IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 3, 2018, accepted August 5, 2018, date of publication August 16, 2018, date of current version September 7, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2865780

Ensemble Data Reduction Techniques and Multi-RSMOTE via Fuzzy Integral for Bug Report Classification

SHIKAI GUO[1], RONG CHEN[1], (Member, IEEE), MIAOMIAO WEI[1], HUI LI[1], AND YAQING LIU[1,2]
[1]College of Information Science and Technology, Dalian Maritime University, Dalian 116026, China
[2]ANHUI Province Key Laboratory of Affective Computing & Advanced Intelligent Machine, Hefei University of Technology, Hefei 230009, China

Corresponding author: Rong Chen (rchen@dlmu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61672122,Grant 61602077, and Grant 61771087, 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, in part by the Fundamental Research Funds for the Central Universities under Grant 3132016348 and Grant 3132018194, and in part by the ANHUI Province Key Laboratory of Affective Computing & Advanced Intelligent Machine under Grant ACAIM20180001.

ABSTRACT Due to the unavoidable bugs appearing in the most of the software systems, bug resolution has become one of the most important activities in software maintenance. To decrease the time cost in manual work, text classification techniques are applied to automatically identify severity of bug reports. In this paper, we address the problem of low-quality and class imbalance for identifying the severity of bug reports. First, we combine feature selection with instance selection to simultaneously reduce the bug report dimension and the word dimension, which could get small-scale and high-quality reduced data set. Then, an improve random oversampling technique, named, RSMOTE, which is presented to weaken the imbalancedness degree of class distribution. Finally, to avoid the random over-sampling uncertainty of RSMOTE, we develop an ensemble learning algorithm, which is based on Choquet fuzzy integral, to combine multiple RSMOTE. We empirically investigate the performance of data reduction on ten data sets of three large open source projects, namely, Eclipse, Mozilla, and GNOME. The results show that our approach can effectively reduce the data scale and improve the performance of identifying the severity of bug reports.

INDEX TERMS Mining software repositories, data reduction, imbalance distribution, fuzzy integral.

I. INTRODUCTION

Bug tracking systems, such as Bugzilla and JIRA, play an important role in the process of managing software bugs. According to statistics, software companies spend over 45 percent of development time in repairing bugs in software development [9], [21]. Along with the increasing scale and complexity of software projects, a large-scale of bug reports are received daily by bug tracking systems. Due to bug repairing is a time-consuming and limited human resources, it is often difficult for developers to take care of all bug reports. In order to repair the bugs of software projects quickly, developers often need to prioritize the severe bug reports. Therefore, an automated technique to help developers to identify the severity of bug reports, would be more preferable to augment productivity.

Two challenges often arise during the automated technique to identify the severity of bug reports: (1) high-dimension (both bug report dimension and word dimension) is found in most bug repositories, and (2) class imbalance occurs for those datasets. High-dimension is caused by the bug reports are submitted by testers from all over the world, and for each tester, the understanding and natural language description of bugs are different, which could result in large-scale and low-quality bug reports in bug repositories [1]. A number of problems may arise due to the large-scale and low-quality, such as extensive computation and a decline in predictive performance. Class imbalance occurs when the number of bug reports in one class (majority-class) is obviously more than the other class (minority-class). This problem is more prevalent in bug repository (such Eclipse, Mozilla, and GNOME), where the proportion of severe bug reports is relatively larger than non-severe bug reports. The primary drawback of imbalanced dataset is that traditional classification algorithms tend to misclassify minority-class bug reports as majority-class bug reports.

Some investigators have try to solve these problem [9], [17]–[19]. For the high-dimension problem, Gao *et al.* [17] presents six filter-based feature ranking techniques to reduce the number of available software metrics. Xuan *et al.* [9] combines feature selection algorithms with instance selection algorithms to reduce the scale of bug datasets as well as improve the data quality. For the class imbalance problem, Lamkanfi *et al.* [18] selected an equal number of reports for each class for inclusion in the training and evaluation sets. However, the manual selection of bug reports from the original dataset may be lossy and result in weak generalizations of the trained classifier. Against imbalanced distribution of training set problem, Yang *et al.* [19] investigated four widely used imbalanced learning strategies (ILS) to solve the imbalance distribution of bug reports from four different open source projects. However, few literatures consider both high-dimension and class imbalance problems that building an automated technique model without any affect by low-quality and imbalance distribution of bug reports.

In this study, we present a process of using two data preprocessing steps, data reduction (for addressing large-scale and low-quality problem) and data sampling (for addressing class imbalance problem) together in the context of identifying the severity of bug reports.

In data reduction step, we employ the combination of feature selection (FS) and instance selection (IS) to get small-scale and high-quality set of bug reports and improve the performance of our approach to identify the severity of bug reports. To avoid the bias of a single algorithm, we consider four commonly used feature selection algorithms, namely One Rule (OneR), Information Gain (IG), ChiSquared attribute selection (CHI), and Relief-F attribute selection (RF) and four instance selection algorithms, namely Condensed Nearest Neighor (CNN), Minimal Consistent Set (MCS), Edited Nearest Neighbor (ENN), and Iterative Case Filter (ICF) [1]–[3].

In data sampling step, we propose a RSMOTE approach to balance the imbalance distribution of bug reports with respect to their severities. We select a bug report from minority-class as a center point, and then choose another bug report from minority-class with the minimum Euclidean distance as the edge point. Then, we randomly sample new bug report points in a multi-dimensional sphere, which is generated among the multi-dimensional rectangle with the center point and edge point as diagonal. In addition, the number of new synthetic bug reports is constrained by the imbalance degree of the original datasets. Furthermore, in order to solve the uncertainty of random sampling, we propose a multiple sampling strategy that applies RSMOTE approach to multiple sampling, then we train classifiers with the balanced datasets obtained from multiple samplings, respectively. Finally, we use Choquet fuzzy integral [20] to integrate the trained classifiers to classify bug reports. Comprehensive experiments have been conducted on public datasets obtained from real-world bug repositories, and the experimental results indicate that our approach could efficiently improve the distribution of bug reports the quality of training datasets. In addition, our approach could efficiently improve the uncertainty of random sampling caused by RSMOTE, which could improve the classification accuracy of identifying the severity of bug reports with an imbalanced distribution. The main contributions of this paper are as follows:

1) We propose a combination of FS and IS to addressing the problem of data reduction. This can be viewed as a combination approach in bug repositories to reduce high-dimension (both bug report dimension and word dimension) and get small-scale and high-quality training dataset.
2) We propose a RSMOTE approach to balance the imbalanced distribution of bug reports. The new synthetic minority bug reports are generated in the neighbourhood of the remaining minority-class samples by RSMOTE. Several experimental shown that RSMOTE can effectively improve the generalization ability of classifiers to identify the severity of bug reports.
3) We propose a multiple sampling mechanism using RSMOTE to solve the random sampling uncertainty of RSMOTE. Firstly, we train the classifiers respectively with the balanced datasets obtained from RSMOTE. Then, we use fuzzy integral to integrate the trained classifiers to obtain the ultimate prediction results. To our knowledge this is the first study exploring to fusion of multi-RSMOTE with fuzzy integral to classify bug reports with an imbalanced distribution.
4) We evaluate our approach using data on ten components from three bug repository datasets (Eclipse, Mozilla, and GNOME). Several experiments demonstrate that our approach yields improved classification performance for identifying the severity of bug reports with imbalanced distribution.

II. METHODOLOGY

In this section, we present a detailed description of our approach to identify the severe bug reports with an imbalanced distribution.

A. MODEL DESCRIPTION

In this section, we propose a model for identifying severe bug reports with an imbalanced severity distribution, as shown in Figure 1. The framework consists of three main phases. Firstly, employ the combination of FS and IS to get small-scale and high-quality set of bug reports (cf. Subsection III.B). Secondly, we use the RSMOTE approach to address the class imbalance problem (cf. Subsection III.C). Finally, duo to the new synthetic bug reports generated via RSMOTE are randomly generated in a certain area, which could cause the new synthetic bug reports to be noisy. To solve this problem, we combine of classifiers and fuzzy integral, while the classifiers are trained by training sets consisting of bug reports generated via RSMOTE (cf. Subsection III.D).

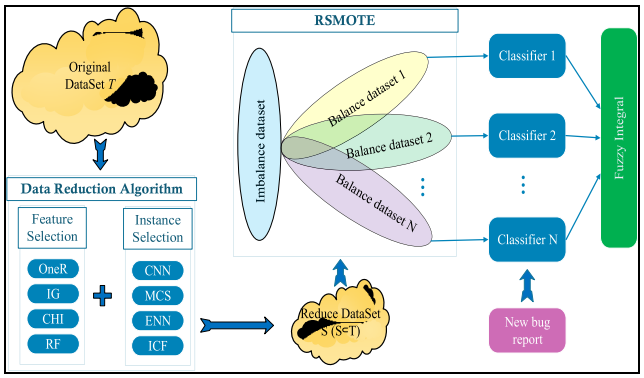

FIGURE 1. The framework of our model for identifying severe bug reports with an imbalanced distribution.

B. DATA REDUCTION ALGORITHM

Since bug reports are submitted by people from all over the world and each person's description of natural language and understanding of bugs are different, resulting in excessive noise that affects classification performance. By preprocessing bug reports (Tokenization, Stopword removal, Stemming), we convert bug reports into a text matrix with two dimensions, namely the bug report dimension and the word dimension [9], [18]. FS and IS are widely used techniques to remove the noisy or non-informative. For a given dataset in a certain application, FS aims to reduce the word dimension, which can obtain a subset of relevant words. Instance selection aims to reduce the bug report dimension, which can obtain a subset of relevant bug reports [9]. In our study, we employ combination of FS and IS to get small-scale and high-quality training reduced set. The reduced set is considered as the representative of the original dataset, which can be handled more easily by automatic techniques than the original dataset.

In our study, the orders of applying FS and IS are viewed as two different orders of bug reports reduction. We use FS□IS to denote the bug data reduction, which first applies FS and then IS; on the other hand, IS□FS denotes first applying IS and then FS. We briefly present how to reduce the bug reports based on FS□IS in Algorithm 1, and the IS□FS has the same process. To avoid the bias from a single algorithm, we examine results of four typical algorithms of feature selection algorithms (OneR, IG, CHI, and RF) and instance selection algorithms (CNN, MCS, ENN, and ICF), respectively.

In Algorithm 1, we briefly present how to reduce the bug data based on FS□IS. Lines 1-5, FS is applied to reduce the number of bug reports. When all the words of bug reports are removed, we remove the bug reports from dataset (lines 6-7). Lines 8-10, IS is applied to reduce the number of words, and we get the reduction dataset by line 11.

C. RSMOTE

In this section, we briefly present RSMOTE how to balance the imbalanced distribution of bug reports. As RSMOTE is an improve synthetic minority over-sampling

Algorithm 1 FS→IS

Input: T, original dataset,
m_I, the final number of bug reports,
n_F, the final number of words,
Output: T_{FI}, the reduction dataset.

1. $T_{FI} \leftarrow \varnothing$
2. $F \leftarrow$ statistics (T)// statistical words information (F) of T
3. **For** each f **in** F
4. $F_V \leftarrow \mathrm{FS}(f)$// apply FS to calculate the weight of all words
5. **end for**
6. $T_F \leftarrow \mathrm{select}(n_F, F_V)$// select the top n_F words from F_V
7. $I \leftarrow$ statistics(T_F)// statistical words information (I) of T_F
8. **For** each i **in** I
9. $I_V \leftarrow \mathrm{IS}(i)$// apply IS to calculate the weight of all bug reports
10. **end for**
11. $T_{FI} \leftarrow \mathrm{select}(m_I, I_V)$ // select the top m_I bug reports from I_V
12. **return** T_{FI}

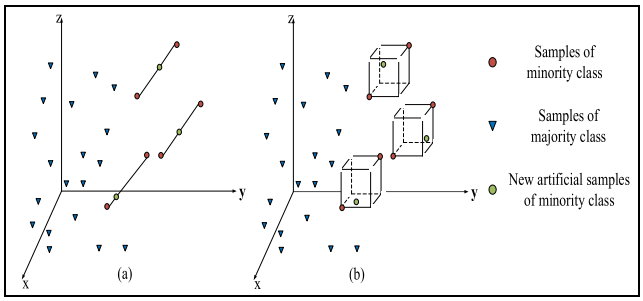

FIGURE 2. Diagrams of synthetic generation using SMOTE (a) and RSMOTE (b).

technique (SMOTE) to deal with imbalance distribution. As can be seen in Figure 2, SMOTE uses linear interpolation between two points to generate a new minority class sample, which limits the range of the sample generation. In order to solve this problem, the new synthetic minority activities are generated in the neighbourhood of the remaining minority-class examples in RSMOTE. Finally, two specified constraints control the new synthetic samples can be generated in robust manner. Compared with SMOTE, it has been improved obviously that the generalization ability of several classifiers by using the RSMOTE.

The RSMOTE algorithm is explained as follows (Algorithm 2). Lines 1-4, we initialization parameters and calculate the imbalance degree of dataset (Im_D). For each bug report, we use the Euclidean distance to find the k nearest neighbours' bug reports, and randomly select Im_D bug reports from the k nearest neighbours' bug reports (lines 5-7). We generate new synthetic minority-class bug reports from high-dimensional space (lines 8-17). Finally, if the new synthetic minority-class bug reports don't meet the specified

Algorithm 2 RSMOTE

Input: T_{FI}, the reduction dataset
MAX_C, sample set of majority class in T_{FI}
MIN_C, sample set of minority class in T_{FI}
A, set of involved words
k, the number of nearest neighbour
Output: P, balanced set of dataset

1. $P \leftarrow \varnothing$//
2. max_n=numbers(*MAX_C*)// numbers(*MAX_C*) is the number of bug reports of MAX_C
3. min_n=numbers(*MIN_C*)// numbers(*MIN_C*) is the number of bug reports of MIN_C
4. *Im_D*=round (max_n / min_n)-1//*Im_D* is the imbalanced degree of dataset.
5. **For each** *s* **in** *MIN_C*
6. *NN*←*nearsetNeighbors*(*s*, *k*) //Use the Euclidean distance to find the *k* nearest neighbours.
7. *M* ←*selectSamples*(*NN*,*Im_D*) //use *selectSamples*(*NN*,*Im_D*) to select *Im_D* samples from *NN* randomly.
8. **For each** *m* **in** *M*
9. *newSample* ← ∅
10. *newValues* ← ∅
11. **For each** *a* **in A**
12. *low* ← *value*(*s*, *a*)−0.5*abs(*value*(*m*, *a*)-*value*(*s*, *a*))// *abs* is used to solve absolute value.
13. *up* ← *value*(*s*, *a*)+0.5*abs(*value*(*m*, *a*)-*value*(*s*, *a*)) // *value* is used to get the value of *s* about *a*.
14. *newValue* ← *value*(*s*, *a*)+*random*(0,1)*(*up- low*)
15. *newValues* ← *newValues*∪{(*a*, *newValue*)}
16. **end for**
17. *newSample* ← *newSample*∪{*newValues*}
18. **if** ((|m-s| > |*newSample*-s|) && (category(select1nn(*newSample*, T_{FI}))= category (s)))
19. //select1nn(*newSample*, T_{FI}) represents select the nearest sample to *newSample* in T_{FI}. category(s) represent the label of s
20. $P \leftarrow P \cup$ {newSample}
21. **else**
22. Goto 9, re-generate new samples.
23. **end if**
24. **end for**
25. **end for**
26. **return** *P*

constraints, RSMOTE will regenerate minority-class bug reports (lines 18-22).

