International Journal of Software Engineering and Its Applications
Vol. 10, No. 2 (2016), pp. 191-206
http://dx.doi.org/10.14257/ijseia.2016.10.2.16

An Experimental Comparison of Three Machine Learning Techniques for Web Cost Estimation

Olawande Daramola, Ibidun Ajala and Ibidapo Akinyemi

Department of Computer and Information Sciences, Covenant University, Ota
olawande.daramola@covenantuniversity.edu.ng, titi_lola@hotmail.com,
ioakinyemi@ covenantuniversity.edu.ng

Abstract

Many comparative studies on the performance of machine learning (ML) techniques for web cost estimation (WCE) have been reported in the literature. However, not much attention have been given to understanding the conceptual differences and similarities that exist in the application of these ML techniques for WCE, which could provide credible guide for upcoming practitioners and researchers in predicting the cost of new web projects. This paper presents a comparative analysis of three prominent machine learning techniques – Case-Based Reasoning (CBR), Support Vector Regression (SVR) and Artificial Neural Network (ANN) – in terms of performance, applicability, and their conceptual differences and similarities for WCE by using data obtained from a public dataset (www.tukutuku.com). Results from experiments show that SVR and ANN provides more accurate predictions of effort, although SVR require fewer parameters to generate good predictions than ANN. CBR was not as accurate, but its good explanation attribute gives it a higher descriptive value. The study also outlined specific characteristics of the 3 ML techniques that could foster or inhibit their adoption for WCE.

Keywords*: Web cost estimation, machine learning, support vector regression, case based reasoning, artificial neural networks*

1. Introduction

Web Cost Estimation is the act of predicting the amount of effort required in order to execute a web development project. Specifically, it entails determining the costs of development, and the amount of resources needed to ensure efficient and timely project delivery, that are within budget [1]. Effective web cost estimate is crucial for the success of web project management, because it assists project managers to manage costs, plan for potential risks, improve development practices, and help to ensure prompt completion of projects within budget [2]. Machine learning (ML) techniques have proved to be more precise for software cost estimation, and by extension web cost estimation when compared to other techniques such as expert judgments, or algorithmic models [1]. Several researchers have conducted comparative studies on the use of ML techniques for web cost estimation. So far, results reported in the literature indicated that Case-Based Reasoning (CBR), Support Vector Regression (SVR) and Artificial Neural Networks (ANN) have the best performance among the different machine learning techniques that have been used for web cost estimation [3-6]. However, most studies on the comparative performance of ML techniques for web cost estimation have failed to give adequate attention to the conceptual differences and similarities that exist in the application of these ML techniques. Information on the relative performance of ML techniques for web cost estimation alone without an understanding of the conceptual differences and similarities that exist among competing ML techniques cannot provide sufficient basis for upcoming practitioners and researchers to make a pragmatic choice of the ML technique to use for estimating the cost of new web projects. In other to make quality decisions

ISSN: 1738-9984 IJSEIA

about information on critical issues such as relative performance, and circumstances that favour or inhibit the application of specific ML techniques, we need to compare the three ML techniques relative to their performance, and identify the conceptual differences and similarities that exist among them.

This paper presents a comparative study of ML techniques for web cost estimation that focuses not just on performance, but also on the conceptual differences and similarities of the three ML techniques for web cost estimation. The motivation for this work is to provide an empirical basis for good decision making by practitioners, and upcoming researchers on the selection of ML techniques for web cost estimation. To achieve this task, we conducted functional approximation experiments using three ML techniques – CBR, ANN, and SVR – on the same dataset, and evaluated their relative performance by using standard metrics such as Magnitude of Relative Error (MRE), Mean Magnitude of Relative Error (MMRE) and Mean Absolute Error (MAE). Similarly, we performed a comparative analysis of the procedures, activities and experiences that accompany the application of the three ML techniques, to determine their conceptual similarities and differences.

The rest of this paper is as follows. Section 2 provides a background on web cost estimation, the three machine learning techniques – ANN, SVR, and CBR – and a review of related work. In Section 3, we give an overview of the methodology used for the experimental study, while Section 4 presents the description of the experiments that we performed on the published dataset that we used. In Section 5, we report the results and findings from the study. The paper is concluded in Section 6 with a brief note.

2. Background and Related Work

Web Cost Estimation (WCE) is the procedure for predicting the expected amount of effort that is required for web development. It involves making cost projections prior to web development in order to ensure efficient allocation of resources to enable timely project delivery, according to the financial plan [1]. WCE can be achieved by using expert judgment, evaluating parametric equations, or application of machine learning (ML) techniques.

2.1. Expert Judgment

WCE can be achieved by using the knowledge and experience of specialists in the field. The authors in [7] proposed an approach to WCE that investigates the application of COBRA. Also, Delphi technique can be seen as the most formal and meticulous method that is based on expert's opinion [8]. Delphi technique was found to enhance estimates and decrease individual prejudice. However, the solutions provided by expert judgment are limited to their own opinions and are subjective, which make them particularly susceptible to biases [9, 10].

