An Optimal Spacing Approach for Sampling Small-sized Datasets for Software Effort Estimation

Samuel Abedu[1], Solomon Mensah[2]*, Frederick Boafo[3], Eva Bushel[2] and Elizabeth Akuafum[2]
[1]Department of Computer Science and Software Engineering, Concordia University, Canada
[2]Department of Computer Science, University of Ghana, Accra - Ghana
[3]Department of Computer Science, Lancaster University Ghana
samuel.abedu@mail.concordia.ca, smensah03@ug.edu.gh, f.boafo@lancaster.edu.gh
{enabushel, eadick}@st.ug.edu.gh

***Abstract*—*Context*: There has been a growing research focus in conventional machine learning techniques for software effort estimation (SEE). However, there is a limited number of studies that seek to assess the performance of deep learning approaches in SEE. This is because the sizes of SEE datasets are relatively small. *Purpose*: This study seeks to define a threshold for small-sized datasets in SEE, and investigates the performance of selected conventional machine learning and deep learning models on small-sized datasets. *Method*: Plausible SEE datasets with their number of project instances and features are extracted from existing literature and ranked. Eubank's optimal spacing theory is used to discretize the ranking of the project instances into three classes (small, medium and large). Five conventional machine learning models and two deep learning models are trained on each dataset classified as small-sized using the leave-one-out cross-validation. The mean absolute error is used to assess the prediction performance of each model. *Result*: Findings from the study contradicts existing knowledge by demonstrating that deep learning models provide improved prediction performance as compared to the conventional machine learning models on small-sized datasets. *Conclusion*: Deep learning can be adopted for SEE with the application of regularisation techniques.**

Keywords*—*Deep learning; Conventional Machine learning; Software effort estimation; Small-sized; Optimal spacing theory.

1. INTRODUCTION

A. Background and Motivation

In software engineering, the use of data for setting up predictive models is vital in estimating the effort required for projects. This data comprises features extracted from at least one software project and a collection of related sets for the various components of the project(s). To set up an SEE predictive model, such datasets are considered for the training and validation needs of the given *x model* developed, before considering the estimated effort for *new* project instance(s).

It is known that deep learning works best for relatively large-sized datasets and has been considered effective for estimating the target or dependent variable when setting up classifiers and predictive models [1]. The key motivating question worth investigating is, *can it work best for relatively small-sized dataset(s)?* We seek to explore this issue in software engineering specifically in the domain of SEE. A leading question that also arises is that, *how can we define small-sized data from a given dataset?* We consider Eubank's optimal spacing theorem [2] a plausible solution for addressing the small-sized issue raised and move on to perform a comparative study considering a sample of deep learning *versus* conventional machine learning techniques for our investigation.

B. Overview of Software Effort Estimation (SEE)

SEE deals with predicting the relevant effort in relation to project cost and resource allocation to enable timely production and delivery of software projects within budget [3]. When project effort is overestimated or underestimated, it can result in devastating consequences for the given company. For example, a company may face wastage of resources and contract loss in the case of effort overestimation, and poor project development quality or uncompleted projects in the case of underestimation. The estimation of software effort is done at the beginning of the project [4]. Accordingly, the precise effort of a software project is mostly determined when the project closes.

Deep learning methods have been successfully applied in various fields, including computer vision, speech recognition, software defect prediction, financial analysis but has not been adequately explored in SEE. Perhaps this is because, deep learning methods perform better on relatively large-sized datasets [5]. Yet, the ability to obtain relatively large datasets in SEE is a challenge because owners of such datasets are reluctant to share and provide mainly due to privacy issues. Thus, researchers use the available data considered *small* for estimating the effort [6]. There is the need to investigate a feasible approach for determining relatively small-sized data and investigating the performance of deep learning on small-sized data in SEE.

