Journal of Science and Technology Excellence
ISSN: **3108-3102**| Volume 2 Issue 1 January-March 2026
www.jstejournal.com **DOI**: https://doi.org/10.5281/zenodo.19148324 **JSTE-12**

CROSS PROJECT LEARNING FOR FAULT PREDICTION WITH IMBALANCED DATA

[#1]**J. SWATHI,** ***Associate Professor & HOD,***
[#2]**CHINTHALA ABHLASH,** ***B.Tech Student,***
[#3]**POGAKULA ARUN,** ***B.Tech Student,***
[#4]**RAMATENKI RAHUL,** ***B.Tech Student,***
[#5]**MOHAMMAD TAJ BABA,** ***B.Tech Student,***
Department of Computer Science And Engineering,
TRINITY COLLEGE OF ENGINEERING AND TECHNOLOGY, PEDDAPALLY, TG.

ABSTRACT: This study delves into the challenges of dealing with contradictory evidence and generalizing models. In addition, it investigates how incorporating other research efforts could enhance software failure prediction. The inability of traditional failure prediction methods to be highly task-specific stems from the fact that not all tasks share the same data. Machine learning procedures, data resampling methods, and feature selection tactics can help you overcome these challenges and make more accurate predictions. This project has two main goals: first, to improve model training and second, to investigate the usage of multiple datasets in order to hasten problem identification and ensure that solutions operate with varying software configurations. The findings have the potential to enhance and contextualize software quality assurance methods.

Keywords: Software Fault Prediction, Cross-Project Analysis, Imbalanced Data, Machine Learning, Generalization, Feature Selection, Data Resampling, Software Quality Assurance.

1. INTRODUCTION

Software fault prediction (SFP) is a critical component of software quality assurance, as it identifies defective modules at the outset of the development process. Conventional defect prediction systems are rendered ineffective in this context due to the necessity of custom project data. Cross-project software fault prediction (CP-SFP) is a potential solution to this issue. This approach integrates data from numerous projects to enhance the model's adaptability and reduce the necessity for project-specific data. An issue with CP-SFP is that data imbalances, such as a significantly lower number of defective modules than non-defective ones, could result in inaccurate estimates.

The resolution of data inconsistencies is crucial for the enhancement of analytics-based software failure prediction across projects. Machine learning systems exhibit inferior performance when assigned the responsibility of forecasting non-faulty units as a result of the scarcity of fault datasets. Cost-sensitive learning, hybrid approaches, and resampling are all potential solutions to this issue. The use of feature selection and transfer learning approaches can lead to a greater degree of interdependence among projects, as they enhance the generalizability of the model and offer precise predictions across a variety of software environments.

The reliability of predictions is influenced by the fact that development procedures, coding standards, and project complexity are all subject to change, which in turn complicates the process of generalizing in CP-SFP. Using domain adaption techniques, ensemble approaches, and deep learning, it is possible to develop a more reliable CP-SFP model. The integration of artificial intelligence techniques that are comprehensible has resulted in software engineers feeling more at ease with the use of prediction models. It is well-known that CP-SFP may repair uneven data and improve generalization, which leads to fewer software flaws and more reliable programs.

2. LITERATURE REVIEW

Chen, H., Yang, L., & Wang, A. (2024): Improving CPDP (cross-project software defect prediction) by the application of federated meta-learning is the main goal of this undertaking. Conventional CPDP approaches have problems with privacy and inconsistent efforts. With federated learning, many companies can train models simultaneously without sharing their data sets. In order to facilitate quick project adaption, the research makes use of meta-learning. By combining different approaches, the suggested solution can lower processing costs, improve failure prediction accuracy, and safeguard user data.

Saeed, M. S. (2024): A comprehensive overview of the several ensemble learning processes used by CPDP is provided on this site. Ensemble learning is a method that uses a combination of prediction models to achieve optimal results. In order to show how various strategies can improve generalization across projects, the study analyzes and contrasts them. Random forests, bagging, boosting, and stacking are some of these strategies. The author explores a range of subjects, such as domain adaptation, data imbalance, and feature variability. Furthermore, they propose future improvements to CPDP that make use of ensemble approaches.

