Fusion of Multi-RSMOTE With Fuzzy Integral to Classify Bug Reports With an Imbalanced Distribution

Rong Chen, *Member, IEEE*, Shi-Kai Guo, Xi-Zhao Wang, *Fellow, IEEE*, and Tian-Lun Zhang

***Abstract*—With the help of automated classification, *severe* bugs can be rapidly identified so that the latent damage to software projects can be minimized. However, bug report datasets commonly suffer from disproportionate number of category samples. When presented with the situation of class imbalance, most standard classification learning approaches fail to properly learn the distributive characteristics of the samples and tend to result in unfavorable performance to predict class label. In this case, imbalanced learning becomes critical to advance classification algorithms. In this paper, we propose an improved synthetic minority oversampling technique to avoid the degraded performance caused by class imbalance in bug report datasets. Moreover, to lessen the chance of occasionalities in random sampling process, we propose a repeated sampling technique to train different, but related classifiers. Finally, an ensemble algorithm based on Choquet fuzzy integral is employed to combine the wisdom of crowds and make better decisions. We conduct comprehensive experiments on several bug report datasets from real-world bug repositories. The results demonstrate that the proposed method boosts the classification performance across the classes of the data. Specifically, compared with various ensemble learning techniques, the Choquet fuzzy integral achieves outstanding results on integrating multiple random oversampling techniques.**

***Index Terms*—Bug report identification, class imbalance, fuzzy integral, software quality.**

I. Introduction

In RECENT years, because of the rapid increment of software development, software systems have become larger and more complex, which directly causes numerous bugs to appear during software development [1], [36]. To insure the reliability of software systems, accurate recognition of bug reports has become increasingly prominent. In bug triaging systems (e.g., Bugzilla [19], JIRA [49], and Mantis [50]), the information of bug reports could help developers reproduce and repair the bugs, and effectively solve problems of software reliability. The bug report with *severe* label tends to indicate that the corresponding bug should be fixed as soon as possible, in this case, the damage caused by *severe* bugs could be reduced and mitigated, greatly. With the increasing amount of information about bugs encountered in triaging system, some forms of automation in identifying the severity of bug reports become an overwhelming research [2].

In fact, bug report datasets are always characterized by imbalanced distributions, whereas most classification approaches expect equal misclassifying costs or balanced class distribution. As a result, such imbalanced bug report data tend to cause degradation of performance in classification learning [2]–[4], [45]. To solve this problem, Lamkanfi *et al.* [3] manually selected a small dataset with a balanced distribution from original bug reports to insure that the classification approaches were not hindered by the class imbalance. However, the bug reports selected manually from imbalanced datasets could tend to result in missing some critical information. To achieve robust methods, Yang *et al.* [2] employed four imbalanced learning strategies (ILS) [i.e., random undersampling (RUS), random oversampling (ROS), synthetic minority over sampling technique (SMOTE), and cost-matrix adjuster (CMA)] to recognize the high-impact bug reports with class imbalance. Although some promising benefits have been shown in [2], there exist inherent drawbacks in these imbalanced learnings. CMA is sensitive to noise data [61]; RUS tends to miss some potentially crucial data and lead to underfitting issues; ROS often causes overfitting because some redundant data may be selected to augment original dataset [7], [8], [21], [60]; in addition, SMOTE suffers from a poor generalization ability due to its simple linear sampling space [28]. Moreover, random sampling can produce uncertainty, and some sampling results will not be in agreement with real dataset distributions.

To solve these problems, an approach to fuse multiple-improved SMOTE with the Choquet fuzzy integral is proposed to recognize the severity of bug reports characterized by imbalanced distribution. First, the improved SMOTE, i.e., rect-

Manuscript received May 8, 2018; revised November 28, 2018; accepted January 28, 2019. Date of publication February 15, 2019; date of current version November 28, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61672122, Grant 61602077, Grant 61772344 and Grant 61732011, in part by the Public Welfare Funds for Scientific Research of Liaoning Province of China under Grant 20170005, in part by the Natural Science Foundation of Liaoning Province of China under Grant 20170540097, and in part by the Fundamental Research Funds for the Central Universities under Grant 3132016348. *(Corresponding author: Xi-Zhao Wang.)*

R. Chen, S.-K. Guo, and T.-L. Zhang are with the College of Information Science and Technology, Dalian Maritime University, Dalian 116206, China (e-mail: shikai.guo@dlmu.edu.cn; rchen@dlmu.edu.cn; tlzhang@dlmu.edu.cn).

X.-Z. Wang is with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China (e-mail: xizhaowang@ieee.org).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org provided by the authors.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TFUZZ.2019.2899809

angle SMOTE (RSMOTE) approach, is used to weaken the imbalance ratio by generating minority-class samples, which are randomly synthesized in a multidimensional rectangle sample space. In addition, with two proposed constraints, the synthetic minority-class bug reports can be generated in a robust way. Second, to avoid the uncertainty caused by random sampling, a repeated sampling technique is proposed to obtain multiple balanced datasets by using RSMOTE. Then, several different classifiers are built on these balanced datasets. At last, an ensemble method based on Choquet fuzzy integral is employed to integrate these trained classifiers to recognize the severity of bug reports [62]. Comprehensive experiments are conducted on three bug repositories, i.e., *Eclipse* [16], *Mozilla* [17], and *GNOME* [18]. These experimental results indicate that RSMOTE could effectively weaken the imbalanced distribution of datasets and improve the generalization capabilities of classifiers. In addition, fusion of multiple RSMOTE could effectively weaken the uncertainty caused by random sampling, and boost the performance of predicting the class label of bug reports.

Our contributions can be summarized as follows.

1) We consider the imbalance phenomenon of bug reports and propose an improved random sampling approach, named RSMOTE. RSMOTE is a random sampling mechanism used to generate minority-class points from high-dimensional sampling space, which is the main omission in SMOTE [28]. The generalization abilities of several different classification learning approaches are significantly improved by the proposed method. In addition, two constraints are applied to provide a robust way to generate new synthetic samples, i.e., scaling the random sampling scope to a reasonable area and distinguishing the majority-class points in a critical region.
2) An approach to fuse multi-RSMOTE with Choquet fuzzy integral is used to solve the uncertainty caused by random sampling. Several different, but related datasets are produced by a repeated sampling process. An ensemble method based on Choquet fuzzy integral is used to integrate the multiclassifiers trained over these balanced datasets. To the best of our knowledge, this is the first endeavor of such technique for exploring the fusion of multiple RSMOTE with Choquet fuzzy integral to classify bug reports.
3) Two evaluation criteria are used in experimental part to evaluate the proposed approach. The results on 16 components show that Choquet fuzzy integral ensemble learning outperforms other popular ensemble methods, such as majority voting, bagging, and Adaboost.

II. Background Knowledge and Motivation

In Sections II-A and II-B, we introduce the automatic bug report classification in software engineering and the propaedeutic of the fuzzy integral, respectively. The motivation of proposing the fusion of classifiers with a fuzzy integral method to recognize the severity of bug reports characterized by an imbalanced distribution is introduced in Section II-C.

A. Automatic Bug Report Classification in Software Engineering

Automatic bug reports classification technique can reduce the latent damage to software projects.

Antoniol *et al.* [30] used three classifiers [Naive Bayes classifier (*NB*), decision trees (*J48*), and logistic regression (*LR*)] to classify the bug report and they analyzed the important features that have a greater impact on the classification. Menzies *et al.* [25] used standard text mining methods to classify the severity of NASA bug reports. In order to improve the performance identifying high-impact bug reports, Yang *et al.* [2] combined four widely used ILS with four classification approaches. In order to recognize the severity of Android bug reports with limited class label, Guo *et al.* [31] proposed a knowledge transferring approach; the knowledge acquired from different software projects (*Eclipse*, *Mozilla*, and *GNOME*) is used to classify the severity of Android bug reports. Xia *et al.* [37] proposed a method, namely ELBlocker, to identify the blocking bug reports with imbalanced distribution. ELBlocker first trains the classifiers over multiple disjoint datasets. Then, ELBlocker uses estimate threshold approach to estimate the weight of each classifier. Finally, all classification results are integrated to identify the blocking bugs.

Xuan *et al.* [32] proposed a ranking approach to recommend appropriate commenters to repair the bugs. This method is based on analyzing the relationship between commenters and bug comments. Anvik and Murphy [33] proposed an automated method to simplify the development process, which could assist the triagers to recommend the component of bug reports and developers. Tian *et al.* [34] performed feature extraction on bug reports, and employed multifactors (“temporal,” “related report,” “severity,” “textual,” “author,” and “product”) to identify the priority of bug reports. Zhang *et al.* [36] proposed a more accurate approach to perform automatic severity prediction and fixer recommendation. The top k historical bug reports which are similar to a new one are searched by using K-nearest neighbor and REP. The features of these reports then are extracted for prediction and recommendation algorithms.

Feng *et al.* [10] proposed three strategies to find bugs as early as possible. The three strategies are diversity strategy, risk strategy, and compound strategy (DivRisk). For mobile crowdsourcing testing, bug reports are often composed of screenshots and text descriptions. Feng *et al.* [55] used multiobjective to prioritize bug reports. One is to use spatial pyramid matching approach to analyze similar screenshots, and the other one is to use natural language processing techniques to measure the distance between bug reports. To overcome the local bias of bug reports, Wang *et al.* [57] proposed a cluster-based method to cluster the similar bug reports and trained the classifiers with most similar bug reports, respectively. Then, they used ensemble approach to predict the true fault bug reports. In their follow-up work, Wang *et al.* [56] proposed an approach called local-based active classification to predict the true fault bug reports, which solves the local bias problem and lacking of labeled bug reports problem.

B. Propaedeutic of Fuzzy Integral

Bug report processing in our paper is transferred to a fusion problem of multiple classifiers. The training set for each classifier is generated by a synthetic mechanism of oversampling with respect to minority class, i.e., for each time, adding a number of new synthetic samples as minority and keeping the majority unchanged. This synthetic process obviously indicates an interaction existing among the multiple classifiers. As a fusion tool, fuzzy integral has the advantages of modeling and handling interactions (such as the subadditivity and superadditivity) among the classifiers in comparison with other fusion schemes [5], [11], [27], [47], [48]. Therefore, we select the fuzzy integral as a fusion tool, which is confirmed experimentally to be successful in following sections.

Definition 1: Given a nonempty set *X*, let Ω be the σ algebra consisting of a group of subset of *X*, the fuzzy measure on Ω is a set function $g : \Omega \to [0, 1]$.

