测中也是如此. 先前的一些 HDP 研究[2,16−19,21,24]也都尝试着解决类不平衡问题, 其大部分使用代价敏感的方法通过给不同类别的样本赋予不同的权重达到缓解类不平衡的目的. 研究表明, 过采样方法是解决类不平衡问题的有效方法, 其效果要比代价敏感的方法以及欠采样的方法好得多[30]. 因此本文使用基于 GAN 网络的过采样方法来处理类不平衡问题. 注意, 实验中使用的源项目是过采样方法处理后的类平衡数据, 而目标项目仍然是原始的类不平衡数据.

(2) 目标项目样本划分

本方法是面向少量已标记的目标项目, 故对目标项目提取少量已标记的样本参与到源项目模型构建中. 实验中将 20% 的目标项目作为已标记样本, 剩下的 80% 作为未标记样本.

(3) 特征提取

对处理好的源项目和目标项目需要提取出与软件缺陷存在强相关性的特征, 然后在提取后的特征上建立相关模型并对目标项目进行预测. 首先针对已标记的源项目和少量已标记的目标项目进行特征组合, 对于源项目中存在而目标项目中没有的特征, 通过补零完成, 同样地, 目标项目中存在而源项目不存在的特征也是进行补零完成. 对于大量未标记的目标项目, 保留其原始的特征信息即可.

2.2 特征子空间生成

由于源项目和目标项目特征差异较大, 需要将源项目和目标项目的特征映射到一个公共子空间中. 为了缓解之前的 HDP 方法以蛮力方式强制对齐域而扭曲具有相同类标签的样本, 我们针对源和目标项目分别构建特征编码器 E_S 和 E_T, 以产生域不变且可区分的特征子空间. 即定义 $f_S(x_i)=E_S(x_i)(x_i\in X_S)$, $f_T(x_i)=E_T(x_i)(x_i\in X_L\cup X_U)$, 其中 E_S 和 E_T 分别是源和目标项目的特征编码器.

本方法最终目标是产生一个共享分类器 C, 该分类器 C 可应用于源数据和目标数据的编码特征 $f(x_i)$, 并正确预测目标项目中未标记样本的标签. 一般, 通过最小化源数据上的经验误差来训练判别式分类器. 我们将源监督分类损失表示为:

$$L_S=\Gamma_{\text{sup}}(X_S,Y_S)=\frac{1}{n_s}\sum_{x_i\in X_S,y_i\in Y_S}\Gamma_{ce}(C(f_S(x_i)),y_i) \tag{1}$$

其中, Γ_{ce} 是交叉熵损失. 可以根据公式 (1) 利用有限的目标标记数据来学习目标网络的参数.

2.3 隐式语义相关迁移

利用公式 (1) 训练分类器会导致过拟合问题, 从而导致模型预测性能下降. 尽管源域项目和目标域项目呈现出异质性, 但对于同一缺陷预测问题, 同一类别的样本往往具有相似的结构特征, 因此会产生相似的类别概率分布, 故我们假设分类器对于两个域中的同一类别应产生相似的分类概率分布[11,15,31], 也即是一种隐性知识. 由此源类别的隐性知识可以促进分类器在目标域上的泛化性能. 因此, 我们将源项目中类 k 的概率输出的平均值视为第 k 个标准值并定义为 $q^{(k)} \in R^K$, 作为第 k 类的“软标签 (soft label)” (可简单理解成样本为有缺陷的概率). 由于有大量可用的带标签的源数据, “软标签”包含许多有价值的类别相关性. 为了充分利用这些相关性, 采用 *Softmax* 来平滑分类器激活, 这可能会在各个类上产生较软的概率分布. 因此, 定义 $q^{(k)}$ 如下:

$$q^{(k)} = \frac{1}{\left|X_S{}^{(k)}\right|} \sum\nolimits_{x_i \in X_S{}^{(k)}} Softmax\left(\frac{C\left(f_S\left(x_i\right)\right)}{T}\right) \tag{2}$$

其中, $X_S{}^{(k)}$ 表示源项目中类 k 的集合, $|\cdot|$代表集合中的样本数. 给定一个带有标签的目标实例, 我们可以使用“软标签”对目标网络进行微调, 以学习语义相关性并将其从源域传输到目标域. 因此, 在学习到的“软标签”的监督下, 相应的损失可以定义如下:

$$\Gamma_{\text{soft}}\left(X_L, Y_L\right) = -\frac{1}{n_l} \sum\nolimits_{x_i \in X_L, y_i \in Y_L} q^{(y_i)^T} \log p_i \tag{3}$$

其中, $p_i = Softmax\left(C\left(f_T\left(x_i\right)\right)\right)$ 是标记的目标样本 x_i 的概率输出. 上述损失可以将语义相关性从源网络迁移到目标网络. 总而言之, 我们进一步考虑了目标数据中已标记样本的监督损失, 并将隐式语义相关损失定义为:

$$L_{ISC} = \Gamma_{\text{soft}}\left(X_L, Y_L\right) \tag{4}$$

这样, 目标网络可以对类边界周围的这些样本执行更好的概括, 并捕获受监督数据中类别之间的语义相关性, 这将实现显著的性能提升.

2.4 显式语义对齐

异构缺陷预测通过对齐源域和目标域的条件分布来学习未标记目标样本的区分表示. 但是, 我们没有未标记目标样本的标签信息. 一种可行的方法是直接利用共享分类器预测的伪标签[32], 这里的伪标签是指分类器预测的结果, 可能有对有错. 预测错误的伪标签会导致对齐过程中的偏差越来越大. 为了避免伪标签的不确定性, 我们考虑揭示基础数据的内在几何知识. 因此, 设计了一种几何伪标签细化机制, 以协助为那些与特征空间中受监督数据的类别质心呈现几何相似性的实例分配伪标签. 首先计算源域和已标记的目标域中类别 k 的质心: $u^{(k)} \in R^{d_{\text{common}}}$, 它是每个类别中两个编码特征的均值向量. 当获得了一组质心 $\left\{u^{(k)}\right\}_{k=1}^{K}$ 后, 使用几何相似度度量第 i 个未标记目标样本分配标签:

$$y_{u_i}^{\langle GS\rangle} = \arg\max_k GS\left(f_T\left(x_{u_i}\right), u^{(k)}\right) \tag{5}$$

其中, $GS(\cdot,\cdot)$ 是潜在特征空间中两个数据点之间的几何关系. 而且, 很容易从共享分类器 C 中获得伪 (预测) 标签, 我们将其定义为 $y_{u_i}^{\langle NN\rangle}$. 使用几何相似性标签和神经网络标签, 只有当 $y_{u_i}^{\langle GS\rangle} = y_{u_i}^{\langle NN\rangle}$ 时, 才会选择未标记的目标样本并为其分配伪标签, 这可以提高伪标签分配的准确性. 这样, 我们有 $X_T = X_L \cup \hat{X}_U$ 、 $Y_T = Y_L \cup \hat{Y}_U$, 其中 $\hat{X}_U, \hat{Y}_U$ 是选定的未标记目标样本及其对应的伪标记的集合.

凭直觉, 很少有未标记的目标样本在早期训练阶段就被正确标注. 随着训练的发展, 越来越多的样本可能会被分配一个经过认可的伪标签. 最终, 大多数样本将被赋予可信的伪标签. 随着训练过程的深入将会使那些具有一致预测标签的样本被接受以参与显式语义对齐, 同时过滤掉不一致的样本. 因此, 我们能够使用设计的显式语义对齐损失来充分学习未标记目标样本的语义表示. 形式上, 每个类在公共要素空间中有 3 个质心, 分别是: 源项目, 目标项目以及源和目标项目组合, 构成三重态质心:

$$u_S{}^{(k)} = \frac{1}{\left|X_S^{(k)}\right|} \sum\nolimits_{x_i \in X_S^{(k)}} f_S\left(x_i\right) \tag{6}$$

$$u_T{}^{(k)} = \frac{1}{\left|X_T^{(k)}\right|}\sum_{x_j \in X_T^{(k)}} f_T\left(x_j\right) \tag{7}$$

$$u_{ST}{}^{(k)} = \frac{1}{\left|X_S^{(k)} \cup X_T^{(k)}\right|}\left(\sum_{x_i \in X_S^{(k)}} f_S\left(x_i\right) + \sum_{x_j \in X_T^{(k)}} f_T\left(x_j\right)\right) \tag{8}$$

其中, $X_T^{(k)}$ 表示 X_T 的子集, 包括类 k 中带有真实标签的已标记的目标样本和经许可分配了伪标签的未标记的目标样本.