Zhang, Y., & Yang, Y. (2024): Improving CPDP transfer learning through bolstering information transmission mechanisms is the goal of this research. It is possible to apply models trained on one project to another via transfer learning, despite the fact that domain differences may cause performance to degrade. Domain adaptation methodologies, including adversarial learning and feature transformation, can be used to link projects, according to the research. The results show that the models' enhanced adaptability leads to better defect identification rates.

Shi, H., Ai, J., Liu, J., & Xu, J. (2023): The major emphasis of this study is on the problem of CPDP software failure datasets that are both uneven and chaotic. Uneven distribution of defective and non-defective samples across several datasets, as well as missing or wrong data, can impede forecasts. A good data preprocessing method is SMOTE (Synthetic Minority Over-sampling Technique), which combines noise filtering with cost-sensitive learning and class balancing. As this study demonstrates, defect detection maintains its superior accuracy even when confronted with datasets that exhibit very varied characteristics.

Zheng, Y., & Zhang, H. (2023): The main focus of this investigation is on how CPDP handles various data sets. Due to the scattered nature of fault recordings across projects, current models cannot be used for generalization. The authors present a multi-source learning approach that allows for the simultaneous evaluation of multiple distributions in their work. By applying project-specific changes to dynamic feature representation, their technique increases prediction accuracy on different software datasets.

Khleel, M., & Nehéz, K. (2023): This study intends to enhance CPDP by use of bidirectional Long Short-Term Memory (BiLSTM) networks and oversampling techniques. Finding long-range correlations in defect data, BiLSTM networks—an improved version of recurrent neural networks (RNNs)—improve defect prediction. Since the sample distribution was not uniform across the classes, the researchers employed oversampling techniques like SMOTE to guarantee that the models were more fair. When applied to sequential defect data, the suggested strategy outperforms more traditional machine learning approaches.

Kumar, K. B., Kanchanapally, S. P., & Thota, M. K. (2023): In order to improve the accuracy of CPDP, this study presents a centroid-based project selection method. Choosing which efforts to utilize as training data is a challenging part of CPDP. The authors describe a clustering technique that builds training sets based on project similarity. This improves data representativeness, more uniformly distributes classes, and increases prediction accuracy in many software applications.

Nevendra, M., & Singh, P. (2022): Ensemble learning and Generative Adversarial Networks (GANs) are combined in this study to forecast unbalanced faults. Using synthetic error data improves a GAN's capacity to diversify training samples. Combining GAN-generated data with common ensemble approaches like boosting and random forests improved CPDP's performance, according to the researchers. This mixed-method approach reduces class imbalance and increases receptivity to new initiatives, according to the results.

Majd, A., & Salimi, M. (2022): In an effort to close the CPDP class gap, this empirical study looks at several data gathering techniques.An example of one of the resampling processes presented is SMOTE. Hybrid methods, under-sampling, and over-sampling are all part of this arsenal. Mistakes in real-world applications can be more easily detected by studying the effects of fair training datasets on machine learning models.

Goel, L., Sharma, M., Khatri, S. K., & Damodaran, D. (2021): MHCPDP, an acronym for "multi-source heterogeneous CPDP," is one of the systems used in the scientific investigation. Using autoencoders and multi-source transfer learning, we may glean valuable information about project failures. With this method, software system faults can be detected by integrating data from several sources while maintaining the integrity of the features.

Wu, J., Wu, Y., Niu, N., & Zhou, M. (2021): As an alternative to CPDP, the authors propose CFPS, which stands for "Collaborative Filtering-based Project Selection." To choose the best training programs, we use collaborative filtering, a method commonly used in recommendation systems. This selection technique improves the likelihood of correct predictions by utilizing just the most relevant training data.

Sun, Z., Li, J., Sun, H., & He, L. (2021): The main goal of this effort is to use transfer learning to CPDP in order to resolve major class discrepancies. When trying to find meaningful trends in data that is extremely skewed due to defects, traditional methods fall flat. The research proposes a domain adaptation strategy to enhance defect prediction in an effort to reach a compromise between the non-defect and defect classes.