1) $g(\emptyset) = 0, g(X) = 1$.
2) $\forall A, B \subseteq \Omega$, if $A \subset B$, then $g(A) \leq g(B)$.
3) If $\{A_n\} \subset \Omega, A_1 \subset A_2 \subset \cdots A_n$, and $\bigcup_{n=1}^{\infty} A_n \in \Omega$, then $\lim_{n\to\infty} g(A_n) = g(\bigcup_{n=1}^{\infty} A_n)$.
4) If $\{A_n\} \subset \Omega, A_1 \supset A_2 \supset \cdots \supset A_n, g(A_1) < \infty$, and $\bigcap_{n=1}^{\infty} A_n \in \Omega$, then $\lim_{n\to\infty} g(A_n) = g(\bigcap_{n=1}^{\infty} A_n)$.

According to Definition 1, fuzzy measure does not require additivity, when $g(A \cup B) < g(A) + g(B), A \cap B = \emptyset$ holds well, the fuzzy measure is called subadditivity, while $g(A \cup B) > g(A) + g(B), A \cap B = \emptyset$ holds well, the fuzzy measure is called superadditivity. For a finite state space *X*, the power set of *X* is usually used as the σ algebra Ω in Definition 1; in this case, a set function satisfying the first two conditions of Definition 1 is defined as fuzzy measure. In ensemble learning, the set of classifiers is finite, therefore, the fuzzy measure and fuzzy integral in our study are defined over finite set. For a generalization of probability measure, the monotonicity could replace the additivity of probability measures, as shown in (1). Regarding a nonadditivitive g, the sum of all individual classifier contribution may be more or less than the contribution of integrated classifiers, as shown in (2) and (3)

$$g(A \cup B) = g(A) + g(B), \forall A, B \subset p(X), A \cap B = \emptyset \quad (1)$$

$$g(A \cup B) \geq g(A) + g(B), \forall A, B \subset p(X), A \cap B = \emptyset \quad (2)$$

$$g(A \cup B) \leq g(A) + g(B), \forall A, B \subset p(X), A \cap B = \emptyset. \quad (3)$$

Moreover, in this paper, we always suppose that let the fuzzy measure be normal, i.e., $g(\emptyset) = 0, g(X) = 1$. We will focus on the special type of fuzzy measures, i.e., λ-fuzzy measure which has been widely used in ensemble learnings [5], [9], [38], [39], [47], [48].

Definition 2: For arbitrary $A, B \subset \Omega$, and $A \cap B = \emptyset$, g is called a λ-fuzzy measure, if g satisfies

$$g(A \cup B) = g(A) + g(B) + \lambda \times g(A) \times g(B) \quad (4)$$

where $\lambda > -1$ and $\lambda \neq 0$. The value of λ can be computed by the following equation.

Property 1: Suppose that g is a fuzzy measure, $A_i \cap A_j = \emptyset$, $(i \neq j, 1 \leq i, j \leq m)$. Then

$$g\left(\bigcup_{i=1}^{m} A_i\right) = \begin{cases} \frac{1}{\lambda}\left(\prod_{i=1}^{m}(1 + \lambda \times g(A_i)) - 1\right), \lambda \neq 0 \\ \sum_{i=1}^{m} g(A_i), \lambda. = 0. \end{cases} \quad (5)$$

Property 2: Let $X = \{x_1, x_2, \ldots, x_n\}$, if a λ-fuzzy measure g is greater than zero at least two individual point, i.e., there exist $\{x_1^*\}, \{x_2^*\} \subset X$, such that $g(\{x_1^*\}) > 0, g(\{x_2^*\}) > 0$.

Then λ can be solved by the following:

$$\lambda + 1 = \prod_{i=1}^{n}(1 + \lambda \times g(\{x_i\})). \quad (6)$$

It is easy to see the following.

1) If $\sum_{i=1}^{n} g(\{x_i\}) < 1$, then $\lambda > 0$.
2) If $\sum_{i=1}^{n} g(\{x_i\}) = 1$, then $\lambda = 0$.
3) If $\sum_{i=1}^{n} g(\{x_i\}) > 1$, then $-1 < \lambda < 0$.

Definition 3: Suppose that *f* is a function $X \to [0, \infty)$, and g is the λ-fuzzy measure. Then Choquet fuzzy integral with respects to g is defined as

$$(C)\int f dg = \int_0^{\infty} g(\Omega_\alpha) d\alpha \quad (7)$$

where $\Omega_\alpha = \{x | f(x) > \alpha, x \in X\}$ and $\alpha \in [0, \infty)$.

C. Motivation

With the continuous expansion of bugs in software development, bug reports play a very important role to insure the reliability of software [36], [59]. Bug reports can not only contain the necessary information to reproduce and fix the problem, but also contain statistical information to evaluate software quality during software development. The severity label of a bug report is used to determine how soon the bug needs to be fixed, which can help to greatly reduce or mitigate the damage caused by severe bugs.

Due to the huge amount of information about bugs reported by bug tracking systems, there is an increasing need to introduce some form of automation in identifying the severity of bug reports [2], [12], [13], [24]. However, original bug report datasets are often characterized by imbalanced distributions, which hinder traditional classification learning. Moreover, the abilities of most imbalanced learning are limited by their inherent drawbacks, e.g., missing crucial data and replicated redundant data. In this case, we propose an improved oversampling approach to address these issues in a robust manner [3], [4], [14], [15]. In addition, to integrate several different, but related classifiers trained by a multiple sampling technique, an ensemble approach based on Choquet fuzzy integral is introduced in our method.

III. Methodology

In this section, we present the proposed model to predict the severity label of bug reports.

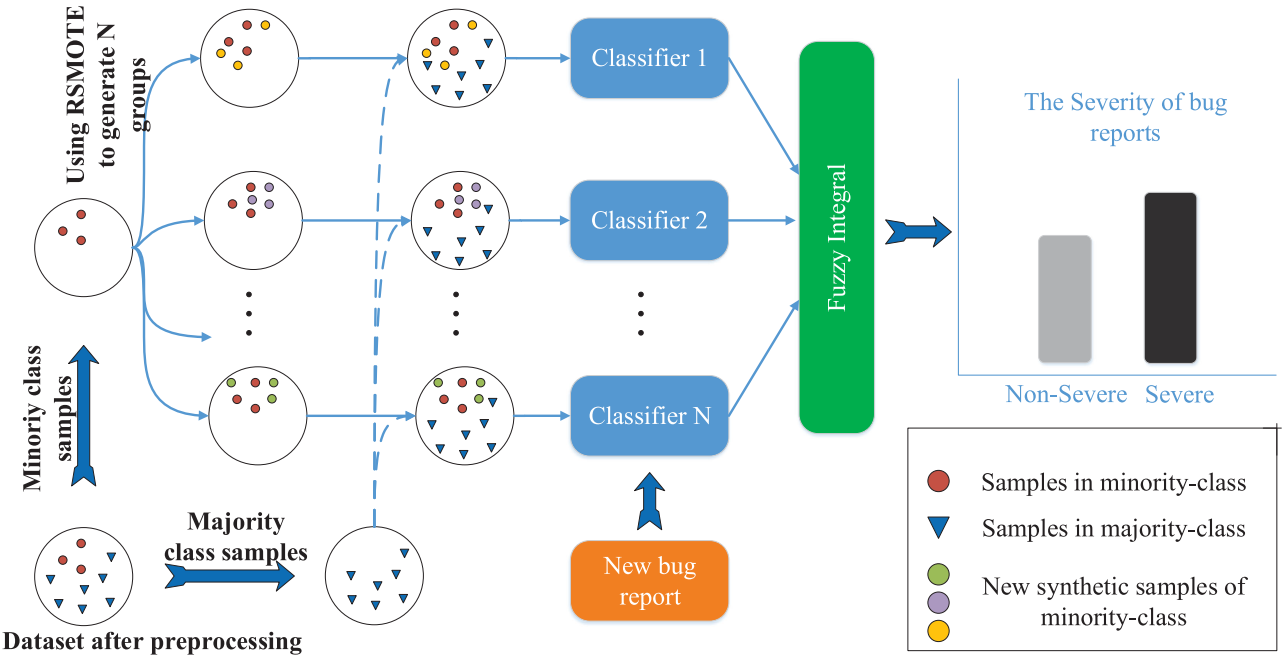


Fig. 1. Entire framework of our approach to address imbalanced issues in bug reports recognition.

A. Model Description

Based on the motivation described in Section II-C, we propose an approach to fuse multi-RSMOTE with the Choquet fuzzy integral to recognize bug reports with class imbalance. As Fig. 1 shows, the framework is composed of two phases: the balance-sample phase and the identification phase. In the balance-sample phase, we first convert bug reports into uniform textual features by using text preprocessing [31]. Then, RSMOTE is used to weaken the imbalanced ratio of bug reports (see Section IV-B). To lessen the uncertainty caused by random sampling, in identification phase, we use the RSMOTE approach to generate multiple balanced datasets. Then, Choquet fuzzy integral is used to fuse multiclassifiers trained by multiple balanced datasets, respectively (see Subsection IV.D). Our approach can not only enhance the generalization ability of oversampling method, but also improve the performance of predicting the class label of bug reports.

B. RSMOTE Approach

In this section, we will introduce the improved random sampling approach in detail. As can be seen in Fig. 2(a), the new synthetic minority-class sample is randomly generated by linear interpolation between two minority-class samples via the SMOTE approach. Instead of a simple linear sampling space, the new synthetic minority-class samples are randomly generated in a multidimensional rectangle area in RSMOTE approach. Finally, two constrains in RSMOTE determine whether the new synthetic minority-class samples can be used to augment the original datasets. The first constraint is scaling the random oversampling scope to a reasonable area. The other constraint is distinguishing the majority-class points in a critical region. Thus, the new synthetic samples can be generated in a robust way. Compared with SMOTE, the generalization ability of oversampling is significantly improved by using RSMOTE due to reasonable constraints. In addition, one can easily note that ROS and SMOTE are the special cases of RSMOTE.

We will introduce the RSMOTE approach and the approach fusing multi-RSMOTE with Choquet fuzzy integral in Sections III-B and III-D, respectively.

Algorithm 1 RSMOTE Algorithm.

Input:
Input the original bug reports (*DT*), the nonsevere bug reports (*T*), the equilibrium number (*N*), the number of features (*n*), and the number of nearest neighbors (*k*).

Output:
$S = DT \cup P'$ (virtual *non-severe* samples).

1: Initialize the virtual samples P';
2: For each X_i in *T*, generate the virtual *nonsevere* samples to P';
   (a) Calculate the Euclidean distances (*R*) between X_i and all other *nonsevere* samples;
   (b) Randomly select $YS_N = \{Y_1, \ldots, Y_j, \ldots, Y_N\}$ from the *k* nearest neighbor samples based on the *R* values;
   (c) For each $Y_j \in YS_N$, randomly generate a new *nonsevere* sample X'_j from an *n*-dimensional rectangle area with X_i and Y_j as the diagonal;
3: Judge whether X'_j satisfies constraints C1 and C2 specified below;
   (a) If the constraints are satisfied, add X'_j to P';
   (b) If not, GOTO Step 2 *(c)* to regenerate X'_j;
4: **return** $S = DT \cup P'$.

