IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

IEEE RELIABILITY SOCIETY SECTION

Received 12 October 2022, accepted 13 December 2022, date of publication 21 December 2022, date of current version 9 January 2023.

Digital Object Identifier 10.1109/ACCESS.2022.3231456

RESEARCH ARTICLE

Improving Cross-Project Software Defect Prediction Method Through Transformation and Feature Selection Approach

YAHAYA ZAKARIYAU BALA[1,3], PATHIAH ABDUL SAMAT[1], KHAIRONI YATIM SHARIF[1], AND NORIDAYU MANSHOR[2]

[1]Department of Software Engineering and Information System, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Serdang 43400, Malaysia
[2]Department of Computer System, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Serdang 43400, Malaysia
[3]Department of Computer Science, Federal University of Kashere, Gombe 771103, Nigeria

Corresponding author: Pathiah Abdul Samat (pathiah@upm.edu.my)

This work was supported in part by University Putra Malaysia, and in part by Tedfund Nigeria.

ABSTRACT In the traditional software defect prediction methodology, the historical record (dataset) of the same project is partitioned into training and testing data. In a practical situation where the project to be predicted is new, traditional software defect prediction cannot be employed. An alternative method is cross-project defect prediction, where the historical record of one project (source) is used to predict the defect status of another project (target). The cross-project defect prediction method solves the limitations of the historical records in the traditional software defect prediction method. However, the performance of cross-project defect prediction is relatively low because of the distribution differences between the source and target projects. Furthermore, the software defect dataset used for cross-project defect prediction is characterized by high-dimensional features, some of which are irrelevant and contribute to low performance. To resolve these two issues, this study proposes a transformation and feature selection approach to reduce the distribution difference and high-dimensional features in cross-project defect prediction. A comparative experiment was conducted on publicly available datasets from the AEEEM. Analysis of the results obtained shows that the proposed approach in conjugation with random forest as the classification model outperformed the other four state-of-the-art cross-project defect prediction methods based on the commonly used performance evaluation metric F1_score.

INDEX TERMS Cross-project, feature selection, software defect, transformation.

LIST OF ABBREVIATIONS

AUC	Area Under Receiver Operating Characteristics Curve.
CKSDL	Cost-sensitive Semi-supervised Dictionary Learning.
CPDP	Cross-Project Defect Prediction.
CTDP	Collective Transfer Learning Defect Prediction.
DBSCAN	Density-Based Spatial Clustering of Applications.
DPC	Density Peaks Clustering.
EM	Expectation-Maximization.
EMD	Earth Mover Distance.
FCR	Feature Class Relevance.
Fr	Feature Relevance.
FRV	Feature Relevance Vector.
FS	Feature Selection.
FV	Feature Vector.
GA	Genetic Algorithm.
GIS	Genetic Instance Selection.
HYDRA	Hybrid Model Reconstruction Approach.
KNN	K-Nearest Neighbor.
LDF	Local Density of Features.
LR	Logistic Regression.

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Ali Babar.

MCW	Multiple Components Weight.
MFTCPDP	Manifold Feature Transformation Cross-Project Defect Prediction.
NASA	National Aeronautics and Space Administration.
NB	Naïve Bayes.
NN	Neural Network.
RF	Random Forest.
S	Source Project.
SDP	Software Defect Prediction.
SFD	Similarities of Features Density.
SF	Selected Features.
SMET	Source Mean Transformation.
SMOT	Source Mode Transformation.
St	Transformed Source.
Sr	Reduced Source.
SVM	Support Vector Machine.
T	Target Project.
TCA+	Transfer Component Analysis plus.
TDS	Training Data Selection.
TFSCPDP	Transformation and Feature Selection Cross-Project Defect Prediction.
Tr	Reduced Target.
UMR	Unified Metric Representation.
WPDP	Within Project Defect Prediction.

I. INTRODUCTION

Currently, there is a significant increase in software complexity [1]. An increase in the size and complexity of the system over time has led to an increase in the demand for efficient methods that can minimize cost and maximize the quality of the software product. In response, testing activities are conducted to verify the correctness of the system's operation through its execution. Defects can be identified by testing. The early identification of defects in the development process can help reduce the cost of maintenance [2]. However, the cost of software testing is almost half of the development cost [3]. Consequently, in a situation where resources are limited, managing the available limited resources is paramount.

A software defect prediction (SDP) model can be applied to detect the most defect-prone component of a software system prior to testing activities, thereby reducing the number of resources required to test every module of the software system [4], [5], [6], [7], [8]. Thus, maintainable and high-quality software can be produced, despite the low budget.

The effectiveness of the SDP model depends on the accuracy of the training set. The training set is normally extracted from previous records of the same project. However, in practice, for new projects or companies that lack local defect data repositories because of the cost of maintaining such repositories, it will be very difficult to apply SDP [9], [10].

An alternative to the traditional SDP is cross-project defect prediction (CPDP), in which the historical data of one project (source) available in repositories are used to predict defects in another project under development (target) [11], [12], [13]. The CPDP resolves the issue of limited data in traditional SDP. However, the predictive performance of CPDP is below the applicability level because of the distribution discrepancy between the source and target projects [1], [14]. Another factor affecting the performance of CPDP is the issue of high-dimensional features in both the source and target project (dataset), some of which are outliers and irrelevant [15]. Therefore, reducing these two issues has become a focus of CPDP research.

Researchers have proposed various approaches to improve the performance of CPDP. This includes the Burak-instance filtering technique proposed by Turhan et al. [9], the Peters instance filtering technique proposed by Peters et al. [16], instance weighting proposed by Ma et al. [17], a combination of multiple models proposed by Xia et al. [18], the collaborative filtering method proposed by Sun et al. [19], transfer component analysis (TCA+) proposed by Nam et al. [20], cost-sensitive semi-supervised dictionary learning (CKSDL) proposed by Wu et al. [21], heterogeneous CPDP using the dictionary learning method proposed by Li et al. [22], multi-source CPDP proposed by Bai et al. [23], and Zhao et al. [24], CPDP based on domain adaptation proposed by Jin [25], and the landmark-selection-based kernelized discriminant subspace alignment (LSKDSA) method for CPDP proposed by Li et al. [26].

