Computer Science Review 38 (2020) 100288

Contents lists available at ScienceDirect

Computer Science Review

journal homepage: www.elsevier.com/locate/cosrev

Review article

Advancement from neural networks to deep learning in software effort estimation: Perspective of two decades

P. Suresh Kumar [a,*], H.S. Behera [a], Anisha Kumari K [b], Janmenjoy Nayak [c], Bighnaraj Naik [d]

[a] Department of Information Technology, Veer Surendra Sai University of Technology, Burla, 768018, India
[b] Deptartment of CSE, Dr. Lankapalli Bullayya College of Engineering (W), Visakhapatnam, 530013, India
[c] Department of CSE, Aditya Institute of Technology and Management (AITAM), Tekkali, AP-532201, India
[d] Department of Computer Application, Veer Surendra Sai University of Technology, Burla, 768018, India

A R T I C L E I N F O

Article history:
Received 28 February 2020
Received in revised form 18 July 2020
Accepted 27 July 2020
Available online 14 August 2020

Keywords:
Software effort estimation
Artificial neural networks
Higher order neural networks
Deep learning neural networks

A B S T R A C T

In the software engineering, estimation of the effort, time and cost required for the development of software projects is an important issue. It is a very difficult task for project managers to predict the cost and effort needed in the premature stages of planning. Software estimation ahead of development can reduce the risk and increase the success rate of the project. Many traditional and machine learning methods are used for software effort estimation by researchers, but always it has been a challenge to predict the effort accurately. In this study, different Artificial Neural Network (ANN) used for effort estimation is discussed. It is observed that the prediction of software effort by using ANN is more precise and better compared to traditional methods such as Function point, Use-case methods and COCOMO etc. Models based on neural networks are competitive in nature as compared to statistical and traditional regression methods. This paper explains the overview of various ANN such as basic NN, higher order NN, and deep learning networks used by the researchers for software effort estimation.

Contents

* Corresponding author.
E-mail addresses: reshu.suri@gmail.com (P. Suresh Kumar), hsbehera_india@yahoo.com (H.S. Behera), anishakushwaha14@gmail.com (Anisha Kumari K), mailforjnayak@gmail.com (J. Nayak), mailtobnaik@gmail.com (B. Naik).

https://doi.org/10.1016/j.cosrev.2020.100288
1574-0137/

1. Introduction

The urge for software projects has been expanding day by day, resulting in continuous software and hardware development. A growth in the demand for software projects has increased the competition among the companies to produce high quality and low-cost products in a short time [1]. Predicting the cost and effort needed for the evolution of a software project is essential and has to be done at the initial stage of development process, preferably during planning and requirement analysis phase for better analysis of project details, resource allocation, scheduling of project task, and monitoring the process [2]. Software estimation involves size, effort, time, cost, and staff [3]. The basic estimation in software projects are (1) development project size, (2) person–month or person–hour, (3) development period and (4) cost of the project [1].

Software organizations require accurate estimation of developing a project, as software demand is increasing. Successful completion of project within time is a crucial task in information technological industries. Hence, software effort estimation is a major task for managing complex and large software projects [4]. Accurate software estimation enables the project manager to plan and allocate resources effectively. Overestimation may lead to misallocation of resources which affects the development of other important projects. Whereas underestimation would be a loss of money and the project could not be completed or the finished product may not reach the expectations of the company and would be of poor quality [5].

Machine learning is appropriate as it is useful to increase the accuracy of training the rules by conducting repeated cycles. The most frequently used machine learning techniques in software effort estimation are linear regression, decision tree, logistic regression, KNN, SVM, Naive Bayes, etc. These techniques give different accuracies on different datasets [6]. So, it is very hard to conclude the best technique with required accuracy level. In spite of that, fair amount of research is done on Software Effort Estimation (SEE) using machine learning methods. From the literature, it is observed that the machine learning methods are capable of over traditional methods in terms of accuracy. SEE is generally based on techniques such as expert judgment, algorithmic models, and various machine learning methods. This expert judgment is a heuristic approach, where the estimation is done by comparing the new project with old projects and it is dependent on estimator's experience [7]. Algorithmic methods such as functional points, constructive cost model and software lifecycle management use mathematical equations for analysis of historical data and project attributes. COCOMO is the most frequently used model for the estimation of effort as it estimates in person–month using software attributes. ANN techniques have proved their significance in estimating the software [8]. In situations with the complex relationship between input and output, neural networks are acknowledged to produce accurate estimation as they map in a nonlinear fashion [9,10]. Hence, neural networks are applied for prediction problems by many researchers extensively. Machine learning techniques are preferred over non-machine learning techniques as their functionality is dependent on the dataset used. They increase the estimation accuracy by training the model and repeating it until no further improvement.

This article is a systematic literature survey on the different neural networks used for software effort estimation which includes (i) the popular techniques used for prediction along with the dataset used for conducting the experiments, (ii) the evaluation measures used to compare and (iii) the findings along with the conclusion of their studies. The studies included in this paper are selected by performing a systematic literature review (SLR). The models under studies includes feed forward neural network with back propagation, Radial basic neural network, General regression neural network, Wavelet neural network, Functional link neural network, Spiking neural network, Long-short term memory neural network, and Elman neural network. Further, this study highlights the most commonly used methods along with frequently used dataset and evaluation metrics. Further, a questionnaire method has been adopted with 11 relevant questions regarding software effort estimation to analyze the research. The remaining paper is organized into 5 sections: Section 2 describes the systematic literature review conducted; Section 3 explains the various networks being neural networks, higher-order neural networks, and deep learning techniques; Section 4 shows the application of neural networks, higher-order neural network and deep learning techniques used for effort estimation; Critical analysis and investigation are done in Section 5; and Section 6 concludes the paper.

2. Procedure conducted in this Systematic Literature Review

In this paper, the Systematic Literature Review (SLR) process is followed by the guidelines given by [11]. The data extraction phase is to analyze the data extracted from the recorded information. Entire information collected to address the research questions is included in the design. Data synthesis is the process of summarizing the extracted data and collating it. The data are usually represented through a table or by plotting a graph. The final report states the focus of the systematic review, research protocol and the conclusion of the review briefing the study.

Planning, execution, and result analysis are done through the following phases:

1. Purpose of Review & Questions
2. Search Strategy
3. Study Selection
4. Study Quality Assessment
5. Data Extraction

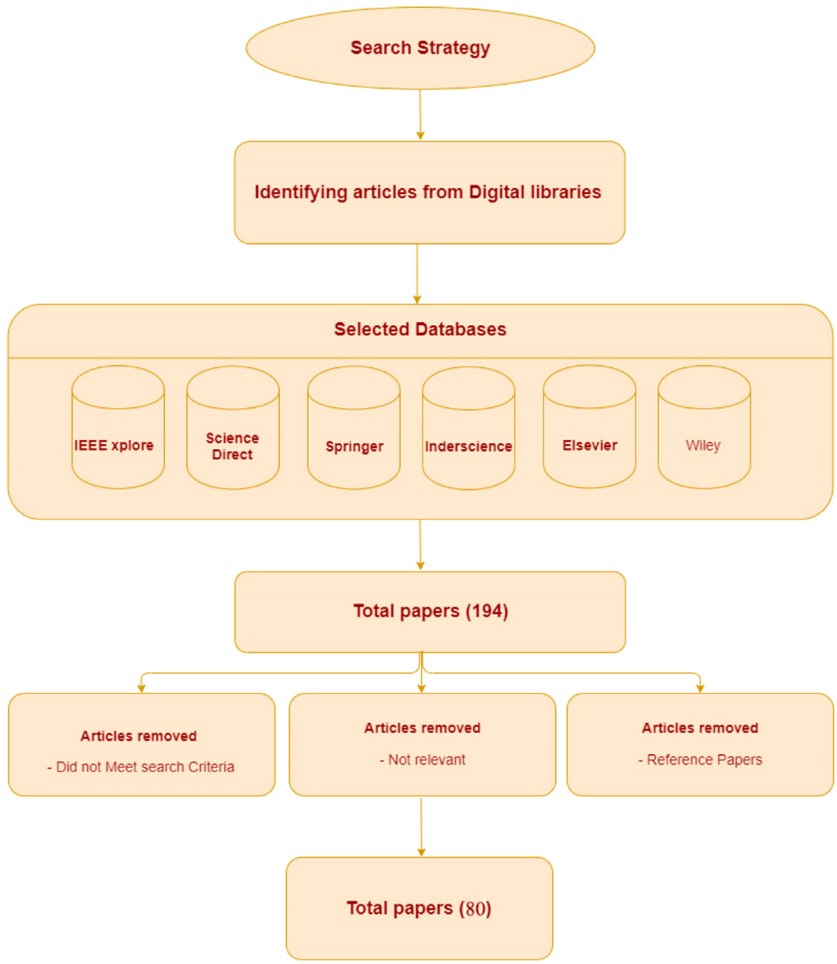

Fig. 1. Systematic literature review.

6. Data Synthesis
7. Reporting the Review

The most important phase and objective of SLR is to formulate research questions. This study focuses on machine learning techniques mostly neural networks used for software effort estimation.

Formulated research questions are as follows:

1. Which dataset is used widely in predicting SEE?
2. Which technique is mostly used in predicting SEE?
3. Which accuracy measure is used mostly in SEE?
4. Which combination techniques are mostly used in predicting SEE?
5. Which publications are dominant in the area of SEE?
6. Which year the substantial research is done in SEE?
7. Which type of software effort is mostly considered?
8. Which combination (dataset vs intelligent method) is used mostly used in predicting SEE?
9. Which intelligent method's performance is better on various datasets?
10. Comparative analysis of previous survey studies with the proposed article.
11. What are the various future suggestive techniques for SEE?

In second phase, the search strategy has been followed to identify the studies which could help us to answer our research questions. Primary search is conducted by major terms such as 'effort estimation' and 'neural networks'. The synonyms and alternatives of the major terms such as 'prediction', 'predicted' and 'estimated' are considered for the research. The various methods fulfilling the objectives are also included. The search string with 'Boolean OR' connecting all synonyms and 'Boolean AND' connecting the major terms would result in all possible studies. Machine learning techniques, hybrid neural network techniques and estimations such as cost, quality and development are considered. We selected the literatures from sources such as IEEE Xplore, Springer Link, Inderscience, Wiley and Elsevier.

The literatures acquired from the second phase are further filtered in Study Selection phase. By considering the research questions, the inclusion and exclusion process has been implemented to classify the studies accurately. The considered exclusion criteria for this study are based on the following parameters: (i) studies which uses other techniques for effort estimation rather than ANN methods, and (ii) studies employing dependent variable other than development/size/cost/time/maintenance and review studies. For example, effort estimation using hybrid technique such as neuro-fuzzy technique and other estimations such as defect estimation are excluded. The primary studies having the major terms have been included, and the final inclusion and exclusion are done by considering the complete study. Filtration of research articles for SLR is as shown in Fig. 1.

For the credibility and applicability of the research study, quality assessment is performed. Quality assessment is an inclusion and exclusion procedure by considering many aspects of research questions. Quality assessment is done by considering some factors such as: whether the goal of the research stated clearly, which datasets are considered for experiment, which estimation

Table 1
Various neural networks in software effort estimation.

S. No	Author(s)	Contribution	Intelligent method	Datasets	Evaluation factors	Ref
1	P. S. Rao et al.	Proposed MLPNN with ABC, and compared with MLPNN, and MLPNN with PCA	MLP, PCA, ABC	COCOMO	MMRE, MdMMRE	[12]
2	Ali Idri, Abran, et al.	Compared the performance of RBFN with c-means and RBFN with APC-III	RBFN, C-means, APC-III	COCOMO'81	PRED, MMRE	[13]
3	Rao and Kumar	Proposed Generalized Regression Neural Network with PCA, and compared with M5, linear regression, SMO poly kernel and RBF kernel	GRNN with PCA	COCOMO'81	MMRE, MdMMRE	[14]
4	Vinay Kumar et al.	Proposed WNN and compared with MLP, RBFN, MLR, DENFIS, and SVM	WNN with Morlet function, WNN with Gaussian function, TAWNN	CF, IBMDPS	MMRE	[15]
5	Benala et al.	Proposed FLANN with C-means and compared with FALNN with k-means, SWR, and CART	FLANN with fuzzy C-means	COCOMO'81, Nasa93, Maxwell	MMRE, MdMRE, PRED (0.25)	[16]
6	Venkataiah et al.	Proposed SNN and compared with RGP, GP-RGP, GP–GP, GMDH-GP, GP-GMDH, PSO–ABE	Spiking neural networks	IBMDSP, ISBSG-10, China	RMSE, MMRE	[17]
7	Choetkiertikul et al.	Proposed Deep learning with story points, compared with lstm+rhn, lstm+rf, lstm+svm, lstm+atlm, lstm+lr	Deep learning with story points	16 open source projects	MAE, MdAE, SA	[18]
8	Praynlin and Latha	Proposed ELMAN neural Network and compared with back propagation algorithm	ELMAN neural network	NASA	MRE, MMRE, PRED(25), RMSE, Error	[19]

accuracy are measured. Other factors that are considered are (i) comparison of the proposed method with other counterparts, (ii) latest publication, and (iii) citation count, etc. The studies which do not or rarely provide this information are excluded. However, any studies which provide this information partly, mostly and completely are included in this study.

3. Preliminaries

An Artificial Neural Network (ANN) is motivated by the human neuron system, is an information-processing model similar to brain processing information. ANN consists of a large number of interconnected elements termed neurons with parallel function and has a regular structure. It mainly subsist of three layers input layer, hidden layer, and output layer [20]. Each layer has interconnected neurons with linked weights and activation functions. The data in each layer transforms passing to the next layer. ANN can learn from its previous inputs and performs the task. Mapping of the communication among the input and output can be done by producing a function using learning algorithms [21]. These are mostly used for pattern recognition, clustering and optimizations.

3.1. Neural networks

Among the existing SEE models, the neural network is the most prominent one. Various types of neural networks have been used to increase the accuracy of the estimated effort. Neural network models such as a feed-forward neural network (FFNN) with a back-propagation learning algorithm, radial basis function neural network, and general regression neural networks (GRNN) are the commonly used neural networks play prominent role in SEE. FFNN excel when used for small scale projects [22,23]. It is found that the general regression neural network surpasses the feed-forward neural network and fuzzy logic [24]. In other studies, it is proved that radial basis function neural network outperforms both feed forward and general regression neural network. Applications of neural network, higher order neural networks, and deep learning neural networks in software effort estimation are indicated in Table 1.

3.1.1. Feed Forward Neural Network with backward propagation

A Feed Forward Neural Network is an simple ANN where the neurons are connected such that the flow of information is only in one direction that is forward, from the starting layer that is input layer through the hidden layers and at the last it reaches output layer [25]. So, it is called a feed-forward neural network. A Completely connected FFNN with one hidden layer is enough to solve the problem with any number of parameters. completely connected means every node in every layer is connected to every node in its previous layer [23]. There is no mandate to maintain certain number of hidden layers in FFNN. A neural network with multiple hidden layers is called Multi-Layer Perceptron (MLP). Usually input layer's neurons number is equivalent to the number of input features in the model [26]. The input is given to every neuron in the starting layer where it is processed and the output is passed to every neuron in the hidden layer. The data in each hidden layer is again processed and received by the preceding hidden layer. Finally, each and every neuron that is in the output layer receives output from the last hidden layer. The final output variable values are received through the output layer [27]. If the input and the output are nonlinear then hidden layers help to facilitate generalization [28]. The variation among the target output and predicted output is an error that is assumed to be caused by the connection weights. The output of a neuron in a layer is obtained from the activation function on the sum of product of inputs and weights. The error is generated from the output of the neuron on the output layer and the error is fed backward. Finally, the associated weights of the network are adjusted and this process is repeated until the error is minimal. The procedure can be better understood with the help of Fig. 2.

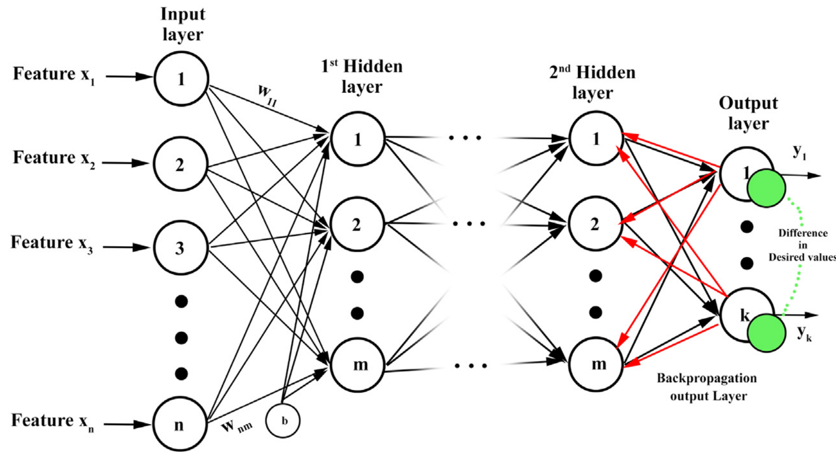

Fig. 2. MLP using back propagation neural network architecture.