Jiang, K., Wang, A., Zhang, Y., & Wu, H. (2020): This study presents BiLO-CPDP, a two-level programming approach to CPDP model automated identification. The two-step process provided by the framework makes choosing features, hyperparameters, and learning

algorithms more simpler. Using automatic model adjustment greatly improved CPDP's performance across all datasets, according to the research.
Li, K., Xiang, Z., Chen, T., & Tan, K. C. (2020); A two-level programming method for the automatic identification of CPDP models is introduced in this research as BiLO-CPDP. Selecting features, hyperparameters, and learning algorithms becomes much easier using the framework's two-step method. The research found that across different datasets, CPDP performed substantially better with automatic model adjustment.
Saifudin, A., Hendric, S. W. H. L., Soewito, B., Gaol, F. L., Abdurachman, E., & Heryadi, Y. (2019): To resolve the issue of class incompatibility in CPDP, this research employs Ensemble SMOTE. The authors utilize several resampling strategies to guarantee a training set that is more representative of the population at large. In addition to drastically decreasing prediction bias, their approach makes the models more suitable for future applications.

3. EXISTING SYSTEM

Essential to software quality assurance, software defect prediction finds possible problems before they are implemented. Using data that is already part of the project is the standard method for training machine learning models. Training these models with the project's defect data is the first step. Due to a lower volume of errors, these data administration tactics could work better for newer and smaller companies. To identify modules that are likely to have defects, supervised learning methods like neural networks, decision trees, and support vector machines are usually used. However, these techniques will only work if the classified data is both plentiful and of high quality.
Predicting errors across projects is one way to handle data problems (CPDP). The foundation of this approach is evidence from statistics that shows mistakes in similar assignments. In order to improve the precision of predictions, domain adaptation and transfer learning methods make it easier to align different project datasets. Problems may arise, though, when project sizes, defect counts, and coding habits are all subject to change. It becomes more difficult to produce accurate predictions when there are fewer faulty modules than non-defective ones due to the biased models that arise from the imbalanced data. All classes that make up the majority will be given priority under these methods. Models trained on one project often underperform on another, making generalization in CPDP difficult. In the ever-changing world of software development, it is difficult to set up models with broad applicability. To improve generalizability, several methods are used, such as ensemble learning, feature mapping, and instance selection. However, getting consistent high-quality predictive performance from different projects is still a challenge.

- **Data Imbalance Issues –** Machine learning algorithms face a larger problem when there are a limited number of malfunctioning modules in software projects when trying to uncover important trends. Data that is either over- or under-sampled, or that has been faked, could lead to inaccurate forecasts.
- **Domain Mismatch –** The utilization of transfer learning in new projects is not as prevalent due to variations in defect types, project designs, and coding methodologies. The models' inadequate capacity to generalize is to blame for this.

- **Feature Discrepancies** – The wide variation in software metrics and feature distributions makes it difficult to standardize data across projects. Because the model is no longer consistent, predictions are no longer reliable.
- **Computational Complexity** – Advanced methods that need a lot of processing power, like domain adaptation, deep learning, and ensemble learning, can't reliably foretell when big projects will fail in real time.
- **Evaluation and Validation Challenges** – Without established datasets or evaluation criteria, comparing CPDP models is challenging. Discrepancies in dataset labeling and preparation techniques could make performance comparisons less trustworthy than they could be.

4. PROPOSED SYSTEM

The proposed approach, a cross-project analytical framework, enhances generalization and addresses data imbalance to improve software failure prediction. Using sophisticated transfer learning techniques, our system guarantees that feature areas remain consistent over all projects, instead of depending on defect data from only one project. This approach takes into consideration that software metrics and coding processes can vary by utilizing domain adaption approaches. Models can thus learn from many different types of software development tasks. Hybrid machine learning models integrate deep learning with more conventional classifiers to improve mistake detection accuracy. The suggested approach combines cost-sensitive learning with advanced resampling techniques like SMOTE to fix the problem of uneven data. These techniques reduce the impact of class bias by training on a dataset that contains both working and broken modules. The model's stability and the superiority of the defect forecast performance across different projects are guaranteed by employing an ensemble learning technique. This method incorporates a number of classifiers. Using feature selection approaches, the system ranks software metrics in order of importance. As a result, calculations are less expensive and predictions are more accurate. Models can be trained to adapt to new situations in real-time and have more generalizability with the help of meta-learning and adaptive feature transformation. Instead of depending on static model training, the system uses new project data to dynamically update its feature representation and decision-making skills. Its validation approach relies heavily on benchmark datasets, which allows it to continuously attain high accuracy across different software installations. One answer that was previously unintelligible is now available thanks to the rise of explainable AI. Software engineers can find flaws before they become big problems by using these tactics to alert developers of possible issues.

