Software Defect Prediction: An Ensemble Learning Approach

Zhenyu Yang¹, Chufeng Jin¹, Yue Zhang¹, Jingjie Wang¹, Bingchang Yuan¹, Heng Li*¹

¹School of Computer Science and Engineering, Central South University, Changsha, Hunan, China
*email: liheng@csu.edu.cn

Abstract. Software defect prediction plays an increasingly critical role in emerging software systems. However, existing software defect prediction approaches typically suffer from low accuracy due to the under/over fitting problems. To address this problem, we propose an ensemble learning approach to achieve the accurate defect prediction, where various machine learning algorithms, i.e., artificial neural network, random forest, k-nearest neighbour methods are integrated together. The proposed software defect prediction workflow is introduced. Experiments are conducted to verify the effectiveness of the proposed method. Extensive experiment results verify that our proposed method can improve the defect prediction accuracy when compared with existing methods.

1. Introduction

With the continual expansion of software scale and the enhancement of function, the complexity of software is increasing, and the possibility of software defects is increasing, which is more likely to lead to software failures [1]. The most common software quality assurance activity is software testing, which can effectively check software errors, but also the most time-consuming and resource-consuming stage of the software development life cycle. Therefore, the concept of software defect prediction arises at the historic moment. Its purpose is to predict software defects as soon as possible according to the basic characteristics of software, allocate test resources reasonably according to the prediction results, give test priority, shorten development cycle, and reduce software cost[2].

At present, machine learning has been widely used in the field of software defect prediction[3]. From the existing research results, the defect prediction effect of single machine learning model is not ideal[4]. A single model either over-fits or lacks generalization ability. To solve this problem, we propose an ensemble learning model. Ensemble learning is a machine learning paradigm that is applicable to both supervised learning and unsupervised learning. We use k-nearest neighbour, logistic regression, random forest, and artificial neural network to build our ensemble leaning model and make an experiment.

In Section II we will introduce some related work about software defect prediction. In Section III, we will introduce the proposed software defect prediction approach. In section IV, we will show our experiment about how to integrate a high-performance software defect prediction model using our proposed algorithm. At last, we analyze the experiment result and make a conclusion.

2. Related work

2.1. Software defect prediction

Software defect prediction has attracted much attention in the field of software engineering. At the same time, it can increase the quality of software products[5], which can reduce development costs.
and improve development efficiency. It constructs the defect prediction model through the defect related metrics in the software, and predicts the software to identify the defects of the software with the use of the model. Related research includes the following three aspects:

2.1.1. **Software defect data processing.** Since software defect data are affected by various factors, there may be some problems in data reliability. Before the model construction, it is necessary to understand the data and carry out appropriate pretreatment work, such as standardization and normalization, so as to better construct the model in the future.

2.1.2. **Construction of software defect prediction model.** It is believed that software defect prediction models could be classified into three categories, namely, models based on supervised learning, models based on semi-supervised learning and models based on unsupervised learning. Their difference is whether the software defect data are sufficient. We need to select the appropriate model according to the number of existing defect data sets to achieve better results.

2.1.3. **Model evaluation.** After the model is constructed, some indicators needed to be used to evaluate the model to determine whether the performance of this model is excellent. Evaluation indexes includes Precision, Recall, ROC curve and AUC, and F1-measure. Firstly, the definition of confusion matrix is given, such as table 1:

| Tab. 1 Confusion matrix |
|--------------------------|
| Real Value               | Prediction         |
|                          | Defective          | Non-defective    |
| Defective                | TP(true positive)  | FN(false negative)|
| Non-defective            | EP(false positive) | TN(true negative)|

Several metrics are defined as follows:
- Precision = TP / (TP + FP).
- Recall rate recall = TP / (TP + FN).
- ROC curve and AUC index: ROC curve can be drawn with false positive rate as the horizontal axis and true positive rate as the vertical axis. The AUC index is the area which is enclosed by ROC curve and horizontal axis. Larger AUC value represents better model effect.
- F1-measure: Harmonized average of precision and recall. F1-measure = 2 * precision * recall / (precision + recall)

In addition, there are many indicators used to evaluate the software defect prediction model, such as MCC, G-mean and so on, but the above four are more representative.

