软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, [doi: 10.13328/j.cnki.jos.000000]

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

一种基于半监督数据分组的软件系统可维护性预测方法*

朱佳俊[1], 王 炜[1,2], 李 彤[1,2], 何 云[1], 向 威[1]

[1](云南大学 软件学院, 云南 昆明 650091)
[2](云南省软件工程重点实验室，云南 昆明 650091)
通讯作者: 王炜, E-mail: wangwei@ynu.edu.cn

摘 要: 软件可维护性预测针对软件演化活动的容易程度建立预测模型，并在此基础上实现可维护性评价的研究。当前可维护性预测研究将带有标记的源代码、开发日志、维护日志等数据作为训练数据，并基于此构建可维护性预测模型。由于训练数据标记过程需要开销很大，导致现有方法无法适用于实际演化活动。针对上述问题，本文提出了一种基于半监督数据分组技术的软件可维护性预测方法 SGMDH。通过与基线方法和监督方法进行比对实验，结果表明，该方法的预测准确率分别比基线方法高 42%(UIMS 数据集)和 45%(QUES 数据集)。同时，本文半监督方法在两个数据集上均比监督方法表现出较高的准确率，说明该方法具有较强的普适性。

关键词: 面向对象软件;软件可维护性;可维护性预测;软件度量;半监督;SGMDH
中图法分类号: TP311.5

中文引用格式: 朱佳俊,王炜,李彤,何云,向威. 一种基于半监督数据分组的软件系统可维护性预测方法.软件学报. http://www.jos.org.cn/1000-9825/ 0000.htm

英文引用格式: Zhu JJ, Wang W, Li T, He Y, Xiang W. An approach of Semi-supervised Group Method of Data Handling algorithm for software maintainability prediction. Ruan Jian Xue Bao/Journal of Software, 2017 (in Chinese). http://www.jos.org.cn/1000-9825/0000.htm

A semi-supervised and group method of data handling based approach for software maintainability prediction

ZHU Jia-Jun[1], WANG Wei[1, 2], LI Tong [1, 2], HE Yun[1], XIANG Wei[1]

[1](School of Software, Yunnan University, Kunming 650091, China)
[2](Key Laboratory for Software Engineering, Yunnan Province, Kunming 650091, China)

Abstract: Software maintainability prediction is used to construct the prediction model for ascertaining the difficulty of software evolution. Based on it, a software maintenance evaluation is carried out. In the current maintainability prediction study, the tagged source code, development log, maintenance log, and other data are used as training data and a maintainability prediction model is built upon them. Since the training data marking process requires significant overhead, results from the existing method cannot be applied to the actual evolution activities. Aimed at solving this problem, we propose a semi-supervised software maintainability prediction method (SGMDH) in this paper. The results show that, compared to the baseline method, the prediction accuracy of the proposed method is 42% higher using the user interface management system (UIMS) dataset and 45% higher using the quality evaluation system (QUES) dataset. Meanwhile, the semi-supervised method shows a higher accuracy rate than the supervised method on two datasets, which indicates that the method has strong generalizability.

Key words: Object-Oriented; Software Maintainability; Maintainability Prediction; Software Metrics; Semi-Supervised; SGMDH

软件可维护性（Software Maintainability）是一个软件质量属性，它体现了对现有软件进行改正、适应

* 基金项目: 国家自然科学基金(61462092, 61379032); 云南省自然科学基金重点项目(2015FA014).
Foundation item: National Natural Science Foundation of China (61462092, 61379032); key project of Natural Science Foundation of Yunnan Province under Grant(2015FA014)
收稿时间: 0000-00-00; 修改时间: 0000-00-00; 采用时间: 0000-00-00; jos 在线出版时间: 0000-00-00
CNKI 在线出版时间: 0000-00-00

或增强的容易程度[1]。Firefox, Linux 等高可信软件的诞生，证明了软件演化是获取可信软件的一条重要途径。文献[2-3]对长生命周期软件进行统计，发现 55%~75%的开销用于软件演化[2, 3]。软件的可维护性预测是决定是否执行演化活动的先决条件，因此建立准确、经济的可维护性预测方法对软件演化研究具有重要的意义。当前可维护性分析研究大多遵循相似的步骤：首先，将源代码、开发日志、维护日志等数据抽象成为可维护性度量因子作为输入；其次，基于输入数据，构建可维护性预测模型进行演化活动成本估计。

按软件开发方法的不同，软件可维护性预测可大致分为两类：结构化软件可维护性预测和面向对象软件可维护性预测。由于面向对象软件是当前软件开发的主要泛型，因此面向对象软件的可维护性预测问题是当前研究的热点。面向对象软件可维护性预测均涉及两个核心问题：度量因子质量的好坏和预测模型。

1991 年 Chidambe 和 Kemerer[4]首次提出了可维护性度量因子，简称 C&K 度量。C&K 度量给出了 6 个级别的度量指标：（1）每个类的加权方法数；（2）继承树的深度；（3）类的子类数目；（4）对象类之间的耦合度；（5）一个类的应用集合；（6）类内聚程度。该方法反映了面向对象技术的特点，研究人员可对指标体系进行裁剪，以适合软件的特点。1993 年 Li 和 Henry[5]在 C&K 度量的基础上对可维护性度量因子进行完善，提出了 L&H 度量。L&H 度量从 8 个指标对可维护性进行了度量：（1）操作复杂性度量；（2）操作参数复杂性度量；（3）震性复杂性度量；（4）操作耦合度量；（5）类的继承性度量；（6）内聚度量；（7）类耦合度量；（8）重用度量。L&H 度量指标考虑了面向对象技术的特点，但没有给出形式化的定义，导致缺乏可操作性。表 1 给出了常用度量因子的解释。

Table 1 Software Metrics

表 1 度量因子

度量因子	描述
DIT	（Depth in the Inheritance Tree）表示类的继承层次。
NOC	（Number Of Children）表示一个类的直接子类数。
RFC	（Response For Class）度量类的响应基数。
LCOM	（Lack of Cohesion Of Methods）度量低内聚的方法。
WMC	（Weighted Method Complexity）度量所有方法的静态复杂度。
MPC	（Message-Passing Coupling）度量类间消息传递的复杂度。
DAC	（Data Abstraction Coupling）度量由抽象数据类型（ADT）引起的耦合复杂度。
NOM	（Number Of Methods）度量一个类中所定义方法的数量。
SIZE2	表示一个类中的属性和方法数。
SIZE1	表示代码行数，用分号来计算。
CHANGE	表示在维护期间代码的改变（增加或删除）行数。

基于软件度量因子，构建软件可维护性预测模型是可维护性分析面临的另外一个问题。当前，相关研究的预测模型主要分为两类：线性模型和非线性模型；线性模型主要采用线性回归方法来构建[6, 7]；而非线性模型以机器学习方法为代表，包括朴素贝叶斯[8]、支持向量机[9]、神经网络[10, 11]、遗传算法[12]等。由于现实中软件预测因子和可维护性间大多存在非线性关系，因而非线性方法在该领域更受关注。表 2 显示了近十年各非线性方法的预测模型结果，表中平均误差率（MMRE）值越小、pred(q)值越大说明模型的预测准确率越高，性能越好。当 MMRE≤0.25[13]和 pred(0.25) ≥0.75[13]或 pred(0.30) ≥0.70[14]时模型是精确的[16,17]。由表 2 可知基于非线性模型的可维护性预测研究在预测精度方面还有很大的提升空间。

Table 2 The results of the study in recent years

表 2 近几年研究结果统计

序号	模型	MMRE	pred(0.25)	pred(0.30)	年份
1	异质集成[15]	0.410	-	0.845	2015
2	GMDH[16]	0.357	0.610	0.710	2014
3	GRNN[16]	0.548	0.440	0.470	2014
4	FF3LBPNN[16]	0.458	0.510	0.590	2014
5	GMDH[12]	0.210	0.690	0.722	2012
6	GA[12]	0.220	0.660	0.722	2012
7	PNN[12]	0.230	0.680	0.750	2012
8	SVM[17]	0.218	-	-	2010
9	MARS[9]	0.320	0.480	0.590	2007
10	MLP[9]	0.420	0.370	0.410	2007
11	贝叶斯[8]	0.452	0.391	0.430	2006
12	回归树[8]	0.493	0.352	0.383	2006
13	向后消除[8]	0.403	0.396	0.461	2006
14	逐步选择[8]	0.392	0.422	0.500	2006
15	GRNN[11]	0.765	-	-	2005

