ISSN 2047-3338

Classification of Cognitive Load Identification for User's Preference in Web Learning System using Particle Swarm Optimization Techniques

L. Jayasimman[1] and E. George Dharma Prakash Raj[2]

[1]Department of Computer Application, J.J. College of Engineering and Technology, Trichy, India
[2]School of Computer Science and Engineering, Bharathidasan University, Trichy, India

***Abstract*— Web based learning systems is a powerful learning environment which provides more benefits to the users compared to traditional learning systems. Web learning systems have effective methods, and adapt a personalized approach based on factors like preferences and emotions. This study emphasizes the system associated with cognitive aspects. Web learning users need innovative methods in the learning system to ensure motivated learning. Web learning system also emphasizes user's cognitive characteristics, experience and demands and so its design requires a thorough understanding of learner's activity to improve the user's cognitive approach based the user's requirements. In this paper user's preferences of web learning system with cognitive load are classified using neural network. To improve the classification accuracy particle swarm optimization is used to optimize the neural network parameters.**

***Index Terms*— Web Learning, Cognitive Load, Multi Layer Perceptron (MLP) and Particle Swarm Optimization Classification Accuracy**

I. INTRODUCTION

COGNITIVE Load Theory (CLT) is gaining increasing importance in the design and evaluation of instruction, both traditional and technology based. Although it is well understood as a theoretical construct, the measurement of cognitive load induced by instructional trials in general, and by multimedia instruction in particular, mainly relies on methods that are either indirect, subjective, or both. The basic idea of cognitive load theory is that cognitive capacity in working memory is limited, so that if a learning task requires too much capacity, learning will be hampered. The recommended remedy is to design instructional systems that optimize the use of working memory capacity and avoid cognitive overload [1].

Based on different sources for cognitive load, Sweller [2] distinguished three types of load. Intrinsic cognitive load relates to the difficulty of the subject matter [3], [4]. Extraneous cognitive load is cognitive load that is evoked by the instructional material and that does not directly contribute to learning (schema construction). Cognitive load theory sees the construction and subsequent automation of schemas as the main goal of learning [5]. The construction of schemas involves processes such as interpreting, exemplifying, classifying, inferring, differentiating, and organizing. The load that is imposed by these processes is denominated germane cognitive load [6].

In many studies there is no direct measurement of cognitive load; the level of cognitive load is induced from results on knowledge post-tests [7]. Three different groups of techniques are used to measure cognitive load, self-ratings through questionnaires, physiological measures (e.g., Heart rate variability, fMRI), and secondary tasks [8]. The user preferences of web learning system with the cognitive load can be identified through questionnaires. Then these preferences are classified using neural network classifier.

Classification is a data mining technique that assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data. Neural networks have emerged as an important tool for classification. The recent vast research activities in neural classification have established that neural networks are a promising alternative to various conventional classification methods [9]. The advantage of neural networks lies in the following theoretical aspects. First, neural networks are data driven self-adaptive methods in that they can adjust themselves to the data without any explicit specification of the functional or distributional form of the model. Second, they are universal functional approximators in that neural networks can approximate any function with arbitrary accuracy.

The relevance in optimization technique will lead to the Particle Swarm Optimization (PSO) is another evolutionary computation technique developed by Eberhart and Kennedy in 1995, which was inspired by the social behavior of bird flocking and fish schooling [10]. PSO based classifier will give the best result for classification. PSO based classification model can be used in various domain where classification of multidirectional real dataset (cancer data, diabetics data) is required. Also this model can be used in many KDD based application like feature selection and clustering [11]. The use of PSO in the design of ANN improves the classification Accuracy.

This paper is organized into the following sections. Section II briefly describes the related works, section III discusses the methodology and genetic algorithm, section IV describes the results obtained and discusses the same. Finally the section V concludes the paper.

II. RELATED WORKS

Jan L. Plass [12] proposed a hybrid model that combines cognitive and software engineering approaches regarding the criteria for the design and evaluation of the user interface of foreign language multimedia software. The proposed approach involves a three step design which includes selection of instructional activity that supports cognitive processes of competence, selection of feature attributes and selection of designs features. It is still pragmatic to be practical. Based on this proposal, contextualized model of interface design, domain specific evaluation criteria are developed to describe how well the user interface is able to support the cognitive processes involved in the development of linguistic and pragmatic skills and competencies in SLA.

