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 an 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 over-sampling techniques.**

***Index Terms*—class imbalance, fuzzy integral, bug report identification, 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, 2]. To insure the reliability of software systems, accurate recognition of bug reports has become increasingly prominent. In bug triaging systems (e.g., Bugzilla [3], JIRA [4], and Mantis [5]), 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 [6].

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 [6, 7, 8, 9]. To solve this problem, A. Lamkanfi et al.[7] 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.[6] employed four imbalanced learning strategies (ILS) (i.e., random under-sampling (RUS), random over-sampling (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 shown in [6], there exist inherent drawbacks in these imbalanced learnings. CMA is sensitive to noise data [10]; RUS tends to miss some potentially crucial data and lead to under-fitting issues; ROS often causes over-fitting because some redundant data may be selected to augment original dataset [11, 12, 13, 14]; in addition, SMOTE suffers from a poor generalization ability due to its simple linear sampling space [15]. 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. rectangle SMOTE (RSMOTE) approach, is used to weaken the imbalance ratio by generating minority-class samples, which are randomly synthesized in a multi-dimensional rectangle sample space. In addition, with two proposed constraints, the synthetic minority-class bug reports can be generated in a robust way. Secondly, to avoid the uncertainty caused by random over-sampling, an repeated sampling technique

This work is supported by the National Natural Science Foundation of China under Grant 61672122, Grant 61602077, Grant 61772344 and Grant 61732011, the Public Welfare Funds for Scientific Research of Liaoning Province of China under Grant 20170005, the Natural Science Foundation of Liaoning Province of China under Grant 20170540097, and the Fundamental Research Funds for the Central Universities under Grant 3132016348. (* *Corresponding author: Xi-Zhao Wang*.)

R. Chen, S.-K. Guo, 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).

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 [16]. Comprehensive experiments are conducted on three bug repositories, i.e., *Eclipse* [17], *Mozilla* [18], and *GNOME* [19]. 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 over-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 oversampling approach, named RSMOTE. RSMOTE is an random over-sampling mechanism used to generate minority-class points from high-dimensional sampling space, which is the main omission in SMOTE [15]. 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 over-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 oversampling. Several different but related datasets are produced by an repeated sampling process. An ensemble method based on Choquet fuzzy integral is used to integrate the multi-classifiers 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.[20] 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.[21] 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.[6] 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.[22] 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.[23] proposed a method, named ELBlocker, to identify the blocking bug reports with imbalanced distribution. ELBlocker firstly 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.[24] 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 [25] 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.[26] performed feature extraction on bug reports, and employed multi-factors (”temporal”, ”related report”, ”severity”, ”textual”, ”author”, and ”product”) to identify the priority of bug reports. Zhang et al.[2] 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.[27] 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.[28] used multi-objective to prioritize bug reports. One is to use Spatial Pyramid Matching (SPM) approach to analyze similar screenshots; and the other one is to use natural language processing techniques (NLP) to measure the distance between bug reports. To overcome the local bias of bug reports, Wang et al.[30] 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.[29] proposed an approach called Local-based Active Classification (LOAF) 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 over-sampling 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 sub-additivity and super-additivity) among the classifiers in comparison with other fusion schemes [31, 32, 33, 34, 35]. 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 $\sigma-$algebra consisting of a group of subset of *X*, the fuzzy measure on Ω is a set function $g : \Omega \to [0, 1]$, such that:

(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 sub-additivity, while $g(A \cup B) > g(A) + g(B), A \cap B = \emptyset$ holds well, the fuzzy measure is called super-additivity. For a finite state space *X*, the power set of *X* is usually used as the $\sigma-$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 equation (1). Regarding a non-additivitive g, the sum of all individual classifier contribution may be more or less than the contribution of integrated classifiers, as shown in equations (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, we always suppose in this paper 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 [31, 34, 35, 36, 37, 38].

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 $X = \{x_1, x_2, \cdots, x_n\}$, 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 equation:

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

It is easy to see

(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 [2, 39]. 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 [6, 40, 41, 42]. 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 imporved oversampling approach to address these issues in a robust manner [7, 8, 43, 44]. 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.

