FP-International Journal of Computer Science Research (IJCSR)
Volume 2, Issue 1, Pages 45-50, March 2015

Development and Analysis of Software technique in CBSE environment using Feed Forward Neural Network

Prof. (Dr.) Amit verma
Head, Department of Computer Science
Chandigarh University, Punjab
amitverma@cumail.in

Pardeep kaur
ME student
Department of Computer Science
Chandigarh University, Punjab
Pardeepdharni664@gmail.com

ABSTRACT
In software development, component reusability seems to be one important factor. This factor directly or indirectly influences the software quality and production and decreases time complexity and saves cost. Basically Component reusability approach is to reuse any programming unit, code, utility functions etc. It also avoids the development from scratch. In this paper, we describe various algorithms and techniques to check the optimized reusable components from list of components. For this scenario, clustering and repository are two important factors. Software developers have to store their relevant components in repository so that they can pick valuable component at right time. For clustering process, algorithms like k-means and k-mode are used. In proposed work, input data collected from software's functional attributes with help of metrics is feed up to neural network's algorithms, to check the optimize reusable component.

Keywords
Component based software engineering, Reuse and Feed forward neural network

1. INTRODUCTION

During timing of 20th century; new technologies have been coming up. As the no. of peoples grows, their need increases, but as needs or desires increased, same they have a short of timings. So in today's time everyone wants everything in less time or before time. So there must be a need of improving methodologies and find out new technologies. Today's in 21th century is world of automotive technologies. So, organizations follow these strategies, developing work products or software according to user by following a step by step method.

During the past forty years, various approaches for software development come into existence. From the last few years, the basic approach or method used by developer is to separate the software into phases and work according to those phases, so that they can concentrate only one phase at a time. The first software development approach after software crisis comes into existence in 1970s. On the basis of different needs of customers and organization's targets, developers refer different approaches for development. Like use different programming languages for development, Fortran 1950s and object oriented programming languages in 1960s. With the increasing competition among various software organizations, it has become very important to optimize processes of organizations, so as to reduce the time to market for the software release .But along with the time to market; the quality has become the most important factor for the success of any software. As software process efficient, it leads better results and increase market of particular organization. The software process related data can be used by the companies for making the future software's, from this situation component based software engineering term come into use.

The next section of this paper precisely describes the introduction about concepts related to reuse component in software development and neural network that how neural helps for make a work automatic and error free while reuse a component. Section 2 provides related work about reuse concept with help of neural network, section 3 provides detail descriptions of algorithms for trained neural network section 4 presents challenges exist in this work, and Section 5 presents overall conclusions.

1.1 Software Development
In the last decades, for the development of software, various approaches come under use. Methods or methodologies must be decided according to the size or importance of software. The first approach of software development exists in 1970s.On that time there was a process followed for separation of software into phases. So that they can develop software within deadlines and can focus on each aspect of software. In the process of separation of phases, software is divided into modules and different work strategies performed on that modulus until it fulfills the user demands.

Evolution of software engineering [17]

1990-1999: This time is, called prominence of internet. From that time internet rise at very fast growth. Programmers were needed to handle maps, photographs, and other images. The growth of browser usage changed, as information display and retrieval methods are changed. As time passing, Computer users

also increased from hundreds or thousands to many millions of international users.

2000 to present: With growth of people's needs, technologies must be need for upgraded and along that demand of software also increased in small organizations. There was need for developing software solutions that must help in simpler, faster methodologies by inventing fast and quick solutions. The use of methodologies comes into use like rapid prototyping and extreme prototyping, which works to simplify many areas like requirements gathering and reliability testing and in today's large software systems, used various documented methodologies.

1.2 Component Based Software engineering

There are number of definitions exist for software components. But in context of CBSE [20] we can say that it is plug and play technique, process of integration and can work with other hardware without compilation. Software components can be some code, utility functions or any programming unit. Components can be product specific, domain specific [19] and domain independent.

Component Based Software Engineering is now widely used in component based development. In software development various challenges exist like software complexity, delivery deadlines, budget etc. It takes large time to implement software from scratch. On that time component based development must be very helpful. Component is independent part perform complete functionalities. Component Based software engineering is used to develop or assemble software from existing components.

1.3 Neural network

Neural networks applications in various areas including functional approximation, nonlinear system identification and control, pattern recognition and pattern classification optimization , English text pronunciation, protein secondary structure prediction and speech recognition.
For neural network training and designing, following sub steps are followed:

• Load data

• Divide data into training, validation and test data

• Calculate the minimum and maximum values in the attribute of input and setting parameters of feed forward neural network like;

- Size of neural network
- Type of transfer function of each layer to be used
- Type of learning function

• Generate neural network

• Perform the training of neural network using training dataset Compare the results of actual output with target values and if not matched, adjust the weights to produce results closer to target

• Network weights are set by adding the error correction value into old weights and biased values and training is stopped while;

- Maximum no. of epochs reached
- Maximum amount of time exceed
- Performance is minimized to goal
- Performance gradient falls below minimum value set

• Validation performance has increased more than maximum fails times since the last time it decreased.