16	ANN[18]	0.265	-	-	2005

综上所述，面向对象软件可维护性分析有两方面的困难：

- 经过 20 多年的发展分析精度始终不高，文献[12]表明对于 User Interface System、Quality Evaluation System 等中等规模的软件系统分析精度仅为 0.210[12]；
- 当前分析模型大多基于监督学习模型，需要大量带有可维护性标签的训练数据。上述数据的获取需要大量开销，限制了研究成果在实际演化环境的应用。

针对上述问题，本文将半监督回归与数据分组方法相结合，提出了一种用于面向对象软件的可维护性分析方法 SGMDH(Semi-supervised and Group Method of Data Handling based approach)。该方法采用聚类抽样法缓解数据分布不平衡问题，同时基于半监督回归思想建立可维护性预测模型，在保证分析精确度的情况下，降低了带标签可维护性数据的使用量，提高了该方法的普适性。

本文篇章结构如下：第 1 节阐述了本文的相关工作；第 2 节介绍了本文采用方法的背景知识；第 3 节对本文提出的 SGMDH 方法构建可维护性预测模型进行阐述；4 节对本文的实验环节进行了说明，包括实验数据集、度量和评价准则、基线方法和实验过程设计，并对实验结果的有效性进行了分析；第 5 节分析了本文方法的局限性和不足；第 6 节对本文的研究工作进行总结。

1 相关工作

软件可维护性预测研究经历了 20 多年的发展，从 1994 年 Coleman[20]等人首次利用 C&K 和 L&H 软件度量对软件可维护性进行预测开始，发现软件度量和可维护性之间有关联，同时证明了 C&K 度量和 L&H 度量可以用来预测软件可维护性。此后研究人员常采用 C&K 度量和 L&H 度量来预测软件可维护性。

2005 年 Thwin 和 Quah[10]基于神经网络方法构建软件可维护性模型，在 HMI 系统和 QUES 系统上进行实验，发现 GRNN 比 ward 神经网络能获得较好的预测性能。2006 年 Koten 和 Gray[8]采用了四种回归方法构建模型，在 UIMS 和 QUES 系统上进行实验，发现基于朴素贝叶斯构建的预测模型比回归树方法、向后消除方法和逐步选择方法能获得更准确的预测结果。2007 年 Zhou 和 Leung[9]采用了多元自适应回归（MARS）方法构建预测模型，并在 UIMS 和 QUES 系统上进行实验，发现与多元线性回归(MLR)、人工神经网络(ANN)、回归树(RT)和支持向量机(SVR)模型相比，MARS 模型能获得更好的预测性能。2009 年 Elish 和 Elish[21]基于 TreeNet 建立预测模型，在 UIMS 和 QUES 系统上与 MARS、MLR、SVR、ANN 和 RT 进行对比实验，发现 TreeNet 较其他四种模型表现出更好的预测能力。2010 年 Jin 和 Liu[17]采用 SVM 和基于模糊集的聚类方法建立预测模型，发现 SVM 和聚类方法能够用来预测软件可维护性。2012 年 Malhotra 和 Chug[12]采用数据处理方法(GMDH)、遗传算法(GA)和概率神经网络(PNN)构建预测模型，发现这三种方法构建的模型预测准确率高于上述其他模型，而且三者中 GMDH 模型取得较好的预测性能。在此基础上，2014 年 Malhotra 和 Chug[16]利用 GMDH、GRNN 和前馈三层反向传播(FF3LBPNN)算法构建预测模型，并在基于 web 的两个应用系统 FLMS 和 EASY 上进行实验，发现 GMDH 比另两种算法能取得更准确的预测结果。2015 年 Elish[15]等人采用了三种异质(heterogeneous)集成的方法（简单平均集成、加权平均集成和最佳训练集成）以多层感知机(MLP)、支持向量机(SVR)、径向基网络(RBF)和 M5 模型树作为个体学器来构建集成预测模型，发现除了简单平均集成外另两种集成模型的性能都比基于 MLP、SVR、RBF 和 M5 建立的单模型好，而且在三种异质集成方法中最佳训练集成获得更好性能。上述所有方法都采用了 C&K 度量和 L&H 度量建立模型，进一步证明了 C&K 度量和 L&H 度量与软件可维护性之间有强关联关系，并能够较好地预测软件可维护性[6-12, 17, 18, 22-24]。

当前研究人员采用了多种方法建立软件可维护性预测模型，但是预测准确率仍然不高（详见表 2），而且利用上述方法构建预测模型需要大量带可维护性标签的软件数据，而在实际环境中获取上述数据十分困难，而且开销巨大，同时也限制了研究成果在实际环境中的应用。目前半监督学习已应用到软件工程领域的缺陷预测[25, 26]和工作量估计[27]中，也被成功应用于人脸识别[28]、临床评估[29, 30]、图像处理[31, 32]、位置查找[33]等实际问题中。

总结相关研究工作，目前半监督方法还未被应用于软件可维护性预测，受到上述方法的启发，本文提

出的 SGMDH 方法基于半监督学习思想，该方法不仅考虑了数据分布不平衡问题，而且借助部分带标记数据样本结合标签传播算法和 GMDH 算法构建预测模型，从而提高预测模型的泛化能力和预测性能，并通过实证研究验证了本文 SGMDH 方法的有效性。

2 背景知识

本节主要介绍了本文采用主要算法的基本理论知识，包括数据分组处理方法和半监督标签传播算法两部分内容。

2.1 分组处理方法

数据分组处理方法（Group Method of Data Handling，简称 GMDH）是 Ivakhnenko[34]在 1968 年提出的一种用于复杂系统建模的方法。其基本思想是：借助自组织原理，通过启发式学习自动发现数据间的相互关系，实现输入和输出之间的非线性映射，并用外准则选择一个最优的模型或网络，实现对问题域的建模。GMDH 方法能够充分合理地利用样本数据，因此在变量多、数据少的复杂问题中能获得较满意的结果[35, 36]，具有较好的预测性能。

具体来说 GMDH 方法假设系统有 m 个输入$x_1, x_2, \dots, x_m$，输出为$\tilde{y}$。GMDH 的目的是建立输入和输出之间的模型（一般是非线性的）。

$$\tilde{y} = f(x_1, x_2, \dots, x_m) \tag{1}$$

由于f的形式不知道，因此可以用输入变量的多项式去逼近f，即 Kopuoronob-Gobor 多项式：

$$y = \sum_{i=1}^{N} a_i x_i + \sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij} x_i x_j + \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N} a_{ijk} x_i x_j x_k + \cdots \tag{2}$$

其中x_i可以是不同的输入变量，也可以是统一输入变量。显然随着变量个数和方程次数的增加，K-G 多项式的项数也在急剧增加。例如，输入变量为 10，多项式为 3 次，则 K-G 多项式有 285 项之多，若次数增大到 10，则从项数将达 10 万个。为了解决 K-G 多项式的计算问题，引入了“部分实现”的概念。所谓“部分实现”是由任意两个输入变量构成的函数：

$$y_k = G(x_1, x_p) \tag{3}$$

基于“部分实现”概念，GMDH 方法基本实现步骤如下：

1.变量选择：根据先验知识，考虑输入输出变量间的相互关系，舍弃对输出影响小的变量；

2.数据分组：按照数据分组方法将原始数据分成拟合和检验两组。拟合组数据用来估计参数，检验组数据用来筛选部分实现；

3.变量自组织：将输入变量集合两两组合，得到$\frac{m(m-1)}{2}$对变量；

4.寻求部分实现：用$\frac{m(m-1)}{2}$对变量利用公式(2)和(3)得到对应的“部分实现”。

5.筛选中间变量：用检验组数据对各个“部分实现”按误差进行选择，误差低于给定阈值的部分实现留下，超过阈值的淘汰；

6.重复执行：将留下的“部分实现”作为新的输入变量，得到下一层的部分实现；

7.停机判断：若步骤 6 的误差大于步骤 5 的误差，或部分实现达到预先设定的阶数，或筛选的结果只留下一个部分实现时停止。由于 GMDH 方法自主地对输入变量进行组合和筛选，能够用局部简单的算法建立整体上复杂分类、回归模型，因此，自该方法诞生起就受到了广泛的关注，已经成为当前完成预测任务的重要方法。

2.2 半监督标签传播方法

按照传统的机器学习理论框架，机器学习可以分为有监督学习和无监督学习两类。在有监督学习中，学习器利用的是已标签样例，而无监督学习中只关注未标签样例。随着数据采集技术和存储技术的发展，

