IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 3 October 2022, accepted 16 October 2022, date of publication 19 October 2022, date of current version 28 October 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3215987

RESEARCH ARTICLE

Comparing Multiple Linear Regression, Deep Learning and Multiple Perceptron for Functional Points Estimation

HUYNH THAI HOC, RADEK SILHAVY, ZDENKA PROKOPOVA, AND PETR SILHAVY
Faculty of Applied Informatics, Tomas Bata University in Zlín, 76001 Zlín, Czech Republic

Corresponding author: Petr Silhavy (psilhavy@utb.cz)

This work was supported by the Faculty of Applied Informatics, Tomas Bata University in Zlín, under Project IGA/CebiaTech/2022/001 and Project RVO/FAI/2021/002.

ABSTRACT This study compares the performance of Pytorch-based Deep Learning, Multiple Perceptron Neural Networks with Multiple Linear Regression in terms of software effort estimations based on function point analysis. This study investigates Adjusted Function Points, Function Point Categories, Industry Sector, and Relative Size. The ISBSG dataset (version 2020/R1) is used as the historical dataset. The effort estimation performance is compared among multiple models by evaluating a prediction level of 0.30 and standardized accuracy. According to the findings, the Multiple Perceptron Neural Network based on Adjusted Function Points combined with Industry Sector predictors yielded 53% and 61% in terms of standardized accuracy and a prediction level of 0.30, respectively. The findings of Pytorch-based Deep Learning are similar to Multiple Perceptron Neural Networks, with even better results than that, with standardized accuracy and a prediction level of 0.30, 72% and 72%, respectively. The results reveal that both the Pytorch-based Deep Learning and Multiple Perceptron model outperformed Multiple Linear Regression and baseline models using the experimental dataset. Furthermore, in the studied dataset, Adjusted Function Points may not contribute to higher accuracy than Function Point Categories.

INDEX TERMS Software effort estimation, function point analysis, industry sector, relative size, multiple perceptron neural network, multiple linear regression, software work effort, one-hot encoding.

I. INTRODUCTION

Software Effort Estimation (SEE) might be one of the most critical phases in designing software projects [1]. Measuring resources (e.g., cost, time) accurately in the initial phase of project development is crucial to a project's short- and long-term success. The estimation helps software project managers determine how much budget is required to complete or maintain project activities. On the other hand, inaccurate estimation might lead to over-allocating resources or creating an untenable project schedule, resulting in project failure [2].

The fundamental estimation of software project effort can be based on expert judgment, project expertise, or theoretical comparisons with completed projects. Such predictive approaches are called non-algorithmic techniques [3], [4], [5]. In contrast, algorithmic-based techniques can be utilized to measure the effort of software projects in terms of software functional size.

Function point analysis (FPA) is a foundational technique for measuring the size of software projects from the user's perspective [5], [6]. Allan J. Albrecht developed this technique in 1979 at IBM. Then, it was extended by the International Function Point Users Group (IFPUG). According to previous reports [7], [8], FPA estimates the software development or maintenance independently of the technology used for implementation. For example, the functional size should be the same regardless of the problem domain, programming language, or development type.

However, it is the most challenging process for estimators in software engineering. The main reason for this could

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

be due to the diverse project lifecycle models, which may need varying amounts of resources at different phases of the project [7]. The standard estimation [8] requires more effort to record activities, increasing the difficulty and duration of the estimate. Furthermore, the experience of software developers, the software team's project history in the same business domain, and a variety of other characteristics, as well as the relationships between these factors, are sometimes not accurately accounted for [9].

According to Ali et al., several machine-learning techniques have been adopted to predict SEE [10]. Multiple Perceptron (MLP) is one of those techniques mentioned in previous reports [11], [12], [13]. Some factors have been empirically determined, such as the number of hidden layers, the number of neurons in each hidden layer, and the learning rate.

In addition, Pytorch was designed mainly by Facebook's AI research team and released in 2016 [14], [15]. It is an open-source machine learning framework, similar to TensorFlow, used to develop and train neural network-based deep learning. Pytorch, on the other hand, employs dynamic computation, allowing for greater flexibility in creating complex architectures. It is less complicated than other frameworks incorporating their programming style, such as TensorFlow [16].

MLP and Pytorch-based deep learning (PyDL) is the focus and proposed models of the current article. They will also be used to compare with the Multiple Linear Regression (MLR) [17], [18], [19] method regarding their performance and accuracy of effort estimation. An IFPUG dataset retrieved from ISBSG (release 2020 R1) will be used as a source of history datasets.

The following sections are organized as follows: Section 2 presents the Background; Section 3 highlights Related Works; Section 4 presents the Problem Formulation; Section 5 describes the Experimental Design; Section 6 presents the Results and Discussion; Section 7 highlights Threats to Validity and Section 8 conveys the Conclusion and Future Works.

II. BACKGROUND

The FPA-IFPUG method [6], [7] is adopted for this research, which is commonly used for counting the software's functional size and complexity based on user needs [6]. This method aims to count a size attribute as the number of transaction function types (EI, EO, EQ) and data function types (EIF, ILF) produced by software projects. In the ISBSG repository, EI, EO, EQ, EIF and ILF are categorised as Function Point Categories Table 1 displays the complexity weights of each component.

According to the Counting Practices Manual [7], which is responsible for creating and revising its regulations, version 4.3.1 (2010), the FPA developed by the IFP, known as the initial function point analysis, is standardized by ISO/IEC 20926:2010. There are four further ISO/IEC Functional Software Measurements (FSM): MarkII, MESMA, COSMIC, and FISMA. These approaches are outside the scope of this study; nevertheless, they may be found in the literature [21].

TABLE 1. Complexity weights of FPA components [19], [20].

Size Attributes (Function Point Categories)	Complexity Weight (CWs)		
	Low	Medium	Large
EI	3	4	6
EO	4	5	7
EQ	3	4	6
EIF	5	7	10
ILF	7	10	15

The FPA possesses most of the qualities that may be used to estimate software projects in the early stages [22]. First, function points may be allocated fully depending on the requirements or design specifications. It appears that the initiatives are still in their early stages. Second, they have nothing to do with language programming, specialized development tools, or any other type of data processing [19]. Furthermore, because the function points are designed based on the user's external perspective of the system, non-technical users of the software may find them simpler to grasp [23].

TABLE 2. General Systems Characteristics [19], [20].

GSC Factors	Characteristic	Description
F1	Data communications	Does the system require backup and recovery?
F2	Distributed Functions	Are Data Required for Communication?
F3	Performance	Does the system include a distributed processing function?
F4	Heavily Used Configuration	Is critical performance required?
F5	Transaction Rate	Will the system work during heavy loads?
F6	Online Data Entry	Does the system require direct data input?
F7	End-User Efficiency	Do data inputs require multiple screens or operations?
F8	Online Update	Are the main files up to date?
F9	Complex Processing	Are inputs, outputs, files, and queries intricate?
F10	Reusability	Is internal processing complicated and complex?
F11	Installation Ease	Is the code designed for reuse?
F12	Operational Ease	Are Conversions and installation Included in Design?
F13	Multiple Sites	Is the system designed for multiple installations in different locations?
F14	Facilitate Change	Is the application designed to make it easy for users to make changes?

A linear combination of size attributes with suitable three degrees of complexity weights is constructed to count function points. Unadjusted Function Points (UFP) are another name for this function count. Equation (1) shows the UFP formula.

$$UFP = \sum_{i=1}^{5} \sum_{j=1}^{3} BCs_{ij} \times CWs_{ij} \tag{1}$$

where BCs_{ij} is the count of component i at level j, and CWs_{ij} is an appropriate weight given in Table 1.