2. RELATED WORK

A research on component based software engineering is going on, brief literature review of work done in the field of Component Based Software Engineering in software development for component reusability with help of neural network is discussed here:

Dr. Winston W. Royce [1] proposed solution for managing the development of large scale software systems. Author tells five important features that are necessary to followed by organizations while development like I) complete the program design before analysis and coding begins ii) Documentation must be current and complete iii) Do the job twice if possible IV) Testing must be planned, controlled and monitored v) Involve the customer. Daniel Svozil et.al. [3] Discusses about the neural network and feed forward neural network that how activation functions perform calculations and compare output with target value and also explain about back propagation of error. He also explains about generalization and training factors of neural network. In this paper, also find the partial derivates for object function with respect to weight and threshold coefficients. Takashi Kobayashi et. al. [4] discusses technique to model pattern based development. It is a method for developing artifacts from artifacts that were produced in previous projects. Pattern structure consists class diagram and object diagram. Two manipulation operations are performed on these pattern structures: pattern instantiation and pattern evolution. Mark Crowne [5] this paper explores the critical product development issues that can lead to failure. The evolution of product development consists three phases: I) startup II) Stabilization III) Growth. For each phase symptoms, issues and solution are analyzed.

Richard W. Selby [6] in this paper basically provides the way of doing reuse software in large scale systems. Four ways defined for checking the actual reuse: Modules without revision, Modules with slight revision(less than 25%), Modules with major revision (more than 25%), and newly developed modules, their

main focus was to achieve 32 percent reuse per project. Suresh Chand Gupta [13] proposed a method for storage and retrieval of components for reuse purpose based on UML diagram, Metadata repository and neural network and one thousand components obtained by using reusable techniques then for better optimizing the search results, neural network is used. Gaurav kumar et. al. [17] discusses optimization of component based software engineering model using neural network. In this paper, present a model by applying CK metrics on component design patterns and values drawn from them and then used unsupervised neural network for check the optimization of reusability of those patterns.

3. Algorithms for trained a neural network for check the optimization of reuse components

3.1 Back propagation algorithm

Back propagation is not just a learning algorithm, it also provides us information that how network behavior is changed by change occurs in weights and biased. In the processing of neurons, if resulted output is not equal to target value, then error back propagates to previous iteration.

Algo_Back propagation

BEGIN
Input:ProblemSize,inputPatterns,Iterationsmx, learn rate
Output: Network
Network ← constructNetworkLayer( )
Networkweight ← initializeWeights(network,ProblemSize)
For (i=1 to iterationsMax)
$Pattern_i$= selectInputPattern(InputPatterns)
$Output_i$ = ForwardPattern($Pattern_i$, Network)
BackPropagateError($Pattern_i$,$Output_i$,Network)
UpdateWeights(Patterni,$Output_i$,Network,Learnrate)
End
Return(Network)

3.2 Particle swarm optimization

PSO is population based stochastic optimization technique, helpful in function optimization, artificial neural network training, Fuzzy control system etc. In every iteration, each particle is update by two values. The first one is fitness value, called pbest value. Another best value is calculated by any particular swarm organizer, best value is globally best called gbest.

The two eqns; v[]=v[]+

c1*rand ()*(pbest []-present[]) +

c2*rand ()*(gbest []-present[]) eq…. (3.1)

present[] =present[] + v[] eq…… (3.2)

Where v[] is the particle velocity and present[] is current particle (solution).The function name rand() is random number between(0,1) and c1, c2 are learning factors.

Algo_PSO
BEGIN For each particle Initialize particle END Do For each particle Calculate fitness value If the fitness value is better than the best fitness value (pbest) in history set current value as the new pbest End Choose the particle with the best fitness value of all the particles as the gbest For each particle Calculate particle velocity according Eq…. 3.1 (explained above) Update particle position according Eq…. 3.2 (explained above) End

3.3 Conjugate Gradient (CG) algorithm:

CG is fast multilayer algorithm used for solving non linear functions. The proposed algorithm basically improves the efficiency of back propagation algorithm, helps in search directions. The results show that it requires less iteration than the both standard CG and neural network algorithms.