Analyses of the methods above revealed that methods developed based on feature transformation improved the predictive performance of the CPDP technique better than the others. However, improvements in the predictive performance of CPDP remain open. We propose a cross-project defect prediction based on transformation and feature selection technique called TFSCPDP. The basic idea behind TFSCPDP is to improve the predictive performance of the CPDP by reducing the issues of distribution differences and high-dimensional features.

Experiment was conducted on the AEEEM datasets to ascertain the effectiveness of the TFSCPDP using the commonly used evaluation metric F-measure. The TFSCPDP outperformed the benchmark approaches based on the experimental results.

The paper is structured as follows: the existing approaches in the area of cross-project defect prediction are briefly described in Section 2; in Section 3, the proposed transformation and feature selection approaches are introduced; Section 4 describes the experimental procedures adopted in the study, which include the performance evaluation metrics, datasets, and result analysis; and Section 5 presents the conclusion and future work.

II. RELATED WORKS

A. CROSS-PROJECT DEFECT PREDICTION

A feasibility study on the CPDP technique was conducted by Briand et al. [27] using a model trained on the historical data of one java project and predicted defects in another java project developed in the same environment.

Zimmermann et al. [10] conducted a large-scale experiment on CPDP. Their results showed that only 3.4% was successful because of the data distribution discrepancy between the source and target projects. This triggered a series of research projects on improving the predictive performance of the CPDP technique.

Some studies have used a similarity measure to choose suitable training data for the CPDP. He et al. [28] and Herbold [29] computed statistical indicators of the features of source projects and constructed a new distribution vector. A similarity measure was used to select the most similar source project for the target project and used as training data for the CPDP. Turhan et al. [9] proposed a Burak filtering method in which k instances in source project that are most similar to the instances in the target project are selected and used as training data for CPDP. Peters et al. [16] proposed an improvement in the Burak filter method in which a clustering algorithm was used to select training data from instances in a source project. Yuan et al. [30] proposed an approach that has two phases; phase1: features from both the source and target projects are clustered using the density peaks clustering (DPC) technique. Phase2: features in the cluster are selected based on three ranking methods: local density of features (LDF), similarities of features (SFD), and feature class relevance (FCR). The selected features were then used as the training dataset for the CPDP.

Some studies have developed semi-supervised cross-project defect prediction methods. Wu et al. [31] proposed a novel approach to reduce the gap between source and target projects. In this approach, intermediate features from the original datasets are extracted and irrelevant instances from the source dataset are illuminated using an autoencoder. Xu et al. [32] proposed a novel approach in which semantic features are pulled directly from the source code instead of using the label of the software defect dataset. A CPDP model was built using the extracted source code for cross-project defect prediction.

Currently, studies have focused on developing the CPDP method using multiple source projects. Poon et al. [33] proposed a credibility-based naïve Bayes model in which a reweighting technique was used to reduce the distributional difference between source and target projects. Wen et al. [34] proposed an approach in which four different techniques were used for source selection (mean-log, median-log, median-score, and std-log) and feature selection techniques such as relief, correlation, oneR, InfoGain, and Gain Ratio. Ren et al. [35] proposed a multisource CPDP model based on the dissimilarity space. In this approach, cluster centers are automatically selected from the target project to form a prototype set. To form a dissimilarity space, the dissimilarity between the source, target, and prototype is calculated using the arccosine method. The earth mover distance (EMD) was used to compute the cost of transforming the data distribution of both training and testing data to be similar. Source projects with low costs were chosen as the training datasets. TrAdaBoost was used as the prediction model. Chen et al. [36] proposed a novel approach called collective transfer learning defect prediction (CTDP). This approach consists of two phases. In the first phase, TCA+ was used to expand the training datasets using four different normalization techniques. Source projects with low costs were chosen as the training datasets. The best classifier was built using expanded training data. In the second phase, the particle swamp optimization algorithm was used to form an ensemble classifier by allocating adaptive weights to each base classifier. Xing et al. [37] proposed an approach in which the distribution difference in the CPDP was reduced by filtering both the source and target projects using correlation-based feature selection with a greedy best-first search algorithm. Ni et al. [38] proposed a novel search-based approach for source data selection, called genetic instance selection (GIS), in which appropriate training datasets were identified based on the fitness F-measure and G-measure output using the KNN filter technique. Liu et al. [39] proposed an approach known as two-phase transfer learning for CPDP. In the first phase, a component called source project estimator (SPE) used two regression models to select the appropriate source project. In the next phase, the two selected projects, conjugated with TCA+, were used to build two prediction models. Finally, the consolidated results obtained from the two prediction models were considered.

B. TRANSFORMATION

Transformation in the context of CPDP is the process by which the source and target project are mapped from original feature space to the common feature space [1].

Some studies have focus on reducing the data distribution differences problem in CPDP through feature transformation and migration. Nam et al. [20] proposed transfer component analysis plus TCA+ method based on feature transformation and migration, in which the TCA was improved by adding normalization (data pre-processing technique). After normalization of the data, TCA+ maps the datasets of both the source and target projects into the latent space where the distribution discrepancy is minimized. Ma et al. [17] proposed a method in which instance weighting and transfer learning are used to predict the distribution of a target project in order to improve the predictive performance of CPDP. Zhao et al. [1] proposed a method based on manifold feature transformation. The source and target were transformed into manifold space to reduce the difference in the data distribution. Finally, the NB classifier was trained on the transformed source project. Ni et al. [38] proposed a CPDP method in which features with high distribution similarity between the source and target projects are selected using cluster algorithm. Fan et al. [40] proposed a method in which instances from source project were extracted and transformed into training data with high correlation with the target data.

An analysis of the methods above revealed that the feature transformation method can effectively improve the predictive performance of the CPDP. However, only a few studies considered the problem of high dimensions in the dataset

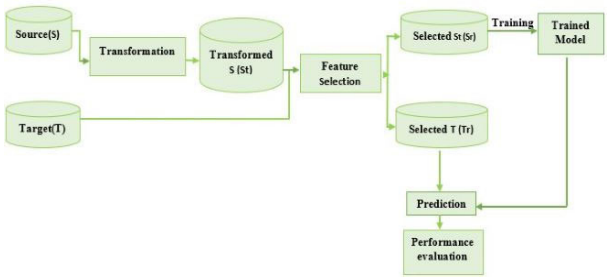


FIGURE 1. Framework for TFSCPDP.

used for CPDP. Therefore, we propose a cross-project defect prediction method based on the transformation and feature selection approach TFSCPDP.