2.2. Algorithmic Models

Algorithmic models predict estimates of effort using parametric equations. Some of the distinguished algorithmic models include Boehm's COCOMO'81, COCOMO II [11], Albrecht's Function Point [11, 12], SLIM [13], Ordinary Least-Squares regression (OLS) [14], and Classification and Regression Trees (CART) [12]. These algorithms require parameters that are not easy to acquire during the early stages of development. In these models, there is an inherent complex relationship between the related attributes, and they are unable to handle categorical data as well as lack of reasoning capabilities [15]. These techniques are applicable only when the variables are linear and data are fine-grained. The limitations of these techniques necessitated the discovery of machine learning [16].

2.3. Machine Learning (ML) for Web Cost Estimation

ML is a branch of artificial intelligence that is focused on design and development of algorithms that permit systems to evolve behavior based on observed data [17]. They are used to group together a set of techniques that represent some of the facets of human mind [16, 18]. The use of ML for web cost estimation (WCE) emerged as a way to overcome the weaknesses of algorithmic models and expert judgment. ML focuses on learning by recognizing complex patterns that exist and making intelligent decisions based on the available data. A learning procedure is applied to the data in order to obtain a good prediction function *f(x)*. Commonly used learning procedures are Linear Regression, Artificial Neural Networks, Decision Trees, Case Based Reasoning, Support Vector Regression, and Bayesian Networks. Table 1 gives a brief summary of the comparative studies that have been done in the field of WCE. The table reveals that ML techniques are more preferred when compared with other WCE techniques. In addition, methods such as CBR, ANN and SVR have relatively better performance when used for WCE due to their capability to learn and generalize from historical data.

Table 1. Overview of Comparative Studies on Application of Machine Learning Techniques for WCE

Reference	Prediction Techniques	Best Technique(s)	Title
Mendes et al. [19]	CBR, linear regression, stepwise regression	CBR	A Comparative Study of Cost Estimation Models for Web Hypermedia Applications
Mendes et al., [20]	Linear regression, stepwise regression	Linear regression	Comparison of Web Size Measures for predicting Web Design and Authoring Effort
Mendes et al., [21]	CBR, linear regression, stepwise regression, CART.	Linear and stepwise regression or CBR.	Cost Estimation Techniques for Web Projects
Ruhe, Jeffery & Wieczorek [7]	COBRA , expert opinion, linear regression	COBRA	Cost Estimation Benchmarking and Risk Analysis
Satyananda Reddy [4]	Expert judgment, ANN	ANN	A Neural Network Approach for Web Cost Estimation
Anna Corazza et al. [5]	SVR, Manual StepWise Regression, CBR, and Bayesian Networks.	SVR	Using Support Vector Regression for Web Development Effort Estimation

2.4. Case-Based Reasoning (CBR)

CBR is a machine-learning paradigm that closely models the human reasoning process. It works by comparing the target project, for which an evaluation is necessary, to similar finished projects with known efforts. The known efforts are used to produce the prediction of the effort for a new project based on attributes similarity between the new project and the completed projects. Applying CBR takes into consideration several parameters such as feature selection of applicable variable, comparison criteria,

normalization, and number of analogies, adaptation analogy and rules. The CBR cycle is a 4-stage process [22], which consist of:

i *RETRIEVE*: this entails retrieving the most similar case or cases to the target problem

ii *RESUSE:* this entails reusing the past information and solution to solve the new problem

iii *REVISE*: this entails modifying the proposed solution to better adapt the target problem

iv *RETAIN:* this entails storing the new solution to be part of the case base for further reference

Usually, similarity measure for CBR is evaluated using equation 1.

$$SIM\ (U,V) = 1 - dist(U,V) = \ 1 - \sqrt[2]{Ewi^2}dist^2(u,v) \tag{1}$$

Where

U = feature vector of a source case; V = feature vector of the objective case;

w_i = normalized magnitude of i^{th} feature;

The normalized distance is:

$$dist(u_i, v_i) = \frac{|\ u_i - v_i|}{|\ maxi - mini\ |} \tag{2}$$

2.5. Support Vector Regression (SVR)

The concept of Support Vector Machines (SVM) originated from the work of Vapnik [23]. It is based on a good mathematical framework that is rooted in statistical learning principle or Vapnik-Chervonenkis (VC) theory [24]. SVM shows the generalization error as an alternative of the error on definite data sets, which helps it to generalize well to hidden data. Hence, SVM has proved to be a good technique for handling classification and regression problems. Support Vector Regression (SVR) is the SVM variant that is used for regression analysis. In contrast to conventional regression techniques, the SVR try to reduce the upper bound on the generalization error rather than reducing the training error. SVR uses structural risk minimization, which entails minimizing generalized error, which performs better than the empirical risk minimization theory that is used by conventional approaches. SVR is based on convex optimization, which ensures that the local minimization is the unique minimization. The characteristics of SVR include enhanced generalization potential, global optimal solution using optimization theory, and the use of Kernel functions for nonlinear problem.