The back-propagation method is the most popular technique for learning from error and weight adjustment of the network. The output of the FFNN can be calculated by Eq. (1).

$$y = x_1 w_1 + x_2 w_2 + \cdots + x_n w_n + b \tag{1}$$

Where
y is output vector
x_i is input vector
w_i is weight vector

3.1.2. Radial Basis Function Neural Network

A Radial Basis Function network consist of 3 layers, input, hidden, and output layers. Every neuron in the starting layer correlates to a predictor variable. Each neuron in hidden layers consists of a radial basis function based on a point where the dimensions are in the same number as a predictor variable. The output of the network is total sum of weighted outputs from the hidden layers and it received from the output layer [29]. Fig. 3 explains the architecture of RBF network [30]. RBF network places one or more neurons in space described by predictor variables. From the point being evaluated Euclidean distance is computed [31]. The main goal of radial basis function is applied on the distance to find the weight of the neuron. With a larger distance in between the point and the neuron, it has lower influence. The Gaussian activation function at middle neurons is a frequent choice for many networks. When the input is modeled as vector then the network's output is calculated as shown in Eq. (2).

$$\phi\left(x\right) = \sum_{i=1}^{N} w_i h_i(x) \tag{2}$$

Where $h\left(x\right) = \exp\left(-\frac{(x-c)^2}{r^2}\right)$
Here 'N' stands number of neurons in hidden layer.

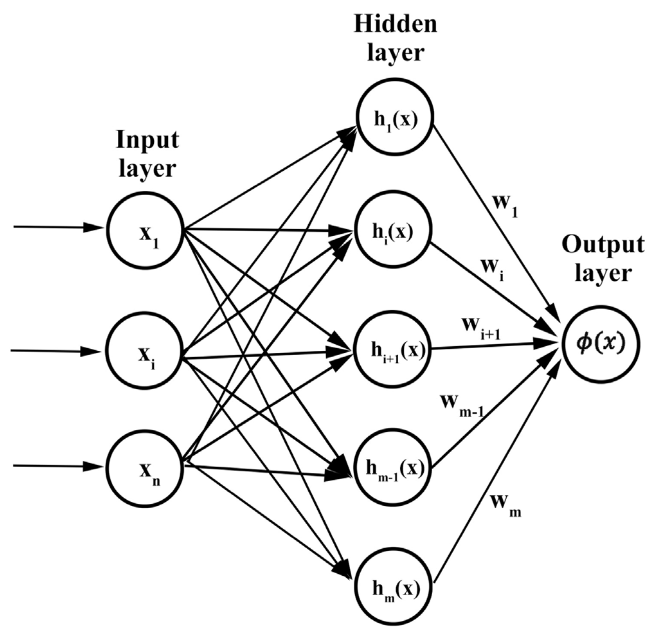

Fig. 3. Radial basis function neural network architecture.

3.1.3. General Regression Neural Network

A General Regression Neural Network (GRNN) consists of a four-layered structure. The output of a layer is fed to its next layer. The first layer has neurons each representing a predictor variable. The second layer consists of pattern neurons where each neuron represents a training row. Euclidean distance is computed from the input features vector to the center of each neuron by applying the radial basic function. The third layer consists of two nodes. The numerator node is the sum of weighs multiplied with the actual output from the previous nodes. The denominator node is the sum of the weights of the previous nodes. The rearmost layer is the output layer which has the fraction value of numerator and the denominator [31]. The architecture of GRNN is shown in Fig. 4 [32]and the output of GRNN is computed using Eq. (3).

$$y(x) = \frac{\sum_{k=1}^{N} y_k K(x, x_k)}{\sum_{k=1}^{N} K(x, x_k)} \tag{3}$$

$$K(x, x_k) = e^{-d_k/2\sigma^2}$$

$$d_k = (x - x_k)^T (x - x_k)$$

Where,
y_k is activation weight at k
$K(x, x_k)$ is radial basic function kernel

3.2. Higher order neural networks

This type of networks uses higher combinations of input and higher order activation function. Higher order neural networks expand the functionality of FFNN by giving nodes at input layer with complete knowledge of its patterns and its relations. Basically, inputs are converted in well understood mathematical form

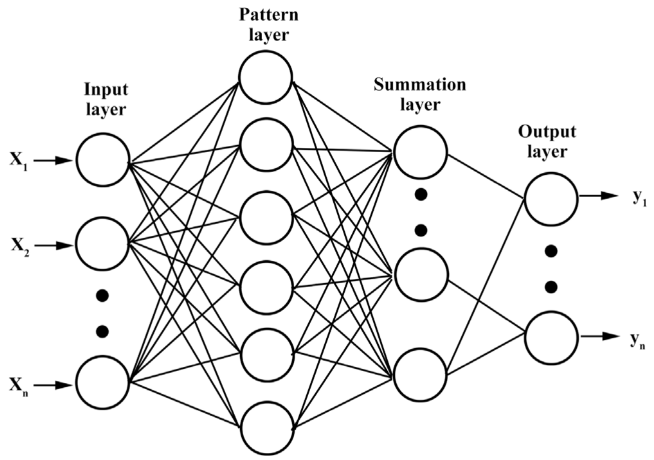

Fig. 4. General regression neural network architecture.

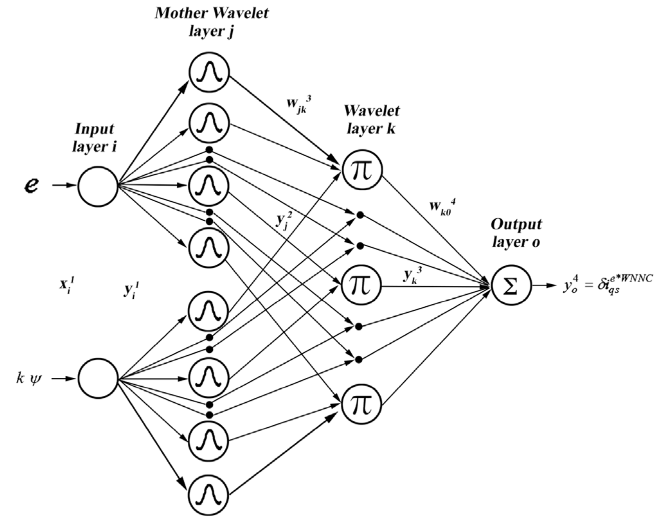

Fig. 5. Wavelet neural network architecture.

so that, there is no need to learn simple mathematical functions. The higher order functions such as squares, cubes, or sine and cosine are used for transforming inputs in training process. [33]. The input to the neuron is the weighted product's sum of its input instead of sum of its input. Such neurons are called Higher-Order Processing Unit. Higher order neural network results are more accurate compared to conventional neural networks [34]. It not only gives accurate results but also requires shorter training period to acquired desired accuracy.

3.2.1. Wavelet neural network

Wavelet neural networks are new kind of networks that is a combination of sigmoidal neural networks and wavelet analysis. It is a 3-layer network, lower layer representing input, middle layer representing hidden layer and the upper layer representing the output layer. The units in the hidden layer are referred as Wavelons. Hidden layer input variables are transferred as translated version of mother wavelet, and their activation functions are drawn from wavelet basis. The weights are modified by using learning algorithm and the wavelet coefficients are summed in the output layer to give the final result. The structure of WNN is illustrated in Fig. 5. This type of neural network is referred as Wavenet [35,36]. The output of the WNN is generated by using Eq. (4).

$$\varphi^{a,b}(x) = |\alpha|^{-1/2}\varphi\left(\frac{x-a}{b}\right) \quad (4)$$

3.2.2. Functional Link Artificial Neural Network

Functional Link Artificial Neural Networks (FLANN) increases the dimensions of input space using a non-linear combination of input to handle linearly non-separable problem [37]. The training set is prepared by increasing the input space dimensions mathematically, resulting higher order combination of input units. It does not add any new information but enhances the given input. It consists of two stages with appropriate learning algorithms in each state. The main difference between other neural networks and FLANN is that it consists of only two layers with hidden layers replaced with non-linear mapping [38]. Fig. 6 displays FLANN architecture and Eq. (5) is used to calculate the output of FLANN [39].

$$\begin{aligned} &w(k+1) = w(k) + \lambda \Pi(k)\Phi(X_k) \\ &S_j = \sum w_{ji}\varphi_i(X) \\ &y = \tanh(S_j) \end{aligned} \quad (5)$$

Where,
X is input pattern vector
w is weight vector
y is output vector

3.3. Deep Learning Neural Networks

A Deep Learning Neural Network (DLNN) is a multi-layered neural network, with a certain level of complexity. DLNN takes both linear and non-linear relationship sets as the inputs and turns it into output by using a set of mathematical principles. The probability of the output is calculated at each layer as it moves forward. At first, DLNN allocates random weights to the connections between the neurons. The input units and weights are multiplied to get the output. If the desired output is not acquired then the weights are adjusted by using the appropriate learning algorithm. DLNN allows the learning of the model both in sequential and parallel processing of information [40].

3.3.1. Spiking Neural Network

In 3rd generation of ANN, Spiking Neural Network (SNN) is one among various networks which demonstrating long-term information storing and solving complex problems [17]. SNN is greatly inspired by the communication process that neurons used for the transformation of information. In biological neurons, spikes are generated when the membrane potential reaches a certain threshold at a particular point. When the neuron spikes, it generates a signal (synaptic input) which is passed to other neurons. The increase or decrease of signal determines the potential of the neuron. SNN consists of spiking neurons with interconnected synapses and adjustable scalar weights. The spikes generated are propagated through synaptic connections. On receiving the synaptic signal, a neuron can either excitatory (increases the membrane potential) or inhibitory (decreases the membrane potential) in nature. The weights are adjusted and changed as a result of learning [41]. Fig. 7 illustrated [42] the detailed SNN architecture and mathematically it is represented in Eq. (6).

$$\begin{aligned} &\tau_m \frac{\partial u}{\partial t} = u(t) - u_r + RI(t) \\ &L\frac{\partial u}{\partial t} = -\frac{1}{R}\left(u(t) - u_r\right) + I(t) \end{aligned} \quad (6)$$

Where,
τ_m is constant time of membrane

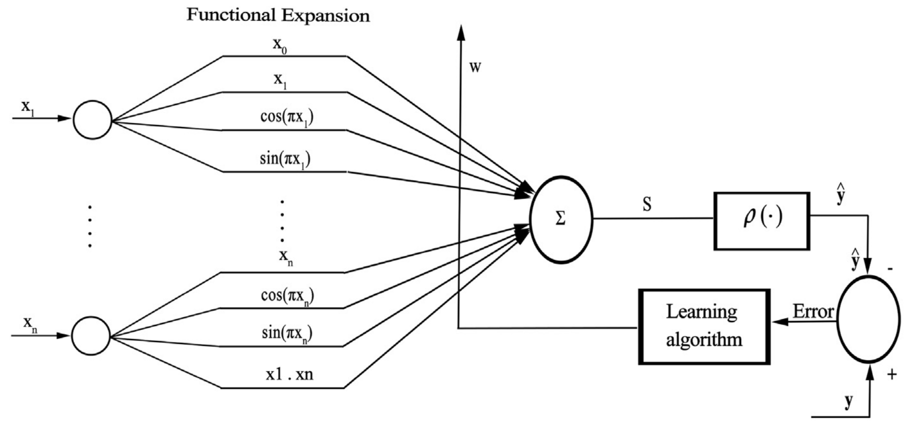

Fig. 6. Functional link neural network architecture.

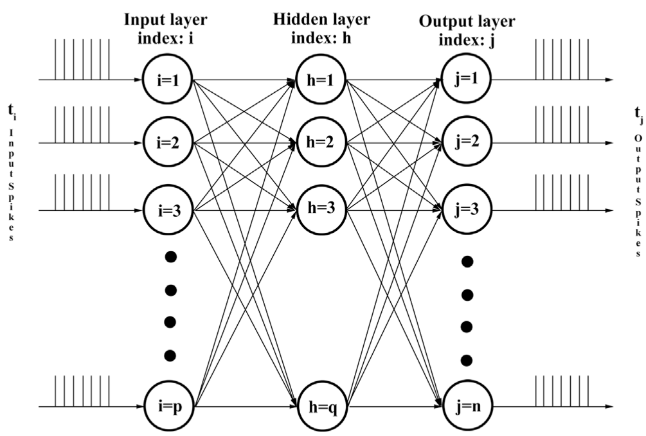

Fig. 7. Spiking neural network architecture.

R is resistance of membrane
u_r is spike reset
u(t) is membrane potential voltage
I(t) is excitatory current

3.3.2. Long short-term memory network

A recurrent neural network has a feedback connection is considered as Long Short-Term Memory (LSTM). An LSTM unit consists of 3 gates (input gate, output gate and forget gate) and a cell. The three gates control the information flow towards and away from the cell whereas the cell stores the accumulated information over arbitrary time intervals [18]. The values of previous layers are stored and used to compute the hidden states. LSTM neural network structure is explained in Fig. 8 [43]. LSTM network is well-suited for preprocessing, classification and predictions based on time-series datasets. The new state of the LSTM network can be calculated by using Eq. (7).

$$\begin{aligned}
h_t &= o_t * \tanh(C_t) \\
C_t &= \left(i_t * \widetilde{C}_t\right) + (f_t * C_{t-1}) \\
\widetilde{C}_t &= \tanh\left(W_c S_{t-1} + W_c X_t\right) \\
f_t &= \sigma\left(W_f S_{t-1} + W_f X_t\right) \\
i_t &= \sigma\left(W_i S_{t-1} + W_i X_t\right) \\
o_t &= \sigma\left(W_o S_{t-1} + W_o X_t\right)
\end{aligned} \tag{7}$$

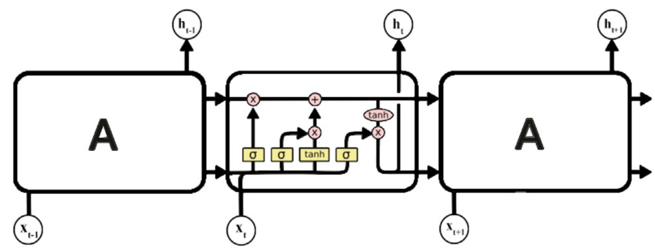

Fig. 8. LSTM neural network architecture.

3.3.3. Elman neural network

An Elman neural network consists of input, hidden, output layer and an extra layer named context layer. Each neuron's output in the hidden layer is fed to the neurons in the context layer. The context layer's output is then used as an additional input signal to hidden layer neurons. The weights on the connection between hidden and context layer are constant and fixed. The weights on the output from the context layer are half of the output range of other neurons [44]. Fig. 9 is helpful in understanding Elman neural network [45]. The network's output can be evaluated using Eq. (8).

$$\begin{aligned}
h_t &= \sigma_h\left(W_h x_t + U_h h_{t-1} + b_h\right) \\
y_t &= \sigma_y\left(W_y h_t + b_y\right)
\end{aligned} \tag{8}$$

Where,
σ_h, σ_y are activation functions.
'W', 'U' are parameter matrix and 'b' is the vectors.

4. Application of neural networks

This section describes numerous studies for software effort estimation using neural networks, higher-order neural networks and deep learning methods by various researchers. The studies aim to illustrate the techniques applied into achieve it and the database used for their experiments is showcased. The performance of various neural networks is compared to study the nature of the problem.

4.1. Neural network in software effort estimation

Vachik S Dave and Dutta [46] compared the regression method and ANN for SEE. This work aimed to prove that FFNN gives better outcome contrasted with other prediction models. The first 13 projects of NASA [10] dataset is considered for training and the rest of 5 projects are considered for testing. Various evaluation

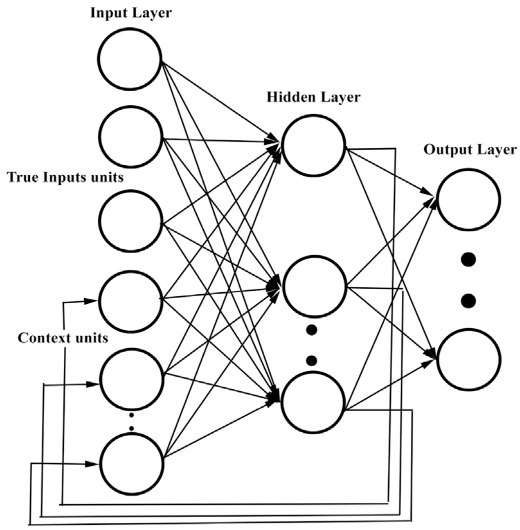

Fig. 9. Elman neural network architecture.

factors such as MMRE, RSD, and MSE are used to evaluate the prediction with COCOMO, NASA dataset. (st_1)

Lopez-Martin, Chavoya and Meda-Campana [47] proposed a FFNN for estimating the duration of the new software project. The used dataset sample is the ISBSG and the evaluation factor is MMRE. They concluded that the accuracy of estimating the duration of software projects by FFNN is better than statistical regression with functional points at a 90% confidence level. (st_2)

To forecast the software development effort, an estimation model based on artificial neural network that makes use of multi-layer feed forward network has been suggested by Madheswaran et al. [48]. The proposed model aims in improving the efficiency of the neural network that correspond to the COCOMO model. The proposed model has been trained on COCOMO dataset using back propagation learning algorithm that iteratively refines a series of training samples. The back-propagation algorithm is also used to analyze the network prediction by comparing it with the actual effort. Further, the proposed model is validated and the results are compared with the COCOMO model. The experimental results show that the proposed model enhances the prediction accuracy of the model and can be efficiently used for predicting the software development effort (st_3).