2.2. **Ensemble learning algorithm and its application**

Ensemble Learning is a machine learning paradigm, which applies to both supervised learning and unsupervised learning. Ensemble learning uses different learners to solve a given problem[6]. Then it combines the results of group learning to compensate the error and make an improvement in the performance of the entire learning model[4]. The commonly used ensemble methods include boosting, bagging, random subspace method (RSM), stacking and voting-based ensemble methods.

As a kind of learning method for combined optimization, ensemble learning can not only obtain a combined model with better performance by combining multiple simple models, but also allow researchers to design combined model for specific machine learning problems to obtain more powerful solutions. Dietterich reveals three ensemble reasons for the success of ensemble learning from a mathematical perspective: statistics, computation, and representation. In addition, the effectiveness of ensemble learning can also be analyzed by deviation variance decomposition.

Ensemble learning algorithms are applied to all aspects of practical problems. In 2020, Ruszczak used ensemble learning algorithms to detect tomato Alternaria infection. In 2021, Selim Buyrukoglu use ensemble learning to detect Alzheimer Disease, Leiyu Dai do the research about landslide risk classification based on ensemble learning. In addition, ensemble learning is also widely used in information retrieval ranking learning, robot automatic outdoor navigation, drug activity detection, chat robot knowledge acquisition, network intrusion detection and gene data analysis.
3. The Proposed Software Defect Prediction Approach

This section will generally introduce some algorithms that have been implemented in the research process, including three machine learning algorithms and one ensemble learning algorithm based on artificial neural networks (ANN), k-nearest neighbour (KNN) and random forest (RF).

3.1. Artificial neural networks

The training of neural network is to adjust the free parameters through the stimulation of the neural network environment, so that the neural network can react to the external environment in a new way. The process of neural network training is to find the activation function using a large number of X and Y data.

Artificial neural network must learn with certain learning criteria before it can work. The learning process is divided into supervised learning and unsupervised learning. Supervised-learning is to give regular expected output, and make the actual output close to the expected output by adjusting the weight[7]. Unsupervised learning gives a measurement scale representing the quality of the method, and the parameters are optimized according to the scale.

3.2. K-nearest neighbour

KNN classification algorithm is a mature method in theory and one of the simplest machine learning algorithms. The idea is that if most of the k most similar samples in the feature space (i.e., the nearest in the feature space) belong to a category, the sample also belongs to that category[8]. The algorithm determines the category of samples to be classified according to the nearest samples. Because all the selected neighbourhoods are correctly classified objects.

3.3. Random forest

The algorithm uses bootstrap method to extract samples from the original data. After extraction, the decision tree model is constructed for the samples. Then, the prediction results are obtained by voting and comprehensively considering the prediction of multiple models.

RF is a combination of decision trees. Its basic principle is to generate different training sets by bagging method, that is, a new training set is generated from the original training set by bootstrap sampling [9]. For the new training set, the decision tree can be generated by using the random feature selection method. It should be noted that do not prune during the growth of the decision tree.

3.4. Ensemble-learning

Stacking method integrates models by layering. We will explain how it works at two levels, firstly, we divide the data set into training set and test set, and then train the training set to obtain some primary learners. Then, the training results are used to predict the test set, and the prediction results are used as the input value of the next stage. In addition, the output value of secondary learner training uses the final label (usually the last level uses logistic regression).

It can prevent over fitting to some extent, because the two stages use different training data. However, due to the need for multiple training, this method requires more training data than other methods [10]. In order to avoid that the proportion of the divided test set is too small, resulting in the weak generalization ability of the generated secondary learner, we will use cross validation or leave a method to train the Stacking algorithm [11].

4. Software defect prediction

This section first introduces the overall process of learning in detail, including data preprocessing and Hybrid ensemble model used in this study.