The remaining sections of the paper are organized as follows: Section 2 presents size of the SEE datasets considered for the study and the central limit theory. Section 3 presents the Eubank's optimal spacing theorem. Section 4 details the methodological procedure, dataset description, data pre-processing, experimental setup, model selection and performance evaluation metrics. Section 5 presents the experimental results and discussion. Lastly, Section 6 concludes the study.

DOI reference number: 10.18293/SEKE2023-176

2. Size of SEE Dataset & Central Limit Theorem

Due to the relatively high cost of data collection, and unwillingness of companies to share software project data (due to privacy issues) [6], it has led to relatively small sizes of SEE datasets (L. Song et al., 2018). As a result of these challenges, historic or archival SEE data in the public domain are mostly used.

The definition of a small SEE dataset is ambiguous since different researchers have different perspectives of the definition (Song et al., 2019). Mostly, researchers failed to justify considering a particular type of dataset as small-sized.

Evidence from the central limit theorem shows that a sample size greater than 30 is sufficient for the central limit theorem to apply. Hence, sample sizes less than or equal to 30 may be considered small. Yet, this statistical definition cannot be adopted for this study because SEE datasets with project instances greater than 30 have been described as small-sized (Song et al., 2019). This makes it imperative for this study to define the conditions for a small-size dataset that is appropriate for this investigation.

3. Eubank's Optimal Spacing Theorem

In a study by Eubank [2], a density quantile function method was introduced to address the optimal spacing selection problem for linear estimation of a given interval. This approach makes use of a quantile function over a given interval, $[p,q] \subset [0,1]$ to enable a successful estimation of location and scale parameters for a censored set of order statistics. Optimal asymptotic spacings are generated from the quantile function. The optimal asymptotical spacing can be considered as a threshold value for the spacing selection over the interval. The density quantile function $Q(u)$ is defined in (1).

$$Q(u) = F^{-1}(u), 0 \leq u \leq 1 \qquad (1)$$

where F is considered as the censored set distribution function and u is regarded as the location parameter for defining the asymptotic optimal spacing.

The theorem was considered to discretize the rankings of the studied datasets into three main classes. The theorem uses the density quantile function to discretize the rankings of the frequency instances per each dataset into three classes. Class 1 is regarded as the first-class (Q_1), class 2 as the second-class (Q_2 and Q_3), and class 3 as the third-class (Q_4). Datasets ranked with total number of instances less than or equal to first class are classified as small-sized datasets. Alternatively, all datasets with ranks greater than or equal to the third class are classified as large-sized datasets, and the remainder as medium-sized datasets, that is in-between the first and third classes.

4. Empirical Framework

A. Dataset Description

The study selected 22 publicly available datasets from the PROMISE and GitHub repositories. The ranking of the data sizes was based on the total number of instances for the studied datasets. The project dataset with the minimum number of instances was ranked the lowest (one) and the project with the maximum number of instances was ranked highest (22).

Six out of the 22 datasets were classified as small-sized based on the classification scheme provided in Section 3. Thus, these 6 datasets were adopted for the study's empirical and comparative analysis. The number of project instances, project features, description of project type, the mean and standard deviation of all 6 datasets (*Albrecht*, *Atkinson*, *Cosmic*, *Kemerer*, *Finnish*, and *Telecom1*) are presented in **Table 1**.

A maximum threshold of 43 project instances based on Eubank's optimal spacing theory is defined for classifying a given SEE dataset as small-sized.

The discretization scheme classified the *Java software project dataset* [6]*, Tukutuku dataset, Bielak dataset, dataset 1*[12], *USP05 dataset* and *China dataset* as large-sized. Similarly, a minimum threshold of 147 project instances is defined for classifying a dataset as large-sized. Datasets with project instances within the range of 44 to 146 inclusive are classified as medium-sized. Thus, 10 datasets, namely (*Miyazaki94*, *Miyazaki*, Qi et al.'s [6] *Web data*, *Nasa*, *Cocomo81*, *Maxwell*, Lopez-Martin's [12] *Data 2*, *Desharnais*, *Nasa93* and *Kitchenham dataset*).

