Journal of
Theoretical and Applied Electronic Commerce Research

MDPI

Article

Delving into Human Factors through LSTM by Navigating Environmental Complexity Factors within Use Case Points for Digital Enterprises

Nevena Rankovic [1,*] and Dragica Rankovic [2]

[1] Department of Cognitive Science and Artificial Intelligence, Tilburg University, 5037 AB Tilburg, The Netherlands
[2] Department of Mathematics, Informatics and Statistics, Union University "Nikola Tesla", 18000 Nis, Serbia; dragica.d.rankovic@gmail.com
* Correspondence: n.rankovic@uvt.nl; Tel.: +31-6-87-69-8997

Abstract: Meeting customer requirements in software project management, even for large digital enterprises, proves challenging due to unpredictable human factors. It involves meticulous planning and environmental factor analysis, ultimately benefiting both companies and customers. This paper came as a natural extension of our previous work where we left ourselves curious about what impact environmental complexity factors (ECFs) have in a use case point (UCP) approach. Additionally, we wanted to possibly decrease the mean magnitude relative error (MMRE) with deep learning models such as long-short-term-memory (LSTM) and gradient recurrent unit (GRU). The data augmentation technique was used to artificially increase the number of projects, since in the industry world, digital enterprises are not keen to share their data. The LSTM model outperformed the GRU and XGBoost models, while the average MMRE in all phases of the experiment for all models achieved 4.8%. Moreover, the post-agnostic models showed the overall and individual impact of eight ECFs, where the third ECF "team experience" on a new project has been shown as the most influential one. Finally, it is important to emphasize that effectively managing human factors within ECFs in UCPs can have a significant impact on the successful completion of a project.

Keywords: environmental complexity factors; digital enterprises; use case points (UCPs); recurrent neural networks (RNNs); post-agnostic models

Citation: Rankovic, N.; Rankovic, D. Delving into Human Factors through LSTM by Navigating Environmental Complexity Factors within Use Case Points for Digital Enterprises. *J. Theor. Appl. Electron. Commer. Res.* **2024**, *19*, 381–395. https://doi.org/10.3390/jtaer19010020

Academic Editor: Saïd Assar

Received: 24 September 2023
Revised: 26 November 2023
Accepted: 9 February 2024
Published: 14 February 2024

1. Introduction

Newly founded or small enterprises are particularly vulnerable to misguided human resource assessments since they often have limited resources in general. Through thorough project planning and a detailed investigation of environmental complexity factors within digital enterprises, the success of projects can be significantly increased [1,2]. Additionally, accurate software development effort estimation can improve project management and cost effectiveness, which is advantageous to both tech companies and their customers [3,4]. Both project clients and project managers place a high value on project development time. The quantity of money needed to invest in a project determines whether it will begin and whether it will be completed within the specified time range. To satisfy consumer needs, digital enterprises also employ, besides human expertise, a range of software tools and services [5]. There are numerous ways to gauge the size, complexity, and development time of software. For determining the actual size of a software project, the Use Case Point approach is frequently employed. This method takes into account the system's use cases when evaluating the amount of work needed to implement it. To accurately estimate the required resources, it examines system users and various circumstances. It uses 21 parameters for evaluation, of which 8 are environmental complexity factors and 13 are system technical qualities [5,6].

This research represents a natural extension of our previous work [4], in which we were intrigued by understanding the impact of environmental complexity factors within the UCP approach. Our goal was to optimize and further reduce the model's relative error by gaining deeper insights into these environmental complexity factors, including compliance with the used development process, experience with applications, team proficiency in technologies, the capabilities of the chief analyst, team motivation, stability requirements, adaptation of team members' working hours, and complexity of the programming language. Properly managing these factors can significantly decrease the likelihood of project failure, especially when utilizing a single machine learning model and a subset of deep learning models, particularly recurrent neural networks (RNNs). Additionally, employing data augmentation techniques to artificially increase the instances in the UCP Benchmark Mendeley dataset [7] helped us conduct our experiment more effectively, as it is well-known that the industry often hesitates to share data, and this approach provided a viable solution. Furthermore, in pursuit of an interpretable component for our top-performing model, we utilized SHAP and LIME post-agnostic models. As a result of these advancements, software project management could become more dependable and successful, benefiting the digital industry and its clients.

The following research objectives are the primary goals that guided us:

RO1: Increase the dataset using data augmentation techniques and further reduce MMRE within the UCP approach by employing deep learning models such as LSTM and GRU.
RO2: Examine the overall impact of ECF within the UCP approach by tracking human resources transactions.
RO3: Determine, using SHAP and LIME, which of the 8 ECF factors is the most influential on the best-performing model.

The rest of the paper is organized as follows: Section 2 delves into the state-of-the-art literature related to advancements in the UCP field. Section 3 presents the experimental setup used in the research methodology. Section 4 showcases the obtained results, while Section 5 discusses them. Concluding remarks are provided in Section 6.

2. Related Work

In this section, we will showcase cutting-edge research rooted in the UCP approach in software project management, highlighting its effective application within digital enterprises. We will explore its standalone effectiveness and its synergistic use in ensemble models, aiming to further enhance accuracy and reliability.

The most contemporary widely used method for estimating the time and costs involved in creating software solutions within digital enterprises is definitely UCP. The lowest relative estimating error when utilizing only this method is around 10% [8]. An estimating error rate of 7.5% was achieved in the study [4] by combining Taguchi's optimization method with this strategy. However, other researchers [9–11] have merged this approach with additional parametric models and various artificial intelligence models. In one study, using Android mobile applications as a case study, the UCP technique was used to estimate the size and effort needed for mobile applications. A modified variation of the UCP approach was additionally introduced [12]. A framework for UCP-based techniques was introduced by the authors in [13] in order to improve reusability in the creation of software applications. Their conclusions showed that the framework effectively met five requirements for quality and showed how it might be used throughout the early stages of software development. A systematic review was carried out in [14] to find publications demonstrating best practices in the field of effort estimating methods that included both UCP and expert judgment-based approaches. Numerous models, including the UCP method and neuro-fuzzy logic, were also researched to improve the estimation's accuracy [15]. It was found that the Neuro-fuzzy logic model with updated use case points and modified environmental factors offers the best fitting accuracy at an early stage when compared to other models. In seven actual software development repositories, the advantages of various work estimating methods were examined. The traditional UCP approach and iUCP, an HCI

(human-centric) augmented model, were compared in this study. They also presented an improved iteration of the initial iUCP effort estimation formula [16]. Introducing "UCP for IoT" or adapting the use case points method to estimate the size and effort for IoT systems was presented in [17]. It was validated with a case study of three IoT systems, demonstrating applicability and highlighting the need for further improvement and data collection in future work.

The possibility of utilizing ensemble ML models to improve software effort and cost estimation was demonstrated in earlier work by [3]. The study [6] The presents a powerful ensemble model, merging seven statistical and machine learning techniques including K-nearest neighbor, random forest, support vector regression, multilayer perception, gradient boosting, linear regression, and decision tree with grid search optimization, showcasing promising estimation accuracy across four datasets.

