An Efficient Learning Scheme for Extreme Learning Machine and Its Application

Kheon-Hee Lee, Miso Jang, Keun Park , Dong-Chul Park, Yong-Mu Jeong and Soo-Young Min

***Abstract*—**An efficient learning scheme for extreme learning machine (ELM) is proposed in this paper. One of the main obstacles in practical use of convolutional neural network (CNN) is their computational complexity in its training stage. The extreme learning machine can be a promising tool for solving the computational complexity in CNNs. The weights between input layer and hidden layer are randomly generated in ELM and these weights can be tuned for more efficient learning. This paper also proposes a variation of ELM, called Average ELM, that replaces input layer's random weights with predetermined weights. This paper shows a feasibility study in tuning these weights and applying extreme learning machine to CNNs for faster training with some initial experimental results. The proposed method can reduce the size of target network without sacrificing the performance of neural network. Experiments on several databases show that the proposed training method can reduce the training time while preserving the classification accuracy.

***Keywords*—**data classification. extreme learning machine. neural networks. random weights.

I. INTRODUCTION

THE data classification task is one of the ongoing important topics in various pattern recognition tasks. Especially, many image classification methods have been proposed and traditional image classification methods consist of a feature extraction procedure and a classifier design process. One very important characteristic of conventional classifier schemes is that their classification accuracy heavily depends on the feature extraction method they adopt and finding a specific best feature extractor for a specific object becomes a critical task for the object classification problem. An example of feature extraction method, HOG (Histogram of Oriented Gradients)[1] is useful for human detection task while other feature extraction methods[2] for general purpose in image classification. On the other hand, the deep learning scheme adder sses this problem of accurate feature extractor for a specific object recognition task. As a class of deep learning models, CNNs [3,4] are known as the first truly successful deep network architecture and are specially designed for image data. Since CNNs have shown that they can project the feature of image without any pre-processing step, CNNs can be considered as a complete classification system.

The training process for CNNs is usually far slower than required. Since CNNs can be considered as an example of feedforward neural network, the gradient decent-based learning methods used for multi-layer perceptron type neural network (MLPNN) are also used for CNNs. One of the solutions to deal with this problem of slow convergence in MLPNN, extreme learning machine (ELM) whose learning speed can be far faster than traditional back-propagation (BP) algorithm while maintaining generalization performance. The extreme learning machine proposed by Huang reported that ELM can reduce the training time significantly, say 100 times faster, when compared with BP algorithm or Support Vector Machine(SVM) while maintaining its classification accuracy at the level of BP or SVM. ELM can be considered as a generalized version of single-hidden-layer-feedforward networks (SLFNs). In ELM, the weights between input layer and hidden layer are randomly generated and the weights between hidden layer and output layer are computed by using the outputs of hidden neurons with randomly assigned input-hidden neuron weights. The randomness in input- hidden neuron weights can lead redundancy and problems in the structure of ELM. This paper handles this problem of randomness in input- hidden neuron weights and proposes an efficient method to overcome the problem by introducing the average concept in output values of hidden neurons.

The rest of this paper is organized as follows: Section II and Section III provide the brief summaries of CNNs and the ELM, respectively. An efficient method for ELM training by introducing the average concept in output values of hidden neurons is proposed in Section IV. The basic idea for combining CNNs and ELM is discussed and initial experiments in Section V. Section VI concludes this paper.

II. CONVOLUTIONAL NEURAL NETWORKS

Convolutional Neural Networks (CNNs) are known as the first successful deep architecture that still keep the characteristics of the tradition neural networks. As a generic deep learning algorithm [5], the abstract level increases from layers to layers. Designing CNNs consists of two types of hidden layers: convolution and sub-sampling layers. The key idea of CNNs is that it minimizes the parameters needed to learn by pooling the

Kheon-Hee Lee, Miso Jang, Keun Park and Dong-Chul Park are with Department of Electronics Engineering, Myong Ji University, Yong In, Gyeonggi-do, 449-728, South Korea (e-mail: parkd@mju.ac.kr).

Soo-Young Min and Yong-Mu Jeong are with Software Device Research Center, Korea Electronics Technology Institute, Song Nam, Gyeonggi-do, South Korea (e-mail: minsy@keti.re.kr).