The outcome of the function point count is determined by multiplying the UFP by the adjustment influent factors, General System Characteristics (GSC). These may aid in the more precise counting of UFP [24]. Furthermore, Value Adjustment Factor (VAF) is defined by the formula:

$$VAF = 0.65 + 0.01 \times \sum_{i=1}^{14} F_i \times Rating_{Influence} \quad (2)$$

where denotes the GSC factor's effect, and the system is influenced by 14 different elements. Table 2 depicts these factors, while Table 3 depicts the influent factors rating.

The following equation can obtain the Adjusted Function Points (AFP):

$$AFP = UFP \times VAF \quad (3)$$

AFP can be used as an input to estimate the effort. The efforts, on the other hand, will be equal to Productivity (PDR) divided by the AFP. As indicated in equation (4), efforts will be equal to PDR into AFP.

$$Effort = AFP \times PDR \quad (4)$$

TABLE 3. Influent Factors Rating [19], [20].

Influence	Rating
None	0
Insignificant	1
Moderate	2
Average	3
Significant	4
Strong significant	5

Casper-Jones presented a rule (5) that might be the suitable solution to measure the software effort in the project's early stages when the productivity value is unknown [25]. It will be used as a baseline model for comparison.

$$Effort_{CasperJones} = \frac{AFP}{150} \times AFP^{0.4} \quad (5)$$

III. RELATED WORKS

In software engineering, there have been several studies that have employed regression models to improve effort estimation. For example, A. Sharma and N. Chaudharyin used MLR in estimating the effort for agile software development [13]. There are three models constructed to estimate the effort for agile software development. Mean Squared Error (MSE) and Mean Magnitude of Relative Error (MMRE) were measured as performance metrics. The data was collected from the Zia dataset [26]. According to the results, the model performs better in decision trees, stochastic gradient boosting, and random forests.

Hoc et al. proposed the Adj-Effort approach to optimize effort estimation in terms of FPA based on the ISBSG release 2020 [27]. They adopted Multiple Linear Regression based on AdamOptimizer [28] with 10-Fold cross-validation to optimize the effort estimation. Their findings were compared with baseline models (Casper-Jones, and FPA-IFPUG) in terms of MMRE, Mean Absolute Error (MAE), and prediction level at 0.25 (PRED(0.25)). As a result, their model outperformed the baseline models.

P. Silhavy et al. provided a novel approach for estimating software development effort [29], which used FPA, categorical variable segmentation (CVS), and stepwise regression. The stepwise regression method creates each segment as an estimating model. ISBSG dataset (Release 13, 2015) was used as the source for observational projects. Compared with baseline methods, such as non-clustered FPA and clustering-based models, the suggested model improves prediction performance in terms of Mean Absolute Percentage Error, Mean Estimation Error, and PRED(0.25). The new CVS model also outperforms the existing techniques in terms of accuracy.

Prokopova et al. used function point approaches to analyze the influence of a few parameters on work effort estimation [30]. This study considered several aspects, including the function point count technique, company location, industry sector, and relative size. Their goal was to see if the productivity acquired from the training dataset could be used to estimate effort and if the performance of the estimates is influenced by the factors used. There are 1,333 selected projects in the ISBSG dataset as a historical dataset (Release 13). The dataset was divided into sub-datasets, including DS1 for training and DS2 for testing, which was made using the hold-out method in a 2:1 ratio. Estimation was done using MATLAB software.

Furthermore, the VAF plays a crucial role in improving the accuracy of AFPs based on 14 General System Characteristics. However, according to the ISBSG, this component may have gone uncounted recently in most projects. As a result of this problem, effort estimation in FPA may be inaccurate. Z. Prokopova et al. introduced Modified Function Points (MFP) methodologies based on the regression model approach in 2018 and investigated the impact of VAF on SEE accuracy [31]. Three techniques for estimating effort were tested in the variants with and without the VAF factor based on ISBSG. As a result, the VAF factor had no bearing on estimating precision.

On the other hand, recent studies have estimated software effort accurately using machine learning. Researchers are attempting to determine which machine learning estimation technique produces the most accurate results based on commonly accepted evaluation measures (e.g., MMRE, PRED(0.25)). Moreover, machine learning-based effort estimation techniques have already been published, although the authors of relevant studies are often unaware of them. Neural network-based models are a relatively new addition to the arsenal of machine learning techniques [6]. They are utilized in software effort estimating because they learn faster and more efficiently with more accurate results [23], [32].

Ramessur and Nagowah proposed a model that uses machine learning approaches to assess and forecast sprint effort while considering a variety of parameters that influence sprint performance [33]. Several regression algorithms were used to validate the model, including linear regression,

K-nearest neighbour, decision tree, polynomial kernel, radius basis function, and MLP. Using the MLP method, the model yielded more accurate estimations with reduced error values. In model evaluation studies, MAE and Root Mean Square Error are two metrics frequently used. Companies of Mauritius were hesitant to share their datasets due to data standards of security.

Suyash Shukla et al. proposed MLP to improve the SEE process [34]. They introduced MLP, Ridge-MLP, Lasso-MLP, Bagging-MLP, and AdaBoost-MLP models and found that AdaBoost-MLP achieved the highest accuracy. Desharnais was used as a historical dataset, and the R-squared metric was used as a comparison [9]. Moreover, Somya Goyal et al. adopted a non-linear technique for effort estimation using MLP with the Back Propagation algorithm [35]. They used the Maxwell dataset in their study with MMRE and the Median of Magnitude of Relative Error. The results outperformed the linear regression technique. N. Rankovic et al. focused on different activation functions in Artificial Neural Networks (ANNs) and used the Taguchi method to improve effort, and cost estimation [23]. A. B. Nassif et al. proposed four models, including MLP, general regression neural network, radial basis function neural network, and cascade correlation neural network [36]. Their goal was to compare neural network models in an unbiased manner, using ISBSG Release 11 as a practical dataset, which has more than 5000 completed projects. MMRE and MAE were adopted for comparing models.

Moreover, E. Okewu et al. focused on the performance of Adam, Root Mean Squared Propagation (RMSProp), Adaptive Gradient Algorithm (Adagrad), and more robust extension of Adagrad (Adadelta) in their research into improving loss quality and training over time [37]. Their findings indicated that RMSProp and Adam make adaptive moment estimations to enhance results.

Categorical factors are frequently used in sociological research [38]. To properly consider using these variables in predictive methods, they must be classified into several distinct dummy categories that may directly contribute to the model. Many researchers have included those as predicted variables in their models by encoding them [38], [39], [40]. According to Sebastian Gnat, there are five ways to encode, including one-hot encoding, cat boost encoding, Helmert encoding, target encoding, and ordinal encoding, and the one-hot encoding method might perform the best [40]. This paper will adopt the one-hot encoding technique for encoding Industry Sector and Relative Size.

According to P. Pospieszny et al., there is a total of 11 factors that impact effort estimation, including Agile, Application Type, Architecture, Development Platform, Development Type, Industry Sector, Language Type, Package customization, Relative Size, Resource Level, and Used Methodology [30]. As mentioned above, P. Silhavy et al. introduced a novel SEE [27]. Their approach is based on FPA and categorical variable segmentation. Relative Size (RS), Industry Sector (IS), and Business Area are three categorical variables mentioned in their research [12]. As a result, they concluded that Relative Size might contribute the most to accuracy compared with Industry Sector and Business Area Type.

IV. PROBLEM FORMULATION

As previously mentioned, RS and IS are significant factors affecting effort estimation accuracy. Accordingly, this paper utilizes both factors by combining selected categorical variables with the IFPUG size attributes for the application of estimating software effort. ANNs, such as MLP, PyDL, and MLR are considered in this study. The latest ISBSG dataset (release R1/2020) is used as an experimental data source. The Scikit-learn/Keras/Pytorch/dummy python library has been selected for implementation.

