IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 14 November 2023, accepted 2 December 2023, date of publication 14 December 2023,
date of current version 3 January 2024.

Digital Object Identifier 10.1109/ACCESS.2023.3343329

RESEARCH ARTICLE

Heterogeneous Cross-Project Defect Prediction Using Encoder Networks and Transfer Learning

RADOWANUL HAQUE[1], AFTAB ALI[1], SALLY MCCLEAN[1], (Member, IEEE), IAN CLELAND[1], AND JOOST NOPPEN[2]
[1]School of Computing, Ulster University, BT15 1ED Belfast, U.K.
[2]Applied Research, BT, IP5 3RE Ipswich, U.K.

Corresponding author: Radowanul Haque (haque-r1@ulster.ac.uk)

This work was supported in part by the BT Ireland Innovation Centre (BTIIC) Project, and in part by the BT and Invest Northern Ireland.

ABSTRACT Heterogeneous cross-project defect prediction (HCPDP) aims to predict defects in new software projects using defect data from previous software projects where the source and target projects have some different metrics. Most existing methods only find linear relationships in the software defect datasets. Additionally, these methods use multiple defect datasets from different projects as source datasets. In this paper, we propose a novel method called heterogeneous cross-project defect prediction using encoder networks and transfer learning (ENTL). ENTL uses encoder networks to extract the important features from source and target datasets. Also, to minimize the negative transfer during transfer learning, we used an augmented dataset that contains pseudo-labels and the source dataset. Additionally, we have used a single dataset to train the model. To evaluate the performance of the ENTL approach, 16 datasets from four publicly available software defect projects were used. Furthermore, we compared the proposed method with four HCPDP methods namely EGW, HDP_KS, CTKCCA and EMKCA, and one WPDP method from existing literature. The proposed method on average outperforms the baseline methods in terms of PD, PF, F1-score, G-mean and AUC.

INDEX TERMS Software defect, software engineering, transfer learning.

I. INTRODUCTION

The importance of software quality assurance has significantly increased within the software development industry. Defect prediction, which involves the identification of probable bugs in software systems, is a crucial measure for mitigating issues and minimising the need for expensive revisions [1]. Software defect prediction has the capability to reliably forecast early-stage software defects. The primary purpose of this method is to predict the likelihood of bugs occurring across software, making it easier for programmers and researchers to identify and correct them [2]. The software defect prediction model makes use of historical data from previous software projects by dividing it into training and testing datasets. The model trains on a training dataset and evaluates the performance using testing data, which is referred to as within-project defect prediction (WPDP). Defects are predicted by WPDP models using data collected from the same project [3]. There have been a number of studies that have suggested various strategies for enhancing the ability to predict the results of WPDP models. One such method is hyperparameter optimisation for an enhanced convolutional neural network (CNN) model for WPDP [4]. Another study investigated the effects of optimising WPDP hyperparameters [5]. One effective approach for reliably predicting software system bugs is the use of logistic regression and ensemble-bagged tree-based prediction models [6]. In another study, a cost-sensitive strategy based on discriminating features has been proposed for software bug prediction [7].

However, in practice, new software projects do not have sufficient historical data for model training. To solve this problem researchers have proposed cross-project defect prediction (CPDP). This approach utilises historical data to create a predictive model that eliminates the requirement for a large amount of previous data to anticipate the result of the project. A number of approaches have been proposed by researchers to enhance the efficiency of CPDP. A range

The associate editor coordinating the review of this manuscript and approving it for publication was Porfirio Tramontana.

of models and strategies have been proposed to address the disparity between the source and target datasets. The aforementioned techniques include domain adaptation learning, as proposed by Jin et al. [2], which aims to mitigate the dissimilarity between the source and target data distributions, A novel method called Transfer Naïve Bayes (TNB) selects significant features and transfers the weights into the training data, as proposed by Ma et al. [8].

However, in the context of actual application, it may not always be viable to utilise uniform metrics for all software data. The software domain and development languages exhibit variations across different projects. In this particular context, it is evident that the conventional CPDP paradigm is ineffective due to the absence of shared common software metrics between the source and target datasets. Researchers have therefore proposed heterogeneous cross-project defect prediction (HCPDP) methods for predicting software defects when the source and target datasets have distinct metrics. The Heterogeneous Cross-Project Defect Prediction framework develops a predictive model by using heterogeneous source and target projects. One study is based on transfer learning, which is the ability to acquire knowledge from one domain and apply it to another. The algorithm is designed to generate a projective matrix that aligns the distributions of heterogeneous source and target projects, enhancing their similarity [9]. Another research paper introduced a defect prediction approach that incorporates multi-source transfer learning and an encoder. This technique aims to construct a defect prediction model for a specific project by using information acquired from many source projects that include distinct metrics [10].

This paper presents an innovative approach to solve the gap between source and target datasets by encoder and transfer learning (ENTL). The proposed approach utilises encoders as a method to effectively extract important features from two separate datasets, referred to as the source and target datasets. Following this, the predictions produced by the initial neural network model trained on the features and labels of the source dataset are effectively used as pseudo-labels for the target dataset. This process contributes to the formation of an augmented dataset. In the following stage, our method utilises a secondary neural network model that is specifically designed for classification problems. This model is equipped with cost-sensitive learning processes to handle the difficulties that arise from class imbalances. The secondary model has been trained using an augmented dataset that combines both labelled data from the source dataset and pseudo-labelled data. The effectiveness of this technique is highlighted by a thorough assessment that includes common evaluation metrics such as PD, PF, F1-score, G-mean and AUC.

The rest of the paper is structured as follows: Section II provides a comprehensive survey of related work in the field to provide context for the study. In Section III, the methodology is described in detail, including the approach and techniques used. Section IV presents the experimental setup. Section V results of the model and compares them to baselines. The results are thoroughly discussed in Section VI, which also highlights the limitations of the proposed approach and provides insightful recommendations for future research. In the conclusion, Section VII summarises the findings and potentially suggests areas for future research. Finally, Section 8 provides a list of references to the sources that we used to collect information and support our research.

II. RELATED WORKS

Researchers have proposed several approaches in the CPDP and HCPDP domains. They are introduced in detail in this section.