In addition to the small-sized datasets, two relatively large datasets, namely the ISBSG and China datasets [13] were also used. These relatively large-sized datasets were added to test the assertion that, deep learning models perform better on larger datasets than conventional machine learning models for SEE. The ISBSG dataset release 10 which contains 4106 cross-organisational projects was used.

Table 1. Description of selected datasets

Dataset	Instances	Features	Description	Mean (Effort)	Std (Effort)
Albrecht	24	7	IBM DP Services Project	21.9	28.4
Atkinson	16	12	Builds to a large telecommunications product at U.K. company X	456.1	241.1
Cosmic	42	10	N/A	4965.5	8457.1
Finnish	38	6	Data collected by the TIEKE organization from IS projects from nine different Finnish companies	7678.3	7135.3
Kemerer	15	5	Large business application	219.2	263.1
Telecom1	18	2	Enhancement to a U.K. telecommunication product	284.3	264.7
ISBSG (R10)	4106	105	Cross-organisational projects compiled by the ISBSG	5925.5	20685.9
China	499	17	Data collected from various software firms	3921.0	6480.9

Cross-organizational projects are heterogeneous in nature and do not share common characteristics like development policy, development team and programming skills. As a result, the 642 projects from the Communication organization type were selected for this study. Similar to the study by Mensah et al. [14], the study selected the project size, development type, language type and development platform as the features and the normalised development effort as the dependent variable.

B. Experimental Setup

Seven prediction models were set up with each model trained on the selected datasets. The models were trained based the leave-one-out cross-validation (LOOCV) approach, and the Mean Absolute Error (MAE) was used to evaluate the performance. The studied datasets were pre-processed using the data pre-processing approach described in Section 4C. The log transformation technique was applied to the datasets to minimise the effect of skewness. Feature selection was also performed to select relevant features from the data for the model training. The early stopping technique was applied to reduce the overfitting of the deep learning models. This study implemented the early stopping regularisation technique as used in the work of Kalichanin-Balich and Lopez-Martin [15] to reduce overfitting and improve the performance of the deep learning models. In addition, dropout was added after each hidden layer to also reduce overfitting.

C. Data Pre-processing

There is the need for high quality data to improve the quality of mining results. Thus, pre-processing the data to achieve high quality is very important when training and validating machine learning models. To achieve high data quality (i.e., resolve issues of missing data values, outliers, and influential data) the datasets were pre-processed before setting up the prediction models as follows: An in-depth analysis of the datasets was performed to address missing data values. Columns of each project instance were examined for null or NaN values. The ISBSG was the only dataset that contained missing values. Project instances in the ISBSG that had missing entries for the selected features were eliminated. Data point with Cook's distance measures three times more than the mean of Cook's distance values were considered to be outliers. Whereas data points whose DFFITS were greater than one was considered influential [14].

D. Model Selection

This section discusses the models selected for this study and the hyperparameter tuning approach used. The models are the Automatically Transformed Linear Model (ATLM), Bayesian Network (BN), ElasticNet (ENR) regression, Support Vector Machine (SVM), Artificial Neural Network (ANN), Deep Neural Network (DNN) and Long-Short Term Memory (LSTM). The hyperparameters of the models were fine-tuned using the Bayesian optimization approach. This approach models the generalisation performance of a prediction model as a Gaussian process and maintains a posterior distribution as observations are made on the results from running the model with different hyperparameters. The Bayesian optimization algorithm picks the hyperparameter values for the next experiment based on the Gaussian process upper confidence bound (UCB).

