Advances in Science, Technology and Engineering Systems Journal Vol. 6, No. 4, 62-71 (2021)

www.astesj.com

ASTESJ

ISSN: 2415-6698

Optimized Component based Selection using LSTM Model by Integrating Hybrid MVO-PSO Soft Computing Technique

Anjali Banga*, Pradeep Kumar Bhatia

Department of Computer Science and Engineering, Guru Jambheshwar University of Science and Technology, Hisar-Haryana, 125001, India

ARTICLE INFO

Article history:
Received: 09 April, 2021
Accepted: 17 June, 2021
Online: 10 July, 2021

Keywords:
CBSE
LSTM
Deep learning
Neural Network

ABSTRACT

Research focused on training and testing of dataset after Optimizing Software Component with the help of deep neural network mechanism. Optimized components are selected for training and testing to improve the accuracy at the time of software selection. Selected components are required to be attuned and accommodating as per requirement. Soft computing mechanism such as PSO and MVO will be used for optimization. Deep Neural-Network mechanism is performing training and testing to get the confusion metrics of true positive/negative and false positive/negative. The accuracy, precision, recall value and f-score are computed to assure accuracy of proposed work. The proposed mechanism is making use of LSTM layer for more accurate output. Proposed research is exploring inadequacy of existing research and extent of incorporation of previous mechanism to soft computing mechanism in CBSE.

1. Introduction

Research is considering the dataset of the CBSE model [1] where the dataset presenting software component selection is trained and during testing of the trained network the confusion matrix is produced. According to the confusion matrix the accuracy, F-score, Recall, and precision values are found. On the other hand, data set of each grade would be passed to a hybrid MVO-PSO optimizer to find an optimized rating for each grade to filter the dataset. It could be said that the optimized value for each result is kept to find accuracy, F-score, Recall, and precision values accordingly. Finally, the comparison of accuracy, F-score, recall as well as precision values for non-optimized dataset to optimized Precision, F-score, and Recall for optimized dataset would be performed.

1.1. CBSE

The method of creating various software projects in different categories has been examined in Software engineering. Computer engineering applications have often been used to achieve this goal. However, the dependability of the software system is a difficult job to predict. CBSE may be regarded as a software reliability mechanism that is able to handle the issue. The broad method that supports the creation of various components depending on current software research was evaluated in terms of component-based software engineering. The newest software is not a simple task for beginners, but CBSE [2] allows developers to minimize efforts during the creation of software. Different influence variables are important in the case of CBSE, such as reusability, reliability, component dependence, and interactions between components. These characteristics promote the development of new software and reduce system complexity. There were many software computing methods that tried to predict software dependability. There are several observations. During software development, the selection of component steps has been discovered. Component-dependent design patterns of software are utilized for the recovery and assembly of components.

1.2. Reinforcement Learning

Reinforcement Learning is determined as Machine Learning. It is a branch of AI. Reinforcement Learning has been considered as a category of Machine Learning. It is also considered a branch of AI. Exactly permit a representative of hardware and computer program to mechanically identify perfect attitude in particular circumstances, for maximizing its efficiency. Exactly a type of Machine Learning method which permits representative of hardware and computer program to mechanically identified perfect attitude within a particular circumstance, for maximizing its efficiency. Reinforcement learning is utilized in various graphical games and uses a multi-agent mechanism to control the approach to environment exploration. It is also integrated with the company of abstraction methods which have the various levels to form powerful games dependent on artificial intelligence.

*Corresponding Author: Anjali Banga, Email: banga.anjali88@gmail.com

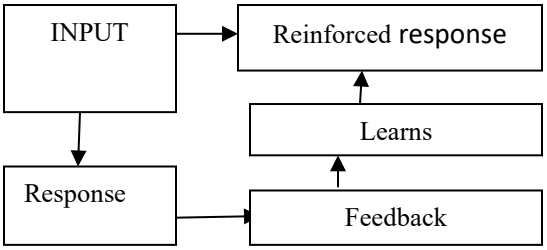

Figure 1: Reinforcement learning

Agent: The performer in the learning system is an agent. All actions are performed by the agent in the environment. The agent gets a reward as per action.

State: The state represents the current status of the environment that plays a significant role in assigning a reward to agents.

Environment: All action of agents is performed in the environment and rewards are providing to the agent as per the state of the environment.

Action: It is a method of representative due to which it communicates and exchanges its setting, in the middle of states. Whenever an action is executed by representatives it gives output inform of reward-dependent on the setting.

Reward: A reward in RL is part of the feedback from the environment. When an agent interacts with the environment, he could observe changes in state and reward signals through his actions.

1.3. Recurrent Neural Network

RNN network has been considered as a class of neural networks where interconnectivity among nodes is forming a directed graph. It is also creating a temporal sequence which is allowing it to show temporal dynamic behavior. RNN is capable to utilize their memory to compute sequences of inputs of different size as these are inherited from neural networks that are based on feed-forward. It is making them unable to implement operations like un segmented as well as interconnected consideration. RNN has been utilized to take into account two different broad categories of networks that are supporting the usual structure. Here one is having finite impulse. But another one is having impulse which is infinite. Such categories of networks have exhibited runtime actions that are not permanent.

1.3.1. Long Short-Term Memory

LSTM is a well-recognized artificial RNN. This is often used in the field of profound education. Feedback connectivity is provided by LSTM. It is not like a neural network feed. Not only are single data points like graphs processed. Sequences of information such as audio and video are also completed. In the case of LSTM networks, categorization is deemed appropriate. It does process and predicts based on information from time series. This is because there may be temporal delays not known throughout time series in important occurrences.