Another interesting approach was to further elucidate the relationship between the impact of data locality on productivity within use case points (UCPs), employing environmental factors for data segmentation and clustering algorithms to create homogeneous subsets. It introduces an ensemble approach using three regression models, each informed by identical training sets, to enhance prediction. The resulting productivity forecasts are refined through weighted averages from each model's output, presenting a nuanced understanding of UCP variables' interplay [18].

The study introduces a fuzzy cognitive mapping (FCM) method to identify optimal machine learning strategies for estimating Web application projects. FCM capitalizes on the interrelated dynamics of system variables, tailoring recommendations to a project's specific configuration. Addressing the vagueness inherent in system variable interactions, the paper advocates the integration of fuzzy numbers into the FCM framework. This enhanced approach yields a promising 70% probability of successfully recommending software estimation techniques, providing a strategic tool for managing the complexities of Web application development [19]. Besides the recurrent neural networks underpinning FCM models, one study proposed a hybrid approach combining particle swarm optimization with a deep learning model to improve the evaluation metrics of the approach [20]. Furthermore, an artificial bee colony guided analogy-based estimation (BABE) model was introduced, which combines the artificial bee colony (ABC) algorithm with analogy-based estimation (ABE) for more accurate estimations. The limitation of this model, however, is its reliance solely on variables used in function point analysis, which do not always capture the complexity and size determined by the intricacies of the actors and the systems they interact with [21].

In software estimation and related areas, it is usual to carry out substantial meta-analyses that typically encompass correlated estimates of effect size [22,23]. The authors in [23] evaluated meta-heuristic algorithms, specifically grey wolf optimizer (GWO) and strawberry (SB), to enhance a deep neural network model for software effort estimation. Testing on nine benchmark functions, the GWO outperforms other methods, proving more accurate in its estimations. Generally, approaches to network meta-analysis used for quantitative data synthesis employ Bayesian optimization for these purposes [24].

Ultimately, in [25] a detailed analysis underscores the significance of accurate software effort estimation for project success within budgets and deadlines. It addresses the issues of overestimation and underestimation and evaluates 35 studies using mean magnitude of relative error and PRED (25) to compare ensemble and solo machine learning techniques, with ensembles frequently providing superior accuracy.

While prior research has primarily focused on improving estimation accuracy by integrating the UCP approach with traditional (parametric) or artificial intelligence (non-parametric) models, it has become evident that there is a gap in the literature. There is a dearth of studies investigating the potential of harnessing deep learning models, especially LSTM, for this purpose, as well as employing data augmentation techniques to artificially expand the dataset, given the challenge of obtaining real data from the industry. Additionally, model-agnostic approaches like SHAP are not mentioned. This innovative

approach ensures that even when the software industry lacks readily available data, precise estimation outcomes can still be achieved in the domain of software project management for digital enterprises.

3. Methodology

In this Section, we will describe the experimental setup of our methodology. Graphical representation of the methodology pipeline is presented in Figure 1.

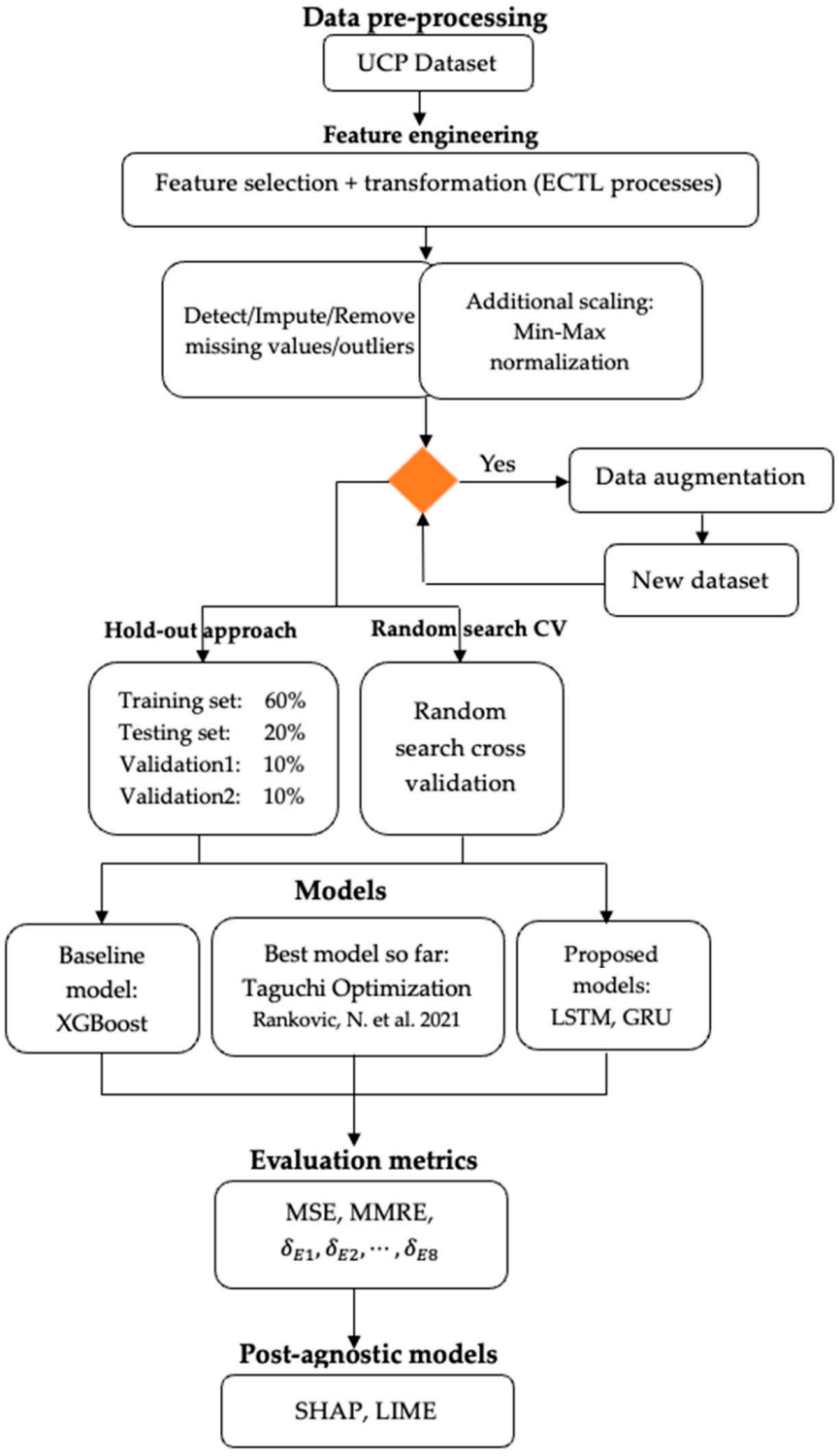

Figure 1. Methodology pipeline [4].