获取大量未标签样例已经比较容易，而由于需要耗费一定的人力和物力，获取大量已标签样例则相对比较困难。因而在很多实际数据集中，未标签样例的数量远大于已标签样例的数量。如果只使用少量已标签样例，那么有监督学习训练得到的学习模型不具有很好的泛化能力，同时造成大量未标签样例的浪费；如果只使用大量未标签样例，那么无监督学习将会忽略已标签样例的价值。因此，研究如何综合利用少量已标签样例和大量的未标签样例来提高学习性能的半监督学习(Semi-supervised Learning)成为当前机器学习和模式识别的重要研究领域之一。

半监督学习的基本思想是利用数据分布上的模型假设，建立学习器对未标签数据的标记。形式化的来说，假设给定某个未知分布的数据集合$S = L \cup U$，其中$L = \{(x_1, y_1), (x_2, y_2) \ldots (x_m, y_{|L|})\}$为已标记数据集合，$U = \{x'_1, x'_2, \ldots, x'_{|U|}\}$是一个未标签数据集合。希望得到函数$f: X \to Y$可以精确地对数据 x 预测到标签 y。其中x_i和x'_i均是 d 维数据，$y_i \in Y$为数据x_i的标签，$|L|$和$|U|$分别是 L 和 U 的大小。目前，常用两种方法来建立训练数据和目标预测之间的关系：(1)直接利用标签数据和未标签数据建立预测模型，常采用协同训练回归的方法[37-40]。(2)先对未标签集合 U 进行处理得到对应标签，再利用标签集合 L 和处理后的集合 U 建立预测模型。

标签传播算法（Label Propagation Algorithm，简称 LPA）是一种典型的标签处理方法，由 Zhu[41]等人于 2002 年提出，是一种基于图的半监督学习方法。LPA 算法的基本思想是：数据集的所有节点构成一个带权无向图，从有标签的节点开始向无标签节点传播，每个节点的标签按相似度将信息传递给相邻节点，各节点根据从相邻节点获得的信息更新自己的标签，经过多次迭代最终完成标签传播过程，相似度越高的节点标签越趋于一致，标签传播越容易也越准确。

具体来说，令$X = \{x_1, x_2, \ldots, x_m, x'_1, x'_2, \ldots, x'_{|U|}\}$，$X$ 的大小为$N = |L| + |U|$，则 LPA 希望通过利用变量集 X，对 L 的标签 y 进行学习，得到 U 中每个数据对应的标签y′。LPA 基于图的思想将集合 S 中的所有数据作为节点创建一个完全连接图 G，边的权重表示该条边所连接的两个节点的相似度，边的权重越大两节点的相似度越高。那么节点 i 和节点 j 所构成边的权重为：

$$w_{ij} = exp\left(-\frac{d_{ij}^2}{\sigma^2}\right) = exp(-\frac{\sum_{d=1}^{N}(x_i^d - x_j^d)^2}{\sigma^2}) \tag{4}$$

为衡量一个节点的标签通过边传播到其他节点的概率，需定义一个 $N \times N$ 的概率传递矩阵 T：

$$T_{ij} = P(i \to j) = \frac{w_{ij}}{\sum_{t=1}^{N} w_{ti}} \tag{5}$$

同时，需要定义一个标注矩阵$Y = [Y_L, Y_U]$，Y_L表示有标签集合的标注矩阵，假设标签集合 L 有 C 个标签，则Y_L大小为$|L| \times C$，第 i 行表示第 i 个样本的标签指示向量，即如果第 i 个样本的类别是C_j，那么该行的第 j 个元素为 1，其他为 0。Y_U表示未标签集合的标注矩阵。

基于图 G，标签传播算法的基本步骤如下：

1.初始化：为每个未标签数据设置一个初始标签，初始化未标注矩阵Y_U；

2.执行传播：根据概率传递矩阵 T 和标注矩阵Y为每个节点计算新标签，并更新标注矩阵，$Y' = [Y_L', Y_U'] \longleftarrow TY$；

3.重置标注矩阵：重置有标签矩阵Y_L，$Y' = [Y_L, Y_U']$

4.重复执行：重复步骤 2 和 3，直到Y'收敛。

3 基于半监督数据分组的软件可维护性预测模型——SGMDH

基于半监督数据分组处理的面向对象可维护性预测方法包括四部分：(1)指标度量；(2)训练数据获取；(3)采用标签传播算法得到新的训练集；(4)基于数据分组方法建立可维护性预测模型，并进行可维护性预测。

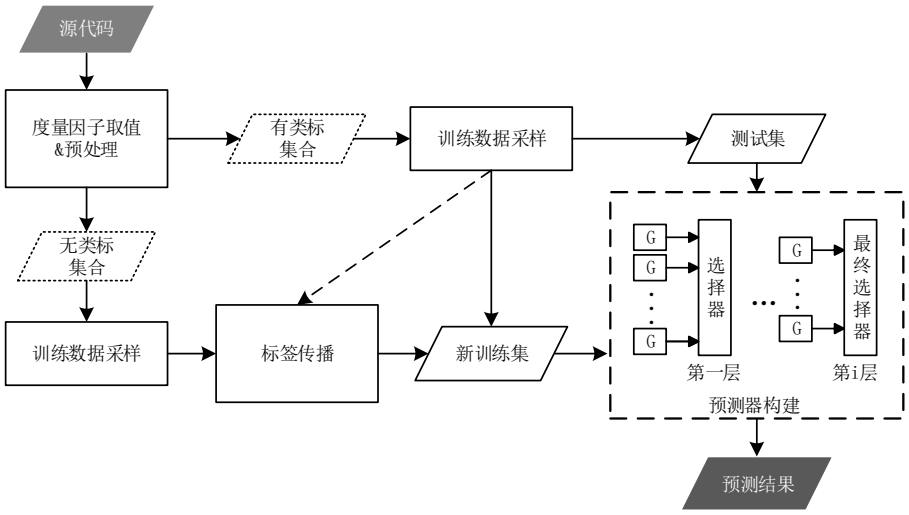


Fig. 1 SGMDH method for maintainability prediction

图 1 SGMDH 可维护性预测方法

3.1 指标度量

由于 C&K 和 L&H 度量具有良好的定义，能够反映面向对象软件系统的固有属性，已经被大量研究采用作为可维护性度量因子。因此，在此我们可采用 C&K 和 L&H 度量指标的一个子集作为可维护性的度量指标。具体各个指标取值如下所示。

定义 1（Depth in the Inheritance Tree, DIT）类的继承层次，$DIT = k, k \in [0, N]$。其中 N 为正整数，DIT=0 表示该节点为根节点。

定义 2（Number Of Children, NOC）类的直接子类数，$NOC = k, k \in [0, N]$。其中 N 为正整数。一个类的直接子类越多，受影响的类越多，越难于演化。

定义 3（Response For Class, RFC）类的响应基数。响应集由类中的局部方法和被该局部方法调用的方法组成。RFC = 局部方法数 + 被局部方法调用的方法数。一个类的响应集越大，越难于演化。

定义 4（Lack of Cohesion Of Methods, LCOM）方法低内聚度。内聚性用局部方法和局部实例变量的相近度来描述。LCOM = 局部方法的互斥集数。内聚性低的方法越多，越难于演化。

定义 5（Weighted Method Complexity, WMC）方法静态复杂程度。由于方法中控制流越多，意味着方法复杂程度越高。因此，WMC = 局部方法的 McCabe 复杂度。一个类的静态复杂度越大，越难于演化。

定义 6（Message-Passing Coupling, MPC）消息传递耦合性。MPC = 消息传递数。消息传递耦合性越强，越难于演化。

定义 7（Data Abstraction Coupling, DAC）数据抽象耦合。一个类可以看作一个抽象数据类型的实现，一个类的抽象数据类型越多，这个类和其他类的耦合性越复杂。DAC = ADT 的数量。数据抽象耦合越高，越难于演化。

定义 8（Number of Methods, NOM）类方法数量。可以表现一个类的操作属性，是接口增量的度量。一个类中的方法越多，这个类的接口就越复杂。NOM = 局部方法数。类方法数量越多，越难于演化。

定义 9（Number of Methods and Attributes, NMA）类方法和属性数。NMA = 属性总数 + 方法总数。类方法和属性数越多，越难于演化。

定义 10（Lines of Code, LC）代码行数。LC =语句结束标识符数量。不同的语言有不同的语句结束标识符，对于 Java 语言语句结束标识符是分号。代码行数越多，越难于演化。

定义 11（Changes）演化期间代码改变行数。CHANGES=代码增加或删除改变行数。

当前很多版本控制工具（例如：SVN 的 diff 工具）可以识别某代码修改前后的异同。通过统计上述修改活动可以获取 CHANGES 指标的取值。代码改变行数越多，越难于演化。