III. PROPOSED APPROACH

In this section, the framework for our approach is described, followed by transformation and the feature selection algorithm and finally the CPDP model construction algorithm.

A. PROPOSED APPROACH FRAMEWORK

As shown in Figure 1. The inputs are the source (S) and the target project (T). First, a new source project (St) is obtained after transformation. Second, St and T pass through the feature selection process. Third, the classifier is trained on a reduced-source project (Sr). Finally, the trained model is used to predict the label of the reduced target project (Tr), and the result is then compared with the real label (performance analysis).

B. PROPOSED TRANSFORMATION APPROACH

Let's define the source and the target project as $S = \{(F_j^{(s)})_{j=1}^{m}, y^{(s)}\}$ and the target project dataset $T = \{(F_j^{(t)})\}_{j=1}^{m}$ where $F_j^{(s)}$ and $F_j^{(t)} \epsilon R^{mx1}$ are the feature vectors with Sn and Tn instances respectively which represent the analysis of source code of software program module. $y^{(s)}$ is a label which denotes the defect-proneness of a module. $y^{(s)} \epsilon \{0, 1\}$ where $y^{(s)} = 0$ and $y^{(s)} = 1$ denotes defective and non-defective module respectively. Assuming that, both the source and the target are homogeneous (have the same metric sets). In cross project settings, the distance between the distributional characteristic of features of the source and the target project $d\left(F_j^{(s)}, F_j^{(t)}\right) = |F_j^{(s)}, F_j^{(t)}|$ is large. Where $d()$ denotes a Euclidean distance. Therefore, the goal of this approach is to minimize $d\left(F_j^{(s)}, F_j^{(t)}\right)$. let's the source feature $F_j^{(s)}$ contains instances $x_{1,j}^{(s)}, x_{2,j}^{(s)}, x_{3,j}^{(s)} \ldots x_{sn,m}^{(s)}$ and corresponding target feature $F_j^{(t)}$ contains instances $x_{1,j}^{(t)}, x_{2,j}^{(t)}, x_{3,j}^{(t)} \ldots x_{tn,m}^{(t)}$ from i = 1 to Sn, Tn and j = 1 to m, then the Source Mean Transformation (SMET) is given by Eq.(1).

$$SMET = \frac{(x_{i,j}^{(s)} * \hat{F}_j^{(t)})}{\hat{F}_j^{(s)}} \tag{1}$$

Source Mode Transformation (SMOT) is given by Eq. (2).

$$SMOT = \frac{(x_{i,j}^{(s)} * \widetilde{F_j^{(t)}})}{\widetilde{F_j^{(s)}}} \tag{2}$$

where $\hat{F}_j^{(s)}$ and $\hat{F}_j^{(t)}$ are the mean of each feature in S and T calculated using Eq. (3) and (4) respectively.

$$\hat{F}_j^{(s)} = \frac{1}{sn} \sum_{i=1}^{sn} x_{i,j}^{(s)} \tag{3}$$

$$\hat{F}_j^{(t)} = \frac{1}{tn} \sum_{i=1}^{tn} x_{i,j}^{(t)} \tag{4}$$

$\widetilde{F_j^{(s)}}$ and $\widetilde{F_j^{(t)}}$ represent the mode of each feature in S and T calculated using Eq. (5) and (6) respectively

$$\tilde{F_j^{(s)}} = \max(x_{1,j}^{(s)}, x_{2,j}^{(s)}, x_{3,j}^{(s)} \ldots x_{sn,m}^{(s)}) \tag{5}$$

and

$$\tilde{F_j^{(t)}} = \max(x_{1,j}^{(t)}, x_{2,j}^{(t)}, x_{3,j}^{(t)} \ldots x_{tn,m}^{(t)}) \tag{6}$$

Transformed source project (St) is given by Eq. (7)

$$St = \{(F_j^{(St)})_{j=1}^{m}, y^{(s)} \tag{7}$$

Whare $F_j^{(St)}$ represents the arithmetic mean of SMET and SMOT calculated using Eq. (8)

$$F_j^{(St)} = 1/2\left(\frac{\left(x_{i,j}^{(s)} * \hat{F}_j^{(t)}\right)}{\hat{F}_j^{(s)}} + \frac{\left(x_{i,j}^{(s)} * \tilde{F_j^{(t)}}\right)}{\tilde{F_j^{(s)}}}\right) \tag{8}$$

C. PROPOSED FEATURE SELECTION APPROACH

The characteristics of software modules are reflected in the software features [41]. However, some software contains many features some of which are irrelevant and contribute in decreasing the prediction performance of software defect prediction model especially CPDP. Therefore, introducing feature selection technique to select more relevant features can improve the predictive performance of software defect prediction model [42], [43], [44].

IV. DESCRIPTION OF DATASETS AND EVALUATION METRICS

A. DATASETS

The AEEEM datasets are some of the open-source datasets commonly used in SDP studies [1]. We can observe from Table 1 that AEEEM contains five software projects (datasets) with 61 features each.

B. EVALUATION METRICS

Three evaluation metrics or measurements were used to evaluate the proposed approach: recall, precision, and F1_score. F1_score is a common evaluation metric used in software defect prediction studies [34]. This is the harmonic representation of precision and recall, calculated using Eq. (9).

$$\boldsymbol{F1\_score} = \frac{(\boldsymbol{2 * recall * precision})}{\boldsymbol{recall + precision}} \tag{9}$$

Algorithm of Proposed Feature Selection Approach

Input: St-Transformed Source project
T-Target project
N-Number of features to be selected
Output: S_r
T_r

1: compute the mean of each feature $F_j^{(s)}$ of S for j=1 to m and mark it as F_r (feature representative)

$$\mathrm{Fr}_j^s = \sum_{i=1}^{sn} x_{1,j}^{(s)}$$

2: store each F_r in a feature vector (**FV**)
$\boldsymbol{FV} = \{F_{r1}^s, F_{r2}^s, \ldots F_{rm}^s\}$

3: compute the arithmetic mean FV_{mean} of FV

$$FV_{mean} = 1/m \sum_{j=1}^{m} F_{rj}^{(s)}$$

4: compute the standard deviation FV_{std} for FV

$$FV_{std} = \sqrt{\frac{\sum (F_{rj}^s - FV_{mean})^{(2)}}{m-1}}$$

5: for each feature $F_j^{(s)}$, compute the Feature Ranking Value (FRV)

$$FRV_j = (3 * FV_{std}) - abs(F_{r1}^s - FV_{mean})$$

6: set the values of FRV_j in descending order
7: select the top Nth FRV along with their corresponding features and mark it as Selected Features (SF)
8: select the common features between the SF and St and SF and T and mark the resulting datasets as Sr and Tr respectively
Return Sr, Tr

TABLE 1. Datasets.

