Regression Models for Software Project Effort Estimation

Huynh Thai Hoc, Ph.D.

Doctoral Thesis Summary

Tomas Bata University in Zlín

Tomas Bata University in Zlín
Faculty of Applied Informatics

Doctoral Thesis Summary

Regression Models for Software Project Effort Estimation

Regresní modely pro odhad úsilí softwarového projektu

Author:	**Huynh Thai Hoc, Ph.D.**
Degree programme:	P3902-Engineering Informatics
Degree course:	3902V023-Software Engineering
Supervisor:	Assoc. Prof. Ing. Zdenka Prokopová, CSc.
Consulting Supervisor:	Assoc. Prof. Ing. Petr Šilhavý, Ph.D.
External examiners:	Prof. RNDr. Ing. Miloš Šeda, Ph.D.
	Assoc. Prof. Oldřich Trenz, Ph.D
	doc. Ing. Radek Matušů, Ph.D

Zlín, October 2023

Published by **Tomas Bata University in Zlín** in the Edition **Doctoral Thesis Summary.**
The publication was issued in the year 2023.

Key words in Czech: *Odhadování pracnosti vývoje softwarových projektů, metoda funkčních bodů, regresní modely, hluboké učení*

Key words: *Software development effort estimation, function points methods, regression models, deep learning, ensemble, deep learning with balancing dataset, transfer learning, LIME, SHAP.*

Full text of the doctoral thesis is available in the Library of TBU in Zlín.

ISBN 978-80-7678-221-1

ABSTRAKT

Odhad úsilí při vývoji softwaru, resp. odhad pracnosti vývoje softwarových projektů, hraje klíčovou roli v oblasti vývoje softwaru a má velký vliv na plánování projektů a přidělování zdrojů. Předkládaná práce přináší významné pokroky v oblasti odhadu úsilí při vývoji softwaru zavedením inovativních technik jako je tzv. přenosové učení (transfer learning) a analýzy datových souborů, s cílem zvýšit přesnost odhadu úsilí, konkrétně v rámci rozšíření metody funkčních bodů. Kromě toho jsou v předkládané práci zkoumané různé přístupy k identifikaci faktorů analýzy funkčních bodů a relevantních kategoriálních faktorů, které přispívají ke zlepšení odhadu úsilí, včetně vícenásobné lineární regrese, neuronových sítí atd.

Prostřednictvím rozsáhlé série experimentů autor práce identifikuje nové faktory ovlivňující odhad úsilí, což vede k přesnějším odhadům ve srovnání se základními modely. Dále je v práci popsaná aplikace technik LIME (Local Interpretable Model-agnostic Explanations) a SHAP (SHapley Additive exPlanations), které umožňují hlubší vhled do černé skříňky predikčních modelů.

Provedený výzkum byl zaměřen na hodnocení účinnosti předem natrénovaných modelů a návrh využití metod tzv. hlubokého učení (deep learning) v kombinaci se strategiemi pro vyvažování kategoriálních proměnných s cílem zlepšit odhad úsilí. Výsledky jasně ukazují, že zahrnutí relevantních faktorů a využití hlubokého učení, jakož i technik přenosového učení, výrazně zlepšuje odhad úsilí při vývoji softwaru. Toto zlepšení odhadu úsilí nabízí týmům zabývajícím se vývojem softwaru přesnější prostředky, což v konečném důsledku vede ke zlepšení plánování a řízení projektů.

Předkládaná práce celkově přispívá k teoretickým i praktickým aspektům odhadu úsilí tím, že poskytuje nové poznatky a inovativní strategie pro zvýšení přesnosti odhadu úsilí při vývoji softwarových projektů.

Key words in Czech: *Odhadování pracnosti vývoje softwarových projektů, metoda funkčních bodů, regresní modely, hluboké učení*

ABSTRACT

Effort estimation plays a crucial role in the domain of software development, employing an influence on project planning and resource allocation. This thesis advances the field of Software Development Effort Estimation (SDEE) by introducing novel transfer learning and dataset balancing techniques to enhance effort estimation accuracy, focusing on the function point analysis. It explores multiple linear regression, feedforward neural networks, and ensemble methods to identify factors affecting effort estimation.

Through a comprehensive series of experiments, this study uncovers new factors that significantly improve effort estimation, resulting in more precise estimates when compared to baseline models. Furthermore, it employs the application of Local Interpretable Model-agnostic Explanations (LIME) and SHapley Additive exPlanations (SHAP) techniques to provide deeper insights into the black-box of predictive models.

This research evaluates the effectiveness of pre-trained models and suggests using deep learning methods in combination with strategies for balancing categorical variables to enhance effort estimation. The results indicate that incorporating relevant factors and employing deep learning and transfer learning techniques enhances SDEE. This improvement in effort estimation offers software development teams a more accurate means of estimation, ultimately leading to improved project planning and management.

In summary, this thesis contributes to both theory and practice in effort estimation by offering innovative insights and strategies to boost accuracy.

Key words: *Software development effort estimation, function points methods, regression models, deep learning, ensemble, deep learning with balancing dataset, transfer learning, LIME, SHAP.*

Contents Of the Thesis

1. INTRODUCTION

1.1 Motivation

The motivation for this thesis is essential to provide the estimating field with a new approach to the effort estimation problem, which might supplement current practices. The following are the key drivers behind this motivation:

i) The absence of categorical variables might result in less effort estimation accuracy measured by traditional function point analysis (FPA) estimation methods. It is a measure based on function points and productivity rate. However, in the early stage of software project development, the productivity rate of that project might be unknown. In addition, the complexity of FPA weight metrics values might be affected by many factors. Suppose the complexity weights assigned to the components are not appropriately identified, it might lead to inaccurate effort estimation. Hence, to estimate the effort required for software development in the initial stages of FPA, the thesis considers estimating the effort by incorporating essential categorical variables such as Industry Sector and Relative Size alongside factors of FPA.

ii) The unavailability of pre-trained models for software effort estimation: In the context of SDEE, transfer learning could enhance estimation model precision by leveraging knowledge from analogous projects or domains [2]–[4], significantly decreasing the time and resources necessary for model training. Leveraging transfer learning might mitigate the challenge of limited data availability often faced in software estimation. However, despite its potential benefits, several studies have compared the performance of transfer learning with the deep learning approach regarding SDEE. However, they did not propose a pre-trained model [3], [4]. This issue promotes the thesis to build a pre-trained learning model by leveraging the advantages of transfer learning and comparing the performance of transfer learning to the deep learning approach in terms of SDEE.

iii) The existing models utilised for effort estimation remain unclear black-boxes covering their internal mechanisms. Consequently, understanding the rationale behind their predictions becomes a formidable challenge for scientists and practitioners. Advanced methodologies such as LIME and SHAP offer promising solutions to tackle this pressing limitation while emphasising the importance of result interpretation. By providing interpretable and transparent insights into the models' decision-making processes, these techniques delegate researchers to understand the influential factors and their complex within the software development context. Consequently, leveraging LIME and SHAP can significantly enhance the validity and trustworthiness of effort estimation models, leading to more informed and scientifically driven project management decisions.

1.2 Problem Statement

Among the various approaches to estimating software effort estimation, one common technique in the software industry is FPA. This method is advantageous as it estimates the size of the software. However, as mentioned in publication [5], it is essential to note that FPA has limitations. One significant drawback is that it relies on fixed complexity weight values established using data from IBM in the 1970s [1]. Given the technological progress and changes to the present year, these values have become outdated. Furthermore, due to the unique nature of each company, using these fixed values tends to result in less accurate estimates.

Therefore, this study will propose a group of factors to estimate effort estimation by incorporating categorical variables such as Industry Sector and Relative Size along with factors of FPA based on the International Software Benchmarking Standards Group (ISBSG) [6] as the historical dataset. The first study uses various approaches such as regression model, random forest, ensemble approach, and deep learning based on multilayer perceptron (DLMLP) [7] to determine the factors of FPA incorporate with Industry Sector and Relative Size lead to more accurate effort estimation. In addition, the effectiveness of these models will be further explored by employing balanced datasets in the DLMLP model to address the limitation of imbalanced data, a common issue in effort estimation research. The thesis might hardly examine all known algorithms and all combinations of factors of FPA. Therefore, selecting some experimental algorithms and combinations of factors are also matters of concern.

As highlighted in the motivation section, the advantages of transfer learning are substantial [2], [8]. It enhances estimation model accuracy by leveraging insights from related projects, significantly reducing the resources and time required for model training. This thesis proposes a transfer learning technique that effectively estimates effort using the ISBSG dataset. Simultaneously, the thesis evaluates the applicability of this technique across similar datasets such as Albrecht, China. Additionally, an endeavour is undertaken to construct a pre-trained model obtained from the ISBSG dataset, intended as a reusable library for researchers.

On the other hand, several machine-learning approaches were adopted to increase effort estimation accuracy[9]–[12]. However, the resultant models remain a mystery. The comprehensive comparison of these influential factors within the predictions holds critical importance, as it might give researchers invaluable insights grounded in the predictions. As illustrated in the motivation section, this research endeavour extensively analyses predicted efforts via LIME [13] and SHAP [14]. Specifically, the focus lies on dissecting LIME and SHAP within the DLMLP model to illuminate how factors impact effort estimation. Due to limited time, this study concentrates on DLMLP models, prioritising a comprehensive exploration within this confined scope.

In summary, this study tackles software effort estimation challenges by combining FPA factors with categorical variables. It employs diverse methodologies to find the most accurate estimation approach. Transfer learning is utilised on the ISBSG dataset and evaluated on other datasets. The study also uses LIME and SHAP techniques for model analysis.

1.3 Research Questions and Hypothesis

In this thesis, five RQs and hypothesis must be answered:

1. **RQ1**: Which yields greater accuracy in software effort estimation: DLMLP, MLR, or Random Forest?
   $\mu DLMLP =$ Mean accuracy of DLMLP
   μMLR = Mean accuracy of MLR
   $\mu RF =$ Mean accuracy of Random Forest
   - H$_1$: $\mu DLMLP > \mu MLR$ and $\mu DLMLP > \mu RF$
     This hypothesis states that DLMLP accuracy is greater than MLR and RF in software effort estimation.
   - The null hypothesis H_0: $\mu DLMLP \leq \mu MLR \; or \; \mu DLMLP \leq \mu RF$
     This hypothesis states that DLMLP is either less or equally accurate as at least one of the other two methods in software effort estimation.
2. **RQ2**: Does dataset balancing enhance the predictive accuracy of DLMLP methods in software effort estimation?
   $\mu DLMLPB =$ Mean accuracy of DLMLP with Balancing
   $\mu DLMLP =$ Mean accuracy of DLMLP without Balancing
   - H$_1$: $\mu DLMLPB > \mu DLMLP$
     This hypothesis states that the accuracy of deep learning with a balancing dataset is more significant than without balancing.
   - The null hypothesis H$_0$: $\mu DLMLPB \leq \mu DLMLP$
     This hypothesis states that deep learning with a balancing dataset is either less or equally accurate as deep learning without balancing.
3. **RQ3**: For software effort estimation, does a combined ensemble of MLR, Random Forest, and DLMLP outperform each standalone model?
   $\mu ENS =$ Mean accuracy of Ensemble
   - H$_1$: $\mu ENS > \mu DLMLP$, $\mu ENS > \mu MLR$, and $\mu ENS > \mu RF$
     This hypothesis states that the accuracy of the ensemble is greater than DLMLP, MLR, and RF in software effort estimation.
   - The null hypothesis H_0:
     $$\mu ENS \leq \mu DLMLP, or \; \mu ENS \leq \mu MLR, or \; \mu ENS \leq \mu RF$$
     This hypothesis states that the ensemble is either less or equally accurate as at least one of the other three methods.
4. **RQ4**: Does transfer learning offer any accuracy advantages over conventional DLMLP approaches in software effort estimation?
   $\mu TL =$ Mean accuracy of Transfer Learning
   - H$_1$: $\mu TL > \mu DLMLP$

This hypothesis states that the accuracy of deep learning by applying transfer learning is more significant than that of deep learning without applying transfer learning.

- The null hypothesis H_0: $\mu TL \leq \mu DLMLP$
  This hypothesis states that deep learning by applying transfer learning is either less or equally accurate as deep learning without applying transfer learning.