Mihalca, et al [13] used a cognitive load framework to examine the role of learner control on performance and instructional efficiency using a genetics training program. In their study comparing three types of instruction (i.e., non-adaptive program control, adaptive program control, and learner control), they predicted that adaptive control would be more effective than both other groups as it better met the needs of learners than program control and was less load bearing than learner controlled environments. While there is some evidence that adaptive control was effective in terms of instructional efficiency the results did not generalize to test-performance measures (near or far transfer). While the study showed considerable promise for embedding adaptive program control into technology based instruction.

Baylari et al [14] proposed a personalized multi agent e-learning system based on item response theory (IRT) and artificial neural network (ANN) which presents adaptive tests (based on IRT) and personalized recommendations (based on ANN). These agents add Adaptivity and interactivity to the learning environment and act as a human instructor which guides the learners in a friendly and personalized teaching environment. The framework constructs adaptive tests that will be used as a post-test in the system. Thus a multi-agent system is proposed which has the capability of estimating the learners' ability based on explicit responses on these tests and presents him/her a personalized and adaptive test based on that ability. Also the system can discover learner's learning problems via learner responses on review tests by using an artificial neural network (ANN) approach and then recommends appropriate learning materials to the student. Experimental results showed that the proposed system can provide personalized and appropriate course material recommendations with the precision of 83.3%, adaptive tests based on learner's ability, and therefore, can accelerate learning efficiency and effectiveness.

III. METHODOLOGY

A. Neural Network

Neural networks are made up of multiple layers of computational units, usually interconnected with each other based on the design of the network [14]. The inputs are fed on the input layer and propagated through the layers to get the output. Output signal is computed using weights, bias and activation function. The propagation rule is used to train the network by back propagating the errors and changing the weights of nodes. The difference between the output obtained and the desired output is the error. A MLP is one of the most frequently utilized neural network techniques for classification and prediction [15].

MLPs often use the back-propagation algorithm for training, and can require large training times especially for large networks, but there are many other types of ANNs. Once the network is trained for a particular problem, however, it can produce results in a very short time. Parallelization of MLP could drastically reduce the training time [16].

In the proposed Parallel neural network, the network is divided into blocks of adjacent neurons and each is allocated to separate processing. For simplicity, it is assumed that these blocks are non-overlapping and rectangular. This approach to parallelization attempts to take advantage of the locality that exists between adjacent neurons. Gaussian and sigmoid activation functions are used in the proposed network.

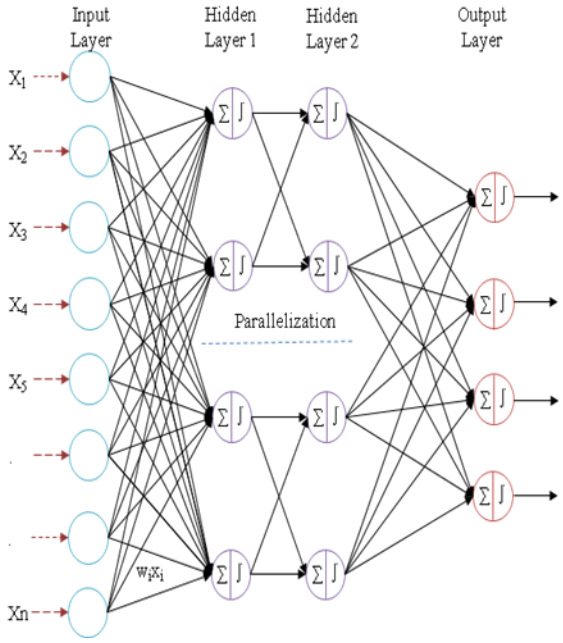

Fig. 1: A typical Parallel Neural Network Model

The net input to y_k to the output layer is computed by

$$y(in)_k = w_{0k} + \sum_i x_i w_{ik}$$

The output is given by

$$y_k = f(y(in)_k)$$

Each output unit $y_k \left(k = 1 \text{ to } m\right)$ whose target is t_k, the error correction is given by

$$\delta_k = (t_k - y_k) f'(y(in)_k)$$

Based on the error obtained, the weights and bias are updated such that

$\Delta\omega_{ik} = \alpha\delta_k n_j$

$\Delta\omega_{0k} = \alpha\delta_k$

δ_k is sent to all the hidden layers and

x_i = *input of neuron i at input layer*

t_i = *target output vector* $(t_1, \ldots..., t_k, \ldots..., t_m)$

α = *learning rate*

w_{0j} = *bias on* j^{th} *hidden unit*

w_{ij} = *weight on* i^{th} *neuron at hidden layer j*

n_{ij} = *neuron i in* j^{th} *hidden layer*

Each output unit updates the bias and weights :

