Handling High-Dimensionality on Software Defect Prediction with FLDA

Conference Paper · October 2021

DOI: 10.1109/ICON-SONICS53103.2021.9616999

5 authors, including:

Rizal Broer Bahaweres
Syarif Hidayatullah State Islamic University Jakarta
62 PUBLICATIONS **159** CITATIONS

SEE PROFILE

Irman Hermadi
Bogor Agricultural University
85 PUBLICATIONS **390** CITATIONS

SEE PROFILE

Arif Imam Suroso
Bogor Agricultural University
72 PUBLICATIONS **139** CITATIONS

SEE PROFILE

Handling High-Dimensionality on Software Defect Prediction with FLDA

Rizal Broer Bahaweres[1] , Ellrica Dewi Herawati Jana[2], Irman Hermadi[3], Arif Imam Suroso[4], Yandra Arkeman[5],
[1,3]Computer Science Dept, IPB University, Bogor, Indonesia
[1,2]Department of Informatics, Syarif Hidayatullah State Islamic University, Jakarta, Indonesia
[4]School of Business, IPB University, Bogor Indonesia
[5]Department of Agro-Industrial Tech., IPB University, Bogor, Indonesia
rizalbroer@computer.org, ellrica.uin17@mhs.uinjkt.ac.id, irmanhermadi@apps.ipb.ac.id,
Arifimamsuroso@apps.ipb.ac.id, yandra.arkeman@gmail.co

***Abstract*—For years, Software Defect Prediction (SDP) has been used as a way to improve software reliability. It is used as a tool to detect the defects on software module before the testing phase. The steps consist of building machine learning model by training dataset with classifier and predict on defective modules. Many datasets suffer from high dimensionality because of the high number of features and those features are mostly irrelevant to predict defect. This results the data to be unnecessarily bulky and the classification process to be time-consuming. We propose a feature extraction technique called FLDA for handling dimensionality problem of dataset and improving the classification performance. We use a total of four dataset from NASA MDP. For classifiers, we use Support Vector Machine (SVM), Random Forest (RF), Naive Bayes (NF), and Multi Layer Perceptron (MLP). Based on the study results, FLDA can significantly reduced the dimension of datasets by creating new feature that contains the most relevant information. FLDA can also shorten the processing time of the classifiers. When compared to another feature extraction technique such as PCA, FLDA can easily outperform it in terms of Recall and AUC.**

***Index Terms*—Feature Extraction, FLDA, PCA, Software Defect Prediction**

I. INTRODUCTION

Nowadays, with the huge expansion and rapid development of technology, ensuring good quality and defect-free becomes a crucial part for developers. However, the relative cost of fixing defects grows significantly through the software development life-cycle. [1] found that resolving the defects in maintenance can cost 100 times more compared to early detection and fix. Software defect prediction (SDP) plays an important role of reducing the cost by recognizing defect-prone modules of a software prior to testing [2]. Application of SDP early in the software life-cycle allows developers to allocate limited resources [3] on software quality assurance and determine the testing priority of different software modules, thus efficiently saving production cost.

SDP is a method of creating machine learning classification models that predicts which modules of the software that are defects prone by using software metrics [4] that were extracted from the software entities. Unfortunately, the growth of software size and complexity is correspondent to the growth of data, that leads to enormous dataset extracted from the software. This is commonly known as dimensionality problem. The dataset often consists of features that are unnecessary, redundant and correlational [5]. It impacts the performance of classification algorithm because of its high computational and memory usage [6]. Data with high-dimensionality would turn the model to be bulky and time-consuming.

In order to handle the dimensionality problem, we use feature extraction. It transforms the original dimension onto a lower dimension while preserving the relevant features and the new constructed features are usually combinations of original features [7]. In the recent years, various methods of feature extraction have been studied and analyzed. The most popular and widely used feature extraction approach is Principal Component Analysis (PCA) introduced by Karl [8]. PCA is an unsupervised learning technique that reduces dimensionality of such dataset. However, since PCA ignores the label information, it has little to do with classification task [9] that is conducted in defect prediction.