Formally, SVR relies on estimating a linear regression function

$$f(\mathrm{v}) = (\mathrm{w}, \mathrm{u}) + \mathrm{b} \tag{3}$$

Where *w* represents the slope and *b* is the offset of the regression line; the equation (3) is solved by minimizing the primal goal function and it is subjected to the corresponding constraints:

$$R[f] = \int Lf(u), v\ p(u,v)dudv$$

$$\min\ \frac{1}{2}w^T w + \ c\sum_{i=1}^{l}(\xi_i + \xi_i^*)$$

$$\text{Subject to} \quad \begin{cases} y_i - w^T u_i - \ b & \leq \ \varepsilon + \xi_i \\ -(y_i - w^T u_i - b) & \leq \ \varepsilon + \xi_i \\ \xi_i, \xi_i^* & \geq 0 \end{cases} \tag{4}$$

Where

$\frac{1}{2}w^T w$ is the smoothness of $f(\mathbf{x})$ (model complexity) and

$c\sum_{i=1}^{l}(\xi_i + \xi_i^*)$ = loss function; such that

$$|\xi_i|_\varepsilon = \begin{cases} 0 & \text{if } |\xi_i| \le \varepsilon \\ |\xi_i| - \varepsilon & \text{otherwise} \end{cases} \quad (5)$$

The equation (5) describes a tube with radius ε around the hypothetical regression function in such a way that if a data point (a support vector) is placed in this tube the loss function equals 0. If a data point lies outside the tube and is used to estimate regression function, the loss is proportional to the magnitude of the Euclidean difference between the data point and the radius ε of the tube. All other data points that are not support vectors are not significant to be included into the model and can be detached after the SVR model has been developed. In most cases, fewer training points make up the regression model.

2.6. Artificial Neural Network (ANN)

ANN is a series of interconnected processing elements (called artificial neurons) that are organized in layers to function together in parallel for the purpose of performing a common task [25, 26]. ANN emulates the adaptive learning and fault tolerance characteristics of the biological nervous systems to solve classification and regression tasks. The neurons of ANN are linked with each other through weighted connections that control the flow of information among the neurons. ANN is trained using available data to understand the underlying pattern. Neural Networks is able to learn from a set of examples to detect by themselves the relationships that link inputs to outputs. During training, both the inputs (representing problem parameters) and outputs (representing the solutions) are presented to the network normally for thousands of cycles. At the end of each cycle, or iteration, the network evaluates the error between the desired output and actual output. It then uses this error to modify the connection weights according to the training algorithms used. After training, the network can be used to predict the solution for a new case not used in training. However, ANN has very weak explanation mechanism, which makes it difficult to understand the reasoning behind its conclusions

2.7. Related Work on Web Cost Estimation

Web projects have short schedules and very dynamic scope [27, 28]. An overview of some of previous research on web cost estimation is presented as follows. In [28], the WebMo model was developed using expert judgments and data from 46 projects using regression analysis. The WebMo model was also developed using nine cost factors and fixed power laws to estimate the effort accurately. An analysis of Web Objects by Reifer shows that these sizing metrics have many advantages in estimating the developmental cost for web applications compared to traditional source lines of code (SLOC) with Function Points (FPs). The authors in [7] continued this research and focused on estimating the developmental attempts for web applications using Web Objects. In the paper, they investigated the applicability of Web Objects as size measurement metrics compared with traditional function points. The work by [29] was based on measuring functionality and productivity in web applications in the context of an industrial dataset. The paper showed that estimation derived using Object-oriented function points (OOFP) and lines of code (LOC) considerably achieved more than using traditional Function

points (FPs). This confirmed the earlier study, which indicated that traditional Function Points (FPs) as unsuitable for estimating productivity, as they did not take into account the reuse of components.

The authors in [30], introduced a different sizing measurement known as Full Function Points (FFPs), but this has not been subjected to full empirical evaluation. FFP is a functional measure based on standard FP techniques. The FFP transactional functions types are identified at the sub-process level, instead of the process level as is done with traditional FP. It can thus be said that FFP takes into account a finer level of granularity, (the sub process level), while FP only considers the process level. In the study, the author claimed without any empirical results that FFP's are the most elastic method for counting the functional size of web applications. From extensive testing and analysis, it has been shown that at the early stage MMWA produces results, which are accurate in estimating the developmental effort of web-based applications. However, this sizing measure has not gained any popularity or continuity from other researchers in a web application development context as W2000 is used as the design framework. The W2000 design framework uses a consolidated methodology or systematic approach to design web applications. By using this framework, it is hard to collect the data of previous projects and it is therefore not relevant for web development estimation. The trend discussed in the literature mainly focused on Web Objects and Function Point Analysis as sizing measurements. However, some recent researches have been conducted such as Case-Based Reasoning [31], Artificial Neural Networks (ANN) [4] and Support Vector Machine [6]. All of these are ML approaches that provide the basis for developmental effort estimation in contrast to algorithmic models.

3. Overview of Methodology

To carry out the objectives of this work, a two-part comparison of the three machine techniques described previously was undertaken. First, empirical evaluation of performance of CBR, ANN, and SVR when applied to the same dataset; and second, a comparative analysis of application procedure and outcome of the three method based on observed similarities and differences from the three experiments.

For the empirical evaluation, experiment was performed on an existing project dataset for Web hypermedia following the example of [32]. The dataset comprised 34 software projects obtained from the tukutuku[1] website. The dataset consist of different metric parameters that are relevant for the estimation of Web hypermedia. The description of these metric parameters is shown in Table 2.