3.1. Dataset Description

For experimental purposes, we used the Use Case Point Benchmark Dataset [7], compiled by Radek Silhavy from three different software companies. UCP, extensively utilized in software project management to determine the size of a software project, is based on a technique created by Karner in 1993 [26]. This technique involves analyzing system use cases to determine the effort required for implementation while considering both the technical and environmental aspects of the system. It utilizes a set of 21 parameters, including 13 technical and 8 environmental complexity factors. The original dataset contains 71 projects with 26 columns, where 4 input quantities are formed/calibrated based on 21 input parameters that represent technical factors and environmental factors. The first column indicates the project ID, the next four columns are the obtained values of UAW, UUCW, TCF, and ECF. The sixth column is the measured real value of the project, and the following 21 columns represent individual values of T1 . . . T13 (technical factors) and E1 . . . E8 (environmental factors). The aggregated 21 factors are reduced to a normalized 6-dimensional vector through the ETL process in the Pentaho Spoon Data Integration tool for visualization purposes [27]. After that, with an SQL query we created a View that provides 6 main input variables, 4 linearly independent and 2 linearly dependent. After that, the range of values for 8 ECFs is observed through user system transactions and use cases Figure 2.

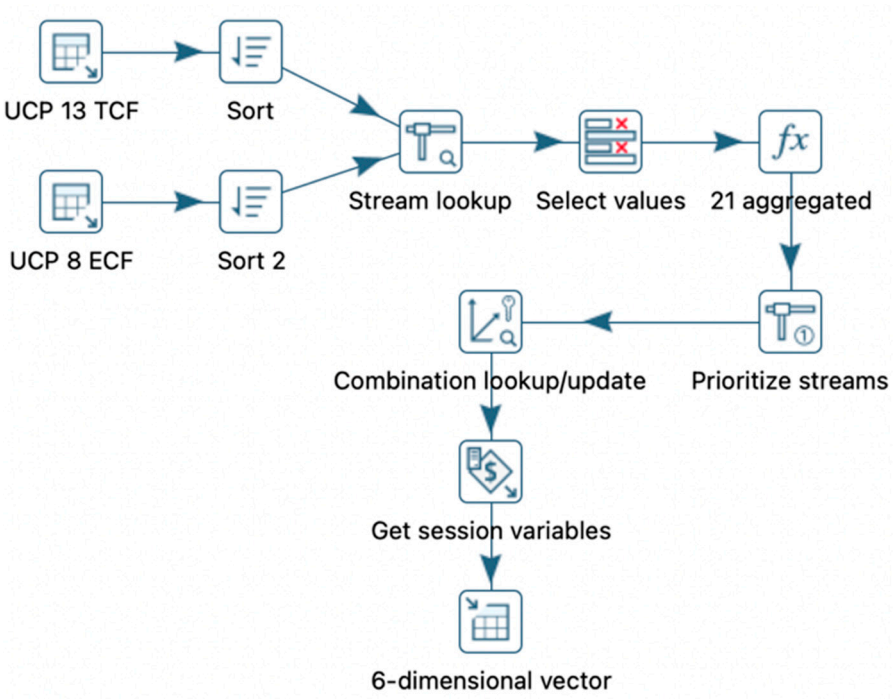

Figure 2. Data pre-preparation and organization in Pentaho Spoon.

System users and use cases are jointly utilized to determine the actual size using the UCP method. System users are categorized into three groups based on their interaction with the system: simple (assigned a weight factor of 1 depending on system interaction), average (assigned a weight factor of 2 depending on internal/external communications), and complex (assigned a weight factor of 3 depending on the complexity of interactions).

Additionally, there are three categories of use cases defined based on the number of transactions executed (number of users and the system for message transmission): simple (assigned a weight factor of 5 for fewer than 3 transactions), average (assigned a weight factor of 10 for 4 to 7 transactions), and complex (assigned a weight factor of 15 for more than 8 transactions).

Moreover, a transaction where data or control information are transferred between an actor (which could be a user or another system) and the system under observation is commonly referred to as a transaction. It is a discrete unit of interaction that can be characterized as an entire series of actions that accomplish an actor's objective.

Real effort is represented by the UCP approach as a 6-dimensional vector Table 1, and its value is determined by calculating the vector's norm [4,26], Formulas (1) and (2):

$$\text{UCP} = (\text{UAW}, \text{UUCW}, \text{UUCP}, \text{TCF}, \text{ECF}, \text{AUCP}) \quad (1)$$

$$\left\| \vec{\text{UCP}} \right\| = \text{UAW} + \text{UUCW} + \text{UUCP} + \text{TCF} + \text{ECF} + \text{AUCP} \quad (2)$$

where UUCP = UAW + UUCW, and AUCP = UUCP $\times$ TXF $\times$ ECF.

Table 1. Input features.

Feature Name UCP Model	Original Type	Description
Unadjusted Actor Weight (UAW)	Numerical	Point size of the software that accounts for the number and complexity of actors
Unadjusted Use Case Weight (UUCW)	Numerical	Complexity and size of the use cases
Unadjusted Use Case Point (UUCP)	Numerical	Unadjusted use case point
Technical Complexity Factor (TCF)	Numerical	Factor that is used to adjust the size based on technical considerations
Environmental Complexity Factor (ECF)	Numerical	Factor that is used to adjust the size based on the considerations
Adjusted Use Case Point (AUCP)	Numerical	Adjusted use case point

ECF is calculated as follows [4,26], Formulas (3) and (4):

$$\text{ECF} = 1.4 + (-0.03 \times \text{FactorE}) \quad (3)$$

$$\text{FactorE} = \sum \text{Weight} * \text{AssignedValue}, \quad (4)$$

where AssignedValue from 0 to 5 and represents a environmental factor of the estimated process Table 2.

Table 2 is an example of calculating one ECF for a single project from the original dataset, using the Karner formula. The total ECF value can have different values for different projects, but within the range of [0.57; 1.12].

Table 2. Eight environmental factors—example.

Factor	Description	Weight	Assigned Value	Weight $\times$ Assigned Value
E1	Compliance with the used development process	1.5	3	4.5
E2	Experience with applications	0.5	3	1.5
E3	Team experience with technologies	1.0	3	2
E4	Capabilities of the chief analyst	0.5	5	2.5
E5	Team motivation	1.0	2	2
E6	Stability of requirements	2.0	1	2
E7	Part-time staff	-1.0	0	0
E8	Programming language complexity	-1.0	4	-4
	Total (EF):			10.5

For each dataset, a weight factor is assigned depending on the number of transactions. Increasing the number of transactions increases the value of FactorE, which leads to a decrease in the overall ECF value, thereby increasing the risk and model error. The same holds true in reverse. Changing the number of transactions between system users and use cases has a significant impact on the ECF value. For each real project, the value can be calculated: δ_{E1}, $\delta_{E2}, \cdots, \delta_{E8}$ and represents the contribution or share of each individual factor. E_i, $i = \overline{(1,\ 8)}$ as follows (5):

$$\text{Factor}E_i = \frac{\text{ECF} - 1.4}{0.03} \qquad i = \overline{(1,\ 8)} \tag{5}$$

The View component in the Pentaho tool is designed to execute queries whenever specific data need to be selected. This improves performance, particularly when it is necessary to optimize the database system. Moreover, using Views ensures consistency in the way data are queried and presented. Any changes can be made by altering the View's definition rather than modifying the SQL query directly. This is beneficial when performing transformations or generating reports. Furthermore, this method of presenting, organizing, and structuring data aligns with business concepts, not just database structures, because the Views are based on the logic defined within them.