1.3.2. RNN and LSTM

An ongoing neural network is also known as an RNN. The category of ANN is examined. The node connections provide a network guided by a graph. You accomplished this with the sequence of time. Standard recurring neural networks have disappearances and explosions. LSTM networks were regarded as RNN types. Besides conventional units, LSTM is supported by special units.

1.3.3. Resolving Overfitting Problem by Dropout Layer

The dropout layer is playing a significant role in resolving the issue of over-fitting. The issue of over-fitting arises during the training of the neural network model. The dropout layer is used to handle such issues. If the training is continued, then the model adopts idiosyncrasies. Sometime training becomes less suitable for data that is new to it. This data could be different samples from the population. The model is considered to over-fit when it is too well-adapted to training as well as validating data.

Over-fitting is traced during plotting by checking the validation loss. The model is over-fitting when training loss is constant or it is decreasing. Techniques known as regularizes are used to minimize the influence of over-fitting. Dropout has been considered one out of them.

Dropout is working by eliminating or dropping out the inputs to layer. These could be input variables in a sample of data that is the output of the previous layer. In other words, the Dropout layer is attached to the model among previous layers. It applies to the results of the last layer that have been fed to the next layer. This is influenced by the simulation of a huge network with various network structures. Dropout rate could be considered to layer as chances of configuring every input to layer.

2. Literature Review

There are several types of research in the field of component-based software systems, optimization mechanisms, and neural networks. Researchers have used the SVM as a Classification Method for the Prediction of a defect in software with code metrics. Moreover, research for Performance Modeling of Interaction protocol for CBSD using OOPS based simulation came into existence. After some time, research related to software components selection optimization for CBSE development was made. Researchers applied particle swarm optimization for the performance prediction of the software components. Building models for optimized CBSE was built in several applications development. Some researchers did reliability estimation and performed prediction and measurement of CBSE. A Genetic Algorithm was proposed to manage SVM for predicting components that might be fault-prone. S. Di Martino proposed genetic algorithm. The objective of their research was to set the SVM in order to forecast components that are fault-prone. M. Palviainen did research to estimate reliability. They predicted and measured of component-based software.

In [3], the author applied PSO to software performance prediction.

In [4], the author proposed optimization model. This model has been developed for selection of software component. It has been used in development of several applications.

In [5], the author used SVM based classifier approach for reusability of software components.

In [6], the author performed multi-objective optimization. They did optimization of software architectures. Authors have used Ant Colony Optimization mechanism to accomplish their objective.

In [7], the author did estimation of software reusability in case of component-based system. They made used of several soft computing techniques in their research.

In [8], the author proposed adaptive Neuro fuzzy model. The objective of their research was to predict the reliability in case of component-based software systems.

In [9], the author introduced research on Test case prioritization. This research considered prioritization to perform regression testing. Research made use of ant colony optimization.

In [10], the author proposed research on Neuro-Fuzzy Model in order to find & optimize the Quality as well as Performance in case of CBSE.

In [11], the author presented dynamic mechanism in order to get software components with support of genetic algorithm.

In [12], the author proposed LSTM-based Deep Learning Models. The objective of research was to perform Non-factoid Answer Selection.

In [13], the author did research on multi-Verse Optimizer. They considered it as nature-inspired mechanism in case of global optimization.

In [14], the author proposed research to detect inconsistency in software component. Author made use of ACO and neural network mechanism.

In [15], the author presented deep learning mechanism in case of short-term traffic forecast. The research considered LSTM network.

In [16], the author proposed multiple target deep learning in case of LSTM.

In [17], the author proposed Preference-based component identification by making use of PSO.

In [18], the author proposed research on deep learning in order to perform solar power forecasting. Their approach made use of AutoEncoder along with LSTM Neural Networks.

In [19], the author presented quality assurance by soft computing mechanism. The research focused in field of component-based software.

In [20], the author did quality prediction by making use of ANN. Their research was based on Teaching-Learning Optimization.

In [21], the author proposed multi-objective model for optimization.

In [22], the author did Component selection considering attributes.

In [23], the author did software reliability prediction by making use of Bio Inspired approach.

In [24], the author proposed identification and selection of software component.

In [25], the author proposed model in order to predicting CBS reliability by making used of soft computing mechanism.

In [26], the author proposed enhanced Ant lion Optimizer along with Artificial Neural Network. This research focused on predicting Chinese Influenza.

In [27], the author Imoize considered software reuse and metrics in software engineering.

In [28], the author focused on improvement of reusability of component-based software. Research considered the advantages of software component by making use of data mining.

In [29], the author introduced hybrid Neuro-fuzzy as well as model for feature reduction in order to perform classification.