Venkatachalam A.R [28] proposed a model that makes use of artificial neural network approach for estimating the software cost that is necessary for controlling the cost of the software to make it more competing in the software industry. The proposed model is constructed using back-propagation neural network. The author has used Neuralwane software package for building the neural network. Further, the sensitivity of the model has been tested corresponding to the adjustment in the number of hidden layer and weight adjustment process. Furthermore, the test results reveal that the proposed model efficiently estimates the cost and time required for the development of the software (st_4).

Wittig and Finnic [49] calculated the development effort of a software by capturing the significant attributes by using back propagation algorithm with artificial neural network. Data from commercial 4GL software project is gathered for the study. Evaluation measure used for the prediction is MMRE and from the simulation results, it is evident that, the prediction made by the neural networks are reasonably accurate than other methods. (st_5)

To overcome the limitation of neural networks in software development effort estimation, a model based on Back propagation three-layer perceptron network has been recommended by Idri et al. [50]. The proposed model aims in mapping the neural network to a system that utilizes fuzzy rules. In the proposed model, sigmoid function in the hidden layer and identity function in the output layer is used for estimating the software development effort. Then the proposed model is trained and tested using COCOMO'81 dataset. Further, Benitez's method is utilized for deriving the if–then fuzzy rules form the network. Finally, from the observations it has been concluded that if fuzzy rules can be interpreted easily, then neural networks can also easily interpret the software development effort. (st_6)

For the successful implementation of project, an estimation model that makes uses of two-layer feed forward network has been proposed by Mukherjee et al. [3]. The proposed model makes use of sigmoid hidden neurons and linear output layer for making precise predictions at the initial stage of software development life cycle. Initially, Levenberg Marquardt back propagation algorithm has been used to train the model. Further, the proposed model has been validated using validation dataset and the experimental results reveal that the proposed model obtains better results when compared with Constructive Cost Model (st_7).

To overcome the shortcoming of the 'Black box' model Idri, Mbarki and Abran [51] used RBFN to provide a natural explanation of the cost estimation models. The Jang and Sun method used for mapping is explored to extract fuzzy rules from neural networks. The COCOMO'81 is used for the training as well as for testing of RBFN. It is found that the accuracy of RBFN is mostly based on hidden neurons of the middle layer. (st_8)

Tong-Seng and Mie Mie Thet [52] used GRNN for quality estimation of software products which includes reliability and maintainability of the product using object-oriented metrics. Two investigations are conducted i.e. a number of deformities in a class and the number of lines changed in a class. They used a neural network to detect different features of patterns in hidden layer slabs. But it is proved that the GRNN result is more accurate than the ward neural network. (st_9)

Based on the Personal Software Process (PSP), lines of code are gathered from the developed program. Lopez-Martin [53] used this lines of code as independent variables for the estimation of software development in three models. They used 163 program samples for verification and 80 program samples for validation. With a comparative analysis among GRNN, multiple linear regression, and fuzzy logic and they inferred that the accuracy of prediction using GRNN is statistically similar to fuzzy logic and multiple linear regressions (st_{10}). Intelligent methods related to an artificial neural network such as FFNN, MLP, GRNN, etc. with other information such as dataset, evaluation factors, author, and year of publication are analyzed in Table 2.

4.2. Higher order neural network in software effort estimation

Like classical ANN methods, higher order ANNs have been an interesting solution to the software effort estimation problems. Vinay Kumar et al. [15] suggested a wavelet neural network (WNN) to forecast the development effort of a software project. Two types of WNN are used in their study, one with Morlet function as transfer function and other as Gaussian function where Threshold acceptance training algorithm for wavelet neural network (TAWNN) is compared with MLR, MLP, DENFIS, RBFN, and SVM in terms of evaluation measures such as MMRE, PRED(25), MdMRE. Simulating with Canadian Function (CF) and IBM Data Processing (IBMDPS) datasets, they claimed the efficacy of the proposed method over others. (St_{52})

Benala et al. [92] proposed active learning using PSO with functional link artificial neural network (FLANN) for SEE. The

Table 2
Various neural network methods (FFNN, MLP etc.) and evaluation factors.

Study. No	Author(s)/Year	Contribution	Intelligent method	Datasets	Evaluation factors	Ref
St_{11}	Jorgensen, 1995	Compared baseline, regression, neural network, pattern recognition in calculating effort.	NN with BP algorithm	Random Data	MRE, MMRE, MdMRE, PRED	[54]
St_{12}	Srinivasan and Fisher, 1995	Compared CARTX and BP Learning methods in predicting effort	FFNN with Back Propagation	COCOMO, Kemerer	MRE	[55]
St_{13}	Finnie, Wittig and Desharnais, 1997	Considered CBR, ANN and regression analysis techniques using functional points	BPNN	299 Project Data	MARE, MRE	[9]
St_{14}	Wittig and Finnie, 1997	Appraise of BP in NN for effort estimation	BPNN	Project Data, Desharnais	MARE, MRE	[56]
St_{15}	Miyoung Shin and Goel, 2000	Proposed RBF for SEE	Radial Basis Functions	NASA	MMRE, PRED	[57]
St_{16}	Shepperd and Kadoda, 2001	In this suggested model and compared stepwise regression, rule induction, Case based reasoning and neural nets	SWR, RI, CBR, ANN	Random Data	MMRE	[58]
St_{17}	Jun and Lee, 2001	Propose search model to find relevant cases for neural network	Quasi optimal neural network	112 cases from companies	MMRE, MRE	[10]
St_{18}	Molokken and Jorgensen, 2009	Conduct Machine learning experiment with software effort estimation.	BPNN with Quickprop Algorithm	Raw Data	Pred	[59]
St_{19}	Idri, Abran and Mbarki, 2006	Compared RBFN with C-means and RBFN with APC-III	RBF	COCOMO'81	MMRE, PRED	[13]
St_{20}	Park and Baek, 2008	Compared old NN techniques with neural network	FFNN	148 Project Data	MRE, Average Error	[60]
St_{21}	Idri et al., 2008	Compared RBFN with different variations	RBF	COCOMO'81, Tukutuku	MMRE, PRED	[61]
St_{22}	Iwata et al., 2009	Establish effort prediction model using ANN and compared with MRA Model	ANN	73 Project Data	MAE, VAE, VRE, MRE	[62]
St_{23}	Reddy and Raju, 2009	Compared NN using BP algorithm with COCOMO model	MLFFNN	COCOMO	MRE	[63]
St_{24}	Li, Xie and Goh, 2009	Proposed ANN for NABE and compared with Linear adjusted ABE, CART, ANN and SWR	ANN–NABE	Albrecht, Desharnais, Maxwell, ISBSG	MMRE, MRE, PRED, MdMRE	[64]
St_{25}	Jodpimai, Sophatsathit and Lursinsap, 2010	By incorporating mathematical principal in ANN, software effort prediction is improved	FFNN,	COCOMO81, NASA60, NASA 93, Albrecht, CF, Desharnais	MMRE, PRED, MRE	[65]
St_{26}	Kalichanin-Balich and Lopez-Martin, 2010	Two group of software projects are compared with FFNN and statistical regression	FFNN, statistical regression	132 Project Data	MER, MMER	[23]
St_{27}	Attarzadeh and Siew Hock Ow, 2010	Proposed COCOMO II based on ANN compared to COCOMO	BPNN	COCOMO, Artificial dataset	MMRE, MRE PRED	[66]
St_{28}	Ajitha et al., 2010	Proposed NN using Use case point approach	BPNN	Random Data	MSE	[67]
St_{29}	Kaur et al., 2010	Compared ANN with Different old estimation models	BPNN	NASA	MRE, MMRE, RMSSE	[68]
St_{30}	Zakrani and Idri, 2010	Investigated RBFN with fuzzy c-means clustering and RBFN with K-means	RBF	COCOMO'81	MMRE, PRED, MRE	[69]
St_{31}	Reddy et al., 2010	Proposed ANN based on RB and GR	RB and GR	COCOMO'81	MARE, VARE, PRED, BRE, MMRE	[70]
St_{32}	López-Martín, Chavoya and Meda-Campaña, 2011	Compared results of GRNN and statistical regression	GRNN	156 Project Data	MMER, MER	[71]

(*continued on next page*)

active learning algorithm reduces the database by detecting necessary data from the database and it is processed by PSO–FLANN. The proposed model is tested on NASA93, MAXWELL and COCOMO'81 and shows the improvement in the accuracy of the

Table 2 (*continued*).

Study. No	Author(s)/Year	Contribution	Intelligent method	Datasets	Evaluation factors	Ref
St_{33}	Lopez-Martin, Isaza and Chavoya, 2012	Prediction of effort using GRNN is better or equal obtained by statistical regression	GRNN	ISBSG	MMER, MER	[24]
St_{34}	Nassif, Capretz and Ho, 2012	A novel ANN mode for effort estimation using use case model with use case point.	FFNN	Random Data	MMER, PRED	[72]
St_{35}	Attarzadeh, Mehranzadeh and Barati, 2012	Proposed ANN–COCOMO II and compared with COCOMO II	BPNN	COCOMO I, NASA 93	MRE, MMRE, PRED	[8]
St_{36}	Sarac and Duru, 2013	Combined COCOMO and ANN with K-Means and compared with COCOMO and ANN	BPNN	COCOMO 81	MRE, MMRE	[73]
St_{37}	Manikavelan and Ponnusamy, 2014	FFNN is used to get appropriate estimation compared to old models	FFNN	25KDSI	KDSI	[25]
St_{38}	Das et al., 2014	Two-Fold approach depending on ANN is proposed	BPNN	NASA	MRE, MMRE	[74]
St_{39}	Kamalakannan et al., 2015	Compared GRN with LSR with one and two independent variables	GRNN	163 Project Data	MER, MMER	[75]
St_{40}	Sarno, Sidabutar and Sarwosri, 2015	Compared COCOMO II, Fuzzy and NN models for effort estimation	FFNN with BP	NASA	MRE	[76]
St_{41}	Rao and Kumar, 2015	Proposed GRNN and compared with RBF Kernel, SMO Poly kernel, M5 Linear Regression,	GRNN	COCOMO	MMRE, MdRE	[14]
St_{42}	Amasaki and Lokan, 2016	Investigated windowing approach with NN to improve mean MAE	FFNN with Windowing approach	ISBSG	MAE	[77]
St_{43}	Rijwani and Jain, 2016	Compared COCOMO II and BPNN	MLFNN with BP	COCOMO II	MRE, MMRE, MSE	[78]
St_{44}	Azzeh and Nassif, 2016	Effort estimation is calculated using hybrid model with SVM and RBF and compared with UCP prediction Model	RBF	45 industrial and 65 educational projects	AE, MAE, MRE, MBRE, MIBRE	[79]
St_{45}	Rao, Reddi	Improve effort estimation by using MLP with Artificial Bee Colony	MLPNN, SVM–RBF, ABC	COCOMO	MMRE, MdMRE, MRE	[12]
St_{46}	de A. Araújo, Oliveira and Meira, 2017	Proposed MDELP and compared with different techniques using different datasets	Hybrid MLP	Albrecht, Desharnais, COCOMO, Kemerer, Kotengray, NASA	MMRE, Pred25	[80]
St_{47}	Arora and Mishra, 2018	Compared neural network system with COCOMO model	MLFFNN	COCOMO	MRE	[81]
St_{48}	Kumari and Pushkar, 2018	CS–COCOMO, CS–ANN–COCOMO II are proposed to calculate effort	FFNN	COCOMO 81, COCOMO NASA	MMRE, Pred (25, 30, 40), VAF	[82]
St_{49}	Pandey et. al, 2019	Multiple linear regressions, Multi-Layer Perceptron Neural Network (MLP-NN), Genetic Algorithm (GA) and Naive forecasting approaches are applied to find best technique for mobile app development	MLR, MLP-NN, GA, Naïve forecasting approach	SAMOA	MMRE, MRE, PRED (25),	[83]
St_{50}	Somya Goya et. al, 2020	Proposes non-linear model based on MLP architecture for effort estimation	PRED_MLA, PRED_MLP_FS	Desharnais Project	MRE, MMRE	[84]
St_{51}	Singh et.al, 2020	Linear Regression (LR), Multi-layer perceptron (MLP), Random Forest (RF) implement using WEKA toolkit used for effort estimation	MLA, LR, RF	NASA, Desharnais project	MMRE, PRE	[85]

model and variants of SEE by using evaluation measures such as MMRE, MdMRE, and PRED. (St_{53})

Some of the intelligent methods, datasets, and evaluation factors used in higher-order neural networks are further analyzed in Table 3.

Table 3
Various higher order neural network methods and evaluation factors.

Study. No	Author(s)/Year	Contribution	Intelligent method	Datasets	Evaluation factors	Ref
St_{54}	Benala, Mall, Satchidanada Dehuri, et al., 2012	Suggested UKW/DBSCAN and FLANN and compared with SVR, RBF and CART	FLANN, UKW/DBSCAN	COCOMO 81, NASA93, Desharnias	MMRE, MdMRE, PRED	[38]
St_{55}	Benala, Mall, Satchidananda Dehuri, et al., 2012	Proposed fuzzy c-means clustering combined with FLANN	FLANN	COCOMO 81, NASA 93, Maxwell	MMRE, MdMRE, PRED	[16]
St_{56}	Benala et al., 2013	Proposed PSO optimized FLANN for effort estimation	PSO–FLANN	COCOMO 81, NASA 93, Maxwell	MMRE, MdMRE, PRED	[86]
St_{57}	Kaushik, Soni and Soni, 2016	Proposed FLANN with intuitionistic fuzzy c-means clustering	IFCM–FLANN	COCOMO 81, NASA 93, Maxwell, China Dataset	MMRE, MdMRE, PRED	[37]
St_{58}	Zahid Hussain Wani and S.M.K. Quadri, 2016	Proposed Artificial Bee Colony trained FLANN for effort estimation	ABC-based FLANN model	NASA 93, COCOMO 81 and COCOMO_SDR	MRE, MMRE and MdMRE	[87]
St_{59}	Lov Kumar and SantanuKu. Rath, 2016	Proposed hybrid approach of FLANN with genetic algorithm (GA), particle swarm optimization (PSO) and clonal selection algorithm (CSA)	FLANN–Genetic, FLANN–CSA	UIMS and QUES	MAE, MMRE, SEM, e, ê	[88]
St_{60}	Tirimula Rao Benalaet al., 2013	Proposed a model based on improved particle swarm optimization (ISO) with FLANN	ISO, FLANN	Cocomo81, Nasa93, Maxwell [8]	MMRE, MdMRE, PRED	[89]
St_{61}	B. Tirimula Rao et al. 2006	FLANN model proposed for validating dataset for software cost estimation	FLANN, CFLANN, LFLANN, PFLANN, COCOMO	standard 60 NASA projects dataset	Root Mean Square Error	[90]
St_{62}	Kaushik et. al, 2019	Proposes hybrid of Wavelet neural network and metaheuristic algorithm for estimation software development effort	WNN–Firefly algorithm, Bat algorithm	COCOMO, NASA93, Maxwell, China	MMRE, MdMRE, PRED (25)	[91]

4.3. Deep learning neural network in software effort estimation

Spiking neural network can resolve complex information processing tasks and is superior to traditional neural network models. The temporal data produced by the utilization of spike trains can be encoded, permitting quick decoding and multiplexing data. Venkataiah, Mohanty and Nagaratna [17] used SNN to ameliorate the accuracy of cost estimation of a software. IBMDPS, ISBSG-10 and China datasets are used for the experiment and performance validation is done using RMSE, MMRE. Simulation results of their experiment proved the effectiveness of the proposed method. (St_{63})

Bilgaiyan, Mishra and Das [93] proposed agile software development (ASD), which is more effective than traditional methods of software development. To solve the effort estimation in ASD, Elman neural network and FFNN with back propagation are applied. These two neural network models are applied on a dataset consisting of 21 project information built on ASD from 6 software companies and the performance validation is done using MMRE, MSE, PRED(x). (St_{64})

Different Intelligent methods, datasets, and other information used in some DLNN are analyzed in Table 4.