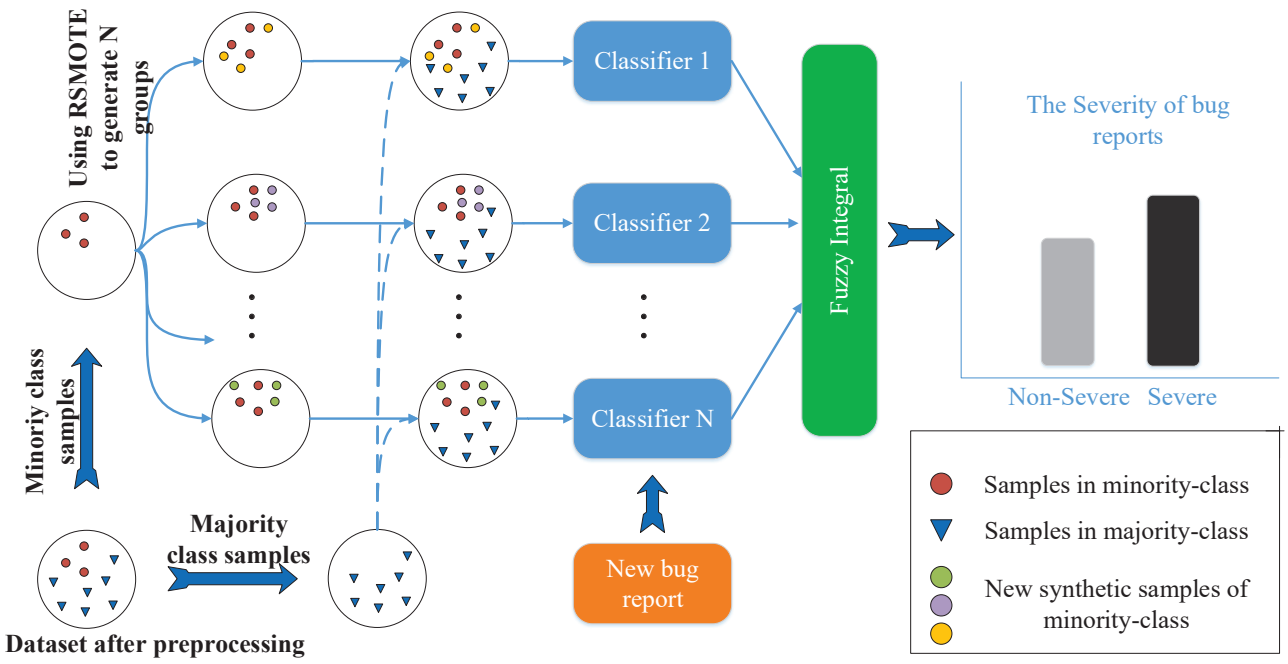


Fig. 1. The entire framework of our approach to address imbalanced issues in bug reports recognition.

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 Figure 1 shows, the framework is composed of two phases: the balance-sample phase and the identification phase. In the balance-sample phase, we firstly convert bug reports into a uniform textual features by using text preprocessing [22]. Then, RSMOTE is used to weaken the imbalanced ratio of bug reports (cf. Subsection IV.B). To lessen the uncertainty caused by random over-sampling, in identification phase, we use the RSMOTE approach to generate multiple balanced datasets. Then, Choquet fuzzy integral is used to fuse multi-classifiers trained by multiple balanced datasets, respectively (cf. 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

We detailly introduce the improved random over-sampling approach (RSMOTE) to balance the bug report datasets in this section. As can be seen in Figure 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 multi-dimensional rectangle area in RSMOTE approach. Finally, two constrains in RSMOTE determine whether the new synthetic minority-class samples to be used to augment the original datasets. The first constraint is scaling the random over-sampling 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

Algorithm 1 RSMOTE Algorithm

Input:

Input the original bug reports (*DT*), the non-severe 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 *non-severe* samples to P';
   (a) Calculate the Euclidean distances (*R*) between X_i and all other *non-severe* samples;
   (b) Randomly select $YS_N = \{Y_1, \cdots, Y_j, \cdots, Y_N\}$ from the *k* nearest neighbor samples based on the *R* values;
   (c) For each $Y_j \in YS_N$, randomly generate a new *non-severe* 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** $S = DT \cup P'$.

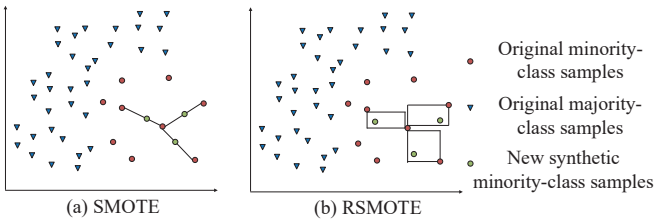


Fig. 2. Diagrams of (a) SMOTE and (b) RSMOTE. Blue triangle represents original major-class samples, green dots represents original minority-class samples, red dots represents new synthetic minority-class samples. Lines and rectangles represent the sampling space of the SMOTE and RSMOTE approaches, respectively.

III.B and III.D 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 $(x_{i1}, \cdots x_{it}, \cdots x_{in})$, where $i \in [1, m]$ and $t \in [1, n]$. Similarly, the attributes of each Y_j can be represented as $\{y_{j1}, \cdots y_{jt}, \cdots, y_{jn}\}$, 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 $\left\{X'_1, \cdots, X'_j, \cdots X'_N\right\}$, where the attributes of each X'_j can be represented as $\left\{x'_{j1}, \cdots, x'_{jt}, \cdots x'_{jn}\right\}$, where $j \in [1, N]$ and $t \in [1, n]$.

In the RSMOTE approach, the interval in which each x'_{jt} is generated between $\left(z^1_{jt}, z^2_{jt}\right)$, 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:

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

where $j \in [1, N]$, $t \in [1, n]$, $random(0, 1)$ 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 $Dis\left(X'_j X_i\right)$ denote the Euclidean distance between X'_j and X_i, and let $Dis\left(Y_j X_i\right)$ denote the Euclidean distance between Y_j and X_i. When $Dis\left(Y_j X_i\right)$ is greater than $Dis\left(X'_j X_i\right)$, this constraint is satisfied.

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

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

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 *non-severe*, 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 Figure 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) [46, 47, 59]. 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 multi-dimensional samples into three-dimensional samples. As shown in Figure 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 Figure 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 consideration to 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 Figure 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 description, it can be observed that the RSMOTE algorithm has the following advantages:

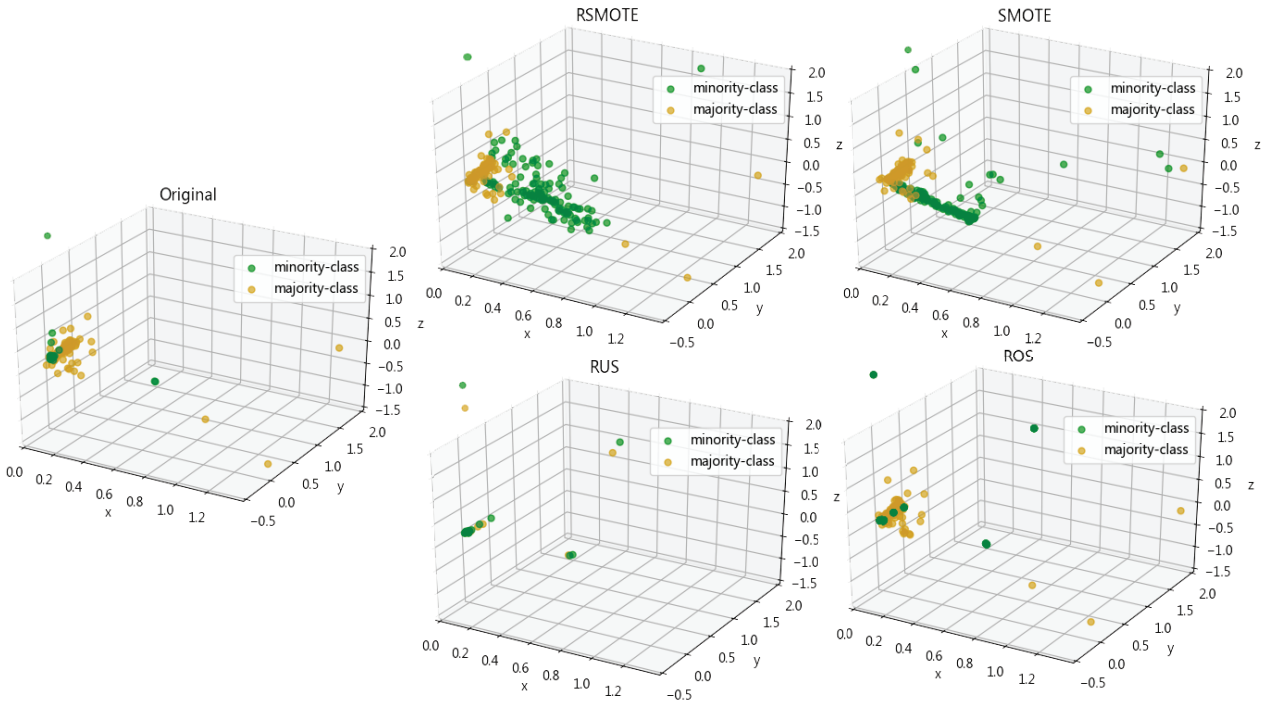


Fig. 3. The three-dimensional distribution of Core-XPConnect by using different ILS.

(1) After over-sampling 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 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 [11, 13, 45, 48, 49], 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 $La = \{La_1, \cdots, La_j, \cdots La_C\}$ is a set with C-class labels. Furthermore, we define a set of classifiers $E = \{E_1, \cdots, E_i, \cdots E_L\}$ in which each classifier is trained over a $Tr_i \in subTrs = \{Tr_1, \cdots, Tr_i, \cdots, Tr_L\}$, 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.,

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

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 $j = 1, \cdots, C$:

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

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

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

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

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 $E=\{E_1,\cdots,E_i,\cdots,E_L\}$, 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 $E=\{E_1,\cdots,E_i,\cdots,E_L\}$, $\forall E_i\in E$, $i\in[1,L]$, let $g^i=g(\{E_i\})$. g^i represents the fuzzy density of classifier E_i. We use the equation (17) to compute g^i:

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

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

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

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

where $\lambda>-1$ and $\lambda\neq 0$. The value of λ can be computed by equation (19):

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

Definition 8. Given $E=\{E_1,\cdots,E_i,\cdots E_L\}$, 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 [16].

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

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 descirbed 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:
   (a) Use the RSMOTE algorithm to generate *subTr* by *Tr*;
   (b) Train the classifiers by *subTr*, respectively;
   (c) Calculate the fuzzy density g^i of the classifier E_i;
   (d) Calculate the λ value.
2: Integrated phase:
   (a) For $\forall x\in Te$, calculate the decision profile $DP(x)$;
   (b) Each column of *DP* is sorted in ascending order to obtain a new decision profile matrix DP';
   (c) Calculate the fuzzy measure $g(A_i)$ based on DP';
   (d) Calculate $u_j(x)$ using equation (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.

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

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

(a) For each *x* in *Te*, we can get a decision profile $DP(x)$ using equation (15).
(b) We sort each column of *DP* in ascending order to obtain a new decision profile matrix DP'. Then, the *k*th column of DP' is $[e_{z_1k},e_{z_2k},\cdots,e_{z_Lk}]^T$, where e_{z_Lk} is the highest support degree and e_{z_1k} is the lowest support degree. The fuzzy densities of the corresponding classifiers are denoted by $(g^{z_1},g^{z_2}\cdots,g^{z_L})$
(c) Then, we let $g(A_1)=g^{z_1}$ and iteratively calculate $g(A_i)$ using equation (18), where $i=1,2,\cdots,L$.
(d) By calculating $u_j(x)$ using equation (20), we obtain $\{u_1(x),u_2(x),\cdots,u_j(x),\cdots,u_C(x)\}$, where $j=1,2,\cdots,C$.

In Step 3, we compute the category label j^* of each *x* based on equation (21):

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

IV. Experimental Design

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

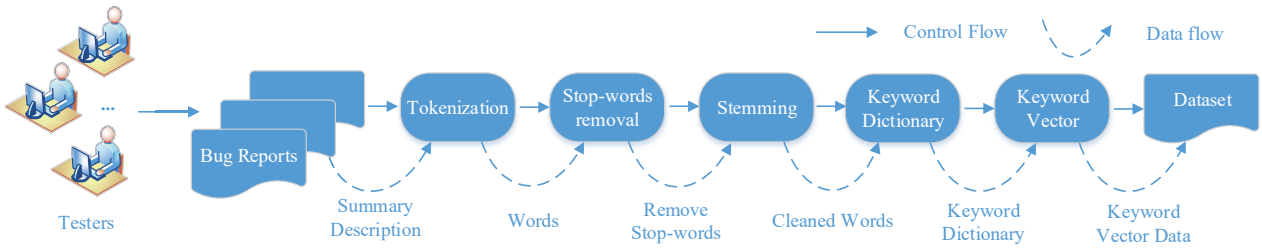


Fig. 4. The workflow of bug report processing.