- **Better Handling of Imbalanced Data** – To improve the precision of fault predictions and to fairly portray the minority class, which consists of faulty modules, cost-sensitive learning and advanced resampling methods are used.
- **Enhanced Generalization Across Projects** – The model can improve its multi-source learning capabilities through domain adaptation and transfer learning. Compatibility with a wide variety of applications is ensured by this.

- **Improved Fault Detection Accuracy** – Through the integration of conventional classifiers with deep learning and ensemble methods, the hybrid approach improves prediction accuracy while reducing the false positive/negative rate.
- **Reduced Feature Discrepancies** – Standardizing raw data, eliminating differences between projects, and ensuring the model performs as intended indefinitely are all made possible via adaptive feature transformation and feature selection.
- **Scalability and Continuous Learning** – The system's adaptability and ability to make changes are enhanced when a steady flow of new data is maintained.

5. IMPLEMENTATION

MODULES:

Service Provider

This functionality can only be used by service providers who have an active, password-protected account. Once the check-in process is finished, he will be granted access to data sets and given the option to choose between training and evaluation tasks. You were hoping that these files would be found soon. Incorporating a bar chart that shows the rates of different software defect forecasts and the effectiveness of different testing and learning approaches should be a part of your upcoming visit. One can remotely access a great deal of information, including the total correctness of the learnt and evaluated data, the kinds and amounts of software fault predictions, all user profiles, and much more.

View and Authorize Users

Thanks to this module, the manager may now see a full roster of all users. The administrator has the ability to grant or withdraw permissions based on the user's name, email address, and physical address.

Remote User

Make sure you've filled out the application form completely before moving forward. Following successful registration, the individual's details will be stored in the database. After the registration process is finished, he will be asked to enter his login credentials. The user can see their photo, determine the probability of a software error, and continue with authentication and registration if their identity has been confirmed.