1) Deep Learning Models

A Deep learning model simulates human brain processes. DNN is made of an input layer, an output layer and multiple hidden layers which are made of hidden nodes called neurons. Note that for ANN, there is a maximum of one hidden layer. Each neuron receives input and process it solving a non-linear function of the inputs by assigning weights. Taking the weights assigned to the neuron as w_{ij} at the start of training. The weights are updated after each training epoch (t) to $w_{ij}^{(t+1)}$ where the change in weights $\left(\Delta w_{ij}^{(t+1)}\right)$ is defined in (2).

$$\Delta w_{ij}^{(t+1)} = \gamma \Delta w_{ij}^{t} - \alpha \frac{\partial L(.)}{\partial w_{ij}} \quad (2)$$

where γ is the learning momentum, α is the learning rate and $L(.)$ is the loss function.

This study adopted the RMSProp optimization algorithm for model training and the optimal learning rate (α) was identified using the Bayesian Optimization algorithm. The Parametric Rectified Linear Unit (PRelu) which improves the accuracy of models over the Relu was used. PRelu preserves the properties of linear models, this makes it easier to be optimized. The PRelu is defined in (3).

$$PRelu(x) = max\,(ax, x) \quad (3)$$

where a is the slope parameter learned by the model through backpropagation. Dropout and early stopping were applied to reduce overfitting. The efficient dropout was identified by using Bayesian Optimization. This study adopted the feed-forward neural network with the backpropagation of errors.

The number of hidden layers used in this study varied between two and five. Also, Bayesian optimization was used to identify the optimum learning rate α and the dropout for the models.

The LSTM has been demonstrated to be efficient in modelling time-series data. Considering that software project starts and ends at distinct time points, software effort data can be classified as pseudo time series data and hence makes the LSTM suitable for this study. Also, considering that no study had considered it in traditional effort estimation, it was adopted. The LSTM is a variant of recurrent neural networks that solves the vanishing gradient problem [16]. It uses a loop in the network that allows information to persist in the network. Given a data (X, Y) with $(x_1, \ldots, x_n) \in X$ and outputs $(y_1, \ldots, y_n) \in Y$. At step t the LSTM reads input x_t, the hidden state h_{t-1} and the previous memory c_{t-1} to compute hidden state h_t. The hidden state h_t is used to predict the output y_t. The memory cell is the element that stores the accumulated information over time.

The LSTM model was also set up with two LSTM layers and the glorot uniform kernel initializer was used. The PRelu activation function was used for all layers in the network.

Dropout and early stopping were applied to reduce overfitting. The dropout and the learning rate (α) of the model were fine-tuned using the Bayesian optimization. The optimum values for the dropout and the learning rate (α) for the ANN, DNN and LSTM models are presented in Table 2.

2) Conventional Machine Learning Models

The ATLM is a baseline model for comparative studies in SEE. It is based on a multiple linear regression, and it performs automatic data pre-processing such as log and square-root transformations on the data and also compares the skewness in the dependent and independent variables. It uses the least skewed data for model construction and effort prediction.

The Bayesian network (BN) is a graphical model that encodes the probabilistic relationship between related variables. BN is determined by a pair as defined in (4).

$$BN = (G, P) \tag{4}$$

where G is a directed acyclic graph with nodes X_i and P is the local probability of all variables in the network. For instance, given node X_1 connected to node X_2 and node X_2 connected to node Y, then the conditional probability of finding Y in (5).

$$P(Y/X_2, X_1) = P(Y/X_2) \tag{5}$$

The joint probability of each variable satisfies Markov's condition that each variable X_i is conditionally independent of the set of its non-descendants. The distribution is factorised in (6).

$$P(X_1, \dots, X_n) = \prod_{i=1}^{n} P\big(X_i | \pi(X_i)\big) \tag{6}$$

where $\pi(X_i)$ is the set of nodes directly connected to X_i. The Bayesian network takes into account the probability effect of each input variable on the dependent variable.