A. Experimental Design

We verify the performance of FMR-FI on *Eclipse*, *Mozilla*, and *GNOME*, which all use the same bug tracking system (*Bugzilla*). In this study, sixteen datasets are selected from three 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 [7, 39], 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 [2], 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 [2, 39, 60], in which the *non-severe* 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., (1) tokenization; (2) stop-word removal; (3) stemming; (4) keyword dictionary; and (5) keyword vector, which is clearly shown in Figure 4.

B. Experimental Setup

In our experiment, we use four well-known ILS (RUS, ROS, SMOTE and CMA) [6] as baseline algorithms to compare with RSMOTE. In addition, we use Weka [61] to implement four popular classification algorithms (*NB*, *KNN*, *J48*, and Random Tree (*RT*)).

There are two integration approaches for the FMR-FI. The one is to integrate the same classifiers, another is to fuse different classifiers. In the 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 [52-56].

Stratified three-fold cross-validation is applied in our experiment, which could keep the distributive characteristics during each training iteration [50, 51, 57, 58, 62]. 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 [6], 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

TABLE I
THE 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

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 [21]. The four evaluation metrics can be computed by the confusion matrix, as presented in Table II.

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

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

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

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

$$\Pr ecision = \frac{TP}{TP + FP}. \tag{23}$$

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

$$\mathrm{Re}call = \frac{TP}{TP + FN}. \tag{24}$$

(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.

$$F - measure = \frac{2 \times \Pr ecision \times \mathrm{Re}call}{\Pr ecision + \mathrm{Re}call}. \tag{25}$$

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-VIII. 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", 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.

In Tables III-V, 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 IV, 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-V, the RSMOTE approach can also yield a better average accuracy than the other ILS. In Table IV, 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 [49]. 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 VI-VIII.

In Tables VI-VIII, 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 Tables VI-VIII, the maximum *F-measures* produced by RSMOTE are higher than that of others ILS (RUS, ROS, CMA and SMOTE). For example, in Table VII, 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 reports 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

TABLE IV
THE 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 VII
THE *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**

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 balance. In this research, 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 Figures 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 Figure 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%. The corresponding improvements of the average *F-measure* are 6.94%, 7.41%, 2.56%, 12.33%, 5.63%, and 9.46%, respectively. In Figure 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 are 1.66%, 0%, 10.06%, and 4.70%. The corresponding increments of average *F-measure* are 4.71%, 1.10%, 10.96%, and 3.61%, respectively. In Figure 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%. 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 improve-

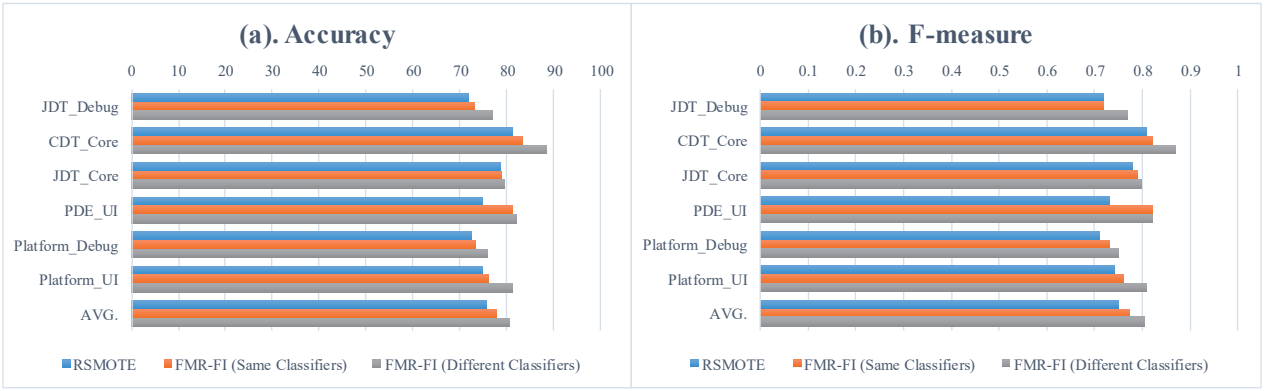


Fig. 5. The performance of predicting the severity of *Eclipse* bug reports.

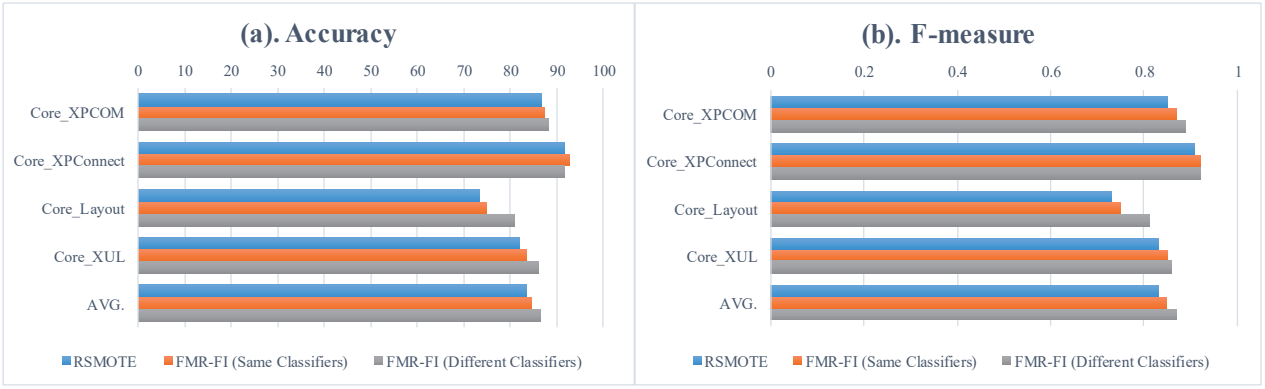


Fig. 6. The performance of predicting the severity of *Mozilla* bug reports.

