To Minimize Fault Report and Bug Fixing Time using an Efficient Integration of Instance and Aspect Preferment Algorithm

K.S. Maharasan¹* and V. Saravanan²

¹Science and Humanities Faculty, Anna University, Chennai, Tamilnadu, India; ²Department of Computer Applications, Sri Venkateswara College of Computer Applications and Management, Coimbatore, Tamilnadu, India

Abstract: Background: Data mining is an emerged and promising technology, and it utilized in the software engineering development process. It does not only enhance the accurateness, it also improves the reliability of the software. A software development application error is a fault or bug in a computer program. It produces wrong or unexpected outcomes. In traditional software development, faults manually triaged through a specialist developer, i.e., a human being triaged.

Methods: Manual fault triage takes long time and produce low accuracy for the huge amount of faults. To resolve above issues, An Efficient Integration of Instance and Aspect Preferment Algorithm (EIIAPA) is proposed to decrease the scale of fault report data concurrently and to enhance the accurateness of data.

Results & Conclusion: The proposed technique helps to validate & verify software application in effective way. Reduction of data on fault triage aims to construct a high-superiority set of fault data in the small-scale system through eliminating the fault report. To applying an algorithm, fault data set and attributes are extracted from every fault data set and train a predictive model based on the historical dataset. Based on Experimental evaluations, proposed methodology reduces 0.06 ET (Execution Time), and improves 0.5 P (Precision), 0.75 R (Recall), 0.39 F-M (F-Measure) and 5.07% (accuracy) compared than existing methodologies.

Keywords: Software development process, data mining, An Efficient Integration of Instance and Aspect Preferment Algorithm (EIIAPA), compiler, fault triage, execution time, precision, recall, f-measure, accuracy.

1. INTRODUCTION

Data mining is emerged as a promising technology utilized in software engineering improvement procedure, and it did not only enhance the accurateness and completeness of software engineering improvement, it also enhanced the consistency of the software. A software error is a fault or bug in a computer program or system, and it causes to generate wrong or unexpected outcomes. The majority of the bugs occurs from mistakes and bugs made by a person in either program design or source code. The bugs are generated by compilers through creating wrong code. Reports regarding errors in a program or source code commonly are called as fault reports or error reports. The major domain of data mining error repository aims to utilize data mining techniques to deal with software engineering product validation & verification issues. Large-scale database of software repositories is utilized for storing the outcomes of software engineering development. Software repository has large-scale and complex data and it is unable to perform software analysis process. Hence, data mining techniques can determine the interesting data on software depositories and real-world software verification & validation issues will be resolved.

A fault depository is also called a software repository, and it plays a vital role to store error information. The software faults are important steps for fixing errors which are properly allotting a developer to assign new fault. It is an expensive process in software development. The software product errors are mass and it is difficult to generate the triaging process. Most of the software industries pay over 45 percent of the cost for fixing errors. In fault depository, the error is maintained as a fault report and records of repeating the errors along with the status of fault fixing. A fault depository offers a data platform to help numerous kinds of tasks on errors such as error identification, fault localization and reopened fault analysis. Two most important complexities are available in software engineering development integration to fault data, and it may pretend to fault depositories. The systems have the large-scale and low superiority. Since day-by-day recorded errors, large numbers of new faults are stored in fault depositories. Low superiority faults have noisy and redundancy issues. The noisy faults are correlated to developers, and a redundant fault denotes the similar feature in the different database. It avoids time limitation of fault management.

To overcome these issues, an Efficient Integration of Instance and Aspects Preferment Algorithm (EIIAPA) is designed to decrease the scale of fault report data concurrently and to enhance the accurateness of data. The proposed tech-
nique helps to validate & verify software application in an effective way. Reduction of data on fault triage aims to construct a high-superiority set of fault data in the small-scale system by eliminating the fault report. The proposed technique reduces the complexities to understand the logical and conceptual view of the software product. The system design effective process hierarchies to studies from client requirement specification to product validations. The proposed technique reduces the developer work burden and bug fixing time. The eliminated fault data includes less number of fault reports and words as compared with the original fault data. The proposed system offers the similar data over the original fault data and to eliminate the fault dimension and the word dimension. The proposed algorithm calculates fitness value and predicts the efficient retrieval result. The proposed framework provides the accuracy of fault report. The paper contribution is followed as:

- To develop An Efficient Integration of Instance and Aspect Preference Algorithm (EIAPA) to decrease the scale of fault report data concurrently and to enhance the accurateness of data.
- To design an effective framework for validating & verifying software application in an effective way.
- To construct a high-superiority set of fault data in the small-scale system by eliminating the fault report.
- To increase the accurateness of fault triage and reduce the execution time compared to existing approaches.