A. RESEARCH QUESTIONS AND HYPOTHESIS FORMULATION

Three research questions (RQs) must be solved to accomplish the task:

1) RQ1: How do the IS and RS influence the estimation accuracy of the MLR, MLP, and PyDL models?
2) RQ2: Which predictors set is the highest performing for the MLR, MLP, and PyDL models?
3) RQ3: Is the PyDL/MLP model estimation more accurate than the MLR model?
4) RQ4: The model improves estimation accuracy when compared to baseline models?

As described later, several experiments will be performed to respond to the RQs. We assume that the estimation error of the PyDL/MLP model is sufficiently low when one of the predictor combinations is significantly lower than the prediction errors of the baseline methods. A statistical hypothesis was tested to determine if the PyDL/MLP model performs better:

$H_0 : \mu_{Metric_{PyDL/MLP}} = \mu_{Metric_{MLR/baseline}}$; there is no prediction capability difference between baseline and tested models. There is no difference in the mean of evaluation metrics, as described later.

Alternative hypothesis:

$H_1 : \mu_{Metric_{PyDL/MLP}} < \mu_{Metric_{MLR/baseline}}$; there is a difference in prediction capability between FPA and the PyDL/MLP or MLR. The mean of the test metric is significantly lower for at least one of the tested models compared with the FPA method.

This paper compares the accuracy of the tested models with that of the baseline model's t-test. The paired t-test for two samples is used as a test of the null hypothesis.

B. EVALUATION METRICS

This section provides standards for validating the precision of each method's effort estimation. There are standard tests to validate the efficiency of experiments, such as Magnitude of Relative Error (MRE) [38] and MMRE. However, Shepperd and MacDonell suggest not using these metrics because of their biased nature [41]. This study uses further

criteria to improve the experiment's efficiency. MAE refers to the discrepancy between the expected and actual results. Moreover, PRED(x), Mean Balance Relative Error (MBRE), Mean Inverted Balance Relative Error (MIBRE) values are considered as additional further research criteria.

Additionally, SA is a denotation of Standardized Accuracy (9), where $\overline{MAE_p}$ is the average value of the large number (typically 1000), runs of random guessing [41], [43], [44]. It is a metric that compares how improved the prediction model is relative to the baseline model [41]. Higher SA values indicate a more accurate predictive model. A negative value implies the accuracy is less sufficient than random guessing. Whereas the objective is to maximize PRED(x) and SA, the objective for all remaining evaluation measures is to be minimized.

Furthermore, R^2 and *Adjusted* $R^2(Adj - R^2)$ are statistical indicators of the goodness-of-fit. Their values range from 0 to 1. Greater R^2 and $(Adj-R^2)$ values represent a more accurate estimation ability of the model [45]. These criterion formulas are expressed as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right| \tag{6}$$

$$MRE_i = \frac{\left|y_i - \hat{y}_i\right|}{y_i} \tag{7}$$

$$SA = (1 - \frac{MAE}{\overline{MAE_p}}) \times 100 \tag{8}$$

$$PRED(x) = \frac{1}{n}\begin{cases} 1 & \text{if } MRE_i \leq x \\ 0 & otherwise \end{cases} \tag{9}$$

$$MBRE = \frac{1}{n}\sum_{i=1}^{n}\frac{\left|y_i - \hat{y}_i\right|}{min(y_i, \hat{y}_i)} \tag{10}$$

$$MIBRE = \frac{1}{n}\sum_{i=1}^{n}\frac{\left|y_i - \hat{y}_i\right|}{max(y_i, \hat{y}_i)} \tag{11}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}{\sum_{i=1}^{n}\left(y_i - \bar{y}_i\right)^2} \tag{12}$$

$$Adj - R^2 = 1 - \frac{n-1}{n-l-1}\left(1 - R^2\right) \tag{13}$$

where n is the number of measurements; l is the number of predictors; y_i is the observed value; $\hat{y}_i$ is the predicted value; $\bar{y}$ is the mean of the observed value.

V. EXPERIMENTAL DESIGN

A. TESTED MODELS

1) MLR
2) MLP
3) PyDL

MLR is the most popular method for observing the relationship between a response variable and predictor variables [2]. It is used to predict SEE based on a given set of independent variables. The formation of MLR is written as a linear equation between a dependent variable and a number of p independent variables, $X_1, X_2, \ldots, X_p$, as follows:

$$y \approx \beta_0 + \beta_1 \times X_1 + \beta_2 \times X_2 + \ldots + \beta_p \times X_p + \epsilon \tag{14}$$

where y is the response variable, which stands for the output in this paper; $X_1, X_2, \ldots, X_p$ can represent one of six combination groups of predictors, including AFP, EI, IO, EQ, EIF, ILF, RS, IS; β_0 is an intercept, and $\beta_1, \beta_2, \ldots, \beta_p$ are regression coefficients; ϵ is presented as an error residual. The intercept and regression coefficients are unknown values. Hoc et al. mentioned that unknown variables might be resolved by adopting the Adam-Optimizer approach with 10-Fold cross-validation based on a historical dataset [28]. This approach will be adopted in this research to determine the unknown values.

MLP creates a network to simulate the process of the human brain [46]. When addressing an effort estimation issue, MLP is used to learn the improved weight values corresponding to each link in the network to achieve the minimum discrepancy between the estimated and real effort.

The first layer is the input layer (see Figure 1). A total of up to six combination groups of predictors are considered in this study. Each group of these predictors is added to this layer. Summary Work Effort (SWE) is the output layer. As previously mentioned, E. Okewu et al. (2020) stated that RMSProp and Adam might make adaptive moment estimations to improve predictive effort estimation. Therefore, RMSProp is used in the hidden layers to adjust the weights of the input variables.

In addition, MSE is used to improve a model's fit [47]. Nwankpa, Ijomah, Gachagan, and Marshall concluded that the Rectified Linear Unit (ReLU) is the common activation function choice [48]. Thus, ReLU is chosen as the activation function in this process. Similarly, 10-Fold cross-validation is also employed to examine its performance and reliability. Moreover, the number of hidden layers and the number of nodes in each hidden layer are suggested experimentally based on the criterion of the highest R^2 value.

Regarding PyDL model, the following requirements are designed: (i) a constructor where the layers, the number of nodes, and type are specified (linear, convolutional, pooling). (ii) Backwards: This function determines the links between the neural network layers. The activation functions are ReLU for the hidden layers. The loss function used in this study is cross-entropy. The optimizer is RMSprop, which is used for neural network training using backpropagation, and the learning rate is 0.001. Based on the experimental results, weight decay and the number of hidden layers are chosen. (iii). The optimizer.step() method calls to update the network weights based on the learning rate parameter value, which defines how much the model changes in response to the estimated loss function value. Finally, 10-Fold cross-validation is also applied in this model.

