Software Engineering and Applications 软件工程与应用, 2023, 12(6), 930-939
Published Online December 2023 in Hans. https://www.hanspub.org/journal/sea
https://doi.org/10.12677/sea.2023.126091

Hans汉斯

一种邻域合成的软件缺陷预测过采样方法

纪晨辉，李英梅

哈尔滨师范大学计算机科学与信息工程学院，黑龙江 哈尔滨

收稿日期：2023年11月7日；录用日期：2023年12月19日；发布日期：2023年12月27日

摘　要

在软件缺陷预测中，数据集的类不平衡是其主要挑战之一。针对该问题，提出了一种邻域合成的过采样方法，该方法充分考虑了缺陷类样本近邻的特征信息，参照周围近邻样本的特征信息合成新样本，使得生成的样本更加多样化，并在缺陷样本较少的区域生成更多的样本，用来增强模型对稀疏缺陷类样本的识别能力。以AEEEM数据集和NASA数据集作为样本，采用F1值作为评估标准进行实验，实验结果显示，这种方法优于其他传统的过采样算法。

关键词

软件缺陷预测，类不平衡，过采样

A Neighborhood Synthesis Software Defect Prediction Oversampling Method

Chenhui Ji, Yingmei Li

School of Computer Science and Information Engineering, Harbin Normal University, Harbin Heilongjiang

Received: Nov. 7th, 2023; accepted: Dec. 19th, 2023; published: Dec. 27th, 2023

Abstract

In software defect prediction, the class imbalance of the data set is one of its main challenges. To solve this problem, a neighborhood synthesis oversampling method is proposed. This method fully considers the characteristic information of the nearest neighbors of defective samples, and synthesizes new samples with reference to the characteristic information of surrounding nearby samples, making the generated samples more diverse and generating more samples in areas with fewer defective samples to enhance the model for identifying sparse defective samples. The AEEEM data set and NASA data set were used as samples, and the F1 value was used as the evaluation criterion for experiments. The experimental results show that this method is better than other tradi-

文章引用: 纪晨辉, 李英梅. 一种邻域合成的软件缺陷预测过采样方法[J]. 软件工程与应用, 2023, 12(6): 930-939.
DOI: 10.12677/sea.2023.126091

tional oversampling algorithms.

Keywords

Software Defect Prediction, Class Imbalance, Oversampling

1. 引言

软件缺陷预测在复杂的软件开发过程中扮演了重要的角色。在软件开发过程中常常存在例如代码错误、设计缺陷等问题，这些缺陷可能导致软件崩溃、功能失效、安全漏洞甚至数据泄露。经过研究表明，对软件缺陷进行修复的成本往往占到总成本的一半以上[1]，及时地预测和修复缺陷有助于提高软件的质量和稳定性，确保系统可靠运行。传统的缺陷检测方法往往依赖手工代码审查，但面对庞大的代码库和时间压力，这样的方式往往非常耗时耗力。然而软件缺陷预测技术不仅解决了以上的问题，还能够帮助开发团队在软件生命周期的早期阶段识别和解决潜在的缺陷问题，从而降低维护成本、提高软件质量。通过利用历史缺陷数据和机器学习技术，可以构建预测模型来识别高风险的代码区域，使开发人员能够有针对性地进行测试和修复[2]。软件缺陷预测在提升软件质量、降低维护成本、保障数据安全等方面具有深远的意义。然而，在数据分布均衡的情况下，利用历史数据训练出来的预测模型在预测软件项目是否存在缺陷时通常是非常有效的。但在医学领域、破产预测领域、自然语言处理领域、电子商务领域、恶意软件检测领域以及各种计算机视觉应用领域中大多数分类问题的不同类别之间的样本分布都是高度不均衡的，这被称为类不平衡问题，同时这也是机器学习领域的一个主要难题[3]。类不平衡问题意味着无缺陷样本的数量远远多于有缺陷样本的数量。在二分类应用中，当类别分布不均匀时，模型在预测时会更倾向于出现频率较高的类别，这会严重影响模型的效率，并产生不平衡的假阳性和假阴性。大多数的机器学习技术在各类别的实例数量大致相等的情况下会有更好的性能[4]。