A. CROSS-PROJECT DEFECT PREDICTION

Briand et al. [11] carried out the first study on CPDP. The researchers conducted an investigation on the transferability of fault-proneness models across different software projects. The researchers gathered defect data from two software projects, which were subsequently utilised to construct and assess models predicting the likelihood of defects. The researchers discovered that the transferability of fault-proneness models between projects is limited by variations in project size and complexity, disparities in the development process, and discrepancies in data quality. The study conducted by Zimmermann et al. [12] investigates the complexities and efficacy of implementing defect prediction models in various software projects. The authors of this comprehensive study examine three crucial elements, namely data, domain, and process, in order to assess their impact on the efficacy of cross-project defect prediction. The authors assess a range of machine learning approaches and statistical models in order to make predictions about defects in software projects across diverse domains and employ distinct development procedures.

Researchers have investigated the domain similarity technique to improve the performance of CPDP. Cohesion metrics such as the lack of method cohesion, coupling metrics including weighted methods per class, and complexity metrics including coupling between objects were studied by Zhang et al. [13]. They presented that domain similarity metrics improve CPDP performance when features are more similar in source and target datasets [13]. Krishna and Menzies [14] employed various product and process metrics in addition to lines of code, McCabe's cyclomatic complexity, number of prior defects, and code churn. They investigated CPDP-guiding bellwether strategies [14].

In recent years, transfer learning has been proposed by researchers to improve the performance of CPDP. Nam et al. [15] extracted common features from source and target projects using latent Dirichlet allocation. The features were used for transfer learning with naive Bayes and logistic regression models. Peters et al. [16] proposed a software module subset selection method called the Peter-Filter, which works by combining clustering and feature selection. Their method has been shown to work well for CPDP. Ma et al. [8] extracted common complexity metrics and applied a deep transfer learning model called DbNet. It uses stacked denoising autoencoders to learn feature representations and a binary classifier for prediction [8].

However, all these studies assume that the source and target datasets have common features. When there are different features in the source and target datasets, these methods fail [17].

B. HETEROGENEOUS CROSS-PROJECT DEFECT PREDICTION

Heterogeneous cross-project defect prediction refers to the extension of the CPDP concept to scenarios when the source and target projects represent significant differences in programming languages, development processes, or other relevant characteristics [9]. Several methods have been proposed by researchers to improve the performance of HCPDP.

Zong et al. [18] propose a novel method for HCPDP based on optimal transport. The proposed method functions by first learning mapping from the target project features and source project features. This mapping is subsequently utilised to determine the optimal transport distance between the two distributions. This distance is then utilised to weight the predictions of external initiatives. The proposed approach was evaluated on numerous projects, and the results revealed that it performed better for HCPDP than other approaches in the literature.

Li et al. [19] propose a novel method for heterogeneous defect prediction by using cost-sensitive transfer kernel canonical correlation analysis (CTKCCA). By including the defect data from the source project, their research aims to overcome the issue of limited defect data in the target project. The suggested technique makes use of CTKCCA to enhance prediction performance and understand the relation between the source and target projects. By using the defect data from the source project, the suggested technique can increase the accuracy of HCPDP. The suggested strategy for predicting HCPDP is thus better than other baseline approaches.

Nam and Kim [20] present a novel approach for addressing heterogeneous defect prediction (HDP) by effectively aligning diverse metrics across several projects. The paper aims to tackle the issue of limited defect data in the target project by utilising the defect data obtained from the source project. The approaches suggested in this study employ machine learning techniques to construct, validate, and enhance bug prediction models by using a collection of metrics obtained from software projects. The study conducts a comparative analysis between the suggested techniques and several baselines, including random forest, logistic regression, and Bayesian network. The findings indicate that the proposed methods exhibit superior performance in terms of accuracy, F1-score, and AUC when compared to the baselines.

Li et al. [17] proposed ensemble learning and multiple-kernel learning. Multiple kernel learning is a kernel-based learning technique that improves the separability of historical defect data by mapping it into a high-dimensional feature space. whereas ensemble learning integrates numerous models in order to enhance the precision and accuracy of the predictive model. In order to tackle the challenges associated with imbalanced and linearly inseparable classification in HDP, an innovative method known as Ensemble Multiple Kernel Correlation Alignment (EMKCA) has been proposed. Ensemble learning is utilised in conjunction with multiple kernel classifiers to enhance the performance of the defect prediction model in EMKCA. By utilising multiple-kernel domain adaptation learning, EMKCA is able to optimise the utilisation of the source and target data information. A multitude of experiments conducted on 30 public datasets have demonstrated that EMKCA exhibits superior performance compared to other methods in the existing literature.

Our proposed approach is different from the current methodologies. We have used a single dataset for training. Also, we have used encoder networks to extract the informative feature representations from the source and target datasets. The transfer learning technique in this approach reuses the source model for generating predictions referred to as pseudo-labels on the target dataset rather than directly transferring model parameters. As the structure of the source data does not overly constrain the model, this approach increases adaptability and lowers the risk of negative transfer. Furthermore, the model undergoes a fine-tuning phase on the augmented dataset, which contains both the source dataset and target dataset features with pseudo-labels. This fine-tuning process facilitates the adjustment of the model to the target dataset.

III. THE PROPOSED ENTL APPROACH

The initial part of this section introduces the framework of the ENTL approach, followed by data preprocessing procedures. Following this, the process of transfer learning is discussed in this section.

A. FRAMEWORK

The framework of ENTL is depicted in Figure 1. The inputs of the model consist of a source dataset and a target dataset. Both datasets went through preprocessing procedures. After the preprocessing step, the encoder is used to extract important features from both the source and the target dataset. Following this, an initial predictive model was constructed using the labels and the features extracted from the source dataset by the encoder network. Subsequently, the predicted labels generated by the first model are combined with the source dataset. that, the final model was trained with the augmented dataset. Finally, the performance of the model is evaluated by comparing it to the target dataset labels.