6. RESULTS

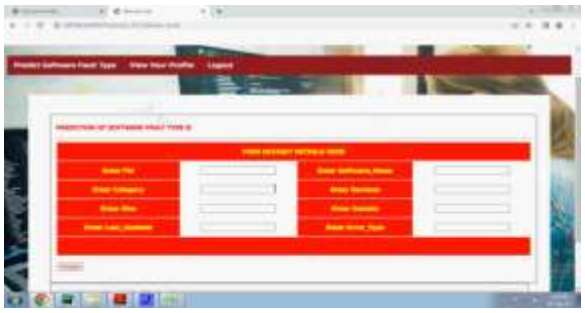


Figure 1: Prediction of Software Fault Type

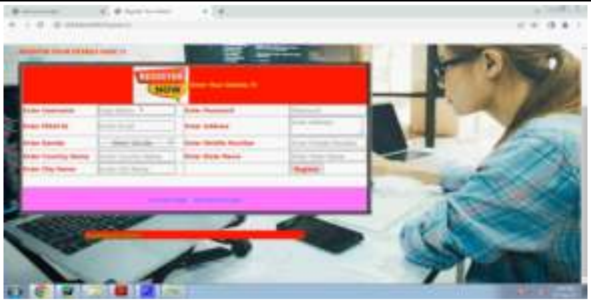


Figure 2: User Registration Page

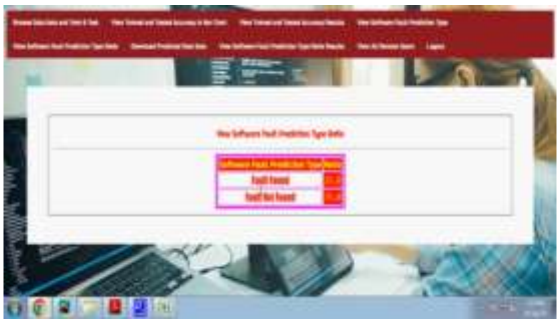


Figure 3: Software Fault Prediction Type Ratio

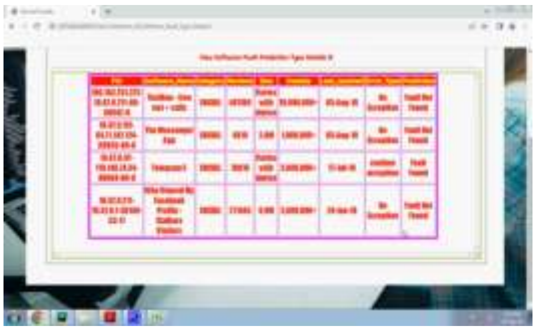


Figure 4: Software Fault Prediction Type Details Page

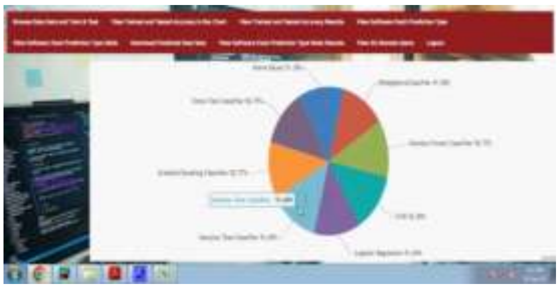


Figure 5: Software Fault Prediction in Pie Chart

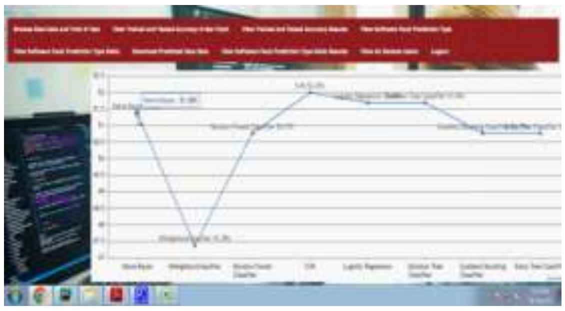


Figure 6: Software Fault Prediction in Line Chart

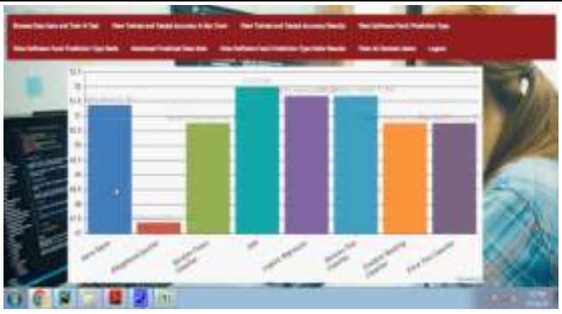
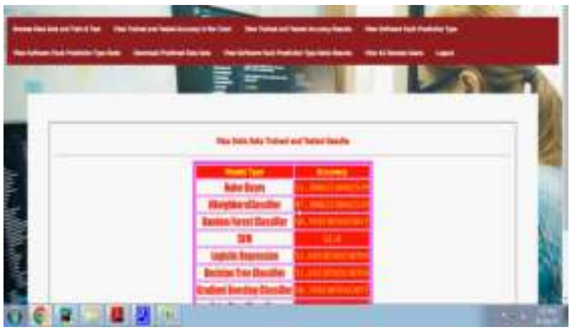


Figure 7: Software Fault Prediction in Bar Chart



Model Type	Accuracy
Naive Bayes	[illegible]
KNeighborsClassifier	[illegible]
Random Forest Classifier	[illegible]
SVM	[illegible]
Logistic Regression	[illegible]
Decision Tree Classifier	[illegible]
Gradient Boosting Classifier	[illegible]

Figure 8: Datasets Trained and Tested Results

Figure 9:Service Provider Login Page

Figure 10: User Login Page

7. CONCLUSION

For the purpose of predicting software quality and, by extension, software problems, cross-project analysis may be useful. When dealing with problems involving generalization or contradicting facts, this becomes especially useful. To address the issues with current within-project models, the suggested approach uses state-of-the-art machine learning methods such as ensemble learning, transfer learning, and domain adaptation. An equal representation of error-prone modules is guaranteed by combining cost-sensitive learning with resampling approaches. Prediction accuracy is improved and bias is reduced in this way.