D. FUZZY INTEGRAL

In this section, we first introduce the notation and definitions used in our work and then present an approach of fusing multi-classifiers with fuzzy integrals (FC-FI).

Let $Tr = \{x|x \in R^m\}$ denote a training set, let $Te = \{x|x \in R^m\}$ denote a testing set, and let $La = \{La_1, La_2, \ldots, La_C\}$ be a set of class labels, where C is the total number of class labels. Let $E = \{E_1, E_2, \ldots, E_L\}$ be a set of classifiers trained on different training sets $subTrs(subTrs = \{Tr_1, Tr_2, \ldots, Tr_i, \ldots, Tr_L\})$, where L is the total number of training sets extended using the RSMOTE and E_i is a basic classifier trained on the Tr_i that has been extended using the RSMOTE approach ($i \in [1, L]$). For all $x \in R^m$, E_i assigns a class label from La to x. We define the classifier output as a C-dimensional vector consisting of support degrees for the classes, as follows [26].

$$E_i(x) = \left(e_{i1}(x), e_{i2}(x), \ldots, e_{ij}(x), \ldots, e_{iC}(x)\right) \quad (2.1)$$

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 an estimate of the posterior probability $p(La_c|x)$. In the following, we will present some related definitions.

Definition 1: Given $E = \{E_1, E_2, \ldots, E_L\}$, $La = \{La_1, La_2, \ldots, La_C\}$, and $Te = \{x|x \in R^m\}$, for each $x \in Te$, the decision profile matrix is shown as follows:

$$DP(x) = \begin{bmatrix} e_{11}(x) & \cdots & e_{1j}(x) & \cdots & e_{1C}(x) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ e_{i1}(x) & \cdots & e_{ij}(x) & \cdots & e_{iC}(x) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ e_{L1}(x) & \cdots & e_{Lj}(x) & \cdots & e_{LC}(x) \end{bmatrix} \quad (2.2)$$

where the ith row of the matrix is the output of classifier E_i and the jth column of the matrix consist of the support degrees from all classifiers $E_1, E_2, \ldots, E_L$ for class La_j.

Definition 2: Given $E = \{E_1, E_2, \ldots, E_L\}$, $E_i \square E$, $i \square [1, L]$, let $g^i = g(\{E_i\})$. g^i is called the fuzzy density of classifier E_i. We use the following formula to calculate g^i:

$$g^i = \frac{p(E_i)}{\sum_{k=1}^{L} p(E_k)} \times d_{sum} \quad (2.3)$$

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

Definition 3: Given $E = \{E_1, E_2, \ldots, E_L\}$, let $P(E)$ be the power set of E. The fuzzy measure on E is a set function g: $P(E) \to [0, 1]$ such that

$$g(\varnothing) = 0, \quad g(\{E\}) = 1. \quad (2.4)$$

$$\text{For } \forall A, B \subseteq E, \quad \text{if } A \subset B, \text{ then } g(A) \le g(B). \quad (2.5)$$

Definition 4: Given $E = \{E_1, E_2, \ldots, E_L\}$, g is called a λ-fuzzy measure. The any subset $(A_k, k \square [1, L])$ of g could be calculated by the following formulas.

$$\begin{aligned} g(A_1) &= g(\{E_1\}) = g^1, \\ g(\{E_k\}) &= g^k, \\ \lambda g(A_k) &= g^k + g(A_{k-1}) + \times g^k \times g(A_{k-1}) \end{aligned} \quad (2.6)$$

where $\lambda > -1$ and $\lambda \neq 0$. The value of λ can be determined using the following formula:

$$\lambda + 1 = \prod_{i=1}^{L} \left(1 + \times g^i\right) \tag{2.7}$$

Definition 5: Given $E = \{E_1, E_2, \ldots, E_L\}$, g is the fuzzy measure on E, the Choquet fuzzy integral of function $f_j : E \rightarrow [0, 1]$ with respect to g is defined as follows [44]. The probability of the test sample in Te belongs to La_j (u_j), $j \in [1, C]$, which is calculated by combine the results of each sub-classifier with fuzzy integrals.

$$u_j = (C) \int f dg = f_j(E_1) + \sum_{i=2}^{L} \left(f_j(E_{i-1}) - f_j(E_i)\right) \times g(A_{i-1}) \tag{2.8}$$

where $0 \leq f(E_1) \leq f(E_2) \leq \ldots \leq f(E_L) \leq 1, f(E_0) = 0$, $A_i \subseteq E, A_i = \{E_1, E_2, \ldots, E_i\}$, $g(A_0) = 0$.

In Algorithm 3, we briefly present how to use Chouqet fuzzy integral to integrate multi-RSMOTE. The algorithm 3 has two main stages: training process and integrated process, which is explained in detail in the follow.

In training process, lines 1-5, we apply RSMOTE approach to multiple sampling, then we train classifiers respectively with the balanced dataset obtained from multiple samplings. Then, we calculate the fuzzy densities based on the classification results of each classifier (lines 6-9). In integrated process, Lines 10-16, we calculate a decision profile (DP), which is based on fuzzy densities of the corresponding classifiers. We sort each line of DP in descending order to obtain a new decision profile matrix DP'. Then, lines 17-19, we calculate the fuzzy measure based on DP'. Finally, we calculate the probability of the test bug reports for each category, and select the max value as the category of bug report (lines 20-23).

Algorithm 3 FC-FI Algorithm

Input: T_{FI}, the reduction dataset, which is obtained by algorithm 1
IC, integrated classifiers
C, the categories of T
Te, test dataset
Output: K, the results of identify *severe* bug reports.
Training process:

1. $RT_{FI} \leftarrow \varnothing$
2. **For each** t_{FI} **in** T_{FI}
   $RT_{FI} \leftarrow RT_{FI} \cup$ RSMOTE(t_{FI})
3. **end for**
4. $IC_{FI}^T \leftarrow$ train(IC, RT_{FI})// Train the classifiers by the RT_{FI}, respectively
5. $G \leftarrow \varnothing$
6. **For each** ic_{FI}^T **in** IC_{FI}^T
7. $G \leftarrow G \cup$ fuzzydensity(ic_{FI}^T)// calculate the fuzzy density of each classifier by equation (2.3)
8. **end for**
9. $L \leftarrow$ calculate λ (G)// calculate the value of λ using equation (2.7).

Integrated process:

10. $DP \leftarrow \varnothing$
11. $DP' \leftarrow \varnothing$
12. **For each** te **in** Te
13. $DP \leftarrow DP \cup$ decisionprofile(te, IC_{FI}^T) // Calculate decision profile DP by equation(2.2)
14. $DP' \leftarrow$ sort(DP)
    // sort each line of DP in descending order to obtain a new decision profile matrix DP'. The fuzzy densities of the corresponding basic classifiers are denoted by $(g^{z_1}, g^{z_2} \cdots, g^{z_L})$
15. $g(A_1) \leftarrow g^{z_1}$
16. **For each** l **in** L
17. $g(A_t) \leftarrow$ fuzzymeasure ($g(A_1)$, DP') // calculate the fuzzy measure according to DP' by equation (2.6)
18. **end for**
19. **For each** c **in** C
20. $U_c \leftarrow U_c \cup$ calculateUc// calculating the probability for each category by using equation (2.8),
21. **end for**
22. $K \leftarrow K \cup \arg \max_{1 \leq c \leq C} \{U_c\}$ // determine the category of te
23. **end for**
24. **return** K

III. EXPERIMENTAL DESIGN

The experimental design used to validate the performance of our approach is described in this section.

A. EXPERIMENTAL DATASET

We perform experiments on datasets from three open source projects, which are *Eclipse* [39], *Mozilla* [40], and *GNOME* [41]. The projects are selected based on three criteria. Firstly, all the projects have a large number of reported issues, which is essential for a good research in the topic. Secondly, all the projects use Bugzilla [42] as an issue tracking system, which leads to an easier manual labeling process. Thirdly, the projects are different from one another in the application domain, which is essential for a general investigation since the distribution of high-impact bugs can be very different in different application domains.

We selected ten components from three bug repositories in this study to validate FC-FI approach are presented in Table 1. *Severe* bug reports include high-severity (e.g., 'blocker', 'critical', 'major') that represents critical errors and *non-Severe* (e.g., 'minor', 'trivial') that denotes unimportant bugs [18]. According to the results of [18], summary attribute of the bug reports contains useful and precise information of the bug, thus, in our study, we select the summary as the text content for classification [28].

B. EXPERIMENTAL SETUP

In our study, we compare the performance of RSMOTE to deal with the imbalance distribution of bug reports with

TABLE 1. Datasets of Eclipse, Mozilla, and GNOME.

Project	Dataset	Non-severe bugs	Severe bugs	Number of Words	Imbalance Degree (M)
Eclipse_JDT-Core	DS-E1	512	1315	1580	2.57
Eclipse_JDT-Debug	DS-E2	291	706	1140	2.43
Eclipse_Platform-Debug	DS-E3	232	404	869	1.74
Eclipse_CDT-Core	DS-E4	114	458	817	4.02
Mozilla_Core-XPCOM	DS-M5	149	748	1489	5.02
Mozilla_Core-XPConnect	DS-M6	40	212	681	5.3
GNOME_Evolution-Calendar	DS-G7	626	2896	1669	4.63
GNOME_Evolution-Contacts	DS-G8	644	1788	1380	2.78
GNOME_Evolution-Shell	DS-G9	495	1210	1203	2.44
GNOME_Panel-Panel	DS-G10	330	1301	1135	3.94

four well-known strategies (RUS, ROS, SMOTE, CMA) [2]. To evaluate the classifiers ensemble performance of FC-FI, we used well-known standard ensemble methods mentioned in the literature, including AdaBoost, bagging, and majority voting [44], [45].

We used stratified 3-fold cross-validation to validate the performance of our approach. In stratified 3-fold cross-validation, each dataset is divided into 3 folds, with each fold containing the same proportion of sample instances belonging to each class. Next, 3 evaluation rounds are performed; in each round, two folds are used as the training dataset, and the remaining fold is used as the testing dataset. The results of all 3 evaluation rounds are aggregated to report the overall performance. Stratified cross-validation is a standard evaluation setting that is widely used in software engineering studies [46], [47].

In our experiments, we let the value of k be 5, and let $N = round(M) - 1$, where $round(M)$ represents the rounded imbalance degree M of the original bug reports. Our approach was implemented in the JAVA programming language for JDK 1.8 and executed on a machine with the following configuration: Intel® Xeon® CPU E5-2620 v3 @ 2.40 GHz, 16 G of RAM, running Windows 10. Many classification algorithms have been studied in the data mining field, each of which behaves differently in the same situation depending on its specific characteristics. In this study, we used six classifiers, namely, Naïve Bayes (*NB*), Naïve Bayes multinomial (*NBM*), Support Vector Machine (*SVM*), K-Nearest Neighbor (*KNN*), Decision Tree (*J48*), and Random Tree (*RT*), and, as implemented in the Weka toolkit [48].

C. EVALUATION METRICS

We use accuracy,*precision*, *recall* and the *F-measure* as our evaluation metrics. These metrics are commonly used measures for evaluating classification performance [23]. They can be derived from the confusion matrix, which captures all four possible classification results, as presented in Table 2. The number of true positives (*TP*) is the number of verified *surprise* test reports that are correctly classified. The number of false positives (*FP*) is the number of verified *non-surprise* test reports that are incorrectly classified as *surprise* test reports. The number of false negatives (*FN*) is the number of verified *surprise* test reports that are incorrectly classified as *non-surprise* test reports. The number of true negatives (*TN*) is the number of verified *non-surprise* test reports that are correctly classified as *non-surprise* test reports. Based on the values of *TP*, *FP*, *FN*, and *TN*, the *precision*, *recall* and *F-measure* are calculated as follows.

TABLE 2. Confusion matrix, which can be used to calculate many 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

- ***Accuracy:*** The accuracy of the model is the number of correct classifications divided by the total number of classifications. The accuracy is defined as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \times 100\%. \quad (5.1)$$

- ***Precision:*** Precision is the proportion of correctly predicted *surprise* bug reports to all bug reports predicted as *surprise*. We formally define the *precision* as follows:

$$Precision = \frac{TP}{TP + FP} \times 100\%. \quad (5.2)$$

- ***Recall:*** Recall is the proportion of the number of correctly predicted *surprise* bug reports to the actual number of *surprise* bug reports. Mathematically, recall is defined as:

$$Recall = \frac{TP}{TP + FN} \times 100\%. \quad (5.3)$$

- ***F-measure:*** *F-measure* is a summary measure that combines both *precision* and *recall*. It evaluates if an increase in *precision* (*recall*) outweighs a reduction in *recall* (*precision*). Mathematically, the *F-measure* is defined as follows:

$$F-measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \times 100\%. \quad (5.4)$$

IV. EXPERIMENTAL RESULTS

In this section, the experimental results are discussed in relation to the specific research questions.

RQ1. Which variants of FS and IS with classifiers perform the best performance for identifying severity of bug reports?