B. PREPROCESSING

During the initial phase of data preparation, we imported two different datasets, one serving as the source and the other as the target. Categorical label columns that included values such as ‘Y’ or ‘N,’ ‘Yes’ or ‘No,’ ‘True’ or ‘False,’

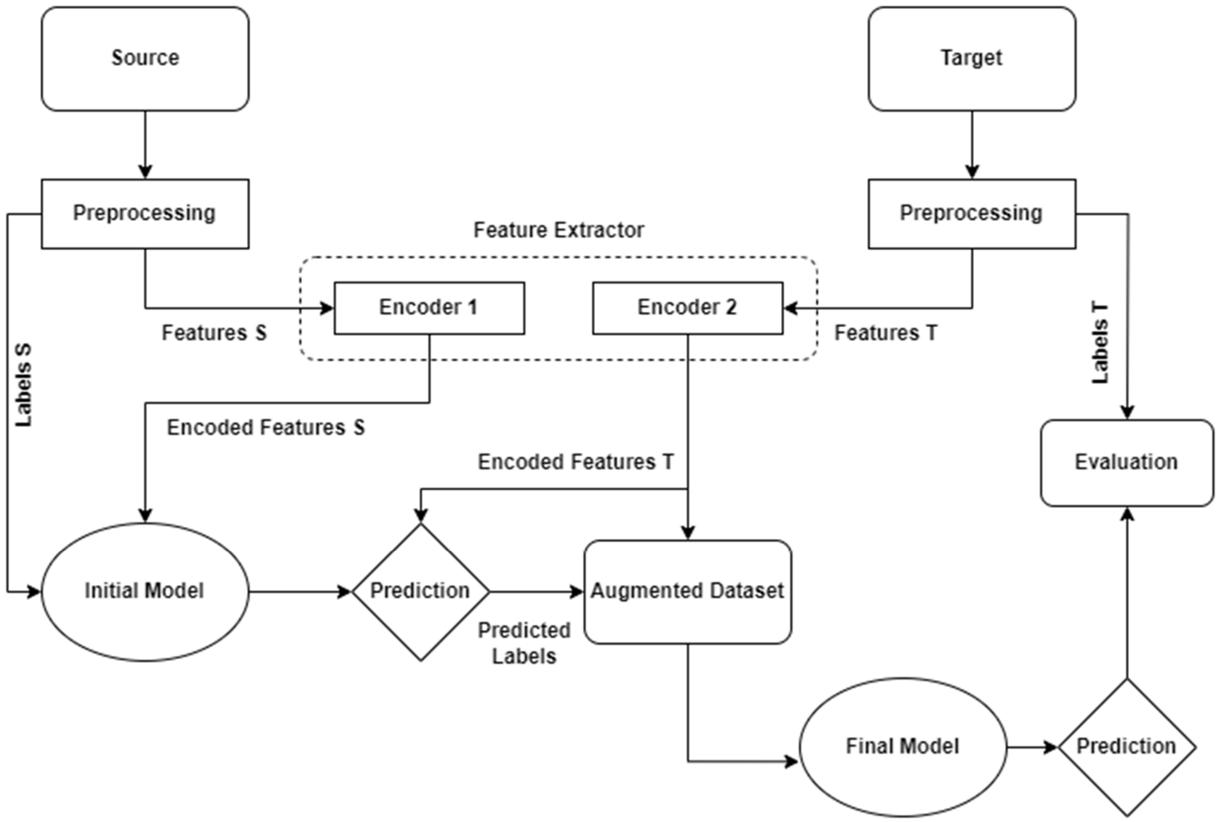

FIGURE 1. The framework of the proposed approach (ENTL).

and 'Buggy' or 'Clean' were converted into a binary representation.

Subsequently, we applied Z-score normalisation to standardise the numerical features in both datasets, ensuring they shared a common scale. We then separated the features and labels from both datasets to simplify model training and testing, establishing a solid foundation for subsequent model development and testing.

C. FEATURE EXTRACTION

Feature extraction is performed by employing two distinct encoder models for both the source and target datasets. Using two distinct encoder models allows the encoder networks to independently learn tailored representations for two different datasets. This is advantageous when the datasets exhibit diverse patterns, ensuring that the learned features capture the unique characteristics of each dataset. These encoder models transform the input feature data into lower-dimensional representations [21]. It takes an input data vector x with n features, represented as $[x_1 + x_2 + \ldots.. + x_n]$, and employs weight matrices W bias terms b, and an activation function f [22]. The process involves a linear transformation, where for each neuron i in the hidden layer, the encoder computes a weighted sum of input features $z_i = W_i * x + b_i$ where W_i represents the weight matrix for neuron i and b_i is the bias term. Subsequently, an activation function f introduces non-linearity, leading to $h_i = f(z_i)$, producing the output values h_i for each neuron. These h_i values collectively form the encoded representation $h = [h_1, h_2, \ldots..h_m]$ where m is the number of neurons in the hidden layer. The feature vector h captures vital patterns and features from the input data. It is important to note that these encoder models are precisely built with the same designs for the source and target datasets. The encoder models ensure the uniformity of feature vector dimensionality by employing a consistent design [23]. The alignment between the source and target datasets is crucial for the future training of the model, as it facilitates the seamless transfer of data from the source dataset to the target dataset. These feature representations are easier to add to later models because of the common dimension. This facilitates the exchange of information between datasets, thereby enhancing the accuracy of classification.

D. TRANSFER LEARNING

Transfer learning is an effective approach that leverages the information gained from a source dataset to improve the performance of a different target dataset [8]. This method is especially useful in cases where there is no labelled data available in the target dataset.

The proposed approach involves the systematic process of transfer learning. Initially, a neural network model is trained using the source dataset. Throughout this training session, the source model is equipped with class weights, which serve as an effective strategy for efficiently handling class imbalances [24]. Let X_s and Y_s be the source dataset features and labels and X_T be the target dataset features. H_s is the encoded source dataset features and H_T is the encoded target dataset features. The initial model M_s is trained on H_s and Y_s to minimize binary cross-entropy loss [25]. The equation is as follows:

$$L_s = -\Sigma(w_0\,(1 - Y_s)\,log(1 - M_s\,(H_s)) + w_1 {*} Y_s log(M_s\,(H_s))) \quad (1)$$

The class weights w_0 and w_1 are used to assign different levels of importance to handle class imbalance.

Once the initial model M_s has been trained, it is used to make predictions on the target dataset. The earlier predictions, often referred to as pseudo-labels, $\hat{Y}_T = M_s(H_T)$. In the final phase, the model undergoes retraining using the augmented dataset (H_A, Y_A), where $H_A = [H_s, H_T]$ and $Y_A = [Y_s, \hat{Y}_{Tbin}]$. The threshold is set to 0.5 to convert $\hat{Y}_T$ to binary. The loss function equation is as follows:

$$L_s = -\Sigma(w_0\,(1 - Y_A)\,log\,(1 - M_T\,(H_A)) + w_1 {*} Y_A log\,(M_T\,(H_A))) \quad (2)$$

The process of retraining enables the model to refine its existing knowledge and enhance its performance in the target domain. By iteratively refining its understanding of both datasets, the model excels at addressing classification challenges and uncovering patterns that might have remained elusive with the initial model.