三重态质心可以促进语义对齐并增强域之间更好的语义一致性. 最大均值差异 (MMD) 已被证明是有效测量两个域之间差异的方法[33,34]. 因此, 我们采用以下显式的语义对齐方式损失来学习域之间更健壮和更具区别性的表示形式:

$$L_{ESA} = \sum_{k=1}^{K}\left(u_S^{(k)} - u_{T2}^{(k)2} + u_S^{(k)} - u_{ST2}^{(k)2} + u_T^{(k)} - u_{ST22}^{(k)22}\right) \tag{9}$$

通过最小化此目标, 将使每个类别的质心在已编码特征子空间中非常接近, 从而导致跨域的语义上一致的表示形式.

最后, 根据隐式语义相关对齐和显示语义对齐中提到的各种损失, 得到总体的目标函数, 如下所示 (其中超参数 α和β 分别平衡了 L_{ISC} 和 L_{ESA} 对优化过程的影响):

$$\min_{C,E_S,E_T} L_S + \alpha L_{ISC} + \beta L_{ESA} \tag{10}$$

3 实验方法

为了验证本方法在异构缺陷数据集上的效果, 我们对提出的 SHSSAN 方法进行了全面评估并与当前的代表性 HDP 方法进行比较. 下面依次描述了用于研究的数据集, 基准对比方法, 一些用于评估模型性能的指标以及实验的相关设置等.

3.1 数据集

我们使用了来自 NASA、AEEEM、SOFTLAB、ReLink 和 PROMISE 的 5 个不同数据集的 30 个项目来进行实验. 表 3 详细列出了实验中使用的项目的相关信息. 其中缺陷率=缺陷样本个数/样本总数×100%, 不平衡率=(样本总数–缺陷样本个数)/缺陷样本个数. 在这里样本是基于文件级的代码.

表 3 从 5 个数据集中提取的 30 个项目的简单统计

数据集	项目	特征个数	样本总数	缺陷样本个数	缺陷率 (%)	不平衡率 (%)
PROMISE	ant	20	745	166	22.28	3.49
	camel	20	965	188	19.48	4.13
	ivy	20	352	40	11.36	7.80
	jedit	20	306	75	24.51	3.08
	log4j	20	135	34	25.19	2.97
	lucene	20	340	203	59.71	0.67
	poi	20	442	281	63.57	0.57
	synapse	20	256	86	33.60	1.98
	tomcat	20	858	77	8.97	10.14
	velocity	20	229	78	34.06	1.94
	xalan	20	723	110	15.21	5.57
	xerces	20	453	69	15.23	5.57
SOFTLAB	AR1	29	121	9	7.44	12.44
	AR3	29	63	8	12.70	6.88
	AR4	29	107	20	18.69	4.35
	AR5	29	36	8	22.22	3.50
	AR6	29	101	15	14.85	5.73

表 3 从 5 个数据集中提取的 30 个项目的简单统计 (续)

数据集	项目	特征个数	样本总数	缺陷样本个数	缺陷率 (%)	不平衡率 (%)
NASA	CM1	37	327	42	12.84	6.79
	MW1	37	253	27	10.67	8.37
	PC1	37	705	61	8.65	10.56
	PC3	37	1 077	134	12.44	7.04
	PC4	37	1 287	177	13.75	6.27
AEEEM	EQ	61	324	129	39.81	1.51
	JDT	61	997	206	20.66	3.84
	LC	61	691	64	9.26	9.80
	ML	61	1 862	245	13.16	6.60
	PDE	61	1 497	209	13.96	6.16
ReLink	Apache	26	194	98	50.52	0.98
	Safe	26	56	22	39.29	1.55
	Zxing	26	399	118	29.57	2.38

NASA 基准数据集由 13 个软件项目组成, 样本数量从 127 到 17 001, 而特征属性的数量是从 20 到 40. NASA 中的每个项目代表一个 NASA 软件系统或子系统, 其中包含相应的缺陷标记数据和各种静态代码度量. 该数据仓库通过使用错误跟踪系统记录每个样本的缺陷数量. 土耳其软件数据集 SOFTLAB 包含 5 个项目, 其样本数范围是 36 到 121. SOFTLAB 的项目是从 PROMISE 数据集中获得的, 具有 29 个度量标准. PROMISE 收集的数据至今已包含 38 个不同软件开发项目的 92 个版本, 每个项目总共有 20 个度量标准. ReLink 数据集是通过提高缺陷数据的质量来提高缺陷预测性能, 其缺陷信息已经过手动验证和纠正, 包含 3 个项目, 每个项目都有 26 个复杂性指标. AEEEM 用来作为不同缺陷预测模型的基准数据集, 每个 AEEEM 数据集都包含 61 个度量标准.

从表 3 可知: 1) 实验所用的软件项目分布广泛, 来自于 5 个不同的数据集, 不同项目的属性个数都不同, 这正好满足异构缺陷预测的需求, 使得实验更具代表性. 2) 每个项目中包含的样本个数都不相同, 且数量差异较大. 最少的只有 36 个样本, 而最多的达到 1 862 个, 这也间接说明实验所使用的项目的多样性. 3) 各项目缺陷率不同, 其中缺陷率最低的是 SOFTLAB 中的 AR1 项目, 只有 7.44%, 而缺陷最多的项目有 63.57% 的缺陷率. 绝大部分数据集中的缺陷率都低于 40%, 这表明实验所使用的数据集能够更好地反映异构缺陷预测中的类不平衡问题.

3.2 对比方法

目前已有的大部分 HDP 方法都选择了与 WPDP 或同构 CPDP 方法进行比较[2,6,11,14,17~22], 主要原因是这些 HDP 方法在提出的时候, 关于异构缺陷预测的研究并不多, 也不了解其性能表现. 现如今已经有大量关于 HDP 的研究, 因此本文在选择对比方法的时候, 不再与 WPDP 和同构 CPDP 方法进行比较, 我们将对比的重点放在已有的 HDP 方法上. 从 Chen 等人[7]的一篇关于 HDP 方法的综述中可知, 在作者比较的 9 种 HDP 方法中, CTKCCA[2]、CLSUP[18]和 MSMDA[22]性能表现最优, 因此我们挑选这 3 个方法作为对比对象. 此外, 在文献 [7] 之后, 也涌现了不少优秀的 HDP 工作, 如 KSETE[21]、CDAA[23]和 FSLBDA[24]等. 由于 FSLBDA 这篇工作没有提供算法步骤, 也没有提供原始的实验代码, 我们无法复现其工作, 因此很遗憾的没有与其进行对比. 因此, 我们最终挑选了 5 种 HDP 方法进行对比, 分别是: CTKCCA、CLSUP、MSMDA、KSETE 和 CDAA. 下面详细介绍这 5 种方法.

(1) CTKCCA

针对 HDP 中的类不平衡和数据线性不可分问题, 作者提出一种新的成本敏感型迁移内核典型相关分析方法. 首先, 在非线性特征空间中学习特征, 使源项目和目标项目数据分布更加相似; 其次采用不同的误分类成本来降低类不平衡的影响.

(2) CLSUP

CLSUP 是一种新颖的成本敏感标签和结构一致的单边投影方法, 首次考虑了使用混合项目数据来进行预测, 同时也考虑了类不平衡问题.