Table 1: Literature review

Author/ Year	Objective of research	Methodology	Limitation
U. Sharma/2012	Proposing reusability of software component	SVM	Slow mechanism has been proposed.
D. Gao/2015	Implementing test case prioritization in case of regression testing	ACO	Research has limited scope
G. Kumar/2015	Estimating and optimizing quality as well as performance in case of CBSE	Neuro fuzzy model	Need to improve the training efficiency
S. Vodithala/2015	Proposing dynamic mechanism to perform retrieval of software component	Genetic algorithm	Work is suffering from limitation of genetic algorithm
Ashu/ 2016	To build efficient IDS	LSTM algorithm	Research is not making using of optimizer to increase performance
O. Bhardwaj/2018	Assuring Quality by soft computing approach in component based software	CBSC	Research has not considered intelligent approach for accurate prediction.
P. Tomar/2018	To forecast prediction of quality by making use of ANN mechanism in case of Teaching-Learning Optimization for component-based software systems	ANN	The research need to do more work on performance and accuracy.
L. Mu/2018	Peforming the multi-objective optimization model of component selection	Multi-objective optimization	The optimization of multiple objectives is challenging and complex operation.
S. Gholamshahi/2019	Performing the preference based identification of component by making use of optimization technique.	PSO	There are several optimization mechanisms such as MVO that could perform better than PSO.
C. Diwaker/2019	Proposing prediction Model for CBSC for Reliability with support of Soft Computing	CBSC	Prediction model need to be more accurate and reliable.
HongpingHu/2019	Proposing improved ALO and ANN for Chinese Influenza Prediction	Ant lion optimization and artificial neural network	The performance of such system is slow there is need to filter the dataset
A. L. Imoize/2019	Reviewing the Software Reuse as well as Metrics in case of software Engineering	Software metrics	The research has limited scope due to lack of technical work

G. Maheswari/2019	To improve the reusability and Measuring Performance Merits in case of CBSE in case of Data Mining	Data mining	Research considered reusability and performance metric but there is need to consider optimization mechanism.
Himansu Das/2020	Proposing Hybrid Neuro-Fuzzy as well as model for feature reduction during classification	Hybrid Neuro-fuzzy	Need to introduce optimization mechanism to increase accuracy during feature reduction.

3. Problem Statement

Many studies shown the selection of software components, but the optimization mechanism is required for better performance. PSO was used to optimize results in previous studies. However, it is found that MVO offers better performance. In addition, an intelligent model should be introduced that may allow for a deep learning approach using RNN based on LSTM. However, it takes lot of time during training and testing. Then it finds accuracy, f-score according to confusion matrix. The optimization has been included to neural network model for better performance. Through the incorporation of the LSTM MVO-PSO hybrid method, the proposed study is needed to address the performance and accuracy problem.

The proposed work is answer for the accuracy and performance issues faced in previous researches.

4. Proposed Work

In the proposed work, the dataset of the CBSE model is considered. This dataset is trained using a neural network mechanism. During testing of the trained network, the confusion matrix is produced. On the basis of this confusion matrix the accuracy, F-score, Recall, and precision values are calculated. On other hand, the dataset is classified grade-wise in order to get optimized value for each grade. The data set of each grade would be passed to a hybrid MVO-PSO optimizer in order to get the optimized rating for each grade. The data of each grade would be filtered on basis of these optimized values. In another word, the data above the optimized value for each result would be kept. Then the accuracy, F-score, Recall, and precision values are calculated considering this filtered dataset. Then the comparison of accuracy, F-score, Recall, and precision values for non-optimized datasets are made to optimize, F-score, Recall, and precision values for the optimized dataset.

The figure illustrates the proposed system is required to train a component-based selection system that should be capable to predict with maximum accuracy. The trained model for component-based selection would be made with the support of LSTM and simulated in a Matlab environment. Existing researches in component-based selection have provided limited accuracy with limited precision, f-score, and recall value. The implementation of such a model is quite challenging but such research opens doors for innovations. There are several existing types of research that have contributed to the field of component-based selection. It has been observed that previous researches have made use of Fuzzy logic, Genetic algorithm, Machine learning mechanism, KNN classification, LSTM model. But these researches are suffering from accuracy issues.

