An Empirical Study on Application of Word Embedding Techniques for Prediction of Software Defect Severity Level

Lov Kumar[1]
Dept. CSIS
BITS Pilani Hyderabad Campus
lovkumar505@gmail.com

Mukesh Kumar[2]
Dept. CS&E
NIT Patna
mukesh.kumar@nitp.ac.in

Lalita Bhanu Murthy[3]
Dept. CSIS
BITS Pilani Hyderabad Campus
bhanu@hyderabad.bits-pilani.ac.in

Prof. Sanjay Misra[4]
Østfold University College, Halden, Norway
ssopam@gmail.com

Vipul kocher[5]
Testaing.Com
vipulkocher@testAing.com

Srinivas Padmanabhuni[6]
Testaing.Com
srinivas@testaing.com

***Abstract*—Software defect severity level helps to indicate the impact of bugs on the execution of the software and how rapidly these bugs need to be addressed by the team. The working team is regularly analyzing the bugs report and prioritizing the defects. The manual prioritization of these defects based on the experience may be an inaccurate prediction of the severity that will delay in fixing of critical bugs. It is compulsory to automate the process of assigning an appropriate level of severity based on bug report results with an objective to fix critical bugs without any delay. This work aims to develop defect severity level prediction models that have the ability to assign severity level of defects based on bugs report. In this work, seven different word embedding techniques are applied to defect description to represent the word, not just as a number but as a vector in n-dimensional space in order to reduce the number of features. Since the predictive ability of the developed models depends on the vectors extracted from text as they are used as an input to the defect severity level prediction models. Further, three feature selection techniques have been applied to find the right set of relevant vectors. The effectiveness of these word embedding techniques and different sets of vectors are evaluated using eleven different classification techniques with Synthetic Minority Oversampling Technique (SMOTE) to overcome the class imbalance problem. The experimental results show that the word embedding, feature selection techniques and SMOTE have the ability to predict the severity level of the defect in a software.**

Keywords*—*Defect Severity Level Prediction, Data Imbalance Methods, Feature Selection, Classification Techniques, Word Embedding.

I. Introduction

Applying data mining techniques on software repositories such as software fault prediction, maintainability prediction, version control systems, source code analysis, bug archives, etc. is an emerging field that has received significant research interest in recent times. Researchers have proposed many tools and methods using machine learning techniques to assist a practitioner in decision making and automating software engineering tasks [1][2][3][4]. However, Forrest et al. observed that the finding and fixing defects in software is a time-consuming and expensive process. They have found that the median time to repair bugs for ArgoUML software is 190 days, and PostgreSQL is 200 days. They have also observed that more than 50% of all fixed bugs in Mozilla took more than 29 days [5][6]. Therefore, it becomes essential to reduce the time and cost of the bug fixing process and also improve the quality of the software system. Defect severity level prediction has been emerged as a novel research field for the effective allocation of resources and plans to fix the defects based on their severity level [3]. These models help to find the severity level of defects that can be used to find the effect of defects on the software. Defect severity level prediction models are designed based on the features extracted from the defect description. Recent research has used different data mining techniques to extract numerical features from defect descriptions for the severity level of defect prediction using machine learning techniques. However, there are three main technical challenges in building defect severity level prediction models for predicting the proper severity level of the defects using defect description.

- **Word Embedding:** The defect severity level prediction models are often developed based on the unstructured form of the description of defects. The unstructured nature of data poses intrinsic challenges. If some sort of numerical features can be assigned using text mining techniques that can use as an input for model development, then it can be utilized for prediction of future severity level of defects. In this work, seven different word embedding techniques such as Continuous Bag of Words Model (CBOW)[1], Skip-gram (SKG)1, Global Vectors for Word Representation (GLOVE)[2], Google news word to vector(w2v)[3], fasttext (FST)[4], Bidirectional Encoder Representations from Transformers (BERT) [5], and generative pre-training model (GPT) [6] have been applied on bugs

[1] https://towardsdatascience.com/nlp-101-word2vec-skip-gram-and-cbow-93512ee24314

[2] https://nlp.stanford.edu/projects/glove/

[3] https://code.google.com/archive/p/word2vec/

[4] https://fasttext.cc/

[5] https://en.wikipedia.org/wiki/BERT_(language_model)

[6] https://openai.com/blog/language-unsupervised/

978-1-4673-8888-7/16/$31.00 ©2016 IEEE

reports to represent the word not just as a number but as a vector in n-dimensional space. The above techniques provide similar representation for similar words and also provide a small number of features as compared to the size of the vocabulary. We have also removed stop-words, spaces, and bad symbols before applying these techniques. The predictive ability of these techniques is compared with frequently used term frequency, inverse document frequency (TFIDF).