(3) MSMDA

针对数据多源和数据隐私问题, 作者提出了一种基于多源选择的流型判别对齐方法. 为了保护数据隐私, 设计了一种基于稀疏表示的双重模糊算法.

(4) KSETE

针对 HDP 中普遍存在的一些问题, 作者提出了一种新的用于 HDP 的内核频谱嵌入迁移集成方法.KSETE 首先解决源数据的类不平衡问题, 然后尝试通过结合内核频谱嵌入, 利用迁移学习和集成学习为源和目标数据集找到潜在的公共特征空间.

(5) CDAA

该方法首次考虑了目标项目中的标签信息对 HDP 的影响, 并采用了 effort-aware 中的 Popt 作为评价指标, 提出了一种新的条件域对抗适应方法来解决 SDP 中的异构问题. CDAA 包括生成器、鉴别器和分类器. 生成器学习如何将源实例空间转移到目标实例空间, 鉴别器学习如何识别生成器生成的伪造实例, 分类器学习如何正确分类实例的标签.

3.3 评估指标

如表 2 所示, 目前 HDP 的评价指标主要有: Accuracy、*Precision*、Pd、Pf、*Recall*、G-mean、Bal、*F*-measure、AUC、PofD20 (effort-aware)、IPR (针对隐私)、MCC 和 IFA 等. 其中 PofD20 (effort-aware)、IPR (针对隐私)、MCC 和 IFA 只用于特定的研究场景. Accuracy、*Precision*、Pd、Pf 和 *Recall* 这 5 个评价指标只考虑了数据的部分特征. G-mean、Bal、*F*-measure 和 AUC 是 4 个综合性的评价指标, 其中 *F*-measure 和 AUC 最具代表性且并被广泛使用, 因此我们选择 *F*-measure 和 AUC 作为本文的评价指标. 对于 G-mean 和 Bal, 首先考虑到其重要性不如 *F*-measure 和 AUC, 其次也考虑到实验的工作量, 多一个评价指标, 会增加大量的实验工作 (每种评价指标共有 672 种预测组合), 因此我们的研究只采用了 *F*-measure 和 AUC.

F-measure 是 *Precision* 和 *Recall* 的谐波平均值的综合度量 (*Precision* 即精确率或查准率, 它表示正确分类为缺陷的缺陷样本数与分类为缺陷的样本数之比; *Recall* 即召回率或真实阳性率, 也叫 Pd, 它表示正确分类为缺陷的缺陷样本数与缺陷样本总数之比).

AUC 是 ROC 曲线下的面积, 该曲线在二维空间中绘制, 其中 Pf (即错误阳性率的可能性, 它表示被错误分类为缺陷的非缺陷实例数与非缺陷实例总数的比率) 为 *x* 坐标, Pd 为 *y* 坐标. 由于 AUC 不受类不平衡的影响并且独立于预测阈值, 因此 AUC 被认为是用于比较不同模型的有用度量, 并被广泛使用[35]. 较高的 AUC 表示较好的预测性能, 而 AUC 为 0.5 时表示随机预测的性能. 具体公式如表 4 所示. 其中真阳性 (*TP*)、假阴性 (*FN*)、假阳性 (*FP*) 和真阴性 (*TN*) 分别是被预测为有缺陷的缺陷样本的数量, 被预测为无缺陷的缺陷样本的数量, 被预测为有缺陷的无缺陷样本数和被预测为无缺陷的无缺陷样本数.

表 4 实验所用评价指标

评估指标	公式
Precision	$Precision=\frac{TP}{TP+FP}$
Recall	$Recall=\frac{TP}{TP+FN}$
F-measure	$F\text{-measure}=\frac{2\times Recall\times Precision}{Recall+Precision}=\frac{2\times TP}{2\times TP+FP+FN}$

3.4 实验设置

异构缺陷预测的思路是分别从不同的数据集中挑选一个项目作为源项目和目标项目. 具体而言, 从 30 个项目中选择一个项目作为目标, 然后依次使用其他不属于目标项目的数据集中的每个项目作为源. 例如, 如果目标项目来自 NASA, 则源项目将选自 SOFTLAB, ReLink, AEEEM 或 PROMISE. 通过这种方式, 我们从 5 个数据集的这

30 个项目中构造了 672 种可能的预测组合 (即源⇒目标).

实验中用到的各项参数设置如下: 使用 3 个原型用于域通用对齐损失, 原型预测的方式是 Cosine, 使用双层的投影网络结构, 8 个通用的公共子空间特征个数, 使用的优化器是 mSGD, 学习率为 0.1, 源项目 *Softmax* 的 T 值为 5, 用于平衡隐式语义相关迁移损失的超参数 α 为 0.1, 用于平衡显示语义对齐损失的超参数 β 为 0.004, 每组数据 (源⇒目标) 测试 1 000 次. 并且所有实验都重复 20 次, 以减轻目标项目随机分割带来的偏差.

对比方法中的 CTKCCA、MSMDA 和 CLUSP 是根据 Chen 等人[7]的实验结果所得. 注意, MSMDA 通过其多源选择框架构造了一组独特的源项目作为目标项目的培训数据. 因此, 实际上只有 30 种预测组合. KSETE 是根据作者提供的源码得到的结果, CDAA 是根据作者提供的算法步骤进行复现而得到的结果. KSETE 和 CDAA 中涉及的实验参数与原作者保持一致.

3.5 统计分析

为了检验 SHSSAN 与其他方法相比能否获得统计意义上更好的预测性能, 我们使用了 3 种统计方法进行统计评估, 包括: Mann-Whitney U 检验、Cliff's δ 效应量检验[36]和 Scott-Knott ESD 检验[37].

3.5.1 Mann-Whitney U 检验

Mann-Whitney U 检验已用于异构缺陷预测研究中[17,19,22], 该方法可以不对数据的分布进行任何假设, 且不要求两个比较样本的大小相同, 因此我们选择此检验方法进行评估. 为了对预测结果进行统计分析, 我们在以 95% 的置信度进行了 20 次非参数 Mann-Whitney U 检验. 随后, 统计了 SHSSAN 与每个对比方法的 Win/Tie/Loss (W/T/L) 结果. "Win"表示 SHSSAN 的结果在 95% 的置信度上显著优于对比方法, "Tie"表示与对比方法性能相近, "Loss"则表示不如对比方法. 通过使用 W/T/L 评估, 可以发现 SHSSAN 在多少个项目上比其他方法表现更好.

3.5.2 Cliff's δ 效应量检验

此外, 为了衡量 SHSSAN 与对比方法之间的差异程度, 我们计算了 Cliff's δ, 这是一种非参数效应量检测. δ 是一种度量, 它表示一种方法中的值比另一种方法中的值大多少倍. δ 的所有可能值都在封闭区间 [−1, 1] 内, 其中 −1 或 1 表示一种方法的所有值均小于或大于另一种方法的值, 0 表示两种方法的度量完全重叠. 表 5 显示了不同 δ 值与其有效性水平之间的关系.

表 5 δ 值与其有效性级别大小对应关系

δ	有效性级别
$-1 \leqslant \lvert\delta\rvert < 0.147$	Negligible (N)
$0.147 \leqslant \lvert\delta\rvert < 0.33$	Small (S)
$0.33 \leqslant \lvert\delta\rvert < 0.474$	Medium (M)
$0.474 \leqslant \lvert\delta\rvert < 1$	Large (L)

3.5.3 Scott-Knott ESD 检验

Scott-Knott ESD 检验使用层次聚类分析将不同的处理值划分为统计学上不同的组, 其没有多重比较检验方法的重叠问题. 它是 Scott-Knott 检验的一种变体, 与 Scott-Knott 相比有两点变化: (1) 不假定数据是正态分布的; (2) 将效应量大小可忽略的任意两个统计学上不同的组合并为一组. 先前的一些 HDP 研究[19,21]使用了该方法.

