APPLICATION OF DEEP LEARNING AND DIMENSIONALITY REDUCTION IN SOFTWARE DEFECT PREDICTION

Ashi Bhardwaj, Anupreet Kaur and Aman Kumar Jain
Delhi Technological University, Delhi, India
{ashibhardwaj10, apkc98, jainamandelhi}@gmail.com

ABSTRACT **: Software Defect Prediction is an important aspect in order to ensure software quality. Deep Learning techniques can also be used for the same.**

In this paper, we propose to extract a set of expressive features from an initial set of basic change measures using Artificial Neural Network (ANN), and then train a classifier based on the extracted features using Decision tree and compare it to three other methods wherein features are extracted from a set of initial change measures using dimensionality reduction techniques that include Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA) and Kernel PCA.

We use five open source datasets from NASA Promise Data Repository to perform this comparative study.

For evaluation, three widely used metrics: Accuracy, F1 scores and Areas under Receiver Operating Characteristic curve are used. It is found that Artificial Neural Network outperformed all the other dimensionality reduction techniques.

Kernel PCA performed best amongst the dimensionality reduction techniques.

Keywords- Deep Learning, Artificial Neural Network, Principal Component Analysis, Linear Discriminant Analysis, Kernel PCA, Decision Tree, Area under ROC curve

INTRODUCTION

In order to build high quality softwares, defect prediction has become an important aspect as a lot of time and effort is put in software testing and its debugging otherwise. Defect prediction techniques are proposed to help prioritize software testing and debugging; they can recommend software components that are likely to be defective to developers. [1] A lot of parameters are considered while predicting whether a software is buggy or not which include number of lines in the code, its complexity, the number of operators and operands used in the code and other factors. We have considered a set of 22 initial features to predict whether the module is buggy.

Deep learning is a new area and the most promising one in the machine learning literature [9], and has been adopted in a lot of research areas and has proven to be very effective, particularly in image processing [10] and speech recognition [11].

Artificial Neural Network is a deep learning algorithm based on the working of biological neural networks. It has a number of nodes connected via weighted edges.

We propose to extract a set of expressive features from an initial set of basic change measures using Artificial Neural Network (ANN) and then train a classifier based on the extracted features using Decision tree and compare it to three other methods wherein features are extracted from a set of initial change measures using dimensionality reduction techniques that include Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA) and Kernel PCA.

Decision Trees fall under supervised learning method for classification and regression that can easily be visualised. It works by applying classification or regression based on a particular function (here Entropy) on a labeled training set. It splits the population or sample on basis of the most significant splitter by identifying the most significant variable from the dataset. They work on the principle of Greediness.

PCA, LDA and Kernel PCA are ways used for dimensionality reduction. LDA and PCA are linear transformation techniques, the difference being LDA to be supervised and PCA to be unsupervised. PCA is more of a generic dimensionality reduction technique while LDA tends to be more specific. PCA treats dataset as a whole while LDA tries to discriminate between classes within the data. On the other hand, KPCA is non linear form of PCA i.e. an extension to PCA that uses kernel methods.

For evaluation, three widely used metrics: Accuracy, F1 scores and Areas under Receiver Operating Characteristic curve have been used.[12] Classification accuracy alone can at times mislead and hence the other two metrics have also been considered. F1 scores are weighted average of Precision and Recall taking into consideration both False Positives and False Negatives. Basically, it is the harmonic mean of the two. ROC curve analysis can be considered as a complete report of sensitivity and specificity. It is found that Artificial Neural Network outperformed all the other dimensionality reduction techniques. Kernel PCA performed best amongst the other dimensionality reduction techniques.

RQ1: How effective is neural network over other dimensionality reduction techniques?
RQ2: Can accuracy alone be used for the evaluation of a model?