$$\omega_{jk}(new) = \omega_{jk}(old) + \Delta\omega_{jk}$$

$$\omega_{0k}(new) = \omega_{0k}(old) + \Delta\omega_{0k}$$

Activation function of sigmoid function is given as follows:

$$g(x) = \frac{1-e^{-x}}{1+e^{-x}}$$

Gaussian activation function:

$$\phi(v_i) = \exp\left(-\frac{\|v_i - c_i\|^2}{2\sigma^2}\right)$$

The above process is continued for the specified number of epochs or when the actual output equals the target output. The learning rate α affects the convergence of the network. A larger value of α may speed up the convergence but might result in overshooting, while a smaller value of α has vice-versa effect. The range generally used is from 0.001 to 10. Thus, a large learning rate leads to rapid learning but there is oscillation of weights, while the lower learning rate leads to slower learning. The gradient descent is very slow if the learning rate α is small and oscillates widely if α is too large. One very efficient and commonly used method that allows a larger learning rate without oscillations is by adding a momentum factor to the normal gradient descent method.

The momentum factor is denoted by $\eta \in [0,1]$ and the value of 0.9 is often used for the momentum factor. Also, this approach is more useful when some training data are very different from the majority of data. A momentum factor can be used with either pattern by pattern updating or batch-mode updating. In case of batch mode, it has the effect of complete averaging over the patterns. Even though the averaging is only partial in the pattern-by-pattern mode, it leaves some useful information for weight updating.

The weight updation formulas used here are:

$$\omega_{jk}(t+1) = \omega_{jk}(t) + \underbrace{\alpha\delta_k z_j + \eta\left[\omega_{jk}(t) - \omega_{jk}(t-1)\right]}_{\Delta\omega_{jk}(t+1)}$$

and

$$w_{ij}(t+1) = w_{ij}(t) + \underbrace{\alpha\delta_j x_i + \eta\left[w_{ij}(t) - w_{ij}(t-1)\right]}_{\Delta w_{ij}(t+1)}$$

The momentum factor also helps in faster convergence.

B. Particle Swarm Optimization (PSO)

PSO is another evolutionary computation technique developed by Eberhart and Kennedy in 1995, which was inspired by the social behavior of bird flocking and fish schooling. PSO has its roots in artificial life and social psychology, as well as in engineering and computer science. It utilizes a "population" of particles that fly through the problem hyperspace with given velocities. At each iteration, the velocities of the individual particles are stochastically adjusted according to the historical best position for the particle itself and the neighborhood best position. Both the particle best and the neighborhood best are derived according to user defined fitness function [10]. The movement of each particle naturally evolves to an optimal or near-optimal solution. It can, therefore, be effectively applied to different optimization problems in power systems. Moreover, PSO has some advantages over other similar optimization techniques such as Genetic Algorithm (GA), namely the following.

1) PSO is easier to implement and there are fewer parameters to adjust.

2) In PSO, every particle remembers its own previous best value as well as the neighborhood best; therefore, it has a more effective memory capability than the GA.

3) PSO is more efficient in maintaining the diversity of the swarm [17].

A differential evolution operator has been proposed to improve the performance of the PSO algorithm in two different ways:

1) It can be applied to the particle's best position to eliminate the particles falling into local minima.

2) It can be used to find the optimal parameters (inertia and acceleration constants) for the canonical PSO (composite PSO) [18].

To begin with a random solutions set or a set of particles are considered. Random velocity is provided to each particle and they fly through problem space. Each particle's memory keeps track of previous best position and corresponding fitness. Each individual's best position value is stored as 'pid'. In other words, 'pid' is best position acquired by individual particle during its movement within a swarm. It has a value called 'pgd', which is best value of all particles 'pid' in the swarm. PSO's basic concept lies in accelerating each particle to its 'pid' and 'pgd' locations [19].