After the secondary model training, an ensemble approach is used. The ensemble prediction combines the predictions from both M_T and the individual XGBoost models (denoted as X_i, where i represents each model in the ensemble). The XGBoost model is trained using the H_A and Y_A. The ensemble prediction, denoted as E, is obtained by averaging these predictions. Mathematically, it can be expressed as follows:

$$E = \frac{s + \sum_{i=1}^{n} X_i}{n} \quad (3)$$

where S represents the predictions from the M_T and X_i represents the predictions from each XGBoost model in the ensemble, and n is the total number of XGBoost models in the ensemble. Individual XGBoost models are employed to harness diverse perspectives and enhance adaptability within the ensemble. The use of individual models allows for iterative learning, enabling the ensemble to capture nuanced patterns and adapt to the intricacies of the dataset. This ensemble prediction approach is based on averaging the outputs from different models, incorporating both the knowledge learned from the M_T model and the domain adaptation capabilities of the individual XGBoost models. The ensemble prediction aims to provide a more robust and accurate prediction for the target dataset by aggregating the insights from multiple models.

Algorithm 1 The ENTL Approach for HCPDP

Input: X_s- source features, Y_s – source labels and X_T- target features
Output: Y_T – target labels

1. Use encoder to extract H_s from X_s and H_T from X_T
2. Assign class weight w to handle class imbalance
3. Initialize an empty list M to store XGBoost models.
4. For n iterations
   - 4.1. Train a neural network model M_S on (H_s, Y_S) with w
   - 4.2. Use M_S to predict $\hat{Y}_T$
   - 4.3. Make an augmented dataset (H_A, Y_A)
   - 4.4. Train M_T on (H_A, Y_A)
   - 4.5. Train XGBoost model XG on (H_A, Y_A)
   - 4.6. Add XG to the list M
5. Initialise an empty list P
6. For each model XG in M:
   - 6.1. Use XG to predict Y_T on H_T
   - 6.2. add Y_T to list P
7. Calculate the ensemble prediction E from Eq.3 by averaging the predictions in P
8. Return E as Y_T

IV. EXPERIMENTAL SETUP

In this section experimental questions, dataset descriptions, evaluation measures, and evaluation settings are explained in detail.

A. EXPERIMENTAL QUESTIONS

To investigate the performance of the proposed approach, we have designed two research questions.

RQ1: Does ENTL outperform other HCPDP methods when using a single dataset for training?

RQ2: Does ENTL outperform the WPDP method?

B. DATASETS

In this study, we have used 16 publicly available datasets from 4 different projects namely AEEM [26], NASA [27], Promise [28] and JIRA.

The name of the project, the number of entries, the number of bugs, the number of metrics, and the percentage of bugs in the dataset are presented in Table 1. The number of metrics is different in all projects. It is important to note that, the baseline methods may outperform some specific datasets but perform worse on other datasets. To ensure a fair comparison, we have used the datasets that are commonly used in all the baseline methods.

C. EVALUATION MEASURES

To ensure a fair comparison, five evaluation metrics that were common in the baseline methodologies were chosen. The metrics for evaluation are PD, PF, F1-score, G-mean, and AUC. Since HCPDP is a binary task, TP (True Positives) is the number of correctly identified positive cases, FN (False Negatives) is the number of actual positive cases that were incorrectly classified as negative, TN (True Negatives) is the number of correctly identified negative cases, and FP (False Positives) is the number of actual negative cases that were

TABLE 1. Experimental dataset descriptions.

Projects	Dataset	Number of Entries	Number of Bugs	Number of Metrics	Bugs%
AEEM	EQ	324	129	61	39.8
	JDT	997	206	61	20.7
	LC	691	64	61	9.3
	ML	1862	245	61	13.2
JIRA	activemq5.0.	1884	293	65	15.6
	Derby10.5.1.1	2705	383	65	14.2
	Hbase0.94.0	1059	218	65	2.6
	Hive0.9.0	1416	283	65	20.0
NASA	KC1	2095	325	21	15.5
	PC1	735	61	37	8.3
	PC3	1099	138	37	12.6
	PC4	1379	178	37	12.9
PROMISE	Lucene2.4	340	203	20	59.7
	Poi3.0	442	281	20	63.6
	Synapse1.2	256	86	20	33.6
	Velocity1.6	229	78	20	34.1

incorrectly classified as positive. The evaluation metrics are explained below:

PD: The term "PD" refers to the metric known as "positive detection," which quantifies the percentage of positive instances accurately recognised as positive by the model [26].

$$PD = \frac{TP}{TP + FN} \tag{4}$$

PF: The term "PF" refers to the percentage of negative examples that the model correctly identified as positive [26].

$$PF = \frac{FP}{FP + TN} \tag{5}$$

F1- score: The F1 Score is a metric that quantifies the harmonic mean of precision and recall, offering a balanced assessment of both precision and recall [29].

$$F1 - score = \frac{2(Precision * Recall)}{Precision + Recall} \tag{6}$$

where,

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

G-mean: The G-mean is a metric that provides a balanced evaluation of both sensitivity (recall) and specificity (true negative rate) [31].

$$G - mean = \sqrt{(Sensitivity * Specificity)} \tag{9}$$

where,

$$Sensitivity = \frac{TP}{TP + FN} \tag{10}$$

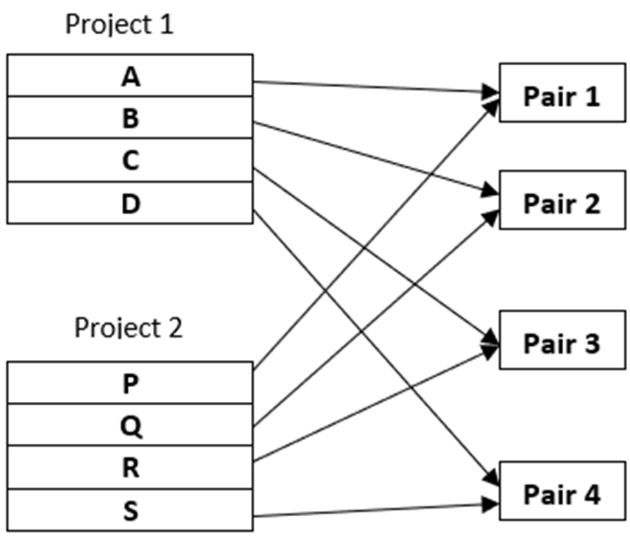

FIGURE 2. Experimental settings.

$$Specificity = \frac{TN}{TN + FP} \tag{11}$$

AUC: Area Under the Curve (AUC) is a metric used to evaluate the efficacy of binary classification models, specifically in ROC curve analysis. It measures the capacity of a model to differentiate between positive and negative classes [31].

