DEEP NEURAL NETWORK ON SOFTWARE DEFECT PREDICTION

Arie Sapta Nugraha[1], Mohammad Reza Faisal[2], Friska Abadi[3], Radityo Adi Nugroho[4], Rudy Herteno[5]
[12345]Ilmu Komputer FMIPA ULM
A. Yani St. KM 36 Banjarbaru, South Kalimantan
Email: 1611016310007@mhs.ulm.ac.id

Abstract

Software defect prediction is often performed in research to determine the performance, accuracy, precision, and performance of the prediction model or method used in research, using various software metric datasets such as NASA MDP. In this research, we used Deep Neural Network to classify the software metrics dataset modules into Defective and Non-Defective. The data validation technique used to validate the model is Stratified 10-Fold Cross Validation. Performance of the Deep Neural Network model is reported using Area Under the Curve (AUC) for evaluation measurement. AUC of Deep Neural Network is obtained as 0.815 on MC1 dataset and 0.889 on PC1 dataset. Both AUC values obtained in the MC1 and PC1 datasets are included in Good Classification category.

Keywords: *Software Defect Prediction, Deep Neural Network, Area Under Curve, NASA MDP, Cross Validation*

1. INTRODUCTION

Software defect prediction is one of the software engineering studies that are of great concern among researchers. Predicting software defects is important for identifying defects in the early stages of software development. This early identification is critical to producing a cost-effective and quality software product [8]. The accurate prediction of defect-prone software modules can certainly assist testing effort, reduce costs, and improve software quality [3].

Popular public data sets for Software Defect Prediction are software metric datasets such as NASA MDP and PROMISE. Predicting software defects using a dataset that uses software metrics may be useful for improving software development quality. Almost all datasets for predicting software defects have software metrics in them [4].

Classification algorithms is a popular machine learning approach for software defect prediction. It categorizes the software code attributes into defective or not defective, which is collected from previous development projects. Classification algorithms can also predict which components are more likely to be defect-prone, supports better-targeted testing resources, and, therefore, improved efficiency. For prediction modeling, software metrics are used as independent variables and fault data is used as the dependent variable [1]. A wide range of classification techniques have already been proposed in the predicting software defect.

In the field of Software Defect Prediction, various techniques has been proposed. Manjula & Florence [9] proposed a hybrid approach by combining genetic

algorithm and with Deep Neural Network on PROMISE datasets. They did feature optimization with genetic algorithm and Deep Neural Network for its classification. Jayanthi & Florence [6] used Artificial Neural Network (ANN) and performed Principle Component Analysis for feature reduction on NASA MDP datasets. Wahono et al. [11] used various classification algorithm, i.e. Logistic Regression, Linear Discriminant Analysis, Naïve Bayes, k-NN, K*, Neural Network, Support Vector Machine and Decision Tree.

In this research, we propose Deep Neural Network to classify the software metrics dataset. The dataset to be used is D'' NASA MDP Datasets [10]. Area Under Curve (AUC) is used to evaluate the performance of classifier.

2. RESEARCH METHODOLOGY

The research method that will be used in this research are comprised of 1) problem understanding 2) data collection 3) data understanding 4) data preparation 5) modeling and 6) evaluation.