Given that we had aggregated data, the ETL process enabled us to perform the necessary transformations to ascertain the scope of ECF factors, that is, to determine their minimum and maximum values within the newly aggregated data. From Table 3, the range of total ECF values is [0.71; 1.12] for the original dataset and in Table 4, it is [0.57; 1.08]. The significance of the ETL process in data analysis lies in its capability to efficiently manage large datasets, ensuring that data from various sources are accurately combined, cleaned, and transformed into a consistent format that is ready for analysis. In this context, the ETL process was crucial in consolidating disparate data sources, thereby allowing us to derive meaningful insights about the range of ECF factors. This range is vital as it represents the variability within the data which could have significant implications for the interpretation of the ECFs and their subsequent impact on the analysis or models that rely on these data. Moreover, the ETL process is not just about preparing data for analysis; it is also about maintaining data integrity and quality. Transforming the aggregated data through ETL ensured that the resulting datasets in Tables 3 and 4 were reliable and that the ECF values reflected true variations rather than discrepancies caused by data entry errors, missing values, or incompatible data formats. Therefore, the ETL process is not a mere step in data preparation; it is a comprehensive approach that enhances the robustness of data analysis, enabling us to derive accurate and actionable insights from the aggregated data. The clear definition of the ECF range is a testament to the meticulous data processing performed through ETL, underscoring its indispensable role in the analytical workflow. This explanation underscores the critical role that ETL processes play in preparing data for analysis, ensuring its quality, and supporting the derivation of accurate insights from data, which is especially important when working with aggregated datasets that can come from various sources and can contain a wide variety of data types and formats

Table 3. Exploratory data analysis—old datasets.

Datasets	ECF	N	Min Value	Max Value	Mean	Standard Deviation
Dataset_1	[0.71; 1.08]	50	5775.0	7920.0	6506.9	653.0
Dataset_2	[0.94; 1.12]	21	6162.6	6525.3	6393.9	118.2
Dataset_3	[0.71; 1.12]	18	2692.1	3246.6	2988.4	233.2
Dataset_4	[0.71; 1.08]	17	2176.0	3216.0	2589.4	352.1

Table 4. Exploratory data analysis—new datasets.

Datasets	ECF	N	Min Value	Max Value	Mean	Standard Deviation
Dataset_1	[0.57; 1.08]	648	4892.3	6548.1	5402.5	538.2
Dataset_2	[0.94; 1.08]	216	5430.7	7123.4	6208.4	456.7
Dataset_3	[0.71; 1.12]	108	43,890.4	6291.3	5467.8	652.9
Dataset_4	[0.57; 1.08]	108	2856.7	4775.6	3818.9	438.0

3.2. Dataset Pre-Processing

To gain even better insights about environmental complexity factors which are related to human factors such as compliance with the used development process, experience with applications, team proficiency in technologies, the capabilities of the chief analyst, team motivation, stability requirements, adaptation of team members' working hours, and complexity of the programming language, we expanded our dataset from 71 [4] to 1080 instances by applying a data augmentation technique [28], recognizing the potential for overfitting with such a small dataset [29]. The range of ECF factor values in the old dataset Table 3 across all phases is [0.71; 1.12], while in the new dataset Table 4, it is [0.57; 1.08].

To achieve this, we employed a data augmentation technique known as the 'Truncated Normal Distribution' [30]. This method involves generating synthetic data points by sampling from a truncated normal distribution tailored to the statistical characteristics of the original dataset. Initially, in dealing with missing data, a prevalent strategy is to impute missing values with the mean value of the corresponding attribute. Considering the heterogeneous nature of the dataset, i.e., projects differ in terms of size, programming languages used, technologies, etc., it becomes imperative to employ the min–max normalization scaling technique [31,32] within a specific narrow range [0, 1]. This ensures the creation of a new scaled dataset based on the original, effectively harmonizing the varied parameter units. After selecting the features, removing the outliers, and conducting the min–max normalization technique along with data augmentation technique the new datasets revealed that there are no missing values. For each input value, the minimum, maximum, and mean values, along with the standard deviation, were provided for the 'Real Effort' values.

Ultimately, the 'Truncated Normal Distribution' as a data augmentation technique can be particularly advantageous in the UCP method of software estimation, which relies on numerical data to assess software project size and complexity. In UCP, handling outliers is crucial, as atypical use cases can disproportionately affect the estimation. The truncation aspect of this technique ensures that the range of data considered for software estimation is realistic, aligning with actual use case scenarios and preventing the skewing of estimations that could result from the inclusion of unrealistic outliers that are sometimes present in a standard normal distribution. Maintaining the properties of a normal distribution within the constraints of real-world data ensures that the UCP method reflects the true scope and complexity of the software project. It enables more accurate modeling of the use cases by reflecting real-world limits on use case complexity, thus enhancing the robustness of the model, and ensuring that the estimations are representative of the range of likely scenarios. By ensuring data consistency in the use case estimations and allowing for customization based on the specific characteristics of the project, the truncated normal distribution aligns with the practical needs of the UCP approach. This tailored fit makes it more suitable for the UCP method than other data augmentation techniques, which may introduce too much variability and undermine the accuracy of the software estimation process.

The train-test split utilized a hold-out approach combined with random search cross-validation. After conducting numerous trial-and-error experiments, we found that the 60:20:10:10 split yielded the most favorable results. This allocation comprised 60% for the training set, 20% for the test set, and two separate validation sets, each containing 10% of the data.

3.3. Model Descriptions

XGBoost will represent a baseline model in this experiment. In the context of investigating human factors for successful project completion, XGBoost harnesses the power of gradient tree boosting and excels at handling software-related data analysis in the context of examining human variables for software project success [33]. It is highly suited for the complexity of software project management since it provides quick computations, excellent prediction accuracy, and built-in safeguards for preventing overfitting [34]. This area benefits especially from XGBoost's capacity to handle complex relationships, deal with outliers, and deal with imbalanced data. It uses boosting approaches to maximize predictions while being guided by an objective loss function, demonstrating its skill in spotting complex connections and patterns in software estimation data [35]. The following parameters have been tuned for the analysis's needs:

1. *subsample*: denotes the fraction of observations to be randomly sampled for each tree;
2. *colsample_bytree*: the subsample ratio of columns when constructing each tree;
3. *max_depth*: the maximum depth of a tree;
4. *min_child_weight*: defines the minimum sum of weights of all observations required in a child;
5. *learning_rate*: the shrinkage made at every step.