- **High-Dimensional Features Data:** The predictive ability of defect severity level prediction models also depends on the features that are considered as the input of the models. Researchers have concluded that the data having high dimension features consisting of redundant and irrelevant features negatively affect the performance of the defect severity level prediction models [2][3][1]. The presence of a huge number of the feature in the case of text analysis poses intrinsic challenges to develop models for predicting the proper severity level of the defects using defect description. In this study, we have used three different feature selection techniques to remove irrelevant features and select the right sets of the relevant features.
- **Imbalanced Data:** The last challenge in building defect severity level prediction model is that the data used for building the models are imbalanced. A dataset is defined as a balanced dataset when the samples of the dependent variable or output variable are approximately evenly distributed across different values of dependent variable [7][8]. In this study, the considered datasets are observed to be not possessing have an equal number of the severity level of the defects. Hence, it has been proposed to apply Synthetic Minority Oversampling Technique (SMOTE) on each dataset in order to get balance data.

Prioritization of defect based on the severity level computed using bugs reports? is a problem encountered by software practitioners and the study presented in this work is motivated by the need to develop defect severity level prediction models using extracted features with the help of word embedding techniques from bugs reports. This study aims to find the best word embedding technique by comparing the predictive ability of the models developed using seven different types of word embedding techniques. It further investigates the application of feature selection techniques, data sampling techniques, and eleven different classification techniques for prediction of severity level of defects.

II. Related Work

Software researchers have used different methods in the past to extract features from bugs report and used these features as an input for developing models. Menzies and Marcus have used various text mining concepts to extract features from the bugs report [9]. They proposed an automated method called SEVERIS and validated these models using the defects report of NASA's Project and Issue Tracking System (PITS). These proposed models help to predict proper severity level of the defects using defect description. Rajni Jindal et al. also done similar work to extract features from defect descriptions using Term Frequency and Inverse Document Frequency (TFIDF) to extract features [10]. They have used the Radial Basis function network for developing defect severities prediction models. Finally, they found that the proposed methods have a high predictive ability to predict the severity levels of the defects. Sari and Siahaan have also followed a similar method for developing models to predict the severity level of the defects based on defect description [11]. They have applied InfoGain gain on extracted features from text to find relevant features for model development. Finally, they have used a support vector machine with an objective to develop defect severities prediction models.

In 2011, David Lo and the team analyzed the performance of models at three different levels of severity: low, medium, and high. It was found that an artificial neural network (ANN) was among the best methods. However, the predictions were less accurate for high severity faults. In 2012, Sharma et al. [12] proposed a priority prediction method using SVM, Naive Bayes, KNN, and Neural Network. This predicted the priority of the newly arrived bug reports, and the accuracy of almost all techniques (except NB) was less than 70% for Eclipse and Open Office projects. In 2014, Gayathri and Sudha developed an enhanced Multilayer Perceptron Neural Network [13]. Comparative analysis of modeling of defect proneness predictions using a dataset of different metrics from NASA MDP (Metrics Data Program) was performed. In 2017, Gupta and Saxena developed a model for the prediction of the existence of bugs in class [14]. The model developed was the object-oriented Software Bug Prediction System (SBPS), and it was trained using the Promise Software Engineering Repository. The Logistic Regression Classifier provided the best accuracy. The average accuracy of the model was found out to be 76.27%.

In the context of software severity level prediction, most of the researchers have used count vectorization and TFIDF to extract numerical features from bugs report. The concepts of these techniques are based on bag-of-words, therefore it has not capability to capture the position of vocabulary in sentences. These methods do not play well with many machine learning models because of high-dimensional features. While in this work, we are attempting to use seven different word embedding techniques that represent the word not just as a number but as a vector in n-dimensional space. The above techniques provide similar representation for similar words. The effectiveness of these word embedding techniques are evaluated using eleven different classification techniques with Synthetic Minority Oversampling Technique (SMOTE) to overcome the class imbalance problem.

III. STUDY DESIGN

This section presents the details regarding various design setting used for this research.

A. Experimental Dataset

In this study, six different software datasets have been used, which are referred to as CDT, JDT, PDE, Platform, Bugzilla, and Thunderbird to validate our proposed models.

These datasets have been collected from msr2013-bug_dataset-master [7]. Mining Software Repositories (MSR) conducted Challenge every year by providing software-related data and motivate participate to apply data mining techniques for finding important patterns. The datasets are the collection of bugs reports wherein each bugs report contains the defect ID, defect description, and severity level of the defects. Table I shows the details of the dataset used for this study. As shown in Table I, the CDT software bugs report consists of 2220 normal defects, 146 minor defects, 288 major defects, 42 trivial defects, 58 blocker defects, 106 critical defects.

TABLE I: Experimental Data Set Description

	Normal	Minor	Major	Trivial	Blocker	Critical
CDT	2220	146	288	42	58	106
JDT	1906	261	430	104	50	106
PDE	2380	91	295	52	27	70
Platform	1485	215	715	145	77	215
Bugzilla	1342	598	352	302	167	114
Thunderbird	1100	387	655	91	25	658

B. Training of Models from Imbalanced Data Set:

After analyzing experimental data as shown in Table I, it is quite evident that the considered datasets suffering from class imbalance problem, i.e., the number of samples in each class, are not same. Therefore, balancing of data is required before applying any classification techniques [15]. This approach help to improve the predictive ability of the developed software defect severity level prediction models [16][17]. In this study, we have performed Synthetic Minority Oversampling Technique (SMOTE) one each dataset in order to get balance data. SMOTE technique is identified as a very popular technique by different researches that helps to improve the predictive ability of the models.