4 实验结果和分析

4.1 SHSSAN 与 Baselines 对比

4.1.1 实验结果

为了探究 SHSSAN 的模型性能, 我们在 30 个异构缺陷项目上对 SHSSAN 和各种优秀的 HDP 方法进行实验. 在两个评估指标上进行全面地对比并得到了一些数据. 表 6 和表 7 分别是 F-measure 和 AUC 两个评估指标的实验结果. 表中的数据是所有异构项目到目标项目的平均结果. 如 AEEEM 中的 EQ 项目是其他 4 个数据集 (NASA、PROMISE、ReLink 和 SOFTLAB) 的 25 个项目的平均结果.

表 6 几种 HDP 方法在所有异构项目中 *F*-measure 的均值

数据集	目标项目	SHSSAN	CTKCCA	CLSUP	MSMDA	KSETE	CDAA
AEEEM	EQ	0.6880	**0.7195**	0.6137	0.6060	0.6317	0.5879
	JDT	**0.6207**	0.5213	0.5761	0.5740	0.5438	0.5917
	LC	0.5657	**0.8760**	0.3262	0.3361	0.4632	0.5576
	ML	**0.5635**	0.4709	0.3718	0.3456	0.4876	0.4987
	PDE	**0.5661**	0.5071	0.3749	0.3821	0.4897	0.5524
NASA	CM1	0.4736	**0.6755**	0.3142	0.3062	0.4378	0.5019
	MW1	**0.5633**	0.5079	0.2912	0.3020	0.3786	0.4317
	PC1	0.4738	**0.7350**	0.2456	0.2520	0.4376	0.5143
	PC3	0.4613	**0.7031**	0.3588	0.3704	0.4109	0.5019
	PC4	0.5750	0.5739	0.3808	0.3838	**0.6215**	0.4567
PROMISE	ant	**0.6402**	0.5882	0.5327	0.5440	0.5523	0.6109
	camel	**0.5346**	0.5214	0.3636	0.3630	0.4765	0.5178
	ivy	**0.6089**	0.5753	0.3545	0.3680	0.4786	0.5328
	jedit	0.6322	**0.8374**	0.5306	0.5368	0.5789	0.6128
	log4j	**0.6634**	0.5844	0.5947	0.5755	0.5743	0.6278
	lucene	0.5630	0.5797	**0.6311**	0.6265	0.5437	0.5209
	poi	0.6850	0.4919	0.7424	0.7692	0.6543	**0.6912**
	synapse	0.6300	**0.8717**	0.5902	0.5555	0.5897	0.5928
	tomcat	0.5833	0.6434	0.3254	0.3273	0.3789	**0.6541**
	velocity	0.6093	**0.7377**	0.5313	0.5490	0.5167	0.5517
	xalan	0.6080	**0.7409**	0.4224	0.4027	0.4643	0.5432
	xerces	**0.6215**	0.5804	0.4277	0.4217	0.4897	0.5743
ReLink	Apache	0.6320	**0.7241**	0.6130	0.5972	0.6235	0.6487
	Safe	**0.7204**	0.5009	0.6352	0.6351	0.6534	0.6826
	Zxing	0.5245	**0.6729**	0.4237	0.4284	0.4028	0.5987
SOFTLAB	AR1	**0.9653**	0.1771	0.1623	0.1536	0.3689	0.2574
	AR3	**0.9800**	0.2193	0.3667	0.3321	0.4765	0.5432
	AR4	**0.6101**	0.3979	0.4600	0.4817	0.4987	0.4357
	AR5	**0.9840**	0.2307	0.5684	0.6713	0.6543	0.7012
	AR6	**0.9960**	0.2910	0.3108	0.3187	0.4012	0.4567
平均		**0.6448**	0.5752	0.4480	0.4505	0.5093	0.5516

表 7 几种 HDP 方法在所有异构项目中 AUC 的均值

数据集	目标项目	SHSSAN	CTKCCA	CLSUP	MSMDA	KSETE	CDAA
AEEEM	EQ	0.7226	0.7179	**0.8143**	0.8084	0.7197	0.7413
	JDT	**0.8386**	0.6564	0.8121	0.8094	0.7543	0.7018
	LC	0.7576	**0.8501**	0.7935	0.7696	0.7434	0.7918
	ML	**0.7535**	0.6420	0.7437	0.6943	0.6954	0.6743
	PDE	0.6813	0.6524	0.7343	0.7406	0.6929	**0.7515**
NASA	CM1	**0.8376**	0.8208	0.6761	0.6716	0.7778	0.8028
	MW1	0.7674	**0.8924**	0.7149	0.7290	0.7254	0.7437
	PC1	0.7498	**0.9144**	0.7031	0.7271	0.7378	0.7543
	PC3	**0.8455**	0.7726	0.7366	0.7760	0.7543	0.7743
	PC4	**0.7665**	0.7270	0.7487	0.7580	0.7289	0.7403
PROMISE	ant	0.7027	0.6861	0.7686	**0.8032**	0.7452	0.7123
	camel	**0.7675**	0.6732	0.6309	0.6232	0.6851	0.6289
	ivy	0.7367	**0.8280**	0.7517	0.7882	0.7299	0.7456
	jedit	0.8285	**0.9353**	0.7499	0.7603	0.7613	0.8154
	log4j	**0.8274**	0.7784	0.8114	0.7983	0.7819	0.7356

表 7 几种 HDP 方法在所有异构项目中 AUC 的均值 (续)

数据集	目标项目	SHSSAN	CTKCCA	CLSUP	MSMDA	KSETE	CDAA
	lucene	0.6896	0.6765	0.6960	0.6741	**0.7024**	0.6728
	poi	**0.8068**	0.7532	0.7913	0.7869	0.7898	0.7314
	synapse	0.8254	**0.9011**	0.7325	0.7225	0.7543	0.8189
	tomcat	**0.8004**	0.7675	0.7783	0.8040	0.7738	0.7489
	velocity	0.7744	0.8436	0.7046	0.7088	0.7278	**0.8545**
	xalan	**0.8270**	0.8113	0.7566	0.7516	0.7586	0.7625
	xerces	0.8021	**0.8678**	0.7678	0.7505	0.7932	0.7329
ReLink	Apache	**0.7944**	0.7801	0.6977	0.6698	0.6968	0.7243
	Safe	**0.7770**	0.5876	0.7636	0.7691	0.7109	0.6897
	Zxing	0.7327	**0.8543**	0.6144	0.6007	0.6987	0.7268
SOFTLAB	AR1	**0.7109**	0.6825	0.5805	0.6526	0.6743	0.7009
	AR3	0.6876	0.5185	**0.7215**	0.7000	0.6627	0.6768
	AR4	**0.7642**	0.6704	0.7074	0.7353	0.7394	0.7548
	AR5	0.7256	0.5222	0.7817	0.8597	0.7765	**0.8634**
	AR6	**0.6832**	0.5976	0.6175	0.6539	0.6487	0.6735
平均		**0.7662**	0.7460	0.7300	0.7366	0.7314	0.7415

从表 6 和表 7 可知, 我们提出的 SHSSAN 方法在两个评价指标上都明显优于现存表现最优的 HDP 方法. 在总共 30 个项目上, 超过半数的项目占优. 其中在 *F*-measure 指标上优势明显, 比其他表现最优的方法提高了 6.96%. 与 5 种 HDP 方法中表现最优的 CTKCCA 相比, SHSSAN 也是所有方法中表现最稳定的. 如在 *F*-measure 指标上, SHSSAN 最高是 0.996 0、最低是 0.461 3, 相差 0.534 7; CTKCCA 最高是 0.876 0、最低是 0.177 1, 相差 0.698 9. CTKCCA 虽然在表现最好的项目上能接近 0.9, 但在表现差的项目上只有不到 0.2, 远低于 SHSSAN 中的 0.461 3. 这充分表明我们的方法不仅性能优异, 而且模型稳定, 能够适应多种异构的数据集.