Table 2. Prediction Parameters for Web Hypermedia

s/no	Variable	Description
1	*Page Count (PAC)*	total number of HTML files used in the application
2	*Program Count (PRC)*	the number of program code units files and java scripts used in the web application
3	*Reused Program Count (RPC)*	The number programs that have been reused or modified
4	*Total Page Complexity (TPC)*	The average number of different types of media per page
5	*Media Count (MEC)*	The number of files used in the web application
6	*Reused Media Count (RMC)*	The number of media files that have been reused

		or modified
7	*Connectivity Density (COD)*	The total number of Internal links divided by the page counts
8	*Total Effort (TE)*	The amount of effort in person hours to design and develop the application

Total Effort (TE) is the dependent variable while others parameters are independent variables. Two features out of the 34 projects contained missing values that are up to 40%, so these features were not included in the experiment. The datasets were pre-processed according to the dictates of a particular technique and software tool used.

3.1. Procedure for CBR Experiment

For the CBR experiment, a CBR tool called CBR-WORKS was used to determine the prediction value of the effort according to jack-knife method (also known as leave one out cross-validation). This procedure is the same as the one adopted by other researchers including [31]. The algorithm in Table 3 outlines the experimental procedure.

Table 3. Procedure for Using CBR for Web Cost Estimation

1. Load data into the CBR tool (e.g. CBR-Works)
2. Identify training and test cases (Selected at random)
3. Test cases are tagged unconfirmed to exempt them from being used as part of training cases
4. Training cases are tagged confirmed and the query case tagged as unconfirmed too
5. Assign weights to each input attribute (i.e. dependent variables)
6. Identify new cases whose effort is to be estimated, tagged unconfirmed and set as query case
7. Select Similarity measure to be used (e.g. *Minimum Measure (MX), Weighted Euclidean Distance (WED), Unweighted Euclidean Distance (UE)*)
8. **If** cases in the Case Base are not exhausted **Then goto Step 6 Else**
   If cases in Case Base is exhausted **Then goto Step 9**
9. Select the most similar cases
10. Computer Estimated effort, Mean Absolute Error (MAE), Magnitude of Relative Error (MRE), Mean Magnitude of Relative Error (MMRE), and Median Magnitude of Relative Error (MdMRE)
11. End

3.2. Procedure for SVR Experiment

The WEKA tool was used for SVR experiment, proper configuration of the three meta-parameters such as “C” the penalty factor, which determines the trade off between model complexity and the training error, “ε” the loss function that controls the width of the ε-insensitive zone, used to fit the training data, and “kernel parameters” was carefully done. The procedure for the experiment is outlined in Table 4.

Table 4. Procedure for Using SVR for Web Cost Estimation

1. Convert the data file to the Attribute Relation File Format (arff)
2. Divide the data to three parts – 70% as training data, 15% as validation data, 15% as testing data
3. Select the appropriate algorithm in WEKA (SMOREG) for training the data and select kernel function and the regression optimizer function
4. Set properties of the kernel function and regression optimizer function then choose Normalize for the filter type
5. Run algorithm and output the prediction
6. Use validation data to validate the performance of the SVR model
7. Use test data to determine performance of SVR model
8. End

3.3. Procedure for ANN Experiment

The MATLAB R2010a tool was used for ANN. The models of ANN used are cascade neural network, feed-forward neural network, and Elman networks. Different ANN architectures were utilized using the same set of inputs. The ANN architecture variations were produced by modifying the number of neurons in the hidden layers empirically, this was carefully done in order to avoid over fitting which could occur due to too many neurons. Transfer function used in the input and hidden layer was hyperbolic tangent sigmoid while for output layer linear function was used. ANN was trained with the gradient descent back-propagation algorithm. The number of training epochs was set to 1000, while the conjugate gradient technique was used to update the weight values. The procedure outline for ANN is presented in Table 5.

Table 5. Procedure for Using ANN for Web Cost Estimation

1. Collect data for previously developed projects
2. Divide the dataset into three parts – training set (70%), validation set (15%), and test set (15%)
3. Design the ANN with the number of neurons in the input layer based on by the unique characteristics that should determine the outcome of the prediction
4. Pre-process the data through normalization of numeric data, 1-of-c encoding of categorical data and probability ratio for ordinal data.
5. Input the training set to train the network
6. After training then validate the trained ANN using the validation dataset
7. Evaluate the performance of the network by using the testing dataset
8. **If** network performance is not satisfactory **then** go back to Step 3
9. **If** network performance is satisfactory **then** Simulate ANN with data from new project and obtain prediction for new project
10. End

4. Description of the Three ML Experiments

In this section, we present a description of each of the machine learning experiments that was performed using CBR, SVR, and ANN.

4.1. Conducting the Experiments

In the CBR experiment, the leave one out cross-validation was used, which is a way of validating the error of the prediction procedure employed. [33]. CBR-WORKS carry out normalization of the data. Procedure below was repeated thirty four times because the numbers of finished project in the case based were 34. Note that the actual effort of the project was known.