C. Word Embedding:

The software bugs report consist of the defect ID and their corresponding defect description. In this work, seven different word embedding techniques including Continuous Bag of Words Model (CBOW), Skip-gram(SKG), Global Vectors for Word Representation (GLOVE), Google news word to vector(w2v), fasttext (FST), Bidirectional Encoder Representations from Transformers (BERT), and generative pretraining model (GPT) have been applied on defect description extracted from bugs reports. We have applied these techniques to represent the word not just as a number but as a vector in n-dimensional space. These vectors are used as input to develop models for assigning appropriate severity levels to the defects present in the bugs reports. We have also removed stopwords, bad symbols, and spaces before applying word embedding. We have also compared the predictive ability of these techniques with term frequency, inverse document frequency(TFIDF).

D. Feature Selection Techniques

After successfully finding a vector of defect description, we have used these vectors as an input of the models. Since we are using these vector of n-dimension as an input of the models, so, the performance of the models also depends upon the selection of important features vectors. In this study, we have used three different features selection techniques, i.e., significant sets of features using rank-sum test, uncorrelated sets of features using cross-correlation analysis, and principal component analysis to remove irrelevant features and select right sets of the relevant feature. We have also compared the predictive ability of the models developed using selected sets of features with original features.

E. Classification Technique:

The predictive ability of different word-emending techniques, feature selection techniques and SMOTE are evaluated using eleven most frequently used classifiers such as multinomial naive bayes (MNB), bernoulli naive bayes (BNB),

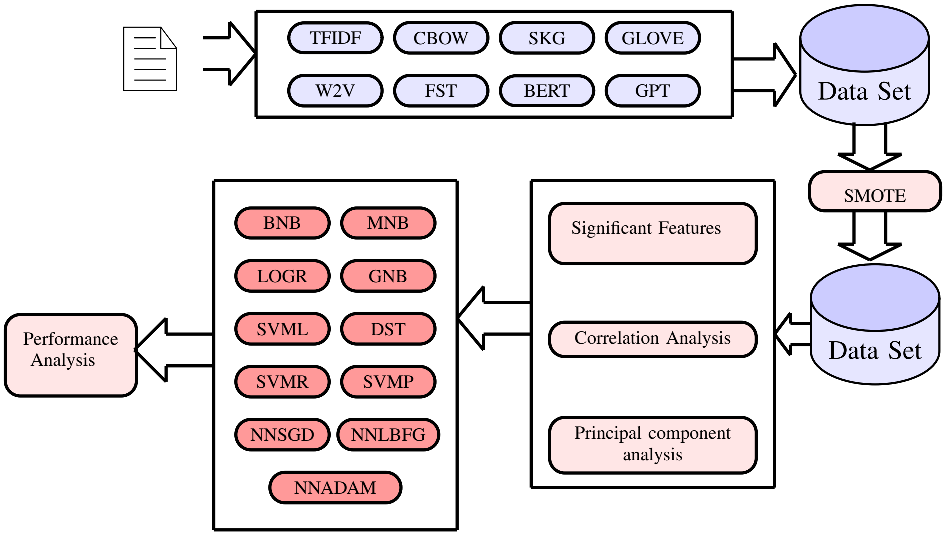

Fig. 1: Framework of proposed work

[7] http://2013.msrconf.org/

gaussian naive bayes (GNB), Logistic Regression (LOGR), decision tree (DST), SVM with linear kernel (SVML), SVM with polynomial kernel (SVMP), SVM with RBF kernel (SVMR), Neural network with LBFG (NNLBFG), Neural network with SGD (NNSGD), and Neural network with ADAM (NNADAM) in software engineering domain [1][18][19].

IV. Research Methodology

In this work, we have applied seven different word embedding methods to extract features from bugs reports and considered these features as an input to develop models for predicting proper severity level of the defects using defect description. These models are trained using eleven different classifiers and validated using 5-fold cross-validation. In this study, we have also considered SMOTE for handling imbalanced data and three feature selection techniques for finding the best combination of relevant features. The detailed overview of our proposed work is giving in Figure 1. The information presented in Figure 1 suggested that the proposed framework is a multi-step process consisting of features extraction from text data using word embedding, handling class imbalance problem using SMOTE, removal of irrelevant features, and finally development of prediction models using eleven different classification techniques.

First, bugs report for a software project is collected from the Bugzilla bug tracking system containing the unique id of defects, description of the defect, and associated severity level of the defects. Next, we have used seven different word embedding to find the numerical representation of defect description. Next, we have used SMOTE techniques to handle the class imbalance problem because the considered dataset is not evenly distributed. The performance of models trained using balanced data is also compared with models developed using original data. After balancing the data, three different features selection techniques such as significant features using rank-sum test, cross-correlation analysis, and principal component analysis are used to remove irrelevant features and select the right sets of reverent features. Finally, eleven different classifiers are used to develop models predicting proper severity level of the defects using defect description. The performance of these developed models is computed and compared using AUC, F-Measure, and accuracy performance values.