RELATED WORKS

X. Yang, D. Lo, X. Xia, Y. Zhang and J. Sun [1] used learning algorithms to predict defects at change level. They made use of deep learning algorithm to predict the same. They first created a Deep Belief Network to extract a set of expressive features from the initial set of linear features and then used Logistic Regression as a classifier to predict buggy and non-buggy changes.
Using F1-scores and cost effectiveness as performance metrics, they concluded that their model gave better results as compared to a simple Logistic Regression trained model and the approach proposed by Kamei et al [2].
T. Jiang, L. Tan, and S. Kim [3] felt that different developers have different coding styles, commit frequencies and experience levels, causing different defect patterns and hence built a separate prediction model for each developer.
S. Wang, T. Liu, and L. Tan [5] leveraged a Deep Learning algorithm to extract the semantic features of programs from the source code and then applied a Deep Belief network to help the algorithm learn from the extracted features. They used 3 performance metrics - Precision, Recall and F1-scores to evaluate the algorithm. They concluded that the automatically learned semantic features could significantly improve both within-project and cross-project defect prediction compared to traditional features.
Jian Li, Pinjia He, Jieming Zhu, and Michael R. Lyu [7] used Convolutional Neural Network to generate effective features from the initially hand crafted features. They then combined the learned features with the hand crafted features for accurate defect prediction. They evaluated the algorithm based on F-measure and concluded the their approach improved the state-of-the-art method.
P. D. Singh and A. Chugh [8] compared 5 Machine Learning algorithms - Particle Swarm Optimization, Naïve Bayes, Decision Tree, Linear Classifier and Artificial Neural Network - on the data sets for Software Defect Prediction. Linear Classifiers outperformed the other algorithms in terms of defect prediction accuracy. Neural Networks, however, had the lowest error rate.
G. E. Hinton and R. R. Salakhutdinov [9] used Deep Auto-encoder Network to reduce the dimensionality of data. They described a way to effectively initialise the weights for the neurons so that the Deep Auto-encoder works in a better way as compared to Principal Component Analysis to produce a low dimensionality data.
J. Ali, R. Khan, N. Ahmad, I. Maqsood [13] used the Breast Cancer data sets to compare two classification models namely, Random Forest and Decision Trees. Using the performance measures as F-Measure, Precision, Accuracy and Recall, they concluded that having the same number of attributes, Random Forest classifier gave better results with large data sets while Decision Trees outperformed when the number of instances were comparatively lesser.
R. Jindal, R. Malhotra, A. Jain [14] used text mining techniques to develop a model to predict the severity level of each defect report based on classification of existing reports done using the machine learning method namely, Radial Basis Function of neural network. The value of Area Under the Curve (AUC), sensitivity and a suitable threshold criterion known as the cut-off point was used to analyse the Receiver Operating Characteristics (ROC) and predict the results obtained from the model. They concluded that the model gave exceptionally well results in predicting high severity defects than in predicting the defects of the other severity levels.
V. A. Kumar, N. Elavarasan [15] used different feature selection and feature extraction techniques to reduce the dimensionality of high dimensional data. Different statistical measures such as information theory, mutual information, information gain, gain ratio, symmetric uncertainty, correlation and chi squared statistics were

used for feature selection. Techniques such as Principal Component Analysis, Principal Feature Analysis, Fisher Criterion and Linear Discriminant Analysis were studied as a part of feature extraction.

S. Mosci, L. Rosasco, A. Verri [16] use Principal Component Analysis and Kernel Principal Component Analysis to reduce the dimensionality of data and investigate their regularisation properties. They conclude that PCA as a preprocessing step is in itself a regularisation step and does not need any separate regularisation. They also provide a method to choose optimal number of parameters for the reduced dimension data.

N. Varghese, V. Verghese, Gayathri. P and Dr. N. Jaisankar [17] did an elaborate study of various feature selection and feature extraction techniques. These include Singular Value Decomposition, Principal Component Analysis, Independent Component Analysis, Canonical Correlation Analysis, Locally Linear Embedding, Linear Discriminant Analysis and Partial Least Squares Regression.

EXPERIMENTAL DESIGN

Open source datasets are easily available online. The five datasets used for this project were taken from NASA Promise Dataset Repository namely pc1, cm1, jm1, kc1, kc2 each having no missing values and 22 attributes that come from McCabe and Halstead features extractors.

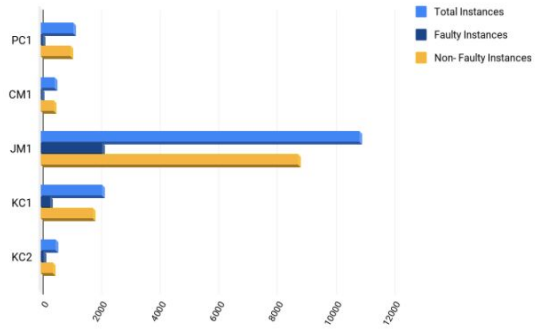


Figure 1: Dataset characteristics

Dataset	Language used	Total Instances	Defective Instances	Non-Defective Instances
PC1	C	1,109	77	1,032
CM1	C	498	49	449
JM1	C	10,885	2,106	8,779
KC1	C++	2,109	326	1,783
KC2	C++	522	105	415