This study will focus on the feature extraction of the preprocessing step. We applied Linear Discriminant Analysis by Fisher (FLDA) [10] as the feature extraction method. It is a supervised learning technique feature extraction to handle high-dimensionality that has been widely used in face recognition [11] [12], healthcare [13] [14] and text

classification [15]. Unlike PCA, FLDA selects the most discriminative features (i.e software metric) with respect to the class label. There are two class labels in SDP, defect-prone and non-defect- prone. FLDA selects the features that has the most relevance to the two classes and projecting the selected features to a new lower dimensional space. In order to create prediction model, machine learning classifier is utilized to learn from the training data and make classification to detect defect in software dataset. The classifiers used in this study are Support Vector Machine, Random Forest , Multi-layer Perceptron and Naive Bayes. The classifiers are selected based on the top four classifiers performance for SDP from a research that was conducted by [16]. Then, the performance results will be compared with PCA.

The research questions in this study are as follow:

1) Does FLDA handle dimensionality problem in software defect prediction?
2) Does FLDA improve the time-performance in classification?
3) How is the performance of using FLDA compared to the PCA feature extraction method in terms of recall and AUC?

The scope of the problem in this study are as follow:

1) Naive Bayes, Multi-layer Perceptron, Support Vector Machine and Random Forest are used to build prediction models,
2) FLDA is used as proposed feature extraction method in preprocessing,
3) The performance metrics are recall and AUC values,
4) PCA is used as the comparison of feature extraction method.

II. LITERATURE REVIEW

A. Related Search

Table I describe some research associated with this study.

TABLE I: Related Work

Ref	SDP	FE LDA	FE Comparison
[14]	-	v	-
[17]	-	v	-
[18]	-	v	-
[19]	v	PCA	-
[20]	v	PCA	-
[16]	v	v	-
Author	v	v	v

In Table I, we can see that on the previous studies, Fisher Linear Discriminant Analysis concept or method was commonly used in health-care studies [14] [17] [18]. Meanwhile most studies of reducing dimensionality in software defect prediction use PCA [19] [20], an unsupervised technique which is known for most popular feature extraction technique. Meaning, they are not considering the class label (defect and non defect) in software defect prediction dataset when creating new features. FLDA, a supervised technique for reducing dimensionality problem is more suitable for performing classification task in software defect prediction.

[16] is one of the few studies that use FLDA to handle dimensionality problem in software defect prediction. However, it still can't prove the superior of LDA compared to the famous method, PCA, in handling dimensionality problem, especially in software defect prediction. PCA has become the most widely used feature reduction technique in Software Defect Prediction [19] [20]. Only few studies about Software Defect Prediction that use FLDA [16]. For feature reduction technique in Software Defect Prediction, many use unsupervised technique like Information Gain [21]. There are also few studies in Software Defect Prediction that use FLDA [16] but there is still no studies that measure the performance of FLDA in Software Defect Prediction by comparing FLDA with another supervised feature extraction technique such as PCA.

B. High-dimensionality

High-dimensionality dataset is a dataset that consists lots of features or independent variables. But in reality, not all features are important. Most of the time, dataset has redundant and irrelevant features to the class label. This will cause a problem called the curse of dimensionality, where the higher dimension of a data will lead to a lower result in the prediction algorithm because of the complex computation process. The large number of features will also cause a heavy and time-consuming model as well as large memory, making it more difficult to process.

C. Feature Extraction

Feature Extraction is a technique to lower features by creating new features that contains the most discriminant features. This method uses a dimensionality-reduction approach which in general, the process of feature extraction is to transform an existing feature matrix into a new, lower-dimensional feature matrix by preserving the most relevant information [22]. This results in a significant reduction in the dimension of the dataset while helping the prediction algorithm to make predictions based on the right data or features. There are two main approaches to dimensionality reduction techniques, unsupervised and supervised approaches. In an unsupervised approach, dimensionality reduction techniques do not need to label the class. Whereas in a supervised approach, it considers the class label. That makes supervised

technique more suitable in performing classification tasks in predicting software defects.