V. Empirical Results and Analysis

In this work, we have applied eight different word embedding, one sampling technique, three feature selection techniques, and eleven classification techniques for developing models to predict proper severity level of the defects using defect description. Each word-embedding is applied on the considered datasets as mentioned in Table I. The effectiveness of these word-emending techniques is evaluated using 11 different most frequently used classifiers. Therefore, a total of 4224 (6 datasets * 8 word-embedding*(1 Original Data+ Smote data)*(3 Feature Selection+ 1 All Features)* 11 different classification technique) distinct prediction models are built in the study. The predictive ability of these trained models are evaluated in terms of AUC, F-Measure, and accuracy performance values. These models are validated with the help of 5-fold cross-validation methods. Table II reports the results achieved by different classifiers on original data and sampled data on different sets of features. The results for other cases are of similar type. Looking at information present in Table II, we can be inferred that:

- The high value of AUC confirm that the developed models have the ability to predict proper severity level of the defects using defect description.
- The models developed using a support vector machine with polynomial kernel have better predictive ability as compared to other classifiers.
- The models trained using neural network with ADAM (NNADAM) training algorithm have better predictive ability as compared LBFG, and SGD traing algorithms.
- The models trained by considering balanced data using smote as an input have better predictive ability as compared to original data.

VI. Comparative Analysis

In this section, we analyze and compare the performance of models developed using different word-embedding, classifiers, sampling techniques, and sets of features. In this paper, we have considered Descriptive statistics, box-plot, and Significant tests to compare the developed models for severity level prediction.

A. Word Embedding

The predictive ability of developed defect severity level prediction models using different word embedding are computed with the help of AUC, F-Measure, and accuracy. They are compared using Descriptive statistics, box-plot, and Significant tests. In this study, seven different word embedding techniques such as Continuous Bag of Words Model (CBOW), Skip-gram (SKG), Global Vectors for Word Representation (GLOVE), Google news word to vector(w2v), fasttext (FST), BERT, and generative pre-training model (GPT) have been used to compute the numerical vector of defects reports.
Comparison of Word Embedding: box-plots: Figure 2 provides the performance value, i.e., AUC, F-Measure, and accuracy of different word embedding in terms of Box-Plot diagrams and descriptive statistics. It is clear from Figure 2 that the models developed by considered word vector computed using GLOVE and w2v have better predictive ability to predict the appropriate severity level to the defects present in the bugs reports as compared to other models. The models developed using w2v achieve 0.70 average AUC value, 0.99 max auc, and 0.87 Q3 AUC i.e., 25% models developed using w2v have 0.87 AUC value. However, the models developed using SKG have low predictive ability as compared to other techniques.

Comparison of Word Embedding: Significant Test: In this study, the Wilcoxon signed-rank test is also applied on the AUC, F-Measure, and accuracy for statistically comparing the ability to predict the appropriate severity level of developed models using different word embedding. The objective of this testing is to find whether the models developed using different word embedding have a significant improvement or not. This test uses p-value to accept or reject the considered null hypothesis. The considered null hypothesis for this paper is "the defect severity level prediction models developed by