Project	#Modules	#Features	#Defect	Defect Ratio
EQ	325	61	129	40%
JDT	997	61	206	21%
ML	1862	61	245	13%
PDE	1492	61	209	14%
LC	399	61	64	9%

V. EXPERIMENTAL RESULTS

A. EXPERIMENTAL SETUP

In this subsection, the parameter settings are described. For the proposed feature selection approach, we set N = 12. Four state-of-the-art approaches, manifold feature transformation cross-project defect prediction MFTCPDP [1], multi-source cross-project defect prediction MSCPDP [24], kernel twin support vector machines with domain adaptation DA-KTSVM [25], and two-phase feature importance amplification TFIA-CPDP [37], were employed to verify the effectiveness of the proposed approach. Four classifiers commonly used in SDP studies, namely, KNN, SVM, RF, and LR, were adopted in this study [45], [46], [47]. Table 4 presents the results for different machine learners in terms of the F1-score. As shown in Table 4, the highest average F1_score was obtained by RF followed by SVM, which was used for the analysis of our proposed approach. All experiments in this study were implemented using Python. In addition, all methods were evaluated using the same metric F1_score on the same dataset, AEEEM, for a fair comparison. Experiments were performed on a PC with the following properties: Windows 10, 64-bit, Intel(R) Core (TM) i7-4600U CPU:2.70 GHz, RAM:8G.

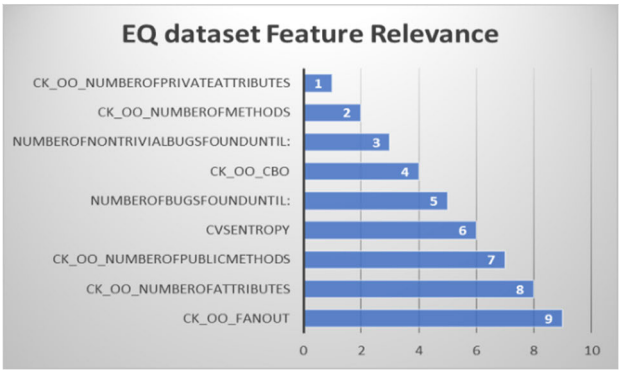


FIGURE 2. Relevant features in EQ data.

B. PREDICTIVE PERFORMANCE OF TFSCPDP BASED ON DIFFERENT VALUES OF PARAMETER N (NUMBER OF SELECTED FEAUTRES)

In this subsection, a comparative study of TFSCPDP based on different values of the parameter N using RF and SVM learners is presented.

1) PREDICTIVE PERFORMANCE OF TFSCPDP BASED ON DIFFERENT VALUES OF PARAMETER N USING RF LEARNER ON AEEEM DATASET

We analyzed the predictive performance of TFSCPDP based on the value of parameter N using RF on AEEEM, as shown in Table 2. TFSCPDP's best performance was obtained with a value of N = 12, considering the number of times it won on individual data (indicated in bold) and the overall F1_score average. Therefore, we can deduce that the best performance of TFSCPDP can be obtained by selecting twelve (12) relevant features using RF as a classifier on the AEEEM, as shown in Table 2.

2) PREDICTIVE PERFORMANCE OF TFSCPDP BASED ON DIFFERENT VALUES OF PARAMETER N USING SVM LEARNER ON AEEEM

Secondly, we analyzed the performance of TFSCPDP based on the value of parameter N using SVM on the AEEEM

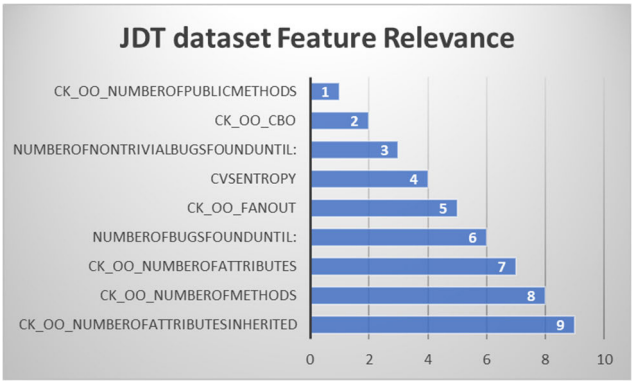


FIGURE 3. Relevant features in JDT data.

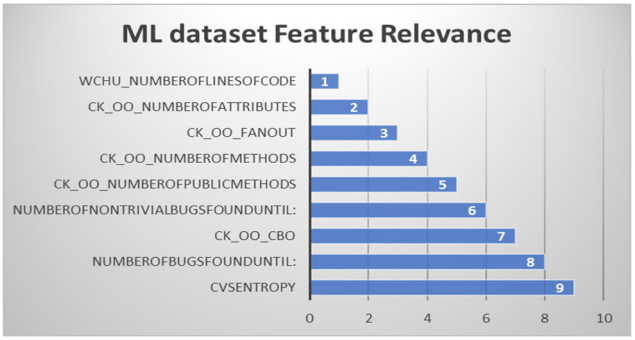


FIGURE 4. Relevant features in ML data.

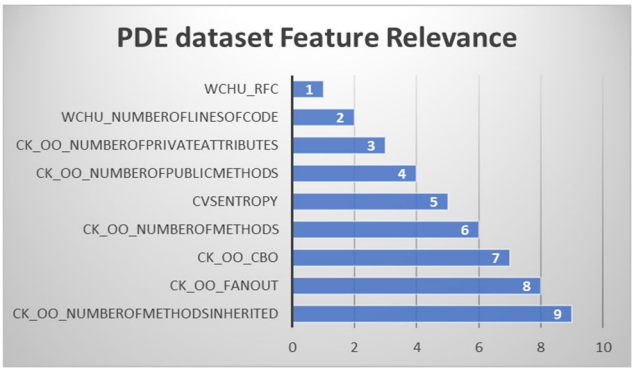


FIGURE 5. Relevant features in PDE data.

dataset, as shown in Table 3, the best performance of TFSCPDP was obtained with a value of N = 18 considering the number of times it won on individual data (indicated in bold) and the overall F1_score average on both the datasets. Therefore, we can deduce that the best performance of TFSCPDP can be obtained by selecting eighteen (18) relevant features using SVM as a classifier on AEEEM datasets.