4.1. Data preprocessing

The data collected from different data sources are ensemble into a data set, and there may be problems such as data redundancy, data missing, outlier and noise in the data set. If introduced, these data may contribute to inaccurate defect prediction models. We apply the resampling strategy of artificially
synthesized new samples to optimize the unbalanced data set, so that the data with defects and the data without defects are in the same order of magnitude.

In this paper, source-based measurement results are taken as the characteristics of code measurement, and various code metrics have been proposed. Defect prediction data for these systems came from publicly available NASA data sets.

In a bid to improve the performance of model training, maximum-minimum standardization is uniformly applied to all data sets. Maximum-minimum standardization refers to linear transformation of original data. MinA and maxA refer to the minimum and maximum values of attribute A respectively, and an original value $x$ of A is mapped to the value $x'$ of the object $[0,1]$ through maximum-minimum standardization.

The maximum-minimum standardized calculation rules are as follows:

$$x' = \frac{x - \text{min}A}{\text{max}A - \text{min}A}$$

4.2. Hybrid ensemble model

The hybrid integration model is constructed on the basis of stacking. The Stacking algorithm starts with training the primary learner from the original data set, and then generates a new data set after training for training the secondary learner [12]. As for the new data set, the output of the primary learner can be regarded as a new input feature. The structure of Stacking is shown below:

![Fig. 1 Stacking structure](image_url)

The hybrid integration model is composed of the input layer, the hidden layer and the output layer. The individual learner in the model layer can be used as the classifier algorithm based on the support vector machine, Bayesian, decision tree, and so on [13]. The output of each node is equal to the input of the next layer node. In the output layer, a suitable classifier is generally selected, and the voting algorithm mechanism is used to obtain good classification performance.

5. Experiment results

Proposed method could be employed in software defect prediction. This section first preprocesses the data set, then compares the impact of stacking algorithm on defect prediction performance under...
different parameter configurations, and finally analyzes the results of defect prediction using stacking integral algorithm.

5.1. Parameter analysis of stacking algorithm

In the process of data set processing, constructing ANN is to find the filter threshold that is most used in defect prediction model. As shown in Figure 2, the network with a threshold of 25 already contains more than 70% of the real node information. It is noted that the AUC value at the threshold of 10 has exceeded that at the threshold of 25, but this is only an individual case and not universal. Moreover, the node information at the threshold of 10 is too small to accurately reflect the situation of the real network. Therefore, the filter threshold parameter in the previous defect prediction experiment in this paper was also set to 25. When using ANN as a primary learner of Stacking defect prediction models, the filter threshold parameter is set to 25 for optimal defect prediction performance.

![Fig. 2 The influence of different thresholds on ANN model](image)

5.2. Stacking model prediction results

In this experiment, the measurement results based on source code were used as the code measurement characteristics, and the training set was learned by primary and secondary learners. Finally, we selected the Stacking model with the best performance from multiple Stacking models, and the prediction results were as follows: the number of defects accounted for 9.1%, and the number of defects accounted for 90.9%. After repeated verification, we finally adopted random forest model, artificial neural network and K-nearest Neighbor model as the primary learner of our Stacking model, and logistic regression model as the secondary learner, so as to obtain the Stacking model with the highest fitting degree in THE NASA data set. The Stacking model received a higher ranking in all metrics compared to a single machine learning model.

Table 2 shows the AUC (Area Under Curve) values for NASA data set defects predicted by different classifiers based on different machine learning algorithm models. The AUC indicator is used to evaluate the accuracy and recall rate of the classifier results. For each software, the prediction results based on different traditional machine learning and Holdout verification of SF model are shown in Table 2. The average AUC value of the predicted results is shown in the AUC column. The accuracy and recall of classifier results are evaluated by AUC index. The accuracy of the predicted results is shown in the Precision column. Recall rate of predicted results is shown in the Recall column. The F1-score of the predicted results is displayed in the F1 column.
### Tab. 2 The performance comparison of different models

|       | AUC  | Accuracy | Precision | Recall | F1   |
|-------|------|----------|-----------|--------|------|
| LR    | 0.765| 0.902    | 0.828     | 0.590  | 0.845|
| KNN   | 0.819| 0.895    | 0.783     | 0.559  | 0.861|
| NB    | 0.712| 0.889    | 0.705     | 0.609  | 0.871|
| SVM   | 0.745| 0.878    | 0.697     | 0.712  | 0.892|
| ANN   | 0.833| 0.895    | 0.735     | 0.709  | 0.982|
| RF    | 0.816| **0.929**| **0.964** | 0.618  | 0.909|
| DT    | 0.774| 0.924    | 0.776     | 0.747  | 0.922|
| Proposed method | **0.958** | 0.925 | 0.853 | **0.853** | **0.956** |