TABLE II: Performance Value: Classification Techniques

	Accuracy										
	OD										
	MNB	**BNB**	**GNB**	**LOGR**	**DST**	**SVML**	**SVMP**	**SVMR**	**NNLBFG**	**NNSGD**	**NNADAM**
TFIDF	78.11	75.56	22.78	78.22	62.44	78.22	77.78	78.00	72.67	78.11	73.89
CBOW	78.11	78.11	73.44	78.11	60.22	78.11	70.89	78.11	72.78	78.11	78.00
SKG	78.11	78.00	60.11	78.00	60.22	78.11	74.56	78.11	74.56	78.11	77.67
GLOVE	78.11	78.00	51.00	77.67	59.56	78.00	71.11	78.00	70.22	78.11	73.89
W2V	78.11	77.67	55.11	78.11	60.11	77.89	70.44	78.22	68.56	78.11	75.33
FST	78.11	77.56	24.22	78.00	58.56	78.11	75.11	78.11	72.89	78.11	78.11
BERT	74.22	76.89	17.22	78.11	59.67	78.22	72.11	78.22	78.11	78.11	78.11
GPT	54.67	73.89	7.33	78.11	72.33	78.11	78.00	78.11	78.11	78.11	78.11
	SMOTE										
TFIDF	60.18	62.60	59.50	64.35	79.53	67.47	86.95	88.85	85.78	83.32	91.9
CBOW	42.94	16.18	58.21	54.38	76.17	48.53	91.92	95.92	90.46	92.06	95.71
SKG	24.55	16.21	25.92	44.62	74.39	42.32	75.95	68.59	72.14	58.24	74.71
GLOVE	45.89	16.43	56.77	76.52	78.60	78.29	95.79	98.28	92.09	95.60	95.65
W2V	46.52	16.22	59.43	79.95	77.56	80.43	95.07	98.18	91.88	95.93	94.08
FST	27.45	15.97	34.14	37.29	77.35	33.71	86.66	82.33	67.96	84.94	87.25
BERT	24.13	15.97	31.22	77.99	78.90	81.07	94.72	84.63	15.97	15.97	15.97
GPT	22.49	19.05	27.93	44.52	68.06	46.57	61.16	50.58	15.97	15.97	15.97
	AUC										
	OD										
TFIDF	0.51	0.51	0.59	0.51	0.52	0.51	0.51	0.50	0.53	0.50	0.54
CBOW	0.50	0.50	0.53	0.50	0.55	0.50	0.56	0.50	0.52	0.50	0.50
SKG	0.50	0.50	0.58	0.51	0.57	0.50	0.56	0.50	0.54	0.50	0.51
GLOVE	0.50	0.51	0.64	0.51	0.56	0.50	0.54	0.50	0.53	0.50	0.55
W2V	0.50	0.50	0.63	0.53	0.54	0.51	0.57	0.50	0.57	0.50	0.54
FST	0.50	0.51	0.54	0.51	0.55	0.50	0.52	0.50	0.51	0.50	0.50
BERT	0.53	0.52	0.57	0.51	0.54	0.50	0.53	0.50	0.50	0.50	0.50
GPT	0.57	0.52	0.54	0.50	0.52	0.50	0.50	0.50	0.50	0.50	0.50
	SMOTE										
TFIDF	0.80	0.82	0.80	0.80	0.87	0.82	0.90	0.93	0.92	0.91	0.95
CBOW	0.67	0.51	0.76	0.76	0.85	0.72	0.95	0.98	0.94	0.96	0.98
SKG	0.56	0.50	0.55	0.71	0.84	0.70	0.87	0.83	0.87	0.78	0.88
GLOVE	0.72	0.50	0.78	0.88	0.87	0.89	0.97	0.99	0.96	0.97	0.97
W2V	0.73	0.51	0.79	0.90	0.86	0.91	0.97	0.99	0.96	0.97	0.96
FST	0.57	0.50	0.60	0.64	0.87	0.65	0.93	0.90	0.82	0.92	0.93
BERT	0.56	0.50	0.58	0.88	0.87	0.90	0.97	0.91	0.50	0.50	0.50
GPT	0.55	0.54	0.59	0.70	0.81	0.73	0.78	0.72	0.50	0.50	0.50

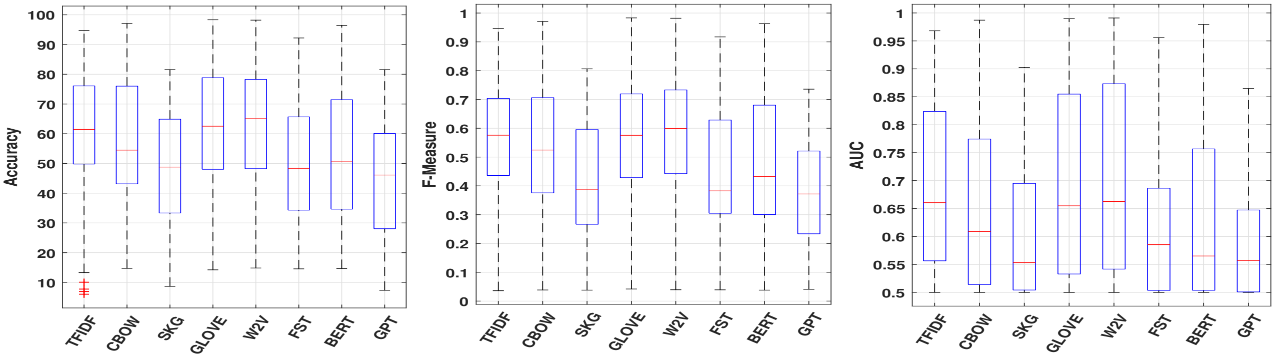

Fig. 2: Performance Box-Plot Diagram: Performance of Different Word Embedding

considering word vector using a different word embedding as an input are significantly same". The considered null hypothesis is only accepted if the obtained p-values using Wilcoxon signed-rank test is greater than 0.05. The results of Wilcoxon signed-rank test on different pairs of word embedding are depicted in Table III. For the purpose of simplicity, we have used only two number for representing results, i.e., 0 means hypothesis accepted (models are significantly same) and 1 means hypothesis rejected (models are significantly different). According to the information present in Table III, the models developed by considering word vector using different word embedding as an input are significantly different for most of the cases.

TABLE III: Significant tests: Different Word Embedding

	TFIDF	CBOW	SKG	GLOVE	W2V	FST	BERT	GPT
TFIDF	0	1	1	0	0	1	1	1
CBOW	1	0	1	1	1	1	1	1
SKG	1	1	0	1	1	0	0	0
GLOVE	0	1	1	0	0	1	1	1
W2V	0	1	1	0	0	1	1	1
FST	1	1	0	1	1	0	0	1
BERT	1	1	0	1	1	0	0	1
GPT	1	1	0	1	1	1	1	0

B. SMOTE

The predictive ability of developed defect severity level prediction models using original data and smote sampled data are computed using AUC, F-Measure, and accuracy performance values and compared using Descriptive statistics, box-plot, and Significant tests.
Comparison of Original Data and SMOTE: box-plots: Figure 4 provides the performance value, i.e., AUC, F-Measure, and accuracy of the models developed using original data and smote sampled data in terms of Box-Plot diagrams and descriptive statistics. The information in Figure 4 demonstrate that the SMOTE data sampling technique plays an important role in improving the predictive ability of the defect severity level prediction models. The models developed using SMOTE sampled data achieve 0.75 average AUC value, 0.99 max auc, and 0.86 Q3 AUC, i.e., 25% models developed using SMOTE sampled data have 0.86 AUC value.