A particular type of RNN, known as the LSTM network, is also well-suited for the task of effective software project management. It was created to address the problem of vanishing gradients, which typically occurs when dealing with long-term dependencies in standard RNNs [36]. The modeling and forecasting of time series data using LSTM have proven to be remarkably effective, adding significant value when estimating project length. This neural network architecture excels at retaining information from previous states and making accurate predictions based on the complex patterns present in the data [37]. Given that prior states must be considered to provide accurate predictions in scenarios with prolonged dependencies, LSTM emerges as the optimal choice. Moreover, LSTM offers a wide range of parameters that can be fine-tuned [38]. In our investigation, we selected the following parameters:

1. defining the number of LSTM units;
2. defining the number of LSTM layers;
3. defining the dropout rate;
4. determining LSTM's time step input.
5. *learning_rate*: the shrinkage made at every step.

GRU models, which belong to the RNN family and are particularly adept at managing sequential data, are an invaluable tool for software project management. In project estimation for digital enterprises, where past trends and contextual knowledge are vital for making precise predictions, GRU models are created to capture and remember long-range connections in data. GRUs have gained popularity recently as a result of their sleek and effective design. While they might not always outperform LSTM in terms of performance, they typically produce outcomes that are competitive while drastically cutting training times [39,40].

3.4. Evaluation Metrics

In this experiment, we employ MSE, MRE, and MMRE, as defined by Formulas (6)–(8), as the metrics to assess the model performances [41,42]. These criteria allow us to thoroughly evaluate the effectiveness of the models in forecasting target values. The precision of regression models is commonly assessed using MSE, a widely recognized statistic. However, it is important to note that the interpretation of MSE may be influenced by the scale of the target variable. To gain deeper insights into the model's prediction

performance and to provide a more easily understandable measure of error, we also utilize both MSE and MMRE for the final observations.

$$\text{MRE} = \frac{1}{n} \cdot \sum_{i=1}^{n} \text{MRE}_i \tag{6}$$

$$\text{MSE} = \frac{1}{n} \cdot \sum_{i=1}^{n} (\text{MRE}_i)^2 \tag{7}$$

$$\text{MMRE} = \text{mean (MRE)} \tag{8}$$

3.5. Post Agnostic Models: SHAP i LIME

Machine learning and deep learning models are viewed as inaccessible functions when using post-agnostic model explanation techniques like SHAP (shapley additive explanations) and LIME (local interpretable model-agnostic explanation). These methods do not need access to internal information like the neural network's structure, learning parameters (weights, biases), or activation levels; instead, they just rely on the model's output. Because of this quality, model-agnostic techniques like SHAP and LIME are flexible and adaptable to different kinds of deep learning and machine learning models [43–45]. Without explicit understanding of the model's internal mechanics, we can learn more about the role and importance of characteristics in the decision-making process by using SHAP as a post hoc method. The SHAP and LIME approach with the top-performing model formulas will be used as follows (9) and (10):

$$\text{SHAP}_{v_i} = \sum_{i=1}^{n} \text{MDk}_i \cdot T_k \tag{9}$$

where SHAP_{vi} represents the SHAP value for the *i*th data instance, represents the marginal contribution of feature (variable) *ki* in the prediction for the *i*th instance, represents the weight of feature *k* included in the Shapley sum [46].

With the use of a Gaussian (RBF) kernel, LIME assigns weights to each generated point. The size of the meaningful weights' circle around the red dot is determined by the kernel width *kw* option [47].

$$\text{RBF}\left(x^{(i)}\right) = \exp\left(-\frac{\left\|x^{(i)} - x^{(ref)}\right\|^2}{kw}\right) \tag{10}$$

4. Results

In this section, we present the findings from our conducted experiment. We start with a comparison of the results among the used models, along with the selected hyperparameter combinations and evaluation metrics. In our study, the target variable had a range of values between 2856.7 and 6548.1. From Table 5, it is evident that the LSTM model achieved the best MRE value of 0.992 on the testing set, and on the training set, it achieved 0.983. Meanwhile, it achieved 0.984 on the first validation set and 0.980 on the second validation set.

Table 5. Model performances.

Models	Training		Testing		Validation1		Validation2	
	MRE	**MSE**	**MRE**	**MSE**	**MRE**	**MSE**	**MRE**	**MSE**
XGBoost	0.935	252.29	0.915	257.80	0.923	255.57	0.931	253.37
Taguchi method	0.933	252.82	0.929	253.92	0.920	256.40	0.917	257.24
LSTM	0.983	239.98	0.992	236.84	0.984	239.73	0.980	240.70
GRU	0.971	242.94	0.980	240.70	0.973	242.44	0.968	243.69
MMRE	0.955		0.954		0.950		0.949	

The GRU network achieved slightly lower results, with an MRE value of 0.980 on the testing dataset and 0.971 on the training dataset. On the first and second validation sets, it achieved values of 0.973 and 0.968, respectively. The optimal number of decision trees for XGBoost was set to n_estimators = 500. The number of units/neurons and layers are the two most critical factors in determining the architecture of deep learning models. We used 64 units for both LSTM and GRU, and the number of layers that provided the highest accuracy was 2. This decision was influenced by the relatively limited dataset, which, after applying data augmentation techniques, contained 1080 cases. The best MMRE value for all algorithms was 0.955 and was achieved during the training phase.

The influence $\delta(E_i)$ of individual ECFs can be seen in Table 6. The factor E3 = "Team experience with technologies", has the greatest impact, averaging 1.9% of the total share of ECFs. In second place is E6 = "Stability of requirements" with 1.0% of the total share of ECFs. The least impact is observed for E7 = "Part-time staff", and E8 = "Programming language complexity", each contributing only 0.1% to the total share of ECFs.

Table 6. The influence $\delta(E_i)$ in total share of ECFs.

ECF Factors	Training	Testing	Validation1	Validation2	Total
	$\delta D1$	$\delta D2$	$\delta D3$	$\delta D4$	$\delta D5$
E1	0.7	0.6	0.7	0.6	0.7
E2	0.5	0.5	0.5	0.6	0.5
E3	1.7	1.8	1.9	2.0	1.9
E4	0.2	0.3	0.4	0.4	0.3
E5	0.1	0.2	0.3	0.2	0.2
E6	0.9	1.0	1.1	1.2	1.0
E7	0.1	0.1	0.1	0.1	0.1
E8	0.1	0.2	0.1	0.1	0.1
Total	4.3	4.7	5.1	5.2	4.8

Our SHAP-based analysis reaffirmed the significance of the ECF factor, specifically E3 = "Team experience with technologies", in the context of the best-performing model, LSTM, as illustrated in Figure 3. Moreover, utilizing the LIME method on LSTM, we acquired insights into the impact of each ECF on the variation of MMRE values, as depicted in Figure 4.