In this study, we use Fisher Linear Discriminant Analysis (FLDA), a feature extraction technique with supervised approach.

D. Machine Learning Classifiers

Based on study that was conducted by [16], SVM, RF, NB, and MLP result as the classifiers that has the best performance. As NB, MLP, and RF result a consistently well performance while SVM has suitability for binary classification problems. Hence, we use those four algorithms as the machine learning classifiers on the proposed method.

E. Performance Evaluation

In this study, we use recall or True Positive Rate (TPR) as used by [14] [16] [17] [18] [13] [21].

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (1)$$

Another evaluation matrix that we use evaluate the performance is AUC [16] [17] [18] [13], since AUC is not affected by the skewness of defect data [23]. The AUC model use value from 0 to 1 to determine the classification performance where the poor performance has value near 0 meanwhile the excellence performance has value near 1.

III. Research Methodology

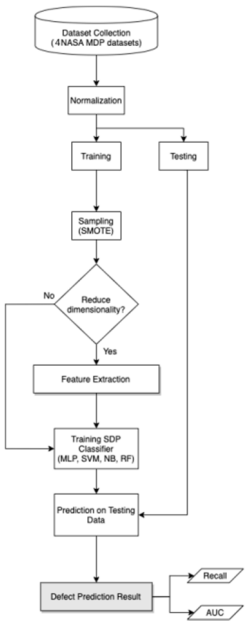

Fig. 1: Methodology

A. Dataset Collection

In this research, we used 4 high-dimensional datasets from NASA Metrics Data Program that we got from http://promise.site.uottawa.ca/SERepository.

NASA MDP datasets contain high dimensional data [24]. All four datasets have various metrics including Line of Code (LOC) metric using different programming languages, Halstead's complexity, McCabe's cyclomatic complexity, etc.

TABLE II: Related Work

Datasets	Number of Attributes	Defects (%)
PC3	37	12,35
PC4	37	13,86
KC1	21	13,9
CM1	21	16,04

B. Normalization

Software metrics in dataset have large variants. The minimum and maximum standard can be very different for each of them. This will affect bad in classification. So, we apply Min-Max Normalization [25] to change the value of each data so that they stay in similar range of 0 to 1.

C. Sampling

One of the problem that occurred in every dataset is imbalanced data. To overcome the issue, we use Synthetic Minority Oversampling (SMOTE) technique. It is one of the most popular and significant oversampling techniques as a standard learning framework for unbalanced data [26]. SMOTE works by increasing the amount of data for the minority class until the two classes (i.e defect-prone and non defect-prone) have a balanced distribution ratio by generating new artificial or synthetic data. The artificial or synthetic data is created by interpolating the nearest minority class instance based on k-nearest neighbor.

D. FLDA Feature Extraction

FLDA is a supervised feature extraction technique with a supervised that chooses the most discriminant features with regards to the class labels. FLDA technique was developed to transform the original dataset's dimension into a lower dimension by maximizing the separation between class and minimizing the variance of each class. Its main purpose is to project the original data matrix into a lower dimensional space. Total between-class and within-class variance, represented by S_B and S_W , can be obtained by the following formula [27]:

$$S_B = \sum_{i=1}^{c} n_i S_{B_i} \quad (2)$$

$$S_W = \sum_{i=1}^{c} n_i S_{W_i} \quad (3)$$

where S_{B_i} represents the between-class variance from the ith class, S_{W_i}represents the between-class variance from the ith class, c represents the number of classes and n represents the sample of the classes.

After calculating the between-class matrix S and within-class matrix SW, construct the lower dimensional matrix space that maximizes eq. (2) and minimizes eq. (3). The transformation matrix W of the FLDA technique can be calculated as follows:

$$\arg\max_x \frac{W^T S_B W}{W^T S_W W} \tag{4}$$

$$S_W W = \lambda S_B W \tag{5}$$

Where λ represents eigenvalues of the transformation matix ($W = S_W^{-1} S_B$).