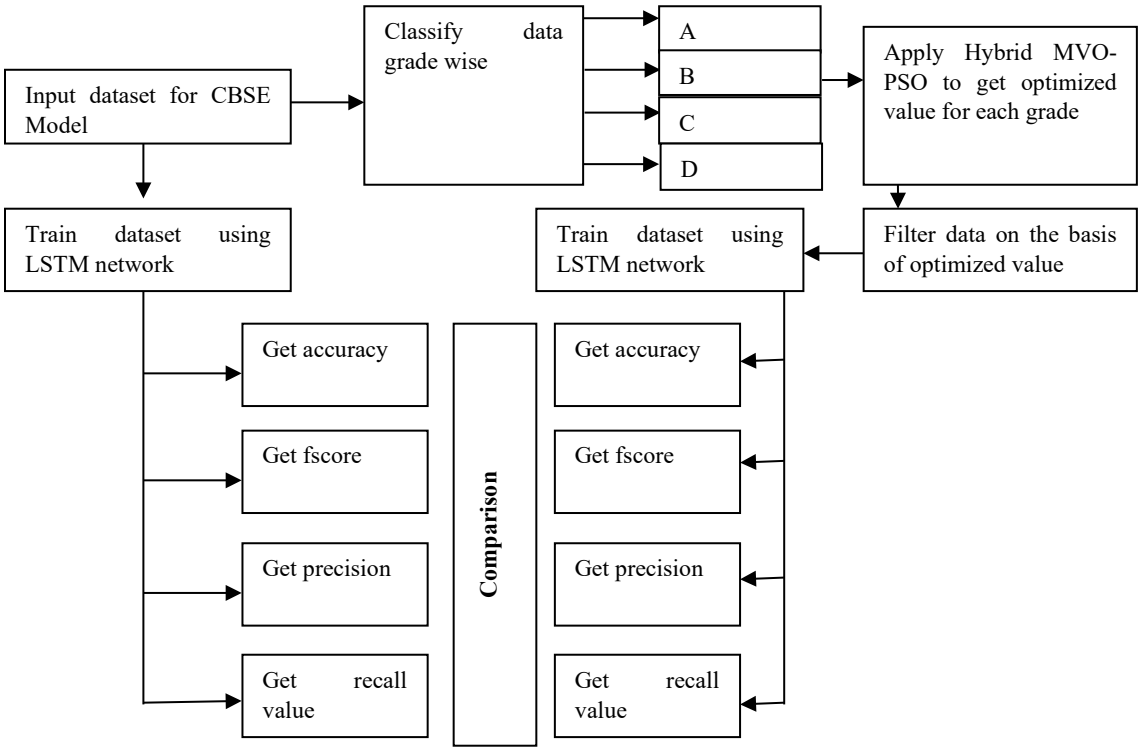

Figure 2: Architecture of Proposed model

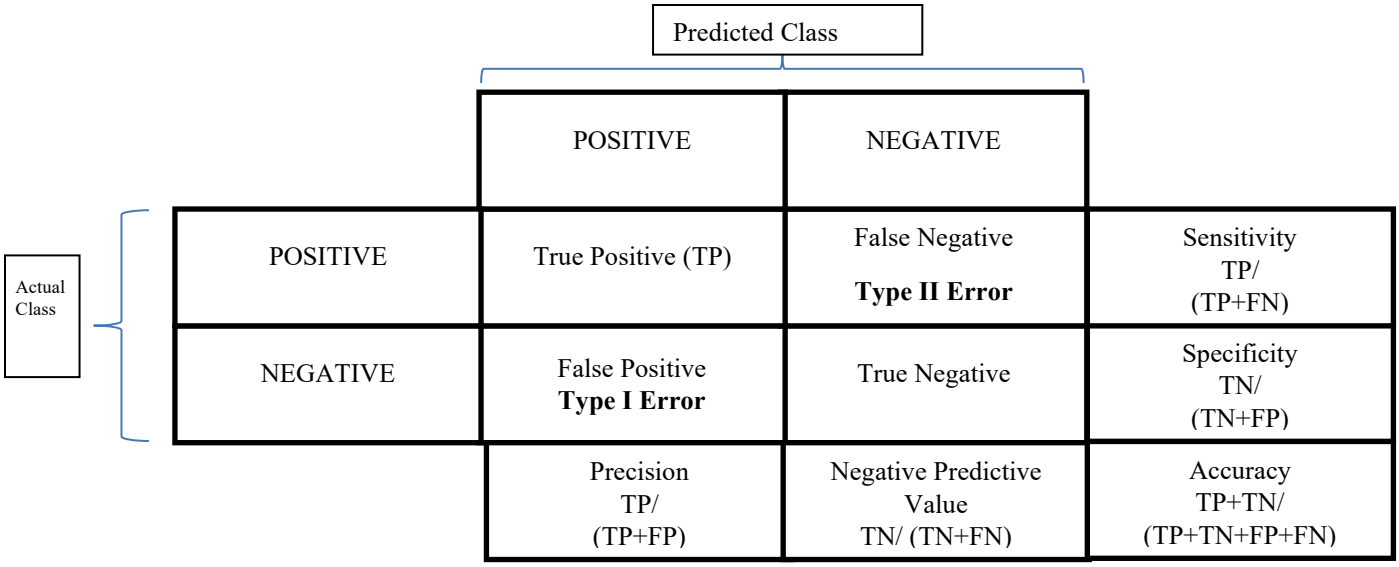

Figure 3: Confusion Matrix

Moreover, the time consumption during training of the network model is more. This research motivates to development of a model that should be trained fast as compared to previous models. Moreover, previous researches have also motivated to increase the accuracy using a two-layer LSTM model considering hidden layer. The proposed research is supposed to provide fast training to the dataset and more accurate prediction as compared to previous researches. The major objective of the research is to study existing literature on various component-based selections. The study and analysis of various component-based selections have been performed during research. Research would propose component-based selection with the support of LSTM layers. Then simulation would be made to perform result analysis. The comparison of existing and proposed work is made afterward.

Long Short-Term Memory networks have been considered as a category of recurrent neural networks. This is found capable to get taught order dependence in case of sequence prediction problems. This is a behavior needed in case of complicated issue domains like translation by machine. Long Short-Term Memory has been considered a complicated field of deep learning. This is difficult to understand Long Short-Term Memory. There has been little work in the field of Long Short-Term Memory. LSTM units are consisting of a 'memory cell'. These memory cells are capable to maintain data in memory for a large time. Users are moving from RNN to LSTM because it is introducing more controlling knobs. They are capable to manage the flow and mixing of Inputs according to trained Weights. So it provides flexibility during the management of outputs. Thus LSTM is providing the ability to manage and good results.

4.1. Performance Parameters

In this section, we will define the following parameters and confusion matrix is produced using true positive (TP), true negative (TN), false positive (FP), false negative (FN).

TP: True positives have been considered as correctly predicted positive values. In other words, the value of the real category is true and the value of the category that has been predicted is also true.

TN: True negative have been considered as correctly predicted negative values. In other words, the value of the real category is false and the value of the category that has been predicted is also false.

- FP: False positive is the case when the actual category is false but the predicted category is true.
- FN: False-negative is the case when the actual category is true but the predicted category is false.

Parameters utilized to confirm results have been f-score, recall, accuracy and precision which have been explained as follow:

1. Accuracy has been considered as intuitive performance measure. This is the ratio of correctly forecasted findings to total findings.

Accuracy = (True Positive + True Negative) / (True Positive + False Positive + False Negative + True Negative)

2. Precision has been considered as ratio of positive observations that have been correctly predicted to total predicted positive observations.