Table 1: Characteristics of Datasets

Feature Name	Description
McCabe's line count of code	It counts the lines of code in module.
McCabe "cyclomatic complexity"	It indicates complexity of the module on basis of number of linearly independent paths.
McCabe "essential complexity"	It indicates the extent to which a flowgraph can be reduced.
McCabe "design complexity"	It indicates cyclomatic complexity of its reduced flowgraph.
Halstead total operators + operands	It gives the count of operators and operands used in the module.
Halstead "volume"	It measures the product of length and log of vocabulary on base 2.
Halstead "program length"	It indicates the length of the program.
Halstead "difficulty"	It is related to the difficulty of the program to write or understand. Also computed as reciprocal of length.
Halstead "intelligence"	It determines amount of intelligence presented in the module. (Length * Volume)
Halstead "effort"	It translates into actual coding time.
Halstead	It is a base Halstead measure.
Halstead's time estimator	It evaluates the testing time of C/C++ codes.
Halstead's line count	It indicates the numbers of lines in the code.
Halstead's count of lines of comments	It indicates the number of lines of comments.
Halstead's count of blank	It gives the count of blank lines present in the module.
IOCodeAndComment	It gives the lines of code and comment in the module.
unique operators	It counts the total number of distinct operators in the module.
unique operands	It counts the total number of distinct operands in the module.
total operators	It counts the total number of operators in the module.
total operands	It counts the total number of operands in the module.
branchCount of the flow graph	It gives the count of branches in the flow graph.
defects	It indicates whether the module has any reported defect or not.

Table 2: Description of the features present in the Dataset

Since the data had more instances of non buggy modules, under sampling was done to prevent the model from being biased towards non buggy instances.

RESEARCH METHODOLOGY

The dataset, used for Software Defect prediction in the project, is taken from NASA Promise Repository. All the 5 data sets have 22 attributes, though each having a different number of instances. Decision Tree classifier is used to make the model learn from the test set and then the model is tested on the training set and

the performance measures are calculated. However, having so many attributes and instances can lead the model to overfit. Hence, we first reduced the dimensionality of the data to a set of 8 cumulated features using 4 different techniques and then trained the model using Decision Tree classifier. A detailed comparison was then made based on the performance metrics that include Accuracy, F1-Scores and Area Under the Receiver Operating Characteristics (ROC).
Following are the algorithms used for Dimensionality Reduction, along with a brief description.

A. Artificial Neural Network (ANN)

This algorithm is somewhat based on the human brain or the human nervous system and uses a set of hidden layers with varied number of nodes called neurons. Each neuron takes inputs from either a few or all of the previous layer neurons and processes the input using initialised weights and an activation function. It then sends the output to many neurons of the next layer. Based on the output and the cost function, the weights are updated over a number of epochs until the parameters best fit the model.
To train our model, we use 2 hidden layers. The first one having 30 neurons and the second consisting of 8 neurons. The cumulated features extracted from the network are then used to train the model using Decision Tree Classifier.

B. Principal Component Analysis (PCA)

The PCA is a statistical data analysis method that transforms the initial set of variables into an assorted set of linear combinations, known as the principal components (PC), with specific properties with respect to variances. This condenses the dimensionality of the system while maintaining information on the variable connections [17].
The PCA algorithm is applied such that it extracts 8 new independent features that explain most the variance of the dataset, regardless of the dependent variable. Since the final class of each instance is not considered while turning the data into a low dimensional one, hence it is an Unsupervised Model.

C. Linear Discriminant Analysis (LDA)

In a high dimensional data, it is difficult to find similarities between different data points and hence the model is difficult to analyse. The LDA algorithm maps down the high dimensional data to a low dimensional space which is then fed to the classifier to train the model. LDA aims to maximize the between-class distance and minimize the within-class distance in the dimensionality reduced space [15].
The LDA algorithm is applied such that it extracts 8 new independent features that separate most the classes of the dataset, that is the buggy and non buggy instances. Since the extracted features are obtained taking into consideration the dependent variable, hence it is a Supervised Model.

D. Kernel Principal Component Analysis (KPCA)

Standard PCA only allows linear dimensionality reduction. However, if the data has more complicated structures which cannot be well represented in a linear subspace, standard PCA will not be very helpful. Kernel PCA thus extends conventional principal component analysis (PCA) to a high dimensional feature space using the kernel algorithm. An intuitive understanding of the Gaussian kernel PCA is that it makes use of the distances between different training data points, which is like k-nearest neighbor or clustering methods [19]. Gaussian kernel PCA reveals more complex hidden structures of the data than standard PCA.
Gaussian RBF, Polynomial, Hyperbolic Tangent are some popular Kernel functions. We leveraged the Gaussian RBF kernel function to reduce the dimensionality of our data set.