Three models, MLR, MLP, and PyDL are tested with the following predictors (model inputs):

- P1: AFP
- P2: EI, EO, EQ, EIF, ILF

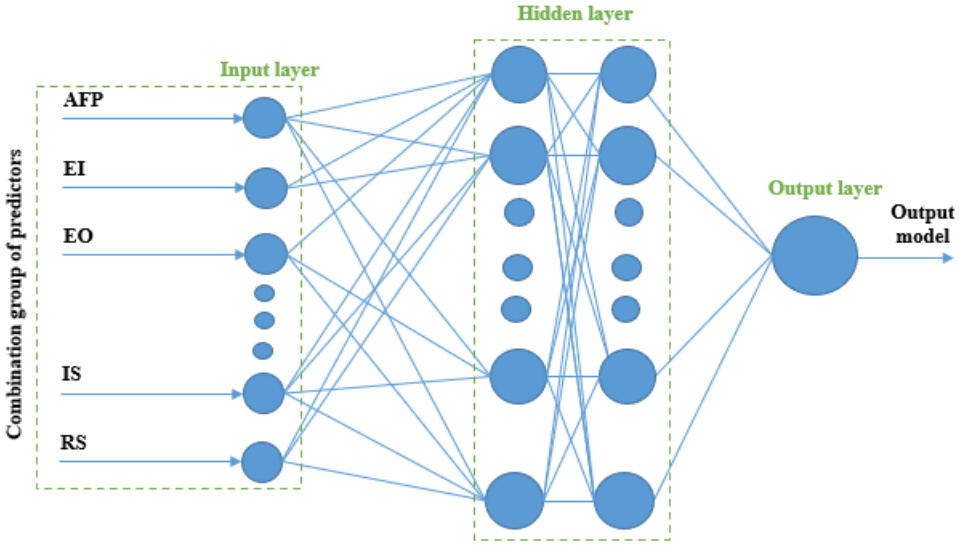

FIGURE 1. The structure of MLP models used to evaluate the effort estimation.

- P3: AFP, IS
- P4: EI, EO, EQ, EIF, ILF, IS
- P5: AFP, IS, RS
- P6: EI, EO, EQ, EIF, ILF, IS, RS

The dependent variable (model output) is SWE, representing development effort in person-hours, as reported in the dataset. Baseline models:

- FPA-IFPUG
- Capers-Jones

B. RESEARCH METHODOLOGY

The experimental process is described in Figure 2. The experimental process starts with dataset processing, as described in detail in the following section. When a dataset is processed, then experiments are performed.

For MLR models:

1) Creating the training and testing sets by using the ratio (80:20) method
2) Setting a one of predictors list (from P1 to P6), the dependent variable was set to SWE
3) Applying MLR to all projects in the training set based on a 10-Fold cross-validation
4) Computing R^2, $Adj - R^2$, and performance metrics (section IV.B)

To obtain a new estimation, the procedure is as follows:

1) For each observation in the testing dataset, use the list of predictors (from P1 to P6)
2) Pre-training model performs an estimation of work effort in hours
3) Calculation of evaluation metrics for projects in the testing set is then computed for each observation

For MLP and PyDL models:

1) Creating the training and testing sets by using the ratio (80:20) method
2) Setting one of the predictors list (from P1 to P6) as input
3) Model output is SWE
4) Applying a training procedure for the MLP and PyDL model based on 10-fold cross-validation of all projects in the training set
5) Computing R^2, $Adj - R^2$, and performance metrics (section IV.B)

To obtain a new estimation, the procedure is as follows:

1) For each observation in the testing dataset, use P1 to P6 as inputs
2) Pre-training MLP and PyDL models perform estimation of work effort in hours
3) The number of hidden layers and number of nodes in each hidden layer is chosen based on the highest R^2 value
4) The evaluation metrics for projects in the testing set are then computed

The expression for R^2, $Adj - R^2$, are given in equation (12) and (13). They are statistical values used to measure how close the observed data are to the predicted data to calculate the approximate accuracy of an ANN. The selected model is most effective with a minimum for the MMRE, MBRE, MIBRE, MAE and a maximum for the PRED(0.3) and SA.

C. DATASET PROCESSING

This paper uses the ISBSG projects repository (release R1/2020) [27] for the source of observational projects. They have 9,592 completed software projects with 251 recorded attributes, and these attributes are divided into several groups,

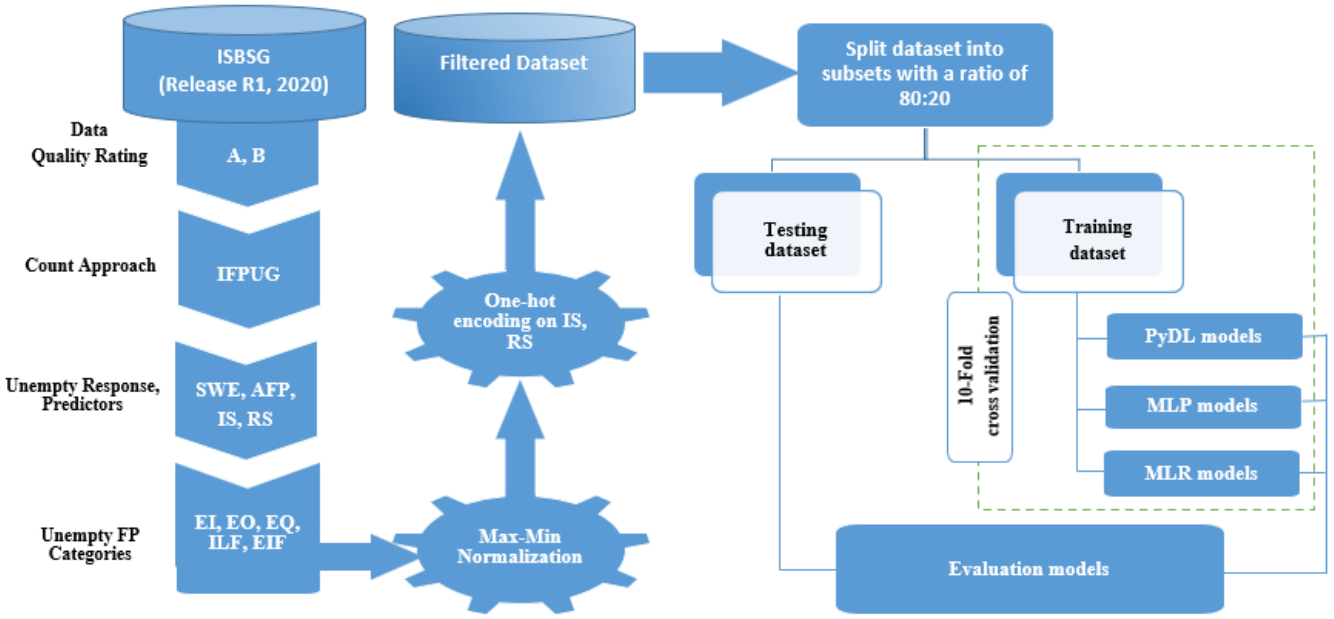

FIGURE 2. Experiment Design Diagram.

including Rating, Development Type, and Sizing. For our case, the Sizing group represents the AFP and VAF. Size attributes are recorded in the Size group attributes. The Effort group covers Normalized Work Effort and SWE. In this paper, SWE provides the total effort in hours recorded against the observed project. Moreover, the rating code denoted A, B, C, or D (from high-quality to low-quality), is illustrated in the Rating group. Industry sector, Relative Size, and Development Type are recorded in the Major group attributes.

Datasets are processed to ensure the availability of all the necessary variables for all of the experiments. The datasets are filtered using the criteria mentioned below.