4.4. Other neural networks used in software effort estimation

Apart from the above-mentioned models, some more neural network models are also used for SEE. Nassif, Capretz and Ho [97] proposed Cascade Correlation Neural Network (CCNN) for SEE using use case diagrams. Software size, productivity and project complexity are used as the input parameters. This model is evaluated on 214, 26 industrial and educational projects respectively using metrics such as MMER and PRED. The CCNN model and a multiple linear regression model are trained with same inputs and evaluated against each other. The result concluded that the proposed CCNN model performs better than multiple linear regression models and could be used for estimating the software effort. (St_{71})

Hui Zeng and Rine [98] constructed NN model for effort estimation using back propagation and Self-Organizing Neural Network (SONN). SONN algorithm has algebraic representation which helps to identify the important factors need for analyzing the software effort estimation. They investigated the proposed model using COCOMO dataset on various evaluation measure such as PRED and found to be better performed model. (St_{72})

4.5. Comparative study on neural networks in software effort estimation

This section indicates the comparative study on different evaluation metrics used for most of the ANN based experiments. Ghose, Bhatnagar and Bhattacharjee [99] used various ANN models such as FFBBPNN, Elman BPNN, cascaded FFBPNN, Generalized Regression NN, and Recurrent NN to illustrate the performance

Table 4
Various deep learning neural network methods and evaluation factors.

Study. No	Author(s)	Contribution	Intelligent method	Datasets	Evaluation factors	Ref
St_{65}	Praynlin and Latha, 2013	Compared BPNN and Elman Neural Network	ELMAN, BPNN	NASA	MMRE, PRED (25), RMSE, Mean, Std. Dev	[19]
St_{66}	Choetkiertikul et al., 2019	Proposed Combined model of two deep learning architectures such as LSTM and RHM	LSTM, RHN	PROJECT DATA	MAE, MdAE	[18]
St_{67}	Edinson and Muthuraj, 2018	Effort is calculated using ANFIS, FCM, SC and compared with Elman neural network.	ELMAN, ANFIS based FCM, SC	COCOMO, Deshernais, Maxwell and IKH	MRE, MMRE, PRED(25), RMSE	[44]
St_{68}	Venkataiah et. al, 2019	Proposed spiking neural network to improve cost estimation	Spiking neural network	IBM, ISGSB, CHINA	MMRE, RMSE	[94]
St_{69}	Ming Qin et. al, 2019	Proposed Deep Neural network for software cost estimation	BiLSTM-CRF	52 project data	–	[95]
St_{70}	V. Resmi et. al, 2019	Proposed Output layer self-connection recurrent neural networks (OLSRNN) with kernel fuzzy c-means clustering (KFCM) to improve software cost estimation	KFCM–OLSRNN	5 publicly available dataset	MMRE, MdMRE, PRED(25)	[96]

comparison on estimation of software effort. For experimental investigation, they used 31 inputs of standard dataset. By using various estimation measures such as MRE, MMRE, MdMRE, PRED and BRE, they showed GRNN gave better result as compared to other NN. (St_{73})

A brief comparative analysis on various intelligent methods, datasets, and other related metrics are shown in Table 5.

5. Critical analysis and investigation

This section gives an overview of the results obtained from the study of all the articles mentioned above. A brief analysis has been conducted with a suitable questionnaire method to illustrate the effectiveness of all the considered methods. The result is represented using tables, graphs answering the question and showing the evidence of all the methods.

5.1. | Q1: Which dataset is used widely in predicting SEE?

Table 6 shows datasets used in 80 studies. From the table it is evident that there are 21 datasets utilized for effort estimation. Among them, COCOMO dataset is the most utilized dataset in 33 (41.25% of all) no. of various investigations. Next is NASA dataset used in 19 (23.75% of all) different studies, and Project data used in 17 (21.25% of all) studies. ISBSG dataset, Desharnais and Maxwell dataset have been used in 8 (10% of all) studies, followed by Random dataset used in 7 (8.75% of all) studies. Next, are the Albrecht, IBMDPS and China dataset used in 4 (5% of all) studies and Kemerer dataset used in 3 (3.75% of all) studies. Lastly CF dataset, artificial dataset, 25KDSI dataset, Kotengray dataset, Canadian function dataset, Tukutuku dataset, IKH dataset, SAMOA, UMIS, QUES and Zia, CD1, CD2 each used in 1 (1.25% of all) study. The graphical portrayal of this investigation is illustrated in Fig. 10.

From all the studies, it is noted that in St_{46}, six datasets are used in the experiment for a better comparison of the result. The datasets used are Albrecht, Desharnais, COCOMO, Kemerer, Kotengray and NASA. Similarly, St_{25} used 5 datasets which are COCOMO, NASA, Albrecht, CF, Desharnais datasets. St_{24}, St_{57}, St_{62} and St_{67} have used 4 datasets. Datasets used in St_{24} are Albrecht, Desharnais, Maxwell, ISBSG dataset and in St_{57} and St_{62} are COCOMO 81, NASA 93, Maxwell and China dataset. Dataset used in St_{67} are COCOMO, Desharnais, Maxwell and IKH.

5.2. | Q2: Which technique is mostly used in predicting SEE?

Prediction of software effort estimation has been done using various ANN techniques. Some studies include multiple techniques for predicting software effort estimation which is later compared to find the best technique among them.

From the various studies of intelligent methods mentioned in Table 7, it is noted that FFNN (including MLP, BP) is the most used technique mentioned in 50 (62.5% of all) studies followed by RBFN used in 15 (18.75% of all) studies. Moreover, the linear regression technique is used in 12 (15% of all) studies. GRNN is used in 9 (11.25% of all) studies and FLANN technique is used in 7 (8.75% of all) studies. COCOMO and Elman neural networks are used in 4 (5% of all) studies. CARTX and COCOMO II both methods are used 3 (3.75% of all) studies. CBR, Use-case, ABC, WNN and Spiking neural network techniques are each used in 2 (2.5% of all) studies. The techniques such as Rule induction, Quasi-optimal neural network, MRA model, ABE, SWR, MDELP, SVR, LSTM, RHM, ANFIS, SONN, RNN, SLR and GMDH each are used in 1 (1.25% of all) study. These are shown clearly in Fig. 11.

5.3. | Q3: Which accuracy measure is used mostly in SEE?

There are a many accuracy measures utilized in SEE for the performance analysis. The accuracy measures used in the literature are shown in Table 8, and a comparative analysis is displayed in Fig. 12. Among them MMRE is mostly used, included in 46 (57.5% of all) studies. The second most used accuracy measure is PRED used in 34 (42.5% of all) studies, followed by MRE used in 29 (36.25% of all) studies. MdMRE is used in 15 (18.75% of all) studies. MMER used in 8 (10% of all) studies and RMSE used in 6 (7.5% of all). MSE, MAE and MER used in 5 (6.25% of all) studies. MARE used in 4 (5% of all) studies. RSD, BRE, VAF, R2 and R are used in 2 (2.5% of all) studies. Average error, VAE, VRE, RMSSE, VARE, KDSI, MdRE, AE, MBRE, MIBRE, Mean, Std. Dev, MdAE, MAPE, fried statistical test, SEM, e, and ê are used in 1 (1.25% of all) study.

Table 5
Various neural network methods used for comparison of evaluation factors.

Study. No	Author(s)	Intelligent method	NN Parameters	Datasets	Evaluation factors	Results	Ref
St_{74}	Heiat, 2002	MLP, RBF	Specified threshold 0.01	Kemerer, IBM	MAPE	MLP method provide accurate results	[100]
St_{75}	Vachik S. Dave and Dutta, 2011	Regression Analysis, FFNN, RBFN	FFNN-learning rate: 0.8 Training iterations: 2000 RBFN- Training iterations: 1000	Cocomonasa_v1	MRE, MMRE, Modified MMRE, RSD	RBFN gives better results than FFNN	[101]
St_{76}	López-Martín, 2015	MLP, GRNN, RBFN, SLR	MLP- New projects 15 neurons in one hidden layer, Enhanced projects 35 neurons in 2 hidden layer (best accuracy) RBFN- Spread 1.1, 5 for new and enhanced projects GRNN- Spread 0.15, 4 for new and enhanced projects	ISBSG	RMSE, MMRE, MMER, MBRE, MIBRE	RBFN gives more accurate results when FP are used as independent variables	[102]
St_{77}	López-Martín and Abran, 2015	MLR, MLP, RBFN	MLP- Input layer 2 neurons, Hidden layer 25 neurons, Output layer 1 neuron RBFNN- bias b=0.8326/spread (best result with spread=25)	ISBSG	Pred(l), Friedman statistical test	MLP and RBFN are statistical better than MLR Model	[103]
St_{78}	Sheta, Rine and KASSAYMEH, 2015	FFNN, RBF	FFNN- Hidden layer 3 neuron-tan sigmoid transfer function Output layer one neuron-linear sigmoid function, Epochs 300 RBF- Spread- 3 Hidden nodes 18	Albrecht	VAF, MMRE	High quality competitive results obtained using ANN Models	[104]
St_{79}	Panda, Satapathy and Rath, 2015	GRNN, GMDH Polynomial NN, CCNN	GRNN- Kernel function Reciprocal, sigma 0.001–0.5 HPNN- No of layers 5(max), Neuron in each layer 10 (fixed), Control data 20% CCNN- Neuron 1–8 Candidate neuron 8, Max steps without improvement 10, Over fitting protection control 3-fold cross validation	Twenty-one projects	MSE, R^2, MMRE, PRED	Cascade Network model is better than models developed so far	[105]
St_{80}	Kaushik et. al, 2020	RBFN, FLANN with Whale Optimization Algorithm (WOA)	–	Zia dataset, Company Dataset −1 (CD1) and Company Dataset-2 (CD2)	MMRE, MdMRE, PRED(0.25)	ANN models perform better when combined with metaheuristic technique	[106]

5.4. | *Q4: Which combination techniques are mostly used in predicting SEE?*

Some techniques are combined to form a hybrid model for a more accurate prediction of effort estimation. The hybrid models used for software effort estimation are shown in Table 9. A mostly used combined technique is PSO–FLANN mentioned in 3 (3.75% of all) studies. SVM–RBF, FCM with FLANN and CCNN–UC each mentioned in 2 (2.5% of all) studies. The combination techniques ANN–NABE, ANN–COCOMO, ANN–RB, ANN–GR, ANN–UC, ANN–COCOMO II, CS–ANN–COCOMO II, COCOMO–ANN, MLP–ABC, CS–COCOMO, UKW/DBSCAN, BiLSTM–CRF, KFCM–OLSRNN, MLP–NCA, FLANN–ABC, FLANN–GA, FLANN–CSA, FLANN–ISO, WNN–Firefly and BAT algorithm, FLANN–WOA and RBFN–WOA each are used in 1 (1.25% of all) study. The graphical representation is shown in Fig. 13. From the analysis, it is evident that among the other NN approaches, FLANN has been the most favorite choice for software effort estimation.

5.5. | *Q5: Which publications are dominant in the area of SEE?*

In our study, we considered IEEE Xplore, Springer, Inderscience, Elsevier and Wiley. From Table 10 and Fig. 14 are used to analyze the publication ratio of software effort estimation. We observe that IEEE Xplore has published most of the studies i.e., 31 (38.75%) studies. Moreover, the Springer Series has published 14 (17.5%) studies. Journal of Systems and Software (JSS) has published 4 (5%) studies and Expert Systems with Applications (ESA) has published 3 (3.75%) studies. Information and Software Technology (IST), Empirical Software Engineering (ESE), Procedia Computer Science (PCS), Applied Soft Computing (AS) and International Journal of Information Technology (IJIT) each have published 2 (2.5%) studies. International Journal of Computer Application (IJCA), Journal of information systems (JIS), Neural Computing and Applications (NCA), International Journal of Recent Trends in Engineering (IJRTE), International Journal of Computer Theory and Engineering (IJCTE), International Review

Table 6
Frequency of datasets used in various studies.

Dataset	COCOMO	NASA	ISBSG	Project Data	Random Data	Kemerer	Desharnais	Albrecht	Maxwell	CF	Artificial	25KDSI	Kotengray	Canadian function	IBMDPS	China	Tukutuku	IKH	SAMOA	UMIS, QUES	Zia, CD1, CD2
St_1	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_2	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_3	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_4	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_5	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_6	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_7	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_8	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_9	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{10}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{11}	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{12}	1	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{13}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{14}	–	–	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{15}	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{16}	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{17}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{18}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{19}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{20}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{21}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–
St_{22}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{23}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{24}	–	–	1	–	–	–	1	1	1	–	–	–	–	–	–	–	–	–	–	–	–
St_{25}	1	1	–	–	–	–	1	1	–	1	–	–	–	–	–	–	–	–	–	–	–
St_{26}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{27}	1	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{28}	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{29}	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{30}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{31}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{32}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{33}	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{34}	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{35}	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{36}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{37}	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–
St_{38}	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{39}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{40}	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{41}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{42}	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{43}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{44}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{45}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{46}	1	1	–	–	–	1	1	1	–	–	–	–	1	–	–	–	–	–	–	–	–
St_{47}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{48}	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{49}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–
St_{50}	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{51}	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{52}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	1	–	–	–	–	–	–
St_{53}	1	1	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–
St_{54}	1	1	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{55}	1	1	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–
St_{56}	1	1	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–
St_{57}	1	1	–	–	–	–	–	–	1	–	–	–	–	–	–	1	–	–	–	–	–
St_{58}	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{59}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–
St_{60}	1	1	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–
St_{61}	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{62}	1	1	–	–	–	–	–	–	1	–	–	–	–	–	–	1	–	–	–	–	–
St_{63}	–	–	1	–	–	–	–	–	–	–	–	–	–	–	1	1	–	–	–	–	–
St_{64}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{65}	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{66}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{67}	1	–	–	–	–	–	1	–	1	–	–	–	–	–	–	–	–	1	–	–	–
St_{68}	–	–	1	–	–	–	–	–	–	–	–	–	–	–	1	1	–	–	–	–	–
St_{69}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{70}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

(continued on next page)

Table 6 (*continued*).

Dataset	COCOMO	NASA	ISBSG	Project Data	Random Data	Kemerer	Desharnais	Albrecht	Maxwell	CF	Artificial	25KDSI	Kotengray	Canadian function	IBMDPS	China	Tukutuku	IKH	SAMOA	UMIS, QUES	Zia, CD1, CD2
St_{71}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{72}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{73}	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{74}	–	–	–	–	–	1	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–
St_{75}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{76}	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{77}	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{78}	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{79}	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{80}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1

Table 7
Intelligent Technique used in various studies.

Intelligent method	FFNN, MLP, BP	Regression	COCOMO	RBFN	GRNN	CARTX	CBR	Rule Induction	QONN	MRA Model	ABE	SWR	Usecase	COCOMO II	ABC	MDELP	WNN	FLANN	SVR	Spiking NN	Elman NN	LSTM	RHM	ANFIS	SONN	RNN	SLR	GMDH
St_1	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_2	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_3	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_4	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_5	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_6	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_7	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_8	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_9	1	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{10}	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{11}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{12}	1	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{13}	1	1	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{14}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{15}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{16}	1	1	–	–	–	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{17}	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{18}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{19}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{20}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{21}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{22}	1	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{23}	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{24}	1	–	–	–	–	1	–	–	–	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{25}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{26}	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{27}	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{28}	1	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{29}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{30}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{31}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{32}	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{33}	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{34}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{35}	1	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{36}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{37}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{38}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{39}	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{40}	1	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{41}	–	1	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{42}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{43}	1	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{44}	–	–	–	1	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{45}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{46}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–
St_{47}	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{48}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{49}	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{50}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{51}	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{52}	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–
St_{53}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{54}	–	–	–	1	–	1	–	–	–	–	–	–	–	–	–	–	–	1	1	–	–	–	–	–	–	–	–	–
St_{55}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{56}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{57}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{58}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{59}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{60}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{61}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{62}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	1	–	–	–	–	–	–	–	–	–	–	–
St_{63}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–
St_{64}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_{65}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_{66}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	1	–	–	–	–	–
St_{67}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	1	–	–	–	–
St_{68}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–
St_{69}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{70}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{71}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{72}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–
St_{73}	1	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	1	–	–
St_{74}	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{75}	1	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{76}	1	–	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–
St_{77}	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{78}	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{79}	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1
St_{80}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–

Table 8
Evaluation Factors used in various studies.

Evaluation Factor	MMRE	MRE	MMER	PRED	RSD	MSE	MdMRE	MARE	Average Error	MAE	VAE	VRE	MER	RMSSE	VARE	BRE	KDSI	MdRE	AE	MBRE	MIBRE	VAF	RMSE	Mean, Std Dev	MdAE	MAPE	Friedman Statistical test	R2, R	SEM	e, ê
St_{1}	1	–	–	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{2}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{3}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{4}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{5}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{6}	1	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{7}	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{8}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{9}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{10}	–	–	1	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{11}	1	1	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{12}	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{13}	–	1	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{14}	–	1	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{15}	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{16}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{17}	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{18}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{19}	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{20}	–	1	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{21}	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{22}	–	1	–	–	–	–	–	–	–	1	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{23}	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{24}	1	1	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{25}	1	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{26}	–	–	1	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{27}	1	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{28}	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{29}	1	1	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{30}	1	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{31}	1	–	–	1	–	–	–	1	–	–	–	–	–	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{32}	–	–	1	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{33}	–	–	1	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{34}	–	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{35}	1	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{36}	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{37}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{38}	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{39}	–	–	1	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{40}	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{41}	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–
St_{42}	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{43}	1	1	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{44}	–	1	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	1	1	1	–	–	–	–	–	–	–	–	–
St_{45}	1	1	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{46}	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{47}	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{48}	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–
St_{49}	1	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{50}	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–
St_{51}	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{52}	1	–	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{53}	1	–	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{54}	1	–	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{55}	1	–	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{56}	1	–	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{57}	1	–	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{58}	1	1	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{59}	1	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	1
St_{60}	1	–	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{61}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_{62}	1	–	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{63}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_{64}	1	–	–	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{65}	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	1	–	–	–	–	–	–
St_{66}	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–
St_{67}	1	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_{68}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_{69}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{70}	1	–	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{71}	–	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{72}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{73}	1	1	–	1	–	–	1	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{74}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–
St_{75}	1	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{76}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_{77}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–
St_{78}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–
St_{79}	1	–	–	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–
St_{80}	1	–	–	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Table 9
Hybrid Techniques used in various studies.