5. **RQ5**: Do the categorical variables (IS and RS) influence effort estimation accuracy?
   $\beta IS =$ Regression coefficient for IS
   $\gamma RS =$ Regression coefficient for RS
   - **Null Hypothesis (H_0)**: $\beta IS = \gamma RS = 0$ (indicating that IS and RS do not affect the accuracy of effort estimation)
   - **Alternative Hypothesis (H_1)**: βIS or γRS is not equal to 0 (indicating RS or IS effect on the accuracy of effort estimation)

1.4 Objectives of the Thesis

This section outlines the objectives pursued in this research, focusing on advancing the state-of-the-art in the identified issues. The specific research objectives present in this thesis can be summarised as follows:

1. To enhance the accuracy of effort estimation in terms of FPA.
2. To evaluate the efficacy of various estimation methodologies, such as multiple linear regression, random forest, deep learning based on multilayer perceptrons, deep learning with balanced datasets; ensemble techniques established by incorporating multiple linear regression, random forest, and deep learning models; transfer learning for effort estimation. This evaluation involves validating the results using appropriate datasets.
3. To introduce a pre-trained model based on the ISBSG dataset, providing a comprehensive and reliable foundation for effort estimations. The relevant other datasets illustrate the performance of effort estimation based on the pre-trained model.
4. To leverage advanced techniques such as LIME and SHAP to gain comprehensive insights into the contribution and local importance of different features, namely EI, EO, EQ, EIF, ILF, IS, and RS, within the proposed effort estimation models in terms of FPA.

Thus, the research objective of this thesis is to establish innovative approaches for estimating the effort required in software product development. These approaches will be rigorously compared with the performance of effort estimation based on the ISBSG dataset and other relevant datasets, facilitating the identification of superior estimation techniques with practical applicability.

2. METHODOLOGY

2.1 Function Point Analysis

The FPA has the most characteristics that can be applied to estimate software projects in their initial stages [15]. First, function points can be fully allotted based on the requirements or design standards. The projects are in their initial phases. Second, they have nothing to do with language programming, specialist development tools, or data processing in general [16]. Furthermore, because the function points are built from the user's point of view, non-technical users of the software may find them easier to grasp [17].

A linear combination of size attributes with appropriate three degrees of complexity weights is built to count function points. This function count is also known as UFP. The UFP formula is shown in equation (1).

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

where BCs_{ij} is the count of component i at level j, and CWs_{ij} is an appropriate complexity weight. VAF is determined by assessing 14 General Systems Characteristics (GSCs), which represent operational aspects of the application process. These GSCs are constraints for non-technical users and have associated descriptions for calculating their impact. The VAF formula is as follows:

$$VAF = 0.65 + 0.01 \times \sum_{i=1}^{14} F_i \times Degree_{Influence} \tag{2}$$

where F_i represents the GSC factor's effect. The AFP can be calculated using the following equation:

$$AFP = UFP \times VAF \tag{3}$$

IFPUG-FPA [18] is widely used for calculating software's functional size and complexity based on user requirements. AFP can be used as an input to estimate the effort. The efforts in terms of IFPUG-FPA will be measured as follow:

$$Effort_{\mathrm{IFPUG-FPA}} = AFP \times PDR \tag{4}$$

2.2 Preprocessing Techniques

2.2.1 ISBSG Dataset

The ISBSG dataset includes various attributes, such as Project Rating, Development Type, Productivity, Industry Sector, Relative Size, and more. In order to ensure that this dataset offers high-quality data valuable for training models, it should be filtered based on the following criteria:

- The Project Rating field is designated with an ISBSG rating code of A, B, C, or D. As mentioned in ISBSG and several publications, the study chose high-quality projects by exclusively considering data projects with A and B ratings. This action led to the number of projects being reduced to 8,619.
- EI, EO, EQ, ILF, ELF, and industry sector, relative size; we have excluded all those not counted, resulting in 1,654.
- Productivity rate values (PDR) that fall outside of the Q1 (first quartile) - 1.5 × IQR to Q3 (third quartile) + 1.5 × IQR range may be eliminated, where IQR is the abbreviation of the InterQuartile Range. As a result, the final number of projects is 1,073 projects.
- In addition, IFPUG counting methods are crucial to this study. Out of 1,073 projects, 1045 are in the IFPUG category (Dataset 1), the main focus of the thesis. The rest are in the NESMA category (Dataset 2), used to assess transfer learning effectiveness.

2.2.2 Other Datasets

The study will expand its analysis to incorporate other datasets, including Desharnais, Albrecht, Kitchenham, and China. A Pearson correlation analysis is conducted on those datasets to identify the key features significantly influencing the actual/effort values. As a results, in the case of Desharnais, the attributes of Length, Transactions, Entities, and PointsAdjust exhibit a positive impact on the effort required. Given the high correlation coefficients observed, particularly with PointsAjust, it was determined that PointsNonAdjust provides redundant information and, therefore, has been excluded from further analysis. For Albrecht, the attributes of Input, Output, Inquiry, File, RawFPCount, and AdjFP are found to affect the effort estimation significantly. Furthermore, in the context of Kitchenham, the duration, AFP, and Estimate attributes hold considerable importance, while for China, the attributes of AFP, Input, Output, Enquiry, File, and Added positively influence the actual development efforts. These findings provide valuable guidance for accurately estimating software effort by considering the influential attributes in each dataset.

2.2.3 Balancing Dataset Technique

The ISBSG dataset encompasses the industry sector feature, which is crucial in the analysis. The class weighting approach is utilised specifically for the industry sector feature to achieve this. By assigning appropriate weights to each category within the industry sector, the deep learning model might effectively account for the inherent class imbalance, leading to more accurate and reliable predictions across different industry sectors. The following diagram of this approach is given in Figure 2-1. Dataset 1 serves as the historical dataset employed in this methodology.

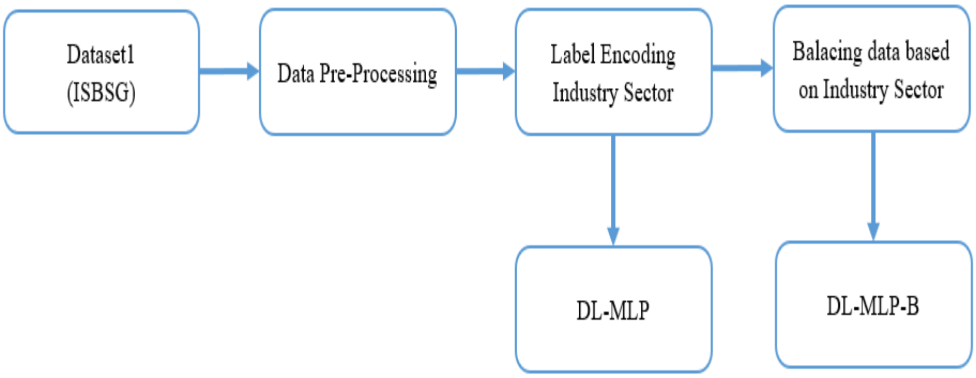


Figure 2-1: The architecture of the DLMLP model with/without balancing based on industry sector factors

2.3 Model Development

This section presents the model development and the thesis study models based on multiple linear regression, deep learning, transfer learning, deep learning with balancing datasets, and ensemble model, which incorporates multiple linear regression and deep learning.

2.3.1 Multiple Linear Regression Model

The MLR technique is employed for statistical analysis to establish the connection between a dependent and two or more independent variables. Multiple regression aims to predict the dependent variable's value based on the independent variables' value [19]. In a multiple regression model, the dependent variable is commonly denoted as the response or outcome variable, whereas the independent variables are termed predictor variables or covariates. It might be used to predict software effort estimation based on a given set of independent variables. The formation of MLR is written as a linear equation between a dependent variable and a bunch of p independent variables $X_1, X_2, \dots, X_p$ as follow:

$$y \approx \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \varepsilon \qquad (5)$$

where y is the response variable, it stands for the output of the model; $X_1, X_2, \dots, X_p$ are predictors or independent variables; β_0 is an intercept, $\beta_1, \beta_2, \dots, \beta_p$ are regression coefficients, and ε is presented as an error residual. The intercept and regression coefficients are unknown values. The regression model estimates these coefficients based on the observed data, and the goal is to find the values of the coefficients that best fit the data.

2.3.2 Random Forest

Random Forest (RF), introduced in 2001 by Breiman [20], is a kind of ensemble of decision trees trained via the bagging method (or sometimes the pasting method). Several poor models are joined to build a superior model. Each tree categorises the attributes of a new entity. The forest chooses the category with the most votes and averages the outputs of the different trees. The growth process of each tree in a random forest can be summarised as follows:

- Sampling: N cases are randomly selected from the original data with replacements to form the training set for each tree. The number of cases in the training set is equal to N.
- Variable Selection: At each tree node, a subset of m variables is chosen randomly from the total M input variables. The value of m is much smaller than M. The node is then split based on the best split determined using the selected m variables.
- Constant Variable Selection: Throughout growing the random forest, the value of m remains constant for all the trees.
- Maximum Growth: Each tree is grown to its fullest extent without the use of any pruning techniques.

2.3.3 Gradient Boosting

This approach involves sequentially fitting new models to improve the accuracy of the response variable. The idea is to construct new base learners most similar to the negative gradient of the loss function, which connects to the entire ensemble. This learning process minimises the traditional squared error loss function by iteratively fitting errors. Extreme gradient boosting (XGBoost) and Histogram Gradient Boosting (HGBoost) are both implementations of gradient boosting, a machine-learning technique employed for predictive modelling.

XGBoost is a widely used gradient-boosting implementation incorporating a gradient-boosting framework with various optimisations to enhance speed and precision. This algorithm is an ensemble of gradient boosting that takes advantage of second-order derivatives of the loss function to identify the most efficient and precise base classifier [21], [22]. Unlike traditional gradient boosting, XGBoost employs second-order gradients.

2.3.4 Multilayer Perceptron Model

The MLP architecture comprises an input, output, and one or more hidden layers. The Input layer receives input data, which is subsequently propagated through the hidden layers to generate the output. Figure 2-2 shows the diagram of one hidden layer of MLP. The input layer, located on the left most layer, comprises a group of neuron features (X) that represent the input features. Every neuron in the hidden layer participates in a weighted linear summation $x_1w_1 + x_2w_2 + x_3w_3 + \cdots + x_mw_m$ to the values from the previous layer, followed by an activation function. The activation function used in each neuron can vary, but common choices include the sigmoid function, ReLU (rectified linear unit), and tanh (hyperbolic tangent) function [23]. The values propagated from the preceding hidden layer are accepted by the output layer and transformed to produce output values.

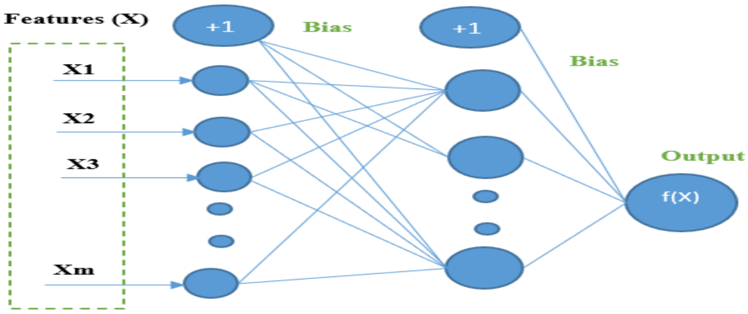


Figure 2-2: The diagram of one hidden layer of MLP

2.3.5 Transfer Learning Technique

This thesis uses Dataset 1 as the source and Dataset 2, Albrecht, and China as targets. Dataset1 and Dataset 2 share the same input and output features, whereas the remaining datasets exhibit similarities in their features but have fewer input features than Dataset 1. Dataset 1 comprises a significantly more extensive set of 1045 projects in comparison to Dataset 2, which consists of only 28 projects. Additionally, when contrasting Dataset 1 with other datasets such as Albrecht and China, it becomes evident that Dataset 1 is more significant than the others.

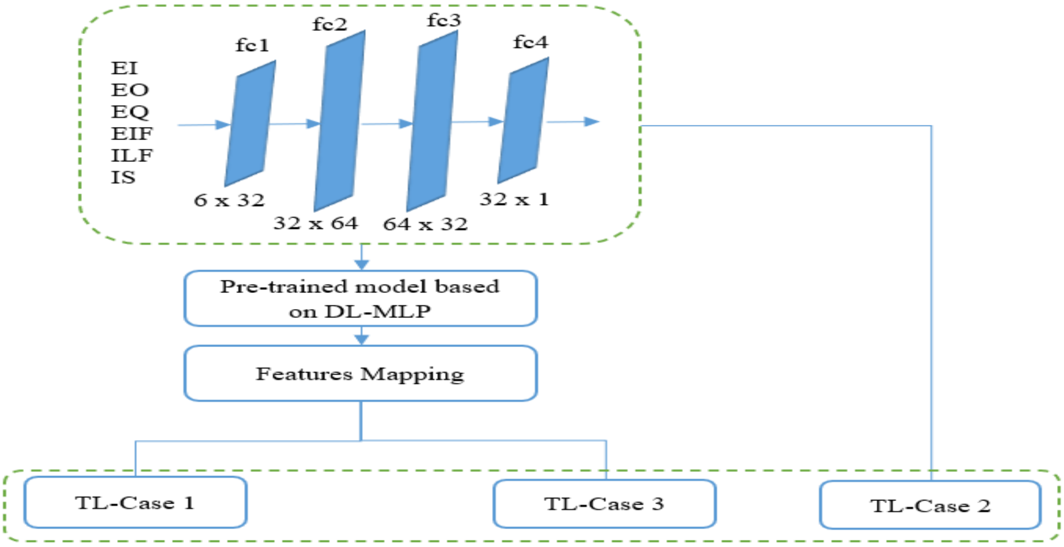


Figure 2-3: The diagram of the transfer learning model

Figure 2-3 illustrates the diagram of transfer learning models, where Dataset 1 is used as an extensive dataset to build the pre-trained model, and Dataset 2, Albrecht, China, are used to clarify the performance of transfer learning models. Features mapping involves translating the characteristics of the new dataset into a format that the pre-trained model might understand. The pre-trained model was initially designed to work with six input features (EI, EO, EQ, EIF, ILF, and Industry Sector). Those features were chosen based on the best performance of effort estimation obtained from those features presented in 4.2.1. This step updates the pre-trained model's input layer to match the new input's size. The scenario is defined into three cases as described below:

- TL-Case 1: Using DLMLP models trained based on Dataset 1 to validate the performance of effort estimation based on a testing dataset of Dataset2.