Decision Tree Algorithm

The Decision Tree algorithm, that is a supervised learning algorithm and works on principles of entropy and information gain, has been used as a classifier. Entropy of a dataset measures the impurity of the dataset i.e., how disordered the data set is.
The most critical aspect of Decision Tree algorithm is the attribute selection method employed at each node of the tree, since there are some attributes that split the data more purely than other attributes. The algorithm works on principle of greediness i.e., it looks for the solution that appears to be best at the moment without looking at the picture at large.
The Decision Tree algorithm uses the Information Gain, which calculates the reduction in entropy or gain in information, to split the data set using a particular attribute.
The algorithm is advantageous as it requires less data cleaning and is not influenced by outliers and missing values to a fair extent.

PERFORMANCE MEASURES

	Predicted Buggy	Predicted Clean
True Buggy	TP	FN
True Clean	FP	TN

Table 3: Confusion Matrix

A. Accuracy

This refers to the ratio of correctly predicted instances of the test set to the total number of instances of the test set.

Accuracy = (TP + TN) / (TP + FN + FP + TN)

B. F1 scores

At times, accuracy paradox can lead to misinterpretation of the results, hence we take another performance metrics called F1 score into consideration. F1 score is the harmonic mean of Precision and Recall, which are also calculated from the confusion matrix.

Precision is the ratio of actual correctly predicted positive (buggy) instances to the total number of predicted positive instances (Precision = $\frac{TP}{TP+FP}$).

Recall is also known as Sensitivity. Recall is the ratio of actual correctly predicted positive (buggy) instances to the total number of actual positive instances (Recall = $\frac{TP}{TP+FN}$)

Taking the harmonic mean, we get F1 score = $\frac{2*Recall*Precision}{Recall+Precision}$

C. Area Under the Curve (AUC)

The performance of the predicted models was evaluated by plotting the Receiver Operating Characteristics (ROC) curve and evaluating the area under the curve. ROC curve, which is defined as a plot of sensitivity on the y-coordinate versus its 1-specificity (it is defined as the ratio of predicted non faulty classes to the number of classes actually non faulty) on the x coordinate, is an effective method of evaluating the quality or performance of predicted models [19].

VALIDATION METHOD

We have used hold out cross validation method to validate the data set. SInce all the data sets used had quite a large number of instances, the training set and test set were divided in the ratio 3:1.

The training set was used to train the classifier and then the model was validated on the test set.

RESULTS

Confusion matrix was obtained by applying various dimensional reduction techniques and training the data set by Decision Tree classifier. Accuracy and F1 score thus obtained are following:

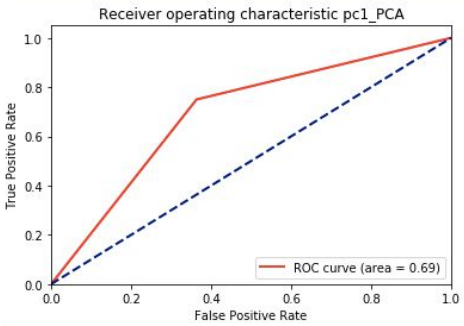
DATASET	ALGORITHM			
	ANN	PCA	LDA	KPCA
PC1	*0.76*	*0.68*	*0.68*	*0.68*
CM1	*0.80*	*0.76*	*0.76*	*0.80*
KC1	*0.74*	*0.72*	*0.72*	*0.71*
KC2	*0.72*	*0.70*	*0.76*	*0.66*
JM1	*0.77*	*0.73*	*0.76*	*0.77*

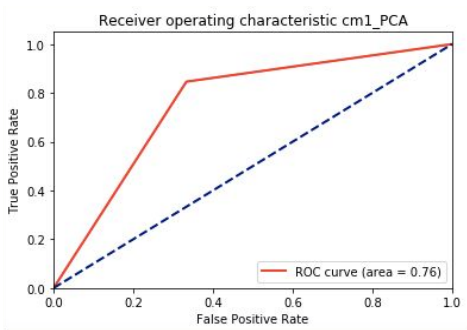
Table 4: Accuracy of each technique

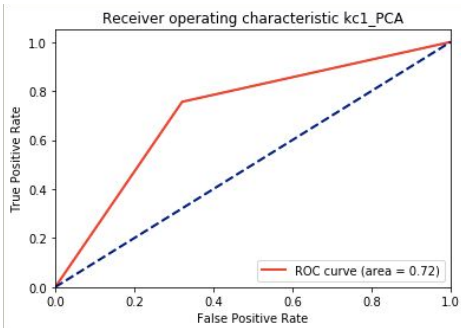
DATASET	ALGORITHM			
	ANN	PCA	LDA	KPCA
PC1	*0.78*	*0.70*	*0.70*	*0.72*
CM1	*0.8*	*0.72*	*0.75*	*0.80*
KC1	*0.73*	*0.71*	*0.69*	*0.70*
KC2	*0.76*	*0.72*	*0.77*	*0.67*
JM1	*0.68*	*0.63*	*0.64*	*0.67*