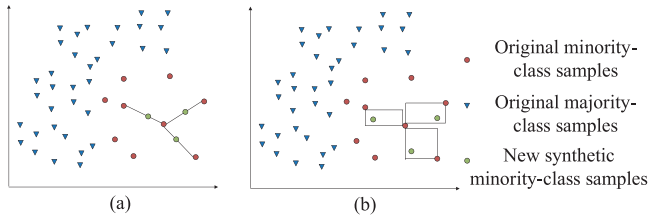


Fig. 2. Diagrams of (a) SMOTE and (b) RSMOTE. Blue triangle represents original major-class samples, green dots represent original minority-class samples, red dots represent new synthetic minority-class samples. Lines and rectangles represent the sampling space of the SMOTE and RSMOTE approaches, respectively.

In Step 1, the set P' is initialized. In Step 2, we generate the new synthetic bug reports to P'. The number of remaining samples is *m*, and the attributes of each X_i can be represented as $(x_{i1}, \ldots x_{it}, \ldots x_{in})$, where $i \in [1, m]$ and $t \in [1, n]$. Similarly, the attributes of each Y_j can be represented as $\{y_{j1}, \ldots y_{jt}, \ldots, y_{jn}\}$, where $j \in [1, N]$ and $t \in [1, n]$. After sufficient iterations, the RSMOTE approach generates a new virtual sample set, which can be represented as $\{X'_1, \ldots, X'_j, \ldots X'_N\}$, where the attributes of each X'_j can be represented as $\{x'_{j1}, \ldots, x'_{jt}, \ldots x'_{jn}\}$, where $j \in [1, N]$ and $t \in [1, n]$.

In the RSMOTE approach, each x'_{jt} is generated between (z^1_{jt}, z^2_{jt}), which can be calculated as follows:

$$z^1_{jt} = x_{jt} - \frac{1}{2} \times |y_{jt} - x_{jt}| \tag{8}$$

$$z^2_{jt} = x_{jt} + \frac{1}{2} \times |y_{jt} - x_{jt}| \tag{9}$$

where $j \in [1, N]$, $t \in [1, n]$, and $|y_{jt} - x_{jt}|$ represents the absolute value of the difference in attribute values between y_{jt} and the sample x_{jt}.

The attributes of the newly generated X'_j can be calculated as follows:

$$x'_{jt} = x_{jt} + \text{random}(0, 1) \times \left(z^2_{jt} - z^1_{jt}\right) \tag{10}$$

where $j \in [1, N]$, $t \in [1, n]$, $\text{random}(0, 1)$ represents the generation of an arbitrary number between 0 and 1.

In Step 3, we judge whether X'_j satisfies the two constraints specified below. When both of these constraints are satisfied, X'_j is added to P'. Otherwise, X'_j is regenerated.

Constraint C1: Let $\text{Dis}(X'_j X_i)$ denote the Euclidean distance between X'_j and X_i, and let $\text{Dis}(Y_j X_i)$ denote the Euclidean distance between Y_j and X_i. When $\text{Dis}(Y_j X_i)$ is greater than $\text{Dis}(X'_j X_i)$, this constraint is satisfied

$$\text{Dis}\left(X'_j X_i\right) = \left\|X'_j - X_i\right\| \tag{11}$$

$$\text{Dis}\left(Y_j X_i\right) = \left\|Y_j - X_i\right\|. \tag{12}$$

Constraint C2: We calculate the Euclidean distances (*R*) between X' and all other original bug reports (*DT*). Then, we find the nearest-neighbor bug report sample *M*. When the severity of *M* is *nonsevere*, this constraint is satisfied.

In Step 4, the RSMOTE approach returns the balanced set of bug reports $S = DT \cup P'$.

C. Case Study of RSMOTE

From Fig. 2, we can see that the RSMOTE approach for generating new synthetic samples can be more flexible and have a wider range. It can make the distribution of the new synthetic minority-class samples be more uniform and reasonable in sample space, thereby improving the generalization capability of the classifiers.

To more intuitively represent the improvement of the RSMOTE approach over other ILS [RUS, ROS, and SMOTE), we take the dataset (Core-XPConnect (*Mozilla*)] in Table I as an example. We use the truncated singular value decomposition (TSVD) approach to reduce the dimensionality to give visual comparison. TSVD is a matrix factorization technique, which is a variant of singular value decomposition (SVD) [51]–[53]. Unlike traditional SVD, TSVD only calculates the first *k* largest singular values, and other singular values are set to 0.

First, we use the RSMOTE, SMOTE, RUS, and ROS approaches to remedy the imbalanced distributions characterizing Core-XPConnect (*Mozilla*). Then, we use the TSVD approach to reduce the multidimensional samples into three-dimensional samples. As shown in Fig. 3, Original represents the original distribution of bug reports, and RSMOTE, RUS, ROS, and SMOTE represent the distributions of bug reports balanced by ILS. Green dots indicate minority-class samples, and yellow dots indicate majority-class samples. From Fig. 3, we can see that the dataset balanced by the RSMOTE approach achieves better distribution in sample space, comparing with Original, SMOTE, RUS, and ROS. RUS removes some majority-class samples from original dataset; in this way, RUS tends to result in shrinking size of training dataset and missing crucial samples. In addition, ROS adds the duplicate of some minority-class samples into the original dataset, however some noise and redundant samples may be augmented to hinder the classification learning.

SMOTE tends to lead the occurrence of overlapping between categories because SMOTE generates new instances for each original minority sample without considering neighboring samples. Moreover, in SMOTE algorithm, the synthetic instance is created along a linear space, which causes the problem of under generalization for high-dimensional instance space. To generate samples in robust manner, the proposed RSMOTE conducts two improvements. One is that RSMOTE breaks the ties introduced by simple linear sampling space and therefore the new synthetic samples generated by our proposed method have a reasonable distribution in feature space of minority class instances, as shown in Fig. 3. The other is that synthetic majority class instances are eliminated in RSMOTE; in this way, a classifier could properly learn the distributive characteristics of minority class instances from the dataset balanced by RSOMTE.

From the above-mentioned description, it can be observed that the RSMOTE algorithm has the following advantages.

1) After oversampling by the RSMOTE algorithm, new synthetic examples are randomly generated in the minority-class space of the original dataset. Compared with SMOTE, RSMOTE eliminates the limitation imposed by the linear interpolation between the minority-class samples, which makes the RSMOTE be more scientific and

TABLE I
DATASET OBTAINED FROM THE BUG REPOSITORIES

Project	Product-Component	Non-severe bugs	Severe bugs	Number of Words	Imbalance Degree (M)
Eclipse	Platform-UI	1173	2982	2822	2.54
	JDT-Core	512	1315	1580	2.57
	JDT-Debug	291	706	1140	2.43
	Platform-Debug	232	404	869	1.74
	CDT-Core	114	458	817	4.02
	PDE-UI	297	791	1055	2.66
Mozilla	Core-Layout	960	2747	2967	2.86
	Core-XPCOM	149	748	1489	5.02
	Core-XUL	122	499	1178	4.09
	Core-XPConnect	40	212	681	5.3
GNOME	Evolution-Calendar	626	2896	1669	4.63
	Terminal-General	264	3346	2082	12.67
	Ekiga-General	156	1482	1349	9.5
	Evolution-Contacts	644	1788	1380	2.78
	Evolution-Shell	495	1210	1203	2.44
	Panel-Panel	330	1301	1135	3.94

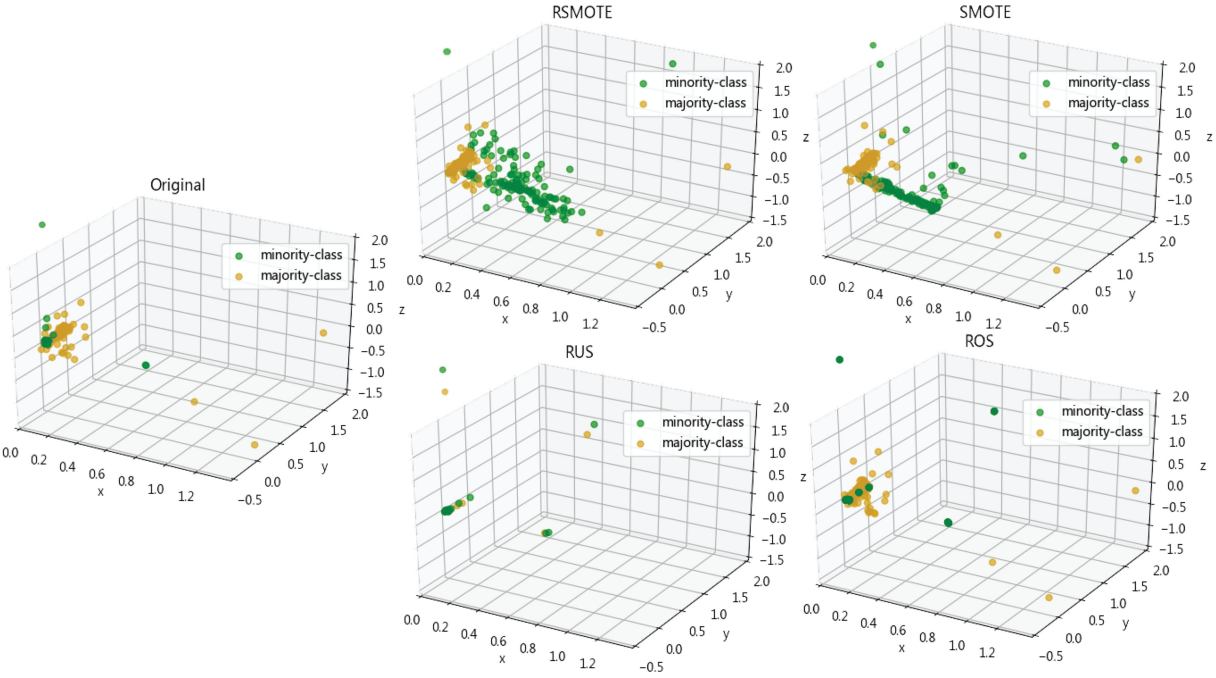


Fig. 3. Three-dimensional distribution of Core-XPConnect by using different ILS.

practical. In addition, the process of RSMOTE consists of two constraints that can provide a robust way to generate new synthetic samples.

2) Compared with RUS, the RSMOTE approach is applicable for data-driven classification learning, due to the oversampling technique keeping the size of majority class and increasing minority-class samples. Moreover, RSMOTE gets rid of the overfitting risk which often leads ROS to correspond too closely to a particular set of data.

D. Fusion of Multi-RSMOTE With Fuzzy Integral (FMR-FI)

Analogously to most of data-level approaches in imbalanced learning [6], [7], [21], [26], some occasionalities encountered in the proposed RSMOTE algorithm tend to hinder the classification learning, e.g., the replicated data from noise or redundant instances. To lessen the chance of occasionalities in synthetic sampling process of RSMOTE, a multiple sampling technique will be proposed in this section. Concretely, multiple sampling processes of RSMOTE are run on an imbalanced dataset, then different balanced datasets are generated and employed to train classifiers. Finally, an ensemble-based algorithm combines the wisdom of crowds (i.e., the trained classifiers) to make better decisions. Due to a strong interaction existing among the individual classifiers, we choose the Choquet fuzzy integral to integrate these trained classifiers.