- TL-Case 2/DLMLP: Using DLMLP models, train them based on 80% of Albrecht, China, Dataset 2 and validate the performance of effort estimation based on 20% of those datasets.
- TL-Case 3: This is a transfer learning approach. DLMLP models trained by Dataset 1 are called pre-trained models and continue to train based on 80% of Albrecht, China, and Dataset 2 and validate the performance of effort estimation based on 20% of the remaining datasets.

2.3.6 Ensemble Model: Incorporating Multiple Linear Regression, Random Forest, and Deep Learning Models

The idea behind ensemble learning is that by incorporating the predictions of multiple models, the variance and bias of the overall model might decline, leading to better performance on unseen data. In 1990, Hansen et al. [24] proposed that utilising an ensemble of neural networks with a majority agreement technique could produce better results than using a single predictor. In this context, an ensemble refers to a group of predictors, and ensemble learning is a method that integrates predictions from multiple models, referred to as the ensemble method.

This thesis selects two base regression models, MLR and RF, for creating ensemble models through stacking regressors. These ensemble models produce predictions, which are then integrated with the output of a DLMLP model using a voting-by-averaging method for regression. The performance of this approach is evaluated with MLR, RF, and DLMLP approaches.

2.4 Model Explainability - Interpretability

Explainability techniques, such as LIME and SHAP, become essential to address this issue. These techniques are pivotal in bridging this gap by unveiling the intricate relationships between these input features and the predicted effort. Doing so gives stakeholders a transparent view of the estimation process, enabling them to understand better the underlying factors influencing model predictions. This transparency enhances the estimation model's credibility and empowers decision-makers to make informed decisions regarding software project planning and resource allocation.

2.4.1 LIME

Applying LIME in the context of effort estimation assists in illuminating how each feature (e.g., EI, EO, EQ) contributed to the predicted effort for a specific instance. LIME dissects the contributing factors underlying a prediction, facilitating an in-depth understanding of the role played by each feature in the model's decision-making process. The positive and negative values associated with the feature indicate their impact on the predicted effort.

2.4.2 SHAP

SHAP provides a unified approach to attribute the contribution of each feature to the predicted effort estimation. It assigns a value to each feature, representing its impact on the prediction in the context of the other features. These values are called SHAP values.

- Positive SHAP Value: A positive SHAP value for an independent variable signifies that the presence or increase in that variable contributes positively to the predicted effort. Higher values or complexity for EI, EO, EQ, etc., features are associated with increased effort.
- Negative SHAP Value: On the other hand, a negative SHAP value for a variable suggests that the presence or increase in that variable contributes negatively to the predicted effort. For features such as EIF and ILF, negative SHAP values indicate that higher values or complexity in these variables are associated with decreased effort in the effort estimation model.

3. EXPERIMENTS

3.1 Conceptual Framework of the Study

3.1.1 The Framework of the Study

As shown in Figure 3-1, there are four primary phases, including collecting the datasets, data preprocessing, building the proposed models, and measuring the performance of proposed models based on performance metrics.

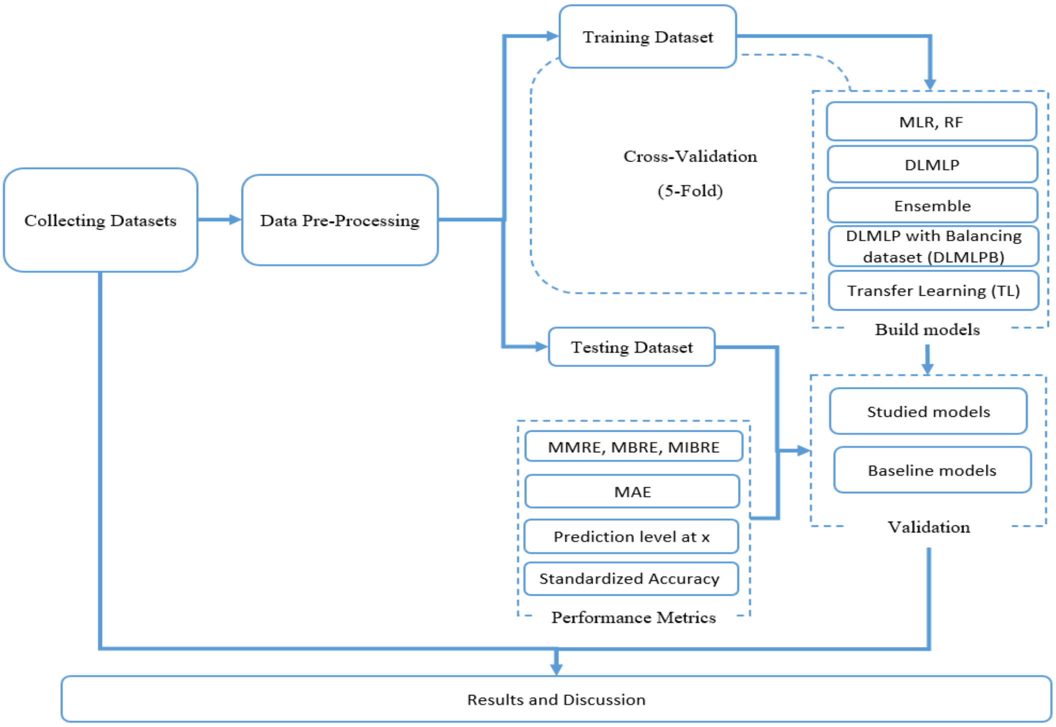


Figure 3-1: The flow diagram of the proposed software effort estimation

Next, data preprocessing is vital in preparing and refining the raw datasets before training models. The process of data preprocessing and the results of this process are illustrated in Section 2.2. The following presents the steps of data preprocessing might summarised as:

- ISBSG dataset: The primary dataset employed in this study. This dataset includes factors relevant to FPA, such as EI, EO, EQ, EIF, ILF, AFP, and a range of categorical variables. The central focus of this thesis is to investigate the impact of categorical variables in conjunction with FPA factors on effort estimation. Due to time constraints, the study narrows its scope to six key predictors denoted as P1, P2, P3, P4, P5, and P6 (see Section 3.1.2). The data preprocessing for the ISBSG dataset is presented in Section 2.2.1. The study adheres to the IFPUG approach, creating two distinct datasets: Dataset 1 and Dataset 2. Dataset 1 is selected based on IFPUG criteria, while the remaining projects are allocated to Dataset 2.
- Other datasets: In alignment with the research objectives for the Albrecht, Desharnais, Kitchenham, and China datasets, Pearson correlation analysis was conducted (see Section 2.2.2). This analysis aims to identify the key features significantly influencing actual/effort values in the context of software development projects.

Following that, the thesis studies proposed models, MLR (see Section 3.2), RF (see Section 3.3), and DLMLP (see Section 3.4). Further study is conducted on the proposed models, including the ensemble (see Section 3.7), deep learning with balancing dataset (see Section 3.6) and deep learning with transfer learning (see Section 3.5). Ensemble models might combine multiple models, including MLR, RF, and DLMLP, to achieve better performance, while deep learning with transfer learning might leverage pre-trained models to improve learning efficiency and accuracy. Those models are trained based on the training datasets.

The study designs eleven predictors from P1 to P6, P_A, P_D, P_C, P_K, and $P_{Dataset2}$ (see Section 3.1.2). For models that adopted predictors P1 to P6, they use training Dataset 1. The models adopted predictor P_A, P_D, P_C, P_K, and $P_{Dataset2}$ use training datasets of Albrecht, China, Kitchenham, and Dataset 2, respectively. The cross-validation with 5-fold is employed for all studied models in the training process.

The details of predictors and the whole configuration of proposed models are shown in the following sections.

3.1.2 Predictors

In this study, the predictors, including AFP, EI, EO, EQ, EIF, ILF, RS, and IS, are categorised into six groups, with each group comprising different combinations of techniques as follows:

- P1: AFP
- P2: EI, EO, EQ, EIF, ILF
- P3: AFP, IS
- P4: EI, EO, EQ, EIF, ILF, IS

- P5: AFP, IS, RS
- P6: EI, EO, EQ, EIF, ILF, IS, RS
- $P_{Dataset2}$: the predictor based on EI, EO, EQ, EIF, ILF, IS, RS and Dataset 2.

Setting one of the lists (from P1 to P6) as independent variables and SWE is the dependent variable. The aim of using these predictors is to find the most appropriate combination that leads to the highest performance in effort estimation.

According to the findings of the Pearson correlation of features on the Promise repository in section 2.2.2, the following factors are chosen as predictors for Desharnais (P_D), Albrecht (P_A), Kitchenham (P_K), and China (P_C):

- P_D: Length, Transactions, Entities, and PointsAjust
- P_A: Input (EI), Output (EO), Enquiry (EQ), File (EIF)
- P_K: duration, AFP, estimate
- P_C: Input (EI), Output (EO), Enquiry (EQ), File (EIF)

3.2 Regression Experiment

The MLR algorithm is implemented using the robust linear regression algorithm available in the Scikit-learn library, a widely used machine learning library. The training dataset is divided into five folds using the K-Fold function [25] to ensure robust evaluation and minimise bias. The shuffle and random_state parameters are incorporated during the data shuffling process to introduce randomness and ensure reproducibility. The shuffle parameter randomly reorders the data, while the random_state parameter sets a fixed seed, guaranteeing that the results can be replicated for further analysis.

Next, for each fold generated by the 5-fold technique, the data is further divided into training and validation sets, leveraging the indices provided by the splitting process. An MLR is created using the LinearRegression() function with default parameters (such as fit_intercept set to be True). The code then trains the model on the training set using the fit() function and uses it to predict the target variable on the validation set using the predict() function.

3.3 Random Forest Experiment

RF is a popular machine learning algorithm used for classification and regression tasks. It might be used for predicting both numerical and categorical variables. It might be computationally efficient and is capable of handling large datasets. RF is designed to mitigate overfitting by combining multiple decision trees and using random subsets of data and features. Having sufficient trees in the ensemble might help reduce the overfitting risk and improve generalisation performance.

Table 3-1: The experimental-based parameters of RF

No	Parameters	Values
1	n_estimators	{120,150, 180, 210}

2	max_depth	{5, 10, 15, 20, 25, 30}
3	random_state	42
4	min_samples_split	Default value
5	min_samples_leaf	Default value
6	max_features	Default value
7	Num folds	5

The experimental parameters for RF, as outlined in Table 3-1, with 'n_estimators' and 'max_depth' organised into distinct value sets; 'random_state' is set to 42; while leaving 'min_samples_split', 'min_samples_leaf', and 'max_features' at their default values. The experimentation is conducted within a cross-validation with 5-fold. Model performance was evaluated using the MAE, and the optimal model configuration is determined based on achieving the minimum MAE through the grid search [26] process.

3.4 DLMLP Experiment

In DLMLP, each neuron receives input from the previous/input layer. A neuron produces an output sent to the next layer after applying an activation function to the weighted sum of its inputs. According to Jason Brownlee [23], ReLU is simple to compute and requires few computational resources. ReLU addresses the issue of vanishing gradients, which might impede learning in deep neural networks. By allowing for faster learning and improved performance [27], ReLU has become a popular choice in many deep-learning applications.

Optimisation algorithms are essential for training deep learning models [28]. Optimisation aims to find the best set of parameters for a model that minimises the loss function, which is the difference between the model's predicted and actual output. Root Mean Squared Propagation (RMSProp) and Adam make adaptive moment estimations to enhance results among Adam, RMSProp, Adaptive Gradient Algorithm and a more robust extension of Adagrad [29]. As a result, the study chose Adam as the optimiser of this model.