In the first research question, we consider four feature selection (OneR, IG, CHI, RF) and four instance selection (CNN, MCS, ENN, ICF) approaches with six classification algorithms (*RT*, *NB*, *NBM*, *KNN*, *SVM*, *J48*), which can have totally 48 variants (i.e., combinations of one of the FS or IS approach with one of the classification algorithms). For each FS and IS approach, we select 30%, 50%, 70%, and 90% as the ratio of the final number of words, respectively. The ratio value set up is based on the study of text instance selection [16]. We use two evaluation metrics mentioned above (accuracy, and *F-measure*) to compare the totally 48 variants. Tables 3-6 presents the performance of the FS and IS approaches with six classifiers to identify the severity of bug reports, respectively. We bold the best results of FS and IS for each variant.

TABLE 3. The accuracy of variants to identify severity of bug reports by using feature selection.

FS(ACC)		IG				CHI				ONER				RF				Original
		30%	50%	70%	90%	30%	50%	70%	90%	30%	50%	70%	90%	30%	50%	70%	90%	
DS-E1	RT	0.76	0.70	0.72	0.70	0.75	0.72	0.73	0.73	0.76	0.73	0.72	0.72	0.75	0.75	0.76	0.74	0.75
	NB	0.78	0.76	0.76	0.76	0.78	0.76	0.76	0.76	0.77	0.76	0.76	0.76	0.75	0.75	0.76	0.75	0.69
	NBM	0.78	0.79	0.79	0.78	0.78	0.79	**0.79**	0.79	0.78	0.79	0.79	0.78	0.77	0.76	0.76	0.77	0.74
	KNN	0.75	0.72	0.73	0.72	0.75	0.72	0.73	0.73	0.76	0.72	0.72	0.70	0.76	0.76	0.77	0.76	0.73
	SVM	0.78	0.78	0.78	0.78	0.78	0.78	0.78	0.78	0.78	0.79	0.78	0.78	0.78	0.78	0.78	0.77	0.76
	J48	0.78	0.78	0.78	0.78	0.78	0.78	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.78	0.78
DS-E2	RT	0.72	0.67	0.68	0.70	0.66	0.70	0.67	0.66	0.71	0.67	0.69	0.70	0.72	0.72	0.72	0.70	0.68
	NB	0.73	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.71	0.71	0.71	0.71	0.71	0.71	0.72	0.70	0.71
	NBM	**0.76**	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.74	0.76	0.75	0.74	0.74	0.74	0.74	0.73	0.74
	KNN	0.72	0.72	0.68	0.71	0.71	0.72	0.68	0.71	0.71	0.71	0.69	0.67	0.73	0.73	0.72	0.74	0.71
	SVM	0.76	0.75	0.74	0.74	0.74	0.75	0.74	0.74	0.75	0.75	0.75	0.76	0.75	0.75	0.75	0.74	0.72
	J48	0.74	0.71	0.71	0.71	0.71	0.71	0.71	0.71	0.73	0.72	0.73	0.72	0.73	0.73	0.73	0.74	0.71
DS-E3	RT	0.69	0.63	0.68	0.66	0.69	0.65	0.64	0.67	0.67	0.66	0.64	0.64	0.70	0.69	0.69	0.63	0.59
	NB	0.69	0.69	0.69	0.69	0.69	0.69	0.69	0.69	0.66	0.67	0.67	0.68	0.67	0.68	0.68	0.68	0.63
	NBM	0.69	0.69	0.70	0.70	0.69	0.69	0.70	0.70	0.69	0.70	0.70	0.70	0.71	0.70	0.68	0.70	0.69
	KNN	0.68	0.69	0.69	0.69	0.68	0.69	0.69	0.69	0.67	0.68	0.67	0.66	0.68	0.69	0.70	0.70	0.52
	SVM	0.69	0.69	0.69	0.69	0.69	0.69	0.69	0.69	0.68	0.68	0.68	**0.71**	0.70	0.69	0.67	0.69	0.70
	J48	0.64	0.64	0.64	0.64	0.64	0.64	0.64	0.64	0.66	0.65	0.65	0.65	0.67	0.67	0.67	0.67	0.59
DS-E4	RT	0.75	0.72	0.74	0.75	0.74	0.70	0.74	0.76	0.73	0.73	0.73	0.73	0.77	0.76	0.76	0.75	0.74
	NB	0.77	0.67	0.67	0.68	0.77	0.67	0.67	0.68	0.68	0.68	0.67	0.68	0.72	0.72	0.72	0.72	0.68
	NBM	0.76	0.79	0.77	0.78	0.76	0.79	0.77	0.78	0.78	0.77	0.77	0.78	**0.81**	0.78	0.79	0.79	0.80
	KNN	0.77	0.77	0.76	0.77	0.77	0.77	0.76	0.77	0.73	0.71	0.73	0.73	0.77	0.77	0.79	0.79	0.81
	SVM	0.78	0.78	0.77	0.76	0.78	0.77	0.77	0.76	0.75	0.77	0.76	0.75	0.77	0.77	0.77	0.77	0.78
	J48	0.80	0.80	0.80	0.80	0.80	0.80	0.80	0.80	0.80	0.80	0.80	0.80	0.80	0.80	0.80	0.80	0.80
DS-M5	RT	0.80	0.80	0.80	0.79	0.81	0.78	0.79	0.80	0.80	0.77	0.76	0.80	0.81	0.78	0.80	0.81	0.78
	NB	0.83	0.78	0.78	0.78	0.83	0.78	0.78	0.78	0.82	0.78	0.78	0.78	0.77	0.77	0.77	0.77	0.68
	NBM	0.83	0.81	0.81	0.81	0.83	0.81	0.81	0.81	0.83	0.81	0.82	0.81	0.81	0.81	0.82	0.82	0.82
	KNN	0.82	0.79	0.79	0.81	0.82	0.79	0.79	0.81	0.82	0.78	0.76	0.78	0.82	0.83	0.83	0.84	0.83
	SVM	0.83	0.82	0.82	0.82	0.83	0.82	0.82	0.82	0.83	0.83	0.83	**0.84**	0.83	0.82	0.83	0.82	0.80
	J48	0.83	0.84	0.84	0.84	0.83	0.84	0.84	0.84	0.83	0.83	0.83	0.83	0.84	0.84	0.84	0.84	0.83
DS-M6	RT	0.82	0.83	0.77	0.71	0.83	0.82	0.78	0.71	0.81	0.79	0.73	0.72	0.81	0.85	0.81	0.80	0.81
	NB	0.84	0.80	0.80	0.79	0.84	0.80	0.80	0.79	0.84	0.83	0.83	0.83	0.79	0.79	0.79	0.80	0.75
	NBM	0.82	0.78	0.78	0.76	0.82	0.78	0.78	0.76	0.81	0.78	0.79	0.77	0.80	0.77	0.77	0.76	0.82
	KNN	0.84	0.81	0.83	0.77	0.84	0.81	0.83	0.77	0.82	0.83	0.76	0.78	**0.85**	**0.85**	0.84	0.84	0.51
	SVM	0.83	0.81	0.81	0.83	0.83	0.81	0.81	0.83	0.82	0.81	0.80	0.82	0.85	0.84	0.84	0.85	0.82
	J48	0.83	0.82	0.82	0.82	0.83	0.82	0.82	0.82	0.82	0.83	0.83	0.83	0.83	0.83	0.82	0.81	0.81
DS-G7	RT	0.84	0.82	0.82	0.81	0.84	0.82	0.82	0.82	0.83	0.83	0.82	0.80	0.83	0.83	0.83	0.82	0.82
	NB	0.81	0.81	0.81	0.81	0.81	0.81	0.81	0.81	0.80	0.81	0.81	0.81	0.81	0.81	0.81	0.81	0.79
	NBM	0.86	0.86	0.86	**0.86**	0.86	0.86	0.86	**0.86**	0.85	0.86	0.86	0.86	0.85	0.85	0.86	0.86	0.84
	KNN	0.83	0.83	0.83	0.82	0.83	0.83	0.83	0.82	0.83	0.82	0.81	0.81	0.83	0.83	0.83	0.83	0.82
	SVM	0.86	0.86	0.85	0.85	0.86	0.86	0.85	0.85	0.84	0.85	0.86	0.85	0.85	0.85	0.86	0.86	0.85
	J48	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.82
DS-G8	RT	0.74	0.74	0.74	0.75	0.75	0.74	0.75	0.75	0.75	0.74	0.74	0.74	0.74	0.74	0.75	0.74	0.72
	NB	0.70	0.73	0.73	0.73	0.70	0.73	0.73	0.73	0.71	0.72	0.72	0.72	0.73	0.73	0.73	0.73	0.74
	NBM	0.79	0.79	0.79	0.79	**0.80**	0.79	0.79	0.79	0.77	0.78	0.79	0.79	0.78	0.78	0.79	0.79	0.78
	KNN	0.74	0.74	0.74	0.75	0.74	0.74	0.74	0.75	0.75	0.74	0.74	0.73	0.75	0.75	0.75	0.75	0.73
	SVM	0.77	0.77	0.77	0.77	0.77	0.77	0.76	0.77	0.76	0.77	0.77	0.76	0.76	0.76	0.76	0.76	0.76
	J48	0.75	0.75	0.74	0.74	0.75	0.75	0.74	0.74	0.74	0.75	0.75	0.74	0.74	0.74	0.74	0.73	0.75
DS-G9	RT	0.77	0.78	0.76	0.78	0.79	0.78	0.77	0.77	0.74	0.79	0.77	0.76	0.79	0.78	0.79	0.79	0.74
	NB	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.76	0.81	0.81	0.81	0.82	0.82	0.82	0.82	0.77
	NBM	0.84	**0.84**	0.84	0.84	0.84	**0.84**	0.84	0.84	0.79	0.84	0.84	0.84	0.83	0.83	0.83	0.84	0.82
	KNN	0.78	0.76	0.77	0.77	0.77	0.76	0.77	0.77	0.76	0.78	0.78	0.78	0.78	0.79	0.79	0.79	0.77
	SVM	0.82	0.82	0.82	0.83	0.82	0.82	0.82	0.83	0.78	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.83
	J48	0.79	0.80	0.80	0.80	0.79	0.80	0.80	0.80	0.76	0.79	0.79	0.79	0.79	0.79	0.79	0.78	0.77
DS-G10	RT	0.83	0.81	0.83	0.83	0.80	0.82	0.82	0.81	0.81	0.84	0.82	0.86	0.83	0.83	0.84	0.80	0.83
	NB	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.79	0.80	0.80	0.81	0.83	0.81	0.81	0.81	0.78
	NBM	0.84	0.85	0.86	0.86	0.84	0.85	0.85	0.86	0.81	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.84
	KNN	0.82	0.82	0.81	0.81	0.82	0.82	0.82	0.81	0.81	0.83	0.82	0.82	0.82	0.81	0.83	0.81	0.79
	SVM	0.83	0.83	0.84	0.84	0.83	0.83	0.83	0.84	0.80	0.84	0.84	0.86	0.85	**0.87**	0.85	0.85	0.86
	J48	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.80	0.83	0.83	0.83	0.83	0.84	0.84	0.84	0.84

From Tables 3-6, it can be found that the performance of identifying severity of bug reports by using four FS and four IS approaches all works much better than the original experiment, except identifying severity of DS-E4 by four IS approaches. For example, as shown in Table 3 and 4, the accuracy and *F-measure* of CHI with 70% as the ratio of the final number of words to identify severity of bug reports for DS-E1 is 0.79 and 0.79. And, the accuracy and *F-measure* of original bug reports to identify severity of bug reports for DS-E1 is 0.78 and 0.75. The average best performance of

TABLE 4. The *F-measure* of variants to identify severity of bug reports by using feature selection.