1) The study only considers data projects with higher quality (A and B ratings), as advised by ISBSG and previous reports [27], [49]. The other hypothesis considers insufficient legitimacy attributable to either an impact of inadequate evidence or a combination of factors [50], lowering the scope of the dataset to 8,619 projects.
2) This study primarily depends on the IFPUG's FPA-based counting approach. The scale is decided by MarkII, MESMA, COSMIC, and FISMA, all of which are skipped, leaving 6,365 records remaining.
3) This paper focuses on the influence between SWE and AFP, EI, EO, EQ, ILF, ELF, Industry Sector, and Relative Size, so all those uncounted will be ignored, leaving 1,515 observational projects.
4) Moreover, the interquartile range (IQR) approach is employed to remove outliers. The values of SWE, AFP, EI, EO, EQ, ILF, and ELF might be considered for elimination if they are out of the range from $Q_1 - 1.5 \times IRQ$ to $Q_3 + 1.5 \times IRQ$, where Q_1 is the first quartile, and Q_3 is the third quartile [51].
5) One-hot encoding technique [9] is applied to convert categorical variables into numerical variables. A dummy variable will be defined corresponding to each kind of category. It means that m dummy variables are identical if a categorical variable contains m – 1 category. This research is employed for both categorical variables, IS and RS. Each type is mapped to a Boolean variable with either a value of zero or one.
6) The features in the dataset have many scales, SWE, AFP, EI, EO, EQ, ILF, EIF, and even PDR, so to ensure that they had the same degree of influence [52], [53], they were scaled using max-min normalization as presented in equation (15).

$$X_i = \frac{x_i - min(X}{max(X) - min(X)} \tag{15}$$

D. VALIDITY EVALUATION

Validity evaluation is an inaccurate evaluation method used to analyze the models in this study, which might affect the validity of conclusions produced from experimental research. Specifically, it pertains to the process of statistical sample validation. The 10-fold cross-validation method was used to mitigate the validity issue, ensuring that the suggested approach is appropriately examined. The selection of parameters in the PyDL, MLP and MLR techniques is another type of evaluation that may affect the validity of the generated findings. In this study, we employ the default parameter settings of the MLP technique and the experimentally obtained parameters for PyDL.

The range of validity of data produced in this paper is concerned with external validity and whether the results can

be utilized in a different environment. The prediction ability was tested using the ISBSG 2020 (release R1).

The performance accuracy of the suggested method is assessed using evaluation metrics. According to previous reports [41], [42], [54], evaluation metrics such as MMRE, MBRE, MIBRE, MAE, PRED(0.3), and SA according to previous reports. As a result, the experimental findings of this study are highly generalizable.

VI. RESULTS AND DISCUSSION

This section presents the results of proposed approaches based on criteria validation. The findings of MLR, MLP, and PyDL models are shown in Table 4.

TABLE 4. The R^2 and $Adj - R^2$ of models PyDL, MLP, and MLR.

Predictors	PyDL		MLP		MLR	
	R^2	$Adj-R^2$	R^2	$Adj-R^2$	R^2	$Adj-R^2$
P1	0.98	0.98	0.02	0.01	-0.05	-0.08
P2	0.96	0.96	0.11	0.05	-0.08	-0.11
P3	0.97	0.97	0.34	0.21	0.06	0.03
P4	0.98	0.98	0.73	0.68	0.08	0.05
P5	0.91	0.91	0.32	0.18	0.07	0.05
P6	0.95	0.95	0.32	0.17	0.07	0.04

As shown in Table 4, the predicted SWE obtained from MLP/PyDL models are fitter than those obtained from MLR models, as seen from the R^2 and $Adj - R^2$ values. The respective MLP/PyDL are more significant than those obtained from MLR. The combination of predictors between EI, EO, EQ, EIF, ILF, and IS created the highest accuracy. Moreover, the findings reveal that PyDL outperforms better than MLP.

TABLE 5. The performance of effort PyDL-based, MLP-based vs. MLR-based estimation methods.

Predictors	MMRE	MBRE	MIBRE	MAE	PRED(0.3)	SA
PyDL						
P1	0.505	0.61	0.32	1026.09	0.50	0.67
P2	0.495	0.60	0.29	971.55	0.54	0.67
P3	0.443	0.54	0.28	885.18	0.55	0.68
P4	**0.225**	**0.28**	**0.19**	**583.81**	**0.72**	**0.72**
P5	0.465	0.61	0.31	904.46	0.54	0.62
P6	0.32	0.54	0.27	901.77	0.54	0.71
MLP						
P1	0.692	0.64	0.34	1500.00	0.39	0.38
P2	0.646	0.63	0.31	1419.59	0.39	0.41
P3	0.477	0.62	0.30	1321.81	0.47	0.38
P4	**0.387**	**0.34**	**0.19**	**1042.09**	**0.61**	**0.53**
P5	0.507	0.62	0.31	1345.59	0.50	0.37
P6	0.487	0.55	0.27	1308.81	0.50	0.39
MLR						
P1	0.636	1.09	0.34	1534.66	0.39	0.29
P2	0.651	0.94	0.34	1522.85	0.39	0.32
P3	0.593	0.85	0.30	1421.43	0.43	0.35
P4	**0.556**	**0.51**	**0.29**	**1346.76**	**0.50**	**0.40**
P5	0.595	0.76	0.33	1406.29	0.39	0.37
P6	0.591	0.63	0.32	1380.30	0.43	0.39

Table 5 presents the results of all criteria (MMRE, MBRE, MIBRE, MAE, PRED(0.3), SA) used for comparing the estimation methods with comparative models. It is clear that MMRE, MBRE, MIBRE, and MAE obtained from estimation models (PyDL, MLP), are less than those obtained from comparative models, while and SA attain the maximum performance. In addition, the values of MMRE, MBRE, MIBRE, MAE, PRED(0.3), and SA based on P4 are better than those found with other combination groups (P1, P2, P3, P5, P6) in both estimation methods vs comparative models.

Table 5 presents predictive statistics adopted to validate the accuracy of MLR and MLP models compared to baseline models (Capers Jones and FPA-IFPUG). As seen, the researched models perform better than the comparative models. The values of MMRE, MBRE, MIBRE, and MAE obtained from MLR-P4/MLP-P4 showed the minimum values. The PRED(0.3) and SA of MLR-P4/MLP-P4 attained maximum values. Significantly, the deep learning model obtained from Pytorch has the best performance compared with MLR and MLP. The PyDL model also demonstrates that P4 is the best predictor to obtain the good-fitness model among six predictors.

TABLE 6. The performance of effort estimation methods vs comparative models.

Predictors	MMRE	MBRE	MIBRE	MAE	PRED(0.3)	SA
MLR - P4	0.556	0.51	0.29	1346.76	0.50	0.40
MLP-P4	0.387	0.34	0.19	1042.09	0.61	0.53
PyDL-P4	**0.225**	**0.28**	**0.19**	**583.81**	**0.72**	**0.72**
Capers Jones	0.725	3.37	0.69	2679.22	0.07	0.00
FPA-IFPUG	0.612	1.46	0.56	2361.77	0.14	0.00

In conclusion, the criteria mentioned above are statistically the most relevant for the creation of our model, answering RQ1:

1) **RQ1: How do the IS and RS influence the estimation accuracy of the MLR, MLP, and PyDL models?**
Table 4 shows that the values of R^2 and $Adj - R^2$ obtained from the studied models without using IS (P1, P2) might be less accurate than models with IS (P3, P4). In fact, R^2 values attained from the proposal models with IS are all larger than those from models without using those categorical variables. However, comparing P3 vs P5 and P4 vs P6 in terms of R^2 and $Adj - R^2$, we can see that the predictive estimation might be less accurate when the RS factor is added to those models. The results reveal that AFP+EI+EO+EQ+ILF+EIF+IS combined with RS might not enhance estimation accuracy only by combining models with IS.
Moreover, **Table 5** presents the accuracy of effort estimation based on PyDL, MLP and MLR models. As can be seen in the three approaches, the performance of effort obtained from P3 and P4 is better than that obtained from P1 and P2, respectively. These findings reveal that IS positive impacts effort estimation. By contrast, when we include more RS factors in models (P5 and P6), their performance might be less accurate than that of the corresponding models (P3 and P4). It might reveal that RS is a negative influence on effort estimation.