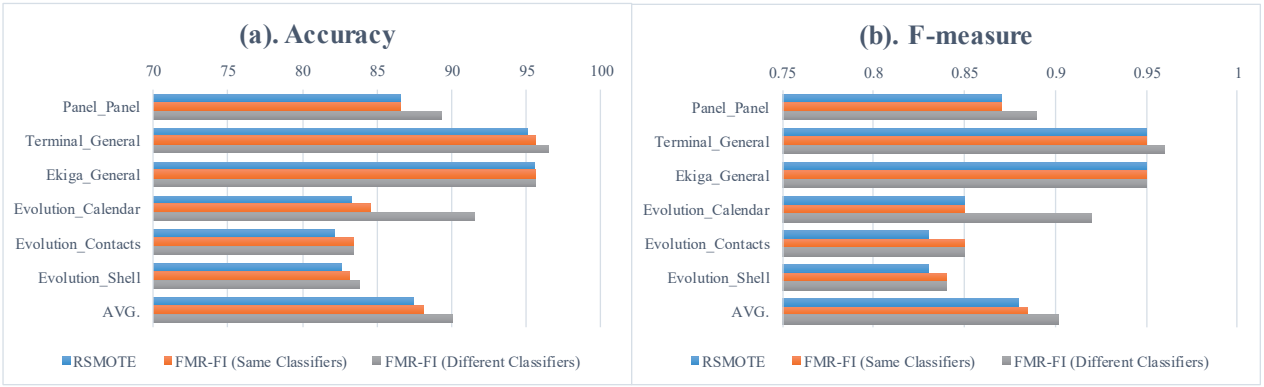


Fig. 7. The performance of predicting the severity of *GNOME* bug reports.

ment 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 fusion of multi-RSMOTE with fuzzy integral 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 multi-classifiers to predict the class label of bug reports. The accuracy and *F-measure* are shown in Figures 8-10, the performance of the FMR-FI is better than that of voting, bagging, and AdaBoost approaches on all datasets. Figure 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. Figure 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. Figure 10 shows the performance in classifying 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

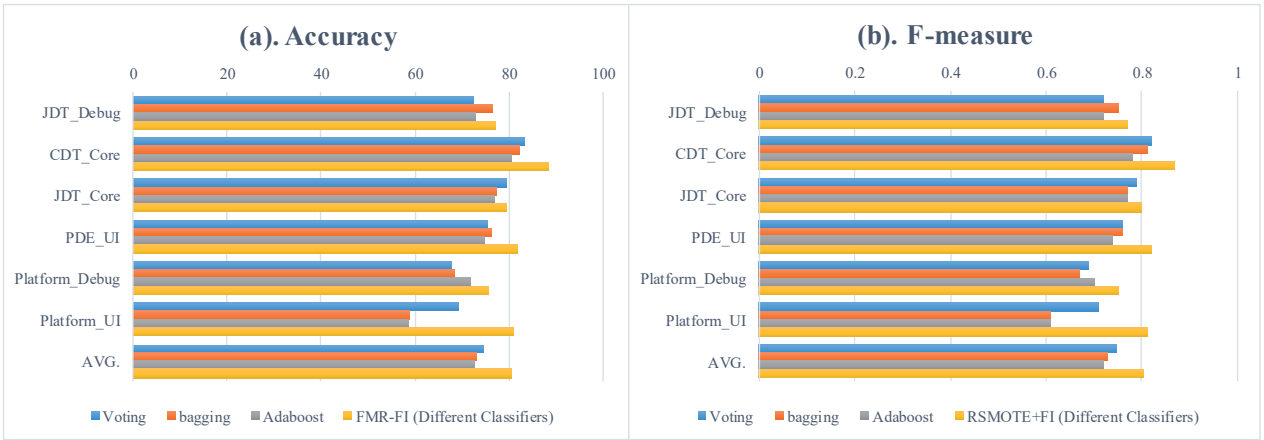


Fig. 8. The performance of predicting the severity of *Eclipse* bug reports.

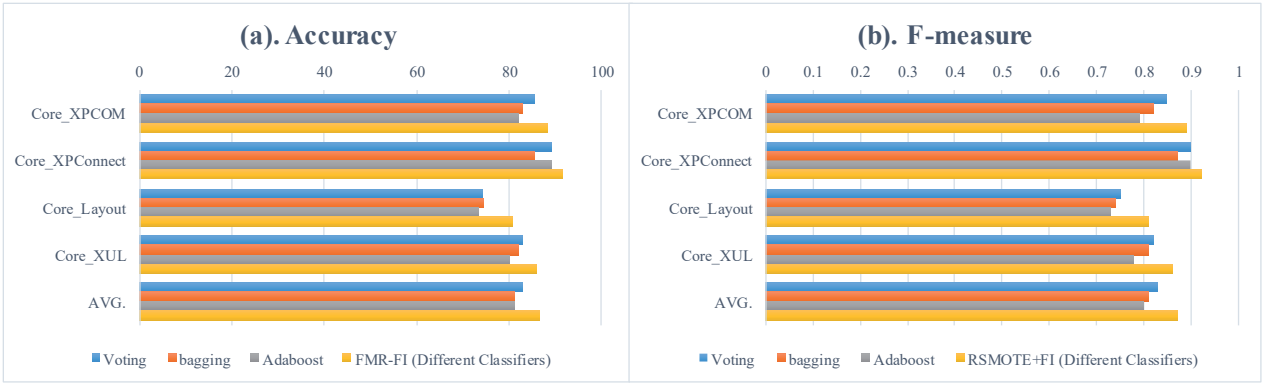


Fig. 9. The performance of predicting the severity of *Mozilla* bug reports.