This study conducted experiments to evaluate different layer configurations for the DLMLP, encompassing architectures with 2, 3, 4, and 5 layers. The optimal architecture of models is identified using the grid search [26] based on the minimum MAE, early stopping with a monitoring criterion based on the minimum MAE also installed. The configuration details provided in Table 3-2 further illuminate the design and training aspects of the deep learning model.

Table 3-2: The experimental-based parameters of DLMLP

No	Parameters	Values
1	Learning rate	{0.01,0.001}
2	Batch size	64
3	Epoch	1260

4	Rate decay	0.999
5	Num fold	5
6	Early stop	True, patience = 5
7	Loss function	Cross Entropy Loss

3.5 Transfer Learning Experiment

As mentioned in 2.2.1, Dataset 1 is a widely recognised and standardised dataset containing many software development projects. This thesis chooses the best model built based on Dataset 1 as the pre-trained model (called the ISBSG model). It is trained on a large amount of data, which allows us to learn generalisable features that might be applied to other datasets.

Table 3-2 presents the input and output variables list among studied datasets (Dataset 2, Albrecht, and China). Firstly, considering the output feature, those datasets share the same target variable ('effort'). Secondly, observation of input variables, those datasets include inputs related to EI, EO, and EQ. They also have input features related to file counts, such as EIF and ILF. Based on these similarities, there is indeed an overlap between Dataset 1 and the studied datasets (Dataset 2, Albrecht, and China) in terms of input variables and output variables. This overlap suggests there is potential for transferring knowledge from the pre-trained model based on Dataset 1 to predicting effort in other datasets.

Table 3-3: The input and output features list among studied datasets

No	Intersect Dataset 1 with	Similarity		Overlap with Dataset 1?
		Inputs	Output	
2	Dataset2	EI, EO, EQ, EIF, ILF, IS	SWE	Yes
3	Albrecht	Input (EI), Output (EO), Inquiry (EQ), File (EIF)	Effort	Yes
4	China	Input (EI), Output (EO), Inquiry (EQ), File (EIF)	Effort	Yes

Transfer learning based on the pre-trained model involves several steps as follows:

- **Step 1**: Choose the best model obtained from DLMLP based on P1 to P6 as the pre-trained model. As discussed in Section 4.2.1, DLMLP-P4, with six predictors EI, EO, EQ, EIF, ILF, and IS, outperform compared with P1, P2, P3, P5, and P6.
- **Step 2**: Load the pre-trained model: This model is trained on Dataset 1, and then choose the best-proposed model to use as the pre-trained model.
- **Step 3:** Feature mapping as following steps:
  - Extract features using the pre-trained model.
  - Map the features between new input features and the features from the pre-trained model.
- **Step 4**: Freeze all layers except for the last one.

- **Step 5**: Create a new optimiser: A new optimiser is created specifically for the last layer of the model, which was set to require gradients in the previous step. Adam optimiser is chosen as the optimiser (the same optimiser as the pre-trained model).
- **Step 6**: Continue training the model.

3.6 Balancing Dataset Experiment

The number of projects in each industry sector might need to be balanced to address this imbalance. In practice, balancing may involve adding more data to underrepresented groups or removing data from overrepresented groups until the number of data points in each group is approximately equal.

Determining class weights is a critical step to address the class imbalance issue in the dataset. Based on the experiment, a class weighting approach is employed to assign different weights to each industry sector based on the number of projects within each sector. The primary objective is to give more importance to underrepresented sectors while training the model. The class weights are determined as follows:

- For each Industry Sector, the ratio of the number of projects before balancing to the number of projects after balancing is calculated.
- The inverse of these ratios is used as class weights. The less-represented sectors are assigned higher weights, while the overrepresented sectors are assigned lower weights.
- The performance of the model is based on the minimum of MAE.

Finally, DLMLPB with a balanced dataset is applied. The configuration of this model is the same as DLMLP presented in Section 3.4.

3.7 Ensemble Model Experiment

The stacking ensemble (SE) for regression incorporates two distinct models: MLR and RF. Data preprocessing is conducted using a 'tree_preprocessor' object to prepare the dataset for modelling. This phase creates two separate pipelines, each tailored to one of the models—MLR and RF. These pipelines integrate preprocessing with the respective model, resulting in two well-defined modelling paths. To summarise, the ensemble model has several steps as follows:

- **Step 1:** Create base models (RF, MLR, DLMLP) using predefined hyperparameters for each model to generate individual predictions.
- **Step 2**: Configure an XGBoost regressor as the meta-model to merge base model predictions.
- **Step 3**: Set up the stacking ensemble, using base models with the final XGBoost model.
- **Step 4**: Employ 5-fold cross-validation to rigorously evaluate the ensemble's accuracy based on the minimum MAE.

- **Step 5**: Apply a voting mechanism that averages predictions from DLMLP and the stacking ensemble in step 5, producing a unified prediction that balances insights from both sources for enhanced accuracy.

3.8 Model Explainability Experiments

This section uses LIME and SHAP techniques to perform model explainability experiments. Due to time limitations, the thesis only analyses LIME and SHAP based on DLMLP.

Regarding LIME, the following steps are undertaken to derive and interpret explanations for individual predictions:

- **Step 1**: Begin by instantiating a LIME explainer using the LimeTabularExplainer library.
- **Step 2**: Select an example instance from the testing dataset (see Table 3-4).
- **Step 3**: Utilise the LIME explainer to generate an explanation for the selected instance.
- **Step 4**: The generated LIME explanation offers insights into the contribution of each feature to the prediction for the specific instance.

Regarding SHAP, these values are derived and interpreted as follows:

- **Step 1**: Conversion from LIME to SHAP. The LIME explanation obtained earlier is converted into a format compatible with SHAP, facilitating a broader perspective of feature importance.
- **Step 2**: A SHAP explainer is established using the SHAP.Explainer library. It takes as input the prediction function and the reshaped training data, enabling the computation of SHAP values.
- **Step 3**: SHAP values are computed for the entire test dataset using the SHAP explainer.
- **Step 4**: Visualise the representation and interpretation.

Table 3-4 presents a specific instance that is being used to demonstrate the application of LIME. This instance serves as an example for illustrating how LIME might be utilised to explain and interpret the relationship between the features and the actual effort in that project.

Table 3-4: The scenario of instance for illustrating LIME

Features	Instance	Specific Unit
EI	209	Function Points
EO	129	Function Points
EQ	24	Function Points
EIF	15	Function Points
ILF	83	Function Points
IS	Communication	Function Points
RS	M2	Function Points
Real Effort	10200	Person-Hours

3.9 Baseline Models

This section proposes three baseline models: one statistical model (stepwise-based regression), one simple artificial neural network (ANN) model from previous research, and IFPUG-FPA. IFPUG-FPA is introduced in Section 2.1. Three baseline models are used to compare the performance of the best model among MLR, RF, and DLMLP based on Dataset 1. The thesis employs a set of metrics, namely MMRE, MBRE, MIBRE, MAE, Pred(0.25), and SA, to evaluate the performance of the best model compared with the baseline models. It is worth noting that the same dataset used for validation by the best-performing model is also employed in assessing the baseline models.

3.9.1 ANN-based Model

A simple ANN-based model with two hidden layers has been employed as the baseline model. The purpose of choosing two hidden layers is to make the model simple and naive to determine the minimum performance that might be expected. If the best model does not perform significantly better than the baseline, it might be overfitting or not appropriately capturing the underly patterns. The parameters adopted in the ANN-based model are presented in Table 3-5.

Table 3-5: The parameters of a simple ANN-based model

No	Parameters	Values
1	Learning rate	0.01
2	Batch size	64
3	Epoch	100
4	Rate decay	0.999
6	Early stop	True, patience = 5
7	Loss function	Cross Entropy Loss

3.9.2 Stepwise-based Regression Model

Stepwise-based regression (SWR) [31] is a technique widely used in statistical modelling, drawing inspiration from previous publications [31]–[33]. This approach to multiple linear regression involves an automated process for selecting independent variables and might be summarised as follows:

- Initialisation: Begin with either a starting model containing predefined terms (backward selection) or a null model (forward selection).
- Model Complexity: Define the desired model complexity, specifying which terms should be included, such as linear, quadratic, or interaction terms.
- Evaluation threshold: Set an evaluation threshold based on the sum of residual errors. This threshold determines whether to add or remove features.
- Iterative Process: The algorithm iteratively adds or removes features while re-evaluating the model at each step.
- Termination: Stepwise regression continues until no further improvement in estimation is achievable based on threshold.

Forward selection initiates with a null model and progressively adds features that meet specific criteria. Conversely, backward selection starts with a full model and removes non-significant features. Consequently, SWR necessitates two significance levels: one for adding features and another for removing features.

4. RESULTS AND DISCUSSION

4.1 Comparison of Model Performance

Table 4-1 focuses on assessing effort estimation methods using Dataset 1. This table provides a detailed analysis of model performance metrics, including MMRE, MBRE, MIBRE, MAE, Pred(0.25)/Pred(0.30), and SA. The rows represent different models, including MLR, RF, DLMLP, the ensemble, and DLMLPB models. The performance evaluation in this table offers insight into the effectiveness of these methods when applied to Dataset 1.

Table 4-1: The performance of effort estimation obtained from MLR, RF, the ensemble, and DLMLPB based on testing of Dataset 1

Predictors /Models	MMRE	MBRE	MIBRE	MAE	PRED		SA
					0.25	0.30	
P1							
MLR	0.9113	1.0057	0.3831	2173.83	0.27	0.32	0.43
RF	0.6879	0.8047	0.3661	2150.85	0.28	0.32	0.46
DLMLP	0.6709	0.7719	0.3637	2066.69	0.29	0.34	0.51
Ensemble	0.5478	0.7229	0.3582	1986.74	0.29	0.34	0.52
DLMLPB	0.6228	0.7702	0.3606	2016.14	0.29	0.34	0.52
P2							
MLR	1.2273	1.3250	0.4134	2254.00	0.26	0.32	0.43
RF	0.9802	1.0675	0.3786	2118.20	0.30	0.38	0.46
DLMLP	0.5526	0.7360	0.3044	1768.61	0.46	0.53	0.58
Ensemble	0.4853	0.6132	0.2920	1669.18	0.46	0.54	0.61
DLMLPB	0.4568	0.5378	0.2874	1464.35	0.47	0.52	0.65
P3							
MLR	0.9081	1.0012	0.3824	2172.63	0.28	0.34	0.45
RF	0.6820	0.7964	0.3626	2123.92	0.32	0.35	0.46
DLMLP	0.6275	0.7812	0.3619	2033.24	0.32	0.36	0.51
Ensemble	0.5362	0.7464	0.3553	2024.51	0.34	0.36	0.52
DLMLPB	0.5639	0.6713	0.3356	1915.79	0.36	0.42	0.54
P4							
MLR	1.1999	1.2851	0.4090	2018.79	0.27	0.32	0.45
RF	0.9784	1.0579	0.3750	2012.14	0.31	0.38	0.47

DLMLP	0.2478	0.4311	0.1572	530.65	0.79	0.84	0.87
Ensemble	0.3119	0.3657	0.2189	1007.28	0.62	0.73	0.75
DLMLPB	0.1871	0.2064	0.1393	494.20	0.82	0.85	0.88
P5							
MLR	1.1551	1.0378	0.5949	2335.22	0.27	0.30	0.4
RF	0.6875	0.8028	0.3645	2145.08	0.32	0.35	0.45
DLMLP	0.6326	0.7830	0.3621	2118.93	0.32	0.36	0.49
Ensemble	0.6115	0.7281	0.3517	1983.02	0.31	0.36	0.52
DLMLPB	0.6855	0.7842	0.3567	2069.72	0.33	0.37	0.51
P6							
MLR	0.8981	1.0129	0.3967	2228.68	0.28	0.32	0.42
RF	0.7756	0.8632	0.3649	2029.29	0.30	0.39	0.48
DLMLP	0.3489	0.4750	0.2219	963.71	0.68	0.71	0.77
Ensemble	0.3599	0.4483	0.2479	1143.07	0.56	0.66	0.72
DLMLPB	0.2586	0.3551	0.1731	550.82	0.76	0.78	0.86

Table 4-2 expands the evaluation by examining the performance of effort estimation methods across a broad spectrum. In addition to Dataset 2, this table incorporates other datasets such as Desharnais, Albrecht, Kitchenham, and China datasets. The evaluation includes a comparison of MLR, RF, the ensemble, and transfer learning cases 1, 2, and 3 (TL-Case1, TL-Case2, TL-Case3, see in Section 2.3.5), along with an ensemble-based approach. As mentioned in Section 2.2.2, TL-Case1 and TL-Case3 for Desharnais and Kitchenham and TL_Case1 for Albrecht and China are not measured due to differences in input features. The metrics used for evaluation are consistent with those in Table 4-1. This comprehensive analysis allows us to assess the effectiveness of these techniques across diverse datasets.