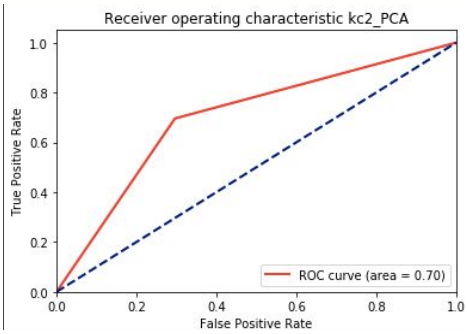
Table 5: F1 Scores of each technique

We plotted ROC curves and found AUC to check the validity of our model. Following are the ROC curves for the respective confusion matrix.









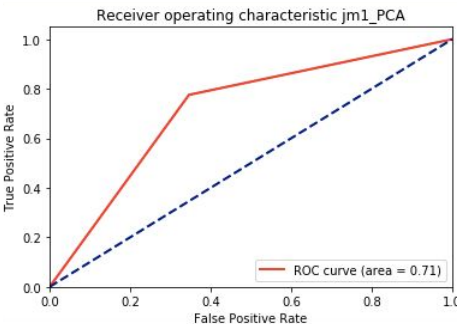
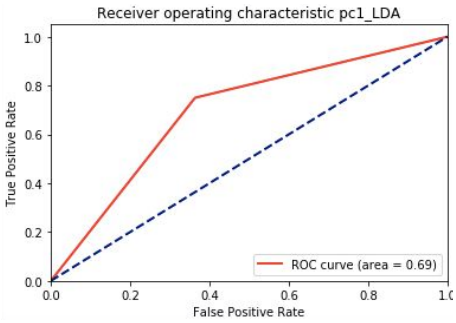
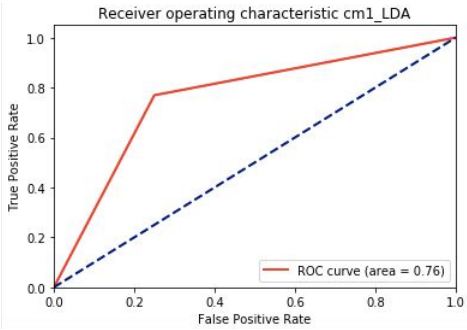
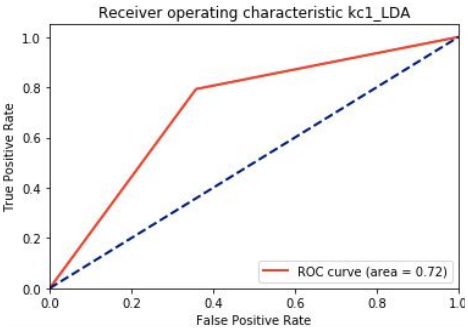
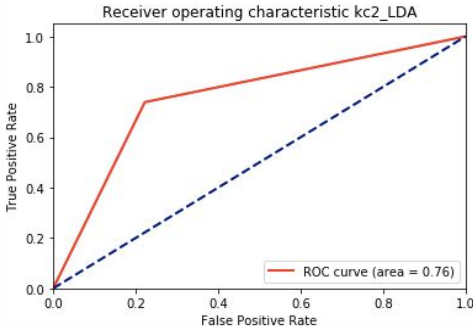