In the rest of the paper, organization follows; Section 2 expresses the related work which is very closest to the proposed methodology. Section 3 demonstrates the system methodology, methodology implementation details. Section 4 illustrates the experimental setup and discusses implemented result. Section 5 concludes overall work with future outcome.

2. MATERIALS AND METHODS

Lee et al. [1] discussed the intrusion detection system for avoiding un-authorized access in computing. The process of methodology was to audit data from the network securely and perform data mining techniques to separate data according to the domain or similar categories. Tsai et al. [2] focused on Software Oriented Computing (SOC) to the current mainstream Object-Oriented Computing (OOC). It provides data reliability and security. Livshits et al. [3] designed Dyna Minetool to investigate software source code check-ins. It discovered highly interrelated strategy and general fault fixes to determine application-specific coding prototypes. Garcia-Teodoro et al. [4] described anomaly-based intrusion detection methodologies to protect target frameworks and networks against malware behaviors. It calculates the normal activity of the system to ensure the privacy. Aydin et al. [5] introduced an integrating two strategies in one system namely a packet header anomaly detection (PHAD) and network traffic anomaly detection (NETAD), called hybrid Intrusions detection systems (HIDS).

Helmer et al. [6] discussed an artificially intelligent framework for intrusion detection and countermeasures in PC frameworks in a network infrastructure. Panda et al. [7] studied an efficiency of generalization and detection of attacks in supervised learning algorithms for network intrusion identifications. Yildiz et al. [8] explained a security model based on key and quantity of environment features of cloud computing such as Utility, Saas, Platform, web and managed services, internet integration in system development life cycle. Suneeha et al. [9] discussed a preprocessing model of web usage mining and investigated the performance in preprocessed data for website design. It focused on web log file and preprocessed the access log file. Podgurski et al. [10] supported for the classifying reported software errors to assist in prioritizing them and diagnosing their causes.

Zhang et al. [11] developed a predominately bottom-up strategy to find suitable responsibility hierarchies. Xie et al. [12] explained the data placement technique in the Hadoop distributed file system (HDFS). The data reformat and re-deployment methodologies designed in HDFS to resolve the data skew issues. Santos et al. [13] described a usage of an opcode-sequence-frequency illustration of execution. It identified and categorized unknown malware software to achieve high identification rate. Rajasegarar et al. [14] discussed a disseminated anomaly detection methodology based on data grouping to recognize different behavior of data and investigate data measurements. Gong et al. [15] developed a genetic algorithm (GA) for network intrusion detection and the software development. It derived a set of classification rules from network review data, and the support-confidence system to review the quality of each rule.

Yao [16] discussed granular computing in data mining technique and focused on large theoretical level problems by ignoring many inappropriate features. The three basic assumptions are taken like a knowledge granules, structures as fundamental knowledge and mining as search appearance. Sodiyi et al. [17] introduced a threat model for recognizing, enumerating and investigating possible threats to computer systems. Kokolakis et al. [18] discussed business process model (BPM) to perform security analysis and design (IS-SAD). The method investigated the theoretical and practical constraints utilization for a mechanism in the framework. Jadhav et al. [19] described the procedure of estimation and collection of the software packages and reports. The review estimated mechanisms for selecting software packages, software assessment mechanisms, and decision makers for estimating software packages.

3. PROPOSED SYSTEM

The section demonstrates the proposed methodology details, implementation of pre-processing steps, and implemented methodology procedure to decrease the scale of fault report and enhance the accuracy of fault triage. Fig. (1) illustrates the working principle of the proposed algorithm with the implementation processing steps. The pre-processing implementation steps are explained below in details:

3.1. Gathering Dataset

For data consideration, it is very significant for collecting and investigating efficient manners. The contents of the dataset are a separate database table or a solitary statistical data matrix. Where, each column of the table describes a particu-
lar variable. After gathering the information, data is to be stored in the storage server.

3.2. Preprocessing of Data Model

In preprocessing of data or data cleaning, the data processes for an illustration of lost values filling, noisy information smoothing or unreliability of information solution. It additionally utilizes to eliminate the unnecessary information & efficient data transformations into a specific format. The fault data information requires to be preprocessed to decrease the size of the fault data set. An issue of reducing the fault data is to describe the order of relationship of the proposed algorithm. It explained as the prediction of fault data reduction orders; a new and reduced fault data set is acquired from the provided fault data set.