2) **RQ2: Which predictors set is the highest performing for the MLR, MLP, and PyDL models?** As seen in **Table 5**, the experimental statistics in terms of MBRE, MIBRE, MAE, PRED(0.3), and SA for evaluating the groups of predictors (particularly P1 and P2) reveal that Function Point Categories (EI, EO, EQ, EIF, ILF) might contribute to higher performance than AFP. The best predictors for estimating effort in MLR, MLP, and PyDL appear to be P4, where MMRE, MBRE, MIBRE, and MAE showed the minimum values, PRED(0.3) and SA attained the maximum values. This conclusion might indicate that we might need Function Point Categories as predictors rather than AFP in terms of effort estimation. As can be seen in equation (4), AFP is measured by UFP multiplied by VAF. It means that we need to measure UFP based on equation (1) as well as calculate VAF based on equation (2). These might increase the complexity of the calculation.
3) **RQ3: Is the PyDL/MLP models estimation more accurate than the MLR model? Table 5** presents the accuracy of the effort estimating approaches based on MLR, MLP, PyDL. As seen, all the corresponding criteria of PRED(0.3), and SA obtained from PyDL/MLP are greater than those obtained from MLR. PyDL-P4/MLP-P4 might be the most accurate among the researched models because MMRE, MBRE, MIBRE, and MAE are the most minimized, while PRED(0.3) and SA are maximized.

TABLE 7. Hypothesis t-test results of MLP and MLR models.

	t-value	p-value
MLP		
P1	2.901	0.007
P2	2.509	0.018
P3	2.339	0.028
P5	2.691	0.012
P6	2.235	0.034
MLR		
P1	2.746	0.010
P2	2.438	0.027
P3	2.725	0.011
P4	2.581	0.016
P5	2.500	0.018
P6	2.471	0.020

In addition, based on the paired t-test results shown in **Table 7** and **Table 8**, we find that the values of MREs are significantly different between MLP- P4 and other models (**Table 7**) and between PyDL-P4 and other models **(Table 8**) due to all the p-values being less than 0.05. This result suggests that one model may reject the null hypothesis and accept the alternative hypothesis. These experimental findings show that PyDL-P4/MLP-P4 is the best suited for estimation accuracy among the studied models, and PyDL-P4 outperforms better than MLP-P4.

4) **RQ4: The model improves estimation accuracy when compared to baseline models?**

TABLE 8. Hypothesis t-test results of PyDL and MLP models.

	t-value	p-value
PyDL		
P1	2.773	0.006
P2	2.773	0.006
P3	2.809	0.005
P5	3.111	0.003
P6	2.166	0.021
MLP		
P1	2.973	0.004
P2	2.642	0.007
P3	2.196	0.019
P4	1.893	0.036
P5	2.221	0.018
P6	2.074	0.025

As discussed in RQ2, PyDL-P4/MLP- P4 is the best-fit model among the studied estimation methods. The criteria validation mentioned in section 4.2 is adopted to validate the effort estimation performance compared with baseline methods. As seen from **Table 6**, MMRE, MBRE, MIBRE, and MAE obtained from PyDL-P4/MLP-P4 are less than those obtained from Capers Jones and FPA-IFPUG, while SA is maximized for PyDL-P4/MLP-P4. Furthermore, paired t-test was applied to determine the differences in MREs between the models (Table 9). As a result, the p-value is less than 0.05, which proves a significant difference between the models.

TABLE 9. Hypothesis t-test results of baseline models.

	t-value	p-value
PyDL		
Effort obtained by Capers Jones	14.404	0.000
FPA-IFPUG	10.170	0.000
MLP		
Effort obtained by Capers Jones	2.211	0.035
FPA-IFPUG	3.302	0.002

VII. THREATS TO VALIDITY

A. INTERNAL THREATS

The most significant threats, in our opinion, are to the study's internal validity or the extent to which conclusions can be drawn about the better parameter setup for PyDL and MLP-based effort estimation. One possible threat to internal validity is the selection of appropriate parameters. There are no specific rules for determining such parameters for each dataset.

Although it is widely acknowledged that the appropriate parameters significantly impact the identification of good fitness models, we have chosen parameter values for the deep learning and MLP model in this study by using experiments. We believe this decision is justified, even though it costs time-consuming. Although the effectiveness of the performance measure used as the preventing criterion has been emphasized, complete certainty in this regard has been challenged, and we are forced to rely on standard performance measures; MMRE, MBRE, MIBRE,PRED(0.30), MAE and SA as the

basis of selection. Because this study was motivated by previous studies that used MMRE, MBRE, MIBRE, PRED(0.3), MAE, and SA as optimization criteria [36], [41], we do not consider the choice an issue. We used the 10-fold cross-validation strategy to compare various adaptation techniques. The main reason is that 10-fold cross-validation has previously been used in studies and is recommended by [18], [25], and [29] for comparing effort estimation models.

B. EXTERNAL VALIDITY

The ISBSG dataset 2020 (the latest version of the ISBSG organization) was used. Compared to the previous version, there are no new additions to the new developments category. Furthermore, we believe that some datasets are too old to estimate software effort estimation because they represent various software development methods and technologies. This could be due to the fact that new development projects have not yet been completed, so there is no new updated information available.

VIII. CONCLUSION AND FUTURE WORKS

In this paper, the groups of six predictors (P1 to P6) are considered input variables for two effort estimation models, where the output is SWE. Multiple Perceptron Neural Networks, Pytorch-based Deep Learning and Multiple Linear Regression with 10-Fold cross-validation are employed for effort estimation models. The IFPUG dataset (ISBSG 2020, release R1) was used as the observational dataset for this study. These datasets were randomly separated into two sub-datasets, 80% of the dataset for training and 20% for testing all the researched models.

We concluded that effort estimation based on predictor P4 brings more accurate results than others. First, using Function Point Categories as input variables might yield higher predictive model accuracy than using AFP. The statistical results such as MBRE, MIBRE, MAE, PRED (0.3) and SA in all three methods obtained from P2/P4 may be better than the results obtained from P1/P5 (see Table 5).

On the other hand, the findings show that Function Point Categories integration with Industry Sector might result in higher accuracy for effort estimation. By contrast, adding Relative Size to the input variables might negatively affect the prediction accuracy. We highly recommend using Industry Sector rather than the Relative Size factor in effort estimation and adopting Function Point Categories rather than AFP.

Regarding the performance of the three different approaches, the findings show that PyDL, and MLP approaches might result in improved good-fitness models compared with MLR. PyDL-P4/MLP-P4 achieved the highest accuracy, with PRED(0.3) of 72%/61% and SA of 72%/53%, and PyDL may yield promising results compared with MLP.

The limitation of this research is that the findings are based on a filtered dataset with limited data. These results should be validated when we have more datasets in the future. Furthermore, while cross-validation with 10-fold was used in this paper, we might not ensure that observational projects are treated fairly in selecting the training and validation datasets in the folds, especially for categorical attributes.

The data augmentation and class weight approaches are standards in deep learning. Data augmentation is an approach to increasing the diversity of training data [55], and class weight is used to determine the weight of each categorical variable when the dataset is unbalanced [56]. These approaches might be employed to ensure the diversity and completeness of selected projects in the imbalanced projects. In the future, these suggested methods should be considered to uncover additional factors that positively impact the predictive model of effort estimation.

Last but not least, Conic Multivariate Adaptive Regression Splines (CMARS) [57] are an alternative to the backward step of MARS proposed by [57] and [58], and Bootstrapping CMARS [59] should be employed to verify their general usefulness, find other potential flaws and limitations, and realize even more effective methods.

REFERENCES

[1] A. Trendowicz and R. Jeffery, "Software project effort estimation," in *Foundations and Best Practice Guidelines for Success, Constructive Cost Model–(COCOMO)*, vol. 12. Springer, 2014, pp. 277–293.