To ease the presentation, some of notations will be established here. Given a training dataset Tr and a testing

dataset Te, we define that $Tr = \{x|x \in R^m\}$ and $Te = \{x|x \in R^m\}$, where x is an example in the *m*-dimensional feature space, and $La = \{La_1, \ldots, La_j, \ldots La_C\}$ is a set with *C*-class labels. Furthermore, we define a set of classifiers $E = \{E_1, \ldots, E_i, \ldots E_L\}$ in which each classifier is trained over a $Tr_i \in \text{sub}Trs = \{Tr_1, \ldots, Tr_i, \ldots, Tr_L\}$, *L* is the number of training datasets processed by RSMOTE. A class label from La is assigned to x by E_i whose output can be considered as a *C*-dimensional vector of support degree for each category, i.e.,

$$E_i(x) = (e_{i1}(x), e_{i2}(x), \ldots, e_{ij}(x), \ldots, e_{iC}(x)) \quad (13)$$

where $e_{ij}(x) \in [0, 1]$ $(1 \le i \le L, 1 \le j \le C)$ denotes the support degree assigned by classifier E_i that x belongs to class La_j. In this paper, $e_{ij}(x)$ is the posterior probability $p(La_j|x)$ that has the following properties, for all $j = 1, \ldots, C$:

$$e_{ij}(x) \ge 0, \sum_{j=1}^{C} e_{ij}(x) = 1. \quad (14)$$

Afterward, some of related definitions will be presented as follows.

Definition 4: Given $E = \{E_1, \ldots, E_i, \ldots E_L\}$, $La = \{La_1, \ldots, La_j, \ldots La_C\}$, and $Te = \{x|x \in R^m\}$, for each $x \in Te$, the decision profile matrix is

$$\text{DP}(x) = \begin{pmatrix} e_{11}(x) & \cdots & e_{1j}(x) & \cdots & e_{1C}(x) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ e_{i1}(x) & \cdots & e_{ij}(x) & \cdots & e_{iC}(x) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ e_{L1}(x) & \cdots & e_{Lj}(x) & \cdots & e_{LC}(x) \end{pmatrix} \quad (15)$$

where the *i*th row of *DP* represents the support degree as mentioned above, and the *j*th column of *DP* represents the support degree estimated by *E* for class La_j.

Definition 5: Given $E = \{E_1, \ldots, E_i, \ldots, E_L\}$, the power set of *E* is represented as $P(E)$. The fuzzy measure on *E* can be represented as a set function g: $P(E) \to [0, 1]$, which is shown as follows:

$$\begin{gathered} g(\emptyset) = 0, g(E) = 1. \\ \text{For } \forall A, B \subseteq E, \text{ if } A \subset B, \text{ then } g(A) \le g(B). \end{gathered} \quad (16)$$

Definition 6: Given $E = \{E_1, \ldots, E_i, \ldots, E_L\}$, $\forall E_i \in E$, $i \in [1, L]$, let $g^i = g(\{E_i\})$. g^i represents the fuzzy density of classifier E_i. We use the following to compute g^i:

$$g^i = \frac{p(E_i)}{\sum_{k=1}^{L} p(E_k)} \times d_{\text{sum}} \quad (17)$$

where $p(E_i)$ represents the validation accuracy of E_i and d_{sum} is the desired sum of fuzzy densities.

Definition 7: Given $E = \{E_1, \ldots, E_i, \ldots, E_L\}$, $A_k = \{E_1, E_2, \ldots, E_k\} \subset E$ $(1 \le k \le L)$. λ-fuzzy measure g defined on A_k could be calculated by the following formulas:

$$\begin{aligned} &g(A_1) = g(\{E_1\}) = g^1 \\ &g(\{E_k\}) = g^k \\ &g(A_k) = g^k + g(A_{k-1}) + \lambda \times g^k \times g(A_{k-1}) \end{aligned} \quad (18)$$

where $\lambda > -1$ and $\lambda \ne 0$. The value of λ can be computed by the following:

$$\lambda + 1 = \prod_{i=1}^{L} (1 + \lambda \times g^i). \quad (19)$$

Definition 8: Given $E = \{E_1, \ldots, E_i, \ldots E_L\}$, g is the fuzzy measure on *E*, the Choquet fuzzy integral of function $f : E \to [0, 1]$ with respect to g is defined as follows [62]:

$$(C)\int f dg = \sum_{i=1}^{L} (f(E_i) - f(E_{i-1})) \times g(A_{i-1}) \quad (20)$$

where $0 \le f(E_1) \le f(E_2) \le \cdots \le f(E_L) \le 1$, $f(E_0) = 0$.

The FMR-FI algorithm is composed of two phases: Training phase and integrated phase, which are described in detail as follows.

Algorithm 2 FMR-FI Algorithm.

Input:
Training set *Tr*, $x \in Te$, The number of balanced datasets (*L*)

Output:
The class label of *x*.

1: Training phase:
1) Use the RSMOTE algorithm to generate *subTr* by *Tr*;
2) Train the classifiers by *subTr*, respectively;
3) Calculate the fuzzy density g^i of the classifier E_i;
4) Calculate the λ value.

2: Integrated phase:
1) For $\forall x \in Te$, calculate the decision profile DP(x);
2) Each column of *DP* is sorted in ascending order to obtain a new decision profile matrix DP′;
3) Calculate the fuzzy measure $g(A_i)$ based on DP′;
4) Calculate $u_j(x)$ using (20).

3: **return** the class label of *x*.

In Step 1, we train the classifiers and calculate the fuzzy densities based on the classification results of each classifier.

1) We use the RSMOTE to generate *L* training subsets from *Tr*, denoted by $subTrs = \{Tr_1, \ldots, Tr_i, \ldots Tr_L\}$.
2) Then, we train a classifier E_i $(i = 1, 2, \ldots, L)$ on each Tr_i in *subTrs* to obtain a set of trained classifiers $E_b = \{E_1, E_2, \ldots, E_L\}$.
3) We calculate the fuzzy density g^i of each classifier using (17).
4) Finally, we calculate the value of λ using (19).

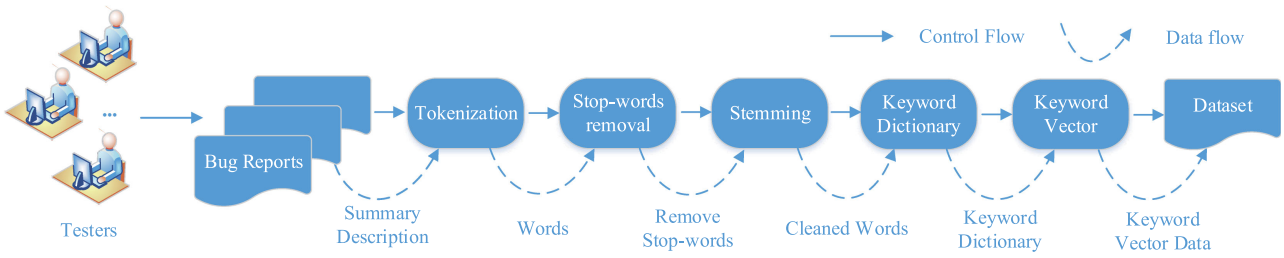


Fig. 4. Workflow of bug report processing.

In Step 2, we calculate the class label of each x in *Te* using fuzzy integrals.

1) For each x in *Te*, we can get a decision profile DP(x) using (15).
2) We sort each column of *DP* in ascending order to obtain a new decision profile matrix DP′. Then, the kth column of DP′ is $[e_{z_1 k}, e_{z_2 k}, \ldots, e_{z_L k}]^T$, where $e_{z_L k}$ is the highest support degree and $e_{z_1 k}$ is the lowest support degree. The fuzzy densities of the corresponding classifiers are denoted by $(g^{z_1}, g^{z_2}, \ldots, g^{z_L})$.
3) Then, we let $g(A_1) = g^{z_1}$ and iteratively calculate $g(A_i)$ using (18), where $i = 1, 2, \ldots, L$.
4) By calculating $u_j(x)$ using (20), we obtain $\{u_1(x), u_2(x), \ldots, u_j(x), \ldots, u_C(x)\}$, where $j = 1, 2, \ldots, C$.

In Step 3, we compute the category label j^* of each x based on the following:

$$j^* = \arg \max_{1 \le j \le C} \{u_j(x)\}. \tag{21}$$

IV. Experimental Design

Several experiments are conducted to validate the performance of FMR-FI, and the experimental design is described in this section.

A. Experimental Design

We verify the performance of FMR-FI on *Eclipse*, *Mozilla*, and *GNOME*, which all use the same bug tracking system (*Bugzilla* [19]). In this study, 16 datasets are selected from 3 bug repositories to validate the FMR-FI approach, as presented in Table I. The datasets are different from each other in the application domain. According to the results of [3] and [59], the summaries of bug reports contain not only useful information, but also a small amount of noise. Thus, we select the summaries as the features of bug reports. The average imbalance degrees of *Eclipse*, *Mozilla*, and *GNOME* are 2.66, 4.32, and 6.00, respectively. Especially for the terminal-general of *GNOME*, the imbalance degree is as high as 12.67.

In the bug repositories (*Eclipse*, *Mozilla* and *GNOME*), the severity level of bug reports is designated as *trivial*, *minor*, *normal*, *major*, *critical*, and *blocker*. As Lamkanfi *et al.* argued in [36], the *normal* severity status is a default option, thus this status tends to be ignored in related works. In our experiment, the setting of the severity level is as the same as [36], [54], and [59], in which the *nonsevere* class includes *trivial* and *minor*, and the *severe* class includes *major*, *critical*, and *blocker*.

In our study, the text preprocessing of bug reports can be summarized as the following five steps, i.e., tokenization, stop-word removal, stemming, keyword dictionary, and keyword vector, which is clearly shown in Fig. 4.

B. Experimental Setup

In our experiment, we use four well-known ILS (RUS, ROS, SMOTE, and CMA) [2] as baseline algorithms to compare with RSMOTE. In addition, we use Weka [58] to implement four popular classification algorithms [*NB*, *KNN*, *J48*, and Random Tree (*RT*)].

There are two integration approaches for the FMR-FI. One is to integrate the same classifiers, another is to fuse different classifiers. In both ways, the winners in *SubTrs* are selected as the objects to be integrated and we will present that the proposed method can further improve the performance of these selected classifiers. Moreover, we compare the ensemble performance of the FMR-FI approach with three well-known standard ensemble methods: Majority voting, bagging, and AdaBoost [40]–[44].