Algo_CG based neural network
BEGIN Select the initial weights randomly with small values and set p=1 Select the next training pair from training set [Tp] applies the input vector [Xp] to the network input and specify the desired output vector. Then calculates the desired output of vector by using these two formulas: $net^{\delta+1}_{pj}=\sum W^{\delta}_{ij}\, out^{\delta}_{i} + biasj^{\delta+1}$

i=1

$out^{\delta+1}_{pj} = f(net_{pj}^{\delta+1}) = 1/(1+e-\beta outpj^{\delta+1\wedge})$

where,

$net^{\delta}+1_{pj}$= Summation of multiple weight with inputs for j

$out^{\delta}+1_{pj}$= The actual output of the network j by using function

$x_j = out_{pj}^{\phi}$ = It represents input patterns, p number of pattern and j is number of net

$net_j = X1\ W1_j + X2\ W2_j + \ldots + Xn\ W_{nj} = \sum^{n}_{i=1} Xi\ Wij$

and;

net_j= The result of summation of j node

X1= Input pattern

$W1_j$= Weight between node j and i

Calculate the errors between the actual output of the network and the desired output

END

3.3.1 Fletcher–Reeves Update Conjugate Gradient (FRUCG) algorithm:

This algorithm find the ratio of the norm squared of current gradient to the norm squared of previous gradient. It is basically used for line search direction, to locate the minimum point. The first search direction is negative of the gradient of performance and succeeding iterations computed from new and previous gradient.

For each iteration, algorithm train any network with net inputs, its weights and transfer functions with derivative functions

Back propagation is used to calculate derivatives of performance of weight and biased with respect to variable X

Algo_FRUCG

BEGIN

Variable X is calculated by;

X=X+a*dX eq…..(1.1)

Where, dX is search direction, parameter a is used to minimize the performance of search direction

Search direction is calculated by:

dX= -gX+dX_old*Z eq…..(1.2)

where, gX is gradient and according to Fletcher-Reeves, variable Z is calculated by;

Z= (normnew_sqr)/(norm_sqr)

Where, norm_sqr is the norm square is the previous gradient and normnew_sqr is the norm square of current gradient.

END

3.3.2 Polak- Ribiere Update Conjugate gradient (PRUCG) algorithm

This version of conjugate gradient algorithm was proposed by Polak and Ribiere, helps in local search direction to locate the minimum point,

Algo_PRUCG

BEGIN

Variable X is calculated by;

Eqn. 1.1 (explained above)

Search direction is calculated by:

Eqn. 1.2 (explained above)

According to Polak- Ribiere, variable Z is calculated by:

Z= (((gX-gX_old)*gX))/(norm_sqr)

Where, norm_sqr is the norm square of previous gradient, and gX_old is gradient on previous iteration

END

3.3.3 Powell-Beale Restarts Conjugate gradient (PBRCG) Algorithm

The all others conjugate gradient algorithm works on reset on negative of gradient. The standard reset point occurs when the no. of iterations is equal to parameters, weight and biases. But this reset method proposed by Powell, can improve the efficiency of network. For check the reset of negative of gradient, based on two factors; first, the algorithm uses a test to determine when to reset the search direction to the negative of gradient. Second, search direction is computed from negative gradient, the previous search direction and last search direction before previous reset direction.

Algo_PBRCG

BEGIN

Check

If there is any orthogonality left between previous gradient and current gradient and this is tested by:

$$gkT\text{-}1gk \geq 0.2 \| gk \|2$$

If this condition is satisfied, then search direction is reset to negative of gradient.

END

For each iteration, algorithm train any network with net inputs, its weights and transfer functions with derivative functions

Back propagation is used to calculate derivatives of performance of weight and biased with respect to variable X

Variable X is calculated by;

Eqn. 1.1 (explained above)

Search direction is calculated by:

Eqn. 1.2 (explained above)

According to Powell-Beale, variable Z is calculated based on two factors, already discussed

3.3.4 Scaled conjugate gradient (SCG) algorithm

The all other parts of conjugate gradient mostly work on line search direction at every iteration. The line search method is quite expensive, because it requires to network response to all training inputs computed for each search. [2] SCG algorithm basically used to avoid time consuming search, developed by Moller. It combines the model trust region and conjugant gradient. Following are the steps for working with Scaled conjugate gradient:

For each iteration, algorithm train any network with net inputs, its weights and transfer functions with derivative functions

Back propagation is used to calculate derivatives of performance of weight and biased with respect to variable X

Variable X is calculated by;

Eqn. 1.1 (explained above)

Scaled conjugate gradient algorithm is based on conjugate directions; no any search direction method is followed in every iteration.

It requires less storage requirements, can work on less parameter and also work on conjugate direction rather than line search direction.

4. CHALLENGES

In software development environment, component based development is challenge in itself. Another big challenge, todays is component identification, means to select right component at right time and there is also need to make the process automatic and error free. So, there must be need of an algorithm that helps in achieve the accuracy and optimization of reuse component, better than already discussed algorithms.