Figure 2: Areas under ROC curves using PCA









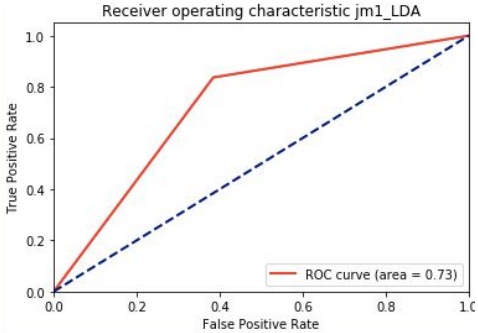
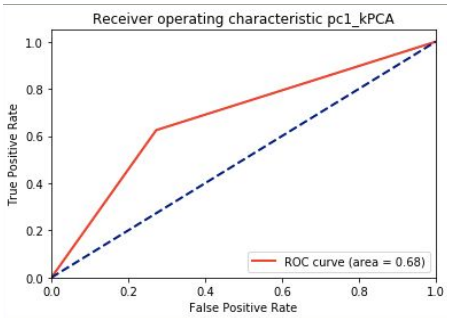
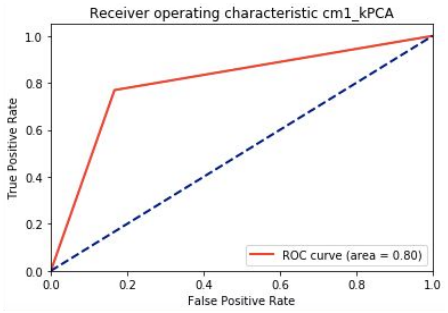
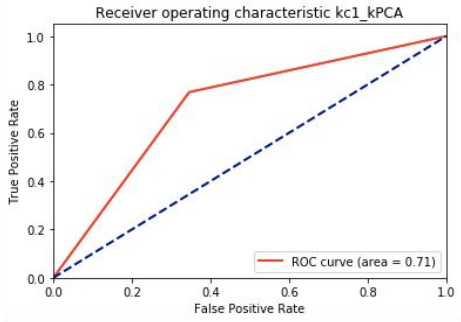
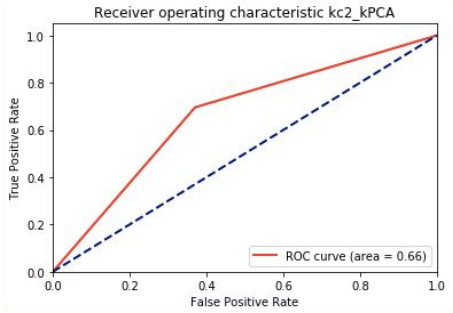


Figure 3: Areas under ROC curves using LDA









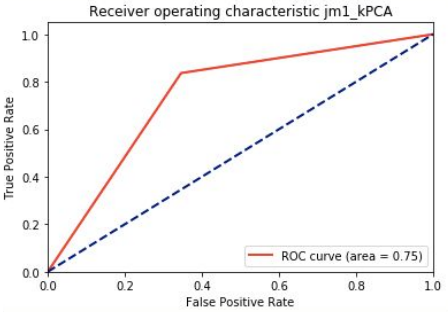
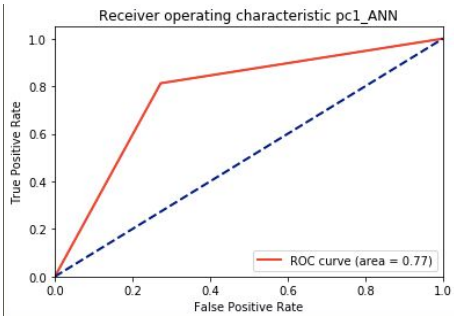
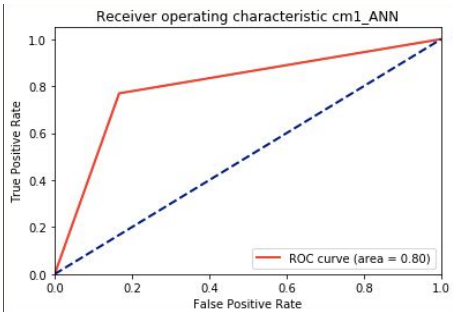
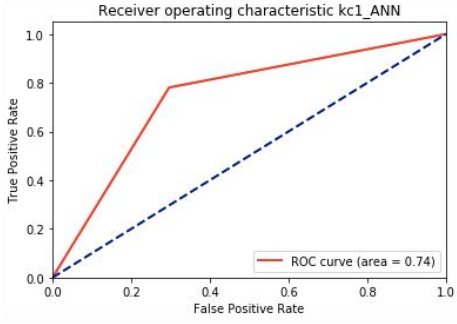
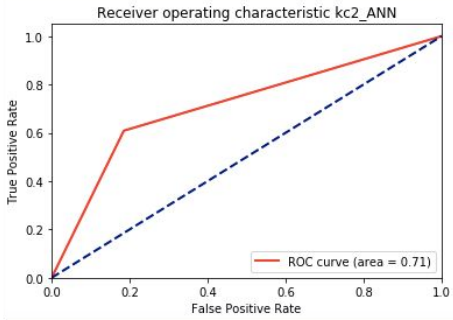


Figure 4: Areas under ROC curves using kernel PCA









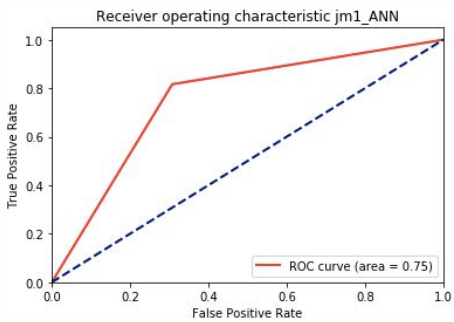


Figure 5: Areas under ROC curves using ANN

DATASET	ALGORITHM			
	ANN	PCA	LDA	KPCA
PC1	*0.77*	*0.69*	*0.69*	*0.68*
CM1	*0.80*	*0.76*	*0.76*	*0.80*
KC1	*0.74*	*0.72*	*0.72*	*0.71*
KC2	*0.71*	*0.70*	*0.76*	*0.66*
JM1	*0.75*	*0.71*	*0.73*	*0.75*