- **Procedure 1**: The chosen case i (i = 1 to 34) was removed from the case base by marking it *unconfirmed*. Marked case is regarded as a new project.
- **Procedure 2**: Remaining thirty-three cases that are not marked as *unconfirmed* were utilized in the processing of the estimation.
- **Procedure 3**: CBR-WORKS find the closest analogies, by checking for the most similar cases to case i in the case base.
- **Procedure 4**: unconfirmed tag was then removed from case i, and the case is added back to the case base and another case was then chosen. At completion of the four procedures, MRE was evaluated. The result of the experiment is enumerated in the Table 6.

Table 6. Results Obtained from Comparing Different Similarity Metrics

SIMILARITY MEASURE	ANALOG(K)	ADAPTATION	MMRE	MdMRE	PRED(25)
UE	1	CA	0.12	0.40	0.74
	2	MEAN	0.15	0.16	0.64
		IRWM	0.13	0.21	0.54
	3	MEAN	0.14	0.12	0.77
		MEDIAN	0.13	0.34	1.00
		IRWM	0.14	0.23	0.70
WE	1	CA	0.10	0.07	0.75
	2	MEAN	0.13	0.11	0.74
		IRWM	0.12	0.12	0.77
	3	MEAN	0.13	0.08	0.71

		MEDIAN	0.15	0.34	1.00
		IRWM	0.14	0.18	0.74
MIN	1	CA	0.14	0.19	0.61
	2	MEAN	0.11	0.15	0.68
		IRWM	0.11	0.17	0.78
	3	MEAN	0.32	0.11	0.81
		MEDIAN	0.23	0.25	1.00
		IRWM	0.31	0.14	0.71

This result was acquired by distance Un-weighted Euclidean Distance (UE**)**, Weighted Euclidean Distance (WE**)**, Minimum Measure (MIN**),** three analogies (1, 2 and 3) and Closet analogy (CA), mean, inverse rank weighted mean (IRWM) as analogy adaptation. WE depict better estimation than UE. Closest analogy gives the best cases when compared with other analogy measures. MIN offered the worst results indicating one sole size measure will not give adequate choice. In conclusion, the estimate for UED and WED were good. MMRE $\leq$ 25% with Pred (25) $\geq$ 75%" suggests good accuracy level so the suggested results were considered [34].

4.2. Conducting the SVR Experiment

The actual application of SVR requires the choice of the values for two sets of parameters: one regarding the Support Vector algorithm including the choice of ε (loss function) and C (penalty factor) and a second set regarding the specific kernel adopted. In the experiments, various parameter settings were tested, the WEKA tool takes as input either an Attribute Relation File Format (arff) or Comma Separated Values (csv) file containing a sparse matrix that represents a training dataset of *n* features and a set of parameters that allow the user to choose the desired kernel, ε and C to train an SVR model. Then, as a second step, the tool takes input of a new arrf/csv file with a sparse matrix representing a test dataset and uses it to generate the predictions for the missing feature exploiting the previously trained model. The result of the experiment is shown in the Table 7.

Table 7. Results Obtained from Comparing Different Kernel Functions

Performance Criteria	SVR(Polynomial kernel)	SVR(RBF
MAE	1.784	53.362
MMRE	0.1784	5.3362
PRED	0.90	0.6

The results demonstrate that Polynomial kernel gives better prediction compared to Radial Basis Function (RBF) kernel.

4.3. Conducting the ANN Experiment

The first step that was taken was to identify and tag the data as input or as output. This was done in Microsoft Excel by identifying the independent variables as the input parameters. In this case data values of PAC, PRC, RPC, TPC, MEC, RMC, COD (see Table 1) and the corresponding dependent variable, which is the total effort (TE) in each case as obtained from the data from the 34 projects.

Variant Artificial Neural Network (ANN) architectures were used, which include Feed-forward Network, Cascade Forward Network, and Elman Neural Network. A feed-forward network has input, hidden and output layers, with weight connection between successive layers in a forward direction. A Cascade-forward networks are similar to a feed-forward network, but include a weight connection from the input to every other layer in the network, while every previous layer in the network is also connected to the following layers. The Elman neural network is a simple recurrent neural network (SRN) that has four layers – an input layer, hidden layer, output layer, and a context layer. The units in the context layer send input to corresponding units in the hidden layer, while each unit of hidden layer send their output back to each unit of the context layer. All the networks were trained using variant back propagation algorithm, with the available data divided into three sets (training, validation, and test set). Samples of inputs and corresponding outputs from the training set were presented to the networks in order to achieve back error propagation learning. The validation set was used to measure network generalization, and to halt training when generalization stops improving. Whilst, the test set was used to determine the generalization ability of the network and evaluate network performance.

The available data from 34 projects were divided randomly into three sets with the following ratio. Training set (includes 24 projects ≈ 70%); Cross validation set (includes 5 projects ≈ 15%). Test set (includes 5 projects ≈ 15%). The implementation was done using MATLAB, and the result obtained is shown in Table 8.

Table 8. Result Obtained from Comparing Different ANN Models

	ANN Models			
Performance Criteria	Cascade Forward ANN	Elman ANN	Recurrent ANN	Feed Forward ANN
MAE	2.083	31.74	24.15	15.46
MMRE	0.2083	3.174	2.415	1.546
Pred (0.25) %	90	30	50	60

The simulation results confirmed that Cascade neural network gave the best performance, among the four ANN models. Feed-forward ANN gave the second best results and Elman ANN gave the worst result. In general, no estimation method is full-proof and hundred percent accurate.