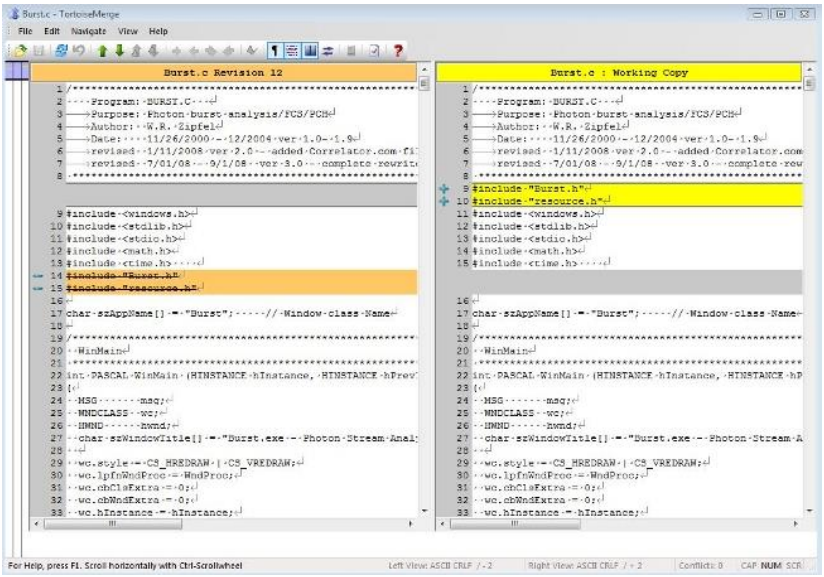


Fig. 2 Reviewing the differences between version of a class by SVN diff tool

图 2 使用 SVN diff 工具识别某代码修改前后异同

由于软件系统的业务领域、编程语言、开发方法及水平的不同，致使软件可维护性度量因子取值值域区间存在较大差异。本文采用 min-max 归一化方法将样本数据映射到[0,1]区间。每个类经过度量后均对应一个 d 维度量因子向量$X=[x_1,x_2,\dots,x_d]$，其中 d 表示度量指标的数目，x_i表示第 i 个度量指标的取值。则经过 min-max 归一化处理后的度量因子X^*为：

$$X^*=\frac{x-\min}{\max-\min} \tag{6}$$

3.2 训练数据获取

经过度量后每一个类对应着一个 d 维度量因子向量X。若一个软件包含 n 个类，则将得到 n 个 d 维的度量因子向量。当对 p 个软件进行度量后将获得 $p\times n$ 个 d 维的度量因子向量。这些向量组成了可维护性分析的训练数据集 S。我们将演化期间代码改变行数作为训练数据的类标。根据定义 11 可知演化期间代码改变行数的获取需要大量的人工介入，开销大，其余度量指标取值相对容易。因此训练数据 S 可分为两个集合 L 和 U，其中 L 为有类标的训练数据，U 为没有类标的训练数据。

在软件可维护性训练数据集中普遍存在数据分布不平衡问题。不平衡的训练数据将无法训练得到能真实反映软件特点的模型。由于演化行为在软件模块中的分布大致符合帕累托原则[42]，即约 80%的软件演化行为集中发生在约 20%的程序模块内[43]。因此，无论对于有类标训练数据集合 L 还是没有类标训练数据集合 U，无需演化的模块对应的度量向量数量远比需要演化模块对应的度量向量数量多。因此我们需要对训练数据集进行分类采样，以解决训练数据不平衡问题。具体来说首先利用 Kmeans 算法分别把训练数据集 L 和 U 聚成 K 簇，再分别从 K 簇中提取相应比例的样本集合构成无类标训练数据集合U'以及有类标训练数据集合L'。

训练集获取算法伪代码描述如下：

输入：可维护性有类标数据集$L=\{(x_{l1}^*,y_{l1}),(x_{l2}^*,y_{l2}),\dots,(x_{lt}^*,y_{lt})\}$，无类标数据集$U=\{x_{u1}^*,x_{u2}^*,\dots,x_{us}^*\}$，质心点数 K，收敛参数$\alpha$，采样比例 p;
过程：

```
For each set L,U do
    Select μ = {μ_1, μ_2, ..., μ_k}
    While μ_1 > α
        For i < n
            C(i) = arg min_j ||x_i^* − μ_j||^2
        End For
        For j< k
```

8 $\mu_j' = \frac{\sum_{i=1}^{n}\{C(i)=j\}x_i^*}{\{C(i)=j\}}$
9 End For
10 End While
11 End For
12 Sampling L', U' from each cluster L_k, U_k
13 For i < k
14 $L(i)' = samplie(pL_i)$
15 $U(i)' = samplie(pU_i)$
16 End For
17 End sampling
输出：有类标训练集$L' = \{(x_{l1}^{*\prime}, y_{l1}'), (x_{l2}^{*\prime}, y_{l2}'), \dots, (x_{la}^{*\prime}, y_{la}')\}$，无类标训练集$U' = \{x_{u1}^*, x_{u2}^*, \dots, x_{ub}^*\}$

算法第 1-11 行分别对数据集的有类标集合 *L* 和无类标集合 *U* 进行聚类，第 2 行初始化聚类中心点，第 4-6 行对每个数据点计算与聚类中心点的距离，选择最小距离的簇类作为所属的类，第 7-9 行对各类簇重新计算质心点，通过第 3-9 行进行迭代直到收敛；第 12-17 行对每个类簇中的数据分别采样，构成有类标训练集L'和无类标训练集U'。

3.3 采用标签传播算法得到新的训练集

根据定义 11 可知，获取演化期间代码改变行数需要大量的人工介入，开销很大。因此有类标的训练数据数量一定较没有类标的训练数据少。为了利用部分无类标训练数据，需要对无类标数据进行标记。

标签传播算法通过传播邻居节点的标签和传递概率来计算自身节点的标签，该算法易于理解、复杂度低且无需先验初始化，当数据量较小时能够高效地对无标签数据进行标记。考虑到可维护性数据样本较小的问题，本文利用标签传播算法获取无类标数据的类标。

算法伪代码描述如下：

输入：有类标训练集$L' = \{(x_{l1}^{*\prime}, y_{l1}'), (x_{l2}^{*\prime}, y_{l2}'), \dots, (x_{la}^{*\prime}, y_{la}')\}$，无类标训练集$U' = \{x_{u1}^*, x_{u2}^*, \dots, x_{ub}^*\}$，无类标初始值 *D*；
过程：
1 N = a + b
2 For each point $(x_i^{*\prime}, y_i')$ do
3 For j < N
4 $w_{ij} = \exp\left(-\frac{d_{ij}^2}{\sigma^2}\right) = \exp(-\frac{\sum_{d=1}^{N}(x_i^d - x_j^d)^2}{\sigma^2})$
5 End For
6 End For
7 For each point $(x_i^{*\prime}, y_i')$ do
8 For j < N
9 $T_{ij} = \frac{w_{ij}}{\sum_{t=1}^{N} w_{ti}}$
10 End For
11 End For
12 $Y_L = f(L_i, C_j)$
13 $Y_U = f(U_i, D)$
14 $Y = [Y_L, Y_U]$
15 For each point $(x_i^{*\prime}, y_i')$ do
16 $y_i'' = \sum_{j=1}^{N} T_{ij} Y_{ij}$
17 $Y' = [Y_L', Y_U'] = f(y_i'')$
18 Set $Y' = [Y_L, Y_U']$
19 End For
20 Return Y_U'
输出：带类标训练集U'

进行标签传播时，我们利用可维护性训练集L'和U'先构造出一个可维护性完全连接图$G =< N, E >$，节点集 *N* 表示训练数据点数，可以从算法第 1 行得到，是无类标数据点和有类标数据点的总和；*E* 表示边集，边上的权重 *w* 表示二者的相似性，可以从第 2-6 行得到所有边的权重，权重越大表示该条边连接的两个节点越相似。第 7-11 行计算出每两个节点间的传递概率，得到所有数据点的概率传递矩阵 *T*。第 12 行定义了一个类标矩阵Y_L，若L'类标训练集含有 *c* 个类标，则Y_L的大小为 *a* ×*c*。第 13 行定义一个无类标矩阵Y_U，并