Table 6: AUC values of each technique

DISCUSSION AND INTERPRETATION

For interpretation purposes, we represent our findings graphically, as follows

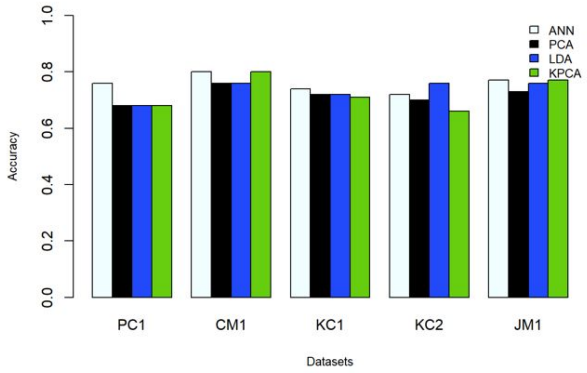


Figure 7: Graphical interpretation considering Accuracy

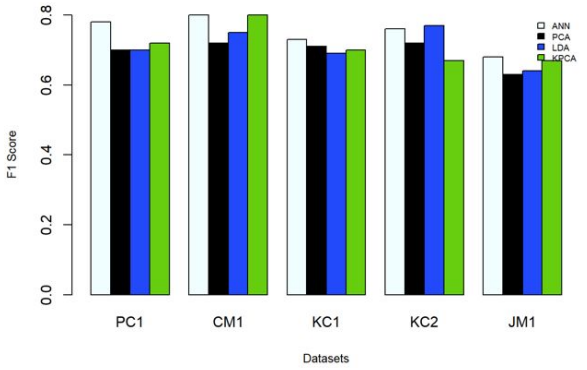


Figure 8: Graphical interpretation considering F1 Scores

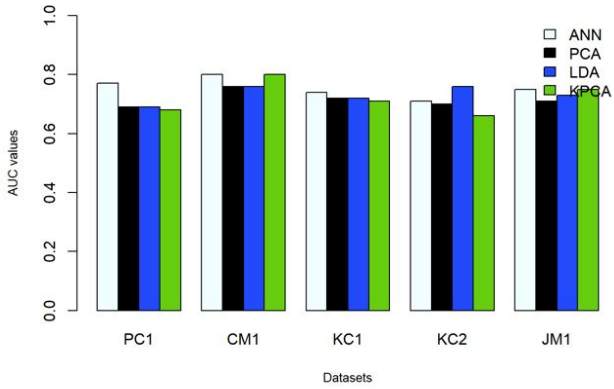


Figure 6: Graphical interpretation considering AUC

We conclude that Artificial Neural Network outperformed the other techniques in 4 out of the 5 datasets considered for defect prediction. This thus shows that Deep Learning can prove to be beneficial when the data under consideration has quite a large number of attributes.
Kernel PCA comes out to be better amongst the other dimensionality reduction techniques.

THREATS TO VALIDITY

In this section, we discuss the various threats to validity of our comparative study.

Construct Validity
We estimate the performance of model using hold out cross validation method. Training and test sets are constructed randomly so they may over fit the data. Using other performance estimation techniques, they might give different results. Apart from the considered attributes, there might be other factors affecting presence of defects.

Internal Validity
The data set used contains information regarding features determined by McCabe and Halstead feature extractors, which are known to have certain limitations.

External Validity
We used only 5 open source data sets taken NASA Promise Repository and so our results may not generalise to all softwares. Replication of this comparative study taking into account other datasets may produce more generalised results.

Conclusion Validity

The datasets being used have class imbalance problem. So, we used AUC to evaluate the performance of our model but it can still be partial for non buggy instances.

CONCLUSION AND FUTURE SCOPE

In this paper, we proposed different dimensionality reduction techniques and compared the results obtained on the basis on prediction accuracy, F1-scores and area under the curve. Artificial Neural Network outperformed all the other techniques in terms of accuracy. The best technique available hence depends on the data set and the performance metrics being considered. The research questions raised in the beginning of the paper have also been answered.

RQ1: How effective is neural network over other dimensionality reduction techniques?

We conclude that Artificial Neural Network outperformed the other techniques in 4 out of the 5 datasets considered for defect prediction. This thus shows that Deep Learning can prove to be beneficial when the data under consideration has quite a large number of attributes.