5. Comparative Analysis of the Machine Learning Techniques

In this section, we present the best results obtained from the three experimental procedures using CBR, SVR, and ANN in order to compare the performance of the three machine learning techniques for web cost estimation. We also discussed the conceptual similarities and differences of the three ML techniques based on our observations while applying the three ML for web cost estimation.

5.1. Analysis of Performance of ML Techniques

Following the suggestion by [34], that MMRE ≤ 25% and PRED ≥ 75% should be used as criteria for acceptable model performance, by comparing the best individual results from the three experiments (see Table 9), we saw that Support Vector Regression (SVR) had a better performance in terms of MMRE and PRED when compared to CBR and ANN. Hence, SVR gives a better accuracy in predicting the actual effort for web cost development based on the data sample that was used.

Table 9. Comparison of Performance of CBR, SVR, and ANN for WCE

SVR(Polynomial)			CBR(Weighted)			ANN(Cascaded)		
M	MM	PR	M	MM	PR	M	MM	PR
1.	0.17	0.9	1.	0.10	0.7	2.	0.20	0.8

5.2. Analysis of Conceptual Similarities of ML Techniques

In terms of conceptual similarities between the three ML techniques observed while using them for WCE, we can deduce certain facts about the characteristics of the three ML techniques, which also aligns with established notions in the literature (see Table 10).

Table 10. Conceptual Similarities of CBR, SVR, and ANN

Characteristics	CBR	SVR	ANN
Mode of Learning	Supervised learning	Supervised learning	Supervised learning
Need for Data Preprocessing	Yes – data can be numbers or strings	Yes – data must be preprocessed to numbers	Yes – data must be preprocessed to numbers
Use of Cross Validation	Yes	Yes	Yes
Use of Standard Training Algorithms	Similarity based	Optimization based	Optimization based
Type of Problem solved	Regression, classification	Regression, classification	Regression, classification
Inductive Capability	Yes – similarity based	Yes – Error reduction	Yes – Error reduction
Instance-based Learning	Yes – instance-based learning is by creating local approximation	Yes - instance-based learning is by creating global approximation	Yes – instance-based learning is by creating global approximation

Supervised Learning: The three ML techniques - CBR, SVR and ANN - require sample inputs and desired outputs in order to be trained, which is the attribute of supervised learning. After training, its trained model needs to be validated by separate data and tested to know how well the assumption generated by the learner generalizes to new examples.

Data Preprocessing: All the three techniques learn faster and give better performance if the input data are pre-processed prior to using them. ANN and SVR require the data to be pre-processed by linear scaling. The goal of linear scaling is to independently normalize each variable to the specified range. Normalization helps to avoid having the

larger value input variables dominate smaller values inputs, and avoids numerical difficulties during the calculation. Hence, this reduces prediction errors. In this study, all data for SVR was scaled to [0, 1] while ANN was scaled to [1, -1] (for *tanh* activation function to avoid fluctuations in the mathematical calculations of an ANN system). CBR can handle both numbers and string data, but in this study, numeric data was used.

Cross Validation: Cross validation method was used while working with the three ML techniques in order to validate the error of the prediction procedure [33]. In each case, a cross validation set was used for generalization in order to produce better output for unseen examples [35].

Training: The standard training algorithm used for SVR and ANN are optimization (Seeking to minimize or maximize a real function by systematically choosing the values of real or integer variables from within an allowed set *i.e.*, Choosing the best element among a set of available alternatives) while CBR uses similarity (computing distance between cases) among cases.

***Type of Problem*:** The three ML techniques can solve regression and classification problems.

Inductive Capability: The three ML techniques have inductive capability that is acquired via supervised learning. SVR and ANN provide solution to a given task by error-reduction while CBR is based on similarity measure.

Instance-based Learning: CBR uses lazy method for instance-based learning. The approximation is local and tends to defer the decision on how to generalize beyond the training data until each new query instance is encountered. In the case of SVR and ANN, they use eager method for instance-based leaning, which generalizes beyond the training data before observing the new queries *i.e.*, they abstract from the training data a general rule to use when processing new queries.

5.3. Analysis of Conceptual Similarities of ML Techniques

The conceptual differences between the three ML techniques based on our experience while conducting the three experiments are presented in Table 11. This also aligns with many of the established notions about the three techniques in the literature.

Table 11. Conceptual Differences of CBR, SVR and ANN

Differences	**CBR**	**SVR**	**ANN**
Handling of non-linear relationship / coarse data	Not so good	Good	Good
Selection of training parameters	Structured, based on type of data	Ad hoc – trial by error	Ad hoc – trial by error
Approach to solving Regression	Nearest Neighbour hood	Optimization	Optimization
Mode of Generalization	Local	Global	Global
Explanation Capability	Good	Poor	Poor
Effectiveness with Sparse Data	Poor	Good	Good
Nature of Technique	White box	Black box	Black box

From Table 11, we observed the following:

Handling of Non-linear relationship / coarse data*:* SVR and ANN have more in common in terms of their ability to handle non-linear and coarse data and have significant strength in this area compared to CBR

Selection of training parameter: while there is a structural approach for selecting the training parameters for CBR, such activities with regards to SVR and ANN are experimental and based on trial and error. However, SVR require less amount of parameter selection during training compared to ANN

Approach to solving Regression: Both SVR and ANN used an optimization approach, while CBR uses nearest neighbourhood derived by computing the degree of similarity

Mode of Generalization*:* While CBR generalizes locally, both SVR and ANN are global.