采样法是解决类不平衡问题常用的方法，该方法易于实施，无需复杂的参数调整，适用于各种应用场景和用户层次。采样方法可以有效缓解数据的类不平衡问题，使模型更加平衡地关注不同类别的样本，从而提升预测的准确性。广泛使用的欠采样方法有 RUS、Cluster Centroids、Tomek Links，以及 NearMiss 等，欠采样方法主要是通过减少数量较多类别的样本来平衡类别分布，提高模型对少数类别的分类能力，同时降低训练数据的规模，从而减少了模型训练所需的时间，但是该方法在去除多数类样本同时也可能会导致信息损失，降低模型的性能。ROS，SMOTE，Borderline-SMOTE 等是常用的过采样方法，过采样方法通过合成样本来平衡数据分布，这不仅可以更好地保留原始数据分布的特点，还可以帮助模型更充分地学习到少数类别的特征，从而提高预测性能。相比于直接删除样本(欠采样)，过采样避免了信息的丢失。现有的过采样算法使用插值的方法来合成样本，但这种方法可能存在一定的局限性，因为它仅以两个点为参考，未充分考虑周围其他样本的信息，使得合成样本的方式过于单一。

针对以上问题，本文提出了一种基于邻域的合成的过采样方法，该方法充分考虑边界点近邻样本的特征信息来合成样本，并根据该样本近邻多数类和少数类的数量差异动态地调整合成样本的数量，在缓解数据分布不平衡的同时也使得生成的样本更具有多样性，本文以 AEEEM 和 NASA 数据集，将该方法同原始数据、随机过采样方法、随机欠采样方法、SMOTE 过采样方法、Borderline Smote 过采样方法、

Safe Level Smote 过采样方法、K-means SMOTE 过采样方法、KNNOR 过采样方法进行对比实验，实验验证该方法是有效的。

2. 相关研究

2.1. 软件缺陷预测研究

软件缺陷预测是软件工程领域的一个关键活动，目的是在软件开发早期识别和预测可能出现的缺陷，从而可以采取适当的措施来减少缺陷数量和影响。其有三个主要步骤，首先是收集历史缺陷数据集并提取度量元，然后利用收集的数据使用机器学习或深度学习等方法构建软件缺陷预测模型，最后结合性能指标评价模型的性能。软件缺陷预测的流程如图 1 所示。