为每一个无类标数据点的类标设置初始值 D。第 14 行把类标矩阵Y_L和无类标矩阵Y_U合并得到完整的训练集标注矩阵 Y。第 15-19 行根据概率传递矩阵 T 和标注矩阵 Y 计算出所有训练数据点的新类标，第 17 行根据新类标更新标注矩阵Y'，但是原有类标数据的标签不能改变，因此第 18 行将类标矩阵Y_L的类标重置为初始值。通过最后第 20 行返回无类标训练集的标签$Y_U{}'$。

经过标签传播后无类标训练数据集合U'中每个数据点都对应一个确定的类标值$y_n{}'$，和有类标训练数据集合L'构成一个新的可维护性训练集$Tr = \{((x_1^*{}', y_1'), ((x_2^*{}', y_2') \ldots ((x_n^*{}', y_n')\}$。

3.4 基于数据分组方法建立可维护性预测模型

利用可维护性训练集建立一个准确的预测模型，核心在于预测算法的选择。近 20 多年的研究采用了多种数据分析算法建立可维护性预测模型，但预测精度始终不高。数据分组处理方法由于其自组织原理不需要先验知识，能够充分合理地利用样本数据，在变量多、数据少的复杂问题中能获得较满意的结果，因而被成功应用于多领域中，同时也证明了方法的有效性[44-46]。为了在较小的训练数据中获得满意的预测结果，本文采用 GMDH 方法构建可维护性预测模型。

数据分组是 GMDH 方法的关键，因为数据使用是否得当，直接影响所建模型的好坏。预测误差平方和(Predicted Residual Sum of Squares，简称 PRESS) 是一种系统的分组方法，它用一种系统的方法将整个训练集的数据分别分配到拟合组和检验组中。例如，一个含 n 个数据的训练集，PRESS 方法则分配$n-1$个数据到拟合组中，剩余一个数据为检验组数据。当检验组分别取训练集中的每个数据时，训练集中的每个数据都分别作了一次检验数据，而每$n-1$个数据也分别作了一次拟合组。在构建模型网络结构时，用检验组对各“部分实现”进行检验，并以此调整模型的网络结构：

$$PRESS = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \tag{7}$$

采用 GMDH 方法构建可维护性预测模型过程如下：

1.变量选择：根据软件度量指标，将定义 1~10 作为输入变量集，定义 11 作为输出变量。

2.数据分组：根据$PRESS$方法将可维护性训练集Tr分成拟合组Tr_A和检验组Tr_B。

3.变量自组织：将输入变量集合$\{DIT, NOC, RFC, LCOM, WMC, MPC, DAC, NOM, NMA, LC\}$两两组合，得到 45 对变量；

4.寻求部分实现：用 45 对变量利用公式(2)和(3)得到对应的 45 个“部分实现”。

5.筛选中间变量：用检验组数据对各个“部分实现”进行检验，利用公式(5)计算误差并进行选择，误差低于给定阈值的“部分实现”留下，超过阈值的淘汰；

6.重复执行：将留下的“部分实现”作为新的输入变量，得到下一层的“部分实现”；

7.停机判断：若步骤 6 的误差大于步骤 5 的误差，或“部分实现”达到预先设定的阶数（即最大层数），或筛选的结果只留下一个部分实现时停止，得到的网络结构即为最终可维护性预测模型。

最后，我们利用构建的模型预测软件可维护性时，首先按定义 1~10 对软件系统进行度量，并以相同的方式对其特征取值进行预处理得到模型的输入变量，然后按照模型的组织结构将这 10 个输入变量进行特定组合得到对应的“部分实现”，经过多层迭代后输出模型的预测结果，即定义 11 定义的目标变量可维护性 CHANGES 值。

4 实验

本节我们利用了两个数据集对本文方法的有效性进行实证研究，对实验采用的数据集、评价指标、基线方法、实验过程及结果分析进行描述，并对本文方法的有效影响因素进行分析。针对本次实证研究本文设置了两个问题：

RQ1：与基线方法相比，SGMDH 方法是否能表现出较优的预测性能？

RQ2：与监督学习相比，本文半监督方法是否能表现出较优的预测性能？

4.1 数据集

本文采用了由 Li 和 Henry[5]公开的两个面向对象基准数据集：UIMS(User Interface System)和 QUES(Q

uality Evaluation System)。这两个数据集共有 110 个类，其中，UIMS 数据集有 39 个类，QUES 数据集有 71 个类。数据集都由 11 个度量因子构成：$\{DIT, NOC, RFC, LCOM, WMC, MPC, DAC, NOM, NMA, LC, CHANGES\}$，前 10 个度量因子为输入变量，*CHANGES* 为目标变量即软件系统可维性，代表了在软件维护期间所修改的代码行数。

4.2 评价指标

为了验证本文方法结果的有效性，我们采用该领域常用的相对误差率 MRE[8, 9, 12, 15, 17, 18]、pred(q) [8, 9, 12, 15, 17, 18] 对可维护性预测值$\hat{y}$与实际值y进行度量和评价。

(1) 相对误差率 MRE（Magnitude of Relative Error）：表示实际值和预测值之间的归一化程度。

$$\mathrm{MRE} = \frac{|y-\hat{y}|}{y} \tag{8}$$

本文用 MRE 的平均值和最大值来度量可维护性预测结果的准确率，分别表示为 MMRE 和 Max.MRE。MMRE 值越小，说明实际值和预测值之间的平均差异越小，预测准确率越高。Max.MRE 值越小，说明实际值和预测值之间的最大差异越小，预测准确率越高。

(2) pred(q)：度量所有预测值中 MRE 值小于或等于特定值的比例[47]，值在[0,1]区间。

$$\mathrm{Pred(q)} = \frac{K}{N} \tag{9}$$

其中，q 是特定值，K 是预测值中 MRE 值小于或等于 q 的数量，N 是预测数据集的总个数，本文用 pred(0.25)来度量。$\mathrm{Pred(0.25)}$值越大说明预测结果中 MRE 值小于或等于 0.25 的数量越多，$\mathrm{Pred(0.25)}$值越接近 1 说明预测准确率越高。

4.3 实验过程

本文实验采用 Python3.5 实现，本节主要对实验过程中的关键环节：数据预处理、基线方法选择、实验设计和实现方法进行描述。

4.3.1 数据预处理

本文采用静态列表分析法和动态实验分析法（PCA 映射）对可维护性数据集进行分析。表 3 给出了两个数据集的静态列表分析，从表 3 可以看出 QUES 数据集的 NOC 度量值为空，对建立可维护性预测模型无用，因此本文用 NOC 除外的 10 个度量因子建立 QUES 可维护性预测模型，用全 11 个度量因子建立 UIMS 可维护性预测模型。

Table 3 Dataset static analysis

表 3 数据集静态分析

Metric	UIMS				QUES			
	Min	Max	Mean	Std	Min	Max	Mean	Std
DIT	0	4	2.15	0.90	0	4	1.92	0.53
NOC	0	8	0.95	2.01	-	-	-	-
MPC	1	12	4.33	3.41	2	42	17.75	8.33
RFC	2	101	23.21	20.19	17	156	54.44	32.62
LCOM	1	31	7.49	6.11	3	33	9.18	7.31
DAC	0	21	2.41	4.00	0	25	3.44	3.91
WMC	0	69	11.38	15.90	1	83	14.96	17.06
NOM	1	40	11.23	10.32	4	57	13.41	12.00
SIZE2	1	61	13.97	13.47	4	82	18.03	15.21
SIZE1	4	439	106.44	114.65	115	1009	275.58	171.60
Change	2	289	46.82	71.89	6	217	64.23	43.13

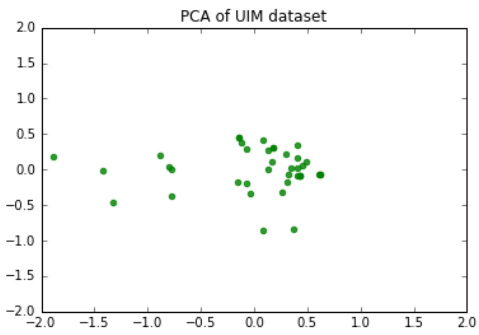


Fig. 3 PCA mapping: UIMS dataset

图 3 PCA 映射: UIMS 数据集

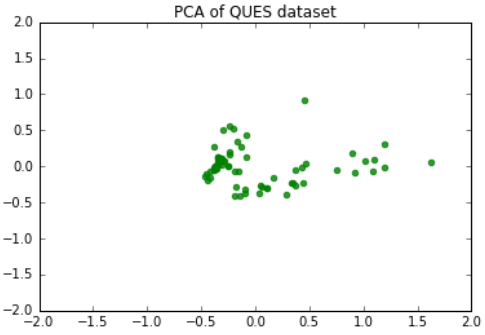


Fig. 4 PCA mapping: QUES dataset

图 4 PCA 映射: QUES 数据集

本文采用 scikit-learn0.18.1 库对数据集进行归一化处理和数据集划分。对于半监督学习，数据集的标签集和无标签集比例为 2:1，我们根据图 3 和图 4 的数据分布特点，对于标签集和无标签集，利用 kmeans 算法分别将 UIMS 数据集聚为 3 簇，QUES 数据集聚为 4 簇。然后从标签集的每个簇中按 3:1 的比例随机抽取 25%的数据作为测试集，再从剩余的数据中按 4:1 的比例抽取 80%的数据构成标签训练集，从未标签集中按 4:1 的比例抽取 80%的数据构成无标签训练集。经过上述处理后，UIMS 数据集含 39 例数据，其中 6 例构成测试集，10 例构成无标签训练集，16 例构成标签训练集；QUES 数据集含 71 例数据，其中 11 例构成测试集，19 例构成无标签训练集，29 例构成标签训练集。