C. PREDICTIVE PERFORMANCE OF THE PROPOSED TFSCPDP USING DIFFERENT LEARNERS ON AEEEM DATASETS

To verify whether the predictive performance of TFSCPDP can be further improved using suitable learners, we investigated the performance of TSFCPDP using different learners (KNN, SVM, RF, and LR) on AEEEM datasets using F1_score as the evaluation metric. For this experiment, the value of parameter N was set to (N = 12), and for KNN, the value of K was set to (K = 3) [23]. As shown in Table 4, TFSCPDP with RF achieved the highest F1_score (mean of 0.80), followed by SVM with an F1_score (mean of 0.79) on the AEEEM dataset. These results indicate that RF and SVM are the best learners for SDP. This conforms with the findings of Alsawalqah et al. [45] and Elish and Elish [46] on RF and SVM, respectively.

TABLE 2. Predictive perfromance of TFSCPDP based on different values of parameter N using RF on AEEEM dataset.

Source→Target	N=9	N=12	N=15	N=18
EQ→JDT	0.75	0.69	0.71	**0.79**
EQ→LC	0.69	0.77	**0.81**	0.75
EQ→ML	0.65	**0.69**	0.65	0.54
EQ→PDE	**0.61**	0.57	0.5	0.58
JDT→EQ	0.72	**0.73**	0.71	0.72
JDT→LC	0.79	**0.9**	0.74	0.82
JDT→ML	0.81	**0.83**	0.81	0.81
JDT→PDE	0.79	0.8	**0.85**	0.8
LC→EQ	0.74	**0.75**	0.72	0.73
LC→JDT	0.86	**0.87**	0.86	0.86
LC→ML	0.82	**0.87**	0.85	**0.87**
LC→PDE	0.87	0.88	0.87	**0.89**
ML→EQ	0.73	0.73	**0.74**	0.72
ML→JDT	0.83	**0.86**	**0.86**	**0.86**
ML→LC	0.91	**0.92**	0.89	0.91
ML→PDE	0.86	0.86	**0.87**	0.85
PDE→EQ	**0.74**	**0.74**	**0.74**	**0.74**
PDE→JDT	**0.85**	0.84	**0.85**	**0.85**
PDE→LC	**0.92**	0.9	**0.92**	0.86
PDE→ML	**0.88**	**0.88**	0.86	0.85
Mean	0.79	**0.8**	0.79	0.79

D. RELEVANT FEATURES SELECTED BY PROPOSED TSFCPDP

We demonstrated the relevant features selected by our proposed TFSCPDP during the feature selection process. Fig. 2, 3, 4, and 5 present the relevance of the features. This analysis

TABLE 3. Predictive perfromance of TFSCPDP based on different values of parameter N using SVM on AEEEM dataset.

Source→Target	N=9	N=12	N=15	N=18
EQ→JDT	0.86	0.86	**0.87**	**0.87**
EQ→LC	**0.87**	0.86	0.86	**0.87**
EQ→ML	**0.85**	0.84	0.81	**0.85**
EQ→PDE	0.61	0.63	**0.68**	0.66
JDT→EQ	0.71	**0.72**	**0.72**	**0.72**
JDT→LC	0.72	0.7	**0.73**	**0.73**
JDT→ML	**0.84**	**0.84**	**0.84**	0.82
JDT→PDE	0.79	0.8	0.8	**0.82**
LC→EQ	0.71	**0.72**	0.7	0.71
LC→JDT	0.85	0.85	**0.86**	**0.86**
LC→ML	0.81	0.82	0.74	**0.85**
LC→PDE	0.79	**0.81**	0.79	0.79
ML→EQ	**0.7**	0.68	0.67	0.65
ML→JDT	**0.84**	**0.84**	**0.84**	**0.84**
ML→LC	0.6	0.67	**0.69**	0.66
ML→PDE	0.75	**0.78**	0.74	0.73
PDE→EQ	**0.7**	0.69	**0.7**	0.69
PDE→JDT	**0.86**	**0.86**	**0.86**	**0.86**
PDE→LC	0.82	**0.86**	0.78	0.85
PDE→ML	**0.89**	**0.89**	**0.89**	0.88
Mean	0.78	**0.79**	0.78	**0.79**

indicated that the selected features have a high tendency to describe software defects.

E. COMPARISON OF TFSCPDP WITH OTHER APPROACHES FROM THE LITERATURE

Finally, the performance of the TFSCPDP was compared with that of four state-of-the-art CPDP methods. MFTCPDP was proposed by Zhao et al. [1], MSCPDP by Zhao et al. [24], and DA-KTSVM by Jin Cong. [25] and the TFIA-CPDP proposed by Xing et al. [37].

The experiment was conducted on the AEEEM datasets, and the following results were obtained: Table 5 presents the performance comparison between the four current state-of-the-art CPDP methods on the AEEEM datasets. The proposed TFSCPDP achieved the highest F1_score with (a mean of 0.80) and we can also observe that the proposed TFSCPDP outperforms the other approaches in many individual datasets (indicated in bold). Furthermore, we can also observe that

TABLE 4. Comparison of the proposed TFSCPDP using different learners on AEEEM dataset in terms of F1_score.