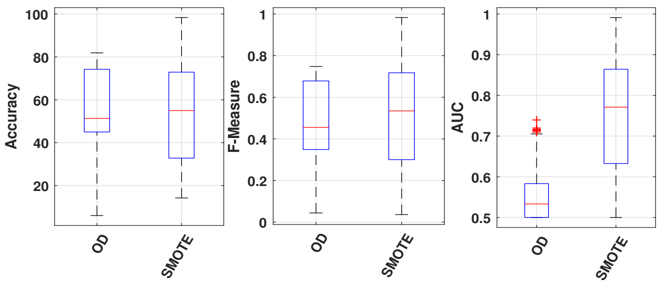

Fig. 4: Performance Box-Plot Diagram: Performance of Original Data and SMOTE

Comparison of Original data SMOTE: Significant Test: In this study, the Wilcoxon signed-rank test is also applied on the AUC, F-Measure, and accuracy for statistically comparing the ability to predict the appropriate severity level of developed models using original data and SMOTE sampled data. The objective of this testing is to find whether the models developed using sampled data have a significant improvement or not. The considered null hypothesis for this paper is "the defect severity level prediction models trained using sampled have not a significant improvement." The considered null hypothesis is only accepted if the obtained p-values using the Wilcoxon signed-rank test is greater than 0.05. In this work, the p-value of the models trained using sampled data and original data is less than 0.05, i.e., our considered hypothesis is rejected. Hence, the models trained using sampled data have significant improvement in predicting defect severity levels.

C. Feature Selection

In this study, we have used three different features selection techniques, i.e., significant sets of features using rank-sum test, uncorrelated sets of features using cross-correlation analysis, and principal component analysis to remove irrelevant features and select right sets of the relevant feature. We have also validated the performance of the models developed using selected sets of features with all features using AUC, F-Measure, and accuracy performance values and compared with the help of Descriptive statistics, boxplot, and Significant tests.
Comparison of Different Sets of Features: box-plots: Figure 3 provides the performance value i.e., AUC, F-Measure, and accuracy of the models trained using selected sets of features and all features. We can see that the models developed using CCRA and AF have slightly better performance as compared to other techniques. The models developed using CCRA achieve 0.65 average AUC value, 0.98 max auc, and 0.78 Q3 AUC i.e., 25% models developed using CCRA have 0.78 AUC value. We can also observed that the models developed using AF have similar performance, but the number of features is more as compared to CCRA features sets.

Comparison of Different Sets of Features: Significant Test: In this study, the Wilcoxon signed-rank test is also applied to the AUC, F-Measure, and accuracy for statistically comparing the ability to predict the appropriate severity level of developed models by considering different sets of features an input. The objective of this testing is to find whether the performance of the models depends on input sets of features or not. The considered null hypothesis for this paper is "the defect severity level prediction models developed by considering different sets of features as an input are significantly same". The considered null hypothesis is only accepted if the obtained p-values using Wilcoxon signed-rank test is greater than 0.05. The results of Wilcoxon signed-rank test are depicted in Table IV. We can see that the models developed using all features, significant sets of features, and uncorrelated sets of features are significantly same.

TABLE IV: Significant tests: Different Sets of Features

	AF	SIGF	CCRA	PCA
AF	0	0	0	1
SIGF	0	0	0	1
CCRA	0	0	0	1
PCA	1	1	1	0

D. Classification Techniques

The predictive ability of developed defect severity level prediction models using different classification techniques are computed using AUC, F-Measure, and accuracy performance values and compared with the help of Descriptive statistics, box-plot, and Significant tests. In this work, we have used eleven different classification techniques such as multinomial naive bayes (MNB), bernoulli naive bayes (BNB), gaussian

TABLE V: Significant tests: Classification Techniques

	MNB	BNB	GNB	LOGR	DST	SVML	SVMP	SVMR	NNLBFG	NNSGD	NNADAM
MNB	0	1	1	1	1	1	1	1	1	0	1
BNB	1	0	1	1	1	1	1	1	1	0	1
GNB	1	1	0	0	1	0	1	1	0	1	1
LOGR	1	1	0	0	1	0	1	1	0	1	0
DST	1	1	1	1	0	1	1	0	1	1	1
SVML	1	1	0	0	1	0	1	1	0	1	0
SVMP	1	1	1	1	1	1	0	1	1	1	1
SVMR	1	1	1	1	0	1	1	0	1	1	1
NNLBFG	1	1	0	0	1	0	1	1	0	1	0
NNSGD	0	0	1	1	1	1	1	1	1	0	1
NNADAM	1	1	1	0	1	0	1	1	0	1	0

naive bayes (GNB), Logistic Regression (LOGR), decision tree (DST), SVM with linear kernel (SVML), SVM with polynomial kernel (SVMP), SVM with RBF kernel (SVMR), Neural network with LBFG (NNLBFG), Neural network with SGD (NNSGD), and Neural network with ADAM (NNADAM) with 5-fold cross-validation to train defect severity level prediction models.