[2] Y.-S. Seo, D.-H. Bae, and R. Jeffery, "AREION: Software effort estimation based on multiple regressions with adaptive recursive data partitioning," *Inf. Softw. Technol.*, vol. 55, no. 10, pp. 1710–1725, Oct. 2013.

[3] S. W. Munialo and G. M. Muketha, "A review ofagile software effort estimation methods," *Int. J. Comput. Appl. Technol. Res.*, 2016.

[4] H. T. Hoc, V. V. Hai, and H. L. T. K. Nhung, "A review of the regression models applicable to software project effort estimation," in *Proc. Comput. Methods Syst. Softw.*, 2019, pp. 399–407.

[5] C. A. Behrens, "Measuring the productivity of computer systems development activities with function points," *IEEE Trans. Softw. Eng.*, vol. SE-9, no. 6, pp. 648–652, Nov. 1983.

[6] IFPUG. *International Function Point Users Group*. Accessed: Dec. 2021. [Online]. Available: http://www.ifpug.org/

[7] International Function Point Users Group, *Function Point Counting Practices Manual*, Princeton Junction, Princeton, NJ, USA, 2010.

[8] P. Suresh Kumar, H. S. Behera, A. K. K. J. Nayak, and B. Naik, "Advancement from neural networks to deep learning in software effort estimation: Perspective of two decades," *Comput. Sci. Rev.*, vol. 38, Nov. 2020, Art. no. 100288.

[9] P. Pospieszny, B. Czarnacka-Chrobot, and A. Kobylinski, "An effective approach for software project effort and duration estimation with machine learning algorithms," *J. Syst. Softw.*, vol. 137, pp. 184–196, Mar. 2018.

[10] A. Ali and C. Gravino, "A systematic literature review of software effort prediction using machine learning methods," *J. Softw., Evol. Process*, vol. 31, no. 10, Oct. 2019.

[11] P. S. Rao, K. K. Reddi, and R. U. Rani, "Optimization of neural network for software effort estimation," in *Proc. Int. Conf. Algorithms, Methodol., Models Appl. Emerg. Technol. (ICAMMAET)*, Feb. 2017, pp. 1–7.

[12] M. Azzeh and A. B. Nassif, "Project productivity evaluation in early software effort estimation," *J. Softw., Evol. Process*, vol. 30, no. 12, p. e2110, 2018.

[13] A. Sharma and N. Chaudhary, "Linear regression model for agile software development effort estimation," in *Proc. 5th IEEE Int. Conf. Recent Adv. Innov. Eng. (ICRAIE)*, Dec. 2020, pp. 1–4.

[14] E. Stevens, L. Antiga, and T. Viehmann. (2020). *Deep Learning With PyTorch*. [Online]. Available: https://pytorch.org/assets/deep-learning/Deep-Learning-with-PyTorch.pdf

[15] V. Nejkovic, M. Radenkovic, and N. Petrovic, "Ultramarathon result and injury prediction using PyTorch," in *Proc. 15th Int. Conf. Adv. Technol., Syst. Services Telecommun. (TELSIKS)*, Oct. 2021, pp. 249–252, doi: 10.1109/TELSIKS52058.2021.9606348.