D. EVALUATION SETTINGS

We have assessed the performance of our model using 16 datasets from 4 different projects. We chose one dataset from a particular project as the source dataset and another dataset from a different project as the target dataset. For example, in the AEEM project, we used EQ as the target dataset, and in the JIRA project, we selected activemaq-5.0.0 as the source dataset. We repeated this process for all the datasets, making sure the source and target datasets came from different projects, as illustrated in Figure 2.

After the training phase, the labels predicted by the model were compared with the actual label of the target dataset. The averaging ensemble approach with an iteration size of 20 is chosen to mitigate the impact of individual model variability and enhance the reliability of predictions. By aggregating predictions over multiple iterations, the ensemble model smooths the potential randomness or biases inherent in individual XGBoost models. This averaging process helps to create a more stable and robust final prediction by reducing the influence of outliers in the individual models.

V. EXPERIMENTAL RESULTS

A. ANSWER TO RQ1: THE PERFORMANCE OF ENTL COMPARED TO OTHER HCPDP METHODS

1) METHODS

There are many HCPDP models proposed by the researchers. For the comparison, we have selected some models such as EGW [18], HDP_KS [20], CTKCCA [19], and EMKCA [17], which have been frequently used for comparison in the literature. EGW applied optimal transport theory, HDP_KS effectively aligned diverse metrics across several projects, CTKCCA utilized transfer kernel canonical correlation analysis to minimize the gap between source and target dataset, and EMKCA used multiple kernel-based learning techniques that improves the separability of historical defect data by mapping it into a high-dimensional feature space. They have tried to minimise the gap between the source and the target datasets in different ways. It is also worth noting that the commonality of datasets and evaluation metrics are also reasons for selecting these methods.

2) RESULTS

Tables 2-6 represent the PD, PF, F1-score, G-mean, and AUC values of ENTL compared to baseline methods. The results are represented as the average and standard deviation of each dataset in a project. The last line depicts the average values across all the projects and the best result is in bold font.

A high PD value and a lower PF reflect the good performance of a model. It can be seen in Tables 2–3 that the proposed ENTL method achieved the highest average PD value and also achieved the highest average PF value. On the contrary, EMKCA had the lowest average PF value, but the PD was also the lowest, which is less than 0.1. However, evaluating all the datasets as a whole, the performance of the ENTL methods on PD and PF was better than that of other baseline methods. Tables 3–4 depict the performance of ENTL in terms of F1-score and G-mean. It can be seen that ENTL achieved the highest average F1-score and G-mean compared to the baseline methods. The difference in f1-score between ENTL and baseline methods such as EGW and HDP_KS is not very high; however, in terms of G-mean, the difference is very high compared to other baselines. In terms of AUC value, ENTL also performs better than the baseline methods. Overall, ENTL performs better in terms of all evaluation metrics, as illustrated in Figure 3.

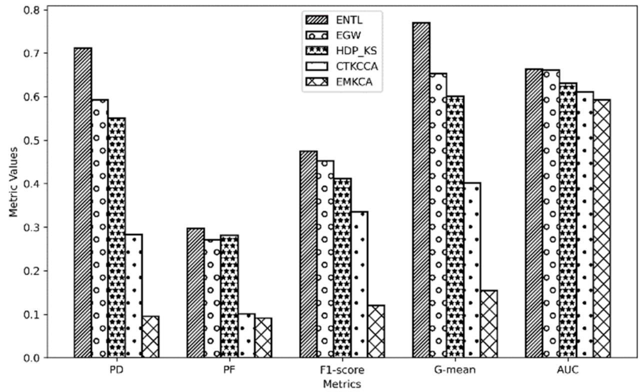

FIGURE 3. The performance of ENTL compared to other HCPDP methods.

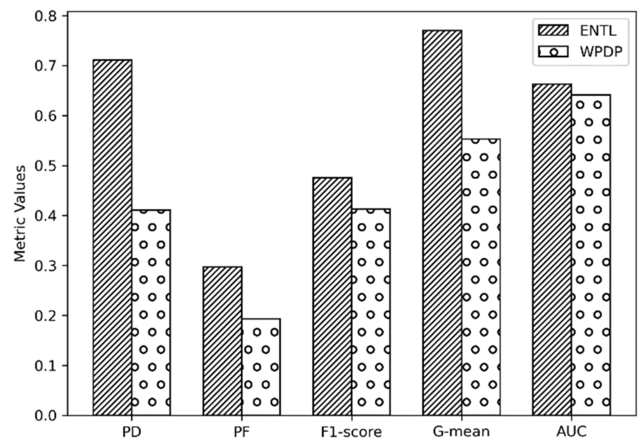

FIGURE 4. The performance of ENTL compared to WPDP method.

B. ANSWER TO RQ2: THE PERFORMANCE OF ENTL COMPARED TO WPDP

1) METHODS

Researchers have proposed a variety of WPDP techniques. In this study, we have selected one method presented by Catal et al. [33]. The reason for selecting this method is that it used 10% of the data for building the predictive model and 90% for testing the model. Which ensures a fair comparison with the HCPDP method.

2) RESULTS

It can be seen from Tables 2–6 that ENTL performs better compared to WPDP in terms of PD, F1-score, G-mean, and AUC. The performance of ENTL is improved in terms of PD, F1-score, G-mean and AUC by 72.99%, 15.01%,

TABLE 2. PD value comparison between ENTL and baselines. Results are in the form of mean ± standard deviation.

Target	ENTL	EGW	HDP_KS	CTKCCA	EMKCA	WPDP
AEEM	**0.751 ± 0.051**	0.598 ± 0.074	0.488 ± 0.038	0.271 ± 0.094	0.080 ± 0.041	0.375 ± 0.078
JIRA	**0.675 ± 0.080**	0.662 ± 0.047	0.515 ± 0.021	0.182 ± 0.029	0.028 ± 0.013	0.395 ± 0.062
NASA	**0.751 ± 0.054**	0.612 ± 0.038	0.581 ± 0.048	0.298 ± 0.279	0.105 ± 0.063	0.272 ± 0.049
PROMISE	**0.663 ± 0.031**	0.505 ± 0.074	0.416 ± 0.041	0.455 ± 0.114	0.151 ± 0.052	0.590 ± 0.097
Average	**0.711**	0.593	0.551	0.283	0.095	0.411

TABLE 3. PF value comparison between ENTL and baselines. Results are in the form of mean ± standard deviation.