Stratified three-fold cross validation is applied in our experiment, which could keep the distributive characteristics during each training iteration [22], [23]. In experimental part, k represents the number of nearest neighbor minority-class samples for each sampling center point. Due to lacking approach to optimize this value, as most related work [2], k is an empirical value. In our study, k is set to 5. In addition, N is used to control the number of new synthetic minority-class samples. N is calculated by the imbalance degree (M), which can be expressed as N=round(M)-1, where round(M) denotes an approximate integer to M and the M of each dataset is shown in Table I. RSMOTE runs oversampling process N times to balance the class distribution.

C. Evaluation Metrics

In our study, four evaluation metrics (accuracy, *precision*, *recall*, and the *F-measure*) is used to evaluate the performance of FMR-FI [25]. The four evaluation metrics can be computed by the confusion matrix, as presented in Table II.

1) *Accuracy:* The accuracy represents the proportion of bug reports correctly classified to the total number of bug reports

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}. \tag{22}$$

TABLE II
CONFUSION MATRIX, WHICH CAN BE USED TO CALCULATE THE EVALUATION METRICS

Confusion Matrix		Actual Severity	
		non-severe	severe
Predicted Severity	non-severe	TP: true positives	FP: false positives
	severe	FN: false negatives	TN: true negatives

2) *Precision:* The *precision* represents the proportion of all bug reports that are predicted to be either *nonsevere* or *severe* and are actually *nonsevere* or *severe*, respectively,

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{23}$$

3) *Recall:* The *recall* represents the proportion of all bug reports that are actually *nonsevere* or *severe* and are correctly predicted to be *nonsevere* or *severe*, respectively,

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{24}$$

4) *F-measure: F-measure* represents the balance and discrepancy between *precision* and *recall*, which can be computed using the *precision* and *recall*. The *F-measure* has a property whereby if either the *precision* or *recall* is low, the *F-measure* also decreases

$$F - \text{measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{25}$$

V. EXPERIMENTAL RESULTS

In this section, we discuss the the specific research questions based on the experimental results.

RQ1: Which strategy is the best? We compare RSMOTE with the other ILS in terms of the severity prediction performance for bug reports with an imbalanced severity distribution.

In this experiment, we compared the results of RSMOTE approach with the original bug reports and the results of RUS, ROS, SMOTE, and CMA, as shown in Tables III, IV, S-I–S-IV (see Supplement). Afterward, for original datasets and each dataset balanced by ILS (RUS, ROS, CMA, SMOTE, and RSMOTE), we used four classifiers (*NB*, *KNN*, *FT*, and *J48*) to predict the severity of bug reports and evaluated their performances. Altogether, the six ILS and four classification algorithms considered here yielded a total of 24 variants (i.e., combinations of one of the ILS and one of the classification algorithms). Therefore, to address this first research question, we wished to investigate which variant has the best performance for identifying the severity of bug reports. We used the accuracy and *F-measure* as evaluation metrics to compare all 24 variants. The detailed results of *Eclipse* and *GNOME* (i.e., accuracy and *F-measure* values) are shown in "supplementary.pdf,"[1] where we retain the original number and name of the tables. From all results in these tables, we can draw several conclusions in the following.

[1]https://github.com/swordlin/FMR-FI

In Tables III, S-I, S-II, we compare the accuracy of classifying the severity of bug reports characterized by imbalanced distributions. With RSMOTE, the classifiers can achieve the highest maximum accuracy in predicting the severity of bug reports. As shown in Table III, the maximum accuracies of RSMOTE for four *Mozilla* components are 86.85%, 91.67%, 73.44%, and 84.54%. Besides, the maximum classification accuracy achieved with RSMOTE for *Mozilla* is higher than those achieved with the others, i.e., Original, RUS, ROS, CMA, and SMOTE, the increments are 5.01%, 13.90%, 3.36%, 2.41%, and 3.57%, respectively. As shown in the AVG_ACC columns in Tables III, S-I, S-II, the RSMOTE approach can also yield a better average accuracy than the other ILS. In Table III, the average accuracy of RSMOTE is also higher than original dataset and other ILS (RUS, ROS, CMA, and SMOTE), the increments are 4.51%, 26.72%, 8.14%, 3.10%, and 3.96%, respectively.

When classifying bug reports characterized by an imbalanced distribution, a classification algorithm may be prone to the majority category. Therefore, its classification performance cannot be objectively reflected by the classification accuracy [26]. In this experiment, we compared the classification effect (*F-measure*) achieved in bug report severity prediction for each component from the *Eclipse*, *Mozilla,* and *GNOME* projects, as shown in Tables IV, S-III, S-IV (see Supplement).

In Tables IV, S-III, S-IV, we compare the performance of the RSMOTE approach with the performances of other ILS when predicting the severity of bug reports following imbalanced distributions. As shown in the MAX_F columns of Table IV, the maximum *F-measures* produced by RSMOTE are higher than that of other ILS (RUS, ROS, CMA, and SMOTE). For example, in Table III, the average *F-measure* of RSMOTE is in excess of those of Original, RUS, ROS, CMA, and SMOTE, and the increments are 5.32%, 20.13%, 6.95%, 3.31%, and 3.92%, respectively.

These experiments suggest that the RSMOTE approach can effectively balance bug report datasets, thereby improving the performance of classifiers for bug report severity prediction. We also observe that the performance predicting the severity of *Mozilla* bug reports is higher than that for *Eclipse* bug reports, while the performance on *GNOME* bug reports is the best. In regard to the average classification performance for predicting the severity of bug reports, the *NB* classifier with the RSMOTE approach is the most suitable for predicting the severity of bug reports from *Eclipse* and *Mozilla*, whereas the *KNN* classifier with the RSMOTE approach is the most suitable for predicting the severity of bug reports for the *GNOME* bug repository. In general, for individual software components, different classification variants achieve different performances in predicting the severity of bug reports. Thus, in the following experimental part, we use the variant with the best performance as a baseline to compare the performance of our proposed approach.

RQ2: Can the fuzzy integral approach improve the stability of RSMOTE when predicting the severity of bug reports characterized by an imbalanced distribution?

As discussed in RQ1, RSMOTE can effectively alter the size of the bug report datasets and provide the same proportion of

TABLE III
ACCURACY OF RSMOTE TO PREDICT THE SEVERITY OF *Mozilla*

		NB	KNN	RT	J48	MAX_ACC	AVG_ACC
Moizlla_Core_XPCOM	Original	67.89	82.94	77.93	**83.28**	83.28	78.01
	RUS	**69.57**	63.55	58.19	52.17	69.57	60.87
	ROS	66.89	**82.61**	77.93	58.19	82.61	71.41
	CMA	82.94	73.58	85.62	**86.62**	86.62	82.19
	SMOTE	81.61	77.59	78.26	**83.95**	83.95	80.35
	RSMOTE	84.62	76.59	85.06	**86.85**	**86.85**	**83.28**
		NB	KNN	RT	J48	MAX_ACC	AVG_ACC
Moizlla_Core_XPConnect	Original	75.00	51.19	**85.71**	84.52	85.71	74.11
	RUS	**83.33**	44.05	61.90	64.29	83.33	63.39
	ROS	75.00	50.00	86.90	**89.29**	89.29	75.30
	CMA	**89.29**	33.33	85.71	**89.29**	89.29	74.41
	SMOTE	85.71	40.48	86.90	**88.10**	88.10	75.30
	RSMOTE	86.90	41.67	86.51	**91.67**	**91.67**	**76.69**
		NB	KNN	RT	J48	MAX_ACC	AVG_ACC
Mozilla_Core_Layout	Original	69.82	**71.76**	**71.76**	69.58	71.76	70.73
	RUS	**69.58**	66.50	64.56	59.06	69.58	64.93
	ROS	70.06	**71.52**	70.15	68.28	71.52	70.00
	CMA	**71.52**	55.10	70.71	70.15	71.52	66.87
	SMOTE	**73.14**	65.13	71.36	72.33	73.14	70.49
	RSMOTE	72.60	70.16	71.17	**73.44**	**73.44**	**71.84**
		NB	KNN	RT	J48	MAX_ACC	AVG_ACC
Moizlla_Core_XUL	Original	74.40	74.06	76.33	**79.71**	79.71	76.13
	RUS	**72.95**	43.00	48.31	65.22	72.95	57.37
	ROS	73.91	**82.13**	71.01	61.84	82.13	72.22
	CMA	80.68	**81.16**	77.29	79.23	81.16	79.59
	SMOTE	76.81	62.80	78.26	79.71	79.71	74.40
	RSMOTE	79.07	**84.54**	79.71	79.23	**84.54**	**80.64**
		NB	KNN	RT	J48	MAX_ACC	AVG_ACC
Moizlla_ALL	AVG.	**76.39**	**64.39**	**75.30**	**75.67**	**80.48**	**72.94**

TABLE IV
F-measure OF RSMOTE TO PREDICT THE SEVERITY OF *Mozilla*

		NB	KNN	RT	J48	MAX_F	AVG_F
Moizlla_Core_XPCOM	Original	0.72	**0.78**	0.76	**0.78**	0.78	0.76
	RUS	**0.73**	0.68	0.63	0.57	0.73	0.65
	ROS	0.71	**0.78**	0.75	0.64	0.78	0.72
	CMA	0.81	0.77	0.82	**0.84**	0.84	0.81
	SMOTE	**0.80**	**0.80**	**0.80**	**0.80**	0.80	0.80
	RSMOTE	0.84	0.79	**0.85**	**0.85**	**0.85**	**0.83**
		NB	KNN	RT	J48	MAX_F	AVG_F
Moizlla_Core_XPConnect	Original	0.78	0.56	**0.82**	0.77	0.82	0.73
	RUS	**0.85**	0.48	0.67	0.69	0.85	0.67
	ROS	0.78	0.55	0.84	**0.89**	0.89	0.77
	CMA	**0.90**	0.34	0.86	0.89	0.90	0.75
	SMOTE	0.86	0.44	0.86	**0.88**	0.88	0.76
	RSMOTE	0.88	0.45	0.86	**0.91**	**0.91**	**0.78**
		NB	KNN	RT	J48	MAX_F	AVG_F
Mozilla_Core_Layout	Original	**0.71**	**0.71**	**0.71**	**0.71**	0.71	0.71
	RUS	**0.71**	0.69	0.67	0.61	0.71	0.67
	ROS	**0.71**	0.70	0.69	0.70	0.71	0.70
	CMA	**0.72**	0.57	0.71	0.70	0.72	0.68
	SMOTE	**0.72**	0.67	0.71	**0.72**	0.72	0.71
	RSMOTE	0.71	0.72	0.72	**0.73**	**0.73**	**0.72**
		NB	KNN	RT	J48	MAX_F	AVG_F
Moizlla_Core_XUL	Original	0.76	**0.82**	0.73	0.72	0.82	0.76
	RUS	**0.74**	0.45	0.52	0.69	0.74	0.60
	ROS	0.75	**0.83**	0.69	0.65	**0.83**	0.73
	CMA	0.79	**0.81**	0.78	0.76	0.81	0.79
	SMOTE	0.75	0.67	**0.77**	0.75	0.77	0.74
	RSMOTE	0.78	**0.83**	0.79	0.76	**0.83**	**0.79**
		NB	KNN	RT	J48	MAX_F	AVG_F
Moizlla_ALL	AVG.	**0.77**	**0.66**	**0.75**	**0.75**	**0.80**	**0.73**