3.3. Feature Selection

In the module utilized to accomplish several goals, such as minimizing the fault data storage, make easy to fault data visualization, reduction of data dimension of the fault dataset for the classification procedure to optimize the time, and enhancing the classifier fault triage accurateness by minimizing the redundant and unrelated information. Data reduction issues are fault dimension and word dimension. The word dimension denotes the quality of data and fault dimension denotes un-useful data. The original dataset replaced with the minimal utilized data set for fault triage. The proposed technique minimizes the number of occurrence through eliminating fault dimension and word dimension.

3.4. Fault Data Reduction

An issue for reduction of fault data is to investigate the order of applying EIIAPA algorithm which is indicated as the computation of reduction orders. Applying an EIIAPA method in fault triage, a fault data set is converted into a text matrix with two dimensions, namely the fault dimension and the word dimension. The data set minimizes the fault reports. It enhances the accuracy of fault triage. It tends to eliminate these words to minimize the computation for fault triage. The fault reduction goal is to enhance the quality of data in fault depositories. The fault reduction goal is to enhance the quality of data in fault depositories and minimizes the scale of fault data. It is minimizing the replication and noisy fault reports with the number of actual faults.

3.5. An Efficient Integration of Instance and Aspect Preference Algorithm (EIIAPA)

An Efficient Integration of Instance and Aspect Preference Algorithm (EIIAPA) is developed to reduce the scale of fault report data concurrently and to improve the accurateness of data. The proposed technique assists to validate & verify software application in effective way. The proposed technique reduces the complexities to understand the logical and conceptual view of software product. The system designs effective process hierarchies to study from client requirement specification to product validations. The proposed technique reduces the developer work burden and bug fixing time for predicting the bugs with various types of sub-classification. In a fault triage, a fault data set is exchanged.
into a text matrix with two dimensions, such as fault dimension and word dimension. The method works to influence the efficient integration of instance and aspect preference to produce a reduction of fault data set. It restored the original fault data set with reduction of fault data set for fault triage. The outcome of reduction of fault data set is new and minimized fault data set. EIIAPA applied consecutively, some of the fault reports may be blank that is all the words in a fault report are eliminated. Such blank fault reports are also eliminated. The method also classifies bug fixing priority level which helps to produce delivery manager to assign the bug fixing to tester with minimal time. Here, the bug report minimizes the time of tester to solve the issues with minimal risk. The eliminated fault data includes less number of fault reports and words as compared with the original fault data. The proposed system offers the similar data over the original fault data and to eliminate the fault dimension and the word dimension.

In fault triage, a fault data set is translated into a text matrix with two dimensions such as fault dimension and a word dimension. An efficient integration of feature and instance selection algorithm main objective to produce a minimal fault data set and restored the original data set with a minimized fault triage. The data set is utilizing an application, to obtain a subset of relevant fault instances \( F_p \) (i.e., fault reports in error data) and a subset of relevant of fault features \( W \) (i.e., words in error data). The data set \( D \) is pruned, and only the fault data with a non-empty projection on \( F_p \) are kept. The outcome of dataset \( DF_p \) includes the data set with at least one word from fault data set. All other data sets are ignored among training and assigned to data set among classification. The data pre-processing mechanisms are tokenization, stop word elimination, stemming procedure and vector space method. The tokenization mechanism utilized to tokenize the summary and explanation of the fault data reports into word vectors. Special characters and non-alphabetical sentences are eliminated to avoid the noisy fault data. The stop word elimination method eliminates the stop sentences in high frequency and does not offer supported data for fault triage. The stemming mechanism utilizes porter stemming method for decreasing inflected sentences their word stem/root form. The vector space method/Term vector mechanism is an algebraic model for describing text file as vector of identifier. The pseudo code of proposed algorithm is given below in detail.

**Input:** Load ARFF Mozilla Fault Dataset

**Output:** Accuracy, Precision, Recall, F-measure and Execution Time

**Procedure**

**Start**

Start the application
Access the developer details;
Assign the Developer authentication details;
Select fault data set;
Preprocess & extract the data set
Assigning new type of faults;

View assigned developer details
Select computational data;
View reduction of computational data;
Get all feature set;

**If** select type of fault

Apply EIIAPA algorithm;
Retrieve fault report;
Update fault rectification status;
View assigned rectification status;
Apply J48 classifier with EIIAPA algorithm for best accuracy;
Predict accuracy, execution time, Precision, Recall and F-Measure;

**Else**

Faults are not predicted;
The new fault is not predicted;

Pseudo Code of Proposed Algorithm

4. RESULT AND DISCUSSION

4.1. Programming Environment

The implementation work is deployed on Intel i\(^{6th}\) processor, 16 GB RAM and 100 GB memory with the windows7 ultimate operating system. The proposed framework is developed in JAVA programming language, JDK 1.8, NETBEANS 8.0.2, with MYSQL database. The proposed technique is used in WEKA library with Mozilla Fault Dataset.