FS(F-measure)		IG				CHI				ONER				RF				Original
		30%	50%	70%	90%	30%	50%	70%	90%	30%	50%	70%	90%	30%	50%	70%	90%	
DS-E1	RT	0.74	0.70	0.72	0.69	0.73	0.72	0.73	0.72	0.75	0.72	0.72	0.72	0.74	0.74	0.76	0.73	0.73
	NB	0.77	0.76	0.76	0.76	0.77	0.75	0.76	0.76	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.70
	NBM	0.77	0.78	0.78	0.78	0.77	0.79	**0.79**	0.78	0.77	0.78	0.78	0.77	0.77	0.76	0.76	0.77	0.73
	KNN	0.72	0.69	0.70	0.69	0.73	0.69	0.70	0.70	0.73	0.71	0.70	0.70	0.74	0.74	0.75	0.73	0.71
	SVM	0.76	0.77	0.77	0.76	0.76	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.76	0.75
	J48	0.75	0.75	0.75	0.75	0.75	0.75	0.74	0.74	0.74	0.75	0.75	0.75	0.74	0.74	0.74	0.75	0.75
DS-E2	RT	0.70	0.67	0.67	0.69	0.65	0.69	0.66	0.65	0.68	0.66	0.68	0.69	0.71	0.70	0.71	0.69	0.68
	NB	0.72	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.69	0.70	0.70	0.70	0.70	0.71	0.71	0.70	0.71
	NBM	0.74	0.74	0.74	**0.74**	**0.74**	0.74	0.74	**0.74**	0.72	0.74	0.74	0.73	0.73	0.72	0.73	0.72	0.72
	KNN	0.68	0.68	0.66	0.66	0.66	0.68	0.66	0.66	0.67	0.68	0.66	0.65	0.70	0.70	0.69	0.69	0.69
	SVM	0.73	0.74	0.73	0.72	0.72	0.74	0.73	0.72	0.72	0.73	0.72	0.74	0.73	0.73	0.73	0.72	0.70
	J48	0.68	0.67	0.67	0.67	0.67	0.67	0.67	0.67	0.68	0.68	0.68	0.67	0.68	0.68	0.68	0.68	0.65
DS-E3	RT	0.67	0.63	0.67	0.65	0.67	0.64	0.64	0.66	0.65	0.65	0.64	0.64	0.69	0.68	0.68	0.62	0.59
	NB	0.68	0.69	0.70	0.69	0.68	0.69	0.70	0.69	0.65	0.67	0.67	0.68	0.67	0.67	0.67	0.68	0.68
	NBM	0.68	0.69	0.70	0.70	0.68	0.69	0.70	0.70	0.68	0.70	0.70	0.70	0.70	0.69	0.68	0.70	0.70
	KNN	0.65	0.67	0.66	0.64	0.65	0.67	0.66	0.64	0.63	0.66	0.63	0.62	0.66	0.66	0.68	0.66	0.53
	SVM	0.67	0.69	0.68	0.68	0.67	0.69	0.68	0.68	0.66	0.67	0.67	**0.71**	0.68	0.67	0.66	0.68	0.70
	J48	0.59	0.61	0.61	0.62	0.59	0.61	0.61	0.62	0.59	0.61	0.61	0.61	0.60	0.60	0.60	0.61	0.59
DS-E4	RT	0.73	0.70	0.72	0.73	0.72	0.69	0.72	0.73	0.72	0.71	0.72	0.72	0.74	0.74	0.73	0.72	0.74
	NB	0.72	0.68	0.68	0.69	0.72	0.68	0.68	0.69	0.68	0.68	0.68	0.68	0.70	0.70	0.70	0.70	0.70
	NBM	0.72	0.76	0.73	0.75	0.72	0.75	0.73	0.75	0.76	0.74	0.74	0.76	**0.78**	0.76	0.76	0.77	0.78
	KNN	0.73	0.73	0.72	0.72	0.73	0.73	0.72	0.72	0.70	0.70	0.71	0.70	0.72	0.71	0.73	0.73	0.74
	SVM	0.73	0.74	0.74	0.73	0.73	0.73	0.73	0.73	0.71	0.72	0.72	0.71	0.72	0.72	0.72	0.73	0.74
	J48	0.71	0.72	0.71	0.71	0.71	0.72	0.71	0.71	0.71	0.71	0.71	0.71	0.71	0.71	0.71	0.71	0.71
DS-M5	RT	0.78	0.79	0.78	0.78	0.78	0.78	0.79	0.77	0.77	0.76	0.76	0.78	0.78	0.76	0.77	0.78	0.76
	NB	0.81	0.79	0.79	0.79	0.81	0.79	0.79	0.79	0.80	0.78	0.78	0.78	0.78	0.78	0.78	0.78	0.72
	NBM	0.81	0.79	0.80	0.80	0.81	0.80	0.80	0.80	0.80	0.80	0.81	0.80	0.80	0.80	0.81	0.82	0.80
	KNN	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.76	0.75	0.76	0.78	0.78	0.78	0.77	0.78
	SVM	0.79	0.80	0.80	0.79	0.79	0.80	0.80	0.79	0.78	0.80	0.80	**0.82**	0.79	0.79	0.80	0.79	0.77
	J48	0.76	0.77	0.77	0.77	0.76	0.77	0.77	0.77	0.76	0.77	0.77	0.77	0.76	0.77	0.77	0.77	0.78
DS-M6	RT	0.81	0.83	0.77	0.73	0.82	0.81	0.78	0.73	0.78	0.79	0.75	0.74	0.78	0.81	0.79	0.78	0.80
	NB	**0.84**	0.81	0.81	0.81	**0.84**	0.81	0.81	0.81	0.82	0.84	0.84	0.84	0.80	0.81	0.80	0.81	0.78
	NBM	0.82	0.79	0.80	0.78	0.82	0.79	0.80	0.78	0.79	0.79	0.80	0.79	0.80	0.78	0.79	0.78	0.81
	KNN	0.80	0.79	0.80	0.76	0.80	0.79	0.80	0.76	0.78	0.81	0.77	0.79	0.79	0.79	0.79	0.80	0.56
	SVM	0.81	0.81	0.81	0.82	0.81	0.81	0.81	0.82	0.79	0.80	0.79	0.80	0.82	0.81	0.81	0.83	0.81
	J48	0.78	0.80	0.80	0.80	0.78	0.80	0.80	0.80	0.76	0.78	0.78	0.78	0.78	0.78	0.79	0.77	0.77
DS-G7	RT	0.82	0.81	0.81	0.81	0.82	0.81	0.81	0.81	0.81	0.82	0.82	0.80	0.82	0.82	0.82	0.81	0.82
	NB	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.78	0.83	0.83	0.83	0.82	0.82	0.82	0.82	0.81
	NBM	0.86	**0.87**	0.87	**0.87**	0.86	**0.87**	0.87	**0.87**	0.82	0.86	0.86	0.86	0.85	0.86	0.86	0.86	0.84
	KNN	0.82	0.81	0.80	0.81	0.82	0.81	0.80	0.81	0.80	0.81	0.80	0.80	0.82	0.82	0.82	0.82	0.79
	SVM	0.84	0.85	0.84	0.85	0.84	0.85	0.84	0.85	0.82	0.84	0.84	0.84	0.84	0.84	0.84	0.85	0.84
	J48	0.81	0.81	0.81	0.81	0.81	0.81	0.81	0.81	0.77	0.80	0.80	0.80	0.81	0.81	0.81	0.81	0.81
DS-G8	RT	0.71	0.73	0.73	0.75	0.72	0.73	0.74	0.74	0.73	0.73	0.73	0.73	0.73	0.73	0.74	0.74	0.71
	NB	0.72	0.74	0.75	0.75	0.72	0.74	0.75	0.75	0.70	0.74	0.74	0.74	0.74	0.74	0.74	0.74	0.74
	NBM	0.79	0.79	0.79	0.79	0.79	0.79	**0.79**	0.79	0.74	0.78	0.78	0.79	0.77	0.78	0.78	0.78	0.77
	KNN	0.71	0.72	0.71	0.72	0.70	0.72	0.71	0.71	0.72	0.71	0.71	0.71	0.74	0.74	0.74	0.73	0.69
	SVM	0.75	0.76	0.76	0.76	0.75	0.76	0.75	0.76	0.73	0.75	0.75	0.75	0.74	0.74	0.75	0.75	0.74
	J48	0.71	0.72	0.72	0.72	0.71	0.72	0.72	0.72	0.68	0.71	0.71	0.71	0.72	0.72	0.72	0.71	0.76
DS-G9	RT	0.75	0.78	0.76	0.77	0.77	0.78	0.76	0.77	0.71	0.78	0.76	0.75	0.78	0.78	0.78	0.78	0.73
	NB	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.76	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.78
	NBM	0.84	**0.84**	0.84	0.84	0.84	**0.84**	0.84	0.84	0.78	0.84	0.84	0.84	0.83	0.83	0.83	0.84	0.82
	KNN	0.76	0.75	0.75	0.74	0.76	0.75	0.75	0.74	0.72	0.78	0.77	0.77	0.77	0.77	0.77	0.78	0.75
	SVM	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.77	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.83
	J48	0.78	0.79	0.79	0.78	0.78	0.79	0.79	0.78	0.73	0.79	0.79	0.79	0.79	0.79	0.79	0.79	0.78
DS-G10	RT	0.82	0.80	0.82	0.82	0.79	0.81	0.81	0.81	0.79	0.83	0.81	0.85	0.81	0.81	0.84	0.80	0.82
	NB	0.83	0.84	0.84	0.84	0.83	0.84	0.84	0.84	0.78	0.82	0.82	0.82	0.84	0.83	0.83	0.82	0.79
	NBM	0.84	0.85	0.86	0.86	0.84	0.85	0.85	0.86	0.77	0.85	0.85	0.85	0.85	0.85	0.86	0.85	0.83
	KNN	0.81	0.80	0.79	0.79	0.81	0.80	0.80	0.79	0.78	0.82	0.81	0.80	0.81	0.80	0.82	0.80	0.74
	SVM	0.82	0.82	0.83	0.83	0.82	0.82	0.82	0.84	0.78	0.83	0.83	0.85	0.83	**0.86**	0.84	0.85	0.86
	J48	0.82	0.81	0.81	0.81	0.82	0.81	0.81	0.81	0.73	0.82	0.82	0.82	0.81	0.83	0.83	0.83	0.82

four feature selection (OneR, IG, CHI, RF) are better than four instance selection (CNN, MCS, ENN, ICF) approaches to identify the severity of bug reports. From Table 3-6, we can see that the performance of identify the severity of bug reports is relatively stable, while FS under different reduction scales (30%, 50%, 70%, and 90%), however, the performance of identify the severity of bug reports becomes very poor, while IS under the reduction scale of 30%. This is caused by the bug reports removed after IS includes large number of words, and the available information to build the classification model

TABLE 5. The accuracy of variants to identify severity of bug reports by using instance selection.

IS(ACC)		ENN				MCS				ICF				CNN				Original
		30%	50%	70%	90%	30%	50%	70%	90%	30%	50%	70%	90%	30%	50%	70%	90%	
DS-E1	RT	0.32	0.61	0.68	0.69	0.65	0.63	0.68	0.73	0.63	0.69	0.69	0.71	0.65	0.66	0.69	0.69	0.75
	NB	0.35	0.65	0.74	0.75	0.69	0.70	0.73	0.76	0.66	0.74	0.74	0.77	0.68	0.69	0.73	0.75	0.69
	NBM	0.29	0.67	0.76	0.78	0.70	0.72	0.76	0.78	0.68	0.73	0.76	**0.79**	0.69	0.73	0.77	0.78	0.74
	KNN	0.30	0.59	0.70	0.70	0.70	0.65	0.69	0.71	0.63	0.69	0.69	0.71	0.66	0.63	0.68	0.70	0.73
	SVM	0.31	0.68	0.75	0.77	0.71	0.71	0.75	0.77	0.68	0.73	0.74	0.77	0.71	0.73	0.75	0.77	0.76
	J48	0.28	0.66	0.76	0.78	0.63	0.71	0.76	0.77	0.62	0.75	0.75	0.78	0.59	0.71	0.75	0.77	0.78
DS-E2	RT	0.30	0.59	0.66	0.66	0.61	0.61	0.66	0.68	0.64	0.64	0.65	0.65	0.61	0.58	0.63	0.67	0.68
	NB	0.31	0.58	0.65	0.69	0.64	0.64	0.67	0.70	0.66	0.68	0.69	0.70	0.64	0.65	0.67	0.69	0.71
	NBM	0.29	0.59	0.73	0.73	0.66	0.67	0.73	0.74	0.70	0.72	0.73	0.73	0.66	0.65	0.73	**0.74**	0.74
	KNN	0.29	0.69	0.71	0.70	0.57	0.56	0.69	0.69	0.58	0.69	0.69	0.70	0.57	0.56	0.68	0.70	0.71
	SVM	0.29	0.66	0.73	0.74	0.67	0.69	0.72	0.74	0.70	0.72	0.73	0.73	0.67	0.69	0.71	0.74	0.72
	J48	0.29	0.52	0.66	0.70	0.54	0.60	0.69	0.70	0.58	0.66	0.69	0.70	0.54	0.62	0.68	0.72	0.71
DS-E3	RT	0.36	0.50	0.59	0.61	0.55	0.61	0.55	0.67	0.58	0.64	0.65	0.66	0.58	0.58	0.62	0.65	0.59
	NB	0.36	0.50	0.64	0.67	0.67	0.66	0.67	0.68	0.64	0.67	0.67	0.68	0.67	0.66	0.66	0.68	0.63
	NBM	0.36	0.52	0.68	0.70	0.66	0.67	0.69	0.70	0.67	0.69	**0.71**	0.71	0.64	0.65	0.68	0.68	0.69
	KNN	0.36	0.64	0.66	0.68	0.62	0.66	0.67	0.69	0.63	0.68	0.68	0.69	0.59	0.64	0.64	0.67	0.52
	SVM	0.36	0.50	0.67	0.70	0.64	0.65	0.65	0.68	0.66	0.67	0.69	0.69	0.65	0.64	0.67	0.69	0.70
	J48	0.36	0.43	0.50	0.64	0.59	0.56	0.60	0.66	0.63	0.64	0.65	0.65	0.55	0.56	0.61	0.62	0.59
DS-E4	RT	0.44	0.69	0.73	0.77	0.78	0.74	0.76	0.79	0.64	0.74	0.76	0.75	0.55	0.64	0.68	0.77	0.74
	NB	0.45	0.58	0.64	0.67	0.65	0.64	0.67	0.69	0.55	0.67	0.71	0.69	0.49	0.54	0.62	0.67	0.68
	NBM	0.51	0.71	0.77	0.79	0.78	0.77	0.78	0.78	0.66	0.76	0.78	0.78	0.65	0.73	0.77	0.78	0.80
	KNN	0.55	0.74	0.76	0.77	0.77	0.77	0.77	0.77	0.65	0.76	0.77	0.77	0.64	0.75	0.76	0.77	**0.81**
	SVM	0.53	0.71	0.74	0.75	0.76	0.76	0.76	0.76	0.69	0.75	0.77	0.77	0.63	0.71	0.74	0.77	0.78
	J48	0.41	0.59	0.71	0.77	0.78	0.76	0.77	0.76	0.54	0.79	0.79	0.80	0.47	0.58	0.76	0.78	0.80
DS-M5	RT	0.64	0.77	0.73	0.76	0.75	0.74	0.76	0.75	0.74	0.73	0.80	0.76	0.70	0.74	0.74	0.77	0.78
	NB	0.62	0.72	0.77	0.78	0.66	0.72	0.76	0.78	0.80	0.81	0.80	0.79	0.65	0.72	0.78	0.78	0.68
	NBM	0.65	0.77	0.81	0.81	0.74	0.79	0.80	0.82	0.81	0.83	0.83	0.82	0.71	0.78	0.81	0.81	0.82
	KNN	0.70	0.81	0.81	0.82	0.61	0.71	0.82	0.82	0.68	0.78	0.74	0.82	0.62	0.74	0.79	0.81	0.83
	SVM	0.60	0.79	0.82	0.82	0.66	0.77	0.81	0.81	0.82	0.82	0.81	0.82	0.65	0.78	0.82	0.82	0.80
	J48	0.51	0.78	0.82	**0.84**	0.63	0.81	0.83	0.84	0.82	0.84	0.83	0.83	0.53	0.82	0.83	0.84	0.83
DS-M6	RT	0.59	0.65	0.79	0.76	0.60	0.76	0.84	0.81	0.58	0.81	0.81	0.78	0.61	0.70	0.74	0.77	0.81
	NB	0.74	0.79	0.77	0.79	0.78	0.81	0.81	0.81	0.75	0.83	0.83	0.80	0.78	0.82	0.82	0.81	0.75
	NBM	0.74	0.78	0.76	0.78	0.74	0.77	0.78	0.76	0.75	0.76	0.76	0.77	0.73	0.78	0.78	0.77	0.82
	KNN	0.63	0.66	0.72	0.69	0.60	0.69	0.69	0.70	0.33	0.69	0.69	0.69	0.67	0.69	0.69	0.71	0.51
	SVM	0.67	0.77	0.81	0.83	0.69	0.81	0.83	0.82	0.74	0.83	0.83	0.82	0.66	0.82	0.82	0.82	0.82
	J48	0.67	0.73	0.83	0.82	0.64	0.82	0.82	0.82	0.80	**0.84**	**0.84**	0.82	0.63	0.74	0.82	0.82	0.81
DS-G7	RT	0.76	0.81	0.80	0.81	0.73	0.76	0.79	0.80	0.76	0.81	0.80	0.81	0.73	0.77	0.80	0.80	0.82
	NB	0.80	0.82	0.82	0.81	0.76	0.78	0.80	0.81	0.80	0.82	0.82	0.81	0.76	0.77	0.80	0.81	0.79
	NBM	0.84	0.86	**0.87**	0.86	0.81	0.84	0.85	0.86	0.84	0.86	**0.87**	0.86	0.81	0.85	0.86	0.86	0.84
	KNN	0.81	0.82	0.82	0.82	0.77	0.81	0.82	0.82	0.81	0.82	0.82	0.82	0.77	0.80	0.81	0.82	0.82
	SVM	0.84	0.85	0.85	0.85	0.77	0.81	0.84	0.85	0.84	0.85	0.85	0.85	0.77	0.80	0.84	0.85	0.85
	J48	0.82	0.83	0.83	0.83	0.75	0.77	0.82	0.83	0.82	0.83	0.83	0.83	0.75	0.77	0.82	0.83	0.82
DS-G8	RT	0.60	0.69	0.71	0.73	0.68	0.69	0.71	0.73	0.67	0.71	0.71	0.72	0.68	0.68	0.72	0.74	0.72
	NB	0.64	0.70	0.70	0.72	0.70	0.70	0.72	0.73	0.72	0.74	0.73	0.73	0.70	0.70	0.71	0.73	0.74
	NBM	0.51	0.75	0.79	0.79	0.76	0.76	0.79	**0.79**	0.73	0.77	0.79	0.79	0.76	0.76	0.79	0.79	0.78
	KNN	0.67	0.73	0.73	0.74	0.73	0.72	0.74	0.74	0.71	0.73	0.73	0.74	0.73	0.72	0.74	0.74	0.73
	SVM	0.63	0.72	0.77	0.76	0.72	0.71	0.76	0.77	0.72	0.75	0.76	0.77	0.72	0.72	0.76	0.77	0.76
	J48	0.61	0.69	0.72	0.75	0.68	0.68	0.72	0.74	0.66	0.74	0.75	0.75	0.68	0.68	0.72	0.75	0.75
DS-G9	RT	0.31	0.73	0.77	0.79	0.72	0.76	0.77	0.79	0.67	0.74	0.75	0.77	0.73	0.76	0.77	0.72	0.74
	NB	0.44	0.80	0.81	0.82	0.80	0.81	0.82	0.82	0.71	0.81	0.82	0.83	0.80	0.81	0.81	0.82	0.77
	NBM	0.29	0.82	0.84	0.84	0.82	0.83	0.83	0.84	0.76	0.83	0.84	0.84	0.82	0.83	0.84	**0.84**	0.82
	KNN	0.42	0.77	0.77	0.77	0.73	0.76	0.77	0.77	0.69	0.74	0.76	0.76	0.73	0.75	0.74	0.76	0.77
	SVM	0.30	0.77	0.81	0.83	0.79	0.79	0.81	0.83	0.73	0.80	0.82	0.82	0.79	0.80	0.82	0.83	0.83
	J48	0.29	0.77	0.79	0.79	0.73	0.78	0.79	0.79	0.70	0.77	0.78	0.79	0.74	0.76	0.79	0.79	0.77
DS-G10	RT	0.71	0.78	0.80	0.81	0.77	0.78	0.81	0.83	0.76	0.79	0.80	0.83	0.77	0.78	0.81	0.83	0.83
	NB	0.78	0.80	0.82	0.82	0.78	0.81	0.82	0.82	0.80	0.80	0.82	0.82	0.78	0.81	0.82	0.82	0.78
	NBM	0.80	0.86	0.86	0.86	0.84	0.86	**0.87**	0.86	0.82	0.86	0.86	0.86	0.84	0.86	**0.87**	0.86	0.84
	KNN	0.80	0.81	0.82	0.81	0.80	0.81	0.82	0.81	0.80	0.81	0.82	0.82	0.80	0.81	0.82	0.81	0.79
	SVM	0.76	0.84	0.85	0.85	0.79	0.84	0.85	0.85	0.79	0.83	0.85	0.85	0.79	0.84	0.85	0.85	0.86
	J48	0.71	0.79	0.83	0.83	0.77	0.78	0.82	0.83	0.73	0.79	0.83	0.83	0.77	0.78	0.82	0.83	0.84