SHSSAN 之所以表现优异且模型稳定, 究其原因是我们充分挖掘了数据中潜在的语义属性. 我们探索了从标记的源数据中学到的隐性知识, 从而在类别之间传递知识. 其次利用伪标签进行显示语义对齐从而提高模型的精度. 在充分利用了源和目标项目中的语义信息, 使得 SHSSAN 表现优于其他 HDP 方法.

4.1.2 统计分析结果

(1) Mann-Whitney U 检验

表 8 显示了 SHSSAN 相对于每种比较方法的 *F*-measure 和 AUC 的 W/T/L 结果. 在大多数情况下, SHSSAN 可以在统计学上显著提高模型的性能. 例如, 在 *F*-measure 上, SHSSAN 与 CTKCCA、CLSUP、MSMDA、KSETE 和 CDAA 相比, 分别在 15/30 (30 个项目中有 15 个项目)、25/30、26/30、23/30 和 16/30 上获得显著的改进.

表 8 SHSSAN 与各种比较方法在 *F*-measure 和 AUC 指标的 W/T/L 结果

对比方法	*F*-measure	AUC
SHSSAN vs. CTKCCA	15/5/10	15/6/9
SHSSAN vs. CLSUP	25/4/1	17/7/6
SHSSAN vs. MSMDA	26/2/2	19/5/6
SHSSAN vs. KSETE	23/4/3	19/8/3
SHSSAN vs. CDAA	16/8/6	14/8/8

(2) Cliff's δ 效应量检验

为了更清楚地比较预测结果, 我们根据表 4 对每个有效性级别中的项目数进行了计数, 统计结果如表 9 所示. 从表 9 我们可以看出, SHSSAN 在大多数情况下可产生显著性改善. 以 SHSSAN 与 MSMDA 的结果为例, 在 *F*-measure 和 AUC 值方面, SHSSAN 分别在 27/30 和 26/30 个项目中实现了不可忽略的差异.

表 9 在 *F*-measure 和 AUC 上 SHSSAN 与其他各种方法对比的有效性结果 (N/S/M/L)

对比方法	*F*-measure	AUC
SHSSAN vs. CTKCCA	5/6/4/15	6/5/7/12
SHSSAN vs. CLSUP	2/1/4/23	5/5/3/17
SHSSAN vs. MSMDA	3/2/4/21	4/2/8/16
SHSSAN vs. KSETE	4/3/5/18	5/4/7/14
SHSSAN vs. CDAA	5/4/5/16	4/7/6/13

(3) Scott-Knott ESD 检验

图 3 和图 4 分别针对 *F*-measure 和 AUC 中的所有项目展示了 SHSSAN 和对比的 HDP 方法的 Scott-Knott ESD 检验结果. *x* 轴代表比较的方法, *y* 轴表示排名值. 每种方法都对应一条垂直线, 表示该方法在所有项目上的排名范围. 垂直线中的点是平均排名, 不同颜色表示具有统计意义的不同组. 排名越小, 代表性能越好. 从图 3 和图 4 中可明显地看出: 对于 *F*-measure 和 AUC, SHSSAN 的平均排名最小, 且被分到一个不包含其他任何 HDP 方法的组, 这意味着 SHSSAN 明显优于其他方法.

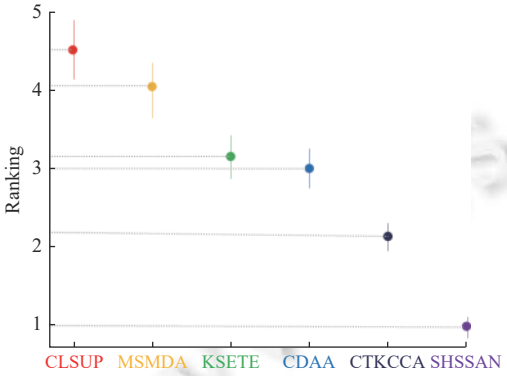

图 3 针对 SHSSAN 和其他 HDP 方法的所有项目在 *F*-measure 上进行 Scott-Knott ESD 检验的结果

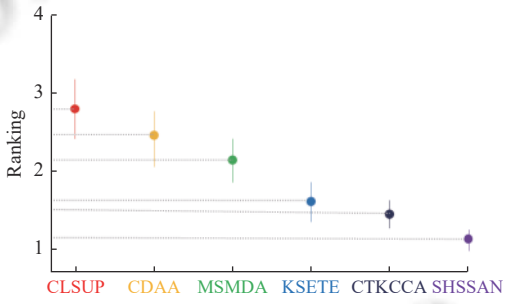

图 4 针对 SHSSAN 和其他 HDP 方法的所有项目在 AUC 上进行 Scott-Knott ESD 检验的结果

4.2 类不平衡问题对 SHSSAN 影响

为了验证类不平衡问题是否影响 SHSSAN 的性能以及验证各种处理类不平衡问题方法的优劣, 我们设计了实验并进行探究.

RQ1: 类不平衡问题是否影响 SHSSAN 的性能?

为了验证我们使用的类不平衡处理方法对 SHSSAN 模型的影响, 我们用未经类不平衡处理的原始数据进行了对比实验, 实验结果如表 10 所示. 其中“有”代表使用了类不平衡处理方法, “无”代表没有.

从表 10 可以明显地看出, 未使用类不平衡处理的原始数据所得的实验结果比使用的在全部 30 个项目上都低很多. 其中, 在 *F*-measure 平均低了 14.70%, 在 AUC 上平均低了 13.13%. 这充分说明了在设计异构缺陷预测模型时, 先处理数据中的类不平衡问题对模型性能的提高具有关键作用. 也说明了之前很多 HDP 工作考虑类不平衡问题是有必要且有意义的.

RQ2: 各种处理类不平衡问题方法对 SHSSAN 影响多大?

同时, 为了探究各种类不平衡处理方法对 SHSSAN 的影响大小, 我们对当前缺陷预测领域常用的一些类不平衡处理方法 (过采样方法 (本文使用)、欠采样方法以及代价敏感方法) 进行了分析实验. 我们分别对原始的缺陷项目使用这 3 种类不平衡处理方法进行数据处理, 然后将处理好的项目进行同步语义对齐进行异构缺陷预测. 实验中欠采样方法是随机删除项目中的无缺陷样本使项目中两类样本个数达到平衡, 代价敏感方法对于缺陷样本和无缺陷样本赋予的权重分别是 0.7 和 0.3 (多次实验所得), 实验结果如图 5 所示.

表 10 类不平衡问题对 SHSSAN 的影响

数据集	项目	*F*-measure (有)	*F*-measure (无)	AUC (有)	AUC (无)
AEEEM	EQ	0.6880	0.5723	0.7226	0.5842
	JDT	0.6207	0.5412	0.8386	0.6959
	LC	0.5657	0.4319	0.7576	0.6348
	ML	0.5635	0.4486	0.7535	0.6158
	PDE	0.5661	0.4218	0.6813	0.5341
NASA	CM1	0.4736	0.3587	0.8376	0.7258
	MW1	0.5633	0.4165	0.7674	0.6359
	PC1	0.4738	0.4136	0.7498	0.6214
	PC3	0.4613	0.3029	0.8455	0.7258
	PC4	0.5750	0.4369	0.7665	0.6253
PROMISE	ant	0.6402	0.4887	0.7027	0.5574
	camel	0.5346	0.3964	0.7675	0.6422
	ivy	0.6089	0.4812	0.7367	0.6038
	jedit	0.6322	0.4732	0.8285	0.6937
	log4j	0.6634	0.5039	0.8274	0.7026
	lucene	0.5630	0.4189	0.6896	0.5587
	poi	0.6850	0.5347	0.8068	0.6725
	synapse	0.6300	0.4772	0.8254	0.7115
	tomcat	0.5833	0.4365	0.8004	0.6687
	velocity	0.6093	0.4418	0.7744	0.6358
	xalan	0.6080	0.4298	0.8270	0.7019
	xerces	0.6215	0.4536	0.8021	0.6852
ReLink	Apache	0.6320	0.4268	0.7944	0.6587
	Safe	0.7204	0.5818	0.7770	0.6443
	Zxing	0.5245	0.4011	0.7327	0.6023
SOFTLAB	AR1	0.9653	0.8036	0.7109	0.5987
	AR3	0.9800	0.7958	0.6876	0.5514
	AR4	0.6101	0.4259	0.7642	0.6548
	AR5	0.9840	0.8069	0.7256	0.5503
	AR6	0.9960	0.8126	0.6832	0.5532
平均		**0.6448**	**0.4978**	**0.7662**	**0.6349**