4.2. Performance Metrics

The proposed method explores the performance metrics to improve preprocessing of fault triage data set. It displays the following evaluation parameters separately, such as execution time, accuracy, precision, recall, F-measure as tabulated in Table 1. The metrics are used to evaluate the efficiency and accuracy of proposed methodology.

4.3. Implemented Results

Table 2 explains the Accuracy, Precision, Recall F-measure, and Execution Time for few input aspects with existing methodologies. Table 2 shows the average value of all estimation aspects with input aspects. The proposed system is estimated with following existing classifiers namely: KNN (K-nearest neighbor) [20], NB (Naïve Bayes) [20], and Simple Logistic (SL) [20] classifiers. The proposed EIIAPA is integrated J48 to predict fault report with high accuracy. The classifier also helps to proposed techniques for minimizing the bugs and improving the accuracy of bug report with minimal time. According to Table 2, it noticed that EIIAPA+J48 algorithm has the best score on every specify aspects for classification.
Table 1. Performance metrics descriptions.

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Description</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution Time</td>
<td>Time to execute one program from beginning to ending time.</td>
<td>( ET = \text{Ending time} - \text{Beginning time} )</td>
</tr>
<tr>
<td>Precision</td>
<td>The number of fault reports is properly labeled between reopened labeled and fault data.</td>
<td>( \text{precision} = \frac{C_a \cap C_r}{C_r} )</td>
</tr>
<tr>
<td>Recall</td>
<td>Recall is defined as the amount of reopened fault reports, and it is correctly labeled.</td>
<td>( \text{Recall} = \frac{C_a \cap C_r}{C_a} )</td>
</tr>
<tr>
<td>F-measure</td>
<td>The weighted harmonic mean of precision and recall. The F-measure maintains the balance between the precision and the recall.</td>
<td>( F = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} )</td>
</tr>
</tbody>
</table>

Table 2. Comparison of execution time, accuracy, precision, recall, and f-measure for mozilla fault dataset.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Execution Time (S)</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB</td>
<td>0.06</td>
<td>0.917</td>
<td>0.583</td>
<td>0.713</td>
<td>68.50</td>
</tr>
<tr>
<td>KNN</td>
<td>0.07</td>
<td>0.918</td>
<td>0.912</td>
<td>0.915</td>
<td>88.68</td>
</tr>
<tr>
<td>SL</td>
<td>0.10</td>
<td>0.920</td>
<td>0.921</td>
<td>0.920</td>
<td>89.30</td>
</tr>
<tr>
<td>EIIAPA+J48</td>
<td>0.04</td>
<td>0.925</td>
<td>0.996</td>
<td>0.959</td>
<td>94.37</td>
</tr>
</tbody>
</table>

Fig. (2). Execution Time for Mozilla Fault Dataset.

Fig. (2) demonstrates the Execution Time, and the proposed algorithm EIIAPA+J48 is comparing with the existing algorithms, such as NB, KNN, and SL.

Fig. (3) demonstrates the Precision, and the proposed algorithm EIIAPA+J48A is comparing with the existing algorithms, such as NB, KNN, and SL.

Fig. (4) demonstrates the Recall, and the proposed algorithm EIIAPA+J48 is comparing with the existing algorithms, such as NB, KNN, and SL.

Fig. (5) demonstrates the F-measure, and the proposed algorithm EIIAPA+J48 is comparing with the existing algorithms, such as NB, KNN, and SL.
Fig. (4). Recall for mozilla fault dataset.

Fig. (5). F-Measure for mozilla fault dataset.

Fig. (6). Accuracy for mozilla fault dataset.

According to Figs. (2, 6) observations, the proposed technique is estimated based on execution time, precision, recall, F-measure and accuracy with existing classifier. The proposed EIIAPA+J48 is calculated with Naïve Bayes (NB), K-Nearest Neighbor (KNN) and Simple Logistic (SL) methodologies behalf of on execution time, precision, recall, F-measure and accuracy to evaluate the efficiency and accuracy of proposed technique. KNN is utilized to classify the unknown aspects depending on general classes. However, it gives low accuracy and takes long to classify the fault dataset. The proposed EIIAPA+J48 classified unknown aspects efficiently. Where, it is observed that SL is the closest competitor on behalf of Accuracy, Precision, Recall and F-measure to proposed technique. Where, SL provides the accuracy in the reduction of fault data classification. However, SL takes more time to classify fault dataset, and it does not provide better result, while the data set was condensed. EIIAPA+J48 improved concurrently, minimizing the scale of fault report data. In terms of ET, the proposed technique is the closest competitor is NB Where; Naive Bayes classifier predicts the data and improves the retrieval time of relevance result. However, the method is unable to predict the accuracy of pruned data. It also enhances the accurateness of data, and it takes less time to predict the fault data set. The proposed EIIAPA+J48 reduces 0.06ET (Execution Time) in seconds and improves 0.5P (Precision), 0.75 R (Recall), and 0.39F-M (F-Measure) and 5.07% (Accuracy). Finally, the paper claims the proposed EIIAPA+J48 methodology which performs best on every evaluation matrix & respective input parameters.