Source→Target	KNN	SVM	RF	LR
EQ→JDT	0.72	**0.87**	0.71	0.35
EQ→LC	0.61	**0.86**	0.81	0.56
EQ→ML	0.56	**0.81**	0.65	0.29
EQ→PDE	0.52	**0.68**	0.50	0.60
JDT→EQ	0.62	**0.72**	0.71	**0.72**
JDT→LC	0.65	0.73	**0.74**	0.62
JDT→ML	0.63	**0.84**	0.81	0.67
JDT→PDE	0.75	0.80	**0.85**	0.76
LC→EQ	0.66	0.70	0.72	**0.73**
LC→JDT	0.82	**0.86**	**0.86**	0.76
LC→ML	0.78	0.74	**0.85**	0.55
LC→PDE	0.72	0.79	**0.87**	0.77
ML→EQ	0.65	0.67	**0.74**	0.67
ML→JDT	0.76	0.84	**0.86**	0.79
ML→LC	0.76	0.69	**0.89**	0.67
ML→PDE	0.72	0.74	**0.87**	0.76
PDE→EQ	0.69	0.70	**0.74**	0.71
PDE→JDT	0.73	**0.86**	0.85	0.75
PDE→LC	0.76	0.78	**0.92**	0.70
PDE→ML	0.74	**0.89**	0.86	0.67
Mean	0.69	0.78	0.79	0.66

the proposed TFSCPDP outperforms other methods across various combinations and on average scale as well.

1) STATISTICAL VALIDATION

To validate our proposed approach, the Wilcoxon rank sum test, a non-parametric statistical tool commonly used in SDP [14], [17], [38], was used to measure the significant difference between the proposed approach and the four current state-of-the-art CPDP methods at the 5% significance level. Four groups of Wilcoxon rank-sum tests were constructed: TFSCPDP and MFTCPDP, TFSCPDP and MSCPDP, TFSCPDP and DA-KTSVM, and TFSCPDP and TFIA-CPDP, denoted as TSF&MF, TSF&MS, TSF&DA, and TSF&TF, respectively. Statistical p-values obtained were 0.02 for TSF&MF, 0.00 for TSF&MS, 0.00 for TSF&DA, and 0.44 for TSF&TF. The p-values of TFS&MF and TFS&MS, and TFS and DA were less than the significant value of 0.05, indicating that the proposed TFSCPDP significantly outperformed the MFTCPDP, MSCPDP, and DA-KTSVM. The p-value of TFS&TF is higher than the significant value of 0.05, indicating that the difference between the proposed TFSCPDP and TFIA-CPDP is not statistically significant. However, TFSCPDP outperformed TFIA-CPDP on average and on the individual datasets (indicated in bold), as shown in Table 5.

VI. THREATS TO VALIDITY

The TFSCPDP proposed in this article has the following threats to the validity

1) Although the predictive performance of TFSCPDP on 10 different datasets is better than that of the current state-of-the-art CPDP methods, this does not guarantee that the same result can be obtained on other datasets.

TABLE 5. Comparison of TFSCPDP with RF classifier to other approaches from the literature on AEEEM dataset based on F1_score results.

Source→Target	MFTCP DP	DA-KTSVM	MSCP DP	TFIA CPDP	TFSCP DP
EQ→JDT	0.53	0.54	0.41	0.79	**0.90**
EQ→LC	0.84	0.47	0.26	0.76	**0.92**
EQ→ML	0.76	0.52	0.24	0.74	**0.90**
EQ→PDE	0.72	0.56	0.30	**0.78**	0.75
JDT→EQ	0.56	0.56	0.23	**0.83**	0.69
JDT→LC	**0.89**	0.55	0.26	0.75	0.73
JDT→ML	**0.84**	0.53	0.26	0.75	0.74
JDT→PDE	0.83	0.62	0.28	0.76	**0.87**
LC→EQ	0.67	0.34	0.31	**0.83**	0.69
LC→JDT	0.57	0.25	0.49	**0.82**	0.73
LC→ML	0.78	0.45	0.30	0.77	**0.84**
LC→PDE	0.78	0.50	0.27	0.78	**0.87**
ML→EQ	0.62	0.56	0.16	**0.86**	0.57
ML→JDT	0.61	0.55	0.31	0.79	**0.80**
ML→LC	**0.88**	0.53	0.12	0.82	0.86
ML→PDE	0.82	0.52	0.23	0.76	**0.88**
PDE→EQ	0.63	0.59	0.23	**0.88**	0.77
PDE→JDT	0.72	0.68	0.39	0.81	**0.83**
PDE→LC	**0.88**	0.51	0.16	0.78	0.86
PDE→ML	0.83	0.34	0.22	0.75	**0.88**
Mean	0.74	0.51	0.27	0.79	**0.80**

2) We carefully selected the most commonly used F1_score as the evaluation metric in this study. However, other evaluation metrics may yield different results.

VII. CONCLUSION AND FUTURE STUDY

This study proposes a new transformation and feature selection approach to improve the performance of cross-project defect prediction. The main aim of the transformation was to reduce the distribution difference between the source and target projects, while the feature selection was to reduce the issues of high dimensionality and irrelevant features.

In the transformation phase, first, the mean distributional characteristic of the source project was used for transformation; second, the mode distributional characteristic of the source was also used for transformation; and finally, the arithmetic mean of the duo was computed and subsequently considered as the transformed source. In the feature selection phase, the distance of each feature of the transformed source was computed from three times the standard deviation. Features with large distances are selected. Our approach effectively improved the performance of CPDP by obtaining a mean F-score of 0.80, which is the highest among the existing CPDP approaches. Therefore, we can conclude that our approach not only reduces the distribution difference between the source and the target but also effectively reduces the impact of high dimensionality and irrelevant features in CPDP. Our findings have proven that TFSCPDP can be useful for SDP applications, particularly when dealing with limited historical records and large software applications that have multiple features.

For future studies in CPDP research, we noticed from the experimental results that a suitable classifier can also improve the performance of CPDP in conjugation with the TFSCPDP approach. Therefore, in the future, it is necessary to focus on combining multiple classifiers to further improve the performance of CPDP.

REFERENCES

[1] Y. Y. Z. Zhu and Q. X. Y. Chen, "Cross-project defect prediction method based on manifold feature transformation," *Future Internet*, vol. 13, no. 216, pp. 1–17, 2021.

[2] B. Boehm and R. Turner, "Management challenges to implementing agile processes in traditional development organizations," *IEEE Softw.*, vol. 22, no. 5, pp. 30–39, Sep. 2005.

[3] W. Rhmann and G. A. Ansari, "Ensemble techniques-based software fault prediction in an open-source project," in *Research Anthology on Usage and Development of Open-Source Software*. Hershey, PA, USA: IGI Global, 2021, pp. 693–709.

[4] L. N. Gong, S. J. Jiang, and L. Jiang, "Research progress of software defect prediction," *J. Softw.*, vol. 30, pp. 3090–3114, Aug. 2019.