Combined Method	ANN–NABE	ANN–COCOMO	ANN–RB	ANN–GR	ANN–UC	ANN-COCOMO II	COCOMO–ANN	SVM–RBF	MLP–ABC	CS–COCOMO	CS-ANN-COCOMO II	PSO–FLANN	UKW/DBSCAN	FCM with FLANN	CCNN–UC	BiLSTM–CRF	KFCM–OLSRNN	MLP–NCA	FLANN–ABC	FLANN–GA	FLANN–CSA	FLANN–ISO	WNN–Firefly & BAT	FLANN–WOA	RBFN–WOA
St_1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_2	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_3	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_4	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_5	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_7	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_8	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_9	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{10}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{11}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{12}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{13}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{14}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{15}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{16}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{17}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{18}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{19}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{20}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{21}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{22}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{23}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{24}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{25}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{26}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{27}	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{28}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{29}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{30}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{31}	–	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{32}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{33}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{34}	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{35}	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{36}	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{37}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{38}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{39}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{40}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{41}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{42}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{43}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{44}	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{45}	–	–	–	–	–	–	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{46}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{47}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{48}	–	–	–	–	–	–	–	–	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{49}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{50}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_{51}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{52}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{53}	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{54}	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–
St_{55}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–
St_{56}	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{57}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–
St_{58}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–
St_{59}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	1	–	–	–	–
St_{60}	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	1	–	–	–
St_{61}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{62}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–
St_{63}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{64}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{65}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{66}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{67}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{68}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{69}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–
St_{70}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–
St_{71}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{72}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

(*continued on next page*)

Table 9 (*continued*).

Combined Method	ANN–NABE	ANN–COCOMO	ANN–RB	ANN–GR	ANN–UC	ANN-COCOMO II	COCOMO–ANN	SVM–RBF	MLP–ABC	CS–COCOMO	CS-ANN-COCOMO II	PSO–FLANN	UKW/DBSCAN	FCM with FLANN	CCNN–UC	BiLSTM–CRF	KFCM–OLSRNN	MLP–NCA	FLANN–ABC	FLANN–GA	FLANN–CSA	FLANN–ISO	WNN–Firefly & BAT	FLANN–WOA	RBFN–WOA
St_{73}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{74}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{75}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{76}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{77}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{78}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{79}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{80}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	1

Table 10
Journal/Conference publications in various studies.

Journal/Conference	IEEE	IJCA	AJIS	NCA	JSS	IST	ESA	IJRTE	ESE	IJCTE	IRCS	JC	MCPR	Springer	PCS	AS	MT	IJSAEM	IJIT	IJITCS	IJLCF	IAJIT	IJNGS	JCSC	IJITP	WPC	IJCSNS
St_{1}	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{2}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{3}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{4}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{5}	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{6}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{7}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{8}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{9}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{10}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{11}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{12}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{13}	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{14}	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{15}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{16}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{17}	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{18}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{19}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{20}	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{21}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{22}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{23}	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{24}	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{25}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{26}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{27}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{28}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{29}	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{30}	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{31}	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{32}	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{33}	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{34}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{35}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{36}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{37}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{38}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{39}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{40}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{41}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{42}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{43}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–
St_{44}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–
St_{45}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{46}	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{47}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{48}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{49}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–
St_{50}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{51}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{52}	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{53}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{54}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{55}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{56}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{57}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–
St_{58}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{59}	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{60}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{61}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1
St_{62}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–
St_{63}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{64}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–
St_{65}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_{66}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–
St_{67}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–
St_{68}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{69}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{70}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–
St_{71}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{72}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{73}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{74}	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{75}	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{76}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–
St_{77}	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{78}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–
St_{79}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–
St_{80}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–

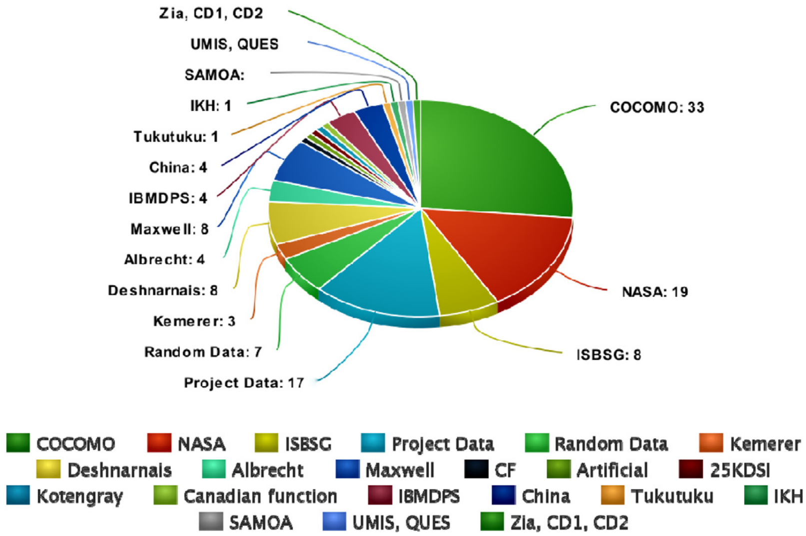

Fig. 10. Datasets used in various studies.

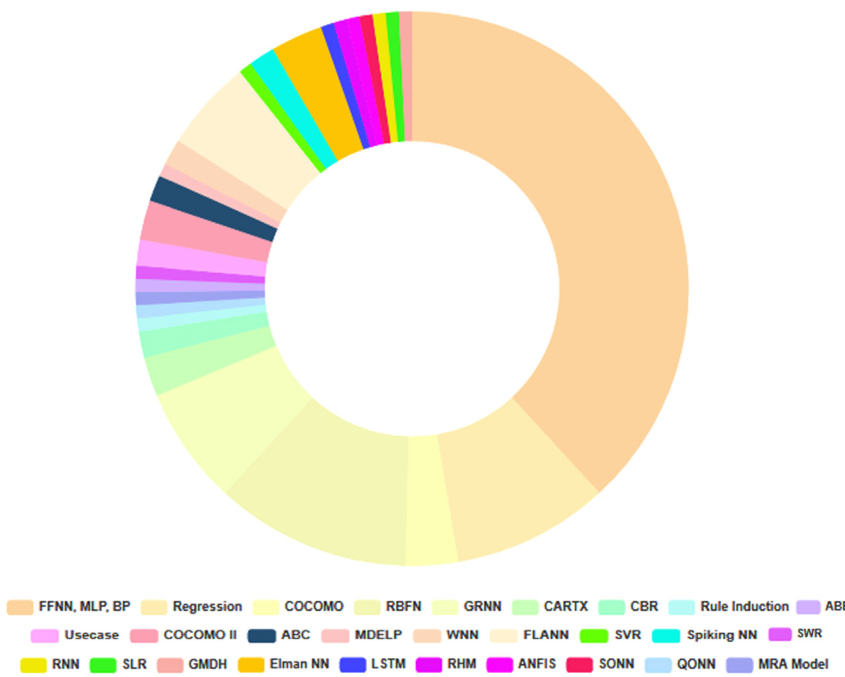

Fig. 11. Intelligent methods in various studies.

on Computers and Software (IRCS), Journal of computing (JC), Mexican Conference on Pattern Recognition (MCPR), Empirical Software Engineering, Microsystem Technologies (MT), International Journal of System Assurance Engineering and Management (IJSAEM), I.J. Information Technology and Computer Science (IJITCS), Journal of latex class files (JLCF), The International Arab Journal of Information Technology (IJAIT), International Journal of Next-Generation Computing (IJNGS), Journal of Circuits, Systems and Computers (JCSC), International Journal of Information Technology Project Management (IJITP), Wireless Personal Communication (WPC) and International Journal of Computer Science and Network Security (IJCSNS) has published 1(1.25%) study.

5.6. | Q6: Which year the substantial research is done in SEE?

For SLR we considered the article from the year 2000 to 2019 presented in Table 11. About 50 (62.5%) no. of studies are published in 2011–2020, 23 (28.75%) studies are published in 2001–2010 and 7 (8.75%) studies published in 1993–2000. The year with most published research works is 2014. There are 9 (11.25%) studies published in 2014, followed by 2010 with 8 (10%) studies and then 2013 and 2015 with 6 (7.5%) studies published. 5 (6.25%) studies are published in 2011, 2012 as well as 2019. 2009, 2016 and 2020 each have 4 (5%) studies. 3 (3.75%) studies are published in the year 2001, 2017 and 2018. In the years 1995, 1997, 2002, 2006 and 2008 2 (2.5%) of studies are published. Only 1 (1.25%) study is published in the years 1993, 1994, 2000, 2003 and 2004. The investigation is appeared in Table 10 and its diagrammatic portrayal is appeared in Fig. 15.

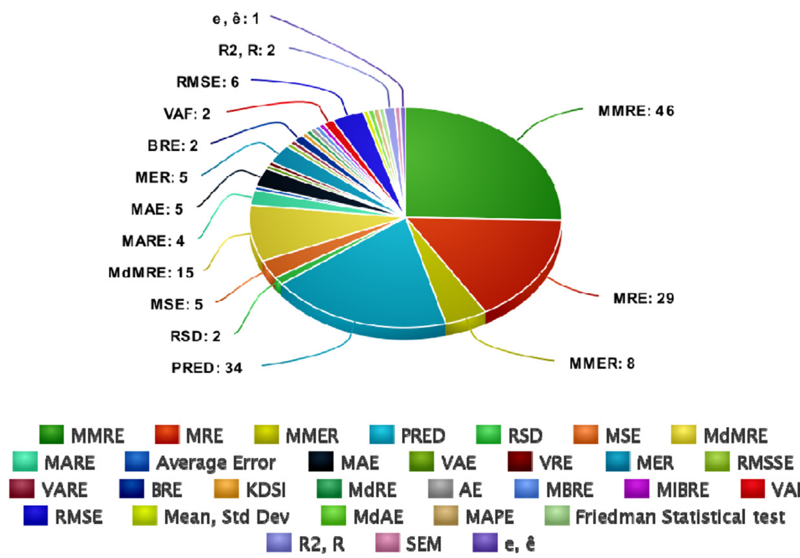

Fig. 12. Evaluation factors in various studies.

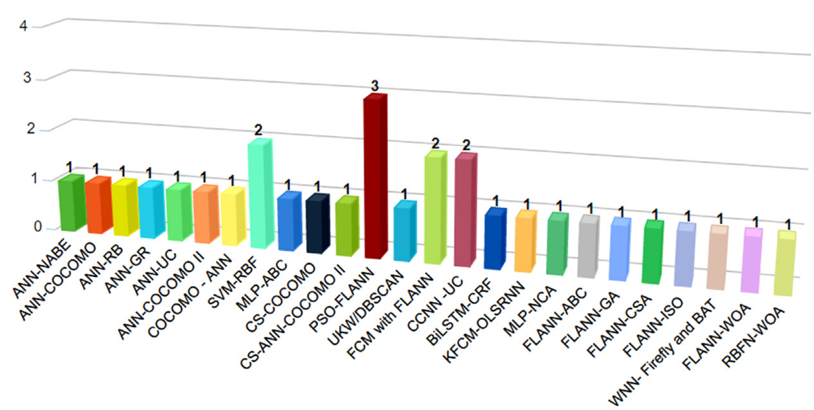

Fig. 13. Hybrid techniques in various studies.

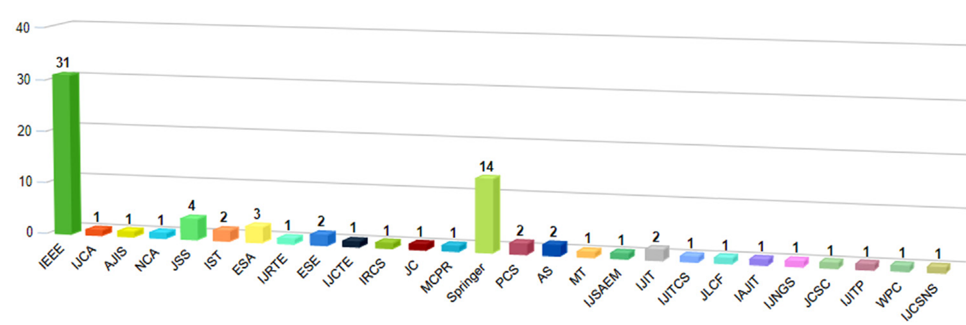

Fig. 14. Journal/conferences in various studies.

5.7. | Q7: Which type of software effort is mostly considered?

SEE incorporated the estimation of development effort, size effort, cost effort, time effort, and maintenances effort. Table 12. shows that, the development effort has been estimated in most of the studies i.e., in 53 (65.4%) studies, cost effort is estimated in 24 (29.6%) studies. Maintenance effort is estimated in 2 (2.4%) studies and size and time effort has been in 1 (1.23%) study. It is noted that many of the researchers are working in development of effort compared to remaining domain of software engineering. Fig. 16 shows the detailed analysis of various effort types considered in the focused research.

5.8. | Q8: Which Combination (Dataset vs Intelligent method) is used mostly used in predicting SEE?

This section analyzes the combination of various methods and the datasets used for the research in the prediction of software effort estimation. The most popular combination is FFNN, MLP, BP method along with COCOMO dataset, found in 20 studies. Next is FFNN, MLP, BP method using NASA dataset in 9 studies

Table 11
Year wise publication frequency of software effort estimation using ANN.

Year	1993	1994	1995	1997	2000	2001	2002	2003	2004	2006	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
St_1	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–
St_2	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_3	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–
St_4	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_5	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_6	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_7	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–
St_8	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_9	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{10}	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{11}	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{12}	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{13}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{14}	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{15}	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{16}	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{17}	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{18}	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{19}	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{20}	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{21}	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–
St_{22}	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–
St_{23}	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–
St_{24}	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–
St_{25}	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{26}	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{27}	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{28}	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{29}	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{30}	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{31}	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
St_{32}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–
St_{33}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–
St_{34}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–
St_{35}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–
St_{36}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_{37}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–
St_{38}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–
St_{39}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–
St_{40}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–
St_{41}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–
St_{42}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–
St_{43}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–
St_{44}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–
St_{45}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–
St_{46}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–
St_{47}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–
St_{48}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–
St_{49}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1
St_{50}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1
St_{51}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1
St_{52}	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–
St_{53}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–
St_{54}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–
St_{55}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–
St_{56}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_{57}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–
St_{58}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–
St_{59}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_{60}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_{61}	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–
St_{62}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–
St_{63}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–
St_{64}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–
St_{65}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–
St_{66}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–
St_{67}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–
St_{68}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–
St_{69}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–
St_{70}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–
St_{71}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–
St_{72}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–
St_{73}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–
St_{74}	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
St_{75}	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–
St_{76}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–
St_{77}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–
St_{78}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–
St_{79}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–
St_{80}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1

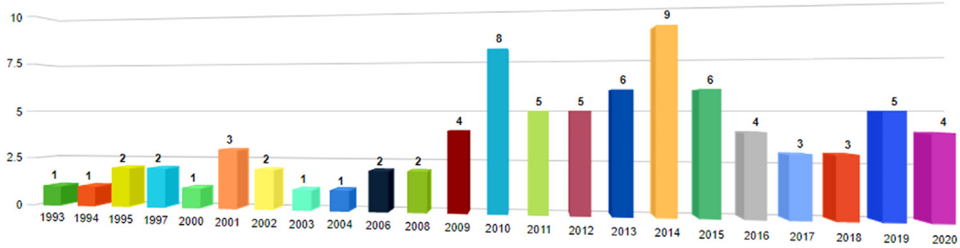

Fig. 15. Year of publication in various studies.