[16] S. Imambi, K. B. Prakash, and G. R. Kanagachidambaresan, *PyTorch in Programming With TensorFlow: Solution for Edge Computing Applications*. Cham, Switzerland: Springer, 2021, pp. 87–104.
[17] R. Silhavy, P. Silhavy, and Z. Prokopova, "Algorithmic optimisation method for improving use case points estimation," *PLoS ONE*, vol. 10, no. 11, Nov. 2015, Art. no. e0141887.
[18] H. L. T. K. Nhung, V. Van Hai, R. Silhavy, Z. Prokopova, and P. Silhavy, "Parametric software effort estimation based on optimizing correction factors and multiple linear regression," *IEEE Access*, vol. 10, pp. 2963–2986, 2022, doi: 10.1109/ACCESS.2021.3139183.
[19] A. J. Albrecht and J. E. Gaffney, "Software function, source lines of code, and development effort prediction: A software science validation," *IEEE Trans. Softw. Eng.*, vols. SE–9, no. 6, pp. 639–648, Nov. 1983.
[20] A. J. Albrecht, "Measuring application development productivity," in *Proc. Joint Share, Guide, IBM Appl. Develop. Symp.*, 1979.
[21] C. Gencel and O. Demirors, "Conceptual differences among functional size measurement methods," in *Proc. 1st Int. Symp. Empirical Softw. Eng. Meas. (ESEM)*, Sep. 2007, pp. 305–313.
[22] G. C. Low and D. R. Jeffery, "Function points in the estimation and evaluation of the software process," *IEEE Trans. Softw. Eng.*, vol. 16, no. 1, pp. 64–71, Jan. 1990.
[23] N. Rankovic, D. Rankovic, M. Ivanovic, and L. Lazic, "A new approach to software effort estimation using different artificial neural network architectures and Taguchi orthogonal arrays," *IEEE Access*, vol. 9, pp. 26926–26936, 2021.
[24] J. E. Matson, B. E. Barrett, and J. M. Mellichamp, "Software development cost estimation using function points," *IEEE Trans. Softw. Eng.*, vol. 20, no. 4, pp. 275–287, Apr. 1994.
[25] H. T. Hoc, V. van Hai, and H. L. T. K. Nhung, "An approach to adjust effort estimation of function point analysis," in *Proc. Comput. Sci. On-Line Conf.*, 2021, pp. 522–537.
[26] S. K. T. Ziauddin and S. Zia, "An effort estimation model for agile software development," *Adv. Comput. Sci. Appl.*, vol. 2, no. 1, pp. 314–324, 2012.
[27] *ISBSG*, Release R1, International Software Benchmarking Standards Group, South Melbourne VIC, USA, 2020.
[28] H. T. Hoc, V. van Hai, and H. L. T. K. Nhung, "AdamOptimizer for the optimisation of use case points estimation," in *Proc. Comput. Methods Syst. Softw.*, 2020, pp. 747–756.
[29] P. Silhavy, R. Silhavy, and Z. Prokopova, "Categorical variable segmentation model for software development effort estimation," *IEEE Access*, vol. 7, pp. 9618–9626, 2019.
[30] Z. Prokopova, P. Silhavy, and R. Silhavy, "Influence analysis of selected factors in the function point work effort estimation," in *Proc. Comput. Methods Syst. Softw.*, 2018, pp. 112–124.
[31] Z. Prokopova, P. Šilhavý, and R. Šilhavý, "VAF factor influence on the accuracy of the effort estimation provided by modified function points methods," in *Proc. Ann. DAAAM, Int. DAAAM Symp.* Danube Adria Association for Automation and Manufacturing (DAAAM), 2018.
[32] Y. Mahmood, N. Kama, A. Azmi, A. S. Khan, and M. Ali, "Software effort estimation accuracy prediction of machine learning techniques: A systematic performance evaluation," *Softw., Pract. Exper.*, vol. 52, no. 1, pp. 39–65, Jan. 2022.
[33] M. A. Ramessur and S. D. Nagowah, "A predictive model to estimate effort in a sprint using machine learning techniques," *Int. J. Inf. Technol.*, vol. 13, no. 3, pp. 1101–1110, Jun. 2021, doi: 10.1007/s41870-021-00669-z.
[34] S. Shukla and S. Kumar, "Applicability of neural network based models for software effort estimation," in *Proc. IEEE World Congr. Services (SERVICES)*, Jul. 2019, pp. 339–342.
[35] S. Goyal and P. K. Bhatia, "A non-linear technique for effective software effort estimation using multi-layer perceptrons," in *Proc. Int. Conf. Mach. Learn., Big Data, Cloud Parallel Comput. (COMITCon)*, Feb. 2019, pp. 1–4.
[36] A. B. Nassif, M. Azzeh, L. F. Capretz, and D. Ho, "Neural network models for software development effort estimation: A comparative study," *Neural Comput. Appl.*, vol. 27, no. 8, pp. 2369–2381, Nov. 2016.
[37] E. Okewu, S. Misra, and F.-S. Lius, "Parameter tuning using adaptive moment estimation in deep learning neural networks," in *Proc. Int. Conf. Comput. Sci. Appl.*, 2020, pp. 261–272.
[38] G. Hutcheson, "Categorical explanatory variables," *J. Model. Manage.*, vol. 6, no. 2, Jul. 2011.
[39] M. K. Dahouda and I. Joe, "A deep-learned embedding technique for categorical features encoding," *IEEE Access*, vol. 9, pp. 114381–114391, 2021.
[40] S. Gnat, "Impact of categorical variables encoding on property mass valuation," *Proc. Comput. Sci.*, vol. 192, pp. 3542–3550, Jan. 2021.
[41] M. Shepperd and S. MacDonell, "Evaluating prediction systems in software project estimation," *Inf. Softw. Technol.*, vol. 54, no. 8, pp. 820–827, Aug. 2012.
[42] S. D. Conte, H. E. Dunsmore, and Y. E. Shen, *Software Engineering Metrics and Models*. San Francisco, CA, USA: Benjamin-Cummings, 1986.
[43] M. Azzeh, A. B. Nassif, and L. L. Minku, "An empirical evaluation of ensemble adjustment methods for analogy-based effort estimation," *J. Syst. Softw.*, vol. 103, pp. 36–52, May 2015.
[44] A. Idri, I. Abnane, and A. Abran, "Evaluating Pred(p) and standardized accuracy criteria in software development effort estimation," *J. Softw., Evol. Process*, vol. 30, no. 4, p. e1925, 2018.
[45] A. S. Hadi and S. Chatterjee, *Regression Analysis by Example*. Hoboken, NJ, USA: Wiley, 2015.
[46] A. K. Jain, J. Mao, and K. M. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, vol. 29, no. 3, pp. 31–44, Mar. 1996.
[47] M. M. Zahra, M. H. Essai, and A. R. Abd Ellah, "Performance functions alternatives of MSE for neural networks learning," *Int. J. Eng. Res. Technol.*, vol. 3, no. 1, pp. 967–970, 2014.
[48] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," 2018, *arXiv:1811.03378*.
[49] A. B. Nassif, M. Azzeh, A. Idri, and A. Abran, "Software development effort estimation using regression fuzzy models," *Comput. Intell. Neurosci.*, vol. 2019, pp. 1–17, Feb. 2019.
[50] C. Jones, *Software Estimating Rules of Thumb*. IEEE, 2007.
[51] P. J. Rousseeuw and M. Hubert, "Robust statistics for outlier detection," *Wiley Interdiscipl. Rev., Data Mining Knowl. Discovery*, vol. 1, no. 1, pp. 73–79, 2011.
[52] S. Gopal Krishna Patro and K. Kumar Sahu, "Normalization: A preprocessing stage," 2015, *arXiv:1503.06462*.
[53] G. Guo and D. Neagu, "Similarity-based classifier combination for decision making," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, vol. 1, Oct. 2005, pp. 176–181.
[54] T. Xia, R. Shu, X. Shen, and T. Menzies, "Sequential model optimization for software effort estimation," *IEEE Trans. Softw. Eng.*, vol. 48, no. 6, pp. 1994–2009, Jun. 2022.
[55] L. Taylor and G. Nitschke, "Improving deep learning with generic data augmentation," in *Proc. IEEE Symp. Comput. Intell. (SSCI)*, Nov. 2018, pp. 1542–1547, doi: 10.1109/SSCI.2018.8628742.
[56] M. Steininger, K. Kobs, P. Davidson, A. Krause, and A. Hotho, "Density-based weighting for imbalanced regression," *Mach. Learn.*, vol. 110, no. 8, pp. 2187–2211, Aug. 2021, doi: 10.1007/s10994-021-06023-5.
[57] G.-W. Weber, İ. Batmaz, G. Köksal, P. Taylan, and F. Yerlikaya-Özkurt, "CMARS: A new contribution to nonparametric regression with multivariate adaptive regression splines supported by continuous optimization," *Inverse Problems Sci. Eng.*, vol. 20, no. 3, pp. 371–400, Apr. 2012, doi: 10.1080/17415977.2011.624770.
[58] F. Yerlikaya, "A new contribution to nonlinear robust regression and classification with MARS and its applications to data mining for quality control in manufacturing," M.S. thesis, Middle East Tech. Univ., 2008.
[59] F. Yerlikaya-Özkurt and İ. Batmaz, "A computational approach to nonparametric regression: Bootstrapping CMARS method," *Mach. Learn.*, vol. 101, nos. 1–3, pp. 211–230, Oct. 2015, doi: 10.1007/s10994-015-5502-3.

HUYNH THAI HOC was born in Tra Vinh, Vietnam, in 1980. He received the B.S. degree in mathematics and computer science from the University of Science (HCMUS), Vietnam, in 2002, and the M.S. degree in geographic information system from the University of Technology (HCMUT), Vietnam, in 2007. He is currently pursuing the Ph.D. degree in software engineering with Tomas Bata University in Zlín, Czech Republic.

He worked as a GIS Developer at the DITAGIS, HCMUT, from 2002 to 2007. From 2007 to 2014, he was a Lecturer with the Faculty of Information Technology, University of Natural Resources and Environment (HCMUNRE). From 2011 to 2018, he was a Lecturer with the Faculty of Information Technology, Industrial University of Ho Chi Minh City (IUH), Vietnam. From 2018 to 2019, he was a Lecturer with the Faculty of Information Technology, School of Engineering and Technology, Van Lang University, Ho Chi Minh City, Vietnam. His research interests include software effort estimation and data science.

RADEK SILHAVY was born in Vsetin, in 1980. He received the B.Sc., M.Sc., and Ph.D. degrees in engineering informatics from the Faculty of Applied Informatics, Tomas Bata University in Zlín, in 2004, 2006, and 2009, respectively.

He is currently an Associate Professor and a Researcher with the Department of Computer and Communication Systems. His major research interests include effort estimation in software engineering and empirical methods in software and system engineering.

ZDENKA PROKOPOVA was born in Rimavska Sobota, Slovakia, in 1965. She received the master's degree in automatic control theory and the Ph.D. degree in technical cybernetics from Slovak Technical University, in 1988 and 1993, respectively.

She worked as an Assistant at Slovak Technical University, from 1988 to 1993. From 1993 to 1995, she worked as a programmer of database systems in the data-lock business firm. From 1995 to 2000, she worked as a Lecturer at the Brno University of Technology. Since 2001, she has been at the Faculty of Applied Informatics, Tomas Bata University in Zlín. She currently holds the position of an Associate Professor with the Department of Computer and Communication Systems. Her research interests include programming and applications of database systems, mathematical modeling, and computer simulation and the control of technological systems.

PETR SILHAVY was born in Vsetin, in 1980. He received the B.Sc., M.Sc., and Ph.D. degrees in engineering informatics from the Faculty of Applied Informatics, Tomas Bata University in Zlín, in 2004, 2006, and 2009, respectively.

From 1999 to 2018, he was appointed as a CTO in a company specialized on database systems development. He currently holds the position of an Associate Professor with Tomas Bata University in Zlín. His major research interests include software engineering, empirical software engineering, system engineering, data mining, and database systems.