The ElasticNet (ENR) performs regularisation and feature selection to get rid off highly correlated estimator variables before developing the model. Given a set of input data with N observation pairs (x_i, y_i) and an approximated regression function $E(Y|X = x) = \beta_0 + x^T\beta$, the ENR finds the optimal β by solving the problem defined in (7).

$$\hat{\beta} = \min_{(\beta_0, \beta) \in R^{p+1}} \left[\frac{1}{2N} \sum_{i=1}^{N} \big(y_i - \beta_0 - x_i^T\big)^2 + \lambda P_\alpha(\beta) \right] \tag{7}$$

where P_α is the elastic net penalty given by

$$P_\alpha(\beta) = \left[\frac{1}{2}(1-\alpha)\beta_j^2 + \alpha|\beta_j| \right]$$

For $j = 1, \dots, p$. The P_α is used in finding highly correlated input variables. α is the hyperparameter tuned for the study. The value of α resulting in the best performance for each dataset is presented in Table 2.

The SVM is a supervised learning approach for classification that has been extended for regression problems as ε-SVM. The ε-SVM uses Lagrange multipliers to find a function $f(X_i)$ that has a deviation of at most ε from the dependent variable and then produces a final solution from a combination of cases from the dataset. The kernel function of the SVM allows the ε-SVM to handle datasets that are of non-linear and complex. The kernel is treated as a hyperparameter. The best performing kernel for the SVM on each dataset is recorded in Table 2.

Table 2. Optimal Hyperparameter values for each Model

	DNN		LSTM		ANN		ENR	SVM
Dataset	**Dropout**	**Learning Rate (α)**	**Dropout**	**Learning Rate (α)**	**Dropout**	**Learning Rate**	**Optimum α**	**Kernel**
Albrecht	0.0294	0.0327	0.0732	0.0092	0.092	0.035	0.9996	RBF
Atkinson	0.2438	0.0064	0.0732	0.0092	0.169	0.028	0.1956	Sigmoid
Cosmic	0.2081	0.072	0.0732	0.0092	0.209	0.068	0.0285	Polynomial
Finnish	0.2081	0.072	0.2081	0.0720	0.102	0.087	1.3e-07	Linear
Kemerer	0.2081	0.072	0.0732	0.0092	0.216	0.057	0.0074	Linear
Telecom	0.2081	0.072	0.0732	0.0092	0.208	0.055	0.0157	Sigmoid
ISBSG	0.2000	0.001	0.2000	0.001	0.200	0.001	0.0001	Polynomial
China	0.2000	0.001	0.2000	0.001	0.200	0.001	0.0018	Linear

E. Performance Measure

MAE is a robust, effective and reliable performance metric for assessing SEE models and measuring the average error magnitude. The MAE is an unbiased error estimator. It has been considered in the evaluation of estimation accuracy and thus it was considered for this study. The MAE is defined in (8).

$$MAE = \frac{1}{n} \sum_{i=1}^{n} \left| E_{A_i} - E_{E_i} \right| \tag{8}$$

where E_A is the actual effort of a project case or instance and E_E is the estimated effort of a project instance with n number of instances. A single performance evaluation measure was used to avoid the ambiguity associated with the conclusion instability issue.

Cliff's δ effect size was used to assess the practical significance of the model. The effect size is less affected by the sample size than other statistical significance tests such as Yuen's [18]. The effect size provides an objective measure of the importance of an experimental effect and in this empirical study, it was used to measure the magnitude of the difference between the actual effort (y) and the predicted effort ($\hat{y}$). It is defined in (9).

$$\delta = \frac{COUNT(y_i > \hat{y}_i) - COUNT(y_i < \hat{y}_i)}{nm} \tag{9}$$

(1)

where y_i is the actual effort and the $\hat{y}_j$ is the predicted effort of each dataset. The number of project cases in the actual and predicted classes are denoted by n and m respectively.

Kampenes et al. [18] provides interpretation for the cliff's delta effect size as follows: (1) negligible refers to $\delta < 0.112$, small is $0.112 \leq \delta < 0.276$, medium is $0.276 \leq \delta \leq 0.427$ and large refers to $\delta \geq 0.428$. This study makes use of an effect size threshold of negligible ($\delta < 0.112$), with the argument, that the difference in magnitude between the actual and predicted effort should be negligible.