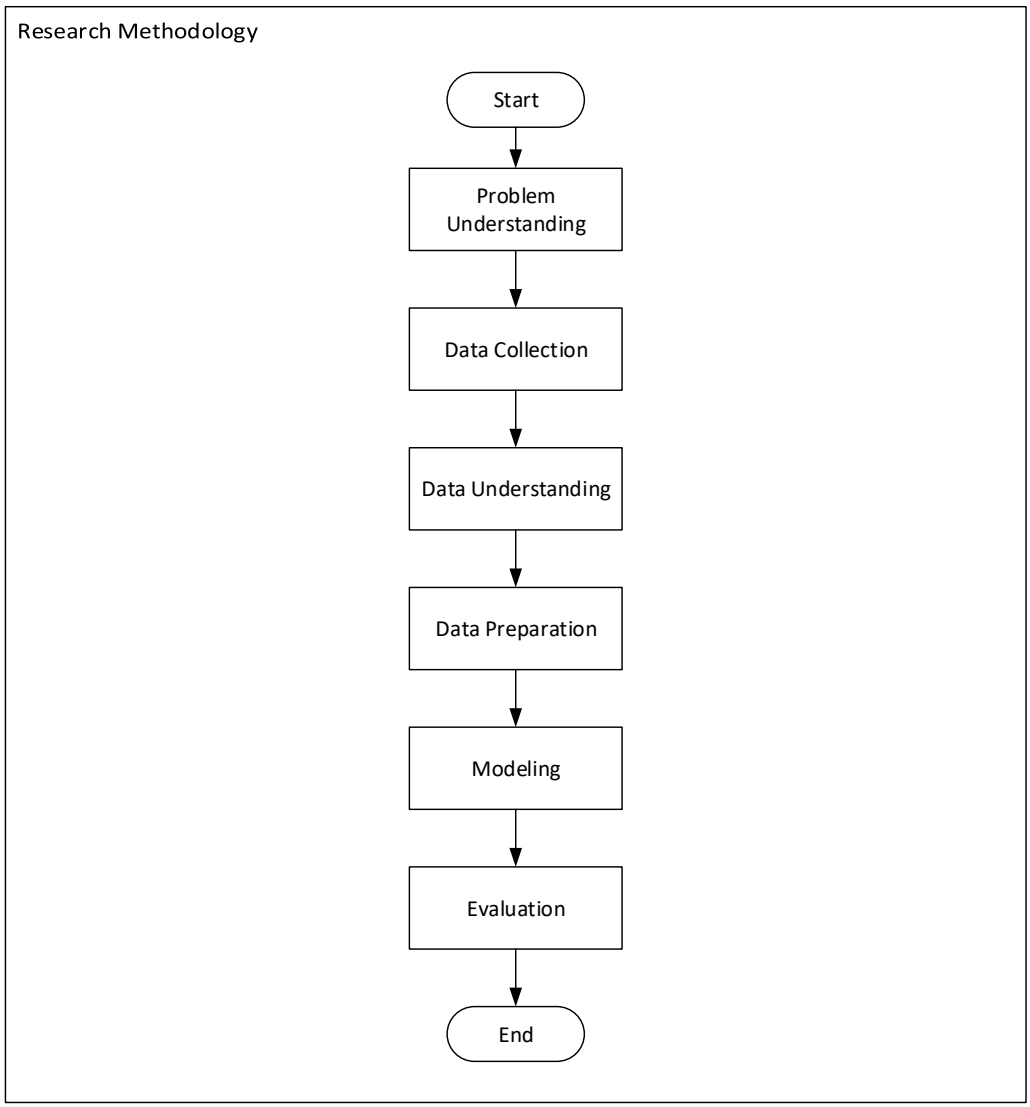

Figure 1 Research Methodology

2.1 Problem Understanding

Before did some data mining on datasets, it is necessary to understand the problems, needs, and goals. In this research, it is quite clear that it has been stated earlier that the goal to be achieved is to create a deep neural network model that can classify software metrics dataset into Defective and Non-Defective.

2.2 Data Collection

The data used in this research are collected from the D'' NASA MDP dataset, which is a dataset of preprocessing results conducted in the research of Shepperd et al. [10].

2.3 Data Understanding

Next is the Data Understanding phase. Data Understanding is needed to understand how the dataset used in this study can achieve the Problem Understanding phase's goals. At this phase, the focus is on understanding the data by exploring the data, understanding the data's features and formats, and verifying the quality of the data. D'' NASA MDP dataset has gone through various data cleaning processes [10], so it only needs a little preparation for the modeling phase.

2.4 Data Preparation

At the data preparation phase, we prepared data for the modeling phase. Some necessary preparations, such as selecting data and changing the data format to fit, are required by the machine learning model.

2.5 Modeling

Machine learning models are made at this modeling phase, including determining the machine learning model to be built, determining the parameters used and determining how the model will be trained using existing data. Simultaneously, parameter and data splitting techniques adjustments will also be used to achieve the best results.

2.6 Evaluation

After the modeling phase, we need to evaluate the results obtained from the modeling stage. This evaluation phase includes evaluating the overall results to determine whether the results obtained reach the goals or not, then reviewing the process that has been executed, whether the process was executed properly, or if something needs to be fixed from the existing process.