对于对照实验的监督学习，利用初始带标签数据集，采用相同的方法将 UIMS 数据集聚为 3 簇，QUES 数据集聚为 4 簇。然后从每个簇中按 5:1 的比例随机抽取数据构成测试集，再从剩余的数据中按 4:1 的比例抽取 80%的数据构成标签训练集。经过上述处理后，UIMS 数据集含 39 例数据，其中 6 例构成测试集，26 例构成训练集；QUES 数据集含 71 例数据，其中 11 例构成测试集，48 例构成训练集。

4.3.2 实验设计

为了验证本文方法的有效性和准确性，我们选取了近十年研究中预测准确率最高的 GMDH 方法作为基线方法（详见表 2），而本文 SGMDH 方法恰好基于 GMDH 方法构建预测模型，可见选择 GMDH 方法作为基线方法显然是合理的。为了回答实验预设的两个问题，本文设计了两组实验：(1) 针对 RQ1 本文设计了一组和基线方法的对照实验，在相同环境和相同参数设置下分别利用两个数据集构建可维护性预测模型。(2) 针对 RQ2 本文设计了第二组实验，在相同环境、相同采样方法和相同实验设置下，对本文的半监督方法与监督方法进行比对实验。

本文实验采用 Python 机器学习工具 scikit-learn0.18.1 实现，其中 GMDH 算法借助了 gmdhpy0.1.1a0 第三方库来实现，实验中模型的参数设置均采用默认值。为了实验的公平性和客观性，实验均采用随机采样方法抽取实验所需数据。

4.4 实验结果与分析

4.4.1 本文方法实验结果

为了验证本文方法可维护性预测性能的优劣性，并回答预设的两个 RQ 问题，我们分别对基线方法、基于神经网络(MLP)和支持向量回归机(SVR)的监督方法和半监督方法，在 UIMS 数据集和 QUES 数据集上进行软件可维护性预测实验。为了使实验更公平客观，我们对每组实验重复执行 20 次，记录其最好的实验结果。表 4 显示了在 UIMS 数据集上进行可维护性预测的实验结果，图 5 显示了各模型在 UIMS 数据集上取得的 MMRE 和 pred(0.25)指标结果对比。表 5 显示了在 QUES 数据集上进行可维护性预测的实验结果，图 6 显示了各模型在 QUES 数据集上取得的 MMRE 和 pred(0.25)指标结果对比。

Table 4 Maintainability Prediction Results: UIMS dataset

表 4 可维护性预测结果: UIMS 数据集

模型		MMRE	Max.MRE	pred(0.25)
Baseline		0.286	0.754	0.571
监督方法	MLP	0.259	0.646	0.571
	SVR	0.220	0.639	0.571
	GMDH	0.250	0.581	0.571
半监督方法	MLP	0.210	0.390	0.5
	SVR	0.171	0.342	0.714
	SGMDH	**0.166**	**0.429**	**0.833**

Table 5 Maintainability Prediction Results: QUES dataset

表 5 可维护性预测结果: QUES 数据集

模型		MMRE	Max.MRE	pred(0.25)
Baseline		0.216	0.747	0.769
监督方法	MLP	0.494	0.839	0.142
	SVR	0.270	0.501	0.383
	GMDH	0.191	0.543	0.786
半监督方法	MLP	0.422	0.708	0.231
	SVR	0.192	0.436	0.692
	SGMDH	**0.118**	**0.305**	**0.929**

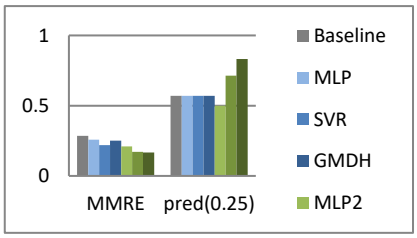


Fig. 5 Prediction accuracy results: UIMS dataset

图 5 预测结果对比: UIMS 数据集

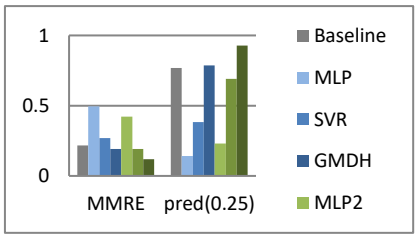


Fig. 6 Prediction accuracy results: QUES dataset

图 6 预测结果对比: QUES 数据集

4.4.2 实验结果分析

(1) RQ1 分析

本节针对 RQ1，对基线方法和本文 SGMDH 方法在 UIMS 数据集和 QUES 数据集上取得的预测结果进行分析和比较。图 7 显示了在 UIMS 数据集上取得的实验结果对比，图 8 显示了在 QUES 数据集上取得的实验结果对比。

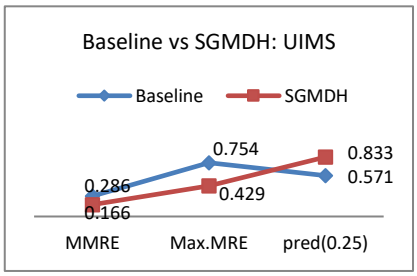


Fig. 7 Baseline & SGMDH prediction results : UIMS dataset

图 7 Baseline 和 SGMDH 方法指标对比: UIMS 数据集

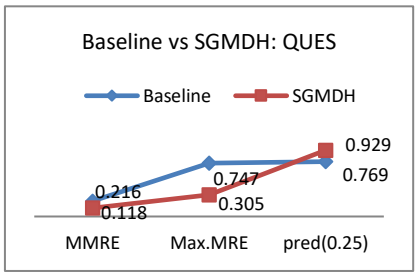


Fig. 8 Baseline & SGMDH prediction results : QUES dataset

图 8 Baseline 和 SGMDH 方法指标对比: QUES 数据集

通过对比基线方法，从图 7 和图 8 可以看出：采用本文方法和基线方法分别在 UIMS 数据集和 QUES 数据集上得到的结果曲线形状基本相似，本文方法的 MMRE 和 Max.MRE 指标都位于基线方法之下，而 pred(0.25)指标都位于基线方法之上，说明本文方法较基线方法具有更准确的预测性能。

基于以上两组数据集的实验结果可以得出以下结论：本文方法在 UIMS 数据集和 QUES 数据集上得到的三个指标结果都比基线方法有较大提升。其中 MMRE、Max.MRE 和 pred(0.25)指标分别在 UIMS 数据集上提升了 42%、43%和 46%，在 QUES 数据集上提升了 45%、59%和 21%。初步验证了本文基于采样的半

监督方法的有效性，表明 SGMDH 方法能够显著提高面向对象软件的可维护性预测能力。

(2) RQ2 分析

本节针对 RQ2，对监督方法和本文半监督方法的可维护性预测性能进行比较和分析，我们分别对两种方法基于 MLP、SVR 和 GMDH 算法构建了可维护性预测模型，并在 UIMS 数据集和 QUES 数据集上进行可维护性预测。图 9 显示了在 UIMS 数据集上取得 MMRE 指标结果对比，图 10 显示了在 UIMS 数据集上取得 pred(0.25)指标结果对比。