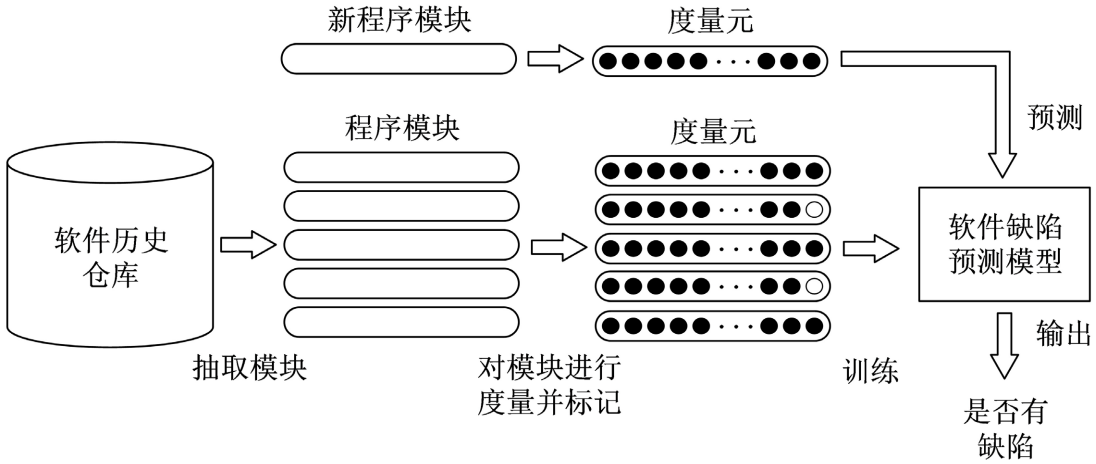

Figure 1. Software defect prediction process
图 1. 软件缺陷预测过程

采样法是在收集历史缺陷数据集并提取度量元之后，针对数据不平衡问题进行处理的一种方法，包括欠采样和过采样以及将这两种方法组合为混合采样方法。

2.2. 不平衡数据采样方法研究

欠采样算法通过减少多数类样本的数量来平衡数据集[5]。随机欠采样方法是从多数类别中随机删除一部分样本，使得样本数目更加均衡。该算法的优势是算法简单、易于实施，但是当样本极度失衡时，由于过度剔除了大量的类别样本会造成信息的过渡损失。基于聚类的欠采样方法(CUS) [6]首先通过对数据集进行聚类，然后删除一些样本来减少多数类样本的数量，实验结果表明该方法取得了较高的准确率。文献[7]提出了一种改进的 K-means 欠采样，其通过聚类分布以及多数类到少数类聚类中心的距离来选择样本，该方法可以保持数据的原始分布，提高性能。Tsai 等[8]提出了基于聚类的实例选择方法(CBIS)，该方法首先使用聚类的方法将相似的多数类样本分组，随后删除没有代表性的多数类来提升分类性能。周等[9]提出了基于聚类欠采样的集成分类算法，该算法使用聚类的方法，在簇中心的邻域内选出特征明显的负样本并形成新的训练集，将该训练集放到代价敏感调整函数的集成算法中，使模型更加关注少数类。Somya [10]提出了基于邻域的欠采样 N-US 算法，该算法不仅提高了少数类样本的可见性，还避免了因大量删除多数类样本所造成的信息丢失问题。实验证明该方法具有强大的预测能力，置信度达到 95%。

尽管这些欠采样方法可以在一些情况下帮助解决数据的不平衡问题，但它们也可能存在一些问题。某些情况下，欠采样可能会导致信息损失，降低模型的泛化能力。然而，过采样方法与欠采样的减少多

数类样本的方式相反，其是通过增加少数类样本的数量以平衡类别分布。SMOTE [11]是一种人工合成样本的过采样方法。与随机过采样不用是，他不是单纯的复制现有样本，而是通过线性插值的方式合成新的少数类样本。具体做法是选择一个少数类别的样本，然后随机选择这个样本的一个最近邻样本，在这两个样本之间的线段上随机选取一个点，生成新样本。Borderline SMOTE [12]是 SMOTE 算法的一个改进方法，它在传统的 SMOTE 算法的基础上进行了优化，专注于生成更具有边界信息的合成样本，从而提升分类器在不平衡数据情境下的性能。该方法的关键创新在于两个阶段的设计：边界样本识别和合成样本生成。首先，根据少数类样本周围不同类别样本的分布，识别出数据集中的边界样本。接着，在边界样本的近邻中随机选则多数类样本进行插值生成新的合成样本。这样的策略保证了生成的合成样本更能够反映数据集的边界特性，从而有效地提升分类器的性能。ADASYN [13]是一种自适应过采样算法，它根据样本的密度来调整合成样本的数量。该方法会为每个少数类样本计算其在特征空间中的密度，并根据密度来确定需要生成的合成样本数量。这样可以在样本密度较低的区域生成更多的合成样本。Ashhadul IslaM 等人提出了 k-最近邻过采样方法(KNNOR) [14]，该算法在生成样本点时，考虑了整个种群的相对密度,这使得该方法能够更可靠地对少数类进行过采样。同时保持抗噪声的弹性 Kmeans-SMOTE [15]算法也首先利用 K-means 聚类技术将样本分成 k 个簇，并保留了那些少数类样本比例较高的簇。然后根据簇的稀疏度不同来调整生成样本的数量，为少数类稀疏的簇生成更多的样本。最后，该算法在这些簇内使用 SMOTE 方法插值生成新样本。饶珍丹等[16]提出了 AJCC-Ram 的过采样方法，通过结合并改进 ADASYN 和 CURE-SMOTE 方法，将样本分为边界区域和中心区域，然后在这两个区域中生成样本，对合成后的数据集中的噪声进行处理，实验结果表明该方法提升了预测结果。Paria 等[17]提出了范围控制合成少数过采样技术，该方法首先针对 SMOTE 方法对所有少数类样本的无差别采样问题设计了一种识别适合用来过采样样本的方法，然后提出了一种改进的样本生成方法，使得生成的样本避免进入多数类样本区域内。文献[18]提出了一种基于径向基函数的过采样方法(ROB)，该方法通过使用径向基函数从少数类中找到适当的区域，以生成合成示例。Maldonado 等人[19]提出了一种新颖的类似于 SMOTE 的方法，即 FW-SMOTE，该方法使用加权闵可夫斯基来度量样本间的距离，选择出对分类影响较大的特征，从而更好地选择出样本的近邻。该方法与传统的 SMOTE 方法和其他最近的变种相比，有着最佳的预测性能。

综上所述，现有的过采样方法通常在两个样本之间线性生成样本，未充分利用周围样本的特征信息，导致生成样本方式相对单一。基于以上分析，本文提出了一种邻域合成的过采样方法，该方法在合成样本时同时参考周围多个近邻的样本信息，合成的样本更具有多样性。