Table 4-2: The performance of effort estimation obtained from MLR, RF, TL-Case1, TL-Case2, TL-Case3, and the ensemble based on testing of Dataset 2, Desharnais, Albrecht, Kitchenham and China datasets

Preditors/ Models	**MMRE**	**MBRE**	**MIBRE**	**MAE**	**PRED**		**SA**
					0.25	**0.30**	
P_D							
MLR	0.4202	0.5795	0.3340	2539.94	0.25	0.38	0.28
RF	0.3850	0.5431	0.2989	2514.43	0.44	0.50	0.29
TL-Case1	-	-	-	-	-	-	-
TL-Case2	0.2076	0.2507	0.1693	1333.22	0.68	0.75	0.65
Ensemble	0.2430	0.3397	0.2231	1860.73	0.50	0.75	0.51
TL-Case3	-	-	-	-	-	-	-
P_A							
MLR	3.2224	0.5479	3.1693	7.13	0.4	0.4	0.54

RF	1.8730	1.9115	0.3906	4.73	0.4	0.4	0.65
TL-Case1	-	-	-	-	-	-	-
TL-Case2	0.3201	0.3220	0.1950	1.44	0.6	0.6	0.84
Ensemble	0.8081	0.8397	0.3368	3.66	0.4	0.4	0.61
TL-Case3	0.1088	0.1839	0.1087	0.38	0.8	0.8	0.90
$\mathbf{P_K}$							
MLR	0.7583	0.5207	0.3193	589.35	0.42	0.43	0.64
RF	0.4716	0.4870	0.2364	471.52	0.50	0.57	0.71
TL-Case1	-	-	-	-	-	-	-
TL-Case2	0.2204	0.3413	0.1594	240.88	0.75	0.78	0.81
Ensemble	0.2276	0.2435	0.1644	262.80	0.75	0.78	0.81
TL-Case3	-	-	-	-	-	-	-
$\mathbf{P_C}$							
MLR	1.6664	1.8916	0.4550	2762.83	0.27	0.28	0.25
RF	1.6386	1.8595	0.4507	2595.62	0.27	0.28	0.29
TL-Case1	-	-	-	-	-	-	-
TL-Case2	0.9833	1.0569	0.2659	1034.31	0.58	0.59	0.72
Ensemble	0.9626	1.0241	0.3235	1447.57	0.47	0.59	0.62
TL-Case3	0.2092	0.2325	0.1578	247.21	0.79	0.83	0.93
$\mathbf{P_{Dataset\ 2}}$							
MLR	0.6681	0.9829	0.1471	1162.98	0.33	0.34	0.00
RF	0.3031	0.3698	0.2571	677.45	0.66	0.67	0.23
TL-Case1	0.4951	1.0574	0.4539	1165.51	0.17	0.17	0.00
TL-Case2	0.2480	0.2557	0.1796	438.48	0.66	0.83	0.43
Ensemble	0.2182	0.2518	0.1872	482.28	0.66	0.67	0.38
TL-Case3	0.1884	0.2310	0.1731	463.10	0.66	0.83	0.40

Table 4-3 displays the evaluation results for effort estimation derived from three baseline models: ANN-based, SWR-based, and IFPUG-PFA. This table offers an in-depth examination of the performance metrics for these models, primarily focusing on the testing of Dataset 1. The assessment of model performance encompasses the analysis of six predictors, denoted as P1 to P6.

Table 4-3: The performance of effort estimation obtained from baseline models (ANN, SWR, IFPUG) based on Dataset 1

Predictors /Models	**MMRE**	**MBRE**	**MIBRE**	**MAE**	**PRED**		**SA**
					0.25	**0.30**	
P1							
ANN	0.6760	0.7769	0.3592	2067s.64	0.26	0.28	0.47
SWR	1.5740	1.6748	0.4380	2373.19	0.22	0.28	0.39
P2							
ANN	0.6081	0.8216	0.3056	1786.68	0.41	0.45	0.54

SWR	1.1787	1.2716	0.4075	2177.96	0.24	0.30	0.44
P3							
ANN	0.6929	0.9452	0.3659	2096.01	0.29	0.35	0.46
SWR	1.5340	1.6240	0.4330	2364.94	0.25	0.30	0.41
P4							
ANN	0.3319	0.3716	0.2063	732.24	0.65	0.71	0.81
SWR	1.1734	1.2640	0.4074	2175.54	0.25	0.31	0.44
P5							
ANN	0.6455	0.7940	0.4156	2152.72	0.31	0.36	0.48
SWR	1.1726	1.0276	0.6099	2310.61	0.27	0.30	0.41
P6							
ANN	0.4628	0.9579	0.2963	1097.54	0.52	0.56	0.72
SWR	0.9092	0.8835	0.4612	2174.54	0.26	0.32	0.44
IFPUG							
IFPUG-PFA	1.7977	1.7989	0.5070	6652.55	0.16	0.18	0.00

4.2 Results and Discussion

4.2.1 Comparing Predictive Accuracy in SDEE: DLMLP, MLR, RF

This study comprehensively analyses model performance across different predictor groups, focusing on MLR, RF, and DLMLP models (see Table 4-1). The objective is to compare the performance of these models individually with each predictor group and to answer RQ1: Is the DLMLP more accurate than the MLR, RF?

In predictors P1 and P2, we observe notable performance trends among the algorithms. In P1, DLMLP achieves the lowest MAE at 2066.69, surpassing RF and MLR. Similarly, in P2, DLMLP consistently outperforms MLR and RF across various metrics, with an MAE of 1768.61 compared to higher MAE values for MLR and RF. These patterns collectively highlight DLMLP's superior predictive performance across different predictors. Similarly, in the P3 predictor group, competitive performance is observed between MLR and RF, with RF exhibiting advantages across all metrics. Nevertheless, DLMLP consistently outperforms RF and MLR within this predictor group. Moreover, in the P4 predictor group, DLMLP achieves the lowest MAE with a value of 530.64, outperforming MLR (2018.79) and RF (2012.14). While other metrics, including MMRE, MBRE, MIBRE, Pred(0.25), and SA, favour RF over MLR, DLMLP's performance exceeds that of both MLR and RF in terms of all metrics. Additionally, P4 predictor group outperforms compared with P1, P2, P3, P5, and P6. Consistent with the trends observed in the P5 and P6 predictors, DLMLP consistently demonstrates superior predictive performance over MLR and RF across multiple metrics, including MNRE, MBRE, MIBRE, and MAE. DLMLP's proficiency in prediction is evident through its consistently lower metric values.

As previously mentioned, the trends observed in the predictor persist in P_D, P_A, P_K, P_C, and $P_{Dataset2}$ predictors. Moreover, DLMLP consistently outperforms MLR and RF regarding MMRE, MBRE, MIBRE, MAE, Pred(0.25), and SA, establishing itself as the preferred model within these scenarios. RF demonstrates improved performance over MLR in the majority of cases.

In conclusion, this study provides definitive answers to the research questions. RQ1, which investigates the accuracy of DLMLP compared to MLR and RF models, confirms that DLMLP outperforms both models across all predictive factors.

4.2.2 Comparing DLMLP vs. Baseline Models

The performance of DLMLP compared with baseline models is presented in Table 4-3. It is noticeable that those figures reveal that the DLMLP consistently outperforms the baseline models across diverse datasets (P1 to P6) based on various performance metrics. DLMLP achieves lower values across metrics, including MMRE, MBRE, and MIBRE. These results imply that DLMLP provides more accurate and less biased effort estimations than the alternative models.

Its superior performance extends to the MAE, demonstrating its effectiveness in minimising the absolute difference between predictions and actual effort values. The strength of the DLMLP further manifests in its ability to provide predictions within a specified tolerance. Higher Pred(0.25) values indicate that DLMLP delivers effort estimation that closely aligns with the actual effort.

4.2.3 Impact of Dataset Balancing on Accuracy of SDEE in DLMLP

The other aim of this study is to comprehensively assess the impact of using a balanced dataset and handling categorical variables in deep learning models in the context of effort estimation. Specifically, the thesis compares model performance with a balanced dataset based on categorical variable handling (DLMLPB) against the model without balancing categorical variables (DLMLP) across predictors P1, P2, P3, P4, P5, and P6. This evaluation answers RQ2 that dataset balancing might enhance the predictive accuracy of DLMLP methods in software effort estimation. Model performance evaluation is based on critical metrics such as MMRE, MBRE, MIBRE, MAE, Pred, and SA. Table 4-1 illustrates the performance comparison between DLMLP and DLMLPB based on datasets P1 to P6. The results reveal that DLMLPB consistently outperformed DLMLP in estimation accuracy across all predictors. This finding provides strong evidence to support the notion that using a balanced dataset and effectively handling categorical variables leads to improved estimation accuracy in deep learning models.

This discovery addresses RQ2 by affirming that dataset balancing improves DLMLP's predictive accuracy in effort estimation. It underscores the importance of dataset balance and proper handling of categorical variables for accurate estimations. Researchers are strongly advised to balance their datasets and use suitable techniques for categorical variables to enhance the accuracy and reliability of their models.

4.2.4 Evaluating Ensemble for SDEE: MLR, RF, and DLMLP

The next objective of this research is to compare the performance of MLR, RF, and DLMLP against ensemble models established by incorporating MLR, RF and DLMLP for effort estimation using eleven predictors: P1, P2, P3, P4, P5, P6, P_D, P_A, P_K, P_C, and $P_{Dataset2}$. Table 4-1 and Table 4-2 present the performance of MLR, RF, DLMLP, and ensemble models.

Table 4-1 shows that the ensemble consistently outperforms MLR, RF, and DLMLP in the P1 predictor, demonstrating superior accuracy with lower MMRE, MBRE, MIBRE, and MAE values, indicating more precise effort estimation. The ensemble also excels in predictive power, with higher Pred(0.25) and SA values than the other algorithms. These trends extend across various predictors, such as P2, P3, P5, P_C, P_K, and $P_{Dataset2}$, where the ensemble maintains its edge in accuracy and predictive prowess. In predictors like P4, P6, P_A, and P_D, the ensemble still outperforms MLR and RF in most metrics, although DLMLP holds a slight advantage in specific cases, such as P4, P6, P_D, P_C, and P_K.

In conclusion, the thorough analysis of different effort estimation scenarios shows that ensemble models consistently outperform individual models like MLR and RF. This finding suggests that ensemble models have the potential to improve effort estimation, mainly when precision is crucial significantly. Considering ensemble models is highly recommended when striving for more accurate and reliable effort estimations.

4.2.5 *A Comparative Analysis of Transfer Learning and DLMLP*

The other objective of this study is to compare the accuracy of the transferred model with the DLMLP-based model trained on the new datasets. This comparison addresses RQ4: "Does DLMLP-based transfer learning offer accuracy over conventional DLMLP?". The study also introduces a pre-trained model based on the ISBSG dataset.

In TL-Case1, DLMLP-based models trained on Dataset 1 are employed to evaluate the performance of effort estimation on Dataset 2. TL-Case2 involves training DLMLP-based models on 80% of the Albrecht, China, and Dataset 2 datasets and evaluating their performance on the remaining 20%. Finally, TL-Case3 employs DLMLP-based models trained on Dataset 1 (pre-trained model) and continued their training on 80% of the new datasets, with an evaluation conducted on the remaining 20% of new datasets.

Table 4-2 illustrates the performance comparison among TL-Case1, TL-Case2, and TL-Case3 across the studied datasets. The results obtained from these three cases provide insights into the efficacy of transfer learning. TL-Case 1 obtained from Dataset2 reveals that the DLMLP model trained on Dataset 1 does not outperform TL-Case2 and TL-Case3. On the other hand, TL-Case3 truly showcases its potential. By combining the strengths of the pre-trained model with further training on the combined datasets, TL-Case3 achieves the lowest MMRE, MBRE, MIBRE, MAE, Pred, and SA values, suggesting superior performance in estimating software effort. These findings collectively emphasise the significance of transfer learning and its ability to enhance the accuracy of effort estimation models in software development projects.

In conclusion, transfer learning offers significant advantages in effort estimation by leveraging prior knowledge and improving the accuracy of predictions. Examining three scenarios (TL-Case1, TL-Case2, and TL-Case3) has provided valuable insights into the effectiveness of transfer learning techniques within this domain. Notably, TL-Case3, which utilised pre-trained models adjusted on a combined dataset, emerged as the most effective strategy, highlighting the potential of transfer learning to improve effort estimation accuracy significantly.