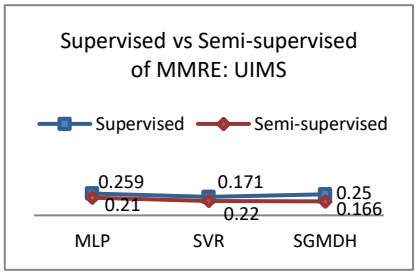


Fig. 9 Accuracy results of MMRE: UIMS dataset

图 9 模型 MMRE 指标对比：UIMS 数据集

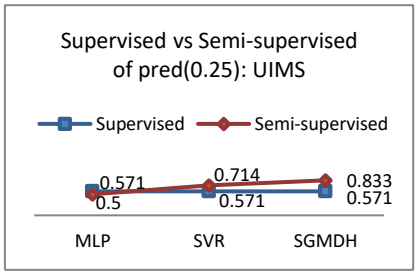


Fig. 10 Accuracy results of pred(0.25): UIMS dataset

图 10 模型 pred(0.25)指标对比：UIMS 数据集

从 UIMS 数据集上取得的实验结果可以看出：

1) 从图 9 可以看出，半监督方法的 MMRE 曲线位于监督方法之下，表明和监督方法相比本文半监督方法能够显著提高可维护性预测性能。基于本文半监督方法构建的 MLP、SVR 和 GMDH 可维护性预测器的预测性能分别比监督方法提升了 19%、22%和 34%。

2) 从图 10 可以看出，半监督方法的 pred(0.25)曲线基本位于监督方法之上，表明半监督方法比监督方法能获得更准确的预测能力。图中基于 MLP 构建的半监督预测器和监督预测器的 pred(0.25)值相当，而基于 SVR 和 GMDH 算法构建的半监督预测器的预测能力分别比监督方法提升了 25%和 46%。

3) 在 UIMS 数据集上验证了本文的半监督方法比监督方法能获得更显著而优异的预测能力，且本文 SGMDH 方法取得了最好的预测性能（MMRE 为 0.166）。

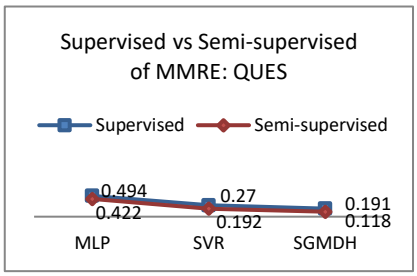


Fig. 11 Accuracy results of MMRE: QUES dataset

图 11 模型 MMRE 指标对比：QUES 数据集

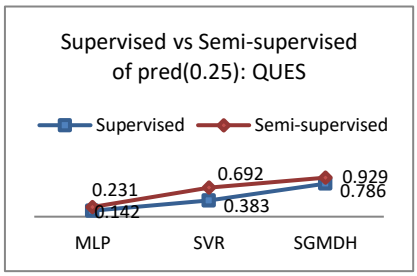


Fig. 12 Accuracy results of pred(0.25): QUES dataset

图 12 模型 pred(0.25)指标对比：QUES 数据集

图 11 显示了在 QUES 数据集上取得 MMRE 指标结果对比，图 12 显示了在 QUES 数据集上取得 pred(0.25)指标结果对比。根据在 QUES 数据集上的实验结果可得：

1) 从图 11 可以看出，半监督方法的 MMRE 曲线位于监督方法之下，表明和监督方法对比本文半监督方法能够显著提高可维护性预测性能。基于本文半监督方法构建的 MLP、SVR 和 GMDH 可维护性预测器的预测性能分别比监督方法提升了 15%、29%和 38%。

2) 从图 12 可以看出，半监督方法的 pred(0.25)曲线位于监督方法之上，表明半监督方法比监督方法能获得更准确的预测能力。基于 MLP、SVR 和 GMDH 算法构建的半监督预测器的准确预测能力分别比监督方法提升了 63%、81%和 18%。

3) 在 QUES 数据集上验证了本文的半监督方法比监督方法能获得更显著而优异的预测能力，且本文 SGMDH 方法取得了最好的预测性能（MMRE 为 0.118）。

基于上述两组实验结果的分析可以得出：与监督方法相比，本文半监督方法在 UIMS 数据集和 QUES 数据集上均能取得最好的预测性能，验证了本文基于采样的半监督方法能够显著提高软件可维护性预测能力，同时本文 SGMDH 方法取得了最优的预测性能。

4.4.3 有效影响因素分析

本文经过实证研究验证了 SGMDH 方法的有效性，影响有效性的因素主要有：(1)存在隐藏缺陷导致实验结果不准确；(2) 采样过程使实验结果存在随机因素；(3) 存在数据质量问题导致构建模型不准确；(4) 方法的一般性问题：提出的半监督方法是否也适用于其他算法；(5) 结论是否可信；(6) 评价指标是否合理。

针对以上影响有效性因素问题，本文做了如下工作：(1) 在编码实现过程中反复对逻辑正确性进行验证，同时请第三者对代码进行检查并进行实验，减小了隐藏缺陷。(2) 通过聚簇再采样的方式减小了采样过程中的随机因素，同时缓解了数据分布不平衡问题。(3) 采用数据预处理方式减小数据质量的影响。(4) 针对本文半监督方法分别基于 MLP、SVR 和 GMDH 算法建立可维护性预测模型，增加了本文半监督方法的一般性；(5) 分别对基线方法及 MLP 和 SVR 方法，在两个公开不同数据分布的 UIMS 和 QUES 数据集上进行实验，增加了本文方法的可信度。(6) 可维护性预测问题重点关注预测准确率问题，误差率指标(MRE)能够反映真实值和预测值的差异程度，pred(q)指标能够反映预测精准率，是可维护性预测领域最常用的重要指标[8-10, 12, 15, 18, 23]，本文采用这两个评价指标对实验结果进行评价，并对实验结果作了进一步分析，验证了本文结论的有效性。

5 存在的局限性和不足

我们认为，SGMDH 方法仍存在一些局限性和不足，具体包括：(1) 本文 SGMDH 方法采用面向对象度量因子对可维护性进行研究，因此本文方法只适用于面向对象软件的可维护性预测。(2) 参数的设置问题，参数设置的好坏与模型性能优劣有很大关系，本文设置的所有模型参数为默认值，通过调参可能会进一步提高可维护性预测性能，因此，参数设置问题有待进一步讨论和验证；(3) 本文使用的未标记数据仅占标记数据的二分之一，可以尝试进一步减少标记数据的比例，从而进一步提高方法的经济适用性。(4) 本文 SGMDH 方法适用于小数据样本，对于样本较大的数据集会增加系统复杂度，使资源开销成倍增加。(5) 本文通过预处理方式减少数据质量的影响，可以考虑对样本特征做进一步分析，根据不同特征的影响程度建立更准确的可维护性预测模型。(6) 本文采用的 UIMS 和 QUES 数据集发布时间较早且数据实例少，可以尝试在更多公开数据集上进行试验，进一步验证本文方法的一般性和普适性。

6 结束语

本文提出了一种基于半监督数据分组处理的面向对象可维护性预测方法 SGMDH，通过在两个公开的数据集 UIMS 和 QUES 上进行实证研究，表明 SGMDH 方法比基线方法和监督方法均能获得更显著而优异的可维护性预测性能，并能有效提高可维护性预测能力，同时由于 SGMDH 方法只使用了部分标记数据，因此具有更高的应用价值和更强的普适性。

半监督方法利用少量标记样本进行学习，分类问题由于涉及的目标变量少而明确，因而应用相对容易。而回归问题由于要预测一个准确的连续变量实数值，问题本身存在一定难度，当在数据量小且只使用部分标记样本进行回归任务时，比分类问题难很多倍。本文 SGMDH 方法将半监督学习应用于软件可维护性预测领域本身就是一种创新和挑战。本文通过实证研究，证明了方法的有效性显然是一种突破性的成功。

但是仍然存在一些问题，因此在未来研究中我们将探索更有利于该领域研究的数据集，尝试用本文提出的方法在其他数据集上进行实验，并结合新技术在半监督学习应用于回归预测领域上进行更深入的研究。

References:

[1] Standard, I. *IEEE Standard Glossary of Software Engineering Terminology*. in *IEEE Std*. 2002.

[2] Bhatt, P., et al., *Influencing factors in outsourced software maintenance.* Acm Sigsoft Software Engineering Notes, 2006. **31**(3): p. 1-6.

[3] Zelkowitz, M.V., *Perspectives in Software Engineering*. 1978: ACM. 197-216.

[4] Chidamber, S.R. and C.F. Kemerer. *Towards a metrics suite for object oriented design*. in *Conference proceedings*. 1991.

[5] Li, W. and S. Henry, *Object-oriented metrics that predict maintainability.* Journal of Systems & Software, 1993. **23**(2): p. 111-122.

[6] Fioravanti, F. and P. Nesi, *Estimation and Prediction Metrics for Adaptive Maintenance Effort of Object-Oriented Systems.* Software Engineering IEEE Transactions on, 2001. **27**(12): p. 1062-1084.