5. CONCLUSION

In this paper, provides the detail about reusability concept in software development and for make the process better and less complex, integrate with feed forward neural network. We also evaluate some of algorithms, like back propagation, Particle swarm optimization, and various conjugate algorithms like, Fletcher-Reeves, Polak-Ribiere, Powell-Beale and Scaled conjugate algorithms to check the optimization and accuracy of reusable component. On the basis of accuracy or compatibility with feed forward neural network, scaled conjugate gradient algorithm can provide better results and second best performer can be Fletcher-Reeves algorithm.

6. REFERENCES

[1] Dr. Winston W. Royce. “Managing the development of large software systems”, pp. 328-338, 1970, IEEE.

[2] Martin Fodslette Mollar. “A scaled conjugate gradient algorithm for fast supervised learning”, University of Aarhus, Vol.6, pp. 525-533, 1993.

[3] Daniel Svozil, Vladimir Kvasnieka, Jie Pospichal. “Introduction to multi-layer feed-forward neural networks”, pp. 43-62, 1997, ELSEVIER.

[4] Takashi kobayashi, Motoshi Saeki. ”Software Development Based on Software Pattern Evolution”, pp. 18-25, 1999, IEEE.

[5] Mark Crowne. “Why software product startups fail and what to do about it”, 2002, pp. 338-343, IEEE.

[6] Richard W. Selby.”Enabling Reuse-Based Software Development of Large-Scale Systems “, pp. 495-510, 2005, IEEE.

[7] Venus Marza, Amin Seyyedi, and Luiz Fernando Capretz. “Estimating Development Time of Software Projects Using a Neuro Fuzzy Approach”, World Academy of Science, Engineering and Technology, pp. 575-579, 2008.

[8] Gholam Reza Shahmohammadia, Saeed Jalili, Seyed Mohammad Hossein Hasheminejad.” Identification of System Software Components Using Clustering Approach”, Journal of object technology, vol. 9, no. 6, pp. 77-98, 2010.

[9] Arvinder Kaur, Kulvinder Singh Mann. “Component Selection for Component based Software Engineering”, International Journal of Computer Applications, Volume 2 –No.1, 2010.

[10] Sonia Manhas, Parvinder S. Sandhu, Viand Chopra, Nirvair Neeru. “Identification of Reusable Software Modules in Function Oriented Software Systems using Neural Network

Based Technique", World Academy of Science, Engineering and Technology Vol: 4, 2010.

[11] N Md Jubair Basha and Dr Chandra Mohan. "A strategy to identify components using clustering approach for component reusability", Computer Science & Information Technology, pp. 397–406, 2012.

[12] Anupama Kaur. "Impact of Training Function Based Neural Network on Reusable Software Modules", International Journal of Computer Science and Information Technologies, Vol. 3, pp. 4024 – 4027, 2012.

[13] Suresh Chand Gupta, Prof. Ashok Kumar. "A Neural Network based Method to Optimize the Software Component Searching Results in K-Model", International Journal of Computer Applications, Volume 72– No.7, pp. 20-27, 2013.

[14] Sandeep Kumar Jain & Manu Pratap Singh. " Estimation for Faults Prediction from Component Based Software Design using Feed Forward Neural Networks", International Journal of Advanced Research in Computer and Communication Engineering ,Vol. 2, Issue 7, 2013.

[15] Nitish Madaan and Jagdeep Kaur. "A Survey on Selection Techniques of Component Based Software", International Journal of Information & Computation Technology. Volume 4, pp. 1245-1250, 2013.

[16] Shital Solanki, H.B.Jethva. "Modified Back Propagation Algorithm of Feed Forward Networks", International Journal of Innovative Technology and Exploring Engineering, Volume-2, Issue-6, pp. 131-134, 2013.

[17] Gaurav kumar and Pradeep kumar bhatia. "Optimization of Component Based Software Engineering Model Using Neural Network", International Journal of Information Technology, Vol. 6 No. 2, pp. 732-742, 2014.

[18] Mahsa Hasani Sadi, Eric Yu. "Analyzing the Evolution of Software Development: From Creative Chaos to Software Ecosystems", pp. 1-11, 2014.

[19] Adnan Khan, Khalid Khan, Muhammad Amir and M. N. A. Khan. "A Component-Based Framework for Software Reusability", International Journal of Software Engineering and Its Applications Vol. 8, No. 10, pp. 13-24, 2014.

[20] Mikio Aoyama," New Age of Software Development: How Component- Based Software Engineering Changes the Way of Software Development?" Department of Information and Electronics Engineering, pp.1-5.

[21] Constantinos Stylianou and Andreas S. Andreou," Software Component Clustering and Retrieval: An Entropy-based Fuzzy k-Modes Methodology, pp. 399-422.