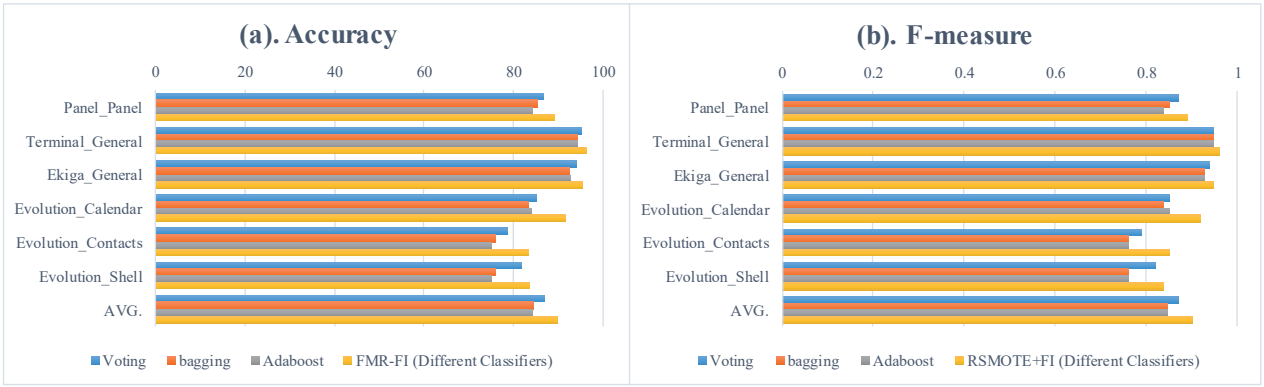


Fig. 10. The performance of predicting the severity of *GNOME* bug reports.

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] Xin Xia, David Lo, Xinyu Wang, Bo Zhou, "Accurate developer recommendation for bug resolution," *In Proceedings of the 20th Working Conference on Reverse Engineering*, WCRE'13, pp. 72-81.

[2] Ahmed Lamkanfi, Serge Demeyer, Quinten David Soetens, Tim Verdonck, "Comparing mining algorithms for predicting the severity of a reported bug," *Proceedings of the European Conference on Software Maintenance and Reengineering*, CSMR 2011, pp.249-258.

[3] Bugzilla, https://www.bugzilla.org/, 2/2/2018 available.

[4] JIRA, https://www.atlassian.com/software/jira/, 11/10/2018 available.

[5] Mantis, https://www.mantisbt.org/, 11/10/2018 available.

[6] Xinli Yang, David Lo, Xin Xia, Qiao Huang, Jian-Ling Sun, "High-Impact Bug Report Identification with Imbalanced Learning Strategies," *J. Comput. Sci. Technol.* 32(1): 181-198 (2017).

[7] Ahmed Lamkanfi, Serge Demeyer, Emanuel Giger, Bart Goethals, "Predicting the severity of a reported bug," *in Mining Software Repositories*, MSR2010, pp.1-10.

[8] Emad Shihab, Audris Mockus, Yasutaka Kamei, Bram Adams, Ahmed E. Hassan, "High impact defects: A study of breakage and surprise defects," *In Proc. the 19th ACM SIGSOFT FSE and the 13th ESEC*, SIGSOFT FSE 2011: 300-310.

[9] Ahmed Lamkanfi, Serge Demeyer, Quinten David Soetens, Tim Verdonck, "Comparing Mining Algorithms for Predicting the Severity of a Reported Bug," CSMR 2011, pp.249-258.

[10] Satuluri Naganjaneyulu, Mrithyumjaya Rao Kuppa, Ali MIrza Mahmood, "An Efficient Wrapper approach for Class Imbalance Learning using Intelligent Under-Sampling," *International Journal of Artificial Intelligence and Applications for Smart Devices*, vol.2 , no.1 (2014), pp.23-40.

[11] David A. Cieslak, Nitesh V. Chawla, "Learning decision trees for unbalanced data," *Machine Learning and Knowledge Discovery in Databases*, 2008, pp. 241-256.

[12] Jianping Zhang and Inderjeet Mani, "KNN approach to unbalanced data distributions: A case study involving information extraction," *Proc. Int. Conf. Mach. Learning, Workshop: Learning Imbalanced Data Sets*, 2003, pp.42-48.

[13] Haibo He, Edwardo A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, 2009, 21(9): 1263-1284.

[14] He Jiang, Liming Nie, Zeyi Sun, Zhilei Ren, Weiqiang Kong, Tao Zhang, Xiapu Luo, "ROSF: Leveraging Information Retrieval and Supervised Learning for Recommending Code Snippets," *IEEE Transactions on Services Computing*, PrePrints, doi:10.1109/TSC.2016.2592909.

[15] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, W. Philip Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, 2002, pp.321-357.

[16] Zhenyuan Wang, Klir, George, "Fuzzy Measure Theory," Plenum, New York, 1992.

[17] Eclipse, http://bugs.eclipse.org/bugs, 2/2/2018 available.