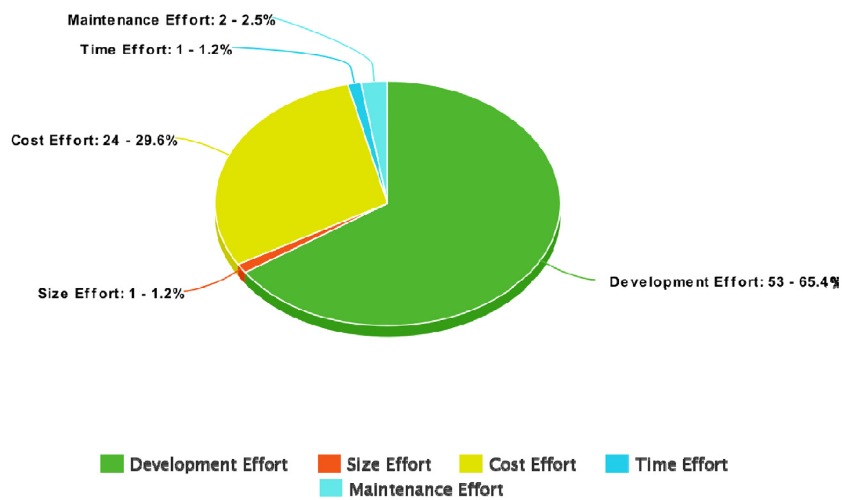

Fig. 16. Effort type in various studies.

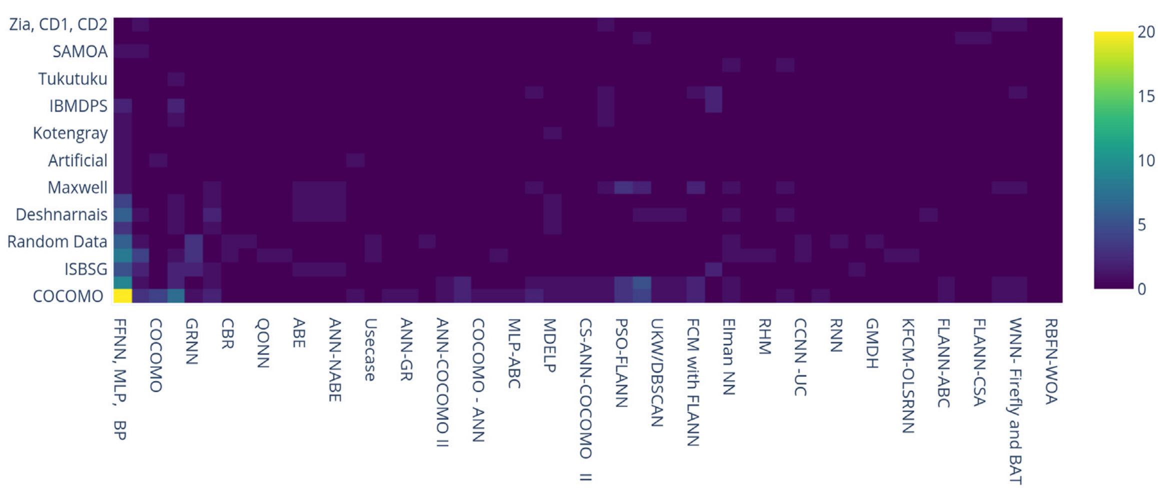

Fig. 17. Statistics of dataset and intelligent method in software effort estimation.

and project dataset in 8 studies. RBFN method using COCOMO dataset is found in 7 studies. FFNN, MLP, BP methods has used Random data and Desharnais dataset in 6 studies and ISBSG dataset in 5 studies. FLANN method has used NASA dataset in 5 studies. Next is FFNN, MLP, BP method with Albrecht dataset, Regression method with Project data, COCOMO method with COCOMO dataset and FLANN method with COCOMO dataset used in 4 studies. FFNN, MLP, BP method with Kemerer method, Regression method with COCOMO dataset, GRNN method using Project data and Random data, PSO–FLANN using COCOMO, NASA and Maxwell dataset has been found in 3 studies. The analysis of dataset and intelligent method is shown in Table 13 and graphically analyzed in Fig. 17.

5.9. | Q9: Which intelligent method's performance is better on various datasets?

Various datasets are used on the neural network models to estimate the effort and the performance may differ for each dataset. In Table 14, the performance of various methods with respect to datasets used for effort estimation is illustrated. Since MMRE is the most commonly used evaluation criteria in most of the studies, the performance of the intelligent methods is shown in terms of MMRE. Using COCOMO as the dataset, GRNN

Table 12
Different effort types of various studies.

Estimation type	Development effort	Size effort	Cost effort	Time effort	Maintenance effort
St_1	1	–	–	–	–
St_2	1	–	–	–	–
St_3	1	–	–	–	–
St_4	–	–	1	–	–
St_5	1	–	–	–	–
St_6	–	–	1	–	–
St_7	1	–	–	–	–
St_8	–	–	1	–	–
St_9	1	–	–	–	–
St_{10}	1	–	–	–	–
St_{11}	–	–	–	–	1
St_{12}	1	–	–	–	–
St_{13}	1	–	–	–	–
St_{14}	1	–	–	–	–
St_{15}	1	–	–	–	–
St_{16}	1	–	–	–	–
St_{17}	1	–	–	–	–
St_{18}	1	–	–	–	–
St_{19}	–	–	1	–	–
St_{20}	1	–	–	–	–
St_{21}	–	–	1	–	–
St_{22}	1	–	–	–	–
St_{23}	1	–	–	–	–
St_{24}	–	–	1	–	–
St_{25}	1	–	–	–	–
St_{26}	1	–	–	–	–
St_{27}	–	–	1	–	–
St_{28}	–	1	–	–	–
St_{29}	1	–	–	–	–
St_{30}	1	–	–	–	–
St_{31}	1	–	–	–	–
St_{32}	1	–	–	–	–
St_{33}	1	–	–	–	–
St_{34}	–	–	1	–	–
St_{35}	1	–	–	–	–
St_{36}	1	–	–	–	–
St_{37}	–	–	1	–	–
St_{38}	1	–	–	–	–
St_{39}	1	–	–	–	–
St_{40}	–	–	1	–	–
St_{41}	1	–	–	–	–
St_{42}	1	–	–	–	–
St_{43}	1	–	–	–	–
St_{44}	1	–	–	–	–
St_{45}	1	–	–	–	–
St_{46}	1	–	–	–	–
St_{47}	–	–	1	–	–
St_{48}	1	–	–	–	–
St_{49}	1	–	–	–	–
St_{50}	1	–	–	–	–
St_{51}	1	–	–	–	–
St_{52}	1	–	1	–	–
St_{53}	1	–	–	–	–
St_{54}	–	–	1	–	–
St_{55}	–	–	1	–	–
St_{56}	–	–	1	–	–
St_{57}	–	–	1	–	–
St_{58}	–	–	1	–	–
St_{59}	–	–	–	–	1
St_{60}	1	–	–	–	–
St_{61}	–	–	1	–	–
St_{62}	1	–	–	–	–
St_{63}	–	–	1	–	–
St_{64}	1	–	–	–	–
St_{65}	1	–	–	–	–
St_{66}	1	–	–	–	–
St_{67}	–	–	1	–	–
St_{68}	–	–	1	–	–
St_{69}	–	–	1	–	–
St_{70}	–	–	1	–	–

(*continued on next page*)

Table 12 (*continued*).

Estimation type	Development effort	Size effort	Cost effort	Time effort	Maintenance effort
St_{71}	–	–	1	–	–
St_{72}	1	–	–	–	–
St_{73}	1	–	–	–	–
St_{74}	1	–	–	–	–
St_{75}	1	–	–	–	–
St_{76}	1	–	–	–	–
St_{77}	–	–	–	1	–
St_{78}	1	–	–	–	–
St_{79}	1	–	–	–	–
St_{80}	1	–	–	–	–

performs better as compared to BPNN, FLANN and ELMAN NN. FLANN has better performance as compared to BPNN, RBFN and ELMAN when NASA dataset is used. Performance of ANN is better compared to GRNN and Spiking neural network while using ISGSB dataset. Recurrent neural network has better performance with random data used for effort estimation. While using Project data as a dataset, CCNN and Elman neural network shows similar performance and is better than FFBPNN and QONN. PRED_MLP shows better performance than ANN, RBF and Elman neural network while Deshernais used as dataset. Using Maxwell dataset, Elman neural network has better performance than ANN, FLANN, LSTM and OSLRNN. FCM–FLANN has better performance than SNN, LSTM and OSLRNN while using CHINA dataset. LSTM shows better performance than FFNN, RBFN, WNN-Gaussian and OSLRNN when ALBRECHT dataset is used. Performance of MLP is better than RBF, WNN-Gaussian and SNN when IBMDPS dataset is used for effort estimation. Deep learning methods (Recurrent neural network, Elman, LSTM) showed better performance in 4 (Random data, Project data, Maxwell and ALBRECHT) dataset. Neural network methods (ANN, MLP and GRNN) showed better performance in 4 (COCOMO, ISGSB, Deshernais and IBMDPS) datasets and Higher Order neural network method (FLANN) showed better performance in 2 (NASA and CHINA) datasets.

5.10. | Q10: Comparative analysis of previous survey studies with the proposed article

From the literatures, it is evident that some of the earlier literature study is conducted on software effort estimation procedures. A comparative analysis of previous studies with present work is illustrated in Table 15. In this section we analyzed the previous review papers and compared with the proposed article. Previous studies surveyed about traditional methods, use case points, expert judgment, ML techniques, non-ML techniques, optimization algorithms, soft computing techniques and neural networks. We surveyed various types of neural networks that include Higher order neural networks and deep learning neural networks. Many of the previous studies have considered the articles ranging from 20 to 76, whereas in our article we considered 80 articles that exclude these review articles used in this section. In the previous studies authors mention comparative analysis for effort type, datasets, evaluation measures, intelligent method where as our study extended the comparative analysis of various aspects such as hybrid technique, year of publication, dataset and intelligent method combination, performance comparison of various intelligent method with respective of MMRE. Our comparative analysis states FFNN (MLP, BP) and RBFN is used extensively in the case of datasets COCOMO and then NASA are used frequently in experimentation. MMRE is mostly used accuracy measure to evaluate the model. FLANN with PSO combination is mostly used technique among all hybrid techniques. The software development effort is calculated frequently with maintenance effort, cost estimation, and size effort. FFNN (MLP, BP) with COCOMO dataset is used mostly in the considered studies. The analytical evidence in this study will help the researchers in particular for enhancing the research using ANN and Deep learning.

5.11. | Q11: What are the various future suggestive techniques for SEE?

Most of the studies are based on FFNN for the prediction of software effort estimation. FFNN estimations are better than other techniques and are widely used by researchers. Other techniques could also be used for effort estimation. Very few researchers have conducted their research in higher-order neural networks and deep learning neural networks. Future study may focus on working and exploring more about these neural networks and their application in effort estimation. Changes in network architecture with new functions may give different results. Therefore, more research on higher-order neural networks especially using deep learning neural networks may lead to a new direction in the field of SEE.

ANN is used for software effort estimation because of its ability to model complex relationship between independent variables and dependent variables and can learn from previous data. It can also produce efficient result for large volume of data with frequently changing condition and is applicable in the field with sound knowledge. But neural networks are considered to be black box where we cannot understand its internal structure and topologies to construct the NN. Higher Order neural network has unique features such as high storage capacity, stronger approximation with faster convergence property and higher fault tolerance capacity. On the other hand, disadvantage of using HONN is that it has limit for the size of input and takes longer time to process. Deep learning neural networks has the ability to learn high-level features with more complexity and abstraction. The deep feature hierarchy in deep learning models helps to achieve effective performance in multiple tasks. This performance is possible at the cost of high complexity. The technique that enables DNN to have efficient results along with cost-effective hardware are critical to expanding and developing in new and existing domains [107].

Table 13
Dataset versus Intelligent method in Software effort estimation.

Dataset Vs Intelligent Method	COCOMO	NASA	ISBSG	Project Data	Random Data	Kemerer	Desharnais	Albrecht	Maxwell	CF	Artificial	25KDSI	Kotengray	Canadian function	IBMDPS	China	Tukutuku	IKH	SAMOA	UMIS, QUES	Zia, CD1, CD2
FFNN, MLP, BP	20	9	5	8	6	3	6	4	1	1	1	1	1	1	2	–	–	–	1	–	–
Regression	3	1	2	4	1	–	1	–	–	–	–	–	–	–	–	–	–	–	1	–	1
COCOMO	4	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
RBFN	7	2	2	1	–	1	1	1	–	–	–	–	–	1	2	–	1	–	–	–	–
GRNN	1	–	2	3	3	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
CARTX	2	1	1	–	–	1	2	1	1	–	–	–	–	–	–	–	–	–	–	–	–
CBR	–	–	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Rule Induction	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
QONN	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
MRA Model	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
ABE	–	–	1	–	–	–	1	1	1	–	–	–	–	–	–	–	–	–	–	–	–
SWR	–	–	1	–	–	–	1	1	1	–	–	–	–	–	–	–	–	–	–	–	–
ANN–NABE	–	–	1	–	–	–	1	1	1	–	–	–	–	–	–	–	–	–	–	–	–
ANN–COCOMO	1	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–
Usecase	–	–	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
ANN–RB	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
ANN–GR	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
ANN–UC	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
ANN–COCOMO II	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
COCOMO II	2	2	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
COCOMO–ANN	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
SVM–RBF	1	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
MLP–ABC	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
ABC	2	1	–	–	–	–	–	–	1	–	–	–	–	–	–	1	–	–	–	–	–
MDELP	1	1	–	–	–	1	1	1	–	–	–	–	1	–	–	–	–	–	–	–	–
CS–COCOMO	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
CS–ANN–COCOMO II	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
WNN	1	1	–	–	–	–	–	–	1	–	–	–	–	1	1	1	–	–	–	–	1
PSO–FLANN	3	3	–	–	–	–	–	–	3	–	–	–	–	–	–	–	–	–	–	–	–
FLANN	4	5	–	–	–	–	1	–	2	–	–	–	–	–	–	–	–	–	–	1	–
UKW/DBSCAN	1	1	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
SVR	1	1	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
FCM with FLANN	2	2	–	–	–	–	–	–	2	–	–	–	–	–	–	1	–	–	–	–	–
Spiking NN	–	–	2	–	–	–	–	–	–	–	–	–	–	–	2	2	–	–	–	–	–
Elman NN	1	1	–	1	1	–	1	–	1	–	–	–	–	–	–	–	–	1	–	–	–
LSTM	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
RHM	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
ANFIS	1	–	–	–	–	–	1	–	1	–	–	–	–	–	–	–	–	1	–	–	–
CCNN–UC	–	–	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
SONN	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
RNN	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
SLR	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
GMDH	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
BiLSTM-CRF	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
KFCM–OLSRNN	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
MLP–NCA	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
FLANN–ABC	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
FLANN–GA	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–
FLANN–CSA	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–
FLANN–ISO	1	1	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	1
WNN–Firefly and BAT	1	1	–	–	–	–	–	–	1	–	–	–	–	–	–	1	–	–	–	–	1
FLANN–WOA	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
RBFN–WOA	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Table 14
Performance of various intelligent methods with respective of different datasets.

Dataset	Method	MMRE
COCOMO	BPNN	0.58
	GRNN	2.08
	FLANN	1.28
	ELMAN	1.15
NASA	BPNN	0.1712
	RBFN	0.2556
	FLANN	0.4
	ELMAN	0.1678
ISGSB	ANN	0.91
	GRNN	0.26
	Spiking Neural network	0.15
Random data	FFBPNN	0.3221
	GRNN	0.3581
	Cascade FFBPNN	0.3899
	Recurrent NN	0.3917
	Elman BPNN	0.3403
Project Data	FFBPNN	0.1349
	CCNN	0.1486
	QONN	0.1315
	Elman NN	0.148
Deshernais	ANN	0.57
	RBF	0.42
	PRED_MLP	0.6347
	PRED_MLP_FS	0.2576
	Elman	0.5721
Maxwell	ANN	1.32
	FLANN	0.42
	Elman	1.3748
	LSTM	0.37
	OSLRNN	0.31
CHINA	SNN	0.23
	FCM–FLANN	0.45
	IFCM–FLANN	0.33
	LSTM	0.41
	OSLRNN	0.32
ALBRECHT	FFNN	0.14
	RBFNN	0.1654
	WNN-Gaussian	0.07
	LSTM	0.41
	OSLRNN	0.34
IBMDPS	MLP	1.82
	RBF	0.3639
	WNN-GAUSSIAN	0.0328
	SNN	0.25

6. Conclusion

This article is a state-of-the-art systematic literature review for various ANN techniques used for software effort estimation. The focus of this paper is to make a quantitative and qualitative analysis of all the literatures in the area of software effort estimation using ANN. The period of review is considered from 2000 to 2020. In this SLR, with considering questionnaire method different aspects of the study such as performance factors, publication frequency, estimation of development, size, time, cost and maintenance year wise analysis etc. are analyzed. From the study, it is evident that many researchers have focussed on development effort. Accurate estimation helps in better planning and execution of a software project. Various models using FLANN, FFNN and BPNN etc. have been utilized for the prediction of software effort by the researchers. From this review, it is evident that several neural networks, higher order neural networks and deep learning neural networks have been utilized for software effort estimation. The analytical perspective is concluded from the used questionnaire method in this article. From Q1, it can be inferred that COCOMO dataset has been frequently used for conducting the experiments. The next commonly used dataset is the NASA dataset and project dataset. From Q2 analysis, it is observed that FFNN has been extensively used along with multi-layer perception and back propagation followed by RBFN. Many other hybrid techniques are combined with ANN for better prediction of various software. From Q3, it is found that the evaluation measure mostly used is MMRE, next comes PRED, MRE, MdMRE in maximum number of researches. From Q4, the mostly used combination/hybrid technique such as PSO–FLANN has been inferred by considering several studies. From Q5 analysis, it is cleared about the dominance of IEEE Xplore publications followed by Springer series over others in publishing the ANN with software effort estimation. Q6 analyzed about the frequency of publications in the considered period of 2000–2020 and found that, year 2014 has published most papers followed by year 2010. Q7 analysis is particular about the various efforts and observed that, the development effort is calculated more often followed by cost effort estimation through ANN. From Q8, we concluded that the combination of FFNN, MLP, BP methods with COCOMO dataset has been widely used followed by FFNN, MLP, BP methods using NASA and Project datasets. Q9 is about the analysis among deep learning methods using Random data, Project data, Maxwell and ALBRECHT dataset. Results are supportive towards the deep learning methods over other classical ANN methods. On an overall analysis, this review is included with several comparative factors such as year of publication, hybrid techniques, combination of intelligent methods and their analysis, performance of various intelligent method with respective of MMRE, impact of publications, year wise data analysis etc for better analyze the state-of-the-art methods used for solving the software effort estimations using ANN. However, apart from all these mentioned studies few directions of software effort estimation is still in infancy. The future trend on these areas should focus on exact and more accurate evaluations which may not cause overrun of budget, impediment in delivery and affecting the quality issues. Moreover, for large scale projects instead of classical ANNs, researchers should emphasize on deep learning based NN for better evaluation of all the quality factors. Also, to meet the parallel demands of various customers with the agreed quality, the module definition along with quantification should be performed with efficient advanced intelligent methods. To meet the expected use cases in medium and large-scale projects, intelligent automation-based techniques are to be introduced for comparison among the potential use cases.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Table 15
Comparison of previous literatures with the present article.