Comparison of Classification Techniques: Descriptive Statistics and box-plots: Figure 5 provides the performance value, i.e., AUC, F-Measure, and accuracy of different classifiers in terms of Box-Plot diagrams and descriptive statistics. It is clear from Figure 2 that the models trained using SVM with polynomial kernel have better predictive ability to predict the appropriate severity level to the defects present in the bugs reports as compared to other models. The models developed using SVM with polynomial kernel achieve 0.73 average AUC value, 0.98 max auc, and 0.89 Q3 AUC i.e., 25% models developed using SVM with polynomial kernel have 0.89 AUC value. However, the models developed using bernoulli naive bayes (BNB) have low predictive ability as compared to other techniques.

Comparison of Classification Techniques: Significant Test: In this study, the Wilcoxon signed-rank test is also applied to the AUC, F-Measure, and accuracy for statistically comparing the ability to predict the appropriate severity level of developed models using different classifiers. The objective of this testing is to find whether the models trained using different classification techniques have a significant improvement or not. The considered null hypothesis for this paper is "the defect severity level prediction models trained using different classifiers are significantly same". The considered null hypothesis is only accepted if the obtained p-values using Wilcoxon signed-rank test is greater than 0.05. The results of Wilcoxon signed-rank test on different pairs of classifiers are depicted in Table V. For the purpose of simplicity, we have used only two number for representing results, i.e., 0 means hypothesis accepted (models are significantly same) and 1 means hypothesis rejected (models are significantly different). While comparing the values present in Table V, we can observed that the models trained using different classifiers are significantly different for most of the cases.

VII. Conclusion

In this paper, we build a model to predict proper severity level of the defects using defect description. Different from existed researches, this work focus on seven different word embedding methods to represent the word not just as a number but as a vector in n-dimensional space. The predictive ability of these methods are evaluated using three sets of features selected using feature selection techniques, and eleven different classifiers with 5-fold cross-validation. We have also used SMOTE techniques in order to handle the class imbalance problem. Finally, the predictive ability of these models are computed and compared using AUC, F-Measure, and accuracy performance values. Our main conclusions are the following:

- The high value of AUC confirms that the developed models using word embedding on balanced data have

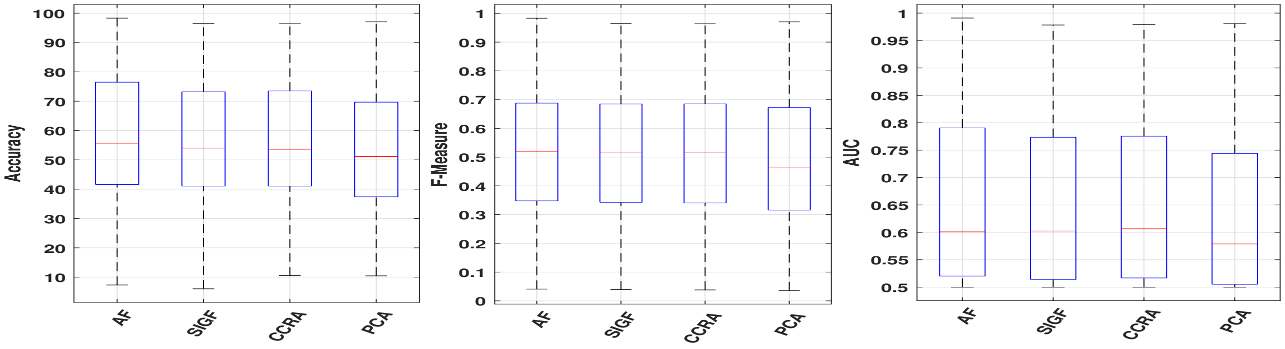

Fig. 3: Performance Box-Plot Diagram: Performance of Different Sets of Features

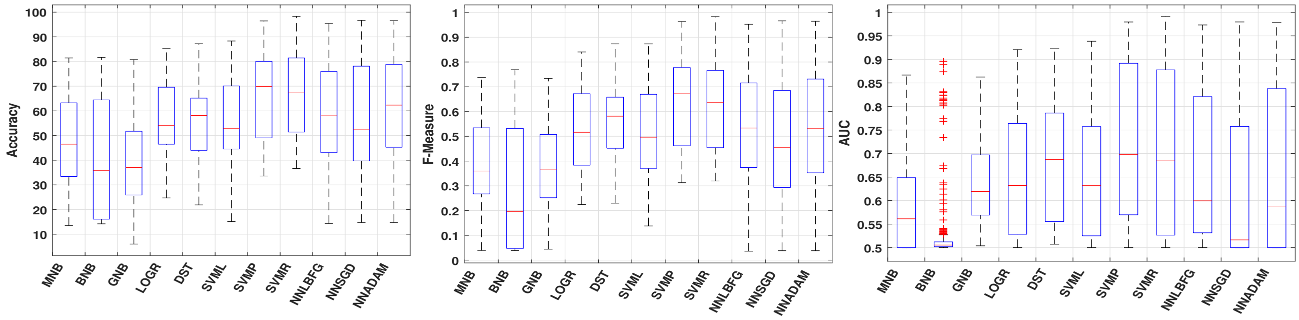

Fig. 5: Performance Box-Plot Diagram: Performance of Different Classification Techniques