is too small. For all datasets, the highest accuracy of identifying the severity by using four FS approaches (OneR, IG, CHI, RF) is higher than original dataset 1.50%. Identifying the severity by using four IS approaches (CNN, MCS, ENN, ICF) and original datasets can get almost the same performance. In addition, the scales of datasets (including the number of words and the number of bug reports) has decreased. From Tables 3-6, we can get that using FS and IS with classifiers perform the better performance than original datasets for identifying severity of bug reports.

TABLE 6. The *F-measure* of variants to identify severity of bug reports by using instance selection.

IS(F-measure)		ENN				MCS				ICF				CNN				Original
		30%	50%	70%	90%	30%	50%	70%	90%	30%	50%	70%	90%	30%	50%	70%	90%	
DS-E1	RT	0.21	0.63	0.69	0.69	0.66	0.64	0.69	0.72	0.62	0.68	0.68	0.70	0.67	0.67	0.70	0.68	0.73
	NB	0.29	0.66	0.74	0.75	0.71	0.72	0.73	0.76	0.67	0.74	0.74	0.76	0.69	0.70	0.74	0.75	0.70
	NBM	0.15	0.69	0.77	0.78	0.71	0.73	0.76	0.77	0.67	0.71	0.75	0.78	0.71	0.74	0.77	**0.78**	0.73
	KNN	0.23	0.59	0.68	0.68	0.68	0.65	0.67	0.69	0.61	0.66	0.67	0.68	0.65	0.64	0.67	0.68	0.71
	SVM	0.19	0.70	0.75	0.76	0.72	0.72	0.75	0.76	0.68	0.72	0.73	0.76	0.72	0.74	0.75	0.76	0.75
	J48	0.12	0.67	0.74	0.76	0.64	0.71	0.75	0.75	0.59	0.69	0.72	0.75	0.61	0.71	0.74	0.75	0.75
DS-E2	RT	0.16	0.61	0.66	0.66	0.61	0.63	0.66	0.67	0.64	0.64	0.65	0.65	0.61	0.60	0.64	0.67	0.68
	NB	0.18	0.60	0.66	0.69	0.65	0.66	0.68	0.70	0.66	0.68	0.69	0.70	0.65	0.66	0.68	0.69	0.71
	NBM	0.13	0.60	0.73	0.73	0.67	0.69	0.74	0.74	0.70	0.72	0.73	0.73	0.67	0.67	0.73	**0.74**	0.72
	KNN	0.14	0.68	0.67	0.65	0.58	0.58	0.68	0.65	0.57	0.65	0.65	0.65	0.58	0.58	0.66	0.66	0.69
	SVM	0.14	0.68	0.73	0.74	0.69	0.70	0.73	0.73	0.69	0.71	0.72	0.72	0.69	0.70	0.72	0.73	0.70
	J48	0.13	0.53	0.66	0.67	0.56	0.61	0.67	0.68	0.58	0.65	0.67	0.68	0.56	0.63	0.67	0.68	0.65
DS-E3	RT	0.19	0.49	0.58	0.61	0.56	0.61	0.55	0.66	0.58	0.63	0.63	0.65	0.59	0.58	0.62	0.64	0.59
	NB	0.19	0.47	0.64	0.68	0.67	0.66	0.68	0.68	0.64	0.67	0.67	0.68	0.67	0.66	0.66	0.69	0.68
	NBM	0.19	0.49	0.69	**0.71**	0.66	0.67	0.70	0.70	0.68	0.69	0.71	0.71	0.65	0.66	0.69	0.69	0.70
	KNN	0.19	0.62	0.63	0.64	0.56	0.61	0.64	0.65	0.61	0.62	0.63	0.64	0.56	0.59	0.62	0.64	0.53
	SVM	0.19	0.48	0.67	0.70	0.65	0.65	0.65	0.68	0.66	0.67	0.68	0.69	0.66	0.65	0.67	0.69	0.70
	J48	0.19	0.34	0.49	0.63	0.57	0.56	0.60	0.65	0.62	0.62	0.61	0.61	0.53	0.55	0.61	0.62	0.59
DS-E4	RT	0.47	0.70	0.71	0.75	0.75	0.71	0.73	0.76	0.65	0.72	0.73	0.72	0.59	0.65	0.69	0.74	0.74
	NB	0.49	0.62	0.66	0.68	0.67	0.66	0.68	0.70	0.57	0.68	0.70	0.69	0.54	0.59	0.65	0.68	0.70
	NBM	0.55	0.72	0.77	**0.78**	0.77	0.76	0.77	0.77	0.67	0.73	0.75	0.75	0.67	0.73	0.75	0.76	0.78
	KNN	0.59	0.70	0.71	0.72	0.72	0.72	0.72	0.72	0.64	0.70	0.71	0.72	0.66	0.72	0.71	0.72	0.74
	SVM	0.57	0.72	0.73	0.72	0.73	0.73	0.73	0.73	0.69	0.72	0.73	0.74	0.65	0.72	0.72	0.74	0.74
	J48	0.44	0.63	0.69	0.73	0.73	0.73	0.73	0.72	0.55	0.72	0.72	0.73	0.51	0.61	0.71	0.71	0.71
DS-M5	RT	0.67	0.78	0.75	0.76	0.76	0.75	0.76	0.75	0.74	0.74	0.78	0.76	0.72	0.75	0.75	0.76	0.76
	NB	0.67	0.75	0.79	0.79	0.70	0.75	0.78	0.79	0.79	0.79	0.80	0.80	0.69	0.75	0.79	0.79	0.72
	NBM	0.69	0.79	0.81	0.81	0.76	0.80	0.81	**0.82**	0.78	0.80	0.81	0.81	0.75	0.80	0.80	0.81	0.80
	KNN	0.70	0.78	0.78	0.79	0.65	0.72	0.78	0.78	0.69	0.78	0.75	0.78	0.66	0.74	0.78	0.79	0.78
	SVM	0.65	0.80	0.81	0.80	0.70	0.78	0.80	0.79	0.79	0.79	0.79	0.80	0.70	0.79	0.80	0.79	0.77
	J48	0.56	0.76	0.77	0.78	0.66	0.78	0.78	0.78	0.75	0.76	0.76	0.77	0.58	0.79	0.77	0.78	0.78
DS-M6	RT	0.64	0.68	0.80	0.76	0.64	0.78	0.83	0.78	0.59	0.79	0.79	0.77	0.64	0.74	0.75	0.77	0.80
	NB	0.77	0.81	0.79	0.80	0.81	0.82	0.82	0.82	0.75	0.81	0.81	0.81	0.81	**0.84**	0.83	0.82	0.78
	NBM	0.78	0.80	0.79	0.80	0.77	0.79	0.80	0.78	0.75	0.76	0.76	0.79	0.76	0.80	0.80	0.79	0.81
	KNN	0.68	0.71	0.75	0.73	0.64	0.73	0.73	0.73	0.37	0.72	0.72	0.73	0.72	0.73	0.73	0.74	0.56
	SVM	0.72	0.79	0.82	0.83	0.73	0.82	0.82	0.81	0.75	0.81	0.81	0.82	0.71	0.83	0.82	0.81	0.81
	J48	0.71	0.76	0.82	0.80	0.69	0.81	0.80	0.78	0.76	0.77	0.77	0.80	0.68	0.76	0.80	0.78	0.77
DS-G7	RT	0.76	0.80	0.79	0.81	0.76	0.78	0.80	0.80	0.76	0.80	0.79	0.81	0.76	0.78	0.80	0.80	0.82
	NB	0.81	0.82	0.83	0.82	0.78	0.80	0.82	0.82	0.81	0.82	0.83	0.82	0.78	0.80	0.82	0.82	0.81
	NBM	0.82	0.84	0.87	0.86	0.83	0.85	0.86	0.86	0.82	0.84	0.87	0.86	0.83	0.86	0.86	**0.87**	0.84
	KNN	0.79	0.79	0.80	0.79	0.79	0.80	0.80	0.80	0.79	0.79	0.80	0.79	0.79	0.80	0.80	0.80	0.79
	SVM	0.83	0.83	0.84	0.85	0.79	0.82	0.84	0.85	0.83	0.83	0.84	0.85	0.79	0.82	0.84	0.84	0.84
	J48	0.77	0.78	0.81	0.81	0.78	0.79	0.82	0.82	0.77	0.78	0.81	0.81	0.78	0.79	0.82	0.82	0.81
DS-G8	RT	0.62	0.70	0.71	0.73	0.69	0.70	0.72	0.73	0.67	0.69	0.71	0.71	0.69	0.70	0.73	0.73	0.71
	NB	0.66	0.72	0.72	0.74	0.72	0.72	0.73	0.74	0.72	0.74	0.74	0.74	0.72	0.72	0.73	0.74	0.74
	NBM	0.50	0.76	**0.79**	0.79	0.77	0.77	0.79	0.79	0.72	0.74	0.78	0.79	0.77	0.77	0.79	0.79	0.77
	KNN	0.69	0.73	0.71	0.71	0.72	0.72	0.72	0.71	0.68	0.68	0.70	0.71	0.72	0.72	0.72	0.71	0.69
	SVM	0.64	0.74	0.77	0.76	0.74	0.73	0.77	0.76	0.71	0.73	0.76	0.76	0.74	0.73	0.76	0.76	0.74
	J48	0.62	0.70	0.74	0.73	0.70	0.70	0.74	0.73	0.64	0.63	0.72	0.73	0.70	0.70	0.73	0.74	0.76
DS-G9	RT	0.18	0.74	0.77	0.78	0.72	0.76	0.77	0.78	0.67	0.74	0.75	0.77	0.74	0.77	0.77	0.72	0.73
	NB	0.40	0.81	0.82	0.82	0.81	0.82	0.82	0.82	0.71	0.81	0.83	0.83	0.81	0.82	0.82	0.83	0.78
	NBM	0.14	0.83	0.84	0.84	0.83	0.83	0.83	0.84	0.74	0.83	0.84	0.84	0.83	0.83	0.84	**0.84**	0.82
	KNN	0.38	0.76	0.76	0.74	0.71	0.75	0.75	0.74	0.65	0.72	0.74	0.73	0.71	0.74	0.73	0.73	0.75
	SVM	0.15	0.78	0.82	0.83	0.79	0.80	0.81	0.83	0.72	0.79	0.82	0.82	0.80	0.81	0.82	0.83	0.83
	J48	0.13	0.78	0.79	0.80	0.74	0.79	0.80	0.80	0.67	0.77	0.78	0.78	0.75	0.77	0.80	0.80	0.78
DS-G10	RT	0.74	0.79	0.80	0.80	0.79	0.79	0.80	0.82	0.77	0.79	0.80	0.82	0.79	0.79	0.80	0.82	0.82
	NB	0.80	0.82	0.83	0.83	0.80	0.82	0.84	0.83	0.81	0.82	0.83	0.83	0.80	0.82	0.84	0.83	0.79
	NBM	0.82	0.87	0.86	0.86	0.85	0.87	**0.87**	0.87	0.81	0.87	0.87	0.86	0.85	0.87	**0.87**	0.87	0.83
	KNN	0.80	0.79	0.80	0.79	0.78	0.79	0.79	0.78	0.78	0.79	0.79	0.79	0.78	0.79	0.79	0.78	0.74
	SVM	0.78	0.84	0.85	0.84	0.81	0.85	0.85	0.84	0.80	0.84	0.85	0.84	0.81	0.85	0.85	0.84	0.86
	J48	0.74	0.81	0.82	0.81	0.79	0.80	0.82	0.82	0.73	0.80	0.83	0.82	0.79	0.80	0.82	0.82	0.82

Furthermore, for each FS and IS approach, it is possible that different ratios can get different results. Thus, in the follow experiments, we set the percentages of selected words and bug reports based on the percentages, which get best performance in this experiment.