5. Result and Discussion

The result as presented in Table 3 show that the deep learning models (DNN and LSTM) recorded the best prediction accuracy for five datasets. The ANN, which is a shallow neural network recorded the best prediction accuracy for two datasets and the baseline ATLM also recorded the best prediction accuracy for one dataset.

The ANN had the best prediction accuracy for the Albrecht dataset with negligible practical significance difference between the predicted and actual efforts. DNN and LSTM models also produced prediction accuracies better than the conventional machine learning models on the Albrecht dataset.

The results of the models using the Atkinson dataset as presented in Table 3 shows the DNN model as having the best prediction accuracy. The DNN recorded a mean absolute error (MAE) of 0.0018 with a 0.0102 Cliff's δ effect size (signifying low practical significance difference). The LSTM and ANN also produced an impressive prediction accuracy with MAE of 0.002 and 0.0036 respectively. The results for the Atkinson dataset showed that the deep learning models outperform the conventional machine learning models. This accounted for the conventional machine learning models recording higher MAEs (poor performance).

For the Cosmic dataset, the LSTM model recorded the best prediction accuracy with an MAE of 0.0075. Again, the ANN and DNN also achieved high prediction accuracy as compared to the conventional machine learning models. The high MAE measured for the conventional machine learning was as a result of more variations in the absolute values of the model's prediction on the cosmic dataset.

Table 3. Performance Evaluation of Models

Dataset	Metric	ATLM	DNN	LSTM	ANN	BN	ENR	SVM
Albrecht	MAE	0.352	0.014	0.024	**0.006**	0.239	0.248	0.186
	Cliff's δ	0.004	0.017	0.010	0.031	0.010	0.010	0.004
Atkinson	MAE	0.052	**0.002**	0.002	0.004	0.038	0.034	0.031
	Cliff's δ	0.000	0.010	0.000	0.010	0.010	0.010	0.010
Cosmic	MAE	0.549	0.020	**0.008**	0.013	0.516	0.475	0.467
	Cliff's δ	0.001	0.005	0.001	0.002	0.002	0.001	0.007
Finnish	MAE	**0.001**	0.012	0.009	0.013	0.001	0.002	0.001
	Cliff's δ	0.002	0.002	0.007	0.002	0.002	0.009	0.002
Kemerer	MAE	0.206	**0.009**	0.024	0.014	0.196	0.173	0.161
	Cliff's δ	0.000	0.031	0.031	0.020	0.020	0.020	0.000
Telecom	MAE	0.206	0.009	**0.003**	0.003	0.216	0.199	0.128
	Cliff's δ	0.004	0.004	0.013	0.022	0.004	0.013	0.031
ISBSG	MAE	0.014	0.005	0.004	**0.004**	0.014	0.014	0.350
	Cliff's δ	0.001	0.000	0.000	0.000	0.001	0.001	0.006
China	MAE	0.006	0.005	**0.005**	0.007	0.007	0.007	0.174
	Cliff's δ	0.000	0.000	0.002	0.000	0.000	0.000	0.000

The values in bold represent the best performance measure for each dataset across the learners – ATLM, DNN, LSTM, ANN, BN, ElasticNet (ENR) and SVM

δ - *Cliff's delta effect size*

For the Finnish dataset, the conventional machine learning models outperformed the deep learning models. The baseline ATLM recorded an MAE of 0.0006. The Bayesian Network (BN), ElasticNet and the SVM recorded MAE of 0.0014, 0.0021 and 0.001 respectively. The DNN, LSTM and ANN also recorded MAE of 0.0117, 0.0092 and 0.0125 respectively.

The DNN proved to be the best model for the Kemerer dataset with an MAE of 0.0088 and Cliff's δ effect size of 0.0306. The LSTM and ANN model recorded MAE of 0.0239 and 0.0143 respectively. The SVM was the best conventional machine learning model with an MAE of 0.1614.