the ability to predict severity levels of the defects present based on defect descriptions.

- The models developed by considered word vector computed using GLOVE and w2v have a better predictive ability as compared to other models.
- The defected severity levels prediction models developed using different word embedding methods are significantly different.
- The models trained on sampled data have significant improvement in predicting defect severity levels.
- The predictive ability of the models developed using significant uncorrelated features has a better ability to predict severity level as compared to all features.
- The models developed using SVM with polynomial kernel achieve significantly better performance as compared to other techniques.

In this study, developed are trained using most frequently used classifiers. Future work can be extended to deep-learning approach to achieve higher accuracy of software severity level prediction.

VIII. Acknowledgements

This research is funded by TestAIng Solutions Pvt. Ltd.

References

[1] R. Malhotra and A. Jain, "Fault prediction using statistical and machine learning methods for improving software quality," *Journal of Information Processing Systems*.

[2] L. Kumar, S. Misra, and S. K. Rath, "An empirical analysis of the effectiveness of software metrics and fault prediction model for identifying faulty classes," *Computer Standards & Interfaces*, vol. 53, pp. 1–32, 2017.

[3] R. Malhotra, N. Kapoor, R. Jain, and S. Biyani, "Severity assessment of software defect reports using text classification," *International Journal of Computer Applications*, vol. 83, no. 11, 2013.

[4] G. Abaei, A. Selamat, and H. Fujita, "An empirical study based on semi-supervised hybrid self-organizing map for software fault prediction," *Knowledge-Based Systems*.

[5] S. Kim and E. J. Whitehead Jr, "How long did it take to fix bugs?" in *Proceedings of the 2006 international workshop on Mining software repositories*, 2006, pp. 173–174.

[6] P. Bhattacharya and I. Neamtiu, "Bug-fix time prediction models: can we do better?" in *Proceedings of the 8th Working Conference on Mining Software Repositories*, 2011, pp. 207–210.

[7] A. More and D. P. Rana, "Review of random forest classification techniques to resolve data imbalance," in *2017 1st International Conference on Intelligent Systems and Information Management (ICISIM)*. IEEE, 2017, pp. 72–78.

[8] N. Junsomboon and T. Phienthrakul, "Combining over-sampling and under-sampling techniques for imbalance dataset," in *Proceedings of the 9th International Conference on Machine Learning and Computing*, 2017, pp. 243–247.

[9] T. Menzies and A. Marcus, "Automated severity assessment of software defect reports," in *2008 IEEE International Conference on Software Maintenance*. IEEE, 2008, pp. 346–355.

[10] R. Jindal, R. Malhotra, and A. Jain, "Software defect prediction using neural networks," in *Proceedings of 3rd International Conference on Reliability, Infocom Technologies and Optimization*. IEEE, 2014, pp. 1–6.

[11] S. Ghaluh Indah Permata, "An attribute selection for severity level determination according to the support vector machine classification result," in *proceedings intl conf information system business competitiveness*, 2012.

[12] M. Sharma, P. Bedi, K. Chaturvedi, and V. Singh, "Predicting the priority of a reported bug using machine learning techniques and cross project validation," in *2012 12th International Conference on Intelligent Systems Design and Applications (ISDA)*. IEEE, 2012, pp. 539–545.

[13] M. Gayathri and A. Sudha, "Software defect prediction system using multilayer perceptron neural network with data mining," *International Journal of Recent Technology and Engineering*, vol. 3, no. 2, pp. 54–59, 2014.

[14] D. L. Gupta and K. Saxena, "Software bug prediction using object-oriented metrics," *Sādhanā*, vol. 42, no. 5, pp. 655–669, 2017.

[15] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[16] N. V. Chawla, "Data mining for imbalanced datasets: An overview," in *Data mining and knowledge discovery handbook*. Springer, 2005, pp. 853–867.

[17] T. R. Hoens and N. V. Chawla, "Imbalanced datasets: from sampling to classifiers," *Imbalanced Learning: Foundations, Algorithms, and Applications*, pp. 43–59, 2013.

[18] S. S. Rathore and S. Kumar, "An empirical study of some software fault prediction techniques for the number of faults prediction," *Soft Computing*, vol. 21, no. 24, pp. 7417–7434, 2017.

[19] R. Malhotra, *Empirical research in software engineering: concepts, analysis, and applications*. CRC Press, 2016.