The bold font in the table corresponds to the maximum value of each column, which is the optimal prediction result obtained when the software uses different models to predict defects. As can be seen from the table, the maximum values are distributed in RF and STACKING rows. For traditional machine learning algorithms, combined with Stacking model, the AUC, Recall, F1 and other evaluation indexes of defect prediction can be improved to achieve better results of defect prediction.

### 6. Conclusion

In this paper, we apply different machine learning modules and ensemble them to achieve ensemble learning. Before the training, the traditional metric features are normalized and verified by Holdout, and we take AUC value to measure the effect of defect prediction. On this basis, a Stacking model is proposed. It turns out that Stacking model does better than traditional single machine learning model.

While single machine learning model possesses high risk of over fitting and poor ability to generalize, stacking has less probability to overfit and better performance in accuracy measurement. However, there are still many details to be improved. As the defect prediction performance is sensitive to the parameters of the basic learner, the appropriate parameters should be selected.

### References

[1] Assim M, Obeidat Q and Hammad M 2020 Software Defects Prediction using Machine Learning Algorithms International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI) (IEEE) pp 1-6

[2] Thota M K, Shajin F H and Rajesh P 2020 Survey on software defect prediction techniques J. International Journal of Applied Science and Engineering 17(4)

[3] Geanderson E, Eduardo F, Adriano V, Markos V and Nivio Z 2020 Understanding machine learning software defect predictions J. Automated Software Engineering 27(3-4)

[4] Matloob F, Aftab S and Iqbal A 2019 A Framework for Software Defect Prediction Using Feature Selection and Ensemble Learning Techniques J. International Journal of Modern Education and Computer Science (IJMECS) 11(12)

[5] Yao W 2020 Research on Static Software Defect Prediction Algorithm Based on Big Data Technology J. International Conference on Virtual Reality and Intelligent Systems (ICVRIS) (IEEE) pp 610-613

[6] Jiang P C 2021 Research on software defect prediction technology based on deep learning 2nd International Conference on Computing and Data Science (CDS) pp 104-107

[7] Kussul N, Lavreniuk M, Skakun S and Shelestov A 2017 Deep Learning Classification of Land Cover and Crop Types Using Remote Sensing Data J. IEEE Geoscience and Remote Sensing Letters 14(5) pp 778-782

[8] Grazia C, Donato I, Vincenzo D, et al. 2021 Human Gait Analysis in Neurodegenerative Diseases: a Review J. IEEE journal of biomedical and health informatics p 1
[9] Joanne B, Emmanuel M and JeanMarc L 2020 Determination of biological joint reaction forces from in-vivo experiments using a hybrid combination of biomechanical and mechanical engineering software J. Mechanics & Industry 21(6)

[10] Li Q, Wang S L, Zhao C F, et al. 2020 HIBOG: Improving the clustering accuracy by ameliorating dataset with gravitation J. Information Sciences (Prepublish)

[11] Koziel S and Dabrowska A P 2019 Reliable data-driven modeling of high-frequency structures by means of nested kriging with enhanced design of experiments J. Engineering Computations 36(7)

[12] Liang T T, Sheng X, Zhou L, et al. 2021 Mobile app recommendation via heterogeneous graph neural network in edge computing J. Applied Soft Computing Journal 103 (Prepublish)

[13] Mohanraj T, Yerchuru Jayanthi, Krishnan H, et al. 2020 Development of tool condition monitoring system in end milling process using wavelet features and Hoelder’s exponent with machine learning algorithms J. Measurement (Prepublish)