The performance of the models on the Telecom dataset showed the LSTM as the best. The LSTM model recorded an MAE of 0.0029 with a 0.0133 Cliff's δ effect size. The DNN and ANN models also achieved MAE of 0.0086 and 0.0034 respectively. The SVM with an MAE of 0.1279 was the best performing convention machine learning model on the Telecom dataset. The baseline ATLM produced an MAE of 0.206.

For the ISBSG, which is a large-sized data, the artificial neural network recorded the best performance with an MAE of 0.0038 and Cliff's delta effect size of 0.0001. The two deep learning models recorded better performance than the conventional machine learning models.

The DNN and LSTM models recorded MAEs of 0.0049 and 0.0044 respectively compared to the MAEs of the ATLM, Bayesian network, Elastic Net and SVM which were 0.0136, 0.0136, 0.0136 and 0.3503 respectively.

Also, for the other large-sized dataset, the China dataset, the LSTM recorded the best MAE, 0.0048 with Cliff's delta effect size of 0.0017. The DNN model also recorded an MAE of 0.0049 with Cliff's delta effect size of 0.0001. These two deep learning models with the artificial neural network outperformed the conventional machine learning models on this dataset. The conventional machine learning models recorded MAEs of 0.0061, 0.0068, 0.0068 and 0.1738 as shown in Table 3.

The magnitude of Cliff's δ effect size measurement for all models were negligible ($\delta < 0.112$) as defined by Kampenes et al. [18]. This means there was little difference between the magnitude of the actual effort values and the predicted effort estimations of each model.

6. Conclusion and Future Work

In this study, a threshold for defining a small-sized dataset using Eubank's optimal spacing theory is introduced in SEE. Twenty-two datasets were selected from recent SEE studies. The classification scheme based on these 22 datasets classified six datasets as small-sized and defined a threshold of 43 project instances for a small-sized dataset. The six datasets were selected for the empirical study in addition to two large-sized datasets. The classification scheme was constructed using Eubank's optimal spacing theorem [2]. This is due to the advantage of splitting a given interval into optimal sizes or thresholds.

The empirical study investigated the prospects of deep learning in SEE by comparing the performance of the ATLM, SVM, Bayesian network, ElasticNet, ANN, DNN and LSTM models. The leave-one-out cross-validation (LOOCV) was applied in

this study for the training and validation needs of the selected models, and performance evaluation was done using MAE.

The results showed the selected deep learning models outperformed the conventional machine learning models on the Atkinson, Cosmic, Kemerer and Telecom datasets. The ANN, which is a shallow neural network, outperformed the deep learning models on the Albrecht dataset, and likewise the ATLM on the Finnish dataset. Also, findings from the study show that deep learning models have better performance on large-sized SEE datasets than conventional machine learning models.

The study concluded from the results that deep learning should be adopted for software effort estimation. However, it recommends techniques like dropout and early stopping to reduce overfitting in the deep learning models.

In future, the empirical study will be extended to evaluate the computational cost of running the deep learning and conventional machine learning models in the software engineering field. This will guide researchers and practitioners on the trade-offs that can be made in choosing either a conventional machine learning or deep learning model.

REFERENCES

[1] B. Barz and J. Denzler, "Deep learning on small datasets without pre-training using cosine loss," *Proceedings - 2020 IEEE Winter Conference on Applications of Computer Vision, WACV 2020*, pp. 1360–1369, Mar. 2020, doi: 10.1109/WACV45572.2020.9093286.

[2] R. L. Eubank, "A Density-Quantile Function Approach to Optimal Spacing Selection," *The Annals of Statistics*, vol. 9, no. 3, pp. 494–500, 1981, doi: 10.1214/AOS/1176345454.