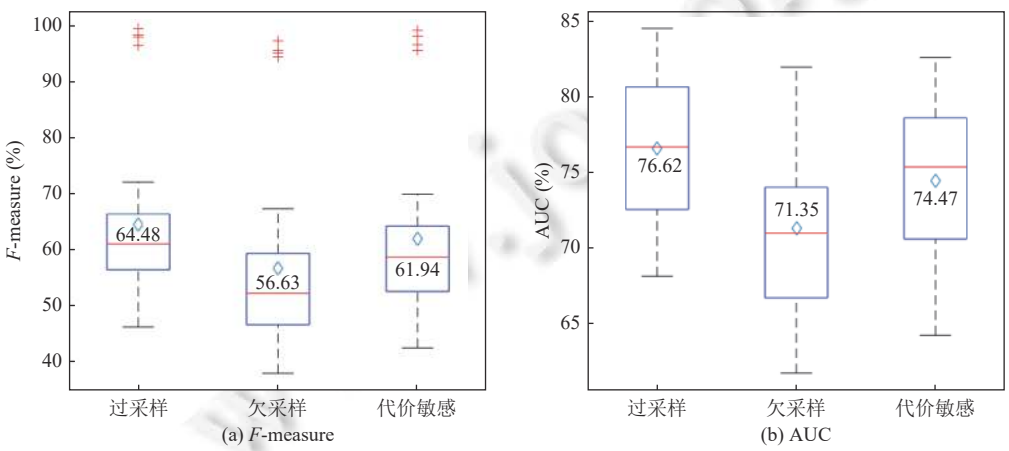

图 5 各种类不平衡处理方法对 SHSSAN 的影响

从图 5 可以明显地看出, 过采样方法在处理类不平衡问题时具有显著的优势, 而欠采样方法效果最差. 在 *F*-measure 和 AUC 上过采样方法分别比欠采样方法和代价敏感方法平均提高了 7.85%、2.54% 和 5.27%、2.15%. 建议以后在研究异构缺陷预测时, 处理类不平衡问题优先使用过采样方法.

4.3 已标记目标项目对 SHSSAN 的影响

为了验证目标项目中已标记样本对模型构建是否有影响, 以及寻找出目标项目中多少已标记样本参与模型构建可以达到最佳的效果, 我们进行了多组对比实验.

RQ1: 目标项目中已标记样本是否对 SHSSAN 有影响?

为了验证已标记的目标项目对 SHSSAN 模型的影响, 我们单独使用源项目数据进行了对比实验, 实验结果如表 11 所示. 其中“20%”代表使用了随机选取的目标项目中 20% 的已标记样本, “0”代表没有使用目标样本.

表 11 已标记的目标项目对 SHSSAN 的影响

数据集	项目	*F*-measure (20%)	*F*-measure (0)	AUC (20%)	AUC (0)
AEEEM	EQ	0.6880	0.5026	0.7226	0.4946
	JDT	0.6207	0.4615	0.8386	0.6123
	LC	0.5657	0.3716	0.7576	0.5447
	ML	0.5635	0.3788	0.7535	0.5258
	PDE	0.5661	0.3523	0.6813	0.4746
NASA	CM1	0.4736	0.2889	0.8376	0.6557
	MW1	0.5633	0.3449	0.7674	0.5254
	PC1	0.4738	0.3432	0.7498	0.5025
	PC3	0.4613	0.2267	0.8455	0.6359
	PC4	0.5750	0.3652	0.7665	0.5357
PROMISE	ant	0.6402	0.4267	0.7027	0.4864
	camel	0.5346	0.3255	0.7675	0.5627
	ivy	0.6089	0.4198	0.7367	0.5136
	jedit	0.6322	0.4023	0.8285	0.6302
	log4j	0.6634	0.3727	0.8274	0.6215
	lucene	0.5630	0.3481	0.6896	0.4983
	poi	0.6850	0.3967	0.8068	0.5841
	synapse	0.6300	0.4025	0.8254	0.6328
	tomcat	0.5833	0.3478	0.8004	0.5786
	velocity	0.6093	0.3756	0.7744	0.5328
	xalan	0.6080	0.3598	0.8270	0.5846
	xerces	0.6215	0.3875	0.8021	0.5741
ReLink	Apache	0.6320	0.3589	0.7944	0.5726
	Safe	0.7204	0.5146	0.7770	0.5632
	Zxing	0.5245	0.3344	0.7327	0.5427
SOFTLAB	AR1	0.9653	0.7283	0.7109	0.4896
	AR3	0.9800	0.7364	0.6876	0.5087
	AR4	0.6101	0.3547	0.7642	0.5503
	AR5	0.9840	0.7368	0.7256	0.4628
	AR6	0.9960	0.7525	0.6832	0.4536
平均		**0.6448**	**0.4239**	**0.7662**	**0.5483**

表 11 详细列出了使用 20% 已标记的目标项目与未使用目标项目 (0%) 的对比情况, 可以看出使用了目标项目中的已标记样本对 SHSSAN 性能提升明显. 其中, 在 *F*-measure 平均提高了 22.09%, 在 AUC 上平均提高了 21.79%. 之所以性能提升明显是因为 SHSSAN 可以挖掘出已标记的目标项目与源项目之间隐藏的概率分布信息, 即分类相同的样本具有类似的概率分布; 同时这些已标记的目标项目能够指导其与源项目和未标记的目标项目进

行显示语义对齐, 通过隐式语义相关迁移和显示语义对齐使得 SHSSAN 可以更加精准地预测出未标记目标项目的类别.

RQ2: 多少已标记目标项目参与模型构建可以达到最佳效果?

从 RQ1 我们已经知道已标记的目标项目能够提高 SHSSAN 的性能, 为了进一步寻找出目标项目中多少已标记项目参与模型构建可以达到最佳的效果, 我们分别使用 5%、10%、15%、…、40%、45% 和 50% 的随机选取已标记目标项目进行实验, 实验结果如图 6 和图 7 所示.

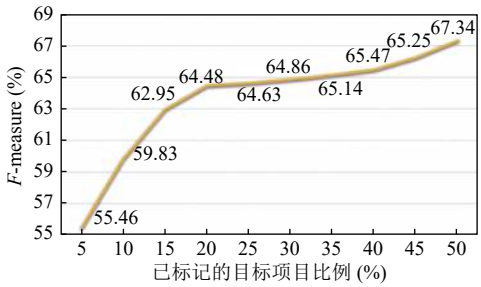

图 6 不同比例的已标记目标项目对 *F*-measure 的影响

图 7 不同比例的已标记目标项目对 AUC 的影响

从图 6 和图 7 中可以看出, 当增加目标项目中已标记样本的比率时, *F*-measure 和 AUC 都呈现增加的趋势, 这说明增加目标项目已标记样本的个数有助于提高模型的性能. 此外, 我们也发现当目标项目中已标记样本比例达到 20% 之后, *F*-measure 和 AUC 增加较缓慢, 这表明当取 20% 的已标记目标项目时, 模型基本达到最佳的状态, 此时再增加目标项目中已标记样本的个数, 对模型整体的影响效果不再明显. 因此, 我们在实验中使用 20% 的已标记目标项目参与到模型的构建中.