RQ2. Compare with individual FS and IS, what the performance of data reduction (combine FS and IS) to identify the severity of bug reports?

According to RQ1, we can see that FS can reduce the number of words and IS can reduce the number of bug reports,

and both FS and IS can improve the performance of identify the severity of bug reports. Compare with individual FS and IS, we want to reduce bug report dimension and the word dimension at the same time. Thus, in this research question, we employ combination of FS and IS with six classification algorithms (*RT*, *NB*, *NBM*, *KNN*, *SVM*, *J48*) to get small-scale and high-quality training reduced set, which is based on the basis results of RQ1. We use two evaluation metrics mentioned above (accuracy, and *F-measure*) to compare the performance of FS, IS, FS→IS, and IS→FS. We bold the best results of feature selection and instance selection for each variant.

From Tables 7 and 8, we can see that the best performance of identify the severity of bug reports by using FS→IS and IS→FS are better than performance of using individual FS, IS, and original datasets, except DS-E3, and the accuracy of DS-E3 by using FS→IS and IS→FS is 0.68, and 0.67, and the accuracy of original DS-E3 is 0.70. In addition, for most datasets, the performance of using FS→IS is better than IS→FS, except DS-E4, and the accuracy of DS-E4 by using FS→IS is 0.80, and the accuracy of DS-E4 by using IS→FS is 0.82. The average performance of identifying the severity of bug reports from high to low is FS→IS, FS, IS, and IS→FS, respectively. However, for DS-E4, IS→FS could get the highest performance of identify the severity of bug reports, the accuracy of IS→FS is 0.82 and the F-measure of IS→FS is 0.78.

TABLE 7. The accuracy of identify severity of bug reports by using data reduction.

Dataset	ACC				
	FS-IS	IS-FS	FS	IS	Original
DS-E1	**0.80**	0.74	0.79	0.78	0.78
DS-E2	**0.76**	0.69	0.75	0.74	0.74
DS-E3	0.68	0.67	**0.71**	0.70	0.70
DS-E4	0.80	**0.82**	0.81	0.79	0.81
DS-M5	0.84	0.83	**0.84**	0.82	0.83
DS-M6	**0.84**	0.83	**0.84**	0.82	0.82
DS-G7	**0.87**	0.82	0.86	0.86	0.85
DS-G8	**0.80**	0.70	0.79	0.79	0.78
DS-G9	**0.84**	0.77	**0.84**	0.84	0.83
DS-G10	**0.88**	0.78	0.86	0.87	0.86

TABLE 8. The *F-measure* of identify severity of bug reports by using data reduction.

Dataset	F-measure				
	FS-IS	IS-FS	FS	IS	Original
DS-E1	**0.79**	0.72	0.79	0.78	0.75
DS-E2	**0.75**	0.68	0.74	0.74	0.72
DS-E3	0.69	0.67	**0.71**	0.71	0.70
DS-E4	0.77	**0.78**	0.77	0.78	0.78
DS-M5	0.82	0.76	**0.82**	0.82	0.80
DS-M6	**0.84**	0.78	**0.84**	0.84	0.81
DS-G7	**0.87**	0.82	0.87	0.86	0.84
DS-G8	**0.79**	0.71	0.79	**0.79**	0.77
DS-G9	**0.84**	0.77	0.84	**0.84**	0.83
DS-G10	**0.87**	0.79	0.86	0.87	0.86

From the Tables 7 and 8, we can get that, FS→IS could not only reduce bug report dimension and the word dimension at the same time, but also improve the performance of identify the severity of bug reports. Compare to FS→IS, FS, and IS, the IS→FS get the worse performance, which is caused by the bug reports removed after IS includes large number of words, and the available information to identify the severity of bug reports is too small after FS.

RQ3. Can RSMOTE improve the performance of identifying the severity of bug reports with an imbalanced distribution?

Typical classification techniques may be adversely affected when the distribution of the bug reports is imbalanced. In order to solve this problem, in this study, we propose RSMOTE to weaken the imbalanced degree of bug reports. Thus, in this research question, we want to investigate which variants of imbalanced learning strategy and classifier perform the best for identifying the severity of bug reports. We consider five imbalanced learning strategies (RSMOTE, RUS, ROS, SMOTE and CMA) and six classification algorithms (*NB*, *NBM*, *SVM*, *KNN*, *RT* and *J48*), we can have totally 30 variants (i.e., combinations of one of the imbalanced learning strategies and one of the classification algorithms). Therefore, in this research question, we want to investigate which variants perform the best for identifying the severity of bug reports. Tables 9-10 present the performance of RSMOTE and the imbalanced learning strategies with six classifiers, respectively. We use the two evaluation metrics mentioned above (accuracy, and *F-measure*) to compare the totally 30 variants.

In Tables 9-10, to show the variants with the best performance, we highlight the highest result values in bold based on the results of the six classifiers for each imbalanced learning strategy. The highest classification results for each imbalanced learning strategy are duplicated in the MAX_ACC column, and the highest results in the MAX_ACC column are further highlighted in bold. To illustrate the overall improvement enabled by each imbalanced learning strategy for all six classifiers, the average of the results obtained by all six classifiers using each imbalanced learning strategy is reported in the AVG_ACC column, and the highest results in the AVG_ACC column are further highlighted in bold.

From Tables 9-10, we observe that RSMOTE approach could get maximum performance (MAX_ACC column) of identify the severity of bug reports with imbalance distribution than RUS, ROS, SMOTE and CMA. The RSMOTE could achieve the average maximum accuracy is 0.81, which was higher than imbalanced learning strategies (SMOTE, ROS, RUS, CMA, Original) by 3.83%, 5.02%, 8.93%, 2.19%, and 3.10%, respectively. The AVG_ACC column shows the generalization ability of ILS to balance the imbalanced distribution of bug reports. We can observe that the RSMOTE could achieve the average accuracy is 0.76, which was higher than imbalanced learning strategies (SMOTE, ROS, RUS, CMA, Original) by 3.01%, 5.10%, 11.84%, 3.01%, and 2.83%, respectively. Thus, it could represent that the balanced dataset by RSMOTE approach have better generalization capabilities than Original, SMOTE, RUS, and ROS.

TABLE 9. The accuracy of ILS performing variants for identifying the severity of bug reports.

ACC		RT	NB	NBM	KNN	SVM	J48	MAX_ACC	AVG_ACC
DS-E1	Original	0.75	0.69	0.74	0.73	0.76	**0.78**	0.78	0.74
	RUS	0.63	0.69	0.71	0.70	**0.72**	0.55	0.72	0.67
	ROS	0.69	0.67	0.73	0.73	0.75	**0.76**	0.76	0.72
	CMA	0.72	0.77	0.72	0.67	0.75	**0.78**	0.78	0.73
	SMOTE	0.71	0.75	0.71	0.67	0.75	**0.76**	0.76	0.72
	RSMOTE	0.72	0.74	0.69	0.75	**0.79**	0.79	**0.79**	**0.75**
DS-E2	Original	0.68	0.71	**0.74**	0.71	0.72	0.71	0.74	0.71
	RUS	0.60	0.68	**0.68**	0.67	0.66	0.55	0.68	0.64
	ROS	0.70	0.68	0.71	0.71	**0.72**	0.69	0.72	0.70
	CMA	0.69	0.71	0.73	0.63	**0.74**	0.65	0.74	0.69
	SMOTE	0.63	0.71	**0.73**	0.65	0.73	0.71	0.73	0.70
	RSMOTE	0.70	0.72	**0.77**	0.68	0.76	0.68	**0.77**	**0.72**
DS-E3	Original	0.59	0.63	0.69	0.52	**0.70**	0.59	0.70	0.62
	RUS	0.55	0.65	**0.66**	0.52	0.63	0.54	0.66	0.59
	ROS	0.67	0.67	0.68	0.52	**0.69**	0.62	0.69	0.64
	CMA	0.70	**0.72**	0.68	0.43	0.71	0.64	0.72	0.65
	SMOTE	0.68	0.70	0.67	0.52	**0.72**	0.58	0.72	0.65
	RSMOTE	0.67	0.72	0.72	0.53	**0.76**	0.63	**0.76**	**0.68**
DS-E4	Original	0.74	0.68	0.80	**0.81**	0.78	0.80	0.81	0.77
	RUS	0.62	0.56	**0.73**	0.73	0.66	0.53	0.73	0.64
	ROS	0.77	0.66	0.76	**0.81**	0.75	0.74	0.81	0.75
	CMA	0.80	0.79	**0.81**	0.74	**0.81**	0.77	0.81	0.79
	SMOTE	**0.80**	0.80	0.76	0.74	0.79	0.76	0.80	0.77
	RSMOTE	0.81	0.81	0.80	0.74	**0.85**	0.80	**0.85**	**0.80**
DS-M5	Original	0.78	0.68	0.82	0.83	0.80	**0.83**	0.83	0.79
	RUS	0.58	0.70	**0.73**	0.64	0.69	0.52	0.73	0.64
	ROS	0.78	0.67	0.78	**0.83**	0.80	0.58	0.83	0.74
	CMA	0.86	0.83	0.82	0.74	0.83	**0.87**	0.87	0.82
	SMOTE	0.78	0.82	0.81	0.78	0.83	**0.84**	0.84	0.81
	RSMOTE	0.85	0.85	0.83	0.77	**0.88**	0.87	**0.88**	**0.84**
DS-M6	Original	0.81	0.75	**0.82**	0.51	0.82	0.81	0.82	0.75
	RUS	0.62	**0.83**	0.82	0.44	0.68	0.64	0.83	0.67
	ROS	0.87	0.75	0.83	0.50	0.85	**0.89**	0.89	0.78
	CMA	0.86	**0.89**	0.85	0.33	0.85	**0.89**	0.89	0.78
	SMOTE	0.87	0.86	0.80	0.40	0.86	**0.88**	0.88	0.78
	RSMOTE	0.87	0.87	0.81	0.42	0.90	**0.92**	**0.92**	**0.80**
DS-G7	Original	0.82	0.79	0.84	0.82	**0.85**	0.82	**0.85**	**0.82**
	RUS	0.70	0.77	0.79	**0.80**	0.78	0.73	0.80	0.76
	ROS	0.82	0.79	0.81	0.81	**0.83**	0.77	0.83	0.80
	CMA	0.81	0.82	0.82	0.80	**0.84**	0.83	0.84	0.82
	SMOTE	0.80	**0.82**	0.78	0.79	0.79	0.81	0.82	0.80
	RSMOTE	0.81	0.81	0.79	**0.83**	0.83	0.83	0.83	0.82
DS-G8	Original	0.72	0.74	**0.78**	0.73	0.76	0.75	0.78	0.75
	RUS	0.68	0.72	**0.74**	0.74	0.72	0.67	0.74	0.71
	ROS	0.72	0.73	**0.75**	0.73	0.74	0.74	0.75	0.73
	CMA	0.77	0.74	0.76	**0.80**	0.75	0.71	0.80	0.75
	SMOTE	0.73	0.73	0.73	**0.77**	0.74	0.72	0.77	0.73
	RSMOTE	0.76	0.72	0.76	**0.82**	0.75	0.73	**0.82**	**0.76**
DS-G9	Original	0.74	0.77	0.82	0.77	**0.83**	0.77	0.83	0.78
	RUS	0.70	0.75	**0.82**	0.75	0.80	0.73	0.82	0.76
	ROS	0.76	0.76	**0.83**	0.76	0.82	0.76	0.83	0.78
	CMA	0.75	0.77	0.79	**0.82**	0.81	0.76	0.82	0.78
	SMOTE	0.75	0.77	**0.83**	0.76	0.83	0.78	0.83	0.79
	RSMOTE	0.76	0.77	0.83	0.83	**0.86**	0.76	**0.86**	**0.80**
DS-G10	Original	0.83	0.78	0.84	0.79	**0.86**	0.84	0.86	0.82
	RUS	0.76	0.77	0.81	0.78	**0.82**	0.78	0.82	0.79
	ROS	0.82	0.77	0.80	0.79	**0.86**	0.84	0.86	0.81
	CMA	0.83	0.80	0.80	**0.86**	0.82	0.85	0.86	0.83
	SMOTE	0.82	0.80	0.79	0.81	**0.86**	0.83	0.86	0.82
	RSMOTE	0.83	0.79	0.80	0.87	**0.87**	0.85	**0.87**	**0.83**

From Tables 9-10, we can find that RUS get the worse performance to balance the imbalance distribution. This is due to RUS removes the majority-class samples in dataset, which could loss information of original dataset. From Tables 9-10, we can see that *SVM* achieves the highest average performance and *NB* get the worse performance to identify the

TABLE 10. The *F-measure* of ILS performing variants for identifying the severity of bug reports.