[3] S. Mensah and P. K. Kudjo, "A classification scheme to improve conclusion instability using Bellwether moving windows," *Journal of Software: Evolution and Process*, vol. 34, no. 9, Sep. 2022, doi: 10.1002/smr.2488.

[4] Y. Mahmood, N. Kama, A. Azmi, A. S. Khan, and M. Ali, "Software effort estimation accuracy prediction of machine learning techniques: A systematic performance evaluation," *Softw Pract Exp*, 2021, doi: 10.1002/SPE.3009.

[5] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[6] F. Qi, X. Y. Jing, X. Zhu, X. Xie, B. Xu, and S. Ying, "Software effort estimation based on open source projects: Case study of Github," *Inf Softw Technol*, vol. 92, pp. 145–157, Dec. 2017, doi: 10.1016/j.infsof.2017.07.015.

[7] L. Song, L. L. Minku, and X. Yao, "A novel automated approach for software effort estimation based on data augmentation," in *ESEC/FSE 2018 - Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, Oct. 2018, pp. 468–479. doi: 10.1145/3236024.3236052.

[8] X. Y. Jing, F. Qi, F. Wu, and B. Xu, "Missing data imputation based on low-rank recovery and semi-supervised regression for software effort estimation," in *Proceedings - International Conference on Software Engineering*, May 2016, vol. 14-22-May-2016, pp. 607–618. doi: 10.1145/2884781.2884827.

[9] V. K. Bardsiri, D. N. A. Jawawi, S. Z. M. Hashim, and E. Khatibi, "A flexible method to estimate the software development effort based on the classification of projects and localization of comparisons," *Empir Softw Eng*, vol. 19, no. 4, pp. 857–884, 2014, doi: 10.1007/s10664-013-9241-4.

[10] L. Song, L. L. Minku, and Y. A. O. Xin, "Software effort interval prediction via Bayesian inference and synthetic bootstrap resampling," in *ACM Transactions on Software Engineering and Methodology*, Jan. 2019, vol. 28, no. 1. doi: 10.1145/3295700.

[11] Q. Song, M. Shepperd, and C. Mair, "Using grey relational analysis to predict software effort with small data sets," in *Proceedings - International Software Metrics Symposium*, 2005, vol. 2005, pp. 321–330. doi: 10.1109/METRICS.2005.51.

[12] C. Lopez-Martin, "A fuzzy logic model for predicting the development effort of short scale programs based upon two independent variables," *Applied Soft Computing Journal*, vol. 11, no. 1, pp. 724–732, 2011, doi: 10.1016/j.asoc.2009.12.034.

[13] F. H. Yun, "China: Effort Estimation Dataset," Apr. 2010, doi: 10.5281/ZENODO.268446.

[14] S. Mensah, J. Keung, S. G. MacDonell, M. F. Bosu, and K. E. Bennin, "Investigating the Significance of the Bellwether Effect to Improve Software Effort Prediction: Further Empirical Study," *IEEE Trans Reliab*, vol. 67, no. 3, pp. 1176–1198, 2018, doi: 10.1109/TR.2018.2839718.

[15] I. Kalichanin-Balich and C. Lopez-Martin, "Applying a feedforward neural network for predicting software development effort of short-scale projects," in *8th ACIS International Conference on Software Engineering Research, Management and Applications, SERA 2010*, 2010, pp. 269–275. doi: 10.1109/SERA.2010.41.

[16] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[17] P. A. Whigham, C. A. Owen, and S. G. MacDonell, "A baseline model for software effort estimation," *ACM Transactions on Software Engineering and Methodology*, vol. 24, no. 3, 2015, doi: 10.1145/2738037.

[18] V. B. Kampenes, T. Dybå, J. E. Hannay, and D. I. K. Sjøberg, "A systematic review of effect size in software engineering experiments," *Information and Software Technology*, vol. 49, no. 11–12. Elsevier, pp. 1073–1086, Nov. 01, 2007. doi: 10.1016/j.infsof.2007.02.015.