4.2.6 *Exploring the Influence of IS and RS on SDEE*

Table 4-1 shows DLMLP and DLMLPB performance with predictors P1 to P6. The results reveal the impact of IS and RS on effort estimation. Comparing P1 (AFP), P3 (AFP, IS), and P5 (AFP, IS, RS), we see that IS significantly enhances accuracy in P3. P3 consistently achieves lower MMRE, MBRE, MIBRE, and MAE, highlighting IS's substantial contribution. P5, including IS and RS, performs similarly to P3, indicating RS adds a minor improvement when AFP and IS are present. These findings stress the importance of including IS in effort estimation models for valuable insights into software development complexities. Furthermore, when comparing P1 (AFP) against P2 (EI, EO, EQ, EIF, ILF), it becomes evident that P2 consistently outperforms P1 regarding accuracy metrics. Including complexity-related predictors in P2, such as EI, EO, EQ, EIF, and ILF, enhances estimation accuracy. However, it is essential to note that including IS and RS in P4 and P6 further improves estimation accuracy beyond the AFP-based model of P1. These findings underscore the critical role of IS and RS in capturing the complex factors that significantly impact effort estimation.

Upon examining the experimental results, a difference in performance between predictors P4 and P6 becomes evident. Predictor P4 represents the inclusion of IS as a predictor, while predictor P6 incorporates both IS and RS as predictors. Surprisingly, including RS in predictor P6 results in lower performance than P4 without RS. This result suggests that RS might have a detrimental effect on the model's overall performance.

Interestingly, the experimental results reveal a surprising outcome: including RS in predictor P6 leads to lower performance than P4, which does not include RS. This unexpected result suggests a potential negative impact of RS on the overall model performance. Further research is needed to understand the underlying factors contributing to this phenomenon and to explore potential approaches to mitigating the adverse effects of RS on effort estimation accuracy.

In conclusion, including IS and RS predictors consistently enhances the accuracy of effort estimation models. Predictor sets incorporating IS and RS, such as P3 and P5, demonstrate superior performance compared to models solely relying on AFP or complexity factors. These findings highlight the importance of considering IS and RS predictors to capture the intricate nature of software development projects and achieve more precise and reliable effort estimation.

4.3 Evaluation against Hypotheses

Table 4-4 and Table 4-5 present the results of the Mann-Whitney U-tests, which are conducted to examine potential significant differences in mean among various machine learning methodologies: DLMLP, MLR, RF, the ensemble, transfer learning, and DLMLPB. The primary aim of these tests is to ascertain whether statistically significant variations in performance among these methodologies exist. The null hypothesis (H0) stipulates significantly less or equal mean accuracy, while the alternative hypothesis (H1) posits the contrary.

Table 4-4: The Mann-Whitney hypothesis test between DLMLP, MLR, RF, the ensemble and DLMLPB models based on P1, P2, P3, P4, P5, P6

No	Model 1	Model 2	P-value					
			P1	P2	P3	P4	P5	P6
0	DLMLP	MLR	0.00	0.01	0.04	0.00	0.04	0.02
1	DLMLP	RF	0.03	0.04	0.00	0.00	0.04	0.00
2	DLMLP	DLMLPB	0.00	0.02	0.04	0.00	0.00	0.02
3	DLMLP	Ensemble	0.01	0.04	0.01	0.60	0.01	0.70
4	MLR	RF	0.02	0.01	0.04	0.00	0.04	0.04
5	MLR	DLMLPB	0.01	0.00	0.03	0.00	0.03	0.01
6	MLR	Ensemble	0.00	0.00	0.00	0.00	0.00	0.00
7	RF	DLMLPB	0.00	0.02	0.04	0.01	0.04	0.00
8	RF	Ensemble	0.00	0.01	0.00	0.00	0.00	0.03
9	DLMLPB	Ensemble	0.01	0.02	0.01	0.04	0.01	0.01

- **DLMLP vs. MLR, and RF**:

As shown in Table 4-4, the p-values resulting from the comparison of DLMLP with MLR and RF, using predictors from P1 to P6, are consistently below the significance threshold of 0.05. Furthermore, when the study extends this comparison to include other predictors (P_A, P_D, P_C, $P_{Dataset2}$), as presented in Table 4-5, the findings that the p-values remain below 0.05, these findings collectively indicate that DLMLP exhibits substantial variations in mean performance compared to MLR and RF. Consequently, the null hypothesis ($\mu DLMLP \leq$

$\mu MLR\ or\ \mu DLMLP \leq\ \mu RF$) is rejected, highlighting that the mean accuracy obtained from DLMLP is greater than MLR and RF in software effort estimation.

Table 4-5: The Mann-Whitney hypothesis test between TL-Case2 (DLMLP), MLR, RF, Ensemble and TL-Case3 models based on P_A, P_D, P_C, $P_{Dataset2}$.

No	Model 1	Model 2	P-value				
			P_A	P_D	P_C	P_K	$P_{Dataset2}$
0	TL-Case2	MLR	0.00	0.00	0.00	0.02	0.00
1	TL-Case2	RF	0.02	0.00	0.00	0.01	0.03
3	TL-Case2	Ensemble	0.25	0.57	0.08	0.08	0.00
4	MLR	RF	0.00	0.00	0.02	0.02	0.04
6	MLR	Ensemble	0.02	0.00	0.00	0.00	0.00
8	RF	Ensemble	0.02	0.00	0.00	0.03	0.00
10	TL-Case3	Ensemble	0.00	#	0.01	#	0.00
11	TL-Case3	RF	0.00	#	0.00	#	0.00
12	TL-Case3	MLR	0.00	#	0.00	#	0.00
13	TL-Case3	TL-Case2	0.03	#	0.01	#	0.01

- **DLMLP vs. DLMLPB**:

Balancing the dataset in DLMLPB yields notable improvements, as evidenced by relatively low p-values: 0.00 for P1, 0.02 for P2, 0.04 for P3, 0.00 for P4, 0.00 for P5, and 0.02 for P6 (see Table 4-4). As a result, the alternative hypothesis ($\mu DLMLPB >\ \mu DLMLP$) is retained in this case, highlighting that the mean accuracy obtained from DLMLPB is greater than DLMLP.

- **Ensemble vs. DLMLP, MLR, and RF**:

In the comparative analysis of mean performance metrics across the ensemble, MLR, and RF concerning predictors P1 to P6, P_A, P_D, P_C, P_K, and $P_{Dataset2}$, as presented in Table 4-4 and Table 4-5, it is evident that the derived p-values for these comparisons consistently fall below the 0.05 significance threshold. This outcome leads to rejecting the null hypothesis ($\mu ENS\ \leq\ \mu MLR, or\ \mu ENS \leq\ \mu RF$), thereby establishing statistically significant mean differences between the ensemble and MLR and RF, confirming that the mean accuracy attained from the ensemble is better than those obtained from MLR and RF. Moreover, when examining the mean accuracy achieved from the ensemble approach in comparison to DLMLP, it is observed that predictors P1, P2, P3, P5, P_A, $P_{Dataset2}$ exhibit p-values fall below 0.05 except P4, P6, P_D, P_C, and P_K. Consequently, the null hypothesis ($\mu ENS\ \leq\ \mu DLMLP$) might be rejected.

In conclusion, this observation suggests that, in general, the mean accuracy obtained from the ensemble is more significant than that obtained from DLMLP, MLR, and RF in software effort estimation.

- **Transfer Learning (TL-Case3) vs. DLMLP**:

The transfer learning model **(TL-Case3)** shows significant mean performance disparities compared to the DLMLP, as the p-values obtained from those models are less than 0.05 in P_A, P_C, and $P_{Dataset2}$. The null hypothesis ($\mu TL \leq \mu DLMLP$) is rejected, indicating that the mean accuracy obtained from the transfer learning model **(TL-Case3)** is significantly greater than DLMLP.

- **Influence of IS and RS in the accuracy of effort estimation**:

Analysing the results in Table 4-6 provides insights into RQ5, which aims to determine whether the categorical variables IS and RS significantly influence effort estimation accuracy. These values indicate that both. βIS and γRS for each predictor are not equal to 0. Consequently, we reject the null hypothesis ($\beta IS = \gamma RS = 0$) and accept the alternative hypothesis, suggesting that the categorical variables influence the accuracy of effort estimation.

Table 4-6: The Regression coefficient for IS and RS obtained from MLR.

Predictors	$\boldsymbol{\beta IS}$	$\boldsymbol{\gamma RS}$	**Description**
P5	-2.11	-358.41	$\beta IS \neq \gamma RS \neq 0$
P6	24.94	-804.60	$\beta IS \neq \gamma RS \neq 0$

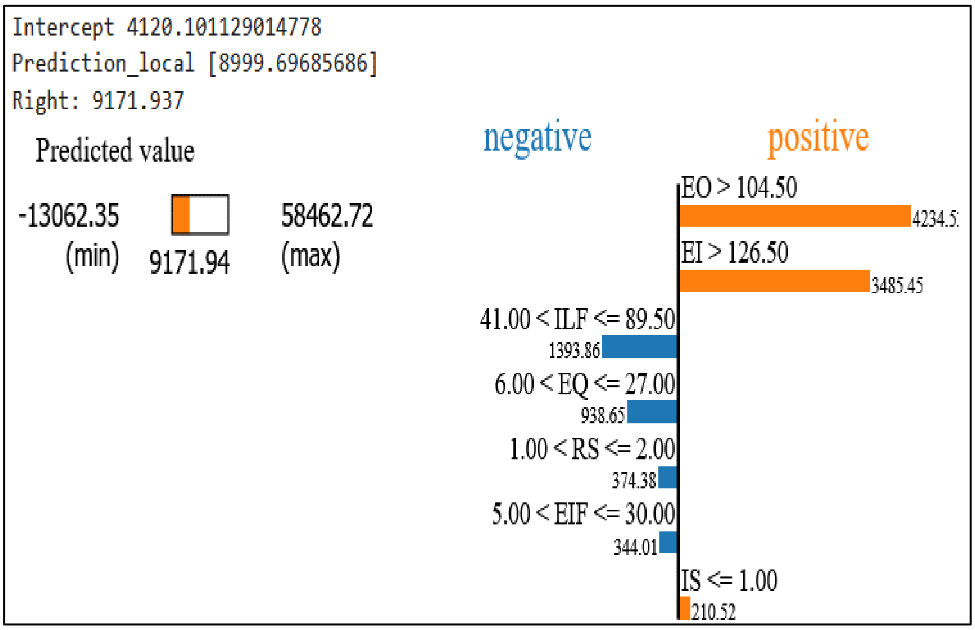
Examining the coefficients in this table reveals that RS (-358.41 and -804.60) has smaller values than IS (-2.11 and 24.94), suggesting a potentially weaker impact on effort estimation accuracy. This finding implies that IS may play a more substantial role in accuracy. These coefficient differences highlight the importance of these variables in influencing effort estimation accuracy, aiding in decision-making for model development and feature selection.

The Mann-Whitney U-test shows that DLMLP outperforms MLR and RF in mean performance. Balancing the dataset in DLMLPB improves mean performance compared to DLMLP. Transfer learning differs significantly in mean performance from DLMLP, while the ensemble approach performs similarly. Regression coefficients for IS and RS from the MLR model reveal their influence: IS moderately affects accuracy, while RS has a minor impact. These insights guide practitioners in selecting ML methods, emphasising performance and categorical variables for informed decisions in practical scenarios.

4.4 Model Explainability Findings - Analysis of Predictor Contributions

4.4.1 LIME

This study explores the interpretation of predicted effort values generated by DLMLP-P6. This model contains input features EO, EIF, ILF, EQ, EI, IS, and RS, where EO, EIF, ILF, EQ, and EI are measured in function points, and IS RS are categorical variables, a predicted effort is measured in person-hours. This method employed for interpretation is LIME, as illustrated in Figure 4-1.



```
Intercept 4120.101129014778
Prediction_local [8999.69685686]
Right: 9171.937
```

Figure 4-1: Interpreting the predicted effort values obtained from DLMLP-P6

An exhaustive analysis of the outcomes derived from LIME's interpretations is presented below:

- LIME predicts approximately 8999.69 (person-hours) for a specific instance, while the actual prediction is 9171.94 (person-hours). The range of predicted effort spans from -13062.35 to 58462.72.
- The feature contributions in this analysis offer valuable insights into the factors influencing the predicted effort. Notably:
  - ✓ ILF: When ILF falls within the range of 41 to 89.5, it negatively affects the predicted effort by contributing to -1393.86 person-hours. This result suggests that an increase in ILF within this range correlates with reducing the predicted effort.
  - ✓ EQ: Falling within the range of 6 to 27, EQ negatively affects the predicted effort, contributing -938.65 person-hours. This result implies that a moderate number of EQ might decrease the predicted effort compared to extreme values.
  - ✓ RS: With values between 1 and 2, RS negatively influences the predicted effort by contributing -374.38 person-hours. This result indicates that a specific range of RS values tends to decrease the predicted effort.
  - ✓ EIF: When EIF falls between 5 and 30, it negatively impacts the predicted effort, contributing to -344.01 person-hours. This finding suggests that an increase in EIFs is associated with a decrease in predicted effort within this range.
  - ✓ EO: When EO exceeds 104.50, it positively influences the predicted effort with a contribution of 4234.5 person-hours. This finding indicates that more external outputs in the project increase the predicted effort.