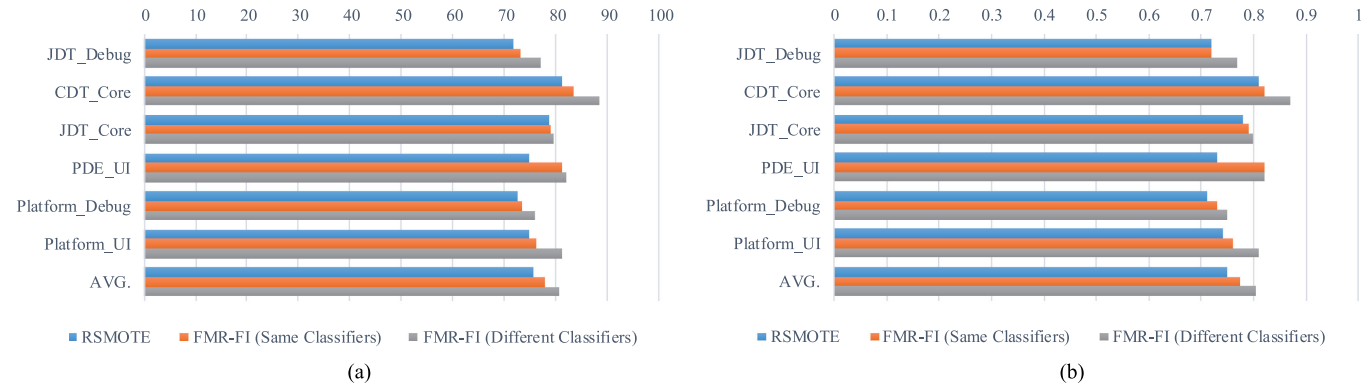


Fig. 5. Performance of predicting the severity of *Eclipse* bug reports. (a) Accuracy (b) F-measure.

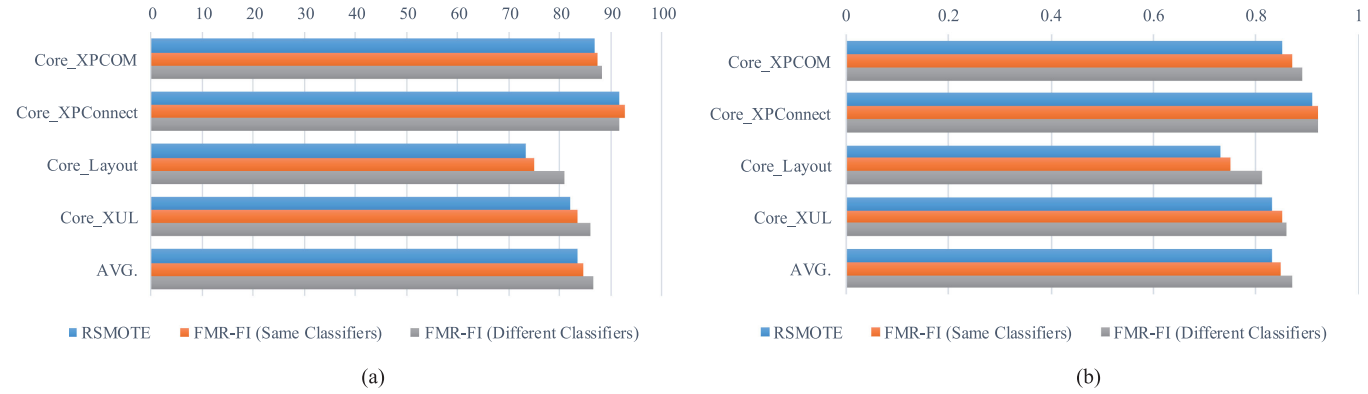


Fig. 6. Performance of predicting the severity of *Mozilla* bug reports. (a) Accuracy (b) F-measure.

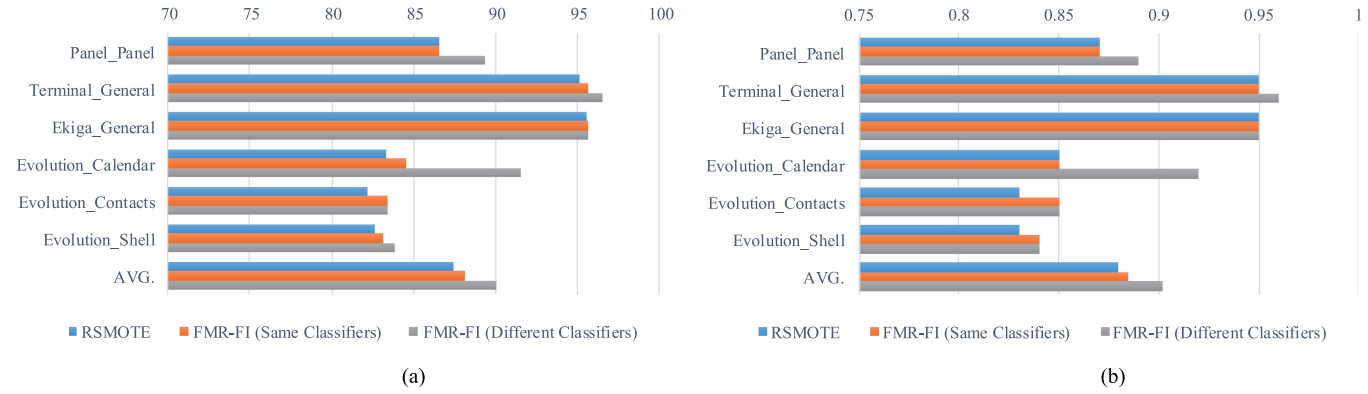


Fig. 7. Performance of predicting the severity of *GNOME* bug reports. (a) Accuracy (b) F-measure.

balance. In this research study, the evaluation metrics (namely, accuracy, and *F-measure*) are used to verify the stability of the approach combining fuzzy integral and RSMOTE. As shown in the experimental results, the fusion method could improve the stability of RSMOTE in most cases.

As shown in Figs. 5–7, the performances achieved by using the FMR-FI approach in integrating the different classifiers are better than those achieved by integrating the same classifiers and are better than the results achieved by using RSMOTE alone. In Fig. 5, the average accuracies achieved by using the FMR-FI approach for integrating different classifiers to classify the severity of bug reports for six *Eclipse* components are higher than those achieved by using RSMOTE alone; the increments are 7.71%, 9.03%, 1.18%, 9.82%, 4.51%, and 8.26%, respectively. The corresponding improvements of the average *F-measure* are 6.94%, 7.41%, 2.56%, 12.33%, 5.63%, and 9.46%, respectively. In Fig. 6, the average accuracies achieved by using the FMR-FI approach for integrating different classifiers to classify the severity of bug reports for four *Mozilla* components are higher than those achieved by using RSMOTE alone; the increments

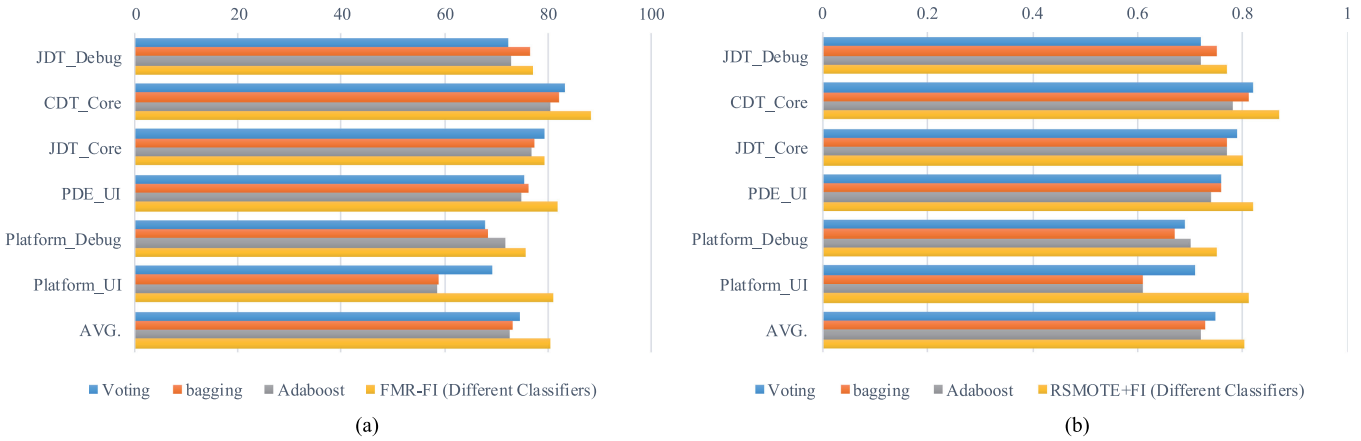


Fig. 8. Performance of predicting the severity of *Eclipse* bug reports. (a) Accuracy (b) F-measure.

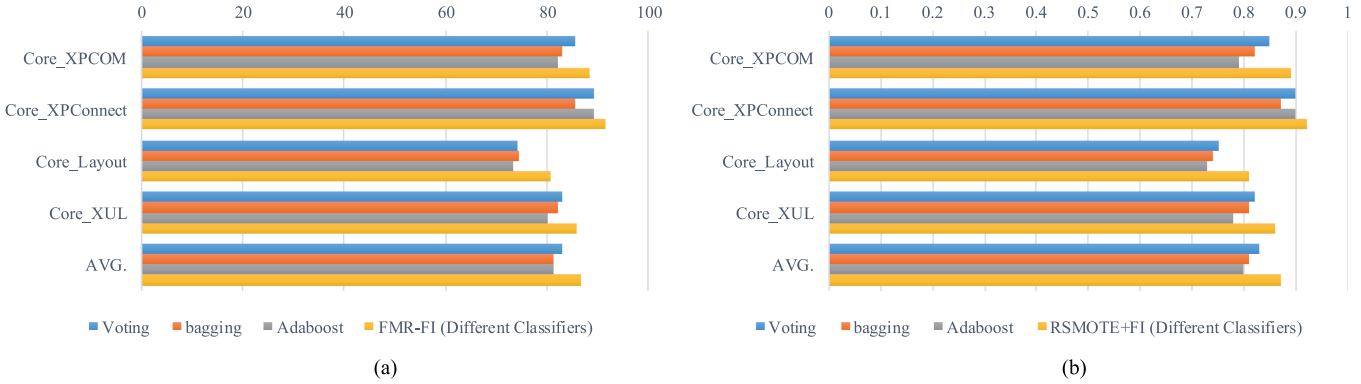


Fig. 9. Performance of predicting the severity of *Mozilla* bug reports. (a) Accuracy (b) F-measure.

are 1.66%, 0%, 10.06%, and 4.70%, respectively. The corresponding increments of average *F-measure* are 4.71%, 1.10%, 10.96%, and 3.61%, respectively. In Fig. 7, the average accuracies achieved by using the FMR-FI approach for integrating different classifiers to classify the severity of bug reports for six *GNOME* components are higher than those achieved by using RSMOTE alone; the increments are 3.26%, 1.43%, 0.06%, 10.90%, 1.48%, and 1.42%, respectively. The corresponding improvements of average *F-measure* are 2.30%, 1.05%, 0%, 8.24%, 2.41%, and 12.05%, respectively.