Precision = True Positive / (True Positive + False Positive)

3. Recall has been considered as ratio of positive observations that have been predicted in correct manner to overall findings in real class - yes.

Recall = (True positive) / (True Positive + False Negative)

4. F1 Score has been weighted average in case of Precision as well as Recall. Score is taking false positives as well as false negatives in consideration.

F1 Score = 2 x (Recall x Precision) / (Recall value + Precision)

5. Simulation

In this section, dataset of 629 packages have been considered for training purposed in research where the average rating is available considering factors such as number of reviews, total sentences, feature requests, feature requests in percentage, problem discoveries, problem discoveries in percentage, GUI Contents, Feature and Functionality, Improvement, Pricing, Resources, Security. A network model has been trained considering this dataset.

The grade is allotted according to the average rating

if Average Rating >4.5 then grade is A
if Average Rating lies between 4 and 4.5 then grade is B
if Average Rating lies between 3 and 4 then grade is C
if Average Rating <3 then grade is D

The classification of record counts according to grade have been discussed below

Table 2: Grade wise record count before optimization

GRADE	Record count
A	114
B	229
C	241
D	45
Total	629

In order to get 100% accuracy, there is need of following confusion matrix

Table 3: Confusion matrix required to get 100% accuracy

	A	B	C	D
A	114	0	0	0
B	0	229	0	0
C	0	0	241	0
D	0	0	0	45

But of the model is trained without optimization the confusion matrix is

Table 4: Confusion matrix before optimization

	A	B	C	D
A	110	1	1	0
B	1	225	2	0
C	2	2	237	1
D	1	1	1	44
Total	114	229	241	45

Considering above confusion matrix overall accuracy has been found

Results
TP: 616
Overall Accuracy: 97.93%

Table 5: Accuracy, precision, recall, f1 score in case of non-optimized dataset for 4 classes

Class	N(truth)	N(classified)	Accuracy	Precision	Recall	F1 Score
1	114	112	99.05 %	0.98	0.96	0.97
2	229	228	98.89 %	0.99	0.98	0.98
3	241	242	98.57 %	0.98	0.98	0.98
4	45	47	99.36 %	0.94	0.98	0.96

5.1. Optimization of GRADE A

Hybrid MVO-PSO is applied in order to get the optimized data set for training;

Hybrid MVO-PSO optimization of GRADE A results in

At iteration 50 the best universes fitness is 0.30119
At iteration 100 the best universes fitness is 0.30119
At iteration 150 the best universes fitness is 0.30119
At iteration 200 the best universes fitness is 0.30119
At iteration 250 the best universes fitness is 0.30119
At iteration 300 the best universes fitness is 0.30119
At iteration 350 the best universes fitness is 0.30119
At iteration 400 the best universes fitness is 0.30119
At iteration 450 the best universes fitness is 0.30118
At iteration 500 the best universes fitness is 0.30118

The best solution for class A obtained by MVO is: 4.6807. The best optimal value for class A of the objective function found by MVO is: 0.30118

Elapsed time is 6.498883 seconds.

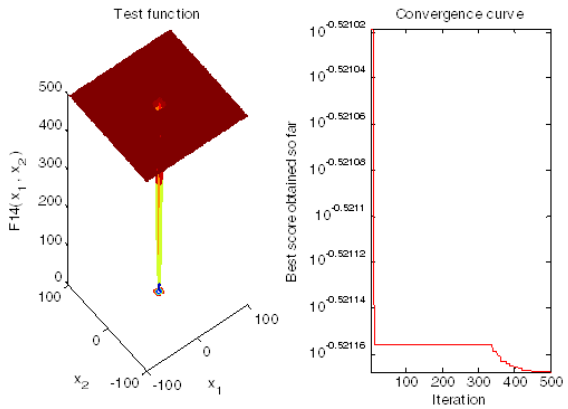

Figure 4: Hybrid optimization of Group A

After optimization 58 components have been selected where average rating is more than 4.6807

5.2. Optimization for Grade B

At iteration 50 the best universes fitness is 0.30119
At iteration 100 the best universes fitness is 0.30119
At iteration 150 the best universes fitness is 0.30119
At iteration 200 the best universes fitness is 0.30119
At iteration 250 the best universes fitness is 0.30119
At iteration 300 the best universes fitness is 0.30119
At iteration 350 the best universes fitness is 0.30119
At iteration 400 the best universes fitness is 0.30119
At iteration 450 the best universes fitness is 0.30119
At iteration 500 the best universes fitness is 0.30118

The best solution for class B obtained by Hybrid MVO-PSO is: 4.2356. The best optimal value for class B of the objective function found by Hybrid MVO-PSO is: 0.30118. Elapsed time is 3.650676 seconds.