3. RESULTS AND DISCUSSION

3.1 Results

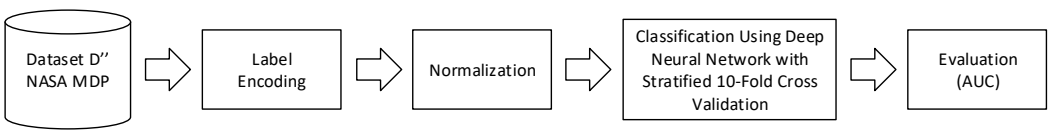

Figure 2 Research Procedure

3.1.1 Collecting Data

The data used in this research are collected from the D'' NASA MDP dataset, which is a dataset of preprocessing results conducted in the research of Shepperd et al. [10]. In this research, MC1 and PC1 datasets are used for applying software defect prediction. These datasets contain various feature sets and a various number of modules. The detail of the datasets is given in Table 1.

Table 1 D'' NASA MDP dataset details

Datasets	Number of Metrics	Number of Modules	Number of Defective	Number of Non-Defective
MC1	38	1988	46	1942
PC1	37	705	61	644

The next step is preprocessing the data to ensure that the data is ready for the modeling process.

3.1.2 Label Encoding

Label Encoding is the process of converting labels into numeric form. The deep neural network model used later uses the sigmoid activation function on the output layer. The sigmoid activation function can only process class labels in the form of numeric form, so it is necessary to change the labels on the dataset D '' NASA MDP to numeric form.

Table 2 Label Encoding results

Initial Label	Label Encoding Result	Meaning of The Label
Y	1	Defective
N	0	Non-Defective

3.1.3 Normalization

Normalization is a scaling technique in which the values are scaled so that the end usually ranges between 0 and 1. In each D'' NASA MDP dataset has a different range of each feature, so it is necessary to carry out the normalization process. The scaling technique used in normalization step is the Min-Max scaling. The formula for the Min-Max is as follows:

$$X' = \frac{X - Xmin}{Xmax - Xmin} \quad ...(1)$$

Example of data that has gone through the label encoding and normalization process are given in Table 3.

Table 3 D'' NASA MDP dataset details

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	Defective
0.069602	0.104110	0.086957	0.033333	1
0.021307	0.010959	0.028986	0.011111	0
0.012784	0.065753	0.086957	0.050000	0
0.001420	0.000000	0.000000	0.000000	0

3.1.4 Classication Algorithm

D'' NASA MDP dataset that has gone through the preprocessing is classified using a Deep Neural Network and validated using the Stratified K-Fold Cross Validation technique with a value of K = 10. Deep Neural Network architecture details are given in Figure 2.

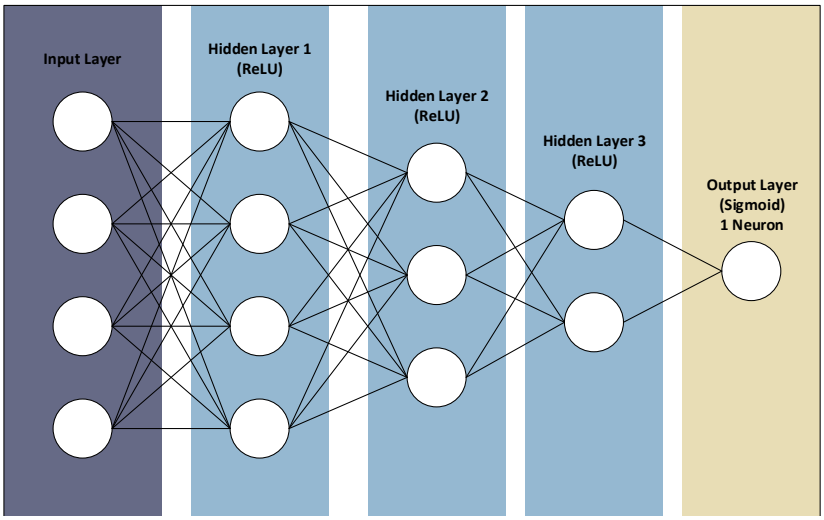

Figure 2 Deep Neural Network Architecture

The data validation technique used in this work is Stratified 10-Fold Cross Validation. Stratified 10-Fold Cross Validation is a variation of Cross Validation. The difference between Stratified 10-Fold Cross Validation and 10-Fold Cross Validation in general is that Stratified 10-Fold can divide ten folds and ensure that the folds have the same number of classes.