E. Principal Component Analysis

Principal component analysis is the most popular and widely used feature extraction method for handling high-dimensional data [28]. PCA is an unsupervised learning method that transforms features into new lower features without considering the class label. In software defect prediction, PCA ignores the class of defect and non defect.

F. Classification and Performance Evaluation

We use jupyter notebook for the classification process. We took 4 best classifiers for software defect prediction based on study that was conducted by [16]. The classification algorithms that we use are SVM, RF, MLP and NB. We use the default scikit-learn [29] parameter for each algorithm. After the classification process, we calculate the recall and AUC as the evaluation metrics.

IV. Result and Discussion

We now answer our research question:

A. ***RQ1***. *Does FLDA handle dimensionality problem in software defect prediction?*

To answer this question, we created two scenarios. First, we test the defect prediction without using any feature extraction, only SMOTE to handle the imbalanced classes. Then we compare it to our proposed method which is using FLDA as the feature extraction with the help from SMOTE. For the classifier, we use the four algorithms that has been stated before. The test is done using Python 3.9.0. The results are in the form of numbers of features.

Fig 2 shows the number of features on all datasets that we implement FLDA are significantly reduced. FLDA reduces the original number of features to C-1 number of features where C is the number of label class.

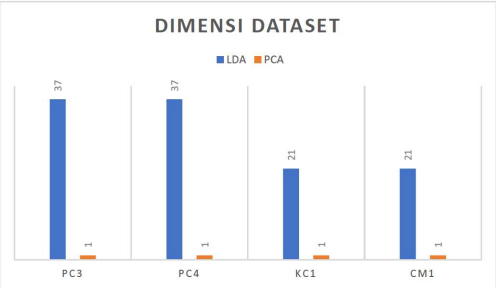

Fig. 2: Number of feature comparison before and after LDA

In our case, we have 2 labels which are Defect and Non Defect. As shown in Fig 2 the features in all datasets are reduced to 1 number of feature. This 1 feature is a new feature that consists all the important and relevant information. Based on the graph, FLDA significantly helps to shape the bulky data to be more lean. It also helps the classifier to train based on the most important and relevant feature.

B. ***RQ2***. *Does FLDA improve the time-performance in classification?*

To answer this question, we test our proposed method, defect prediction using FLDA as feature extraction, with the help from SMOTE to handle class imbalance. Then we compare it to the prediction without feature extraction. To make a fair comparison, we apply SMOTE on both prediction. The test is done using Python 3.9.0. The results are in the form of processing time in classification phase.

TABLE III: Processing time comparison with and without feature extraction

Classifiers	Non FLDA	FLDA
SVM	11,98s	11,32s
NB	11,87s	0,24s
RF	5,11s	2,63s
MLP	30,99s	14,20s

Table III shows that the time performance has improved when FLDA as feature extraction is used with these four classifiers. The time spent to predict defect with all classifier is much faster.

C. ***RQ3***. *How is the classification performance of FLDA compared to the PCA feature extraction method in terms of recall and AUC?*

To answer this question, we create another two scenarios. First, we test defect prediction using a widely popular method for feature extraction called Principal Component Analysis (PCA) as the comparison. To make a fair comparison, we also pair PCA with SMOTE to help the imbalanced class. Then we compare the models of four classifiers that

use PCA as the feature extraction with the models of four classifiers that use FLDA. The test is done using Python 3.9.0 and the results are in the form of Recall and AUC.