3. 基于邻域合成的过采样方法

该方法主要分为样本预处理和样本合成两个阶段。

3.1. 样本预处理

首先对样本进行预处理，将原始样本按照标签分为多数类样本和少数类样本。SMOTE 在合成新样本时的一个缺陷就是没有区分地对所有的样本进行过采样，使得新生成的样本容易落在多数类样本周围，从而影响分类结果。所以对少数类样本进行分类，按照该样本周围少数类样本的数量将其分为三类，若样本周围没有其他的少数类样本，则将其划为噪声类样本；样本周围少数类近邻数量占总近邻数量的一半以上的该点划为安全类样本；其余样本划为边界类。样本预处理伪代码如算法 1 所示。对于安全类样本，在其周围生成样本对分类的效果影响不大，所以本文将在边界类样本周围合成新样本。样本点类型划分如图 2 所示。

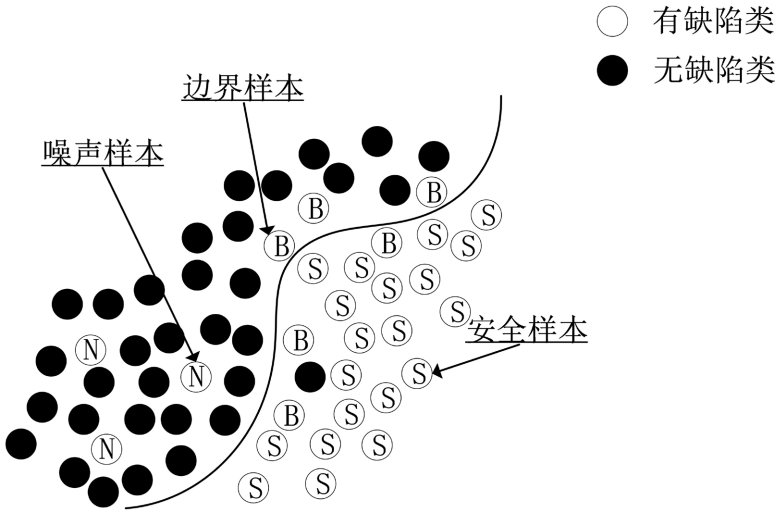

Figure 2. Sample point division
图 2. 样本点划分

3.2. 样本合成

对于每一个边界点 B，找到其 k 个最近邻点。然后将这些最近邻点分成两类：Bnmaj (属于多数类)和 Bnmin (属于少数类)。这一步是为了划分该边界点的多数类和少数类近邻，用于后续的合成样本生成。计算 Bnmaj 和 Bnmin 中样本的数量差异，即 G = len (Bnmaj) – len (Bnmin)。这个 G 值代表多数类和少数类样本的数量差。G 越大，则该边界点少数类的邻居越少，需要生成更多的合成样本。

对于每个特征，计算在该边界点 B 及其邻居中，该特征的最大值和最小值，并将这些值保存到 T_Max_attr [attr]和 T_Min_attr [attr]列表中。接下来，要找到每个特征的中间范围。将这个中间范围保存到 Max_attr [attr]和 Min_attr [attr]列表中，其中 Max_attr [attr]保存该特征的最大值，Min_attr [attr]保存了该特征的最小值。Max_attr [attr]，Min_attr [attr]这个范围即为可以合成的特征值的范围。最后，在 Max_attr [attr]和 Min_attr [attr]范围内随机生成特征值。将生成的合成样本添加到列表 O 中，将 O 与原始数据集 I 合并，得到过采样后的数据集 A。伪代码如算法 2 所示。

算法 1：样本预处理阶段

输入：Imbalanced dataset I, Number of nearest neighbors K

输出：Safe sample S; Boundary sample B; Noise sample N

1. Split dataset I into majority class samples Y and minority class samples T
2. for T_i ∈ T
3. Find the K nearest neighbors of T_i in data set I
4. Calculate the number D_i of minority class samples in the neighborhood of T_i
5. if D_i = 0
6. Add D_i to *list* S
7. elif K/2 <= D_i <= k
8. Add D_i to *list* B
9. else
10. Add D_i to *list* N
11. End