Target	ENTL	EGW	HDP_KS	CTKCCA	EMKCA	WPDP
AEEM	0.278 ± 0.039	0.265 ± 0.021	0.251 ± 0.052	0.115 ± 0.097	**0.105 ± 0.124**	0.169 ± 0.060
JIRA	0.325 ± 0.081	0.278 ± 0.013	0.296 ± 0.011	**0.022 ± 0.011**	0.025 ± 0.011	0.105 ± 0.043
NASA	0.250 ± 0.054	0.281 ± 0.025	0.284 ± 0.026	0.121 ± 0.064	**0.037 ± 0.018**	0.061 ± 0.016
PROMISE	0.338 ± 0.031	0.255 ± 0.036	0.282 ± 0.021	0.151 ± 0.124	**0.142 ± 0.036**	0.412 ± 0.054
Average	0.297	0.271	0.282	0.101	**0.091**	0.193

TABLE 4. F1-score comparison between ENTL and baselines. Results are in the form of mean ± standard deviation.

Target	ENTL	EGW	HDP_KS	CTKCCA	EMKCA	WPDP
AEEM	0.325 ± 0.0156	**0.430 ± 0.121**	0.372 ± 0.116	0.311 ± 0.085	0.111 ± 0.044	0.353 ± 0.111
JIRA	0.449 ± 0.058	**0.457 ± 0.025**	0.363 ± 0.024	0.282 ± 0.035	0.027 ± 0.010	0.421 ± 0.037
NASA	**0.470 ± 0.194**	0.337 ± 0.075	0.322 ± 0.068	0.191 ± 0.076	0.145 ± 0.068	0.311 ± 0.067
PROMISE	**0.607 ± 0.078**	0.548 ± 0.031	0.463 ± 0.044	0.525 ±0.075	0.215 ± 0.481	0.586 ± 0.123
Average	**0.475**	0.453	0.412	0.335	0.12	0.413

TABLE 5. G-mean value comparison between ENTL and baselines. Results are in the form of mean ± standard deviation.

Target	ENTL	EGW	HDP_KS	CTKCCA	EMKCA	WPDP
AEEM	**0.712 ± 0.037**	0.657 ± 0.044	0.579 ± 0.031	0.400 ± 0.097	0.145 ± 0.063	0.547 ± 0.048
JIRA	**0.780 ± 0.073**	0.691 ± 0.022	0.583 ± 0.019	0.305 ± 0.041	0.032 ± 0.004	0.592 ± 0.043
NASA	**0.831 ± 0.121**	0.662 ± 0.031	0.633 ± 0.035	0.317 ± 0.141	0.185 ± 0.102	0.500 ± 0.043
PROMISE	**0.758 ± 0.098**	0.600 ± 0.391	0.507 ± 0.030	0.571 ± 0.069	0.249 ± 0.073	0.581 ± 0.037
Average	**0.770**	0.653	0.601	0.402	0.154	0.553

39.24 %, and 3.43% respectively, as illustrated in Figure 3. The WPDP method achieved a lower average PF value than ENTL. However, the PD value was also considerably lower than the ENTL method. Overall, the proposed ENTL method on average outperforms the WPDP method in terms of all evaluation metrics.

VI. DISCUSSION

The objective of this research is to predict defects in software projects using previous software defect datasets. The new software project can have different software metrics than the source dataset. To solve this data heterogeneity, we have proposed a novel method (ENTL) in this research. The datasets

TABLE 6. AUC value comparison between ENTL and baselines. Results are in the form of mean ± standard deviation.

Target	ENTL	EGW	HDP_KS	CTKCCA	EMKCA	WPDP
AEEM	0.622 ± 0.029	**0.665 ± 0.038**	0.656 ± 0.047	0.542 ± 0.035	0.515 ± 0.044	0.573 ± 0.049
JIRA	0.657 ± 0.051	**0.690 ± 0.022**	0.642 ± 0.012	0.581 ± 0.012	0.551 ± 0.012	0.685 ± 0.048
NASA	**0.727 ± 0.088**	0.665 ± 0.029	0.698 ± 0.035	0.497 ± 0.113	0.625 ± 0.033	0.713 ± 0.069
PROMISE	**0.647 ± 0.043**	0.621 ± 0.025	0.606 ± 0.028	0.662 ± 0.101	0.525 ± 0.023	0.597 ± 0.047
Average	**0.663**	0.661	0.631	0.611	0.593	0.641

used in this research are publicly available. The proposed model has been evaluated by comparing the predicted label with the actual label of the target dataset. Also, the proposed method was compatible with existing methods proposed by researchers. ENTL performed better on average than the existing method in the literature.

Additionally, the existing methodologies described in the literature utilise several datasets collected from various projects to train their models. In contrast, our proposed approach utilises a single dataset as the primary source for training, while making predictions on a separate dataset. This technique also solves the issue of having a limited dataset to train the model. This technique introduces trade-offs and potential advantages. On one hand, utilising a single dataset limits available training data compared to leveraging diverse datasets across projects, potentially hindering model generalizability. However, it reduces heterogeneity and noise, allowing models to fit more closely to the specific project's characteristics and contributing to a potentially more interpretable model. While it does not integrate the knowledge from several projects, it demonstrates on average superior performance in terms of PD, PF, F1-score, G-mean, and AUC.

A. LIMITATIONS

In our study, we have used 16 datasets from four publicly available software defect projects to evaluate the performance of our proposed model. There are other benchmark software defect datasets used in CPDP and HCPDP, such as the Relink and SOFTLAB datasets. These datasets are left for future work. Furthermore, most of the work used for comparison does not provide code for their method; we have used the results that are available in their papers. Additionally, to handle the class imbalance in a more accurate way, we will use other methods, such as SMOTE, in future works.

VII. CONCLUSION

Recently HCPDP has gained much research interest. In heterogeneous cross-project scenarios, the training and testing datasets have different features. It can be applicable to find defects in new software that does not have any labelled data. In this paper, we have proposed a novel encoder networks and transfer learning (ENTL) approach to HCPDP. To reduce the negative transfer during transfer learning we have used an augmented dataset containing pseudo-labels. We have used 16 datasets from 4 different projects to evaluate the proposed approach. We have used a wide range of evaluation metrics such as PD, PD, F1-score, G-mean and AUC. While working on this project, we have also found that the datasets are imbalanced. Without treating the imbalanced dataset, the model overfits with the majority class and struggles to predict the minority class properly. To handle this class imbalance problem, we used cost-sensitive learning. The performance of the model is compared with four HCPDP methods and one WPDP method from existing literature. On average it performs better than the baselines.