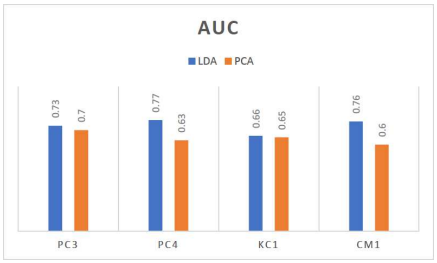

Fig. 3: AUC comparisons with FLDA and PCA

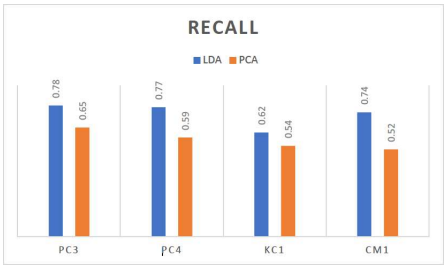

Fig. 4: Recall comparisons with FLDA and PCA

Fig 3 shows the average AUC result of the four classifiers on each dataset. The results of prediction using FLDA varies, with PC4 leads the rank as it has AUC of 77%. Overall, based on fig 3, all datasets that use FLDA as feature extraction have higher AUC result compared to those that use PCA. This shows that FLDA has improved the prediction better than the compared method.

Another metric that we use for performance evaluation is Recall. Fig 4 shows the results of prediction in the form of Recall. Based on the graph, it can be seen that our proposed method could easily outperform PCA on all datasets. PC3 lead the rank as it has Recall of 78%. In the other hand, the highest PCA could get is Recall of 74% on PC3 dataset.

V. Conclusion and Future Work

We proposed a feature extraction technique called FLDA for handling the dimensionality problem and improving the classification performance. We use a total of four dataset from NASA MDP. Based on the study results, FLDA can significantly reduced the dimension of datasets by creating new feature that contains the most relevant information. FLDA can also shorten the processing time of the classifiers. When compared to another feature extraction technique such as PCA, FLDA can easily outperform it in terms of Recall and AUC.

For future study, experiment on other feature extraction techniques beside PCA could be conducted for comparison.

References

[1]M. Dawson, D. Burrell, E. Rahim, and S. Brewster, "Integrating software assurance into the software development life cycle (sdlc)," *Journal of Information Systems Technology and Planning*, vol. 3, pp. 49–53, 01 2010.

[2]Z. Li, "Progress on approaches to software defect prediction," *IET Software*, vol. 12, pp. 161–175(14), June 2018.

[3]M. M. NezhadShokouhi, M. A. Majidi, and A. Rasoolzadegan, "Software defect prediction using over-sampling and feature extraction based on mahalanobis distance," *The Journal of Supercomputing*, vol. 76, no. 1, pp. 602–635, 2020.

[4]D. Radjenovi ć, M. Heričko, R. Torkar, and A. Živkovič, "Software fault prediction metrics: A systematic literature review," *Information and Software Technology*, vol. 55, no. 8, pp. 1397 – 1418, 2013.

[5]H. Tong, B. Liu, and S. Wang, "Software defect prediction using stacked denoising autoencoders and two-stage ensemble learning," *Information and Software Technology*, vol. 96, pp. 94 – 111, 2018.

[6]A. Janecek, W. Gansterer, M. Demel, and G. Ecker, "On the relationship between feature selection and classification accuracy," in *New challenges for feature selection in data mining and knowledge discovery*, pp. 90–105, 2008.

[7]J. Tang, S. Alelyani, and H. Liu, "Feature selection for classification: A review," *Data classification: Algorithms and applications*, p. 37, 2014.

[8]S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," in *2014 Science and Information Conference*, pp. 372–378, IEEE, 2014.

[9]Y. E. Lin and Y. C. Guo, "Optimal uncorrelated unsupervised discriminant projection," in *Applied Mechanics and Materials*, vol. 701, pp. 54–57, Trans Tech Publ, 2015.

[10]R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[11]Q.-x. Gao, L. Zhang, and D. Zhang, "Face recognition using flda with single training image per person," *Applied mathematics and computation*, vol. 205, no. 2, pp. 726–734, 2008.

[12]S. Noushath, G. H. Kumar, and P. Shivakumara, "Diagonal fisher linear discriminant analysis for efficient face recognition," *Neurocomputing*, vol. 69, no. 13-15, pp. 1711–1716, 2006.