算法 2：样本合成阶段

输入：Imbalanced dataset I, Number of nearest neighbors K

输出：the oversampled dataset after processing with NSOS

1. Split dataset I into majority class samples Y and minority class samples T
2. While Y > T
3. for B_i ∈ B
4. Find the K nearest neighbors Ki of B_i in dataset I
5. For K_i ∈ K
6. If K_i ∈ Y
7. Add K_i to $list\ B_{imaj}$
8. Else
9. Add K_i to list B_{imin}
10. G = len(B_{imaj}) − len(B_{imin})
11. Compute maximum value of each attribute in B [1:Bnum] [attr]
12. Compute minimum value of each attribute in B [1:Bnum] [attr]
13. Find the intermediate value between the maximum and minimum values and save it into Max_attr [attr] and Min_atr [attr]
14. For len (2 * G)
15. For i ∈ attr:
16. O_i [attr] = random [Max_attr [attr], Min_atr[attr]]
17. Add O_i to $list$ O
18. A = union (I, O)

4. 实验及结果分析

4.1. 数据集

实验使用 AEEEM 数据集和 NASA 数据集来衡量该过采样算法的性能，其中 AEEEM 数据集有 5 个数据集，每个数据集有 61 个特征，这些特征从多方面考虑了软件的开发过程。NASA 数据集是由美国国家航空航天局(NASA)提供的一组多样化的数据，用于软件工程领域的研究和评估。这些数据集涵盖了多个软件项目和应用领域，可用于研究软件缺陷预测。本文研究的是软件缺陷预测问题中软件是否有缺陷的二分类问题。所以对上述两个数据集中的类型标签设置为有缺陷和无缺陷两种类型。AEEEM 和 NASA 数据集信息如表 1、表 2 所示。

Table 1. Information of AEEEM datasets
表 1. AEEEM 数据集信息

项目	特征数	样本数	有缺陷数	无缺陷数	IR
EQ	61	324	195	129	1.51
JDT	61	997	791	206	3.84
LC	61	691	627	64	9.80
ML	61	1862	1617	245	6.60
PDE	61	1497	1288	209	6.16

Table 2. Information of NASA datasets
表 2. NASA 数据集信息

项目	特征数	样本数	有缺陷数	无缺陷数	IR
CM1	21	498	449	49	9.16
JM1	21	10885	8779	2106	4.17
KC1	21	2109	1783	326	5.47
KC2	21	520	415	105	3.95
PC1	21	1109	1032	77	13.4

不平衡率(Imbalance Rate, IR)：表示缺陷类样本数与无缺陷类样本数的比值。

4.2. 评价指标

二分类问题中，预测结果会出现四种，如表 3 所示，其中 clean 代表无缺陷，buggy 代表有缺陷，混淆矩阵如表 3 所示。

Table 3. Confusion matrix
表 3. 混淆矩阵

预测值 / 真实值	有缺陷	无缺陷
有缺陷	Ture Positive$(\mathrm{TP})_{Clean \gg Clean}$	False Negative$(\mathrm{FN})_{Buggy \gg Clean}$
无缺陷	False Positive$(\mathrm{TP})_{Clean \gg Buggy}$	Ture Negative$(\mathrm{TN})_{Buggy \gg Buggy}$

其中 TP 表示该样本有缺陷且被模型预测为有缺陷；TN 表示该样本无缺陷且被模型预测为无缺陷；FN 表示该样本有缺陷但被模型预测为无缺陷；FP 表示该样本无缺陷但被模型预测为有缺陷。混淆矩阵的中的四种结果的组合可以形成多种性能指标，如精确度，查准率，召回率，F-measure，G-mean，AUC 度量，MCC 等，计算公式如下：

精确度(accuracy)表示正确分类的样本数占总样本数的比例。

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{1}$$

查准率(Precision)衡量的是模型在所有预测为有缺陷的样本中，真实为有缺陷的样本所占的比例。

$$\mathrm{Pr}ecision = \frac{TP}{TP + FN} \tag{2}$$

召回率(Recall)衡量的是模型在所有真实有缺陷的样本中，成功预测为有缺陷的样本所占的比例。

$$\mathrm{Re}call = \frac{TP}{TP + FN} \tag{3}$$

F-measure 又叫是一个综合性能指标，它结合了查准率(Precision)和召回率(Recall)两个重要的分类模型评估指标。

$$F - measure = 2 \times \frac{\mathrm{Pr}ecision \times \mathrm{Re}call}{\mathrm{Pr}ecision + \mathrm{Re}call} \tag{4}$$