Velocity of the ith particle of d dimension is given by:

$$v_{id}(k+1) = w \times v_{id}(k) + c_1 \times rand_1 \times (p_{id}(k) - x_{id}(k)) + c_2 \times rand_2 \times (p_{gd}(k) - x_{id}(k))$$

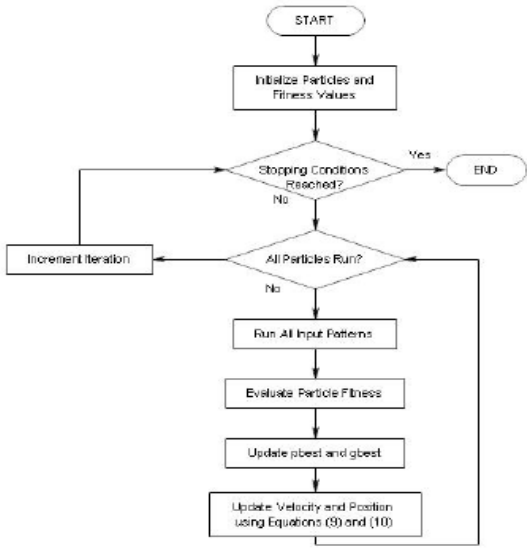

Fig. 1: Flowchart for PSO

The position vector of the ith particle of d dimension is updated as follows:

$$x_{id}(k+1) = x_{id}(k) + v_{id}(k+1)$$

IV. EXPERIMENTAL RESULTS

The cognitive behaviour of 100 students studying in undergraduate and postgraduate courses was captured using questionnaires. They were initially subjected to go through a known subject and an unknown subject in a popular online learning website. Typical questions were in the areas of

- Learning ability
- Indication about meaningfulness of error messages
- Preference to read text rather than to listen to a lecture
- Visualization of content read as a mental picture

Typical questions in the questionnaire are as follows:

1. I prefer content that is challenging so I can learn new things.
2. Compared with other websites this website is better in terms of content.
3. I am so nervous during the online test that I cannot remember facts I have learned
4. I often choose advanced concept links even if they require more work
5. I am sure I can do an excellent job on the problems and tasks assigned for this session
6. I think I will receive a good grade in this class
7. Even when I do poorly on a test I try to learn from my mistakes
8. I think that what I am learning in this class is useful for me to know
9. I think that what we are learning in this website is interesting
10. Understanding this subject is important to me

Table 1: Neural network parameters

Input Neuron	15
Output Neuron	3
Number of Hidden Layer	2
Number of processing elements – upper	4
Number of processing elements – lower	4
Transfer function of hidden layer – upper	Gaussian
Transfer function of hidden layer – lower	Sigmoid
Learning Rule of hidden layer	Momentum
Step size	0.1
Momentum	0.7
Transfer function of output layer	Sigmoid
Learning Rule of output layer	Momentum
Step size	0.1
Momentum	0.7
Learning Rate	0.2
Number of Iterations	1000

Table 2 and Fig. 3 show the classification accuracy of cognitive load for different algorithms. It is observed that the proposed PSO MLP performs better than all the other algorithms.

Table 2: Classification Accuracy for Cognitive Load

Neural Network Algorithms	Cognitive Load
MLP	72%
Parallel MLP	76%
Proposed GO PMLP	85%
Proposed PSO PMLP	94%

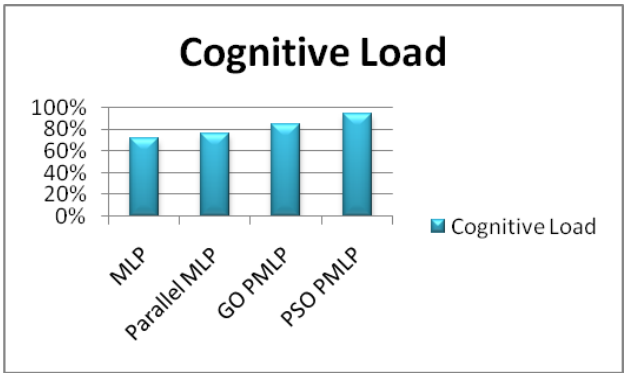

Fig. 3: Classification Accuracy for Cognitive Load

V. CONCLUSION

The classification accuracy for classifying the user preferences of web learning system is enhanced using proposed PSO PMLP algorithm. The user preferences of web learning system with cognitive load are identified using questionnaires. This study considers a new method to classify the users need based on his/her cognitive behavior. The user preferences are classified using PMLP and the user choices for questionnaires are the dataset for the neural network. PSO is another evolutionary computation technique. The parameters of the PMLP are optimized using PSO which brought the significant improvement in the classification accuracy. Finally in comparison, PSO PMLP algorithm has shown an improvement over the other existing MLP algorithms.

REFERENCES

[1]. Ton de Jong, "Cognitive load theory, educational research, and instructional design: some food for thought", Instructional Science, Volume 38, pp. 105-134, 2010.