- ✓ EI: An EI value more excellent than 126.50 positively contributes 3485.45 person-hours, signifying that an elevated count of external inputs increases predicted effort.
- ✓ IS: When IS is 0.0 or 1.0, it positively contributes 210.52 person-hours, suggesting that a smaller interface size is associated with higher predicted effort.

The LIME results suggest that EI, EO, and IS are the key features impacting the predicted effort. These variations in LIME's interpretations emphasise the importance of comparing results. LIME's visualisations further aid in understanding these nuanced interpretations, enabling a more comprehensive analysis of feature influence.

4.4.2 SHAP

Figure 4-2 presents the feature contributions obtained from Dataset 1 of the testing dataset for DLMLP-P6. This consistency highlights their crucial roles in effort estimation within these models.

- The EI, EO, EQ, ILF, and EIF features demonstrate relatively consistent trends, with a positive contribution associated with higher values and a minor negative contribution linked to lower values across all three models, while IS has a slight positive contribution.
- Notably, the feature RS appears to have no significant contribution to predictions, irrespective of its value.

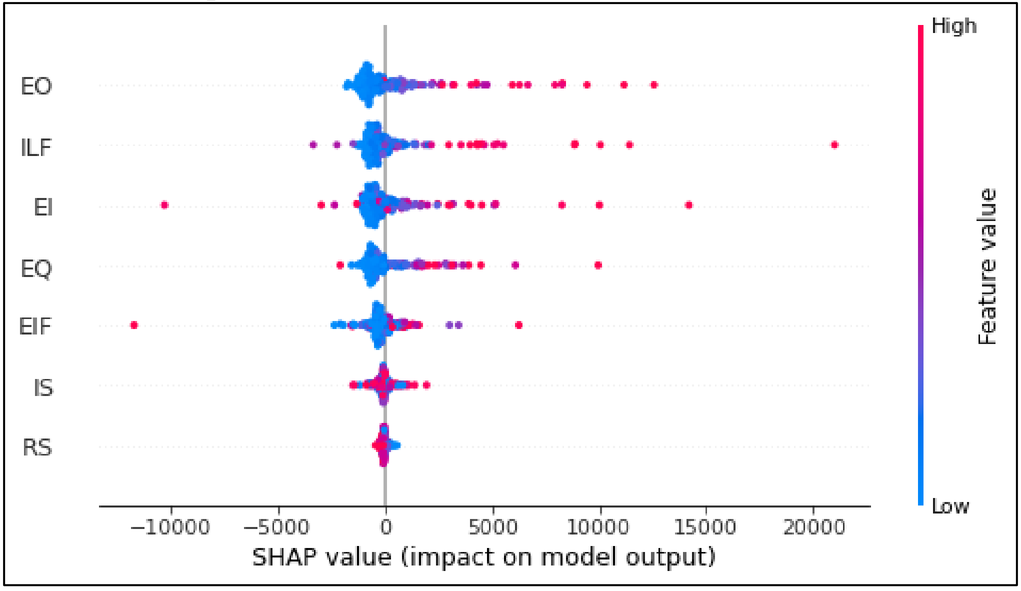


Figure 4-2: The contributions of each feature in DLMLP-P6

The consistency in the contributions of EI, EO, EQ, ILF, and EIF suggests their critical roles in effort estimation across these models. However, the negligible contribution of RS merits further exploration to comprehend its influence on the model's predictive performance. These findings contribute a deeper understanding of interpretability and feature importance in software effort estimation using deep learning models.

5. CONTRIBUTIONS

This study seeks to provide specific contributions to the domain of SDEE by addressing several key research areas:

- **Comparative Analysis of Predictive Models:**

This research extensively evaluates predictive models across distinct predictors, specifically MLR, RF, and DLMLP. The objective is to determine the superior model for SDEE. Findings reveal that DLMLP consistently surpasses MLR and RF across multiple performances, including MMRE, MBRE, MIBRE, MAE, Pred(0.25), and SA. Consequently, DLMLP emerges as the preferred predictive model for SDEE.

- **Impact of Dataset Balancing on Accuracy:**

This study examines the influence of dataset-balancing techniques and the handling of categorical variables in deep learning models by comparing DLMLP (unbalanced dataset) to DLMLPB (balanced dataset). The outcomes indicate that DLMLPB consistently outperforms DLMLP across all predictor sets, underscoring the significance of dataset balancing and effective categorical variable management in enhancing estimation accuracy.

- **Ensemble Models for SDEE:**

The research evaluates ensemble models that combine MLR, RF, and DLMLP to assess their effectiveness in SDEE across various predictor sets. The findings demonstrate that ensemble models, mainly when precision is pivotal, exhibit superior performance compared to individual models. Nonetheless, it is noteworthy that DLMLP retains a slight advantage in specific scenarios, suggesting that the choice between ensemble models and DLMLP should hinge on the specific requirements of the effort estimation.

- **Transfer Learning for Enhanced Accuracy:**

The study investigates the efficacy of transfer learning in the context of SDEE by comparing DLMLP-based models trained on different datasets. The results emphasise the potential of transfer learning, particularly when employing a pre-trained model and fine-tuning it on the new dataset. By starting with a pre-trained model as a foundation, researchers and practitioners might save valuable time and resources that would otherwise be required for extensive model training.

- **Influence of Categorical Variables:**

This research examines the impact of IS and RS predictors on effort estimation accuracy across diverse predictor sets. The findings underscore the imperative nature of incorporating IS predictors into effort estimation models, as they encapsulate crucial information regarding the intricacies of the software development process. While RS predictors exhibit some influence on accuracy, further investigation is warranted to comprehend their nuanced impact.

6. CONCLUSION

The thesis evaluates effort estimation using three methods, MLR, RF, and DLMLP, across diverse datasets, primarily from ISBSG (2020), with supplementary datasets. Eleven predictors are considered: six combinations (P1-P6) from ISBSG and individual predictors (P_A, P_D, P_C, P_K, and $P_{Dataset2}$). The results answer RQ1, showing that DLMLP consistently outperforms MLR and RF in SDEE accuracy. Comparative analysis confirms DLMLP's superiority across various performance metrics compared to these baseline models.

Additionally, this study investigates the impact of two categorical variables, the industry sector and relative size factor, along with FPA factors as input features. These variables are chosen to assess their influence on DLMLP, MLR, and RF models. The research aims to address dataset imbalance using class weights (RQ2) and compares DLMLP's performance on the original dataset with that on the balanced dataset (DLMLPB). The findings may reveal if the dataset-balancing approach in this study outperforms the unbalanced dataset.

The thesis explores ensemble techniques that combine the examined models. Stacking is applied to MLR and RF, using XGBoost as the final estimator. The ensemble results are further combined with DLMLP using a voting method. These experiments involve eleven predictor variables (P1 to P6, P_A, P_D, P_C, P_K, and $P_{Dataset2}$). In general, the ensemble approach performs better than individual models. These findings offer insights into addressing RQ3, suggesting the ensemble approach may outperform individual models.

The study explores three scenarios with a pre-trained model. Scenario one applies the model to a new test dataset, TL-Case1. In scenario two, the DLMLP architecture is used to create a model on a new training dataset, TL-Case2, which yields better prediction results than TL-Case1. The final scenario, TL-Case3, involves further training the pre-trained model, known as transfer learning, and outperforms TL-Case1 and TL-Case2 in predictions. This finding suggests that transfer learning improves prediction accuracy, addressing RQ4. A research library has also been created (https://github.com/huynhhoc/effort-estimation-by-using-pre-trained-model) for researchers to use or enhance the pre-trained model for improved accuracy.

As discussed in Section 4.2.6, IS demonstrates a slight effect on the accuracy, while RS has a relatively small effect. This observation is supported by analyses using LIME and SHAP, which might answer for RQ5 that IS has a positive effect on effort estimation, while RS has a negative one. The findings obtained from LIME and SHAP also reveal that EI and EO positively impact effort estimation compared with EQ, EIF and ILF.

7. REFERENCES

[1] A. J. Albrecht, "Measuring application development productivity," in *Proc. Joint Share, Guide, and IBM Application Development Symposium, 1979*, 1979.

[2] N. Agarwal, A. Sondhi, K. Chopra, and G. Singh, "Transfer Learning: Survey and Classification," in *Smart Innovations in Communication and Computational Sciences*, S. Tiwari, M. C. Trivedi, K. K. Mishra, A. K. Misra, K. K. Kumar, and E. Suryani, Eds., Singapore: Springer Singapore, 2021, pp. 145–155.

[3] L. L. Minku, "Multi-stream online transfer learning for software effort estimation: Is it necessary?," in *Proceedings of the 17th International Conference on Predictive Models and Data Analytics in Software Engineering*, 2021, pp. 11–20.

[4] L. L. Minku and X. Yao, "Can cross-company data improve performance in software effort estimation?," in *Proceedings of the 8th International Conference on Predictive Models in Software Engineering*, 2012, pp. 69–78.

[5] V. Van Hai, H. Le Thi Kim Nhung, and H. T. Hoc, "A review of software effort estimation by using functional points analysis," in *Proceedings of the Computational Methods in Systems and Software*, Springer, 2019, pp. 408–422.

[6] ISBSG, "ISBSG," International Software Benchmarking Standards Group, Release R1.

[7] C. López-Martín, A. Chavoya, and M. E. Meda-Campaña, "Use of a feedforward neural network for predicting the development duration of software projects," in *2013 12th International Conference on Machine Learning and Applications*, IEEE, 2013, pp. 156–159.

[8] A. Corazza, S. Di Martino, F. Ferrucci, C. Gravino, and F. Sarro, "From Function Points to COSMIC - A Transfer Learning Approach for Effort Estimation," in *Product-Focused Software Process Improvement*, P. Abrahamsson, L. Corral, M. Oivo, and B. Russo, Eds., Cham: Springer International Publishing, 2015, pp. 251–267.

[9] A. Ali and C. Gravino, "A systematic literature review of software effort prediction using machine learning methods," *Journal of software: evolution and process*, vol. 31, no. 10, p. e2211, 2019.

[10] S. Shukla and S. Kumar, "Applicability of neural network based models for software effort estimation," in *2019 IEEE World Congress on Services (SERVICES)*, IEEE, 2019, pp. 339–342.

[11] A. G. Priya Varshini, K. Anitha Kumari, D. Janani, and S. Soundariya, "Comparative analysis of Machine learning and Deep learning algorithms for Software Effort Estimation," *J Phys Conf Ser*, vol. 1767, no. 1, p. 012019, 2021, doi: 10.1088/1742-6596/1767/1/012019.

[12] W. Amaral, L. Rivero, G. B. Junior, and D. Viana, "Using Machine Learning Technique for Effort Estimation in Software Development," in *Proceedings of the XVIII Brazilian Symposium on Software Quality*, in SBQS'19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 240–245. doi: 10.1145/3364641.3364670.

[13] M. T. Ribeiro, S. Singh, and C. Guestrin, " 'Why should i trust you?' Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[14] L. Antwarg, R. M. Miller, B. Shapira, and L. Rokach, "Explaining anomalies detected by autoencoders using SHAP," *arXiv preprint arXiv:1903.02407*, 2019.

[15] G. C. Low and D. R. Jeffery, "Function points in the estimation and evaluation of the software process," *IEEE transactions on Software Engineering*, vol. 16, no. 1, pp. 64–71, 1990.

[16] A. J. Albrecht and J. E. Gaffney, "Software function, source lines of code, and development effort prediction: a software science validation," *IEEE transactions on software engineering*, no. 6, pp. 639–648, 1983.

[17] 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.

[18] IFPUG, "http://www.ifpug.org/," International Function Point Users Group.

[19] 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, 2013.

[20] L. Breiman, "Arcing the edge," Technical Report 486, Statistics Department, University of California at …, 1997.

[21] H. Aljamaan and A. Alazba, "Software defect prediction using tree-based ensembles," in *Proceedings of the 16th ACM international conference on predictive models and data analytics in software engineering*, 2020, pp. 1–10.

[22] Aurélien Géron, "Ensemble Learning and Random Forests," in *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2nd ed., O'reilly, 2019, pp. 189–212.

[23] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," *arXiv preprint arXiv:1811.03378*, 2018.

[24] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Trans Pattern Anal Mach Intell*, vol. 12, no. 10, pp. 993–1001, 1990.

[25] P. Refaeilzadeh, L. Tang, and H. Liu, "Cross-validation.," *Encyclopedia of database systems*, vol. 5, pp. 532–538, 2009.

[26] X. Ma, Y. Zhang, and Y. Wang, "Performance evaluation of kernel functions based on grid search for support vector regression," in *2015 IEEE 7th international conference on cybernetics and intelligent systems (CIS) and IEEE conference on robotics, automation and mechatronics (RAM)*, IEEE, 2015, pp. 283–288.