G-mean 是一种用于评估分类模型性能的指标，衡量模型在正负类样本中的平衡性能。是查准率(Precision)和召回率(Recall)的几何平均值。

$$G - mean = \sqrt{\mathrm{Pr}ecision \times \mathrm{Re}call} \tag{5}$$

AUC 指标用于评估分类模型性能的一项重要指标。它衡量的是模型在不同阈值下的召回率(True Positive Rate)与假正例率(False Positive Rate)之间的权衡。通常情况下，AUC 的值越高，说明模型的预测性能越好。

在软件缺陷预测中，数据往往都是不平衡的，因此使用 accuracy 作为衡量指标显然是不合理的，Precision 和 Recall 生成的值互相矛盾，因此也不适合作为衡量指标，AUC 虽然稳定，但是其也不适用于检测软件缺陷预测中的类不平衡问题。因此，选取一个适当的评估指标以评估模型性能是很重要的。综合考虑，本文将采用 F-measure 作为模型性能的衡量指标。F-measure 综合考虑了查准率和查全率两个关键性能指标，提供了全面的性能评估，并且在权衡查准率和查全率方面表现平衡，有助于评估模型在错误识别和错误漏报之间的平衡情况，适用于处理类别不平衡的数据集。

4.3. 实验设计

本实验将用基于邻域合成的过采样方法与多个处理类不平衡的采样算法进行比较，其中包括原始数据(OD)、随机过采样(ROS)、SMOTE 过采样(SMO)、Borderline Smote 过采样(BLS)、ADASYN 过采样(ADA)、Safe Level Smote 过采样(SLS)、K-means SMOTE 过采样(KMS)、KNNOR 过采样(KNN)，使用朴素贝叶斯分类器进行对比实验，实验将进行 10 次十折分层交叉验证，实验结果精确到小数点后两位取平均值作为实验结果，以 F-measure 作为评价指标衡量算法的性能。

4.4. 结果分析

表 4 和表 5 分别为 9 种方法对 AEEEM 和 NASA 数据集 F1-measure 指标的实验结果，为了更直观地观察实验结果，根据表 4 和表 5 分别得出图 3 和图 4 柱状图。

Table 4. F1 values of NSOS and other sampling methods in the AEEEM dataset
表 4. NSOS 与其他采样方法在 AEEEM 数据集中的 F1 值

项目	OD	ROS	RUS	SMO	BLS	SLS	KMS	KNN	NSOS
EQ	0.5848	0.5837	0.5872	0.5862	0.5951	0.5788	0.5337	0.5967	0.6027
JDT	0.5483	0.5472	0.5469	0.5465	0.5608	0.5460	0.5142	0.5691	0.5674
LC	0.3747	0.3710	0.3739	0.3812	0.3763	0.3732	0.3766	0.3639	0.3887
ML	0.3638	0.3691	0.3573	0.3697	0.3732	0.3680	0.3045	0.2274	0.3848
PDE	0.3747	0.3753	0.3855	0.3956	0.3833	0.3896	0.3260	0.3861	0.4054
平均	0.4493	0.4493	0.4502	0.4558	0.4578	0.4511	0.4110	0.4286	0.4698

Table 5. F1 values of NSOS and other sampling methods in the NASA dataset
表 5. NSOS 与其他采样方法在 NASA 数据集中的 F1 值

项目	OD	ROS	RUS	SMO	BLS	SLS	KMS	KNN	NSOS
CM1	0.2700	0.2776	0.2737	0.2817	0.2790	0.2755	0.2700	0.1397	0.2823
JM1	0.2289	0.2282	0.2302	0.2328	0.2818	0.2331	0.0871	0.2654	0.2991
KC1	0.3870	0.3999	0.3992	0.4063	0.4185	0.4030	0.3236	0.4329	0.4174
KC2	0.4870	0.4824	0.4922	0.4920	0.5489	0.4939	0.4033	0.5472	0.5524
PC1	0.2811	0.2730	0.2625	0.2489	0.2644	0.2542	0.2642	0.0966	0.2581
平均	0.3308	0.3322	0.3315	0.3323	0.3585	0.3319	0.2696	0.2964	0.3619

通过实验结果可以看出在 AEEEM 和 NASA 两个数据集上 NSOS 的表现都很出色。在 AEEEM 数据集上，该方法取得了 4 次最优，1 次第 2 的成绩，尤其在 EQ 和 ML 和 PDE 数据集上，该方法的实验效果提升明显。在平均结果上看，该算法的 F1 值为 0.4698，明显好于原始数据与其他 7 种采样方法。在