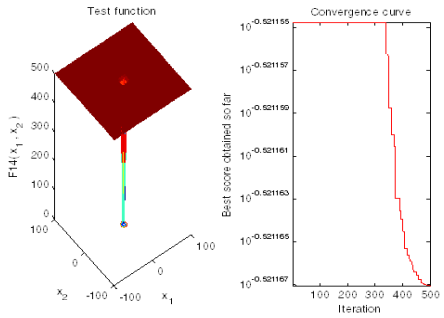

Figure 5: Hybrid MVO-PSO simulation to get optimized value for grade B

After optimization 120 components have been selected where average rating is more than 4.2356

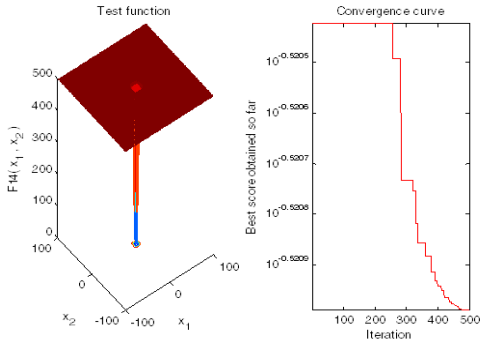

Figure 6: Hybrid MVO-PSO simulations to get optimized value for grade C

5.3. Optimization of Grade C

At iteration 50 the best universes fitness is 0.3017
At iteration 100 the best universes fitness is 0.3017
At iteration 150 the best universes fitness is 0.3017
At iteration 200 the best universes fitness is 0.3017
At iteration 250 the best universes fitness is 0.3017
At iteration 300 the best universes fitness is 0.30149
At iteration 350 the best universes fitness is 0.3014
At iteration 400 the best universes fitness is 0.30134
At iteration 450 the best universes fitness is 0.30132
At iteration 500 the best universes fitness is 0.30131

The best solution for class A obtained by MVO is: 3.6263. The best optimal value for class A of the objective function found by MVO is: 0.30131. Elapsed time is 3.911880 seconds.

After optimization 139 components have been selected where average rating is more than 3.6263

5.4. Optimization of Grade D

At iteration 50 the best universes fitness is 0.30156
At iteration 100 the best universes fitness is 0.30156
At iteration 150 the best universes fitness is 0.30156
At iteration 200 the best universes fitness is 0.30156
At iteration 250 the best universes fitness is 0.30156
At iteration 300 the best universes fitness is 0.30156
At iteration 350 the best universes fitness is 0.30153
At iteration 400 the best universes fitness is 0.30151
At iteration 450 the best universes fitness is 0.3015
At iteration 500 the best universes fitness is 0.30149

The best solution for class A obtained by MVO is: 2.4891. The best optimal value for class A of the objective function found by MVO is: 0.30149. Elapsed time is 2.961058 seconds.

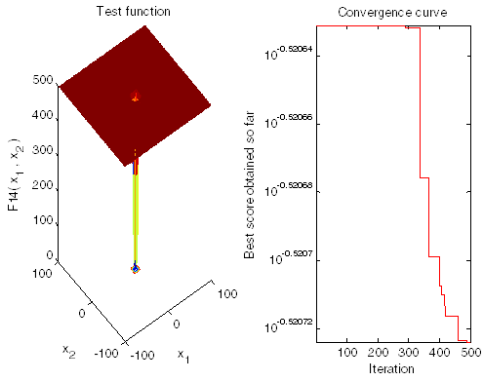

Figure 7: Hybrid MVO-PSO simulations to get optimized value for grade D

After optimization 31 components have been selected where average rating is more than 2.4891. After grade wise optimization the following components would be selected according to grade.

Table 6: Grade wise record count after optimization

GRADE	Component count
A	58
B	120
C	139
D	31
Total	348

But of the model is trained with optimization the confusion matrix is

Table 7: Confusion matrix produced after filter dataset considering optimization value

	A	B	C	D
A	58	0	0	0

B	0	119	1	0
C	0	1	137	0
D	0	0	1	31
Total	58	120	139	31

Considering above confusion matrix overall accuracy has been found

Results
TP: 345
Overall Accuracy: 99.14%

Table 8: Accuracy, precision, recall, f1 score in case of optimized dataset for 4 classes

Class	N(truth)	N(classified)	Accuracy	Precision	Recall	F1 Score
1	58	58	100%	1.0	1.0	1.0
2	120	120	99.43%	0.99	0.99	0.99
3	139	138	99.14%	0.99	0.99	0.99
4	31	32	99.17%	0.97	1.0	0.98

6. Comparative Analysis

This section is comparing the accuracy, precision, recall and F1 score before optimization and after optimization. Comparison of accuracy for previous and optimized data set has been shown in table 9.