[27] J. Brownlee, "A gentle introduction to the rectified linear unit (ReLU)," *Machine learning mastery*, vol. 6, 2019.

[28] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[29] E. Okewu, S. Misra, and F.-S. Lius, "Parameter tuning using adaptive moment estimation in deep learning neural networks," in *International Conference on Computational Science and Its Applications*, Springer, 2020, pp. 261–272.

[30] V. N. Gudivada, M. T. Irfan, E. Fathi, and D. L. Rao, "Chapter 5 - Cognitive Analytics: Going Beyond Big Data Analytics and Machine Learning," in *Handbook of Statistics*, vol. 35, V. N. Gudivada, V. V Raghavan, V. Govindaraju, and C. R. Rao, Eds., Elsevier, 2016, pp. 169–205. doi: https://doi.org/10.1016/bs.host.2016.07.010.

[31] P. Silhavy, R. Silhavy, and Z. Prokopova, "Categorical variable segmentation model for software development effort estimation," *IEEE Access*, vol. 7, pp. 9618–9626, 2019.

[32] S. Amasaki and T. Yokogawa, "The effects of variable selection methods on linear regression-based effort estimation models," in *2013 Joint Conference of the 23rd International Workshop on Software Measurement and the 8th International Conference on Software Process and Product Measurement*, IEEE, 2013, pp. 98–103.

[33] R. Silhavy, P. Silhavy, and Z. Prokopova, "Analysis and selection of a regression model for the use case points method using a stepwise approach," *Journal of Systems and Software*, vol. 125, pp. 1–14, 2017.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND SYMBOLS

Abbreviation	**Definition**
ADASYN	Adaptive Synthetic Sampling Approach
AFP	Adjusted Function Points
ANN	Artificial Neural Network
CMMI	Capability Maturity Model Integrated
COCOMO	Constructive Cost Model
CSBSG	Chinese Software Benchmarking Standard Group
DL	Deep learning
DLMLP	Deep learning - Multilayer perceptron
DLMLPB	Deep learning - Multilayer perceptron with balancing dataset
DTF	Decision Tree Forest
EI	External Inputs
EIF	External Interface File
EO	External Outputs
EQ	External Inquiry
FFNN	FeedForward Neural Network
FPA	Function Point Analysis
GMM	Gaussian Combination Model
GMM	Gaussian Combination Model
GSC	General Systems Characteristic
HGBoost	Histogram Gradient Boosting
IFPUG	International Function Point Users Group
ILF	Internal Logical File
IQR	Interquartile Range
IS	Industry Sector
ISBSG	International Software Benchmarking Standards Group
LIME	Local Interpretable Model-agnostic Explanations
MAE	Mean Absolute Error
MBRE	Mean Balance Relative Error
MFP	Modified Function Points
MIBRE	Mean Inverted Balance Relative Error
MLP	Multilayer perceptron
MLR	Multiple Linear Regression
MMRE	Mean Magnitude of Relative Error

MRE	The Magnitude of Relative Error
MSE	Mean Square Error
NESMA	Netherlands Software Metrics Association
PDR	Productivity Rate
PRED(x)	Prediction at level x
ReLU	Rectified Linear Unit
RF	Random Forest
RQ	Research Question
RS	Relative Size
SA	Standardised Accuracy
SDEE	Software Development Effort Estimation
SDO	Software Development Organization
SHAP	SHapley Additive exPlanations
SMOTE	Synthetic Minority Over-sample Technique
SVD	Singular Value Decomposition
SVM	Support Vector Machine
SVR	Support Vector Regression
TL-Case 1	Transfer Learning – Case 1
TL-Case 2	Transfer Learning – Case 2
TL-Case 3	Transfer Learning – Case 3
UFP	Unadjusted Function Points
VAF	Value Adjustment Factor
WBS	Work Breakdown Structure

LIST OF PUBLICATIONS

Journals:

1. **Hoc, H. T.**, R. Silhavy, Z. Prokopova and P. Silhavy, "Comparing Multiple Linear Regression, Deep Learning and Multiple Perceptron for Functional Points Estimation," in *IEEE Access*, vol. 10, pp. 112187-112198, 2022.
2. **Hoc, H. T.**, R. Silhavy, Z. Prokopova and P. Silhavy, "Comparing Stacking Ensemble and Deep Learning for Software Project Effort Estimation," in IEEE Access, vol. 11, pp. 60590-60604, 2023.
3. **Hoc, H. T.**, Silhavy P., Fajkus M., Prokopova Z, Silhavy R. Propose-Specific Information Related to Prediction Level at x and Mean Magnitude of Relative Error: A Case Study of Software Effort Estimation. *Mathematics*. 2022.
4. **Hoc, H. T.**, Silhavy P., Dey SK, Hoang SD, Prokopova Z, Silhavy R. Analysing Public Opinions Regarding Virtual Tourism in the Context of COVID-19: Unidirectional vs. 360-Degree Videos. *Information*. 2023.
5. DEY, Sandeep Kumar, Duc Sinh HOANG, **Hoc, H. T.**, Quynh Giao Ngoc PHAM. Engaging virtual reality technology to determine pro-environmental behaviour: The Indian context. Geojournal of Tourism and Geosites 2022.
6. Kondamudi, B. R., Hoang, S. D., Tuckova, Z., Dey, S. K., **Hoc, H. T.**, & Kumar, B. R. (2023). Tourists' Perception and Influence Factors in Virtual Tourism Using Bayesian Sentimental Analysis Model in Vietnam Based on e WOM for Sustainable Development. Journal of Law and Sustainable Development, 11(3), e338. https://doi.org/10.55908/sdgs.v11i3.338.
7. Pham P.T., **Hoc, H. T.**, B.Popesko, Sinh D.H., Tri B.T, "Impact of Fintech's Development on Bank Performance: An Empirical Study from Vietnam.", accept submission by GamaIJB, Volume 26 No.1, 2023.

Conferences:

8. **Hoc, H. T**., Van Hai, V., Nhung, H. L. T. K., & Jasek, R. (2023). Improving the Performance of Effort Estimation in Terms of Function Point Analysis by Balancing Datasets. In *Proceedings of 6th CoMeSySo 2022*.
9. **Hoc, H. T**., Van Hai, V., & Le Thi Kim Nhung, H. (2020). AdamOptimizer for the optimisation of use case points estimation. In *Software Engineering Perspectives in Intelligent Systems: Proceedings of 4th CoMeSySo 2020.*
10. **Hoc, H. T**., Van Hai, V., & Nhung, H. L. T. K. (2021). An approach to adjust effort estimation of function point analysis. In *Software Engineering and Algorithms: Proceedings of 10th CSOC 2021, Vol. 1* (pp. 522-537).
11. **Hoc, H. T.**, Van Hai, V., & Le Thi Kim Nhung, H. (2019). A review of the regression models applicable to software project effort estimation. *Computational Statistics and Mathematical Modeling Methods in Intelligent Systems: Proceedings of 3rd CoMeSySo 2019, Vol. 2 3*, 399-407.

12. Van Hai, V., Le Thi Kim Nhung, H., & **Hoc, H. T.** (2021). Empirical Evidence in Early Stage Software Effort Estimation Using Data Flow Diagram. In *Software Engineering and Algorithms: Proceedings of 10th CSOC 2021*.
13. Nhung, H. L. T. K., Van Hai, V., **Hoc, H. T.** Analyzing Correlation of the Relationship between Technical Complexity Factors and Environmental Complexity Factors for Software Development Effort Estimation.
14. Hai, V. V., Nhung, H. L. T. K., & **Hoc, H. T.** (2021). Calibrating Function Complexity in Enhancement Project for Improving Function Points Analysis Estimation. In *Software Engineering Application in Informatics: Proceedings of 5th CoMeSySo 2021, Vol. 1* (pp. 857-869).
15. Le Thi Kim Nhung, H., **Hoc, H. T.**, & Van Hai, V. (2020). An evaluation of technical and environmental complexity factors for improving use case points estimation. In *Software Engineering Perspectives in Intelligent Systems: Proceedings of 4th CoMeSySo 2020, Vol. 1 4* (pp. 757-768).
16. Hai, V. V., Nhung, H. L. T. K., & **Hoc, H. T.** (2020). A Productivity optimising model for improving software effort estimation. In *Software Engineering Perspectives in Intelligent Systems: Proceedings of 4th CoMeSySo 2020, Vol. 1 4* (pp. 735-746).
17. Van Hai, V., Le Thi Kim Nhung, H., & **Hoc, H. T.** (2019). A review of software effort estimation by using functional points analysis. *Computational Statistics and Mathematical Modeling Methods in Intelligent Systems: Proceedings of 3rd CoMeSySo 2019, Vol. 2 3*, 408-422.
18. Nhung, H. L. T. K**., Hoc, H. T.**, & Hai, V. V. (2019). A review of use case-based development effort estimation methods in the system development context. Intelligent Systems Applications in Software Engineering: Proceedings of 3rd *CoMeSySo* 2019, Vol. 1 3, 484-499.
19. Dey, S.K., Hoang, D.S, **Hoc, H.T.**, & Pham, Q.G.N. (2022). ENGAGING VIRTUAL REALITY TECHNOLOGY TODETERMINE PRO-ENVIRONMENTAL BEHAVIOUR: THE INDIAN CONTEXT X. GeoJournal of Tourism and Geosites,41(2).
20. Dey, S.K., Hung, V.V., **Hoc, H.T.**, Pham, Q.G.N. (2022). AVR Technologies in Sustainable Tourism: A Bibliometric Review. In: Bashir, A.K., Fortino, G., Khanna, A., Gupta, D. (eds) Proceedings of International Conference on Computing and Communication Networks, vol 394. Springer, Singapore.
21. Nguyen T.T.N., Cartocci A., **Hoc H. T.**, Tong L. T., Mozafari M., Dang T.K., Nguyen T.Z., AI-aided automatic severity scoring system for Hidradenitis Suppurativa, 12th Hybrid Conference of the EHSF 2023 Hidradenitis Suppurativa / Acne Inversa-Tagung 2023.

Book Editor:

22. Zuzana Tučková, Sandeep Kumar Dey, **Hoc H. T.**, Sinh Duc Hoang, Impact of Industry 4.0 on Sustainable Tourism: Perspectives, Challenges and Future, published by Emerald, October, 2023.

CURRICULUM VITAE

Personal Information

Full name: Huynh Thai Hoc
Address: 30/2C Trung My Tan Xuan, Hoc Mon, Ho Chi Minh City, Vietnam
Nationality: Vietnamese
Orcid ID: 0000-0003-3845-8466
Scholar ID: xoesuc8AAAAJ
Email: huynh_thai@utb.cz; hoc.ht@vlu.edu.vn; huynhhoc@gmail.com

Work Experiences

- July 2023 – September 2023: Lead researcher for Internal Geospatial Data Science Bootcamp at Valhko company, France.
- March 2022 – January 2023: Internship at Torus Actions, Toulouse, France.
- 2018 – ongoing: Lecturer at the Faculty of Information Technology, School of Engineering and Technology, Van Lang University, HCMC, Vietnam.
- 2011 – 2018: Lecture at Faculty of Information Technology, University of Industry (UIH), Ho Chi Minh City, Vietnam.
- 2014 – January 2019: Developer at Capgemini Vietnam, HCMC, Vietnam.
- 2007 – 2014: Lecture at Faculty of Information Technology, University of Natural Resources and Environment (HCMUNRE), HCMC, Vietnam.
- 2002 – 2007: GIS developer at DITAGIS, HCMC, Vietnam.

Education

- 2019 – 10/2023: PhD scholar at the Faculty of Applied Informatics, Tomas Bata University, Zlin, the Czech Republic.
- 2004 – 2007: master's degree in Geographic Information Systems, University of Technology (HCMUT), Ho Chi Minh City, Vietnam.
- 1998 – 2002: bachelor's degree in mathematics and computer Science, University of Science (HCMUS), Ho Chi Minh City, Vietnam.

Programming Languages

- R programming, Python
- C/C++; Core Java; .NET

Data scientist skills

- Pytorch, Tensorflow
- Random Forest, XGBoost, SVM, Regression models, ensemble
- LIME/SHAP, Generative Models

Technical skills

- SQL Server, PostgreSQL/PostGIS, MongoDB
- .NET
- Design patterns (MVC), Web service (Restful, SOAP)

Research Interests

Data Scientist, GIS, Developer.

Research Activities at Tomas Bata University, Zlin

- IGA projects and Competition projects.

Regression Models for Software Project Effort Estimation

Regresní modely pro odhad úsilí softwarového projektu

Doctoral Thesis Summary

Published by: Tomas Bata University in Zlín,
nám. T. G. Masaryka 5555, 760 01 Zlín, the Czech Republic

Edition: published electronically
Typesetting by: Huynh Thai Hoc, Ph.D.
This publication has not undergone any proofreading or editorial review.
First Edition
Publication year: 2023

ISBN 978-80-7678-221-1