NASA 数据集中来看，该方法取得了 3 次的最优成绩，尤其在样本数最大的 JM1 数据集上，该方法的性能相对于其他方法有着显著的提升，并且从平均值上看，该方法也同样优于其他方法，可以看出该方法在处理不同领域的软件项目时具有良好的泛化能力。从综合来看，NSOS 方法在多个数据集上都表现出良好的性能，相对于其他方法，它在 F1 值上具有明显的优势，尤其在处理复杂和大规模数据集时表现出色。这表明 NSOS 在不平衡数据的软件缺陷预测中是有效的。

5. 总结与展望

为了改善软件缺陷预测中有缺陷类和无缺陷类的不平衡问题，本文提出了一种邻域合成的过采样方法，通过对少数类样本进行分类，并重点在边界类样本周围合成新样本。合成过程考虑了多数类和少数类的邻居关系以及特征的范围来生成具有多样性的合成样本。经过实验验证，该方法在 AEEEM 和 NASA 数据集上都取得了较为理想的效果。未来可以进一步探索该算法的性能和稳健性，并在不同领域和任务中进行广泛应用。可以考虑优化参数选择，例如邻居数量 k 和合成样本数量的设定，以适应不同数据集的特性。此外，可以与其他数据处理技术结合，如特征选择，噪声处理，或者集成学习来进一步提高预测效果。

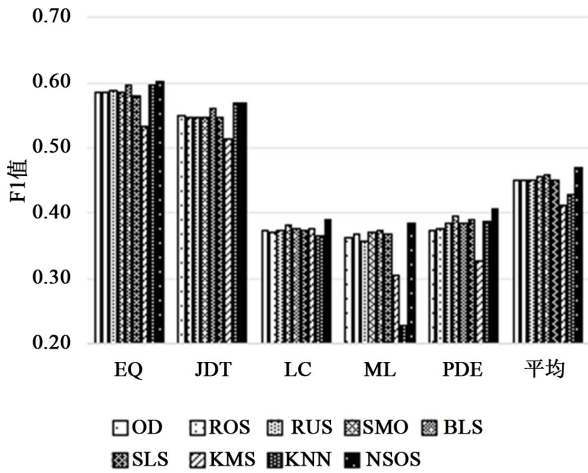

Figure 3. F1 values of NSOS and other sampling methods in the AEEEM dataset
图 3. NSOS 与其他采样方法在 AEEEM 数据集中的 F1 值

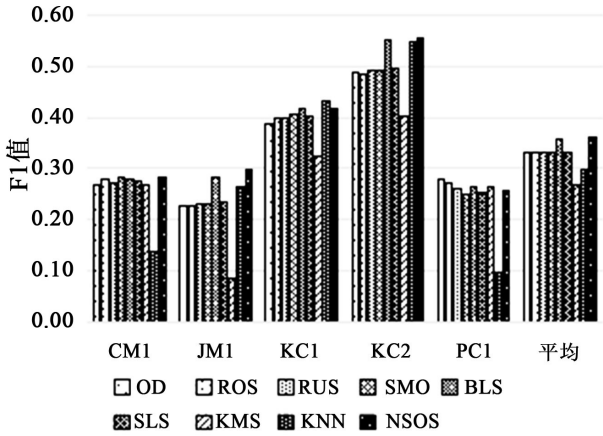

Figure 4. F1 values of NSOS and other sampling methods in the NASA dataset
图 4. NSOS 与其他采样方法在 AEEEM 数据集中的 F1 值

参考文献

[1] La Toza, T.D., Venolia, G. and De Line, R. (2006) Maintaining Mental Models: A Study of Developer Work Habits. *Proceedings of the 28th International Conference on Software Engineering*, Shanghai, 20-28 May 2006, 492-501. https://doi.org/10.1145/1134285.1134355

[2] 宫丽娜, 姜淑娟, 姜丽. 软件缺陷预测技术研究进展[J]. 软件学报, 2019, 30(10): 3090-3114. https://doi.org/10.13328/j.cnki.jos.005790

[3] Soltanzadeh, P. and Hashemzadeh, M. (2021) RCSMOTE: Range-Controlled Synthetic Minority Over-Sampling Technique for Handling the Class Imbalance Problem. *Information Sciences*, **542**, 92-111. https://doi.org/10.1016/j.ins.2020.07.014

[4] Khleel, N.A.A. and Nehéz, K. (2023) A Novel Approach for Software Defect Prediction Using CNN and GRU Based on SMOTE Tomek Method. https://doi.org/10.21203/rs.3.rs-2305042/v1