For future works, we will use more software defect datasets and HCPDP baselines to verify the proposed approach. Additionally, we will use other class imbalance handling methods in future to improve the method performance.

REFERENCES

[1] N. E. Fenton and M. Neil, "A critique of software defect prediction models," *IEEE Trans. Softw. Eng.*, vol. 25, no. 5, pp. 675–689, Sep. 1999, doi: 10.1109/32.815326.

[2] C. Jin, "Cross-project software defect prediction based on domain adaptation learning and optimization," *Expert Syst. Appl.*, vol. 171, Jun. 2021, Art. no. 114637, doi: 10.1016/j.eswa.2021.114637.

[3] F. Yang, Y. Huang, H. Xu, P. Xiao, and W. Zheng, "Fine-grained software defect prediction based on the method-call sequence," *Comput. Intell. Neurosci.*, vol. 2022, pp. 1–15, Aug. 2022, doi: 10.1155/2022/4311548.

[4] C. Pan, M. Lu, B. Xu, and H. Gao, "An improved CNN model for within-project software defect prediction," *Appl. Sci.*, vol. 9, no. 10, p. 2138, May 2019, doi: 10.3390/app9102138.

[5] M. M. Öztürk, "Comparing hyperparameter optimization in cross- and within-project defect prediction: A case study," *Arabian J. Sci. Eng.*, vol. 44, no. 4, pp. 3515–3530, Apr. 2019, doi: 10.1007/s13369-018-3564-9.

[6] A. Ali, M. Abu-Tair, J. Noppen, S. McClean, Z. Lin, and I. McChesney, "Contributing features-based schemes for software defect prediction," in *Artificial Intelligence XXXVI* (Lecture Notes in Computer Science). 2019, pp. 350–361, doi: 10.1007/978-3-030-34885-4_27.

[7] A. Ali, N. Khan, M. Abu-Tair, J. Noppen, S. McClean, and I. McChesney, "Discriminating features-based cost-sensitive approach for software defect prediction," *Automated Softw. Eng.*, vol. 28, no. 2, p. 11, Nov. 2021, doi: 10.1007/s10515-021-00289-8.

[8] 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, Mar. 2012, doi: 10.1016/j.infsof.2011.09.007.

[9] X. Yin, L. Liu, H. Liu, and Q. Wu, "Heterogeneous cross-project defect prediction with multiple source projects based on transfer learning," *Math. Biosci. Eng.*, vol. 17, no. 2, pp. 1020–1040, 2020, doi: 10.3934/mbe.2020054.

[10] K. Zhu, N. Zhang, S. Ying, and D. Zhu, "Within-project and cross-project just-in-time defect prediction based on denoising autoencoder and convolutional neural network," *IET Softw.*, vol. 14, no. 3, pp. 185–195, Jun. 2020, doi: 10.1049/iet-sen.2019.0278.

[11] 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, doi: 10.1109/TSE.2002.1019484.

[12] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, and B. Murphy, "Cross-project defect prediction," in *Proc. 7th Joint Meeting Eur. Softw. Eng. Conf. ACM SIGSOFT Symp. Found. Softw. Eng.* New York, NY, USA: ACM, Aug. 2009, pp. 91–100, doi: 10.1145/1595696.1595713.

[13] F. Zhang, Q. Zheng, Y. Zou, and A. E. Hassan, "Cross-project defect prediction using a connectivity-based unsupervised classifier," in *Proc. IEEE/ACM 38th Int. Conf. Softw. Eng. (ICSE)*. New York, NY, USA: ACM, May 2016, pp. 309–320, doi: 10.1145/2884781.2884839.

[14] R. Krishna and T. Menzies, "Bellwethers: A baseline method for transfer learning," *IEEE Trans. Softw. Eng.*, vol. 45, no. 11, pp. 1081–1105, Nov. 2019, doi: 10.1109/TSE.2018.2821670.

[15] 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, doi: 10.1109/ICSE.2013.6606584.

[16] F. Peters, T. Menzies, and A. Marcus, "Better cross company defect prediction," in *Proc. 10th Work. Conf. Mining Softw. Repositories (MSR)*, May 2013, pp. 409–418, doi: 10.1109/MSR.2013.6624057.

[17] Z. Li, X.-Y. Jing, X. Zhu, H. Zhang, B. Xu, and S. Ying, "Heterogeneous defect prediction with two-stage ensemble learning," *Automated Softw. Eng.*, vol. 26, no. 3, pp. 599–651, Sep. 2019, doi: 10.1007/s10515-019-00259-1.

[18] X. Zong, G. Li, S. Zheng, H. Zou, H. Yu, and S. Gao, "Heterogeneous cross-project defect prediction via optimal transport," *IEEE Access*, vol. 11, pp. 12015–12030, 2023, doi: 10.1109/ACCESS.2023.3241924.

[19] Z. Li, X.-Y. Jing, F. Wu, X. Zhu, B. Xu, and S. Ying, "Cost-sensitive transfer kernel canonical correlation analysis for heterogeneous defect prediction," *Automated Softw. Eng.*, vol. 25, no. 2, pp. 201–245, Jun. 2018, doi: 10.1007/s10515-017-0220-7.

[20] J. Nam and S. Kim, "Heterogeneous defect prediction," in *Proc. 10th Joint Meeting Found. Softw. Eng.*, 2015, pp. 508–519, doi: 10.1145/2786805.2786814.

[21] S. Chen and W. Guo, "Auto-encoders in deep learning—A review with new perspectives," *Mathematics*, vol. 11, no. 8, p. 1777, Apr. 2023, doi: 10.3390/math11081777.

[22] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006, doi: 10.1126/science.1127647.

[23] L. Li, Y. Fang, J. Wu, J. Wang, and Y. Ge, "Encoder–decoder full residual deep networks for robust regression and spatiotemporal estimation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 9, pp. 4217–4230, Sep. 2021, doi: 10.1109/TNNLS.2020.3017200.

[24] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study1," *Intell. Data Anal.*, vol. 6, no. 5, pp. 429–449, Nov. 2002, doi: 10.3233/IDA-2002-6504.

[25] M. Yeung, E. Sala, C.-B. Schönlieb, and L. Rundo, "Unified focal loss: Generalising dice and cross entropy-based losses to handle class imbalanced medical image segmentation," *Computerized Med. Imag. Graph.*, vol. 95, Jan. 2022, Art. no. 102026, doi: 10.1016/j.compmedimag.2021.102026.