[13]I. Beheshti, H. Demirel, F. Farokhian, C. Yang, H. Matsuda, A. D. N. Initiative, *et al.*, "Structural mri-based detection of alzheimer's disease using feature ranking and classification error," *Computer methods and programs in biomedicine*, vol. 137, pp. 177–193, 2016.

[14]E. Dogantekin, A. Dogantekin, D. Avci, and L. Avci, "An intelligent diagnosis system for diabetes on linear discriminant analysis and adaptive network based fuzzy inference system: Lda-anfis," *Digital Signal Processing*, vol. 20, no. 4, pp. 1248–1255, 2010.

[15]Q. Chen, L. Yao, and J. Yang, "Short text classification based on lda topic model," in *2016 International Conference on Audio, Language and Image Processing (ICALIP)*, pp. 749–753, IEEE, 2016.

[16]A. Kalsoom, M. Maqsood, M. A. Ghazanfar, F. Aadil, and S. Rho, "A dimensionality reduction-based efficient software fault prediction using fisher linear discriminant analysis (flda)," *The Journal of Supercomputing*, vol. 74, no. 9, pp. 4568–4602, 2018.

[17]G. R. Banu, "Predicting thyroid disease using linear discriminant analysis (lda) data mining technique," *Communications on Applied Electronic Journal*, 2016.

[18]I. Beheshti, H. Demirel, A. D. N. Initiative, *et al.*, "Feature-ranking-based alzheimer's disease classification from structural mri," *Magnetic resonance imaging*, vol. 34, no. 3, pp. 252–263, 2016.

[19]N. Dhamayanthi and B. Lavanya, “Software defect prediction using principal component analysis and naïve bayes algorithm,” in *Proceedings of International Conference on Computational Intelligence and Data Engineering*, pp. 241–248, Springer, 2019.
[20]Z. Xu, J. Liu, X. Luo, Z. Yang, Y. Zhang, P. Yuan, Y. Tang, and T. Zhang, “Software defect prediction based on kernel pca and weighted extreme learning machine,” *Information and Software Technology*, vol. 106, pp. 182–200, 2019.
[21]A. Hamdy and A. El-Laithy, “Smote and feature selection for more effective bug severity prediction,” *International Journal of Software Engineering and Knowledge Engineering*, vol. 29, no. 06, pp. 897–919, 2019.
[22]M. M. NezhadShokouhi, M. A. Majidi, and A. Rasoolzadegan, “Software defect prediction using over-sampling and feature extraction based on mahalanobis distance,” *The Journal of Supercomputing*, vol. 76, no. 1, pp. 602–635, 2020.
[23]M. Kondo, C.-P. Bezemer, Y. Kamei, A. E. Hassan, and O. Mizuno, “The impact of feature reduction techniques on defect prediction models,” *Empirical Software Engineering*, vol. 24, no. 4, pp. 1925–1963, 2019.
[24]J. Suntoro, F. W. Christanto, and H. Indriyawati, “Software defect prediction using aweig+ adacost bayesian algorithm for handling high dimensional data and class imbalance problem,” *International Journal of Information Technology and Business*, vol. 1, no. 1, pp. 36–41, 2018.
[25]S. G. PATRO and K. K. Sahu, “Normalization: A preprocessing stage,” *IARJSET*, 03 2015.
[26]A. Fern ández, S. Garcia, F. Herrera, and N. V. Chawla, “Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary,” *Journal of artificial intelligence research*, vol. 61, pp. 863–905, 2018.
[27]A. Tharwat, T. Gaber, A. Ibrahim, and A. E. Hassanien, “Linear discriminant analysis: A detailed tutorial,” *AI communications*, vol. 30, no. 2, pp. 169–190, 2017.
[28]I. T. Jolliffe and J. Cadima, “Principal component analysis: a review and recent developments,” *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, p. 20150202, 2016.
[29]F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.