[7] Misra, S.C., *Modeling Design/Coding Factors That Drive Maintainability of Software Systems.* Software Quality Journal, 2005. **13**(3): p. 297-320.

[8] Koten, C.V. and A.R. Gray, *An application of Bayesian network for predicting object-oriented software maintainability.* Information & Software Technology, 2006. **48**(1): p. 59-67.

[9] Zhou, Y. and H. Leung, *Predicting object-oriented software maintainability using multivariate adaptive regression splines*. 2007: Elsevier Science Inc. 1349-1361.

[10] Thwin, M.M.T. and T.S. Quah, *Application of neural networks for software quality prediction using object-oriented metrics.* Journal of Systems & Software, 2003. **76**(2): p. 147-156.

[11] Thwin, M.M.T. and T.S. Quah, *Application of neural networks for software quality prediction using object-oriented metrics.* Journal of Systems & Software, 2005. **76**(2): p. 147-156.

[12] R Malhotra, A.C., *Software Maintainability Prediction using Machine Learning Algorithms.* Software Engineering : An International Journal (SEIJ), 2012. **Vol. 2**.

[13] Conte, S.D., H.E. Dunsmore, and V.Y. Shen, *Software engineering metrics and models*. 1986: Benjamin/Cummings Publishing Company, Inc.

[14] Macdonell, S.G., *Establishing relationships between specification size and software process effort in CASE environments.* Information & Software Technology, 1997. **39**(1): p. 35-45.

[15] Elish, M.O., H. Aljamaan, and I. Ahmad, *Three empirical studies on predicting software maintainability using ensemble methods.* Soft Computing, 2015. **19**(9): p. 2511-2524.

[16] Malhotra, R. and A. Chug, *Application of Group Method of Data Handling model for software maintainability prediction using object oriented systems.* International Journal of System Assurance Engineering & Management, 2014. **5**(2): p. 1-9.

[17] Jin, C. and J.A. Liu. *Applications of Support Vector Mathine and Unsupervised Learning for Predicting Maintainability Using Object-Oriented Metrics*. in *Second International Conference on Multimedia and Information Technology*. 2010.

[18] Aggarwal, K.K., et al., *Application of Artificial Neural Network for Predicting Maintainability using Object Oriented Metrics.* Enformatika, 2006. **15**.

[19] Coleman, D., et al., *Using metrics to evaluate software system maintainability.* Computer, 1994. **27**(8): p. 44-49.

[20] Elish, M.O. and K.O. Elish. *Application of TreeNet in Predicting Object-Oriented Software Maintainability: A Comparative Study*. in *European Conference on Software Maintenance and Reengineering*. 2009.

[21] Bengtsson, O. and J. Bosch. *Architecture Level Prediction of Software Maintenance*. in *European Conference on Software Maintenance and Reengineering*. 1999.

[22] Quah, T.S. and M.M.T. Thwin, *Application of Neural Networks for Software Quality Prediction*

Using Object-Oriented Metrics. 2003. **76**(2): p. 147-156.

[23] Lucia, A.D., E. Pompella, and S. Stefanucci, *Assessing effort estimation models for corrective maintenance through empirical studies.* Information & Software Technology, 2005. **47**(1): p. 3-15.

[24] Li, M., et al., *Sample-based software defect prediction with active and semi-supervised learning.* Automated Software Engineering, 2012. **19**(2): p. 201-230.

[25] He, Q., B. Shen, and Y. Chen. *Software Defect Prediction Using Semi-Supervised Learning with Change Burst Information*. in *Computer Software and Applications Conference*. 2016.

[26] Jing, X.Y., et al. *Missing data imputation based on low-rank recovery and semi-supervised regression for software effort estimation*. in *International Conference on Software Engineering*. 2016.

[27] Yi, Y., et al., *Semi-supervised local ridge regression for local matching based face recognition.* Neurocomputing, 2015. **167**(C): p. 132-146.

[28] Cheng, B., et al., *Predicting Clinical Scores Using Semi-supervised Multimodal Relevance Vector Regression*. 2011: Springer Berlin Heidelberg. 241-248.

[29] Khan, F.M. and Q. Liu. *Transduction of Semi-supervised Regression Targets in Survival Analysis for Medical Prognosis*. in *IEEE International Conference on Data Mining Workshops*. 2011.

[30] Tang, Y., et al. *Local semi-supervised regression for single-image super-resolution*. in *IEEE International Workshop on Multimedia Signal Processing*. 2011.

[31] Xie, L., M.A. Carreiraperpinan, and S. Newsam, *Semi-supervised regression with temporal image sequences.* 2010. **119**(5): p. 2637-2640.

[32] Bai, X., et al. *Semi-supervised regression for evaluating convenience store location*. in *International Jont Conference on Artifical Intelligence*. 2009.

[33] Ivakhnenko, A.G., *The Group Method of Data Handling-A rival of the Method of Stochastic Approximation.* Soviet Automatic Control, 1968. **13**.

[34] TAMURA, H. and T. KONDO, *Heuristics free group method of data handling algorithm of generating optimal partial polynomials with application to air pollution prediction†.* International Journal of Systems Science, 1980. **11**(9): p. 1095-1111.

[35] Onwubolu, G.C., *Design of hybrid differential evolution and group method of data handling networks for modeling and prediction*. 2008: Elsevier Science Inc. 3616-3634.

[36] Hady, M.F.A. and F. Schwenker. *Semi-supervised Learning for Regression with Co-training by Committee*. in *International Conference on Artificial Neural Networks*. 2009.

[37] Zhou, Z.H. and M. Li. *Semi-supervised regression with co-training*. in *International Joint Conference on Artificial Intelligence*. 2005.

[38] Zhou, Z.H. and M. Li, *Semisupervised Regression with Cotraining-Style Algorithms.* IEEE Transactions on Knowledge & Data Engineering, 2005. **19**(11): p. 908-913.

[39] Xu, M., F. Sun, and X. Jiang. *Multi-label learning with co-training based on semi-supervised regression*. in *International Conference on Security, Pattern Analysis, and Cybernetics*. 2014.

[40] Zhu, X. and Z. Ghahramani, *Learning from Labeled and Unlabeled Data with Label Propagation.* 2002.

[41] Mayrhauser, A.V. and A.M. Vans, *Program comprehension during software maintenance and evolution.* Computer, 1995. **28**(8): p. 44-55.

[42] Vixie, P., *Software Engineering in Open Sources: Voices from the Open Source Revolution.* Contact Center Call Center & Ip Solutions, 1999.

[43] Srinivasan, D., *Letters: Energy demand prediction using GMDH networks.* Neurocomputing, 2008. **72**(1): p. 625-629.

[44] Abdel-Aal, R.E., M.A. Elhadidy, and S.M. Shaahid, *Modeling and forecasting the mean hourly wind speed time series using GMDH-based abductive networks.* Renewable Energy, 2009. **34**(7): p. 1686-1699.

[45] Mrugalski, M., *An unscented Kalman filter in designing dynamic GMDH neural networks for robust fault detection.* International Journal of Applied Mathematics & Computer Science, 2013. **23**(1): p. 157-169.

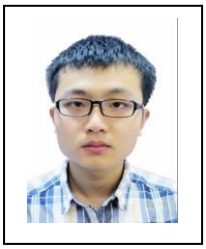
[46] Fenton, N. and S.L. Pfleeger. *Software metrics (2nd ed.): a rigorous and practical approach*. 1997.

朱佳俊(1991-)，女，云南普洱人，云南大学硕士研究生，CCF 会员，主要研究领域为软件工程，软件演化，数据挖掘。

王炜(1979-)，男，云南昆明人，2009 年于云南大学系统分析与集成专业获博士学位，现为云南大学副教授，CCF 会员，主要研究领域为软件工程，软件演化，数据挖掘。发表论文 15 篇，出版教材 1 部，主持省部级项目 5 项。

李彤(1963-)，男，河北石家庄市人，2007 年于英国 De Montfort 大学软件工程专业获博士学位，现为云南大学软件学院院长、教授、博士生导师，CCF 高级会员，主要研究领域为软件工程，信息安全。发表论文 100 余篇、专著 2 部、教材 5 部，主持国家级项目 5 项、省部级项目 14 项、其他项目 20 余项。

何云(1989-)，男，云南建水人，云南大学博士研究生，CCF 学生会员，主要研究领域为软件工程，软件演化，数据挖掘。主持云南省教育厅科学基金研究生项目一项。



向威(1994-)，男，湖南邵阳人，云南大学硕士研究生，CCF 学生会员，主要研究领域为软件工程，软件演化，数据挖掘。