[26] S. Yatish, J. Jiarpakdee, P. Thongtanunam, and C. Tantithamthavorn, "Mining software defects: Should we consider affected releases," in *Proc. IEEE/ACM 41st Int. Conf. Softw. Eng. (ICSE)*, May 2019, pp. 654–665, doi: 10.1109/ICSE.2019.00075.

[27] M. Shepperd, Q. Song, Z. Sun, and C. Mair, "Data quality: Some comments on the NASA software defect datasets," *IEEE Trans. Softw. Eng.*, vol. 39, no. 9, pp. 1208–1215, Sep. 2013, doi: 10.1109/TSE.2013.11.

[28] M. Jureczko and L. Madeyski, "Towards identifying software project clusters with regard to defect prediction," in *Proc. 6th Int. Conf. Predictive Models Softw. Eng.* New York, NY, USA: ACM, Sep. 2010, pp. 1–10, doi: 10.1145/1868328.1868342.

[29] X. Chen, Y. Mu, Y. Qu, C. Ni, M. Liu, T. He, and S. Liu, "Do different cross-project defect prediction methods identify the same defective modules?" *J. Softw., Evol. Process*, vol. 32, no. 5, May 2020, Art. no. e2234, doi: 10.1002/smr.2234.

[30] L. Chen, B. Fang, Z. Shang, and Y. Tang, "Negative samples reduction in cross-company software defects prediction," *Inf. Softw. Technol.*, vol. 62, pp. 67–77, Jun. 2015, doi: 10.1016/j.infsof.2015.01.014.

[31] D. Ryu, O. Choi, and J. Baik, "Value-cognitive boosting with a support vector machine for cross-project defect prediction," *Empirical Softw. Eng.*, vol. 21, no. 1, pp. 43–71, Feb. 2016, doi: 10.1007/s10664-014-9346-4.

[32] C. Tantithamthavorn, A. E. Hassan, and K. Matsumoto, "The impact of class rebalancing techniques on the performance and interpretation of defect prediction models," *IEEE Trans. Softw. Eng.*, vol. 46, no. 11, pp. 1200–1219, Nov. 2020, doi: 10.1109/TSE.2018.2876537.

[33] C. Catal and B. Diri, "A systematic review of software fault prediction studies," *Expert Syst. Appl.*, vol. 36, no. 4, pp. 7346–7354, May 2009, doi: 10.1016/j.eswa.2008.10.027.

RADOWANUL HAQUE received the B.S. degree in electronic and telecommunication engineering from International Islamic University Chittagong, Bangladesh, and the M.S. degree in computer science from Ulster University, U.K. His research interests include intelligent software engineering, machine learning, deep learning, and digital twins.

AFTAB ALI is currently a Lecturer with the School of Computing, Ulster University, Belfast, U.K. With over a decade of experience in academia, he was a Lecturer and a Researcher at various universities. His research work has resulted in the publication of scholarly articles in esteemed international journals and conferences. These publications primarily revolve around the utilization of artificial intelligence to improve software productivity and bolster cybersecurity measures. He is the lead academic on two projects, namely Software Bug Prediction and Future IoT Security, with the BT Ireland Innovation Centre (BTIIC). He also leads a team of graduate students, supervising both M.Sc. and Ph.D. students. He is a Co-Investigator with the BT Ireland Innovation Centre (BTIIC £6.2 million) and the PwC Advanced Engineering and Research Centre (ARC £3.13 million). His research interests include cybersecurity, trust management in the Internet of Things (IoT), blockchains, digital twins, artificial intelligence, and machine learning applications.

His expertise is sought after in the publishing domain, where he serves as a reviewer and a guest editor for several international journals.

SALLY MCCLEAN (Member, IEEE) received the first degree in mathematics from Oxford University, U.K., the M.Sc. degree in mathematical statistics and operational research from Cardiff University, and the Ph.D. degree in Markov and semi-Markov models from Ulster University.

She is currently a Professor of mathematics with Ulster University. Much of this work has been focused on healthcare and business applications, particularly concerning patient or customer modeling and pervasive technologies. More recently she has been with the Invest Northern Ireland (INI) funded Ireland Innovation Centre (BTIIC) mainly carrying out research into process modeling and mining. She was the UU Principal Investigator for several years. She has been a grant-holder on more than £15 million worth of funding, mainly from the EPSRC, Industry, the EU, and charities. She has published more than 500 research articles. Her main research interests include stochastic modeling and optimization, particularly for healthcare planning, and computer science, specifically databases, process mining, sensor technology, and telecommunications. She is a fellow of the Royal Statistical Society, a fellow of the Operational Research Society, a fellow of the Institute of Mathematics and its Applications, and the past President of the Irish Statistical Association. She was a recipient of the Ulster University's Senior Distinguished Research Fellowship.

IAN CLELAND received the B.Sc. degree in biomedical engineering and the Ph.D. degree in computer science from Ulster University, in 2009 and 2012, respectively. He is currently a Senior Lecturer with the School of Computing, Ulster University, where he leads the research theme of human–computer interaction within the Pervasive Computing Research Centre. His research combines wearable, pervasive, and mobile computing with data science and artificial intelligence to produce innovative digital solutions, mainly in the healthcare domain. He has secured externally funded research, as a Principal Investigator and a Co-Investigator, to the value of more than £7.89 million. Most notably he is a Co-Investigator on the Connected Health Innovation Center (CHIC-Phase 2 £3.36 million) and the PwC Advanced Engineering and Research Centre (ARC £3.13 million). His research interests include sensor-based activity recognition and the use of synthetic data to support data-driven modeling. He is the Track Chair of AmI for Health and [Ambient, Active and Assisted Living (A3L)], UCAmI2023, and the Vice Chair of the Executive Committee for the Alzheimer's Association Technology and Dementia Professional Interest Area.

JOOST NOPPEN received the M.Sc. and Ph.D. degrees in computer science from the University of Twente, The Netherlands, with a focus on software engineering.

He is currently the Chief Researcher of Software with the Department of Research, British Telecommunications (BT), leading a team of researchers focused on the future of software engineering. He has been a grant-holder of more than £3 million worth of funding. He has published more than 100 research articles, book chapters, and books. His primary research focuses on understanding the impact and leveraging of novel approaches and technologies, such as artificial intelligence on the practice of software development and proposing new ideas and tools together with academic and industry research partners. His research interests include software engineering, software development, software development processes, artificial intelligence in engineering, design decision optimization, fuzzy set theory, probability theory, trade-off analysis, and software product lines among others. He was a recipient of a Marie Curie Fellowship from the European Union.