3.1.5 Evaluation

Evaluate each method's classification results on the dataset using AUC to get the performance value of each classification algorithm. AUC is an excellent method to obtain performance results from a classification algorithm in general and AUC is also usually used to compare one classification algorithm with another [5]. The AUC is a popular performance measure in class imbalance, and a high AUC value indicates better performance [7]. The AUC measure is calculated as the area of the ROC curve using the equation below.

$$AUC = \frac{1+TP_{rate}-FP_{rate}}{2} \quad ...(2)$$

The general guidelines used for the classification of the AUC value as used in the research of Wahono et al. [9] are as follows:

1. 0.90 - 1.00 = Excellent Classification
2. 0.80 - 0.90 = Good Classification
3. 0.70 - 0.80 = Fair Classification
4. 0.60 - 0.70 = Poor Classification
5. 0.50 - 0.60 = Failure

Below are the results of AUC evaluation of software defect prediction using D'' NASA MDP dataset MC1 and PC1. The model used to predict this dataset is the Deep Neural Network (DNN). The evaluation results are in the form of AUC performance values which can be seen in Table 4.

Table 4 AUC results

Dataset	AUC
MC1	0.815
PC1	0.889

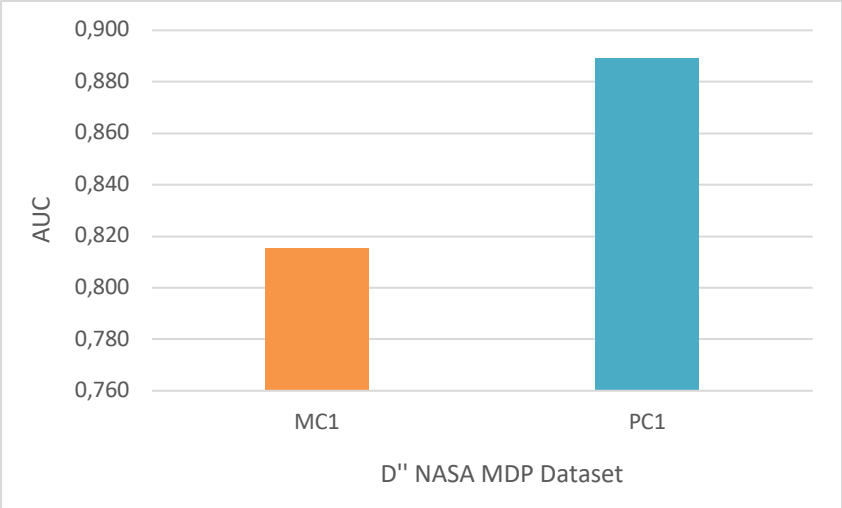

Figure 3 D'' NASA MDP Dataset AUC Results

According to AUC results are shown in Table 4, AUC of Deep Neural Network is obtained as 0.815 on MC1 dataset and 0.889 on PC1 dataset. MC1 has a higher AUC value than PC1 AUC value. Both AUC values obtained in the MC1 and PC1 datasets are included in Good Classification category based on general guidelines.

3.2 Discussion

Software defect prediction used D" NASA MDP datasets. D" NASA MDP dataset is different from the original NASA MDP dataset. D" NASA MDP dataset has been through a data cleaning process, so the dataset has no missing value than the original NASA MDP dataset, which has many missing values in each dataset.

Before dividing it into training data and testing data, preprocessing is done to make it easier for the model to train the data. The preprocessing step that needs to perform is not too much because the dataset has been cleaned, making the preprocessing step easier. The first thing to do at the preprocessing step is label encoding. This label encoding process helps change the shape of the labels on the dataset to a form that is easier for machine learning models to read because most machine learning algorithms work better with numerical data, especially for deep learning models. Data normalization was applied on the D" NASA MDP dataset. Data normalization needs to be applied because the dataset D" NASA MDP has a very varied range of values on each feature. The very varied range of values will affect the

classifier's performance so that it is necessary to normalize it. The technique used in the normalization stage is Min-Max with a scaling value between 0 and 1.