Explanation Capability: CBR have good explanation mechanism, which facilitate understanding of relationship among attributes and eventual results. This enables more user-involvement in performing the estimation task. In contrast, SVR and ANN aside from predicted result that is returned, do not provide any other details pertaining to attribute relationships, influence of attribute on the effort, and relevance of attribute, which can aid user's understanding.

Effectiveness with Sparse Data: Not having large amount of data is not a disadvantage for both SVR and ANN, but this is a weakness for CBR.

Nature of Technique**:** Generally, the use of CBR is more transparent and it is easier for the user to understand the inner workings of the system in terms of nature of computation and strength of relationship among attributes. SVR and ANN do not reveal the internal workings of the system but only returns relatively accurate estimates. It will be more difficult to ascertain whether the model has been built correctly, and validate whether the correct model has been built.

6. Conclusion

In this paper, an experimental assessment of the performance and applicability of three prominent machine learning technique for web cost estimation (WCE) has been carried out.

We did this with the objective of not only determining the relative performance accuracy, but to identify what the techniques have in common and how they differ in the context of applying them for WCE.

The findings from the study show that SVR, when used with the polynomial kernel function have a slightly better performance compared to cascaded ANN, while CBR is the least accurate. In addition, SVR require fewer parameters to be selected, which are the kernel function (KF), and loss function (ε) and penalty factor (C) that control the difference between model complexity and the training error during training. For ANN, there is the need to specify more parameters using trial and error such as number of hidden layers, hidden neurons, bias input, a learning algorithm and a transfer function for the architecture, and learning rate. In contrast, CBR does not need the combination of parameters to build up its prediction model. It simply retrieves cases from a case base and uses simple feature similarity and case similarity formulas, it does not forecast from scratch. While the predictive power of CBR is lower when compared to SVR and ANN, it has better explanation mechanism, which makes its results easier to validate, thus CBR enables more efficient user participation. Although, data preprocessing is important for all three ML techniques, it is less demanding for CBR.

Generally, the three ML techniques have some conceptual similarities but there are more conceptual similarities between SVR and ANN on one hand compared to CBR.

Indeed, in most cases, either SVR or ANN would be worthy alternative techniques for web cost estimation. In addition, significant conceptual differences exist between CBR, SVR and ANN, although the differences between SVR and ANN are fewer. The recommendation from this study is that one technique is not adequate for all situations, but a cautious combination of techniques is likely to generate sensible estimates. We do hope that with this study, professionals, and future researchers will be guided not just by the performance of these three ML techniques but also on the mode of application of these techniques for WCE, noting their differences and similarities in making a profitable decision.

In future work, we shall explore the possibility of developing hybrid Neural CBR or SVR CBR platforms that will leverage the strengths of a hybrid of machine learning techniques for web cost estimation from a single platform.

Acknowledgements

The Covenant University Centre for Research Innovation and Development (CUCRID) supported this research.

References

[1] E. Mendes, "An Overview of Web Effort Estimation", Advances in Computers, vol. 78, **(2010)**, pp. 223-270.
[2] G. Huang, "Cost Modeling Based on Support Vector Regression for Complex Products During the Early Design Phases", Blacksburg, Virginia, **(2007)**.
[3] E. Mendes, S. Counsell and N. Mosley, "Towards the Prediction of Development Effort for Hypermedia Applications", HT'01 **(2001)**, pp. 249–258.
[4] C. S. Reddy and R. KVSVN, "An Optimal Neural Network Model for Software Effort Estimation", International Journal of Software Engineering, vol. 3, no. 1, **(2010)**, pp. 63-78.
[5] A. Corazza, S. Di Martino, F. Ferrucci, C. Gravino and E. Mendes, "Using Support Vector Regression for Web Development Effort Estimation", In Software Process and Product Measurement, Springer Berlin Heidelberg, **(2009)**, pp. 255-271.
[6] Corazza, S. Di Martino, F. Ferrucci, C. Gravino and E. Mendes, "Investigating the use of Support Vector Regression for web effort estimation", Empirical Software Engineering (2011), 16(2), pp. 211-243.
[7] M. Ruhe, R. Jeffery and I. Wieczorek, "Cost estimation for web applications", In Proceedings of 25th International Conference on Software Engineering **(2003)**, pp. 285-294, IEEE.
[8] W. G. Sullivan, D. J. Kulonda and J.A. White, "Capital investment analysis for engineering and management", Prentice Hall **(2005)**.
[9] R.T. Hughes, "Expert judgment as an estimating method", Information and Software Technology **(1996)**, 38(2): pp. 67-75.
[10] Rush, and R. Roy, "Expert judgment in cost estimating: Modelling the reasoning process", Concurrent Engineering **(2001)**, 9(4), pp. 271-284.
[11] Boehm, C. Abts, and S. Chulani, "Software development cost estimation approaches—A survey" Annals of Software Engineering **(2000)**,10(1-4), pp.177-205.
[12] B. W. Boehm, "Software engineering economics", (Vol. 197), Englewood Cliffs (NJ): Prentice-hall (1981).
[13] L. H. Putnam, "A general empirical solution to the macro software sizing and estimating problem", IEEE Trans. Software Eng,(1978), 4, pp.345-361
[14] L.C. Briand, and I. Wieczorek, "Software Resource Estimation", Encyclopedia of Software Engineering. Volume 2, P-Z (2nd ed., 2002), Marciniak, John J. (ed.) New York: John Wiley & Sons, **(2002)**, pp. 1160-1196.
[15] G.D. Boetticher, "An assessment of metric contribution in the construction of a neural network-based effort estimator. **(2001)**, http://sce.uhcl.edu/boetticher/SCASE01.pdf
[16] Schofield, "An empirical investigation into software estimation by analogy", Unpublished Doctoral Thesis, Dept. of Computing, Bournemouth Univ., UK **(1998)**.
[17] Chao W. L., J. J. Ding, "Integrated Machine Learning Algorithms for Human Age Estimation", NTU **(2011)**.
[18] S.J. Huang, and N. H. Chiu, "Applying fuzzy neural network to estimate software development effort", Applied Intelligence, **(2009)**, 30(2), pp. 73-83.
[19] Mendes, I. Watson, C. Triggs, N. Mosley, and S. Counsell, "A comparative study of cost estimation models for web hypermedia applications" Empirical Software Engineering, **(2003)**, 8(2), pp. 163-196.