4.4 隐式语义相关迁移和显式语义对齐对 SHSSAN 的影响

为了验证隐式语义相关迁移和显示语义对齐对模型性能是否有影响, 我们进行了消融实验. 我们分别验证了单独使用隐式语义相关迁移、显示语义对齐以及二者同时不使用 (即表 12 中 none 列) 的实验结果, 并与原始的 SHSSAN 进行对比 (即表 12 中 full 列). 对于公式 (10), 我们分别令 α=0、β=0 即可验证单独使用其中一种方法对模型整体的影响, 同时令 α=0、β=0 即表示二者都不使用.

表 12 隐式语义相关迁移和显示语义对齐分别对 SHSSAN 的影响

指标	α=0	β=0	none	full
F-measure	0.5342	0.5487	0.3765	0.6448
AUC	0.6543	0.6649	0.4832	0.7662

表 12 是消融验证的结果, 当隐式语义相关迁移和显示语义对齐都不使用时, 模型效果很差, 在 *F*-measure 和 AUC 上比全部使用的分别低了 26.83% 和 28.30%; 如果只使用隐式或显式其中一种, 其模型性能比都不使用提升很多, 但仍然要比同时使用的效果差. 其中, 当只使用显式语义对齐时 (即 α=0) 模型效果要比只使用隐式语义相关迁移 (即 β=0) 略差. 通过消融实验可知: 隐式语义相关迁移和显式语义对齐对模型的整体性能都有着不可或缺的作用, 二者有机结合能使 SHSSAN 发挥最佳的效果.

5 效度威胁

在本文中, 由于进行了广泛的实验研究, 因此应考虑对有效性的一些潜在威胁. 为了解决我们工作中观察到的潜在局限性, 以下内容讨论了对结构、内部和外部有效性的威胁.

5.1 结构有效性

我们提出的方法假设可以将缺陷类适当地分为两个类, 但是对于缺陷项目可能并非总是如此. 一些缺陷项目

可能有多个类别组成, 我们简单地将样本划分成缺陷和非缺陷两个类, 可能对实验结果有所影响. 但是这些类别中的多数类别一般都是包含缺陷的样本, 这种二分类的方法对实验结果影响甚小. 其次, 实验过程中采样和分割方法可能会影响分类结果. 实验中每次迭代都是从项目中随机抽取数据可能会对实验结果造成影响, 但我们增加了实验次数 (20 次) 以最大限度地降低实验数据的随机性对实验结果的影响. 最后, 由于没有通用的评估指标来评估不平衡数据的预测性能, 我们使用了两种不同的指标: *F*-measure 和 AUC 来评估我们的方法与其他 HDP 方法的优劣. 其他评估指标上是否具有相同的性能, 需要进一步进行验证.

5.2 内部有效性

一方面, 我们提出的隐式语义相关迁移是基于分类器对于异构域中的同一类别应产生相似的分类概率分布的假设, 真实的异构域中相同类别是否遵循类似的分类概率有待进一步验证. 另一方面, 在显式语义对齐中, 为了避免伪标签的不确定性, 利用基础数据的内在几何知识为那些与特征空间中受监督数据的类别质心呈现几何相似性的样本分配伪标签. 关于质心相似性来分配伪标签的合理性也有待进一步验证.

5.3 外部有效性

首先, 我们的实验使用了来自 5 个不同数据集的 30 个项目, 充分考虑了数据集的异构性. 但是由于精力有限, 我们不可能使用所有的异构缺陷项目, 因此我们方法在其他数据集上的表现如何还未可知. 其次, 我们的实验数据集都是来源于开源项目, 对于在商业软件中的效果如何还有待验证. 最后, 我们尽可能多的与优秀的 HDP 方法进行比较, 但是还是有很多优秀的 HDP 方法没有与之比较, 我们计划在将来的工作中完成.

6 总 结

本文提出了一种新颖的基于同步语义对齐的异构缺陷预测方法来解决 HDP 问题. 与现有的 HDP 方法相比, 我们不仅解决了 HDP 中常见的数据线性不可分和类不平衡问题, 还同时挖掘了隐式和显式语义知识, 以促进跨域异构数据之间的对齐. 隐式语义知识有助于更好地保存分类分布相关性, 而带有几何语义标签细化的显式对齐过程则可增强跨域语义对齐. 在 30 个公共异构数据集上进行了大量对比实验, 针对 *F*-measure 和 AUC 两个常用指标进行测试, 实验结果表明: 与目前表现最好的几个 HDP 方法相比, 我们的方法更为有效.

在未来的工作中, 我们将探索 SHSSAN 与更多优秀的 HDP 方法进行比较、在更多评估指标上的性能表现以及在更多异构缺陷预测项目上的应用.

References:

[1] Wang S, Liu TY, Tan L. Automatically learning semantic features for defect prediction. In: Proc. of the 38th IEEE/ACM Int'l Conf. on Software Engineering. Austin: IEEE, 2016. 297–308. [doi: 10.1145/2884781.2884804]

[2] Li ZQ, Jing XY, Wu F, Zhu XK, Xu BW, Ying S. Cost-sensitive transfer kernel canonical correlation analysis for heterogeneous defect prediction. Automated Software Engineering, 2018, 25(2): 201–245. [doi: 10.1007/s10515-017-0220-7]

[3] Chen X, Gu Q, Liu WS, Liu SL, Ni C. Survey of static software defect prediction. Ruan Jian Xue Bao/Journal of Software, 2016, 27(1): 1–25 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4923.htm [doi: 10.13328/j.cnki.jos.004923]

[4] Ma Y, Luo GC, Zeng X, Chen AG. Transfer learning for cross-company software defect prediction. Information and Software Technology, 2012, 54(3): 248–256. [doi: 10.1016/j.infsof.2011.09.007]

[5] Nam J, Pan SJ, Kim S. Transfer defect learning. In: Proc. of the 35th Int'l Conf. on Software Engineering. San Francisco: IEEE, 2013. 382–391. [doi: 10.1109/ICSE.2013.6606584]

[6] Xu Z, Yuan PP, Zhang T, Tang YT, Li S, Xia Z. HDA: Cross-project defect prediction via heterogeneous domain adaptation with dictionary learning. IEEE Access, 2018, 6: 57597–57613. [doi: 10.1109/ACCESS.2018.2873755]

[7] Chen HW, Jing XY, Li ZQ, Wu D, Peng Y, Huang ZG. An empirical study on heterogeneous defect prediction approaches. IEEE Trans. on Software Engineering, 2021, 47(12): 2803–2822. [doi: 10.1109/TSE.2020.2968520]

[8] Chen X, Wang LP, Gu Q, Wang Z, Ni C, Liu WS, Wang QP. A survey on cross-project software defect prediction methods. Chinese Journal of Computers, 2018, 41(1): 254–274 (in Chinese with English abstract). [doi: 10.11897/SP.J.1016.2018.00254]

[9] Turhan B, Mısırlı AT, Bener A. Empirical evaluation of the effects of mixed project data on learning defect predictors. Information and

Software Technology, 2013, 55(6): 1101–1118. [doi: 10.1016/j.infsof.2012.10.003]

[10] Xia X, Lo D, Pan SJ, Nagappan N, Wang XY. Hydra: Massively compositional model for cross-project defect prediction. IEEE Trans. on Software Engineering, 2016, 42(10): 977–998. [doi: 10.1109/TSE.2016.2543218]

[11] Nam J, Fu W, Kim S, Menzies T, Tan L. Heterogeneous defect prediction. IEEE Trans. on Software Engineering, 2018, 44(9): 874–896. [doi: 10.1109/TSE.2017.2720603]

[12] Li S, Xie BH, Wu JS, Zhao Y, Liu CH, Ding ZM. Simultaneous semantic alignment network for heterogeneous domain adaptation. In: Proc. of the 28th ACM Int'l Conf. on Multimedia. Lisboa: ACM, 2020. 3866–3874. [doi: 10.1145/3394171.3413995]

[13] He P, Li B, Ma YT. Towards cross-project defect prediction with imbalanced feature sets. arXiv: 1411.4228, 2014.