S. No	Year	Author	No of Articles considered for analysis	Methods considered	Comparative analysis	Observation	Reference
1	2020	Reddy and S Behera	20	Swarm based algorithms for optimization	Effort type, Dataset, Evaluation measure	1) Development Effort, Cost Estimation is used mostly 2) frequently used datasets are COCOMO, NASA, MAXWELL, ISBSG 3) Mostly used evaluation measures are MMRE, PRED, MARE, and MdMRE	[108]
2	2020	Suresh Kumar and Behera	76	ANN, Fuzzy Logic, Evolutionary Computation	Dataset, Technique, Evaluation Factor	1) COCOMO and NASA as datasets 2) Neural Networks as soft computing technique 3) MMRE as Evaluation Factor are mostly used in their literature	[22]
3	2019	Mahmood et al	34	Use case points, Expert judgment-based	Techniques (UCP, Expert judgment), dataset, Accuracy Measure	1) Expert judgment SEE is mostly used mostly between UCP and Expert Judgment 2) Industrial project Datasets are used among all sorts of data 3) Most commonly used accuracy measure is MRE, MMRE, and PRED (25) 4) ML approaches are picking up the researcher's fascination towards more investigation into SEE	[109]
4	2019	Ali and Gravino	75	ML and non-ML Techniques	Technique, Dataset, Accuracy measure	1) ANN as a Technique 2) NASA as Dataset 3) MMRE as accuracy measure are used widely in these studies4) ANN and SVM out performed well in all ML techniques 5) Regression techniques are mostly used in non-ML techniques 6) SVM and Regression combination predictions better compared to ML and non-ML techniques	[110]
5	2012	Dave and Dutta	21	Neural Network methods, Traditional SEE models, Halstead model, Walston–Felix model, Bailey–Basili model and Doty model	Knowledge in applying NN to SEE, pros and cons of NN in SEE	1) ANN Models are better compared to most popular traditional model and regression analysis 2) ANN are more better than well-known model COCOMO 3) FNNN is suitable than major benchmark models such as Halstead model, Walston–Felix model, Bailey–Basili model and Doty model 4) Size of the database is an important criteria when using Neural Networks, and mostly used database is COCOMO	[20]

(*continued on next page*)

Table 15 (*continued*).

S. No	Year	Author	No of Articles considered for analysis	Methods considered	Comparative analysis	Observation	Reference
6		**Proposed Article**	**80**	**Neural Networks, Higher order neural networks, and deep learning methods**	**Dataset, Intelligent method, Accuracy measure, Combination techniques, publication, Year, Dataset and intelligent method combination, performance of intelligent method, Previous review comparison**	**1) FFNN (MLP, BP(and RBFN is used extensively 2) COCOMO and then NASA are used frequently in experimentation 3) MMRE is Mostly used measure 4) FLANN with PSO is mostly used technique among all hybrid techniques 5) More no of publications are from IEE, and Springer series 6) More no of articles published in the year 2014 7)Development effort calculated frequently 8) FFNN (MLP, BP) with COCOMO dataset is used mostly**	–

References

[1] I.F. de Barcelos Tronto, J.D.S. da Silva, N. Sant'Anna, Comparison of artificial neural network and regression models in software effort estimation, in: 2007 Int. Jt. Conf. Neural Networks, IEEE, 2007, pp. 771–776, http://dx.doi.org/10.1109/IJCNN.2007.4371055.

[2] I. Attarzadeh, S.H. Ow, Software development cost and time forecasting using a high performance artificial neural network model, Commun. Comput. Inf. Sci. (2011) 18–26, http://dx.doi.org/10.1007/978-3-642-18129-0_4.

[3] S. Mukherjee, R.K. Malu, Optimization of project effort estimate using neural network, in: 2014 IEEE Int. Conf. Adv. Commun. Control Comput. Technol, IEEE, 2014, pp. 406–410, http://dx.doi.org/10.1109/ICACCCT.2014.7019474.

[4] R. Shukla, A.K. Misra, Software maintenance effort estimation – Neural network vs regression modeling approach, Int. J. Comput. Appl. 1 (2010) 83–89, http://dx.doi.org/10.5120/595-688.

[5] E. Papatheocharous, A.S. Andreou, Problem of attribute selection for software cost estimation: Input backward elimination using artificial neural networks, IFIP Adv. Inf. Commun. Technol. (2010) 287–294, http://dx.doi.org/10.1007/978-3-642-16239-8_38.

[6] P. Suresh Kumar, H.S. Behera, Estimating software effort using neural network: An experimental investigation, in: A.K. Das, J. Nayak, B. Naik, S. Dutta, D. Pelusi (Eds.), Comput. Intell. Pattern Recognit, Springer, Singapore, Singapore, 2020, pp. 165–180, http://dx.doi.org/10.1007/978-981-15-2449-3.

[7] D.K.K. Reddy, H.S. Behera, Software effort estimation using particle swarm optimization: Advances and challenges, in: P.D. Das A., Nayak J., B. Naik, Dutta S. (Eds.), Comput. Intell. Pattern Recognit., Springer Singapore, Singapore, 2020, pp. 243–258, http://dx.doi.org/10.1007/978-981-15-2449-3_20.

[8] I. Attarzadeh, A. Mehranzadeh, A. Barati, Proposing an enhanced artificial neural network prediction model to improve the accuracy in software effort estimation, in: 2012 Fourth Int. Conf. Comput. Intell. Commun. Syst. Networks, IEEE, 2012, pp. 167–172, http://dx.doi.org/10.1109/CICSyN.2012.39.

[9] G.R. Finnie, G.E. Wittig, J.-M. Desharnais, A comparison of software effort estimation techniques: Using function points with neural networks, case-based reasoning and regression models, J. Syst. Softw. 39 (1997) 281–289, http://dx.doi.org/10.1016/S0164-1212(97)00055-1.

[10] E.S. Jun, J.K. Lee, Quasi-optimal case-selective neural network model for software effort estimation, Expert Syst. Appl. 21 (2001) 1–14, http://dx.doi.org/10.1016/S0957-4174(01)00021-5.

[11] B. Kitchenham, Procedures for Performing Systematic Reviews, Keele, 2004. http://www.annsaudimed.net/doi/10.5144/0256-4947.2017.79.

[12] P.S. Rao, K.K. Reddi, R.U. Rani, Optimization of neural network for software effort estimation, in: 2017 Int. Conf. Algorithms, Methodol. Model. Appl. Emerg. Technol, IEEE, 2017, pp. 1–7, http://dx.doi.org/10.1109/ICAMMAET.2017.8186696.

[13] A. Idri, A. Abran, S. Mbarki, An experiment on the design of radial basis function neural networks for software cost estimation, in: 2006 2nd Int. Conf. Inf. Commun. Technol., IEEE, Damascus, Syria, 2006, pp. 1612–1617, http://dx.doi.org/10.1109/ICTTA.2006.1684625.

[14] P.S. Rao, R.K. Kumar, Software effort estimation through a generalized regression neural network, in: Adv. Intell. Syst. Comput., 2015, pp. 19–30, http://dx.doi.org/10.1007/978-3-319-13728-5_3.

[15] K. Vinay Kumar, V. Ravi, M. Carr, N. Raj Kiran, Software development cost estimation using wavelet neural networks, J. Syst. Softw. 81 (2008) 1853–1867, http://dx.doi.org/10.1016/j.jss.2007.12.793.

[16] T.R. Benala, R. Mall, S. Dehuri, V.L. Prasanthi, Software effort prediction using fuzzy clustering and functional link artificial neural networks, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2012, pp. 124–132, http://dx.doi.org/10.1007/978-3-642-35380-2_16.

[17] V. Venkataiah, R. Mohanty, M. Nagaratna, Prediction of software cost estimation using spiking neural networks, in: Smart Intell. Comput. Appl. Smart Innov. Syst. Technol., Springer, Singapore, 2019, pp. 101–112, http://dx.doi.org/10.1007/978-981-13-1927-3_11.

[18] M. Choetkiertikul, H.K. Dam, T. Tran, T. Pham, A. Ghose, T. Menzies, A deep learning model for estimating story points, IEEE Trans. Softw. Eng. 45 (2019) 637–656, http://dx.doi.org/10.1109/TSE.2018.2792473.

[19] E. Praynlin, P. Latha, Performance analysis of software effort estimation models using neural networks, Int. J. Inf. Technol. Comput. Sci. 5 (2013) 101–107, http://dx.doi.org/10.5815/ijitcs.2013.09.11.

[20] V.S. Dave, K. Dutta, Neural network based models for software effort estimation: a review, Artif. Intell. Rev. 42 (2014) 295–307, http://dx.doi.org/10.1007/s10462-012-9339-x.

[21] S. Aljahdali, A.F. Sheta, N.C. Debnath, Estimating software effort and function point using regression, support vector machine and artificial neural networks models, in: 2015 IEEE/ACS 12th Int. Conf. Comput. Syst. Appl., IEEE, 2015, pp. 1–8, http://dx.doi.org/10.1109/AICCSA.2015.7507149.

[22] P. Suresh Kumar, H.S. Behera, Role of soft computing techniques in software effort estimation: An analytical study, in: A.K.D., et al. (Eds.), Comput. Intell. Pattern Recognit., in: Advances in Intelligent Systems and Computing, vol. 999, 2020, pp. 807–831, http://dx.doi.org/10.1007/978-981-13-9042-5_70.

[23] I. Kalichanin-Balich, C. Lopez-Martin, Applying a feedforward neural network for predicting software development effort of short-scale projects, in: 2010 Eighth ACIS Int. Conf. Softw. Eng. Res. Manag. Appl., IEEE, 2010, pp. 269–275, http://dx.doi.org/10.1109/SERA.2010.41.

[24] C. Lopez-Martin, C. Isaza, A. Chavoya, Software development effort prediction of industrial projects applying a general regression neural network, Empir. Softw. Eng. 17 (2012) 738–756, http://dx.doi.org/10.1007/s10664-011-9192-6.

[25] D. Manikavelan, R. Ponnusamy, Software cost estimation by analogy using feed forward neural network, in: Int. Conf. Inf. Commun. Embed. Syst., IEEE, 2014, pp. 1–5, http://dx.doi.org/10.1109/ICICES.2014.7033820.

[26] S. Laqrichi, F. Marmier, D. Gourc, J. Nevoux, Integrating uncertainty in software effort estimation using Bootstrap based neural networks, IFAC-PapersOnLine 48 (2015) 954–959, http://dx.doi.org/10.1016/j.ifacol.2015.06.206.

[27] T.M. Khoshgoftaar, A.S. Pandya, D.L. Lanning, Application of neural networks for predicting program faults, Ann. Softw. Eng. 1 (1995) 141–154, http://dx.doi.org/10.1007/BF02249049.

[28] A.R. Venkatachalam, Software cost estimation using artificial neural networks, in: Proc. 1993 Int. Conf. Neural Networks (IJCNN-93-Nagoya, Japan), IEEE, Nagoya, Japan, 1993, pp. 987–990, http://dx.doi.org/10.1109/IJCNN.1993.714077.

[29] H. Hamza, A. Kamel, K. Shams, Software effort estimation using artificial neural networks: A survey of the current practices, in: 2013 10th Int. Conf. Inf. Technol. New Gener., IEEE, 2013, pp. 731–733, http://dx.doi.org/10.1109/ITNG.2013.111.

[30] Radial Basis Function Networks, (n.d.). https://www.saedsayad.com/artificial_neural_network_rbf.htm.

[31] A.B. Nassif, M. Azzeh, L.F. Capretz, D. Ho, Neural network models for software development effort estimation: a comparative study, Neural Comput. Appl. 27 (2016) 2369–2381, http://dx.doi.org/10.1007/s00521-015-2127-1.

[32] J. Hou, Artificial Neural Network for Spectrum Unfolding Bonner Sphere Data, University of Tennessee, Knoxville, 2008.

[33] M.G. Epitropakis, V.P. Plagianakos, M.N. Vrahatis, Higher-order neural networks training using differential evolution, in: Int. Conf. Numer. Anal. Appl. Math. Hersonissos, Crete, Greece, 2006, pp. 376–379, http://dx.doi.org/10.13140/2.1.4155.4882.

[34] S. Xu, J.W.Z. Lu, A.Y.T. Leung, V.P. Iu, K.M. Mok, A novel higher order artificial neural networks, in: AIP Conf. Proc., 2010, pp. 1507–1511, http://dx.doi.org/10.1063/1.3452131.

[35] A.K. Alexandridis, A.D. Zapranis, Wavelet neural networks: A practical guide, Neural Netw. 42 (2013) 1–27, http://dx.doi.org/10.1016/j.neunet.2013.01.008.

[36] F.F.M. El-Sousy, A. Kh, Wavelet–neural–network control for maximization of energy capture in grid connected variable speed wind driven self-excited induction generator system, in: Discret. Wavelet Transform. - A Compend. New Approaches Recent Appl., InTech, 2013, http://dx.doi.org/10.5772/51253.

[37] A. Kaushik, A.K. Soni, R. Soni, An improved functional link artificial neural networks with intuitionistic fuzzy clustering for software cost estimation, Int. J. Syst. Assur. Eng. Manag. 7 (2016) 50–61, http://dx.doi.org/10.1007/s13198-014-0298-2.

[38] T.R. Benala, R. Mall, S. Dehuri, K. Chinna Babu, Software effort prediction using unsupervised learning (clustering) and functional link artificial neural networks, in: 2012 World Congr. Inf. Commun. Technol., IEEE, 2012, pp. 115–120, http://dx.doi.org/10.1109/WICT.2012.6409060.

[39] S.K. Nanda, D.P. Tripathy, Application of functional link artificial neural network for prediction of machinery noise in opencast mines, Adv. Fuzzy Syst. 2011 (2011) 1–11, http://dx.doi.org/10.1155/2011/831261.

[40] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Netw. 61 (2015) 85–117, http://dx.doi.org/10.1016/j.neunet.2014.09.003.

[41] A. Tavanaei, M. Ghodrati, S.R. Kheradpisheh, T. Masquelier, A. Maida, Deep learning in spiking neural networks, Neural Netw. 111 (2019) 47–63, http://dx.doi.org/10.1016/j.neunet.2018.12.002.

[42] D. Soni, Spiking Neural Networks, the Next Generation of Machine Learning, 2018, https://towardsdatascience.com/spiking-neural-networks-the-next-generation-of-machine-learning-84e167f4eb2b.

[43] J. Li, H. Xu, X. He, J. Deng, X. Sun, Tweet modeling with LSTM recurrent neural networks for hashtag recommendation, in: 2016 Int. Jt. Conf. Neural Networks, IEEE, 2016, pp. 1570–1577, http://dx.doi.org/10.1109/IJCNN.2016.7727385.

[44] P. Edinson, L. Muthuraj, Performance analysis of FCM based ANFIS and ELMAN neural network in software effort estimation, Int. Arab J. Inf. Technol. 15 (2018) 94–102.

[45] N. Sundaram, S.N. Sivanandam, Elman Neural Network Mortality Predictor for Prediction of Mortality Due to Pollution, 2016.