[5] X. Chen, Q. Gu, W. Liu, S. Liu, and C. Ni, "Survey of static software defect prediction," *J. Softw.*, vol. 27, no. 1, pp. 1–25, 2016.

[6] H. Tracy, B. Sarah, B. David, D. Gray, and S. A. Counsell, "A systematic literature review on fault prediction performance in software engineering," *IEEE Trans. Softw. Eng.*, vol. 38, no. 6, pp. 1276–1304, Nov. 2012.

[7] Y. Li, Z. Q. Huang, Y. Wang, and B. W. Fang, "Survey on data driven software defects prediction," *Acta Electron.*, vol. 45, pp. 982–988, Apr. 2017.

[8] Z. Li, X.-Y. Jing, and X. Zhu, "Progress on approaches to software defect prediction," *IET Softw.*, vol. 12, no. 3, pp. 161–175, Jun. 2018.

[9] B. Turhan, T. Menzies, A. B. Bener, and J. Di Stefano, "On the relative value of cross-company and within-company data for defect prediction," *Empirical Softw. Eng.*, vol. 14, no. 5, pp. 540–578, 2009.

[10] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, and B. Murphy, "Cross-project defect prediction: A large scale experiment on data vs. domain vs. process," in *Proc. 7th Joint Meeting Eur. Softw. Eng. Conf. ACM SIGSOFT Symp. Found. Softw. Eng.*, New York, NY, USA, 2009, pp. 91–100.

[11] S. Herbold, A. Trautsch, and J. Grabowski, "Global vs. local models for cross-project defect prediction," *Empirical Softw. Eng.*, vol. 22, no. 4, pp. 1866–1902, Aug. 2017.

[12] X. Chen, L. P. Wang, Q. Gu, Z. Wang, C. Ni, W. S. Liu, and Q. Wang, "A survey on cross-project software defect prediction methods," *Chin. J. Comput.*, vol. 41, pp. 254–274, Jan. 2018.

[13] S. Chen, J. M. Ye, and T. Liu, "Domain adaptation approach for cross-project software defect prediction," *J. Softw.*, vol. 31, no. 2, pp. 266–281, 2020.

[14] S. Tang, S. Huang, C. Zheng, E. Liu, C. Zong, and Y. Ding, "A novel cross-project software defect prediction algorithm based on transfer learning," *Tsinghua Sci. Technol.*, vol. 27, no. 1, pp. 41–57, Feb. 2022.

[15] A. Saifudin and Y. Yulianti, "Dimensional reduction on cross project defect prediction," *J. Phys., Conf. Ser.*, vol. 1477, no. 3, Mar. 2020, Art. no. 032011.

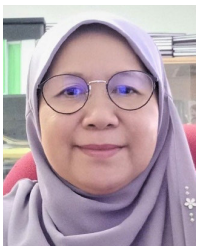
[16] F. Peters, T. Menzies, and A. Marcus, "Better cross company defect prediction," in *Proc. 10th Work. Conf. Mining Softw. Repositories*, San Francisco, CA, USA, May 2013, pp. 409–418.

[17] Y. Ma, G. Luo, X. Zeng, and A. Chen, "Transfer learning for cross-company software defect prediction," *Inf. Softw. Technol.*, vol. 54, no. 3, pp. 248–256, 2012.

[18] X. Xia, D. Lo, S. J. Pan, N. Nagappan, and X. Wang, "HYDRA: Massively compositional model for cross-project defect prediction," *IEEE Trans. Softw. Eng.*, vol. 42, no. 10, pp. 977–998, Oct. 2016.