[14] Yu Q, Jiang SJ, Zhang YM. A feature matching and transfer approach for cross-company defect prediction. Journal of Systems and Software, 2017, 132: 366–378. [doi: 10.1016/j.jss.2017.06.070]

[15] Jing XY, Wu F, Dong XW, Qi FM, Xu BW. Heterogeneous cross-company defect prediction by unified metric representation and CCA-based transfer learning. In: Proc. of the 10th Joint Meeting on Foundations of Software Engineering. Singapore: Association for Computing Machinery, 2015. 496–507. [doi: 10.1145/2786805.2786813]

[16] Cheng M, Wu GQ, Jiang M, Wan HY, You GA, Yuan MT. Heterogeneous defect prediction via exploiting correlation subspace. In: Proc. of the 28th Int'l Conf. on Software Engineering and Knowledge Engineering. Redwood: KSI Research Inc. and Knowledge Systems Institute Graduate School, 2016. 171–176.

[17] Li ZQ, Jing XY, Zhu XK, Zhang HY. Heterogeneous defect prediction through multiple kernel learning and ensemble learning. In: Proc. of the 2017 IEEE Int'l Conf. on Software Maintenance and Evolution. Shanghai: IEEE, 2017. 91–102. [doi: 10.1109/ICSME.2017.19]

[18] Li ZQ, Jing XY, Zhu XK. Heterogeneous fault prediction with cost-sensitive domain adaptation. Software Testing, Verification and Reliability, 2018, 28(2): e1658. [doi: 10.1002/stvr.1658]

[19] Li ZQ, Jing XY, Zhu XK, Zhang HY, Xu BW, Ying S. Heterogeneous defect prediction with two-stage ensemble learning. Automated Software Engineering, 2019, 26(3): 599–651. [doi: 10.1007/s10515-019-00259-1]

[20] Xu Z, Ye SZ, Zhang T, Xia Z, Pang S, Wang Y, Tang YT. MVSE: Effort-aware heterogeneous defect prediction via multiple-view spectral embedding. In: Proc. of the 19th IEEE Int'l Conf. on Software Quality, Reliability and Security. Sofia: IEEE, 2019. 10–17. [doi: 10.1109/QRS.2019.00015]

[21] Tong HN, Liu B, Wang SH. Kernel spectral embedding transfer ensemble for heterogeneous defect prediction. IEEE Trans. on Software Engineering, 2021, 47(9): 1886–1906. [doi: 10.1109/TSE.2019.2939303]

[22] Li ZQ, Jing XY, Zhu XK, Zhang HY, Xu BW, Ying S. On the multiple sources and privacy preservation issues for heterogeneous defect prediction. IEEE Trans. on Software Engineering, 2019, 45(4): 391–411. [doi: 10.1109/TSE.2017.2780222]

[23] Gong LN, Jiang SJ, Jiang L. Conditional domain adversarial adaptation for heterogeneous defect prediction. IEEE Access, 2020, 8: 150738–150749. [doi: 10.1109/ACCESS.2020.3017101]

[24] Wang AL, Zhang YT, Wu HB, Jiang KY, Wang MH. Few-shot learning based balanced distribution adaptation for heterogeneous defect prediction. IEEE Access, 2020, 8: 32989–33001. [doi: 10.1109/ACCESS.2020.2973924]

[25] Wang C, Mahadevan S. Heterogeneous domain adaptation using manifold alignment. In: Proc. of the 22nd Int'l Joint Conf. on Artificial Intelligence. Barcelona: AAAI Press, 2011. 1541–1546.

[26] Li W, Duan LX, Xu D, Tsang IW. Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2014, 36(6): 1134–1148. [doi: 10.1109/TPAMI.2013.167]

[27] Chen WY, Hsu TMH, Tsai YHH, Wang YCF, Chen MS. Transfer neural trees for heterogeneous domain adaptation. In: Proc. of the 14th European Conf. on Computer Vision. Amsterdam: Springer, 2016. 399–414. [doi: 10.1007/978-3-319-46454-1_25]

[28] Tsai YHH, Yeh YR, Wang YCF. Learning cross-domain landmarks for heterogeneous domain adaptation. In: Proc. of the 29th IEEE Conf. on Computer Vision and Pattern Recognition. Las Vegas: IEEE, 2016. 5081–5090. [doi: 10.1109/CVPR.2016.549]

[29] Yao Y, Zhang Y, Li XT, Ye YM. Heterogeneous domain adaptation via soft transfer network. In: Proc. of the 27th ACM Int'l Conf. on Multimedia. Nice: Association for Computing Machinery, 2019. 1578–1586. [doi: 10.1145/3343031.3350955]

[30] Japkowicz N, Stephen S. The class imbalance problem: A systematic study. Intelligent Data Analysis, 2002, 6(5): 429–449. [doi: 10.3233/IDA-2002-6504]

[31] Weiss K, Khoshgoftaar TM, Wang DD. A survey of transfer learning. Journal of Big Data, 2016, 3(1): 9. [doi: 10.1186/s40537-016-0043-6]

[32] Hsieh YT, Tao SY, Tsai YHH, Yeh YR, Wang YCF. Recognizing heterogeneous cross-domain data via generalized joint distribution adaptation. In: Proc. of the 9th IEEE Int'l Conf. on Multimedia and Expo. Seattle: IEEE, 2016. 1–6. [doi: 10.1109/ICME.2016.7552878]

[33] Pan YW, Yao T, Li YH, Wang Y, Ngo CW, Mei T. Transferrable prototypical networks for unsupervised domain adaptation. In: Proc. of the 2019 IEEE/CVF Conf. on Computer Vision and Pattern Recognition. Long Beach: IEEE, 2019. 2234–2242. [doi: 10.1109/CVPR.

2019.00234]
[34] Tzeng E, Hoffman J, Zhang N, Saenko K, Darrell T. Deep domain confusion: Maximizing for domain invariance. arXiv:1412.3474, 2014.
[35] Li ZQ, Jing XY, Zhu XK. Progress on approaches to software defect prediction. IET Software, 2018, 12(3): 161–175. [doi: 10.1049/iet-sen.2017.0148]
[36] Cliff N. Ordinal Methods for Behavioral Data Analysis. New York: Psychology Press, 2014.
[37] Tantithamthavorn C, McIntosh S, Hassan AE, Matsumoto K. The impact of automated parameter optimization on defect prediction models. IEEE Trans. on Software Engineering, 2019, 45(7): 683–711. [doi: 10.1109/TSE.2018.2794977]

附中文参考文献:

[3] 陈翔, 顾庆, 刘望舒, 刘树龙, 倪超. 静态软件缺陷预测方法研究. 软件学报, 2016, 27(1): 1−25. http://www.jos.org.cn/1000-9825/4923.htm [doi: 10.13328/j.cnki.jos.004923]
[8] 陈翔, 王莉萍, 顾庆, 王赞, 倪超, 刘望舒, 王秋萍. 跨项目软件缺陷预测方法研究综述. 计算机学报, 2018, 41(1): 254–274. [doi: 10.11897/SP.J.1016.2018.00254]

李伟漳(1981－), 女, 博士, 副研究员, CCF 专业会员, 主要研究领域为机器学习, 软件可靠性.

黄志球(1965－), 男, 博士, 教授, CCF 杰出会员, 主要研究领域为软件工程, 软件安全性, 形式化方法.

陈翔(1980－), 男, 博士, 副教授, CCF 高级会员, 主要研究领域为软件缺陷预测, 软件缺陷定位, 回归测试, 组合测试.

贾修一(1983－), 男, 博士, 副教授, CCF 高级会员, 主要研究领域为机器学习, 粒计算, 数据挖掘.

张恒伟(1994－), 男, 硕士生, 主要研究领域为机器学习, 软件缺陷预测.