[20] Mendes, N. Mosley, and S. Counsell, "Comparison of Web size measures for predicting Web design and authoring effort" In IEE Proceedings of Software, (**2002**, June), 149(3), pp. 86-92), IET.
[21] E. Mendes, (Ed.), "Cost estimation techniques for web project", **(2007)**, IGI Global.
[22] A. Aamodt, and E. Plaza, "Case-based reasoning: Foundational issues, methodological variations, and system approaches", AI communications, **(1994)**, 7(1), pp. 39-59.
[23] V.N. Vapnik, "Statistical Learning Theory" Wiley-Interscience **(1998)**.
[24] A. J. Smola, and B. Schölkopf, "A tutorial on support vector regression", Statistics and computing, **(2004)**,14(3), pp. 199-222.
[25] S. Ayed, "Parametric cost estimating of highway projects using neural networks", National Library of Canada, Bibliothèque nationale du Canada **(1999)**.
[26] G. Musa, O. Daramola, E. A. Owoloko, and O. O. Olugbara, "A Neural-CBR System for Real Property Valuation", Journal of Emerging Trends in Computing and Information Sciences, **(2013)**, 4(8), 611-622.
[27] D. Lowe, "Web system requirements: an overview", Requirements Engineering, **(2003)**, 8(2), 102-113.
[28] D. J. Reifer, "Web development: estimating quick-to-market software", Software, IEEE, **(2000)**, 17(6), pp.57-64.
[29] M. Morisio, I. Stamelos, V. Spahos, and D. Romano, "Measuring functionality and productivity in web-based applications: a case study", In Proceedings of Sixth International Software Metrics Symposium, **(1999)**, pp. 111-118, IEEE.
[30] T. Rollo, "Functional size measurement and COCOMO—a synergistic approach", In Proceedings of Software Measurement European Forum (SMEF), Rome, Italy, **(2006)**, pp.259–267.
[31] E. Mendes, N. Mosley, "Further Investigation into the Use of CBR and Stepwise Regression to Predict Development Effort for Web Hypermedia Applications", Proceedings of the International Symposium on Empirical Software Engineering **(2002)**, pp.79-136
[32] E. Mendes, N. Mosley and S. Counsell, "Investigating early web size measures for web cost estimation", In Proceedings of EASE'2003 Conference, Keele **(2003)**, pp. 1-22.
[33] L. Angelis, L. and I. Stamelos, I, "A simulation tool for efficient analogy based cost estimation", Empirical Software Engineering (2000), 5, pp.35–68.
[34] S. Conte, H. Dunsmore, and V. Shen, "Software Engineering Metrics and Models", Benjamin/Cummings, Menlo Park, California **(1986)**.
[35] J. Sodikov, "Cost Estimation of Highway Projects in Developing Countries: Artificial Neural Network Approach", Journal of the Eastern Asia Society for Transportation Studies, **(2005)**, 6, pp.1036 - 1047.

Authors

Olawande Daramola, He is an Associate Professor of Computer Science in the Department of Computer and Information Sciences of Covenant University Nigeria. His research interests include the Semantic Web, Intelligent Systems, and Automated Software Engineering.

Ibidapo Akinyemi, He is a faculty in the Department of Computer and Information Sciences, Covenant University, Ota, Nigeria. He holds Bachelor of Science degree in Mathematical Sciences (Computer Science Option) and Masters of Science degree in Computer Science from University of Agriculture, Abeokuta, Ogun State, Nigeria. He had his PhD degree in Computer Science from Covenant University, Ota, Nigeria. His areas of research are on Computational Intelligence, Mathematical modeling, Databases and Software Engineering.

Ibidun Ajala has a Bachelors and Master's degree in Computer Science. She works as a Data Administrator in Nigeria.