F-measure		RT	NB	NBM	KNN	SVM	J48	MAX_ACC	AVG_ACC
DS-E1	Original	0.73	0.70	0.73	0.71	**0.75**	0.75	0.75	0.73
	RUS	0.64	0.70	0.72	0.70	**0.73**	0.57	0.73	0.68
	ROS	0.69	0.69	0.74	0.71	0.75	**0.76**	0.76	0.72
	CMA	0.72	0.77	0.69	0.66	0.71	**0.78**	0.78	0.72
	SMOTE	0.71	0.74	0.72	0.68	**0.76**	0.75	0.76	0.73
	RSMOTE	0.72	0.74	0.70	0.70	**0.80**	0.78	**0.80**	**0.74**
DS-E2	Original	0.68	0.71	**0.72**	0.69	0.70	0.65	0.72	0.69
	RUS	0.62	0.69	**0.69**	0.67	0.67	0.56	0.69	0.65
	ROS	0.70	0.69	**0.72**	0.69	0.71	0.66	0.72	0.69
	CMA	0.68	**0.72**	0.69	0.63	0.69	0.66	0.72	0.68
	SMOTE	0.64	0.70	**0.74**	0.67	0.73	0.68	0.74	0.69
	RSMOTE	0.70	0.72	**0.78**	0.69	0.77	0.66	**0.78**	**0.72**
DS-E3	Original	0.59	0.68	0.70	0.53	**0.70**	0.59	0.70	0.63
	RUS	0.55	0.65	**0.67**	0.52	0.63	0.54	0.67	0.59
	ROS	0.67	0.67	0.69	0.53	**0.69**	0.61	0.69	0.64
	CMA	0.70	0.70	0.66	0.43	**0.70**	0.63	0.70	0.64
	SMOTE	0.68	0.68	0.68	0.51	**0.72**	0.59	0.72	0.64
	RSMOTE	0.68	0.71	0.73	0.50	**0.77**	0.63	**0.77**	**0.67**
DS-E4	Original	0.74	0.70	**0.78**	0.74	0.74	0.71	0.78	0.73
	RUS	0.64	0.60	**0.75**	0.70	0.69	0.57	0.75	0.66
	ROS	0.75	0.70	**0.76**	0.74	0.72	0.73	0.76	0.73
	CMA	**0.80**	0.78	0.78	0.74	0.73	0.77	0.78	0.77
	SMOTE	0.71	0.78	0.77	0.74	**0.78**	0.74	0.78	0.75
	RSMOTE	0.80	0.81	0.82	0.76	**0.85**	0.77	**0.85**	**0.80**
DS-M5	Original	0.76	0.72	**0.80**	0.78	0.77	0.78	0.80	0.77
	RUS	0.63	0.73	**0.75**	0.68	0.73	0.57	0.75	0.68
	ROS	0.75	0.71	0.78	**0.78**	0.78	0.64	0.78	0.74
	CMA	0.82	0.81	0.79	0.77	0.77	**0.84**	0.84	0.80
	SMOTE	0.80	0.80	0.81	0.80	**0.82**	0.80	0.82	0.81
	RSMOTE	0.85	0.84	0.85	0.79	**0.88**	0.85	**0.88**	**0.84**
DS-M6	Original	0.80	0.78	**0.81**	0.56	0.81	0.77	0.81	0.76
	RUS	0.67	**0.85**	0.84	0.48	0.72	0.69	0.85	0.71
	ROS	0.84	0.78	0.84	0.55	0.84	**0.89**	0.89	0.79
	CMA	0.86	**0.90**	0.79	0.34	0.79	0.89	0.90	0.76
	SMOTE	0.86	0.86	0.82	0.44	0.86	**0.88**	0.88	0.79
	RSMOTE	0.86	0.88	0.83	0.45	0.91	**0.91**	**0.91**	**0.81**
DS-G7	Original	0.82	0.81	0.84	0.79	**0.84**	0.81	0.84	0.82
	RUS	0.74	0.79	**0.81**	0.79	0.81	0.76	0.81	0.78
	ROS	0.81	0.81	0.82	0.79	**0.83**	0.79	0.83	0.81
	CMA	0.82	0.82	0.79	0.82	0.79	**0.83**	0.83	0.81
	SMOTE	0.81	0.81	0.80	0.80	0.81	**0.82**	0.82	0.81
	RSMOTE	0.82	0.81	0.81	**0.85**	0.84	0.83	**0.85**	**0.83**
DS-G8	Original	0.71	0.74	**0.77**	0.69	0.74	0.76	0.77	0.74
	RUS	0.70	0.72	**0.74**	0.73	0.74	0.68	0.74	0.72
	ROS	0.70	0.73	0.75	0.68	0.74	**0.76**	0.76	0.73
	CMA	0.77	0.72	0.72	**0.81**	0.69	0.72	0.81	0.74
	SMOTE	0.72	0.71	0.73	**0.77**	0.75	0.72	0.77	0.73
	RSMOTE	0.72	0.71	0.77	0.76	0.77	**0.83**	**0.83**	**0.76**
DS-G9	Original	0.73	0.78	0.82	0.75	**0.83**	0.78	0.83	0.78
	RUS	0.71	0.76	**0.82**	0.76	0.81	0.74	0.82	0.77
	ROS	0.76	0.77	**0.83**	0.75	0.83	0.77	0.83	0.78
	CMA	0.75	0.78	0.78	**0.82**	0.80	0.76	0.82	0.78
	SMOTE	0.76	0.77	**0.83**	0.77	0.83	0.77	0.83	0.79
	RSMOTE	0.77	0.77	0.83	0.83	**0.86**	0.76	**0.86**	**0.80**
DS-G10	Original	0.82	0.79	0.83	0.74	**0.86**	0.82	0.86	0.81
	RUS	0.78	0.79	0.83	0.75	**0.83**	0.80	0.83	0.80
	ROS	0.81	0.79	0.81	0.74	**0.86**	0.84	0.86	0.81
	CMA	0.83	0.80	0.78	**0.85**	0.78	0.84	0.85	0.81
	SMOTE	0.81	0.80	0.80	0.81	**0.86**	0.83	0.86	0.82
	RSMOTE	0.83	0.80	0.81	0.87	**0.88**	0.84	**0.88**	**0.84**

severity of bug reports after ILS balance the imbalance distribution. The highest average accuracy is 0.78, which was higher than other classifiers (*RT*, *NB*, *NBM*, *KNN*, *J48*) 4.87%, 4.27%, 1.20%, 11.16%, and 4.97%, respectively.

Furthermore, different variants have significantly different performance of identifying the severity bug reports. For example, RSMOTE+*J48*, CMA+(*NB* or *J48*), and ROS+*J48* are the top-3 best performing variants for

identifying the severity of bug reports for DS-M6. and RSMOTE+*KNN*, CMA+*KNN*, and Original+*NBM* are the top-3 best performing variants for identifying the severity of bug reports for DS-G8. From these results, we observe that in most cases, the classifiers achieved higher performance with RSMOTE than with the other ILS (RUS, ROS, SMOTE, and CMA). Therefore, RSMOTE approach could effectively improve the performance of identifying surprise bug reports of camel and wicket.

RQ4. Can combine RSMOTE with data reduction could improve the performance of identifying the severity of bug reports.

Due to original dataset are imbalance distribution, which could affect the basic classification performance of identifying the severity of bug reports. Based on RQ3, we can get that RSMOTE approach has better generalization capabilities than ILS (SMOTE, RUS, CMA and ROS). However, the random sampling uncertainty of RSMOTE and original dataset could contain large amounts of noise data. In order to solve this problem, according to RQ2, we could observe the best scale of data reduction (FS and IS) for each dataset, which could effectively reduce the noise and improve the performance of identify the severity of bug reports. Thus, in this research question, we employ combination of data reduction with RSMOTE to identify the severity of bug reports.

To answer this question, Firstly, we use the RSMOTE approach to balance the imbalance distribution of original datasets. Then, we use data reduction (combine FS and IS) to get small-scale and high-quality training reduced set from balanced datasets by RSMOTE, which could not only reduce bug report dimension and the word dimension at the same time, but also improve the performance of identify the severity of bug reports. We select the scale of data reduction based on the previous research questions (RQ2). Finally, we use the classification algorithms, which can be any one of six popular text classification algorithms (*RT*, *NB*, *NBM*, *KNN*, *SVM*, *J48*), to build classifiers. We use the two evaluation metrics mentioned above (accuracy and F-measure) to verify the performance of combining RSMOTE with data reduction to identify the severity of bug reports. We bold the best results for each variant.

TABLE 11. The accuracy of combining RSMOTE with data reduction to identify the severity of bug reports.

Dataset	ACC				
	Original	Data Reduction	RSMOTE	RSMOTE(FS-IS)	RSMOTE(IS-FS)
DS-E1	0.78	0.80	0.79	**0.82**	0.78
DS-E2	0.74	0.76	0.77	0.74	**0.78**
DS-E3	0.70	0.68	0.76	**0.77**	0.71
DS-E4	0.81	0.82	0.85	**0.87**	0.79
DS-M5	0.83	0.84	0.88	**0.88**	0.83
DS-M6	0.82	0.84	**0.92**	0.86	0.86
DS-G7	0.85	0.87	0.83	**0.88**	0.84
DS-G8	0.78	0.80	0.82	**0.83**	0.77
DS-G9	0.83	0.84	0.86	**0.87**	0.84
DS-G10	0.86	0.88	0.87	**0.89**	0.86

From Tables 11 and 12, we can see that the best performance of identify the severity of bug reports by using

TABLE 12. The *F-measure* of combining RSMOTE with data reduction to identify the severity of bug reports.

Dataset	F-measure				
	Original	Data Reduction	RSMOTE	RSMOTE(FS-IS)	RSMOTE(IS-FS)
DS-E1	0.75	0.79	0.80	**0.80**	0.76
DS-E2	0.72	0.75	0.78	0.73	**0.78**
DS-E3	0.70	0.69	**0.77**	0.76	0.71
DS-E4	0.74	0.78	0.85	**0.85**	0.74
DS-M5	0.78	0.82	**0.88**	0.88	0.80
DS-M6	0.81	0.84	**0.91**	0.84	0.83
DS-G7	0.84	0.87	0.85	**0.88**	0.84
DS-G8	0.77	0.79	0.76	**0.80**	0.78
DS-G9	0.83	0.84	0.86	**0.86**	0.85
DS-G10	0.86	0.87	**0.88**	0.88	0.87

combine RSMOTE with data reduction (FS→IS) are better than performance of using individual data reduction, RSMOTE, RSMOTE with data reduction (IS→FS), and original datasets, except DS-E2 and DS-M6. In addition, for most datasets, the performance of RSMOTE with data reduction (FS→IS) and RSMOTE is better than RSMOTE with data reduction (IS→FS). This is caused by the bug reports removed after IS includes large number of words, and RSMOTE has little available information to balance the imbalance distribution after FS. From Tables 11 and 12, we can find that using RSMOTE with data reduction (FS→IS) to identify the severity of bug reports, most variants achieve the average values of 0.74-0.89 in terms of accuracy, and 0.73-0.88 in terms of *F-measure*.

From the Tables 11 and 12, we can get that, using RSMOTE with data reduction (FS→IS) to identify the severity of bug reports could not only balance the imbalance distribution of bug reports, but also reduce bug report dimension and the word dimension at the same time. There could reduce the impact of noise on identifying the severity of bug reports, while reducing reduce bug report dimension and the word dimension.

RQ5. Can the multi-classifier fuzzy-integral with RSMOTE approach outperform state-of-the-art approaches?

As discussed in regard to RQ4, RSMOTE with data reduction (FS→IS) can effectively improve the performance of identifying the severity of bug reports. in order to solve the random sampling uncertainty of RSMOTE, we use FC-FI approach to ensemble multi-RSMOTE. To better demonstrate the superiority of FC-FI approach, in this experiment, we compared the fusion of multi-RSMOTE with fuzzy integral approach with three classic classifier ensemble approaches: voting, bagging, and AdaBoost. In order to compare the performance of integrated methods, we use the balanced dataset obtained from RSMOTE approach. We used the accuracy and *F-measure* evaluation metrics defined above to verify the performance of the multi-classifier fuzzy-integral RSMOTE approach. We bold the best results for each variant.

In terms of the accuracy and *F-measure* values, the average performance of FC-FI approach is better than the performances of the RSMOTE with data reduction, major voting, bagging, and AdaBoost approaches with RSMOTE. In Tables 13- 14, which shows the severity prediction results

TABLE 13. The accuracy of identifying the severity of bug reports.

Dataset	ACC				
	RSMOTE(DR)	Voting	Bagging	Adaboost	FC-FI
DS-E1	0.82	0.78	0.81	0.82	**0.83**
DS-E2	0.78	0.79	0.82	0.81	**0.83**
DS-E3	0.77	0.75	0.77	0.81	**0.81**
DS-E4	0.87	0.88	0.85	0.87	**0.89**
DS-M5	0.88	**0.89**	0.84	0.76	**0.89**
DS-M6	0.86	0.90	0.85	0.90	**0.92**
DS-G7	0.88	0.88	0.78	0.79	**0.92**
DS-G8	0.83	0.79	0.73	0.80	**0.85**
DS-G9	**0.87**	0.85	0.86	0.87	0.85
DS-G10	0.89	0.79	0.85	0.88	**0.90**

TABLE 14. The *F-measure* of identifying the severity of bug reports.

Dataset	F-measure				
	RSMOTE(DR)	Voting	Bagging	Adaboost	FC-FI
DS-E1	0.80	0.79	0.80	0.82	**0.82**
DS-E2	0.78	0.80	**0.81**	0.79	0.80
DS-E3	0.76	0.73	0.78	**0.80**	0.79
DS-E4	0.85	0.82	0.82	0.85	**0.87**
DS-M5	**0.88**	0.86	0.82	0.78	0.88
DS-M6	0.84	0.85	0.81	0.88	**0.91**
DS-G7	0.88	0.86	0.75	0.77	**0.90**
DS-G8	0.80	0.74	0.73	0.80	**0.81**
DS-G9	**0.86**	0.83	0.85	0.85	0.84
DS-G10	0.88	0.80	0.83	0.87	**0.88**

for the *Eclipse* bug reports, the average accuracy of fusion of multi-RSMOTE with fuzzy integral approach is higher than the average accuracies of the RSMOTE with data reduction, major voting, bagging, and AdaBoost approaches by 3.00%, 4.82%, 6.62%, and 4.70% respectively, and the average *F-measure* is similarly higher by 2.00%, 5.24%, 6.25%, and 3.59%, respectively. These experiments show that the performance of the proposed fusion of multi-RSMOTE with fuzzy integral approach achieves better performance than all three classic classifier ensemble approaches (voting, bagging, and AdaBoost).