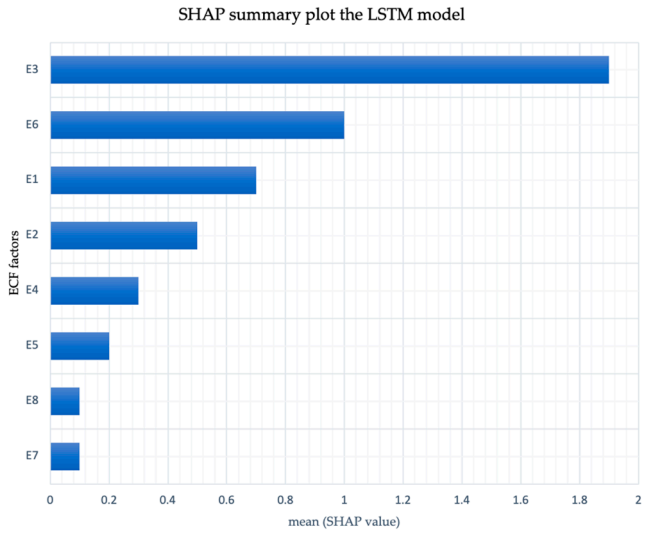

Figure 3. SHAP method on LSTM.

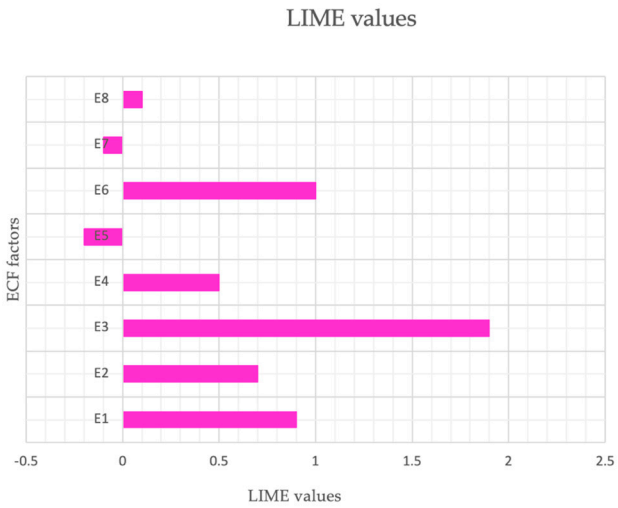

Figure 4. LIME method on LSTM.

5. Discussion

This section discusses the experimental results by addressing the research questions outlined in the introduction.

The first research objective aimed to expand the original dataset using augmentation techniques, given the challenge in obtaining real industry data, as companies are often reluctant to share such information. Additionally, the objective was to further reduce the model error, specifically MMRE, in comparison to the best result achieved in prior research [4]. The applied data augmentation techniques contributed to a reduction in MMRE for all models used throughout the experiment, including XGBoost as the baseline model. In the previous study, utilizing the Taguchi optimization technique, the minimum MMRE achieved was 7.5% with two different artificial neural network architectures. In contrast, in this experiment, the MMRE for all proposed models averaged 4.8%, with the best-performing model being LSTM, which achieved an accuracy of 99.2%.

The second research objective was associated with the overall impact of ECF factors within the UCP approach, specifically the influence of human factors, skills, and knowledge within digital enterprises through transactions on successful project completion. While each ECF represents a portion of human resources, the most influential factor is skill and knowledge of the team, i.e., E3 = "Team experience with technologies", contributing to 1.9% of the total impact on successful project implementation within digital enterprises. Client requirements can be complex, especially when working on a new project, and this complexity is evident through the influence of E6 = "Stability of requirements" in the overall distribution of ECFS. Finally, SHAP confirmed the influential factors through an analysis of the best-performing LSTM model, while LIME revealed a negative influence of specific factors E_i = (E5 = "Part-time staff" and E7 = "Team motivation") on the overall distribution of ECF in terms of MMRE changes.

Limitations and Future Directions

Applying Shapley values to software project management indeed presents a significant challenge due to computational complexity, especially when dealing with modern models like deep neural networks with high-dimensional inputs. However, it is essential to emphasize that the intricacy of these models, particularly LSTM networks, introduces its own set of limitations that must be considered. LSTM networks, renowned for their ability to handle sequential data, have their constraints. They require substantial computational resources and can struggle when faced with intricate feature relationships [48,49]. These limitations can significantly impact their effectiveness in software project management predictions. Balancing the intricacies and limitations of both LSTM networks and Shapley value computations, future research in software project management should explore innovative and

integrated approaches to effectively address these challenges. Given these constraints, future research in project management with diverse datasets should explore deep learning models. Recurrent neural networks, notably fuzzy cognitive maps, demonstrate potential in handling missing data and extracting intricate patterns from extensive datasets [50,51]. Leveraging these models could help mitigate challenges related to missing data, ultimately enhancing the precision, reliability, and applicability of project management predictions by conducting additional 'what-if' simulations.

6. Conclusions

In conclusion, our comprehensive study in the domain of project management within digital enterprises yielded valuable insights. We successfully expanded our dataset through augmentation techniques, significantly reducing the MMRE across all models, with LSTM emerging as the top performer, achieving an impressive accuracy of 99.2%. Furthermore, our investigation highlighted the paramount role of human factors, particularly team expertise, in project success within digital enterprises. The meticulous analysis using SHAP and LIME affirmed these findings, shedding light on influential factors and offering a deeper understanding of their impact. This study not only enhances our understanding of digital project management but also underscores the critical importance of skilled teams in navigating the complexities of the digital landscape, paving the way for more successful endeavors and data-driven decision-making in future projects. Moreover, it reaffirms that, in the realm of digital project management, the human element remains an irreplaceable cornerstone for success, highlighting the need for continuous investment in developing and nurturing the skills and expertise of project teams. These findings stress that the collaborative synergy of skilled professionals is the driving force behind achieving excellence and innovation in digital enterprises.

The utilization of UCP approach offers tangible benefits for project management and requirements engineering. One significant benefit is the ability of managers to leverage UCP size metrics as a tool for forecasting productivity, and by extension, the efforts required for a project. This means that managers have the option to selectively use data that are the most representative, sharing similar environmental factor values, rather than relying on a broader data set. A positive aspect of this approach is the relative ease of measuring environmental factors during the initial project stages, which does not necessitate extensive expertise. Nonetheless, the creation of a comprehensive guide detailing the assessment of these factors is essential to minimize uncertainties in their measurement.

Another advantage of the UCP approach is its capacity to address the challenges of over or underestimating software development efforts in the early stages. With a more accurate forecast, managers can place more effective bids on software development projects. Our future directions will also be devoted to the subsequent research to delve into how local nuances and combined methodologies affect the accuracy of productivity predictions based on environmental factors.