Comparison of Accuracy for previous and optimized data set has been shown below

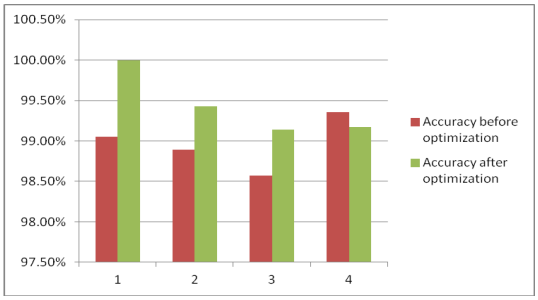

Figure 8: Comparison of accuracy

Comparison of precision for previous and optimized data set has been shown below

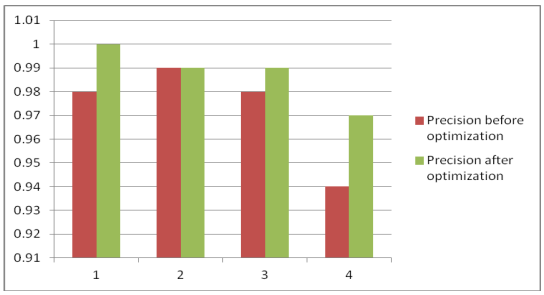

Figure 9: Comparison of precision

Comparison of recall for previous and optimized data set has been shown below

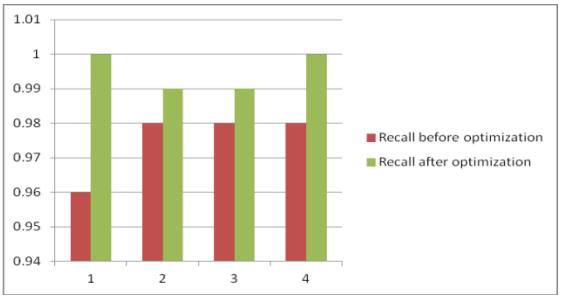

Figure 10: Comparison of recall

Comparison of fscore for previous and optimized data set has been shown below

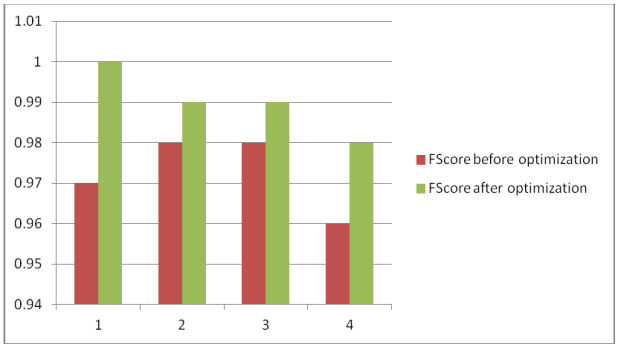

Figure 11: Comparison of Fscore

Table 9: Comparison of Accuracy, Precision, Recall, Fscore

Class	Accuracy before optimization	Accuracy after optimization	Precision before optimization	Precision after optimization	Recall before optimization	Recall after optimization	FScore before optimization	FScore after optimization
1	99.05%	100%	0.98	1.0	0.96	1.0	0.97	1.0
2	98.89%	99.43%	0.99	0.99	0.98	0.99	0.98	0.99
3	98.57%	99.14%	0.98	0.99	0.98	0.99	0.98	0.99
4	99.36%	99.17%	0.94	0.97	0.98	1.0	0.96	0.98

7. Conclusion

It has been concluded from simulation that the optimized dataset is capable to produce more accurate result as compared to non-optimized mechanism. Research has considered training and testing of dataset after Optimizing Software Component by deep neural network mechanism to improve the accuracy at the time of software selection. Deep Neural-Network mechanism has performed training and testing to get the confusion metrics of true

positive/negative and false positive/negative. Training has been performed using LSTM neural network mechanism to produce confusion matrix for getting accuracy, F-score, Recall and precision values. Simulation result concludes that the accuracy, F-score, Recall and precision values in case of optimized mechanism are better than non-optimized mechanism.

Table 10 shows that proposed work is providing high reliability and feasibility as compare to previous research models.

Table 10: Comparison of proposed work to existing researches

	Prediction of quality using ANN based on Teaching-Learning Optimization in component-based software systems [20]	PCI-PSO : Preference-Based Component Identification Using Particle Swarm Optimization [17]	A Hybrid Neuro-Fuzzy and Feature Reduction Model for Classification [29]	Proposed work
Optimization	✓	✓	✗	✓
Use of Neural network	✓	✗	✓	✓
CBSE	✓	✓	✗	✓
Performance	✓	✓	✗	✓
Accuracy	✗	✗	✓	✓
Reliability	✗	✗	✓	✓
Feasibility	✗	✗	✓	✓

8. Scope of Research

Such research could play a significant role in the field of software development, AI, big data processing, and many other fields where prediction is the major objective. Such a mechanism is suitable to provide an efficient and accurate approach to perform forecasting and decision-making in different areas. Moreover, further researches could use this research as a base in order to get more fruitful results.

References

[1] S. Di Martino, F. Ferrucci, C. Gravino, F. Sarro, "A genetic algorithm to configure support vector machines for predicting fault-prone components," Lecture Notes Computer Science (including Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), **6759**, 247–261, 2011, doi: 10.1007/978-3-642-21843-9_20.

[2] M. Palviainen, A. Evesti, E. Ovaska, "The reliability estimation, prediction and measuring of component-based software," Journal of System and Software, **84**(6), 1054–1070, 2011, doi: 10.1016/j.jss.2011.01.048.

[3] A. A. A. Saed, W. M. N. W. Kadir, "Applying particle swarm optimization to software performance prediction an introduction to the approach," in 2011 5th Malaysian Conference in Software Engineering, MySEC 2011, 207–212, 2011, doi: 10.1109/MySEC.2011.6140670.

[4] J. F. Tang, L. F. Mu, C. K. Kwong, X. G. Luo, "An optimization model for software component selection under multiple applications development," Europian Journal of Operational Research, **212**(2), 301–311, 2011, doi: 10.1016/j.ejor.2011.01.045.

[5] U. Sharma, P. Singh, S. S. Kang, "Reusability of Software Components Using SVM Based Classifier Approach" , International Journal of Information Technology and Knowledge Management, **5**(1), 117–122, 2012.

[6] C. Mueller, "Multi-Objective Optimization of Software Architectures Using Ant Colony Optimization," Lecture Notes on Software Engineering, **2**(4), 371–374, 2014, doi: 10.7763/lnse.2014.v2.152.

[7] C. Singh, A. Pratap, A. Singhal, "Estimation of software reusability for component based system using soft computing techniques," in 2014 5th International Conference - Confluence The Next Generation Information Technology Summit, 788–794, 2014, doi: 10.1109/CONFLUENCE.2014.6949307.

[8] K. Tyagi, A. Sharma, "An adaptive neuro fuzzy model for estimating the reliability of component-based software systems," Applied Computing and Informatics, 10(1–2), 38–51, 2014, doi: 10.1016/j.aci.2014.04.002.