[18] Mozilla, http://bugzilla.mozilla.org, 2/2/2018 available.

[19] GNOME, http://bugzilla.gnome.org, 2/2/2018 available.

[20] Giuliano Antoniol, Kamel Ayari, Massimiliano Di Penta, Foutse Khomh, and Y.-G. Gueheneuc, "Is it a bug or an enhancement?: a text based approach to classify change requests," *Proceedings of the conference of the center for advanced studies on collaborative research*, 2008, pp.304-318.

[21] Tim Menzies, Andrian Marcus, "Automated severity assessment of software defect reports," *in Proceedings of IEEE International Conference on Software Maintenance*, ICSM 2008, pp.346-355.

[22] Shikai Guo, Rong Chen, Hui Li, "Using Knowledge Transfer and Rough Set to Predict the Severity of Android Test Reports via Text Mining," *Symmetry*, 2017, 9(8): 161.

[23] Xin Xia, David Lo, Emad Shihab, Xinyu Wang, Xiaohu Yang, "EL-Blocker: Predicting blocking bugs with ensemble imbalance learning," *Information and Software Technology*, 61: 93-106 (2015).

[24] Jifeng Xuan, He Jiang, Hongyu Zhang, Zhilei Ren, "Developer recommendation on bug commenting: a ranking approach for the developer crowd," *SCIENCE CHINA Information Sciences*, 60(7): 072105:1-072105:18 (2017).

[25] John Anvik, Gail C. Murphy, "Reducing the effort of bug report triage: Recommenders for development oriented decisions," *ACM Transactions on Software Engineering and Methodology*, 2011, 20(3): 10.

[26] Yuan Tian, David Lo, Xin Xia, Chengnian Sun, "Automated prediction of bug report priority using multi-factor analysis," *Empirical Software Engineering* 20(5): 1354-1383 (2015).

[27] Yang Feng, Zhenyu Chen, James A. Jones, Chunrong Fang, Baowen Xu, "Test report prioritization to assist crowdsourced testing," *in Proceedings of the 10th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering*, FSE 2015, 2015, pp. 225-236.

[28] Yang Feng, James A. Jones, Zhenyu Chen, Chunrong Fang, "Multi-objective Test Report Prioritization using Image Understanding," *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, 2016, pp. 202-213.

[29] Junjie Wang, Song Wang, Qiang Cui, Qing Wang, "Local-Based Active Classification of Test Report to Assist Crowdsourced Testing," *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, 2016, pp.190-201.

[30] Junjie Wang, Qiang Cui, Qing Wang, Song Wang, "Towards effectively test report classification to assist crowdsourced testing," *in Proceedings of ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, ESEM 2016, vol.6, no.1, pp.6-10.

[31] M. Sugeno, "Fuzzy measures and fuzzy integrals-A survey," *Fuzzy Automata and Decision Processes, M.M. Gupta, G.N Saridis and B.R. Gaines, Eds*. Amsterdam: North-Holland, pp: 89-102.1977.

[32] Changzhong Wang, Qiang He, Mingwen Shao, Qinghua Hu, "Feature selection based on maximal neighborhood discernibility," *International Journal of Machine Learning and Cybernetics*, vol.9(11), pp.1929-1940, 2018.

[33] Wu Deng,Shengjie Zhang, Huimin Zhao, Xinhua Yang, "A novel fault diagnosis method based on integrating empirical wavelet transform and fuzzy entropy for motor bearing," *IEEE Access*, 2018,6(1): 35042-35056.

[34] Rana Aamir Raza Ashfaq, Xi-Zhao Wang, "Impact of fuzziness categorization on divide and conquer strategy for instance selection," *Journal of Intelligent and Fuzzy Systems*, 2017, vol.33(3), pp.1007-1018.

[35] Xi-zhao Wang, Rana Aamir and Ai-Min Fu, "Fuzziness based sample categorization for classifier performance improvement," *Journal of Intelligent and Fuzzy Systems*, 2015, vol.29(3), pp.1185-1196.

[36] Ajoy Kanti Das, "Weighted fuzzy soft multiset and decision-making," *International Journal of Machine Learning and Cybernetics*, vol.9(5), pp.787-794, 2018.

[37] Syed Shahnewaz Ali, Tamanna Howlader, S. M. Mahbubur Rahman, "Pooled shrinkage estimator for quadratic discriminant classifier: an analysis for small sample sizes in face recognition," *International Journal of Machine Learning and Cybernetics*, vol.9(3), pp.507-522, 2018.

[38] Jagadeesh Gopal, Arun Kumar Sangaiah, Anirban Basu, Xiao Zhi Gao, "Integration of fuzzy DEMATEL and FMCDM approach for evaluating knowledge transfer effectiveness with reference to GSD project outcome," *International Journal of Machine Learning and Cybernetics*, vol.9(2), pp.225-241, 2018.

[39] Tao Zhang, Jiachi Chen, Geunseok Yang, Byungjeong Lee, Xiapu Luo, "Towards more accurate severity prediction and fixer recommendation of software bugs," *Journal of Systems and Software*, vol(117), pp.166-184, 2016.

[40] Martin F. Porter, "An algorithm for suffix stripping," *Program*, vol.14, no.3, pp.130-137, 1980.

[41] Chuan Yue, "Normalized projection approach to group decision-making with hybrid decision information," *International Journal of Machine Learning and Cybernetics*, vol.9(8), pp.1365-1375, 2018.

[42] Guiwu Wei, Fuad E. Alsaadi, Tasawar Hayat, Ahmed Alsaedi, "Projection models for multiple attribute decision making with picture fuzzy information," *International Journal of Machine Learning and Cybernetics*, vol.9(4), pp.713-719, 2018.