CONCLUSION

The paper presents an Efficient Integration of Instance and Aspect Preferment Algorithm (EIIAPA) which is developed to reduce the scale of fault report data concurrently and to improve the accurateness of data. The proposed technique assists to validate & verify software application in an effective way. The proposed technique reduces the complexities to understand the logical and conceptual view of software product. The system designs effective process hierarchies to study from client requirement specification to product validations. The proposed technique reduces the developer work burden and bug fixing time for predicting the bugs with various types of sub-classification. Applying an algorithm for a new fault data set and attributes are extracted from every fault data set and train a predictive model based on the historical dataset. The reduction of data in fault triage is to build a high-superiority set of fault data and eliminating the fault report. It is non-informative or redundant. The proposed EIIAPA+J48 reduces 0.06ET (Execution Time) in seconds and improves 0.5P (Precision), 0.75 R (Recall), and 0.39F-M (F-Measure) and 5.07% (Accuracy). Finally, the paper claims the proposed EIIAPA+J48 methodology which performs best on every evaluation matrix & respective input parameters.

In the future work, the reduction of data outcomes can be improved in fault triage to investigate how to prepare a high quality of fault data set and tackle a task of particular domain software, calculating reduction orders, plan to pay efforts to investigate the potential relationship between the fault data set attributes and reduction orders.

LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BPM</td>
<td>Business Process Model</td>
</tr>
<tr>
<td>EIIAPA</td>
<td>Efficient Integration of Instance and Aspect Preferment Algorithm</td>
</tr>
<tr>
<td>ET</td>
<td>Execution Time</td>
</tr>
</tbody>
</table>
To Minimize Fault Report and Bug Fixing Time

GA = Genetic Algorithm
HDFS = Hadoop Distributed File System
IS-SAD = Information Systems - Security Analysis and Design
JDK = Java Development Kit
KNN = K-nearest neighbour
NB = Naïve Bayes
OOC = Object Oriented Computing
PC = Personal Computer
SaaS = Software as a Service
SL = Simple Logics
SOC = Software Oriented Computing

ETHICS APPROVAL AND CONSENT TO PARTICI-
PATE
Not applicable.

HUMAN AND ANIMAL RIGHTS
No Animals/Humans were used for studies that are the basis of this research.

CONSENT FOR PUBLICATION
Not applicable.

AVAILABILITY OF DATA AND MATERIALS
The data supporting the findings of the article is available in The Mozilla and Eclipse Defect Tracking Dataset at https://github.com/ansymo/msr2013-bug_dataset and mozilla-central at https://hg.mozilla.org/mozilla-central/

FUNDING
None.

CONFLICT OF INTEREST
The authors declare no conflict of interest, financial or otherwise.

ACKNOWLEDGEMENTS
Declared none.

REFERENCES
[7] Panda M, Patra MR. A comparative study of data mining algo-
rithms for network intrusion detection. In Emerging Trends in En-
ergineering and Technology. 2008. ICETET'08. First International Conference on (pp. 504-507). IEEE. [http://dx.doi.org/10.1109/ICETET.2008.80]
posium on (pp. 763-767). IEEE. [http://dx.doi.org/10.1109/IS-
SPAN.2009.157]
[13] Santos I, Brezo F, Ugarte-Pedrozo X, Bringas PG. Opcode se-
quences as a representation of executables for data-mining-based unknown malware detection. Inf Sci 2013, 231: 64-82. [http://dx.doi.org/10.1016/j.ins.2011.08.020]
[15] Gong RH, Zulkernine M, Abolmaesumi P. A software implementa-
tion of a genetic algorithm based approach to network intrusion d-
tection. In Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, 2005 and First ACIS Inter-
national Workshop on Self-Assembling Wireless Networks. SNPD/SAWN 2005. Sixth International Conference on (pp. 246-
253). IEEE.
[18] Kokolakis SA, Demopoulos AJ, Kiountouzis EA. The use of bus-