[9] M. Tan, C. dos Santos, B. Xiang, B. Zhou, "LSTM-based Deep Learning Models for Non-factoid Answer Selection," 1, 1–11, 2015, Available: http://arxiv.org/abs/1511.04108.

[10] D. Gao, X. Guo, L. Zhao, "Test case prioritization for regression testing based on ant colony optimization," in 2015 6th IEEE International Conference on Software Engineering and Service Science, 275–279, 2015, doi: 10.1109/ICSESS.2015.733905.

[11] G. Kumar, P. K. Bhatia, "Neuro-Fuzzy Model to Estimate & Optimize Quality and Performance of Component Based Software Engineering," ACM SIGSOFT Software Engineering Notes, **40**(2), 1–6, 2015, doi: 10.1145/2735399.2735410.

[12] S. Vodithala, S. Pabboju, "A dynamic approach for retrieval of software components using genetic algorithm," in 2015 6th IEEE International Conference on Software Engineering and Service Science, 406–410, 2015, doi: 10.1109/ICSESS.2015.7339085.

[13] S. Mirjalili, S. M. Mirjalili, A. Hatamlou, "Multi-Verse Optimizer: a nature-inspired algorithm for global optimization," Neural Computing and Application, 27(2), 495–513, 2016, doi: 10.1007/s00521-015-1870-7.

[14] A. Verma, S. Gupta, I. Kaur, "Inconsistency detection in software component source code using ant colony optimization and neural network algorithm," Indian Journal of Science and Technology, **9**(40), 2016, doi: 10.17485/ijst/2016/v9i40/101127.

[15] Z. Zhao, W. Chen, X. Wu, P. C. V. Chen, J. Liu, "LSTM network: A deep learning approach for short-term traffic forecast," IET Image Processing, **11**(1), 68–75, 2017, doi: 10.1049/iet-its.2016.0208.

[16] L. Sun, J. Du, L. Dai, C. Lee, "Multiple-target deep learning for LSTM-RNN based speech enhancement," in 2017 Hands-free Speech Communications Microphone Arrays, doi: 10.1109/HSCMA.2017.7895577.

[17] S. M. H. Hasheminejad, S. Gholamshahi, "PCI-PSO: Preference-based component identification using particle swarm optimization," Journal of Intelligence System, **28**(5), 733–748, 2021, doi: 10.1515/jisys-2017-0244.

[18] A. Gensler, J. Henze, B. Sick, N. Raabe, "Deep Learning for solar power forecasting - An approach using AutoEncoder and LSTM Neural Networks," in 2016 IEEE International Conference on Systems, Man, and Cybernetics, 2858–2865, 2017, doi: 10.1109/SMC.2016.7844673.

[19] O. Bhardwaj, S. Kumar Jha, "Quality assurance through soft computing techniques in component based software," Proceeding 2017 International Conference Smart Technology Smart Nation, SmartTechCon, 277–282, 2018, doi: 10.1109/SmartTechCon.2017.8358382.

[20] P. Tomar, R. Mishra, K. Sheoran, "Prediction of quality using ANN based on Teaching-Learning Optimization in component-based software systems," Software- Practice Experience., **48**(4), 896–910, 2018, doi: 10.1002/spe.2562.

[21] L. Mu and C. K. Kwong, "A multi-objective optimization model of component selection in enterprise information system integration," Computer Industrial Engineering, **115**, 278–289, 2018, doi: 10.1016/j.cie.2017.11.013.

[22] P. Chatzipetrou, E. Alégroth, E. Papatheocharous, M. Borg, T. Gorschek, and K. Wnuk, "Component selection in software engineering - Which attributes are the most important in the decision process?," in 44th Euromicro Conference on Software Engineering and Advanced Applications, 198–205, 2018, doi: 10.1109/SEAA.2018.00039.

[23] C. Diwaker, P. Tomar, R. C. Poonia, V. Singh, "Prediction of Software Reliability using Bio Inspired Soft Computing Techniques," Journal of Medical Systems, **42**(5), 2018, doi: 10.1007/s10916-018-0952-3.

[24] S. Gholamshahi, S. M. H. Hasheminejad, "Software component identification and selection: A research review," Software - Practice and Experience, **49**(1), 40–69, 2019, doi: 10.1002/spe.2656.

[25] C. Diwaker, "A New Model for Predicting Component-Based Software Reliability Using Soft Computing," IEEE Access, **7**, 147191–147203, 2019, doi: 10.1109/ACCESS.2019.2946862.

[26] H. Hu, Y. Li, Y. Bai, J. Zhang, M. Liu, "The Improved Antlion Optimizer and Artificial Neural Network for Chinese Influenza Prediction," Complexity, 2019, doi: 10.1155/2019/1480392.
[27] A. L. Imoize, D. Idowu, Imoize, Agbotiname Lucky, T. Bolaji, "A Brief Overview of Software Reuse and Metrics in Software Engineering," An International Scientific Journal, **122**(2), 56–70, 2019.
[28] G. Maheswari, K. Chitra, "Enhancing reusability and measuring performance merits of software component using data mining," International. Journal of Innovative Technology Exploring Engineering, **8**(6), 1577–1583, 2019, doi: 10.35940/ijitee.F1318.0486S419.
[29] H. Das, B. Naik, H. S. Behera, "A Hybrid Neuro-Fuzzy and Feature Reduction Model for Classification," Advances in Fuzzy System, 2020, doi: 10.1155/2020/4152049.