[2]. Sweller, J. "Instructional design in technical areas", Camberwell, Australia: ACER Press, 1999.

[3]. Cooper contemporary Educational Psychology, volume 2, pp. 1-21, 1998.

[4]. Sweller, J., and Chandler, P., "Why some material is difficult to learn", Cognition and Instruction, volume 12, pp. 185–233, 1994.

[5]. Sweller, J., van Merriënboer, J. J. G., and Paas, F., "Cognitive architecture and instructional design", Educational Psychology, Volume 10, pp. 251–296, 1998.

[6]. Mayer, R. E., and Moreno, R., "Aids to computer-based multimedia learning", Learning and Instruction, volume 12, pp. 107–119, 2002.

[7]. Richard Mayer, Cognitive theory of Multimedia Learning, Cambridge handbook of multimedia learning, New York: Cambridge University Press, pp. 19-30, 2005.

[8]. Paas, F., Tuovinen, J.E., Tabbers, H. & Van Gerven, P.W.M., "Cognitive load measurement as a means to advance cognitive load theory", Educational Psychologist, Volume 38, pp. 63–71, 2003.

[9]. Guoqiang Peter Zhang, "Neural Networks for Classification: A Survey", IEEE Transactions on Systems, Man, And Cybernetics—Part C: Applications and Reviews, Volume 30, Number 4, pp.451-461, 2000.

[10]. Kennedy J., Eberhart R., "Particle Swarm Optimization", Proceedings of IEEE International Conference on Neural, Volume 4, pp.1-12, 1995.

[11]. Sarita Mahapatra, Alok Kumar Jagadev, Bighnaraj Naik, "Performance Evaluation of PSO Based Classifier for Classification of Multidimensional Data with Variation of PSO Parameters in Knowledge Discovery Database", International Journal of Advanced Science and Technology, Volume 34, pp. 27-43, 2011.

[12]. Jan L Plass, "Design and Evaluation of the User Interface of Foreign Language Multimedia software: A Cognitive Approach; Language Learning & Technology", Volume 2, Number 1, pp. 35-45, 1998.

[13]. Loredana Mihalcaa, Ron J.C.M. Saldenb, Gemma Corbalanc, Fred Paas d, Mircea Micleaa "Effectiveness of cognitive-load based adaptive instruction in genetics education" Computers in Human Behavior ,Volume 27, PP. 82–88, 2011.

[14]. Baylari, A., Montazer, G. A., "Design a personalized e-learning system based on item response theory and artificial neural network approach", Expert Systems with Applications, Vol. 36, No. 4, pp. 8013-8021, 2009.

[15]. Fausett, L., "Fundamentals of neural networks", Englewood Cliffs, NJ: Prentice-Hall, 1994.

[16]. Wu, D., Yang, Z., & Liang, L., "Using DEA-neural network approach to evaluate branch efficiency of a large Canadian bank", Expert Systems with Applications, Volume 31, pp. 108-115, 2006.

[17]. Qinghai Bai, "Analysis of Particle Swarm Optimization Algorithm", Computer and Information Science, Volume 3, Number 1, pp. 180-183, 2010.

[18]. S. Kannan, S. Mary Raja Slochanal, P. Subbaraj and Narayana Prasad Padhy, "Application of Particle swarm optimization technique and its variants to Generation expansion planning problem", Electric Power Systems Research, (Elsevier Science Ltd) Volume 70, Issue 3, pp. 203-210, 2004.

[19]. Mojtaba Ahmadieh Khanesar, Hassan Tavakoli, Mohammad Teshnehlab, Mahdi Aliyari Shoorehdeli, " Novel Binary Particle Swarm Optimization", Particle Swarm Optimization, 2009, ISBN: 978-953-7619-48-0.

L. Jayasimman working as a Assistant Professor, with department of Computer Application, J.J. College of Engineering and Technology, Trichy, India. He received his M.Tech degree in Bharathidasan University, Trichy, India in 2008. Now he is pursuing PhD (Computer Science) in Bharathidasan University.

Dr. E. George Dharma Prakash Raj working as a Assistant Professor with School of Computer Science and Engineering, Bharathidasan University, Trichy, India. He has more than twenty years of experience in Academics. He is a Chairman/Member in Board of Studies in Colleges and Universities in India. He is also a member in Research Organizations, International Program Committee Member, Reviewer for international/National journals and conferences.