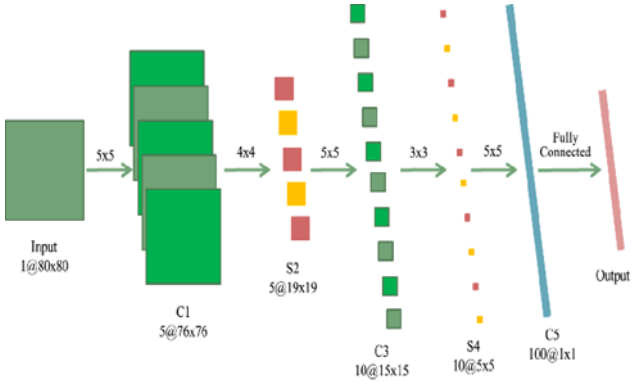


Fig. 1 An example of CNN Architecture for image classification

filter responses over the convolution and subsampling layers to obtain higher level abstract features. The convolution layers perform convolutions over feature maps in previous layers. The *j* feature map in the *i* layer, denoted as m_{ij}, is convolved with some small kernels *k* as follows:

$$m_{ij} = tanh(\sum\_n * k_{ijn} + b_{ij}) \quad (1)$$

where *tanh* is the hyperbolic tangent function and b_{ij} is a trainable bias for each map while k_{ijn} is the trainable kernel for m_{ij} corresponding to each map $m_{(i-1)j}$. The sub-sampling layers compute the spatial average of a small region ($n \times n$ pixels), multiply it by a weight w_{ij} , then add a trainable bias and pass through the activation function as follows:

$$m_{ij} = tanh(w_{ij} \sum m_{(i-1)j}{}^{n \times n} + b_{ij}) \quad (2)$$

One of the advantages of CNNs is that we can use it in various structure of design for each particular data set. An example of CNNs is shown in Fig. 1. One example of CNN structures is analyzed in detail [6]. Other CNN structures can be obtained by adjusting filter size, and input size, or number of maps in each layer.

III. Extreme Learning Machine

All the parameters in feedforward-type neural networks are to be determined by using learning algorithms. The back-propagation (BP) training algorithm based on the gradient decent method has been most widely used for training of feedforward type neural networks. In order to overcome the slow training performance, the extreme learning machine (ELM) has been proposed [7]. Major advantage in ELM over conventional BP is that ELM learns without iteratively tuning hidden weights. When the weights connecting the hidden layer, β, and the output layer are solved by minimizing the following approximation error:

$$\min \| H \beta - T \|^2 \quad (3)$$

where H is the hidden layer output matrix and T is the training data target matrix.

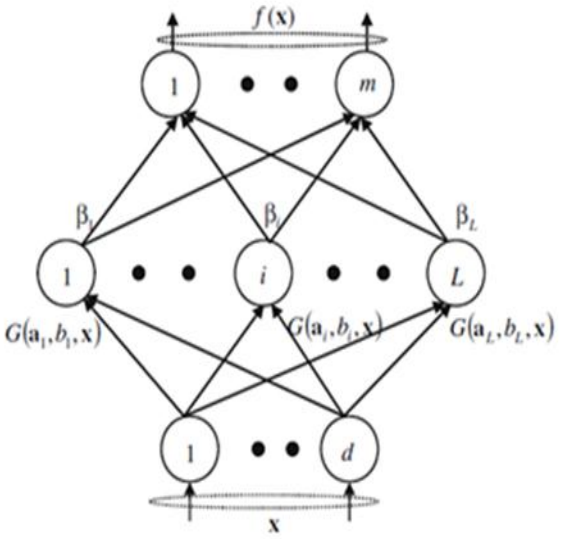


Fig. 2 Notation for Extreme Learning Machine

The optimal solution of β can be given as :

$$\beta^* = H^+ T \quad (4)$$

where H^+ denotes a generalized inverse of the matrix H.

There exist many different methods for solving Eq. (3) including Gaussian elimination, iterative methods, and single value decomposition. In summary, the procedure for obtaining the output weights in ELM consists of the following three steps:

Step 1. Assign randomly selected numerical values between 0 and 1 to input weights and the bias of the hidden layer neurons.

Step 2. Calculate the output matrix *H*.

Step 3. Calculate the output weights β that satisfies Eq. (4).