[5] Yap, B.W., Rani, K.A., Rahman, H.A.A., *et al*. (2014) An Application of Oversampling, Undersampling, Bagging and Boosting in Handling Imbalanced Datasets. In: Herawan, T., Deris, M., Abawajy, J., Eds., *Proceedings of the First International Conference on Advanced Data and Information Engineering* (*DaEng*-2013), Springer, Singapore, 13-22. https://doi.org/10.1007/978-981-4585-18-7_2

[6] Arafat, M.Y., Hoque, S. and Farid, D.M. (2017) Cluster-Based Under-Sampling with Random Forest for Multi-Class Imbalanced Classification. 2017 11th International Conference on Software, Knowledge, *Information Management and Applications* (*SKIMA*), Malabe, 6-8 December 2017, 1-6. https://doi.org/10.1109/SKIMA.2017.8294105

[7] Song, A. and Xu, Q. (2018) Imbalanced Data Classification Based on MBCDK-Means Undersampling and GA-ANN. *Artificial Neural Networks and Machine Learning-ICANN* 2018, Rhodes, October 4-7 2018, 349-358. https://doi.org/10.1007/978-3-030-01421-6_34

[8] Tsai, C.F., Lin, W.C., Hu, Y.H., *et al*. (2019) Under-Sampling Class Imbalanced Datasets by Combining Clustering Analysis and Instance Selection. *Information Sciences*, **477**, 47-54. https://doi.org/10.1016/j.ins.2018.10.029

[9] 周传华, 朱俊杰, 徐文倩, 等. 基于聚类欠采样的集成分类算法[J]. 计算机与现代化, 2021(11): 72-76.

[10] Goyal, S. (2022) Handling Class-Imbalance with KNN (Neighbourhood) Under-Sampling for Software Defect Prediction. *Artificial Intelligence Review*, **55**, 2023-2064. https://doi.org/10.1007/s10462-021-10044-w

[11] Chawla, N.V., Bowyer, K.W., Hall, L.O., *et al*. (2002) SMOTE: Synthetic Minority Over-Sampling Technique. *Journal of Artificial Intelligence Research*, **16**, 321-357. https://doi.org/10.1613/jair.953

[12] Han, H., Wang, W.Y., Mao, B.H. (2005) Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. In: Huang, D.S., Zhang, X.P., Huang, G.B., Eds., *International Conference on Intelligent Computing*, Springer, Berlin, 878-887. https://doi.org/10.1007/11538059_91

[13] He, H., Bai, Y., Garcia, E.A., *et al*. (2008) ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning. 2008 *IEEE International Joint Conference on Neural Networks* (*IEEE World Congress on Computational Intelligence*), Hong Kong, 1-8 June 2008, 1322-1328.

[14] Islam, A., Belhaouari, S.B., Rehman, A.U., *et al*. (2022) KNNOR: An Oversampling Technique for Imbalanced Datasets. *Applied Soft Computing*, **115**, Article ID: 108288. https://doi.org/10.1016/j.asoc.2021.108288

[15] Douzas, G., Bacao, F. and Last, F. (2018) Improving Imbalanced Learning through a Heuristic Oversampling Method Based on K-Means and SMOTE. *Information Sciences*, **465**, 1-20. https://doi.org/10.1016/j.ins.2018.06.056

[16] 饶珍丹, 李英梅, 董昊, 等. 多层次过采样集成的不平衡数据缺陷预测模型[J]. 小型微型计算机系统, 2023, 44(4): 888-896. https://doi.org/10.20009/j.cnki.21-1106/TP.2021-0634

[17] Soltanzadeh, P. and Hashemzadeh, M. (2021) RCSMOTE: Range-Controlled Synthetic Minority Over-Sampling Technique for Handling the Class Imbalance Problem. *Information Sciences*, **542**, 92-111. https://doi.org/10.1016/j.ins.2020.07.014

[18] Koziarski, M., Krawczyk, B. and Woźniak, M. (2019) Radial-Based Oversampling for Noisy Imbalanced Data Classification. *Neurocomputing*, **343**, 19-33. https://doi.org/10.1016/j.neucom.2018.04.089

[19] Maldonado, S., Vairetti, C., Fernandez, A., *et al*. (2022) FW-SMOTE: A Feature-Weighted Oversampling Approach for Imbalanced Classification. *Pattern Recognition*, **124**, Article ID: 108511. https://doi.org/10.1016/j.patcog.2021.108511