V. RELATED WORK

Automatic classification technology to support the bug reports could minimize the latent damage of software project in software maintenance activities.

Antoniol *et al.* [22] builds automatic text classification techniques to identify whether a bug report contains real bug or not. They find that alternating decision trees, Naive Bayes classifier, and logistic regression could effective identify real bug from other kinds of bug reports. Menzies and Marcus [23] proposes an automated approach, namely, SEVERity to help the test engineer assign the severity levels to bug reports, which is based on standard machine learning techniques and text mining. Guo *et al.* [38] transfers the knowledge of labeled bug reports from bug repositories (*Eclipse*, *Mozilla*, *GNOME*) to identify the severity of Android bug reports, which compensates for the lack of labeled information for Android bug reports. Xia *et al.* [43] proposes a ELBlocker to classification the blocker bug reports from bug repositories. According to the imbalance degree of original dataset, ELBlocker divides original dataset into multiple disjoint sets. Then ELBlocker combines the results of multiple classifiers, which are trained by multiple disjoint sets.

Xuan *et al.* [35] analyzes the commenters of bug reports, and represents a ranking method to automatic recommend developer to solve the new bug reports in bug repository. Yang *et al.* [19] compares the performance of four imbalance learning strategies and four classification algorithms to identify the high-impact bug reports with imbalanced distribution. In order to reduce the effort of bug report triage, Anvik and Murphy [25] proposes an approach to assist triager to recommend the developers, which is based on machine learning. Tian *et al.* [49] considers multi-information (''temporal,'' ''textual,'' ''author,'' ''related-report,'' ''severity,'' and ''product'') to recommend the priority level of bug reports. Zhang *et al.* [28] proposes an automatic method to identify the severity of bug reports and fixer recommendation. Firstly, for a new coming bug report, they find the top k nearest neighbors' bug reports based on K Nearest Neighbor (KNN) classification alogrithm. Then, they identify the severity of bug reports and fixer recommendation by extracting their features (e.g., assignees and similarity).

Feng *et al.* [30] proposes a test report prioritization method to assist crowdsourced testing, which is based on three strategies. The three strategies could find the riskiest and most diverse bug reports to assist the developer find bugs, respectively. In their subsequent article, Feng *et al.* [31] considers that the bug reports with screenshots and descriptive text. They could assist inspections of crowdsourced bug reports by using a multi- objective optimization-based prioritization technique. Wang *et al.* [32] proposes a cluster-based classification approach to identify ''good'' bug reports from the crowdsoured testing bug reports. The approach could overcome the local bias of crowdsoured testing bug reports. In their subsequent article, Wang *et al.* [33] proposes a method named Local-based Active Classification (LOAF) to address bug reports with the local bias problem and the lack of labeled information in crowdsoured testing.

To improve the data quality, Khoshgoftaar *et al.* [34] and Gao *et al.* [17] use six feature selection approaches to reduce the bug dataset, meanwhile, use three data sampling approaches (RUS, ROS, SMOTE) to handle imbalanced defect data. Shivaji *et al.* [6] proposes a framework to examine multiple feature selection algorithms and remove noise features in classification-based defect prediction. Besides feature selection in defect prediction. Kim *et al.* [36] introduced an approach to measure the noise resistance in defect prediction and how to detect noise data. Xu [27] uses seven approaches to reduce the dimension of dataset, which could improve the performance of classification results. Yang *et al.* [29] uses three commonly feature selection, IG, CHI, Correlation Coefficient to reduce the noise from 4 open-source components from *Eclipse* and *Mozilla*. Zhang [3958] represents a concept profile-based approach to assign the

severity of bug reports. They build the concept profiles based on historical bug reports. Then, the new bug reports come, they assign the severity of new coming bug report by measuring the similarity and severity concepts.

VI. CONCLUSION

In the context of software maintenance, a bug with high severity is typically associated with fatal errors and crashes. Therefore, the automatic prediction of the severity of bug reports could considerably reduce manpower requirements and improve the efficiency of bug resolution. However, two challenges often arise during the automated technique to identify the severity of bug reports: (1) high-dimensionality (both bug reports and words) is found in most bug repositories, and (2) class imbalance occurs for those datasets. In this paper, we address the problem of data reduction and data imbalance distribution for identify the severity of bug reports. We combine FS with IS to simultaneously reduce data scale on the bug report dimension and the word dimension to get small-scale and high-quality set of original dataset. Then, we improved random oversampling technique, named RSMOTE, is presented to weaken the imbalancedness degree of severity bug reports. And further, to avoid the uncertainty of random over-sampling, we develop an ensemble learning algorithm, which is based on Choquet fuzzy integral, to combine multiple RSMOTE. We empirically investigate the performance of data reduction on 10 datasets of three large open source projects, namely *Eclipse*, *Mozilla* and *GNOME*. The results show that our approach can effectively reduce the data scale and improve the performance of identify the severity of bug reports.

REFERENCES

[1] G. Lang, Q. Li, and L. Guo, "Discernibility matrix simplification with new attribute dependency functions for incomplete information systems," *Knowl. Inf. Syst.*, vol. 37, no. 3, pp. 611–638, 2013.

[2] Y. Fu, X. Zhu, and B. Li, "A survey on instance selection for active learning," *Knowl. Inf. Syst.*, vol. 35, no. 2, pp. 249–283, 2013.

[3] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, Mar. 2003.

[4] A. K. Farahat, A. Ghodsi, and M. S. Kamel, "Efficient greedy feature selection for unsupervised learning," *Knowl. Inf. Syst.*, vol. 35, no. 2, pp. 285–310, 2013.

[5] M. Rogati and Y. Yang, "High-performing feature selection for text classification," in *Proc. CIKM*, 2002, pp. 659–661.

[6] S. Shivaji, E. J. Whitehead, R. Akella, and S. Kim, "Reducing features to improve code change-based bug prediction," *IEEE Trans. Softw. Eng.*, vol. 39, no. 4, pp. 552–569, Apr. 2013.

[7] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, "A review of feature selection methods on synthetic data," *Knowl. Inf. Syst.*, vol. 34, no. 3, pp. 483–519, 2013.

[8] M. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of ReliefF and RReliefF," *Mach. Learn.*, vol. 53, nos. 1–2, pp. 23–69, Oct. 2003.

[9] J. Xuan *et al.*, "Towards effective bug triage with software data reduction techniques," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 1, pp. 264–280, Jan. 2015.

[10] J. A. Olvera-López, J. F. Martínez-Trinidad, and J. A. Carrasco-Ochoa, "Restricted sequential floating search applied to object selection," in *Proc. MLDM*, 2007, pp. 694–702.

[11] H. Zhang and G. Sun, "Optimal reference subset selection for nearest neighbor classification by tabu search," *Pattern Recognit.*, vol. 35, no. 7, pp. 1481–1490, 2002.

[12] P. Hart, "The condensed nearest neighbor rule (Corresp.)," *IEEE Trans. Inf. Theory*, vol. IT-14, no. 3, pp. 515–516, May 1968.

[13] B. V. Dasarathy, "Minimal consistent set (MCS) identification for optimal nearest neighbor decision systems design," *IEEE Trans. Syst., Man, Cybern.*, vol. 24, no. 3, pp. 511–517, Mar. 1994.

[14] C. S. Penrod and T. J. Wagner, "Another look at the edited nearest neighbor rule," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-7, no. 2, pp. 92–94, Feb. 1977.

[15] H. Brighton and C. Mellish, "Identifying competence-critical instances for instance-based learners," in *Instance Selection and Construction for Data Mining*, vol. 608. Boston, MA, USA: Springer, 2001, pp. 77–94.

[16] P. Willett, "The Porter stemming algorithm: Then and now," *Program*, vol. 40, no. 3, pp. 219–223, 2006.

[17] K. Gao, T. M. Khoshgoftaar, and A. Napolitano, "Impact of data sampling on stability of feature selection for software measurement data," in *Proc. ICTAI*, 2011, pp. 1004–1011.

[18] A. Lamkanfi, S. Demeyer, E. Giger, and B. Goethals, "Predicting the severity of a reported bug," in *Proc. MSR*, May 2010, pp. 1–10.

[19] 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.

[20] Z. Wang and G. Klir, *Fuzzy Measure Theory*. New York, NY, USA: Plenum, 1992.

[21] R. S. Pressman, *Software Engineering: A Practitioner's Approach*, 7th ed. New York, NY, USA: McGraw-Hill, 2010.

[22] G. Antoniol, K. Ayari, M. Di Penta, F. Khomh, and Y.-G. Guéhéneuc, "Is it a bug or an enhancement: A text-based approach to classify change requests," in *Proc. CASCON*, 2008, p. 23.

[23] T. Menzies and A. Marcus, "Automated severity assessment of software defect reports," in *Proc. ICSM*, Sep. 2008, pp. 346–355.

[24] P. C. Hansen, "Regularization, GSVD and truncated GSVD," *BIT Numer. Math.*, vol. 29, no. 3, pp. 491–504, 1989.

[25] J. Anvik and G. C. Murphy, "Reducing the effort of bug report triage: Recommenders for development-oriented decisions," *ACM Trans. Softw. Eng. Methodol.*, vol. 20, no. 3, 2011, Art. no. 10.

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

[27] Y. Xu, "A comparative study on feature selection in unbalance text classification," in *Proc. ISISE*, Dec. 2012, pp. 44–47.

[28] 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, Jul. 2016.

[29] C.-Z. Yang, C.-C. Hou, W.-C. Kao, and I.-X. Chen, "An empirical study on improving severity prediction of defect reports using feature selection," in *Proc. APSEC*, Dec. 2012, pp. 240–249.

[30] Y. Feng, Z. Chen, J. A. Jones, C. Fang, and B. Xu, "Test report prioritization to assist crowdsourced testing," in *Proc. ESEC/SIGSOFT FSE*, 2015, pp. 225–236.

[31] Y. Feng, J. A. Jones, Z. Chen, and C. Fang, "Multi-objective test report prioritization using image understanding," in *Proc. ASE*, 2016, pp. 202–213.

[32] J. Wang, Q. Cui, Q. Wang, and S. Wang, "Towards effectively test report classification to assist crowdsourced testing," in *Proc. ESEM*, 2016, pp. 6-1–6-10.

[33] J. Wang, S. Wang, Q. Cui, and Q. Wang, "Local-based active classification of test report to assist crowdsourced testing," in *Proc. ASE*, 2016, pp. 190–201.

[34] T. M. Khoshgoftaar, K. Gao, and N. Seliya, "Attribute selection and imbalanced data: Problems in software defect prediction," in *Proc. ICTAI*, 2010, pp. 137–144.

[35] 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.

[36] S. Kim, H. Zhang, R. Wu, and L. Gong, "Dealing with noise in defect prediction," in *Proc. ICSE*, 2011, pp. 481–490.

[37] T. Zhang, G. Yang, B. Lee, and A. T. S. Chan, "Predicting severity of bug report by mining bug repository with concept profile," in *Proc. SAC*, 2015, pp. 1553–1558.

[38] 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.

[39] Eclipse. Accessed: Jun. 2, 2018. [Online]. Available: http://www.bugs.eclipse.org/bugs

[40] *Mozilla*. Accessed: Jun. 2, 2018. [Online]. Available: http://www.bugzilla.mozilla.org
[41] *GNOME*. Accessed: Jun. 2, 2018. [Online]. Available: http://www.bugzilla.gnome.org
[42] *Bugzilla*. Accessed: Jun. 2, 2018. [Online]. Available: https://www.bugzilla.org/
[43] 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, May 2015.
[44] S. B. Kotsiantis and D. Kanellopoulos, "Combining bagging, boosting and dagging for classification problems," in *Proc. KES*, 2007, pp. 493–500.
[45] S. C. Bagui, "Combining pattern classifiers: Methods and algorithms," *Technometrics*, vol. 47, no. 4, pp. 517–518, 2005.
[46] X. Xia, Y. Feng, D. Lo, Z. Chen, and X. Wang, "Towards more accurate multi-label software behavior learning," in *Proc. CSMR-WCRE*, 2014, pp. 134–143.
[47] X. Xia, D. Lo, X. Wang, and B. Zhou, "Tag recommendation in software information sites," in *Proc. MSR*, 2013, pp. 287–296.
[48] *Weka*. Accessed: Jun. 2, 2018. [Online]. Available: http://www.cs.waikato.ac.nz/ml/weka/
[49] 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.

SHIKAI GUO received the B.Sc. degree in computer science from the Information Science and Technology College, Dalian Maritime University, Dalian, China, in 2012, where he is currently pursuing 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.

RONG CHEN (M'10) received the M.S. and Ph.D. degrees in computer software and theory from Jilin University, China, in 1997 and 2000, respectively. He was with Sun Yat-sen University, China. He is currently a Professor with the College of Information Science and Technology, Dalian Maritime University. His research interests are in software diagnosis, collective intelligence, activity recognition, Internet and mobile computing. He is a member of the ACM.

MIAOMIAO WEI received the bachelor's degree in computer science and technology from the Information Science and Technology College, Shandong University of Science and Technology, Qingdao, China, in 2017. She is currently pursuing the master's degree in computer science and technology with the Information Science and Technology College, Dalian Maritime University, Dalian, China. Her current research interests include mining software repositories and imbalance learning from big data.

HUI LI received the bachelor's degree in software engineering, the M.Sc. degree in computer science and technology, and the Ph.D. degree in computer architecture from Northeastern University, China, in 2006, 2008, and 2013, respectively. He is currently an Associate Professor with the School of Information Science and Technology, Dalian Maritime University, China. His current research interests include software engineering and complex networks.

YAQING LIU received the Ph.D. degrees in computer application technology from Dalian Maritime University, Dalian, China, in 2011. He is currently an Associate Professor of software engineering with Dalian Maritime University. His research interests are in software testing and quality assurance, model-driven development, and software maintenance.