Kernel PCA comes out to be better amongst the other dimensionality reduction techniques.

RQ2: Can accuracy alone be used for the evaluation of a model?

At times, class imbalance can lead to accuracy paradox, hence we cannot say that accuracy metrics always gives the correct interpretations. Thus, along side accuracy, we have also taken into consideration F1-scores and Area under Receiver Operating Characteristic curve.

In the future, we plan to improve the neural network model by changing various parameters which include the number of hidden layers, neurons in each layer, optimizers and the cost function.

Various other techniques need to be experimented for reducing the dimensionality of the parameters and as per the current techniques, changing the number of components taken can also result in certain improvement.

We also plan to test other classifiers such as Naive Bayes, K-Nearest Neighbors, Kernel Support Vector Machines and ensemble techniques such as Random Forest and compare the results to those found by using Decision Tree classifier.

ACKNOWLEDGEMENTS

This research was partially supported by Delhi Technological University and Dr. Ruchika Malhotra, Associate Professor, Department of Computer Science and Engineering, Delhi Technological University.

REFERENCES

1. X. Yang, D. Lo, X. Xia, Y. Zhang, and J. Sun, “Deep learning for just-in-time defect prediction,” in *QRS’15: Proc. of the International Conference on Software Quality, Reliability and Security*, 2015.
2. Y. Kamei, E. Shihab, B. Adams, A. E. Hassan, A. Mockus, A. Sinha, and N. Ubayashi, “A large-scale empirical study of just-in-time quality assurance,” *TSE*, vol. 39, no. 6, pp. 757–773, 2013.
3. T. Jiang, L. Tan, and S. Kim, “Personalized defect prediction,” in *ASE*, 2013, pp. 279–28.
4. M. Tan, L. Tan, S. Dara, and C. Mayeux, “Online defect prediction for imbalanced data,” in ICSE’15, pages 99–108.
5. S. Wang, T. Liu, and L. Tan, “Automatically learning semantic features for defect prediction,” in *ICSE’16: Proc. of the International Conference on Software Engineering*, 2016.
6. Y. Kamei, S. Matsumoto, A. Monden, K.-i. Matsumoto, B. Adams, and A. E. Hassan, “Revisiting common bug prediction findings using effort-aware models,” in *ICSM*, 2010, pp. 1–10.
7. Jian Li, Pinjia He, Jieming Zhu, and Michael R. Lyu, “Software Defect Prediction via Convolutional Neural Network”.
8. P. D. Singh and A. Chugh, “Software Defect Prediction Analysis Using Machine Learning Algorithms,” in International Conference on Cloud Computing, Data Science & Engineering – Confluence, 2017.
9. G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

10. G. E. Hinton, “Learning multiple layers of representation,” *Trends in cognitive sciences*, vol. 11, no. 10, pp. 428–434, 2007.
11. L. Deng, J. Li, J.-T. Huang, K. Yao, D. Yu, F. Seide, M. Seltzer, G. Zweig, X. He, J. Williams *et al.*, “Recent advances in deep learning for speech research at microsoft,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, 2013, pp. 8604–8608.
12. F. Rahman and P. Devanbu, “How, and why, process metrics are better,” in *ICSE*, 2013, pp. 432–441.
13. J. Ali, R. Khan, N. Ahmad, I. Maqsood, “Random Forests and Decision Trees,” in International Journal of Computer Science Issues, Vol. 9, Issue 5, No 3, September 2012
14. R. Jindal, R. Malhotra, A. Jain, “Software Defect Prediction Using Neural Networks,” IEEE Conference 2014.
15. V. A. Kumar, N. Elavarasan, “ A Survey on Dimensionality Reduction Technique,” in International Journal of Emerging Trends & Technology in Computer Science, Volume 3, Issue 6, November-December 2014.
16. S. Mosci, L. Rosasco, A. Verri, “Dimensionality Reduction and Generalization,” in the 24th International Conference on Machine Learning, Corvallis, OR, 2007.
17. N. Varghese, V. Verghese, Gayathri. P and Dr. N. Jaisankar, “A Survey Of Dimensionality Reduction And Classification Methods,” in International Journal of Computer Science & Engineering Survey, Vol.3, No.3, June 2012.
18. Q. Wang, “Kernel Principal Component Analysis and its Applications in Face Recognition and Active Shape Models,” Rpi, Troy, Ny, Usa, 2011. Copyright 2011
19. J. Hanley, BJ. McNeil, “The meaning and use of the area under a Receiver Operating Characteristic ROC curve”, Radiology, 143, 1982, pp. 29-36.