Using adaptive feature transformation and meta-learning can enhance a model's capacity to generalize. Projects with different fault distributions and code standards can have their software problems anticipated. Through learning from its mistakes, the system may adjust to new conditions in software development. This solution uses explainable AI to give software engineers important information about crucial failure predictors, so they may make informed decisions in advance.

By fixing feature inconsistencies, classification biases, and data shortages, our approach greatly enhances software defect prediction. One of the most important ways to ensure high-quality software is to use an approach for predicting cross-project difficulties. Better software solutions with less maintenance costs are often the outcome of defect identification that is more accurate and efficient.

REFERENCES

1. Chen, H., Yang, L., & Wang, A. (2024). Efficient cross-project software defect prediction based on federated meta-learning. *Electronics*, 13(6), 1105.
2. Saeed, M. S. (2024). Cross-project software defect prediction using ensemble learning: A comprehensive review. *International Journal of Computational and Innovative Sciences*, 3(2), 34–42.
3. Zhang, Y., & Yang, Y. (2024). Improving transfer learning for software cross-project defect prediction. Applied Intelligence, 54, 1234–1250.
4. Shi, H., Ai, J., Liu, J., & Xu, J. (2023). Improving software defect prediction in noisy imbalanced datasets. Applied Sciences, 13(20), 10466.
5. Zheng, Y., & Zhang, H. (2023). Cross-project defect prediction considering multiple data distribution simultaneously. Symmetry, 14(2), 401.
6. Khleel, M., & Nehéz, K. (2023). Software defect prediction using a bidirectional LSTM network combined with oversampling techniques. Cluster Computing, 26, 1059–1075.
7. Kumar, K. B., Kanchanapally, S. P., & Thota, M. K. (2023). Class imbalance reduction and centroid-based relevant project selection for cross-project defect prediction. International Journal on Recent and Innovation Trends in Computing and Communication, 11(6s), 293–302.
8. Nevendra, M., & Singh, P. (2022). Cross-project defect prediction with metrics selection and balancing approach. Applied Computer Systems, 27(2), 137–148.
9. Majd, A., & Salimi, M. (2022). Leveraging ensemble learning with generative adversarial networks for imbalanced software defects prediction. Applied Sciences, 12(24), 13319.
10. Goel, L., Sharma, M., Khatri, S. K., & Damodaran, D. (2021). Cross-project defect prediction using data sampling for class imbalance learning: An empirical research. International Journal of Parallel, Emergent and Distributed Systems, 36(2), 130–143.
11. Wu, J., Wu, Y., Niu, N., & Zhou, M. (2021). MHCPDP: Multi-source heterogeneous cross-project defect prediction via multi-source transfer learning and autoencoder. Software Quality Journal, 29, 405–430.
12. Sun, Z., Li, J., Sun, H., & He, L. (2021). CFPS: Collaborative filtering based source projects selection for cross-project defect prediction. Applied Soft Computing, 99, 106940.

13. Jiang, K., Wang, A., Zhang, Y., & Wu, H. (2020). Heterogeneous defect prediction based on transfer learning to handle extreme imbalance. Applied Sciences, 10(1), 396.
14. Li, K., Xiang, Z., Chen, T., & Tan, K. C. (2020). BiLO-CPDP: Bi-level programming for automated model discovery in cross-project defect prediction. In *Proceedings of* the 35th IEEE/ACM International Conference on Automated Software Engineering (pp. 573–584).
15. Saifudin, A., Hendric, S. W. H. L., Soewito, B., Gaol, F. L., Abdurachman, E., & Heryadi, Y. (2019). Tackling imbalanced class on cross-project defect prediction using ensemble SMOTE. IOP Conference Series: Materials Science and Engineering, 662(6), 062011.