[19] Z. Sun, J. Li, H. Sun, and L. He, "CFPS: Collaborative filtering based source projects selection for cross-project defect prediction," *Appl. Soft Comput.*, vol. 99, Feb. 2021, Art. no. 106940.
[20] J. Nam, S. J. Pan, and S. Kim, "Transfer defect learning," in *Proc. 35th Int. Conf. Softw. Eng. (ICSE)*, San Francisco, CA, USA, May 2013, pp. 382–391.
[21] F. Wu, X. Y. Jing, Y. Sun, J. Sun, L. Huang, F. Cui, and Y. Sun, "Cross-project and within-project semisupervised software defect prediction: A unified approach," *IEEE Trans. Rel.*, vol. 67, no. 2, pp. 581–597, Jun. 2018.
[22] Z. Q. Li, X. Y. Jing, F. Wu, X. K. Zhu, B. W. Xu, and S. Ying, "Cost-sensitive transfer kernel canonical correlation analysis for heterogeneous defect prediction," *Autom. Softw. Eng.*, vol. 25, no. 2, pp. 201–245, Aug. 2017.
[23] J. Bai, J. Jia, and L. F. Capretz, "A three-stage transfer learning framework for multi-source cross-project software defect prediction," *Inf. Softw. Technol.*, vol. 150, Oct. 2022, Art. no. 106985.
[24] Y. Zhao, Y. Zhu, Q. Yu, and X. Chen, "Cross-project defect prediction considering multiple data distribution simultaneously," *Symmetry*, vol. 14, no. 2, p. 401, Feb. 2022.
[25] C. Jin, "Cross-project software defect prediction based on domain adaptation learning and optimization," *Expert Syst. Appl.*, vol. 171, Jun. 2021, Art. no. 114637.
[26] Z. Li, J. Niu, X.-Y. Jing, W. Yu, and C. Qi, "Cross-project defect prediction via landmark selection-based kernelized discriminant subspace alignment," *IEEE Trans. Rel.*, vol. 70, no. 3, pp. 996–1013, Sep. 2021.
[27] L. C. Briand, W. L. Melo, and J. Wust, "Assessing the applicability of fault-proneness models across object-oriented software projects," *IEEE Trans. Softw. Eng.*, vol. 28, no. 7, pp. 706–720, Jul. 2002.
[28] P. He, Y. He, L. Yu, and B. Li, "An improved method for cross-project defect prediction by simplifying training data," *Math. Problems Eng.*, vol. 2018, pp. 1–18, Jun. 2018.
[29] S. Herbold, "Training data selection for cross-project defect prediction," in *Proc. 9th Int. Conf. Predictive Models Softw. Eng.*, 2013, pp. 1–10.
[30] Z. Yuan, X. Chen, Z. Cui, and Y. Mu, "ALTRA: Cross-project software defect prediction via active learning and TrAdaBoost," *IEEE Access*, vol. 8, pp. 30037–30049, 2020.
[31] J. Wu, Y. Wu, N. Niu, and M. Zhou, "MHCPDP: Multi-source heterogeneous cross-project defect prediction via multi-source transfer learning and autoencoder," *Softw. Quality J.*, vol. 29, no. 2, pp. 405–430, 2021.
[32] Z. Xu, P. Yuan, T. Zhang, Y. Tang, S. Li, and Z. Xia, "HDA: Cross-project defect prediction via heterogeneous domain adaptation with dictionary learning," *IEEE Access*, vol. 6, pp. 57597–57613, 2018.
[33] W. N. Poon, K. E. Bennin, J. Huang, P. Phannachitta, and J. W. Keung, "Cross-project defect prediction using a credibility theory based naive Bayes classifier," in *Proc. IEEE Int. Conf. Softw. Quality, Rel. Secur. (QRS)*, Jul. 2017, pp. 434–441.
[34] W. Wen, B. Zhang, X. Gu, and X. Ju, "An empirical study on combining source selection and transfer learning for cross-project defect prediction," in *Proc. IEEE 1st Int. Workshop Intell. Bug Fixing (IBF)*, Feb. 2019, pp. 29–38.
[35] S. Ren, W. Zhang, H. S. Munir, and L. Xia, "Dissimilarity space based multi-source cross-project defect prediction," *Algorithms*, vol. 12, no. 1, p. 13, Jan. 2019.
[36] J. Chen, K. Hu, Y. Yang, Y. Liu, and Q. Xuan, "Collective transfer learning for defect prediction," *Neurocomputing*, vol. 12, p. 91, Nov. 2019.
[37] Y. Xing, W. Lin, X. Lin, B. Yang, and Z. Tan, "Cross-project defect prediction based on two-phase feature importance amplification," *Comput. Intell. Neurosci.*, vol. 2022, pp. 1–14, Apr. 2022.
[38] C. Ni, X. Chen, and W. S. Liu, "Cross-project defect prediction method based on feature transfer and instance transfer," *J. Softw.*, vol. 30, pp. 1308–1329, Jan. 2019.
[39] C. Liu, D. Yang, X. Xia, M. Yan, and X. Zhang, "A two-phase transfer learning model for cross-project defect prediction," *Inf. Softw. Technol.*, vol. 107, pp. 125–136, Mar. 2019.
[40] G. Fan, X. Diao, H. Yu, and I. Chen, "Cross-project defect prediction method based on instance filtering and transfer," *Comput. Eng.*, vol. 46, pp. 197–202, Jan. 2020.
[41] T. M. Khoshgoftaar, K. Gao, A. Napolitano, and R. Wald, "A comparative study of iterative and non-iterative feature selection techniques for software defect prediction," *Inf. Syst. Frontiers*, vol. 16, no. 5, pp. 801–822, 2014.
[42] S. Liu, X. Chen, W. Liu, J. Chen, Q. Gu, and D. Chen, "FECAR: A feature selection framework for software defect prediction," in *Proc. IEEE 38th Annu. Comput. Softw. Appl. Conf.*, Jul. 2014, pp. 426–435.
[43] Q. Yu, S.-J. Jiang, R.-C. Wang, and H.-Y. Wang, "A feature selection approach based on a similarity measure for software defect prediction," *Frontiers Inf. Technol. Electron. Eng.*, vol. 18, no. 11, pp. 1744–1753, 2017.
[44] R. Vashisht and S. A. M. Rizvi, "Feature extraction to heterogeneous cross project defect prediction," in *Proc. 8th Int. Conf. Rel., Infocom Technol. Optim.*, Jun. 2020, pp. 1221–1225.
[45] H. Alsawalqah, N. Hijazi, M. Eshtay, H. Faris, A. A. Radaideh, I. Aljarah, and Y. Alshamaileh, "Software defect prediction using heterogeneous ensemble classification based on segmented patterns," *Appl. Sci.*, vol. 10, no. 5, p. 1745, Mar. 2020.
[46] K. O. Elish and M. O. Elish, "Predicting defect-prone software modules using support vector machines," *J. Syst. Softw.*, vol. 81, no. 5, pp. 649–660, 2008.
[47] Y. Zhang, D. Lo, X. Xia, and J. Sun, "Combined classifier for cross-project defect prediction: An extended empirical study," *Frontiers Comput. Sci.*, vol. 12, no. 2, pp. 280–296, Apr. 2018.

YAHAYA ZAKARIYAU BALA received the B.Sc. and M.Sc. degrees in computer science from Adamawa State University, Mubi, Nigeria, in 2008 and 2014, respectively. He is currently pursuing the Ph.D. degree in software engineering with Universiti Putra Malaysia (UPM). He is also a Lecturer with the Department of Computer Science, Faculty of Science, Federal University of Kashere, Nigeria. His research interests include software defect prediction and cross-project software defect prediction.



PATHIAH ABDUL SAMAT received the B.Sc. and M.Sc. degrees in computer science from Universiti Teknologi Malaysia (UTM), in 1996 and 1998, respectively, and the Ph.D. degree in computer science from Universiti Kebangsaan Malaysia (UKM), in 2012. She is currently a Senior Lecturer with the Department of Software Engineering and Information System, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia (UPM). Her research interests include formal software verification and model checking.

KHAIRONI YATIM SHARIF received the Ph.D. degree from the University of Limerick, Ireland. He is currently a Senior Lecturer with the Department of Software Engineering and Information System, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia. He is also an Adjunct Associate Professor with the Shibaura Institute of Technology, Japan. His research interest includes programmers' information need, particularly identifying programmers' information needs with regards to their task contexts, such as software maintenance, program comprehension, code concept mapping, fault localization, and agile development.

NORIDAYU MANSHOR received the B.Sc. degree in computer science from Universiti Putra Malaysia (UPM), the M.Sc. degree in computer science from Universiti Teknologi Malaysia (UTM), and the Ph.D. degree in computer science from Universiti Sains Malaysia (USM). She is currently a Senior Lecturer with the Department of Computer System, Faculty of Computer Science and Information Technology, UPM. Her research interests include pattern recognition, image processing, and computer vision.