Thus, these experiments show that the FMR-FI approach for integrating different classifiers can provide reliable performance in classifying the severity of bug reports in the *Eclipse*, *Mozilla,* and *GNOME* bug repositories. This improvement in performance can be attributed to two factors. One factor is that the fusion of multi-RSMOTE with the fuzzy integral approach weakens the occasionality caused by random sampling process and improves the generalization ability of the RSMOTE approach. The other factor is that the FMR-FI approach for integrating different classifiers can complement the classification performance of the classifiers, resulting in a higher overall performance than that of individual classifiers. In addition, the performance improvement in classifying the severity of bug reports in the *Eclipse* bug repository using FMI-FI is higher than that for *Mozilla*, and the performance improvement for *Mozilla* is higher than that for *GNOME*.

RQ3: Can the FMR-FI approach outperform state-of-the-art approaches?

In order to demonstrate the superiority of the FMR-FI approach, in this experimental part, the proposed FMR-FI approach is compared with three popular classifier ensemble approaches (namely, voting, bagging, and AdaBoost). Two evaluation indexes (i.e., accuracy and *F-measure*) are used to evaluate the performance of fusion of multiclassifiers to predict the class label of bug reports. The accuracy and *F-measure* are shown in Figs. 8–10 and the performance of the FMR-FI is better than that of voting, bagging, and AdaBoost approaches on all datasets. Fig. 8 shows the performance in classifying the severity of *Eclipse* bug reports. The average accuracies are 8.16%, 10.03%, and 11.04% higher than that of voting, bagging, and AdaBoost, respectively, and the average *F-measure* are 7.35%, 10.30%, and 11.57% higher than that of other ensemble methods, respectively. Fig. 9 shows the performance in classifying the severity of *Mozilla* bug reports. The average accuracies are 4.39%, 6.58%, and 6.63% higher than that of voting, bagging, and AdaBoost, respectively. And the average *F-measure* are 4.82%, 7.41%, and 8.75% higher than that of other ensemble methods, respectively. Fig. 10 shows the performance in clas-

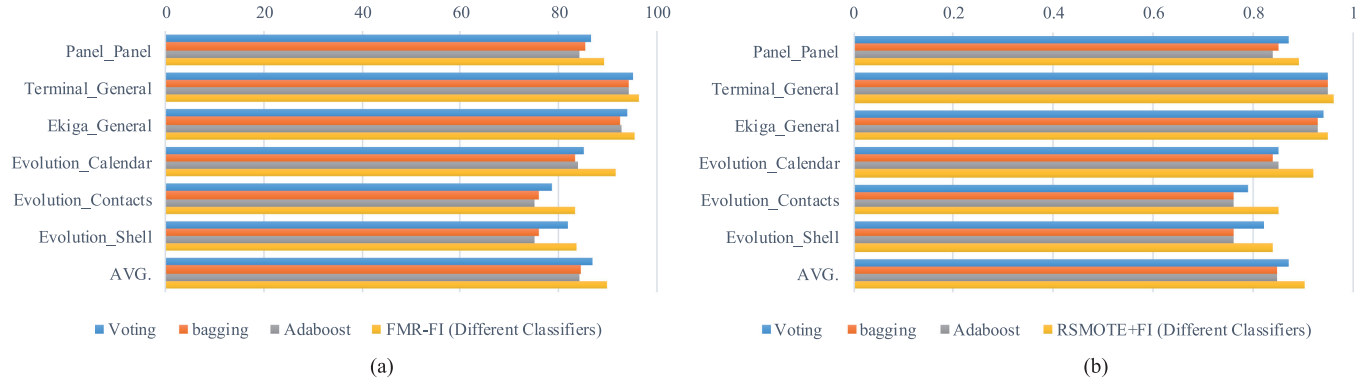


Fig. 10. Performance of predicting the severity of *GNOME* bug reports. (a) Accuracy (b) F-measure.

sifying the severity of *GNOME* bug reports. The average accuracies are 3.53%, 6.33%, and 6.76% higher than that of voting, bagging and AdaBoost, respectively and the average *F-measure* are 3.65%, 6.29%, and 6.29% higher than that of other ensemble methods, respectively.

We also could find that the performance of the FMR-FI approach in classifying the severity of *GNOME* bug reports is higher than that for *Mozilla* bug reports and is higher than that for *Eclipse* bug reports. These experiments also show that for all datasets from the *Eclipse*, *Mozilla,* and *GNOME* bug repositories, the FMR-FI method leads to a better performance than the three widely used classifier ensemble approaches (namely, voting, bagging, and AdaBoost). In addition, the classification performance of the voting approach is generally better than that of the bagging and AdaBoost approaches for classifying the severity of the *Eclipse*, *Mozilla,* and *GNOME* bug reports.

VI. Conclusion and Future Work

In this study, we propose a method to fuse the results of classifiers via a Choquet fuzzy integral to boost the performance for predicting the class label of bug reports with class imbalance. First, we propose an RSMOTE method to alter the size of the bug report datasets. Then, we build several classifiers over different, but related training datasets generated via RSMOTE. Finally, the trained classifiers are integrated by Choquet fuzzy integral to obtain the ultimate prediction results. Several experiments are conducted on 16 datasets from *Eclipse*, *Mozilla*, and *GNOME*. The experimental results statistically demonstrate that FMR-FI can effectively improve the classification performance for severity prediction.

In the future work, we plan to apply the FMR-FI approach to cover more software projects, especially the industrial projects, so as to demonstrate an even broader applicability of this method. We also plan to research an improved synthetic sampling approach for imbalanced learning.

References

[1] X. Xia, D. Lo, X. Wang, and B. Zhou, "Accurate developer recommendation for bug resolution," in *Proc. 20th Work. Conf. Reverse Eng.*, 2013, pp. 72–81.

[2] X. Yang, D. Lo, X. Xia, Q. Huang, and J.-L. Sun, "High-impact bug report identification with imbalanced learning strategies," *J. Comput. Sci. Technol.*, vol. 32, no. 1, pp. 181–198, 2017.

[3] A. Lamkanfi, S. Demeyer, E. Giger, and B. Goethals, "Predicting the severity of a reported bug," in *Proc. 7th IEEE Work. Conf. Mining Softw. Repositories*, May 2010, pp. 1–10.

[4] E. Shihab, A. Mockus, Y. Kamei, B. Adams, and A. E. Hassan, "High impact defects: A study of breakage and surprise defects," in *Proc. 19th ACM SIGSOFT Symp. 13th Eur. Conf. Found. Softw. Eng.*, 2011, pp. 300–310.

[5] M. Sugeno, "Fuzzy measures and fuzzy integrals-A survey," in *Fuzzy Automata and Decision Processes*, M. M. Gupta, G. N Saridis, and B. R. Gaines, Eds., Amsterdam, The Netherlands: Elsevier, 1977, pp. 89–102, .

[6] J. Zhai, S. Zhang, and C. Wang, "The classification of imbalanced large data sets based on mapreduce and ensemble of ELM classifiers," *Int. J. Mach. Learn. Cybern.*, vol. 8, no. 3, pp. 1009–1017, 2017.

[7] D. A. Cieslak and N. V. Chawla, "Learning decision trees for unbalanced data," in *Machine Learning and Knowledge Discovery in Databases*. Antwerp, Belgium: Springer, 2008, pp. 241–256.

[8] J. Zhang and I. Mani, "KNN approach to unbalanced data distributions: A case study involving information extraction," in *Proc. Int. Conf. Mach. Learn., Workshop: Learn. Imbalanced Data Sets*, 2003, pp. 42–48.

[9] A. K. Das, "Weighted fuzzy soft multiset and decision-making," *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 5, pp. 787–794, 2018.

[10] Y. Feng, Z. Chen, J. A. Jones, C. Fang, and B. Xu, "Test report prioritization to assist crowdsourced testing," in *Proc. 10th Joint Meeting Found. Softw. Eng.*, 2015, pp. 225–236.

[11] C. Wang, Q. He, M. Shao, and Q. Hu, "Feature selection based on maximal neighborhood discernibility," *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 11, pp. 1929–1940, 2018.

[12] M. F. Porter, "An algorithm for suffix stripping," *Program*, vol. 14, no. 3, pp. 130–137, 1980.

[13] C. Yue, "Normalized projection approach to group decision-making with hybrid decision information," *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 8, pp. 1365–1375, 2018.

[14] T. Menzies and A. Marcus, "Automated severity assessment of software defect reports," in *Proc. IEEE Int. Conf. Softw. Maintenance*, 2008, pp. 346–355.

[15] J. Zhai, H. Xu, and Y. Li, "Fusion of extreme learning machine with fuzzy integral," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 21, pp. 23–34, 2013.

[16] Eclipse. (2018). [Online]. Available: http://bugs.eclipse.org/bugs

[17] Mozilla. (2018). [Online]. Available: http://bugzilla.mozilla.org

[18] GNOME. (2018). [Online]. Available: http://bugzilla.gnome.org

[19] Bugzilla. (2018). [Online]. Available: https://www.bugzilla.org/

[20] J. Zhai, L. Zang, and Z. Zhou, "Ensemble dropout extreme learning machine via fuzzy integral for data classification," *Neurocomputing*, vol. 275, pp. 1043–1052, 2018.

[21] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.

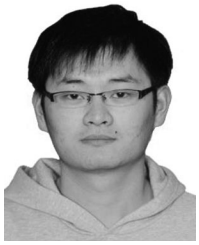
[22] W. Deng, H. Zhao, L. Zou, G. Li, X. Yang, and D. Wu, "A novel collaborative optimization algorithm in solving complex optimization problems," *Soft Comput.*, vol. 21, no. 15, pp. 4387–4398, 2017.