[43] Tim Menzies, Andrian Marcus, "Automated severity assessment of software defect reports," *in Proceedings of IEEE International Conference on Software Maintenance*, ICSM 2008, pp.346-355.

[44] Junhai Zhai, Hongyu Xu, Yan Li, "fusion of extreme learning machine with fuzzy integral", *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 2013, vol.21, Suppl.2, pp.23-34.

[45] Junhai Zhai, Liguang Zang, Zhaoyi Zhou, "Ensemble Dropout Extreme Learning Machine via Fuzzy Integral for Data Classification," *Neurocomputing*, 2018, 275:1043-1052.

[46] Nathan Halko, Per-Gunnar Martinsson, Joel A. Tropp, "Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions," *SIAM Review* 2011, vol.53(2), pp.217-288.

[47] Radim Rehurek, "Fast and Faster: A Comparison of Two Streamed Matrix Decomposition Algorithms," *CoRR abs/1102.5597*, 2011.

[48] Junhai Zhai, Sufang Zhang, Chenxi Wang, "The Classification of Imbalanced Large Data Sets Based on MapReduce and Ensemble of ELM Classifiers," *Int. J. Machine Learning & Cybernetics*, 8(3): 1009-1017 (2017).

[49] Cheng G. Weng, Josiah Poon, "A new evaluation measure for imbalanced datasets," *In Proceedings of the Seventh Australasian Data Mining Conference*, 2008, vol.87, pp.27-32.

[50] Chao Zhang, Deyu Li, Jiye Liang, "Hesitant fuzzy linguistic rough set over two universes model and its applications," *International Journal of Machine Learning and Cybernetics*, vol.9(4), pp.577-588, 2018.

[51] Lijuan Zheng, Hongwei Wang, Song Gao, "Sentimental feature selection for sentiment analysis of Chinese online reviews," *International Journal of Machine Learning and Cybernetics*, vol.9(1), pp.75-84, 2018.

[52] Wu Deng, Huimin Zhao, Xinhua Yang, Juxia Xiong, Meng Sun,Bo Li, "Study on an improved adaptive PSO algorithm for solving multi-objective gate assignment," *Applied Soft Computing*, 2017, 59:288-302.

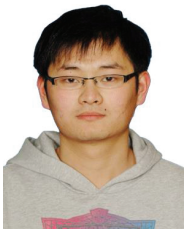
[53] Lior Rokach, "Ensemble-based classifiers," *Artificial Intelligence Review*, vol.33, pp.1-39, 2010.

[54] Gang Wang, Jianshan Sun, Jian Ma, Kaiquan Xu, Jibao Gu, "Sentiment classification: The contribution of ensemble learning," *Decision Support Systems*, vol.57, pp.77-93, 2010.

[55] Wu Deng,Rui Yao, Huimin Zhao,Xinhua Yang, Guangyu Li, "A novel intelligent diagnosis method using optimal LS-SVM with improved PSO algorithm," *Soft Computing*, 2017,DOI: 10.1007/s00500-017-2940-9.

[56] Sotiris B. Kotsiantis, Dimitris Kanellopoulos, "Combining bagging, boosting and dagging for classification problems," *Lecture Notes in Computer Science*, 2007, pp.493-500.
[57] Wu Deng, Huimin Zhao, Li Zou, Guangyu Li, Xinhua Yang, Daqing Wu, "A novel collaborative optimization algorithm in solving complex optimization problems," *Soft Computing*, 2017, 21(15):4387-4398.
[58] Xin Xia, David Lo, Xinyu Wang, Bo Zhou, "Tag recommendation in software information sites," *In Proc. the 10th Working Conference on Mining Software Repositories*, 2013, pp.287-296.
[59] Serena Morigi, Lothar Reichel, Fiorella Sgallari, "A truncated projected SVD method for linear discrete ill-posed problems," *Numerical Algorithms*, 2006, vol.43(3), pp.197-213.
[60] I. Herraiz, D. German, J. Gonzalez-Barahona, and G. Robles, "Towards a Simplification of the Bug Report Form in Eclipse," *in 5th International Working Conference on Mining Software Repositories*, May 2008.
[61] Weka, http://www.cs.waikato.ac.nz/ml/weka/, 2/2/2018 available.
[62] Chengnian Sun, David Lo, Siau-Cheng Khoo, "Towards more accurate retrieval of duplicate bug reports," *Proceedings of 26th IEEE/ACM International Conference on Automated Software Engineering*, ASE 2011: 253-262.

Rong Chen He received the M.S. and Ph.D. degree in computer software and theory from the Jilin University, China, in 1997 and 2000. He is currently a professor of the College of Information Science and Technology at the Dalian Maritime University, and has previously held position at Sun Yat-sen University, China. His research interests are in software diagnosis, collective intelligence, activity recognition, Internet and mobile computing. He is a member of the IEEE and a member of the ACM.



Shi-Kai Guo received the BSc degree in computer science in 2012 and currently pursuing the Ph.D. degree in computer science and technology from the Information Science and Technology College, Dalian Maritime University, Dalian, China. 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 2001 to 2014, he has been a Full Professor and the Dean of the College of Mathematics and Computer Science, Hebei University, Hebei, China. From 1998 to 2001, he was a Research Fellow with the Department of Computing, Hong Kong Polytechnic University, Hong Kong. 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 was a 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 Co-Chair 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 of the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS-PART B: CYBERNETICS, *Information Sciences Journal*, and the *International Journal of Pattern Recognition and Artificial Intelligence*. He is a fellow of the IEEE and a fellow of the CAAI.

Tian-Lun Zhang received the BSc degree 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 pursuing 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.