[46] V.S. Dave, K. Dutta, Application of feed-forward neural network in estimation of software effort, in: Proc. Int. Symp. Devices MEMS, Intell. Syst. Commun., 2011, pp. 5–9.

[47] C. Lopez-Martin, A. Chavoya, M.E. Meda-Campana, Use of a feedforward neural network for predicting the development duration of software projects, in: 2013 12th Int. Conf. Mach. Learn. Appl., IEEE, 2013, pp. 156–159, http://dx.doi.org/10.1109/ICMLA.2013.182.

[48] M. Madheswaran, D. Sivakumar, Enhancement of prediction accuracy in COCOMO model for software project using neural network, in: Fifth Int. Conf. Comput. Commun. Netw. Technol., IEEE, 2014, pp. 1–5, http://dx.doi.org/10.1109/ICCCNT.2014.6963021.

[49] G.E. Wittig, G. Finnic, Using artificial neural networks and function points to estimate 4GL software development effort, Australas. J. Inf. Syst. 1 (1994) 87–94, http://dx.doi.org/10.3127/ajis.v1i2.424.

[50] A. Idri, T.M. Khoshgoftaar, A. Abran, Can neural networks be easily interpreted in software cost estimation?, in: 2002 IEEE World Congr. Comput. Intell. 2002 IEEE Int. Conf. Fuzzy Syst. FUZZ-IEEE'02. Proc. (Cat. No.02CH37291), IEEE, 2002, pp. 1162–1167, http://dx.doi.org/10.1109/FUZZ.2002.1006668.

[51] A. Idri, S. Mbarki, A. Abran, Validating and understanding software cost estimation models based on neural networks, in: Proceedings. 2004 Int. Conf. Inf. Commun. Technol. from Theory to Appl. 2004, IEEE, 2004, pp. 433–434, http://dx.doi.org/10.1109/ICTTA.2004.1307817.

[52] Tong-Seng Quah, Mie Mie Thet Thwin, Application of neural networks for software quality prediction using object-oriented metrics, in: Int. Conf. Softw. Maintenance, 2003. ICSM 2003. Proceedings, IEEE Comput. Soc., 2003, pp. 116–125, http://dx.doi.org/10.1109/ICSM.2003.1235412.

[53] C. Lopez-Martin, Applying a general regression neural network for predicting development effort of short-scale programs, Neural Comput. Appl. 20 (2011) 389–401, http://dx.doi.org/10.1007/s00521-010-0405-5.

[54] M. Jorgensen, Experience with the accuracy of software maintenance task effort prediction models, IEEE Trans. Softw. Eng. 21 (1995) 674–681, http://dx.doi.org/10.1109/32.403791.

[55] K. Srinivasan, D. Fisher, Machine learning approaches to estimating software development effort, IEEE Trans. Softw. Eng. 21 (1995) 126–137, http://dx.doi.org/10.1109/32.345828.

[56] G. Wittig, G. Finnie, Estimating software development effort with connectionist models, Inf. Softw. Technol. 39 (1997) 469–476, http://dx.doi.org/10.1016/S0950-5849(97)00004-9.

[57] Miyoung Shin, A.L. Goel, Empirical data modeling in software engineering using radial basis functions, IEEE Trans. Softw. Eng. 26 (2000) 567–576, http://dx.doi.org/10.1109/32.852743.

[58] M. Shepperd, G. Kadoda, Comparing software prediction techniques using simulation, IEEE Trans. Softw. Eng. 27 (2001) 1014–1022, http://dx.doi.org/10.1109/32.965341.

[59] K. Molokken, M. Jorgensen, A review of software surveys on software effort estimation, in: 2003 Int. Symp. Empir. Softw. Eng. 2003. ISESE 2003. Proceedings, IEEE Comput. Soc., 2009, pp. 223–230, http://dx.doi.org/10.1109/ISESE.2003.1237981.

[60] H. Park, S. Baek, An empirical validation of a neural network model for software effort estimation, Expert Syst. Appl. 35 (2008) 929–937, http://dx.doi.org/10.1016/j.eswa.2007.08.001.

[61] A. Idri, A. Zahi, E. Mendes, A. Zakrani, Software cost estimation models using radial basis function neural networks, in: Softw. Process Prod. Meas, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 21–31, http://dx.doi.org/10.1007/978-3-540-85553-8_2.

[62] K. Iwata, Y. Anan, T. Nakashima, N. Ishii, Using an artificial neural network for predicting embedded software development effort, in: 2009 10th ACIS Int. Conf. Softw. Eng. Artif. Intell. Netw. Parallel/Distributed Comput., IEEE, 2009, pp. 275–280, http://dx.doi.org/10.1109/SNPD.2009.49.

[63] C.S. Reddy, K. Raju, A concise neural network model for estimating software effort, J. Recent Trends Eng. Technol. 1 (2009) 188–193.

[64] Y.F. Li, M. Xie, T.N. Goh, A study of the non-linear adjustment for analogy based software cost estimation, Empir. Softw. Eng. 14 (2009) 603–643, http://dx.doi.org/10.1007/s10664-008-9104-6.

[65] P. Jodpimai, P. Sophatsathit, C. Lursinsap, Estimating software effort with minimum features using neural functional approximation, in: 2010 Int. Conf. Comput. Sci. Its Appl., IEEE, 2010, pp. 266–273, http://dx.doi.org/10.1109/ICCSA.2010.63.

[66] I. Attarzadeh, Siew Hock Ow, Proposing a new software cost estimation model based on artificial neural networks, in: 2010 2nd Int. Conf. Comput. Eng. Technol., IEEE, 2010, pp. V3-487-V3-491. http://dx.doi.org/10.1109/ICCET.2010.5485840.

[67] S. Ajitha, T. Suresh Kumar, D. Evangelin Geetha, K. Rajani Kanth, Neural network model for software size estimation using use case point approach, in: 2010 5th Int. Conf. Ind. Inf. Syst., IEEE, 2010, pp. 372–376, http://dx.doi.org/10.1109/ICIINFS.2010.5578675.

[68] J. Kaur, S. Singh, K.S. Kahlon, P. Bassi, Neural network-a novel technique for software effort estimation, Int. J. Comput. Theory Eng. (2010) 17–19, http://dx.doi.org/10.7763/IJCTE.2010.V2.109.

[69] A. Zakrani, A. Idri, Applying radial basis function neural networks based on fuzzy clustering to estimate web applications effort, Int. Rev. Comput. Softw. 5 (2010) 516–524.

[70] P.V.G.D.P. Reddy, K.R. Sudha, P.R. Sree, S.N.S.V.S.C. Ramesh, Software effort estimation using radial basis and generalized regression neural networks, J. Comput. 2 (2010) 87–92, http://arxiv.org/abs/1005.4021.

[71] C. López-Martín, A. Chavoya, M.E. Meda-Campaña, Software development effort estimation in academic environments applying a general regression neural network involving size and people factors, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2011, pp. 269–277, http://dx.doi.org/10.1007/978-3-642-21587-2_29.

[72] A.B. Nassif, L.F. Capretz, D. Ho, Estimating software effort using an ANN model based on use case points, in: 2012 11th Int. Conf. Mach. Learn. Appl., IEEE, 2012, pp. 42–47, http://dx.doi.org/10.1109/ICMLA.2012.138.

[73] O.F. Sarac, N. Duru, A novel method for software effort estimation: Estimating with boundaries, in: 2013 IEEE INISTA, IEEE, 2013, pp. 1–5, http://dx.doi.org/10.1109/INISTA.2013.6577643.

[74] P.K. Das, M.K. Majumder, B.K. Kaushik, S.K. Manhas, Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS 2012), December 28-30, 2012, Springer India, New Delhi, 2014, http://dx.doi.org/10.1007/978-81-322-1602-5.
[75] C. Kamalakannan, L. Padma Suresh, S.S. Dash, B.K. Panigrahi, Power Electronics and Renewable Energy Systems, Springer India, New Delhi, 2015, http://dx.doi.org/10.1007/978-81-322-2119-7.
[76] R. Sarno, J. Sidabutar, Sarwosri, Improving the accuracy of COCOMO's effort estimation based on neural networks and fuzzy logic model, in: 2015 Int. Conf. Inf. Commun. Technol. Syst., IEEE, 2015, pp. 197–202, http://dx.doi.org/10.1109/ICTS.2015.7379898.
[77] S. Amasaki, C. Lokan, On applicability of fixed-size moving windows for ANN-based effort estimation, in: 2016 Jt. Conf. Int. Work. Softw. Meas. Int. Conf. Softw. Process Prod. Meas, IEEE, 2016, pp. 213–218, http://dx.doi.org/10.1109/IWSM-Mensura.2016.041.
[78] P. Rijwani, S. Jain, Enhanced software effort estimation using multi layered feed forward artificial neural network technique, Proc. Comput. Sci. 89 (2016) 307–312, http://dx.doi.org/10.1016/j.procs.2016.06.073.
[79] M. Azzeh, A.B. Nassif, A hybrid model for estimating software project effort from use case points, Appl. Soft Comput. 49 (2016) 981–989, http://dx.doi.org/10.1016/j.asoc.2016.05.008.
[80] R. de A. Araújo, A.L.I. Oliveira, S. Meira, A class of hybrid multilayer perceptrons for software development effort estimation problems, Expert Syst. Appl. 90 (2017) 1–12, http://dx.doi.org/10.1016/j.eswa.2017.07.050.
[81] S. Arora, N. Mishra, Software cost estimation using artificial neural network, in: Adv. Intell. Syst. Comput., 2018, pp. 51–58, http://dx.doi.org/10.1007/978-981-10-5699-4_6.
[82] S. Kumari, S. Pushkar, Cuckoo search based hybrid models for improving the accuracy of software effort estimation, Microsyst. Technol. 24 (2018) 4767–4774, http://dx.doi.org/10.1007/s00542-018-3871-9.
[83] M. Pandey, R. Litoriya, P. Pandey, Validation of existing software effort estimation techniques in context with mobile software applications, Wirel. Pers. Commun. 110 (2020) 1659–1677, http://dx.doi.org/10.1007/s11277-019-06805-0.
[84] S. Goyal, P.K. Bhatia, Feature selection technique for effective software effort estimation using multi-layer perceptrons, Lect. Notes Electr. Eng. 605 (2020) 183–194, http://dx.doi.org/10.1007/978-3-030-30577-2_15.
[85] A.J. Singh, M. Kumar, Comparative analysis on prediction of software effort estimation using machine learning techniques, SSRN Electron. J. (2020) 1–6, http://dx.doi.org/10.2139/ssrn.3565822.
[86] T.R. Benala, K. Chinnababu, R. Mall, S. Dehuri, A particle swarm optimized functional link artificial neural network (PSO-FLANN) in software cost estimation, 2013, pp. 59–66, http://dx.doi.org/10.1007/978-3-642-35314-7_8.
[87] Z.H. Wani, S.M.K. Quadri, Artificial bee colony-trained functional link artificial neural network model for software cost estimation, 2016, pp. 729–741, http://dx.doi.org/10.1007/978-981-10-0451-3_65.
[88] L. Kumar, S.K. Rath, Hybrid functional link artificial neural network approach for predicting maintainability of object-oriented software, J. Syst. Softw. 121 (2016) 170–190, http://dx.doi.org/10.1016/j.jss.2016.01.003.
[89] T.R. Benala, R. Mall, S. Dehuri, Software effort estimation using functional link neural networks optimized by improved particle swarm optimization, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), in: LNCS, vol. 8298, 2013, pp. 205–213, http://dx.doi.org/10.1007/978-3-319-03756-1_18.
[90] B. Rao, B. Sameet, G. Swathi, A novel neural network approach for software cost estimation using functional link artificial neural network (FLANN), J. Comput. 9 (2009) 126–131, http://paper.ijcsns.org/07_book/200906/20090618.pdf.
[91] A. Kaushik, N. Singal, A hybrid model of wavelet neural network and metaheuristic algorithm for software development effort estimation, Int. J. Inf. Technol. (2019) http://dx.doi.org/10.1007/s41870-019-00339-1.
[92] T.R. Benala, R. Mall, S. Dehuri, P. Swetha, Software effort estimation using functional link neural networks tuned with active learning and optimized with particle swarm optimization, in: B.K. Panigrahi, P.N. Suganthan, S. Das (Eds.), Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), Springer International Publishing, Cham, 2015, pp. 223–238, http://dx.doi.org/10.1007/978-3-319-20294-5_20.
[93] S. Bilgaiyan, S. Mishra, M. Das, Effort estimation in agile software development using experimental validation of neural network models, Int. J. Inf. Technol. 11 (2019) 569–573, http://dx.doi.org/10.1007/s41870-018-0131-2.
[94] V. Venkataiah, R. Mohanty, M. Nagaratna, Prediction of software cost estimation using spiking neural networks, Smart Innov. Syst. Technol. 105 (2019) 101–112, http://dx.doi.org/10.1007/978-981-13-1927-3_11.
[95] M. Qin, L. Shen, D. Zhang, L. Zhao, Deep learning model for function point based software cost estimation -an industry case study, in: Proc. - 2019 Int. Conf. Intell. Comput. Autom. Syst. ICICAS 2019, 2019, pp. 768–772, http://dx.doi.org/10.1109/ICICAS48597.2019.00165.
[96] V. Resmi, S. Vijayalakshmi, Kernel fuzzy clustering with output layer self-connection recurrent neural networks for software cost estimation, J. Circuits, Syst. Comput. 29 (2019) 1–17, http://dx.doi.org/10.1142/S0218126620500917.
[97] A.B. Nassif, L.F. Capretz, D. Ho, Software effort estimation in the early stages of the software life cycle using a cascade correlation neural network model, in: 2012 13th ACIS Int. Conf. Softw. Eng. Artif. Intell. Netw. Parallel/Distributed Comput., IEEE, 2012, pp. 589–594, http://dx.doi.org/10.1109/SNPD.2012.40.
[98] Hui Zeng, D. Rine, Estimation of software defects fix effort using neural networks, in: Proc. 28th Annu. Int. Comput. Softw. Appl. Conf. 2004. COMPSAC 2004, IEEE, 2004, pp. 20–21, http://dx.doi.org/10.1109/CMPSAC.2004.1342658.
[99] M.K. Ghose, R. Bhatnagar, V. Bhattacharjee, Comparing some neural network models for software development effort prediction, in: 2011 2nd Natl. Conf. Emerg. Trends Appl. Comput. Sci., IEEE, 2011, pp. 1–4, http://dx.doi.org/10.1109/NCETACS.2011.5751391.
[100] A. Heiat, Comparison of artificial neural network and regression models for estimating software development effort, Inf. Softw. Technol. 44 (2002) 911–922, http://dx.doi.org/10.1016/S0950-5849(02)00128-3.
[101] V.S. Dave, K. Dutta, Neural network based software effort estimation & evaluation criterion MMRE, in: 2011 2nd Int. Conf. Comput. Commun. Technol. ICCCT-2011., vol. 1, 2011, pp. 347–351, http://dx.doi.org/10.1109/ICCCT.2011.6075192.
[102] C. López-Martín, Predictive accuracy comparison between neural networks and statistical regression for development effort of software projects, Appl. Soft Comput. 27 (2015) 434–449, http://dx.doi.org/10.1016/j.asoc.2014.10.033.
[103] C. López-Martín, A. Abran, Neural networks for predicting the duration of new software projects, J. Syst. Softw. 101 (2015) 127–135, http://dx.doi.org/10.1016/j.jss.2014.12.002.
[104] A. Sheta, D. Rine, S. Kassaymeh, Software effort and function points estimation models based radial basis function and feedforward artificial neural networks, Int. J. Next-Gener. Comput. 6 (2015) 192–205, http://dx.doi.org/10.13140/RG.2.1.3749.0004.
[105] A. Panda, S.M. Satapathy, S.K. Rath, Empirical validation of neural network models for agile software effort estimation based on story points, Proc. Comput. Sci. 57 (2015) 772–781, http://dx.doi.org/10.1016/j.procs.2015.07.474.
[106] A. Kaushik, D.K. Tayal, K. Yadav, The role of neural networks and metaheuristics in agile software development effort estimation, Int. J. Inf. Technol. Proj. Manag. 11 (2020) 50–71, http://dx.doi.org/10.4018/IJITPM.2020040104.
[107] V. Sze, Y.H. Chen, T.J. Yang, J.S. Emer, Efficient processing of deep neural networks: A tutorial and survey, Proc. IEEE. 105 (2017) 2295–2329, http://dx.doi.org/10.1109/JPROC.2017.2761740.
[108] D.K.K. Reddy, H.S. Behera, Software Effort Estimation using Particle Swarm Optimization: Advances and Challenges, 2020, pp. 243–258, http://dx.doi.org/10.1007/978-981-15-2449-3_20.
[109] Y. Mahmood, N. Kama, A. Azmi, A systematic review of studies on use case points and expert-based estimation of software development effort, J. Softw. Evol. Process. (2020) 1–20, http://dx.doi.org/10.1002/smr.2245.
[110] A. Ali, C. Gravino, A systematic literature review of software effort prediction using machine learning methods, J. Softw. Evol. Process. 31 (2019) 1–25, http://dx.doi.org/10.1002/smr.2211.