Author Contributions: Both authors contributed equally to this research. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: This study is based on the publicly available dataset accessible via reference [7] provided in the manuscript. The code will be available upon reasonable request through the corresponding author's email: n.rankovic@uvt.nl.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Vavpotič, D.; Kalibatiene, D.; Vasilecas, O.; Hovelja, T. Identifying key characteristics of business rules that affect software project success. *Appl. Sci.* **2022**, *12*, 762. [CrossRef]
2. Khan, J.; Jaafar, M.; Mubarak, N.; Khan, A.K. Employee mindfulness, innovative work behaviour, and IT project success: The role of inclusive leadership. *Inf. Technol. Manag.* **2022**, 1–15. [CrossRef]
3. Marapelli, B.; Carie, A.; Islam, S.M. Software effort estimation with use case points using ensemble machine learning models. In Proceedings of the 2021 International Conference on Electrical, Computer and Energy Technologies (ICECET), Cape Town, South Africa, 9–10 December 2021; pp. 1–6. [CrossRef]
4. Rankovic, N.; Rankovic, D.; Ivanovic, M.; Lazic, L. A novel UCP model based on artificial neural networks and orthogonal arrays. *Appl. Sci.* **2021**, *11*, 8799. [CrossRef]
5. Rankovic, N.; Rankovic, D.; Ivanovic, M.; Lazic, L. A new approach to software effort estimation using different artificial neural network architectures and Taguchi orthogonal arrays. *IEEE Access* **2021**, *9*, 26926–26936. [CrossRef]
6. Nhung, H.L.T.K.; Van Hai, V.; Silhavy, P.; Prokopova, Z.; Silhavy, R. Incorporating statistical and machine learning techniques into the optimization of correction factors for software development effort estimation. *J. Softw. Evol. Process* **2023**, e2611. [CrossRef]
7. Silhavy, R. Use Case Points Benchmark Dataset. Mendeley Data V1. 2017. Available online: https://data.mendeley.com/datasets/2rfkjhx3cn/1 (accessed on 1 February 2021).
8. Carroll, E.R. Estimating software based on use case points. In Proceedings of the 20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications—OOPSLA '05, San Diego, CA, USA, 16–20 October 2005; pp. 257–265. [CrossRef]
9. Nassif, A.B.; Capretz, L.F.; Ho, D. Enhancing Use Case Points Estimation Method using Soft Computing Techniques. *J. Glob. Res. Comput. Sci.* **2010**, *1*, 12–21.
10. Azzeh, M. Fuzzy Model Tree for Early Effort Estimation Machine Learning and Applications. In Proceedings of the 12th International Conference on Machine Learning and Applications, Miami, FL, USA, 4–7 December 2013; pp. 117–121. [CrossRef]
11. Urbanek, T.; Prokopova, Z.; Silhavy, R.; Sehnalek, S. Using Analytical Programming and UCP Method for Effort Estimation. In Proceedings of the Modern Trends and Techniques in Computer Science; Advances in Intelligent Systems and Computing. Springer: Berlin/Heidelberg, Germany, 2014; Volume 285, pp. 571–581. [CrossRef]
12. Kaur, A.; Kaur, K. Effort Estimation for Mobile Applications Using Use Case Point (UCP). In *Smart Innovations in Communication and Computational Sciences*; Springer: Singapore, 2019; pp. 163–172. [CrossRef]
13. Ani, Z.C.; Basri, S.; Sarlan, A.A. Reusability assessment of UCP-based effort estimation framework using object-oriented approach. *J. Telecommun. Electron. Comput. Eng.* **2017**, *9*, 111–114.
14. Mahmood, Y.; Kama, N.; Azmi, A.A. Systematic review of studies on use case points and expert-based estimation of software development effort. *J. Softw. Evol. Process.* **2020**, *32*, e2245. [CrossRef]
15. Gebretsadik, K.K.; Sewunetie, W.T. Designing Machine Learning Method for Software Project Effort Prediction. *Comput. Sci. Eng.* **2019**, *9*, 6–11.
16. Alves, R.; Valente, P.; Nunes, N.J. Improving software effort estimation with human-centric models: A comparison of UCP and iUCP accuracy. In Proceedings of the 5th ACM SIGCHI Symposium on Engineering Interactive Computing Systems, London, UK, 24–27 June 2013; pp. 287–296. [CrossRef]
17. Silhavy, R.; Bures, M.; Alipio, M.; Silhavy, P. More Accurate Cost Estimation for Internet of Things Projects by Adaptation of Use Case Points Methodology. *IEEE Internet Things J.* **2023**, *10*, 19312–19327. [CrossRef]
18. Azzeh, M.; Nassif, A.B.; Martín, C.L. Empirical analysis on productivity prediction and locality for use case points method. *Softw. Qual. J.* **2021**, *29*, 309–336. [CrossRef]
19. Pandey, P.; Litoriya, R. Fuzzy Cognitive Mapping Analysis to Recommend Machine Learning-Based Effort Estimation Technique for Web Applications. *Int. J. Fuzzy Syst.* **2020**, *22*, 1212–1223. [CrossRef]
20. Sreekanth, N.; Rama Devi, J.; Shukla, K.A.; Mohanty, D.K.; Srinivas, A.; Rao, G.N.; Alam, A.; Gupta, A. Evaluation of estimation in software development using deep learning-modified neural network. *Appl. Nanosci.* **2023**, *13*, 2405–2417. [CrossRef]
21. Shah, M.A.; Jawawi, D.N.A.; Isa, M.A.; Younas, M.; Abdelmaboud, A.; Sholichin, F. Ensembling Artificial Bee Colony with Analogy-Based Estimation to Improve Software Development Effort Prediction. *IEEE Access* **2020**, *8*, 58402–58415. [CrossRef]
22. Pustejovsky, J.E.; Tipton, E. Meta-analysis with Robust Variance Estimation: Expanding the Range of Working Models. *Prev. Sci.* **2022**, *23*, 425–438. [CrossRef] [PubMed]
23. Khan, M.S.; Jabeen, F.; Ghouzali, S.; Rehman, Z.; Naz, S.; Abdul, W. Metaheuristic Algorithms in Optimizing Deep Neural Network Model for Software Effort Estimation. *IEEE Access* **2021**, *9*, 60309–60327. [CrossRef]
24. Shim, S.R.; Kim, S.J.; Lee, J.; Rücker, G. Network meta-analysis: Application and practice using R software. *Epidemiol. Health* **2019**, *41*, e2019013. [CrossRef] [PubMed]
25. Mahmood, Y.; Kama, N.; Azmi, A.; Khan, A.S.; Ali, M. Software effort estimation accuracy prediction of machine learning techniques: A systematic performance evaluation. *Softw. Pract. Exper.* **2022**, *52*, 39–65. [CrossRef]
26. Gustav, K. *Resource Estimation for Objectory Projects*; Objective Systems SF AB: Kista, Sweden, 1993; pp. 1–9.
27. Maja, M.M.; Letaba, P. Towards a data-driven technology roadmap for the bank of the future: Exploring big data analytics to support technology roadmapping. *Soc. Sci. Humanit. Open* **2022**, *6*, 100270. [CrossRef]