Deep Neural Network is used as a classification algorithm. Deep Neural Network needs to be trained first before being able to classify data into Defective and Non-Defective. We set optimizer, learning rate, batch size and epoch as follows:

- Optimizer : Adam
- Learning Rate : 0.01
- Batch Size : 64
- Epoch : 50

The data validation technique used in this work is Stratified 10-Fold Cross Validation. Stratified 10-Fold Cross Validation is a variation of Cross Validation. The difference between Stratified 10-Fold Cross Validation and 10-Fold Cross Validation in general is that Stratified 10-Fold can divide ten folds and ensure that the folds have the same number of classes. This technique is good to avoid when one of the folds contains only one of the two classes because the data used is unbalanced. Several tests also show that using stratification in the K-Fold Cross Validation validation can slightly improve the results [12].

AUC value is used in this work as an indicator to evaluate the software defect prediction model's performance. The AUC evaluation results are calculated by collecting the probability values obtained from the model prediction results and the original label from the validation results using the stratified k-fold cross validation in one data frame. After that, we calculate AUC based on data in the data frame. This AUC calculation technique is called the AUC Merge [2]. AUC results from the software defect prediction model using DNN obtained the AUC value as 0.815 on MC1 dataset and 0.889 on PC1 dataset. According to the general guidelines for the classification of AUC values, the AUC evaluation results on the MC1 and PC1 datasets are proved that the DNN model used on both datasets is a good classification model.

4. CONCLUSION

Early detection of software defects has an important role in the software development life cycle, especially at the software testing stage, affecting the quality of the software. Researchers have been developed various techniques in previous studies to overcome the problem of predicting software defects. This paper proposed Deep Neural Network as a classifier to classify the modules in the software metrics dataset into Defective and Non-Defective. According to the results of this research, the Deep Neural Network model's performance is reported using Area Under the Curve (AUC) for evaluation measurement. AUC of Deep Neural Network is obtained as 0.815 on MC1 dataset and 0.889 on PC1 dataset. Both AUC values obtained in the MC1 and PC1 datasets are included in Good Classification category based on general guidelines.

REFERENCES

[1] C. Catal. 2011. Software fault prediction: A literature review and current trends. *Expert Syst. Appl.* vol. 38, no. 4, pp. 4626–4636.

[2] Forman, G., Scholz, M. 2010. Apples-to-apples in cross-validation studies: pitfalls in classifier performance measurement. *ACM SIGKDD Explorations Newsletter*. 12(1):49–57.

[3] Hall, T., Beecham, S., Bowes, D., Gray, D., and Counsell, S. 2012. A Systematic Literature Review on Fault Prediction Performance in Software Engineering. *IEEE Transactions on Software Engineering*. vol. 38, no. 6, pp. 1276-1304, doi: 10.1109/TSE.2011.103.

[4] Harikesh, B. Yadav & Dilip, K. Yadav. 2015. Construction of Membership Function for Software Metrics. *Journal of International Conference on Information and Communication Technologies*.

[5] Japkowicz, N. 2013. Assessment Metrics for Imbalanced Learning. *Imbalanced Learning: Foundations, Algorithms, and Applications*.

[6] Jayanthi, R., Florence, L. Software defect prediction techniques using metrics based on neural network classifier. *Cluster Comput 22*. 77–88 (2019). https://doi.org/10.1007/s10586-018-1730-1

[7] Liu, Xu-Ying, & Zhou, Zhi-Hua. 2013. Ensemble Methods for Class Imbalance Learning. *Imbalanced Learning: Foundations, Algorithms, and Applications*.

[8] Malhotra, R., & S. Kamal. 2018. An Empirical Study to Investigate Oversampling Methods for Improving Software Defect Prediction using Imbalanced Data. *Neurocomputing Journal*.

[9] Manjula, C. & L. Florence. 2018. Deep Neural Network Based Hybrid Approach for Software Defect Prediction Using Software Metrics. *Cluster Computing*. 22: 9847–9863.

[10] Shepperd, M., Q. Song, Z. Sun & C. Mair. 2013. Data Quality: Some Comments on the NASA Software Defect Datasets. *IEEE Transactions On Software Engineering*. Volume 39, No. 9.

[11] Wahono, R. S., Herman, N. S., Ahmad, S. 2014. A Comparison Framework of Classification Models for Software Defect Prediction. *Advanced Science Letters*. 20: 1940-1945.

[12] Witten, I. H., Frank, E., & Hall, M. A. 2011. *Data Mining Third Edition*. San Fransisco: Elsevier Inc.