[23] X. Xia, D. Lo, X. Wang, and B. Zhou, "Tag recommendation in software information sites," in *Proc. 10th Work. Conf. Mining Softw. Repositories*, 2013, pp. 287–296.
[24] G. Wei, F. E. Alsaadi, T. Hayat, and A. Alsaedi, "Projection models for multiple attribute decision making with picture fuzzy information," *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 4, pp. 713–719, 2018.
[25] T. Menzies and A. Marcus, "Automated severity assessment of software defect reports," in *Proc. IEEE Int. Conf. Softw. Maintenance*, 2008, pp. 346–355.
[26] C. G. Weng and J. Poon, "A new evaluation measure for imbalanced datasets," in *Proc. 7th Australas. Data Mining Conf.*, vol. 87, pp. 27–32, 2008.
[27] W. Deng, S. Zhang, H. Zhao, and X. Yang, "A novel fault diagnosis method based on integrating empirical wavelet transform and fuzzy entropy for motor bearing," *IEEE Access*, vol. 6, pp. 35042–35056, 2018.
[28] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. Philip Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, 2002.
[29] C. Zhang, D. Li, and J. Liang, "Hesitant fuzzy linguistic rough set over two universes model and its applications," *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 4, pp. 577–588, 2018.
[30] G. Antoniol, K. Ayari, M. Di Penta, F. Khomh, and Y.-G. Gueheneuc, "Is it a bug or an enhancement?: A text based approach to classify change requests," *Proc. Conf. Center Adv. Stud. Collaborative Res.*, 2008, pp. 304–318.
[31] S. Guo, R. Chen, and H. Li, "Using knowledge transfer and rough set to predict the severity of android test reports via text mining," *Symmetry*, vol. 9, no. 8, p. 161, 2017.
[32] J. Xuan, H. Jiang, H. Zhang, and Z. Ren, "Developer recommendation on bug commenting: A ranking approach for the developer crowd," *Sci. China Inf. Sci.*, vol. 60, no. 7, pp. 072105:1–072105:18, 2017.
[33] J. Anvik and G. C. Murphy, "Reducing the effort of bug report triage: Recommenders for development oriented decisions," *ACM Trans. Softw. Eng. Methodology*, vol. 20, no. 3, p. 10, 2011.
[34] Y. Tian, D. Lo, X. Xia, and C. Sun, "Automated prediction of bug report priority using multi-factor analysis," *Empirical Softw. Eng.*, vol. 20, no. 5, pp. 1354–1383, 2015.
[35] C. Sun, D. Lo, and S.-C. Khoo, "Towards more accurate retrieval of duplicare bug reports," in *Proc. 26th IEEE/ACM Int. Conf. Automated Soft. Eng.*, 2011, pp. 253–262.
[36] A. Lamkanfi, S. Demeyer, Q. David Soetens, and T. Verdonck, "Comparing mining algorithms for predicting the severity of a reported bug," in *Proc. Eur. Conf. Soft. Maintenance Reengineering*, 2011, pp. 249–258.
[37] X. Xia, D. Lo, E. Shihab, X. Wang, and X. Yang, "ELBlocker: Predicting blocking bugs with ensemble imbalance learning," *Inf. Softw. Technol.*, vol. 61, pp. 93–106, 2015.
[38] S. Shahnewaz Ali, T. Howlader, and S. M. M. Rahman, "Pooled shrinkage estimator for quadratic discriminant classifier: An analysis for small sample sizes in face recognition," *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 3, pp. 507–522, 2018.
[39] J. Gopal, A. K. Sangaiah, A. Basu, and X. Z. Gao, "Integration of fuzzy DEMATEL and FMCDM approach for evaluating knowledge transfer effectiveness with reference to GSD project outcome," *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 2, pp. 225–241, 2018.
[40] W. Deng, H. Zhao, X. Yang, J. Xiong, M. Sun, and Bo Li, "Study on an improved adaptive PSO algorithm for solving multi-objective gate assignment," *Appl. Soft Comput.*, vol. 59, pp. 288–302, 2017.
[41] L. Rokach, "Ensemble-based classifiers," *Artif. Intell. Rev.*, vol. 33, pp. 1–39, 2010.
[42] G. Wang, J. Sun, J. Ma, K. Xu, and J. Gu, "Sentiment classification: The contribution of ensemble learning," *Decis. Support Syst.*, vol. 57, pp. 77–93, 2010.
[43] W. Deng, R. Yao, H. Zhao, X. Yang, and G. Li, "A novel intelligent diagnosis method using optimal LS-SVM with improved PSO algorithm," *Soft Comput.*, pp. 1–18, 2017, doi: 10.1007/s00500-017-2940-9.
[44] S. B. Kotsiantis and D. Kanellopoulos, "Combining bagging, boosting and dagging for classification problems," in *Proc. Knowl.-Based Intell. Inf. Eng. Syst. 17th Italian Workshop Neural Netw. Proc. 11th Int. Conf.*, 2007, pp. 493–500.
[45] A. Lamkanfi, S. Demeyer, Q. David Soetens, and T. Verdonck, "Comparing mining algorithms for predicting the severity of a reported bug," in *Proc. 15th Eur. Conf. Softw. Maintenance Reengineering*, 2011, pp. 249–258.
[46] L. Zheng, H. Wang, and S. Gao, "Sentimental feature selection for sentiment analysis of Chinese online reviews," *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 1, pp. 75–84, 2018.
[47] R. A. R. Ashfaq and X.-Z. Wang, "Impact of fuzziness categorization on divide and conquer strategy for instance selection," *J. Intell. Fuzzy Syst.*, vol. 33, no. 3, pp. 1007–1018, 2017.
[48] X.-Z. Wang, R. Aamir, and A.-M. Fu, "Fuzziness based sample categorization for classifier performance improvement," *J. Intell. Fuzzy Syst.*, vol. 29, no. 3, pp. 1185–1196, 2015.
[49] JIRA. (2018). [Online]. Available:https://www.atlassian.com/software/jira/
[50] Mantis. (2018). [Online]. Available: https://www.mantisbt.org/
[51] S. Morigi, L. Reichel, and F. Sgallari, "A truncated projected SVD method for linear discrete ill-posed problems," *Numer. Algorithms*, vol. 43, no. 3, pp. 197–213, 2006.
[52] N. Halko, P.-G. Martinsson, and J. A. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM Rev.*, vol. 53, no. 2, pp. 217–288, 2011.
[53] R. Rehurek, "Fast and faster: A comparison of two streamed matrix decomposition algorithms," 2011, arXiv:1102.5597.
[54] I. Herraiz, D. German, J. Gonzalez-Barahona, and G. Robles, "Towards a simplification of the bug report form in eclipse," in *Proc. 5th Int. Work. Conf. Mining Softw. Repositories*, May 2008, pp. 145–148.
[55] Y. Feng, J. A. Jones, Z. Chen, and C. Fang, "Multi-objective test report prioritization using image understanding," in *Proc. 31st IEEE/ACM Int. Conf. Automated Softw. Eng.*, 2016, pp. 202–213.
[56] J. Wang, S. Wang, Q. Cui, and Q. Wang, "Local-based active classification of test report to assist crowdsourced testing," in *Proc. 31st IEEE/ACM Int. Conf. Automated Softw. Eng.*, 2016, pp. 190–201.
[57] J. Wang, Q. Cui, Q. Wang, and S. Wang, "Towards effectively test report classification to assist crowdsourced testing," in *Proc. ACM/IEEE Int. Symp. Empirical Softw. Eng. Meas.*, vol. 6, no. 1, 2016, pp. 6–10.
[58] Weka. (2018). [Online]. Available: http://www.cs.waikato.ac.nz/ml/weka/
[59] T. Zhang, J. Chen, G. Yang, B. Lee, and X. Luo, "Towards more accurate severity prediction and fixer recommendation of software bugs," *J. Syst. Softw.*, vol. 117, pp. 166–184, 2016.
[60] H. Jiang *et al.*, "ROSF: Leveraging information retrieval and supervised learning for recommending code snippets," *IEEE Trans. Services Comput.*, vol. 12, no. 1, pp. 34–46, Jan./Feb. 2019, doi: 10.1109/TSC.2016.2592909.
[61] S. Naganjaneyulu, M. Rao Kuppa, and A. M. Mahmood, "An efficient wrapper approach for class imbalance learning using intelligent undersampling," *Int. J. Artif. Intell. Appl. Smart Devices*, vol. 2, no. 1, pp. 23–40, 2014.
[62] Z. Wang and G. Klir, *Fuzzy Measure Theory.* New York, NY, USA: Plenum, 1992.

Rong Chen (M'10) received the M.S. and Ph.D. degrees in computer software and theory from the Jilin University, Changchun, China, in 1997 and 2000.

He is currently a Professor with the College of Information Science and Technology at the Dalian Maritime University, Dalian, China, and has previously held position at Sun Yat-sen University, Guangzhou, China. His research interests include software diagnosis, collective intelligence, activity recognition, Internet and mobile computing.

Dr. Chen is a member of the Association for Computing Machinery (ACM).



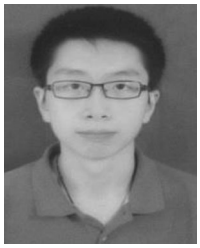
Shi-Kai Guo received the B.Sc. degree in computer science, in 2012, from the Information Science and Technology College, Dalian Maritime University, Dalian, China, where he is currently working toward the Ph.D. degree in computer science and technology.

His research interests include mining software repositories, search-based software engineering, fuzzy measures and integrals, and imbalance learning from big data.

Xi-Zhao Wang (M'03–SM'04–F'12) received the Doctoral degree in computer science from the Harbin Institute of Technology, Harbin, China, in 1998.

From 1998 to 2001, he was a Research Fellow with the Department of Computing, Hong Kong Polytechnic University, Hong Kong. From 2001 to 2014, he has been a Full Professor and the Dean of the College of Mathematics and Computer Science, Hebei University, Hebei, China. Since 2014, he has been a Full Professor with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His current research interests include supervised and unsupervised learning, active learning, reinforcement learning, manifold learning, transfer learning, unstructured learning, uncertainty, fuzzy sets and systems, fuzzy measures and integrals, rough set, and learning from big data.

Dr. Wang is a Fellow of the CAAI. He was the recipient of many awards from the IEEE International Conference on Systems, Man, and Cybernetics (SMC) Society. He is a member of the Board of Governors of the IEEE SMC in 2005, from 2007 to 2009, and from 2012 to 2014, the Chair of the Technical Committee on Computational Intelligence of the IEEE SMC, and a Distinguished Lecturer of the IEEE SMC. He was the Program Cochair of the IEEE SMC 2009 and 2010. He is the Editor-in-Chief of the *International Journal of Machine Learning and Cybernetics*. He is also an Associate Editor for the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS-PART B, *Information Sciences Journal*, and the *International Journal of Pattern Recognition and Artificial Intelligence*.



Tian-Lun Zhang received the B.Sc. degrees in information management and information system, and the M.Sc. degree in software engineering from Hebei University, Hebei, China, in 2014, and 2016. He is currently working toward the Ph.D. degree in computer science and technology from the Information Science and Technology College, Dalian Maritime University, Dalian, China.

His current research interests include fuzzy measures and integrals, computer vision, and imbalance learning from big data.