28. Maharana, K.; Mondal, S.; Nemade, B. A review: Data pre-processing and data augmentation techniques. *Glob. Transit. Proc.* **2022**, *3*, 91–99. [CrossRef]
29. Chen, D.; Liu, Y.; Huang, L.; Wang, B.; Pan, P. Geoaug: Data augmentation for few-shot nerf with geometry constraints. In *European Conference on Computer Vision*; Springer: Cham, Switzerland, 2022; pp. 322–337. [CrossRef]
30. Nielsen, F. Statistical divergences between densities of truncated exponential families with nested supports: Duo Bregman and duo Jensen divergences. *Entropy* **2022**, *24*, 421. [CrossRef] [PubMed]
31. Pei, X.; Zhao, Y.; Chen, L.; Guo, Q.; Duan, Z.; Pan, Y.; Hou, H. Robustness of machine learning to color, size change, normalization, and image enhancement on micrograph datasets with large sample differences. *Mater. Des.* **2023**, *232*, 112086. [CrossRef]
32. Islam, M.J.; Ahmad, S.; Haque, F.; Reaz, M.B.I.; Bhuiyan, M.A.S.; Islam, M.R. Application of min-max normalization on subject-invariant EMG pattern recognition. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 1–12. [CrossRef]
33. Zhang, C.; Zou, X.; Lin, C. Fusing XGBoost and SHAP models for maritime accident prediction and causality interpretability analysis. *J. Mar. Sci. Eng.* **2022**, *10*, 1154. [CrossRef]
34. Zhang, P.; Jia, Y.; Shang, Y. Research and application of XGBoost in imbalanced data. *Int. J. Distrib. Sens. Netw.* **2022**, *18*, 15501329221106935. [CrossRef]
35. Ben Jabeur, S.; Stef, N.; Carmona, P. Bankruptcy prediction using the XGBoost algorithm and variable importance feature engineering. *Comput. Econ.* **2023**, *61*, 715–741. [CrossRef]
36. Li, D.C.; Lin, M.Y.C.; Chou, L.D. Macroscopic big data analysis and prediction of driving behavior with an adaptive fuzzy recurrent neural network on the internet of vehicles. *IEEE Access* **2022**, *10*, 47881–47895. [CrossRef]
37. Tan, K.L.; Lee, C.P.; Anbananthen, K.S.M.; Lim, K.M. RoBERTa-LSTM: A hybrid model for sentiment analysis with transformer and recurrent neural network. *IEEE Access* **2022**, *10*, 21517–21525. [CrossRef]
38. Ariza-Colpas, P.P.; Vicario, E.; Oviedo-Carrascal, A.I.; Butt Aziz, S.; Piñeres-Melo, M.A.; Quintero-Linero, A.; Patara, F. Human activity recognition data analysis: History, evolutions, and new trends. *Sensors* **2022**, *22*, 3401. [CrossRef] [PubMed]
39. Al Hamoud, A.; Hoenig, A.; Roy, K. Sentence subjectivity analysis of a political and ideological debate dataset using LSTM and BiLSTM with attention and GRU models. *J. King Saud Univ.-Comput. Inf. Sci.* **2022**, *34*, 7974–7987. [CrossRef]
40. Wang, F.; Laili, Y.; Zhang, L. Trust Evaluation for Service Composition in Cloud Manufacturing Using GRU and Association Analysis. *IEEE Trans. Ind. Inform.* **2022**, *19*, 1912–1922. [CrossRef]
41. Rankovic, D.; Rankovic, N.; Ivanovic, M.; Lazic, L. The Generalization of Selection of an Appropriate Artificial Neural Network to Assess the Effort and Costs of Software Projects. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*; Springer International Publishing: Cham, Switzerland, 2022; pp. 420–431. [CrossRef]
42. Rankovic, D.; Rankovic, N.; Ivanovic, M.; Lazic, L. Convergence rate of Artificial Neural Networks for estimation in software development projects. *Inf. Softw. Technol.* **2021**, *138*, 106627. [CrossRef]
43. Slack, D.; Hilgard, S.; Jia, E.; Singh, S.; Lakkaraju, H. Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, 7–8 February 2020; pp. 180–186. [CrossRef]
44. Panati, C.; Wagner, S.; Brüggenwirth, S. Feature relevance evaluation using grad-CAM, LIME and SHAP for deep learning SAR data classification. In Proceedings of the 2022 23rd International Radar Symposium (IRS), Gdansk, Poland, 12–14 September 2022; pp. 457–462. [CrossRef]
45. Gramegna, A.; Giudici, P. SHAP and LIME: An evaluation of discriminative power in credit risk. *Front. Artif. Intell.* **2021**, *4*, 752558. [CrossRef] [PubMed]
46. Holzinger, A.; Saranti, A.; Molnar, C.; Biecek, P.; Samek, W. Explainable AI methods-a brief overview. In *xxAI-Beyond Explainable AI: International Workshop, Held in Conjunction with ICML 2020, Vienna, Austria, 18 July 2020, Revised and Extended Papers*; Springer International Publishing: Cham, Switzerland, 2022; pp. 13–38. [CrossRef]
47. Sahay, S.; Omare, N.; Shukla, K.K. An Approach to identify Captioning Keywords in an Image using LIME. In Proceedings of the 2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), Greater Noida, India, 19–20 February 2021; pp. 648–651. [CrossRef]
48. Ye, X.; Fang, F.; Wu, J.; Bunescu, R.; Liu, C. Bug Report Classification Using LSTM Architecture for More Accurate Software Defect Locating. In Proceedings of the 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), Orlando, FL, USA, 17–20 December 2018; pp. 1438–1445. [CrossRef]
49. Dam, H.K.; Tran, T.; Pham, T.; Ng, S.W.; Grundy, J.; Ghose, A. Automatic Feature Learning for Predicting Vulnerable Software Components. *IEEE Trans. Softw. Eng.* **2021**, *47*, 67–85. [CrossRef]
50. Stach, W.; Pedrycz, W.; Kurgan, L.A. Learning of Fuzzy Cognitive Maps Using Density Estimate. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2012**, *42*, 900–912. [CrossRef] [PubMed]
51. Sharma, A.; Tselykh, A. Machine Learning-Enabled Estimation System Using Fuzzy Cognitive Mapping: A Review. In *Proceedings of Third International Conference on Computing, Communications, and Cyber-Security*; Singh, P.K., Wierzchoń, S.T., Tanwar, S., Rodrigues, J.J.P.C., Ganzha, M., Eds.; Lecture Notes in Networks and Systems; Springer: Singapore, 2023; Volume 421. [CrossRef]