ELM has the advantage in training speed and generalization performance while having problems including robustness. Regularized ELM was proposed to alleviate the robustness problem in ELM by minimizing the regularized cost function of least squared estimated regularization with the following formulation [8]:

$$\min L_{REM} = \frac{1}{2} \| \beta \|^2 + \frac{C}{2} \| H \beta - T \|^2 \quad (5)$$

where C is a scale parameter.

By setting the gradient of L_{REM} with respect to β to zero, we can find the output weight matrix β when the number of training samples is larger than the number of hidden neurons as follows:

$$\beta = (\frac{I}{C} + H^T H)^{-1} H^T Y \quad (6)$$

The output weight matrix matrix β when the number of training samples is smaller than the number of hidden neurons is given as:

$$\beta = H^T (\frac{I}{C} + H^T H)^{-1} Y \quad (6)$$

TABLE II
COMPARE CLASSIFICATION STABILITY

Data	ELM			ELM with EBP-weight		
	Acc	Dev	Time	Acc	Dev	Time
Wine	0.9825	0.0281	0.0402	0.9838	0.0274	0.1621
Iris	0.9667	0.0359	0.0264	0.9733	0.0353	0.3208
Segment	0.9298	0.0171	0.2142	0.9416	0.0158	1.0675
Satimage	0.8591	0.0089	0.8507	0.8598	0.0083	1.0118
Ionosphere	0.8812	0.0573	0.0500	0.8729	0.0504	0.2646

TABLE Ⅲ
COMPARE CLASSIFICATION CAPABILITY OF ELM WITH AVERAGE ELM

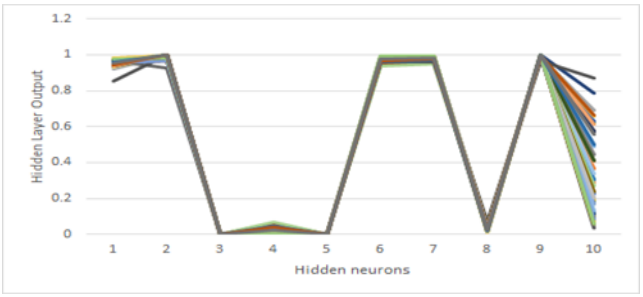
Data	ELM			Average ELM		
	Acc	Dev	Time	Acc	Dev	Time
Wine	0.9813	0.0312	0.0198	0.9450	0.0581	0.0770
Iris	0.9653	0.0447	0.0144	0.9427	0.0667	0.0868
Segment	0.88	0.0166	0.0766	0.7692	0.0520	0.3073
Satimage	0.80	0.0138	0.2105	0.6826	0.0809	0.9243
Ionosphere	0.8618	0.0507	0.0278	0.7735	0.0770	0.3410

More recently, Huang et al. propose an architecture that combines the kernel method and ELM, called Extreme Learning Machine with Kernel (KELM). In this scheme, the outputs of the hidden layer of ELM can be considered as a nonlinear mapping of samples and KELM constructs a kernel function instead of $H^T H$[9].

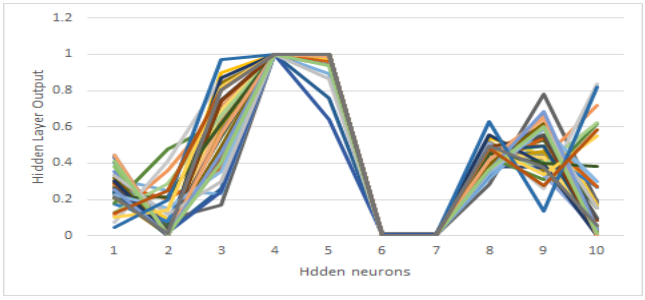
IV. AVERAGE EXTREME LEARNING MACHINE

The activation function for ELM can be described as G(*a*,*x*,*b*) where *a*, *x*, and *b* denote input weight, input, and bias, respectively. Since *a* and *b* are obtained randomly, the resulting G(*a*,*x*,*b*) can be considered as a function of *x* with random feature and hidden layer output mapping, *H*, is often called as random feature mapping. However, when we train the input layer with Back Propagation algorithm, we can notice that hidden neuron outputs are not random with binary classes as shown in Fig. 3. This clearly shows that the output of hidden neurons can be clustered by the data classes and these are not random. Based on the observation on the activation of the hidden neurons for different data classes, the random feature mapping of ELM can be replaced with the trained weights.

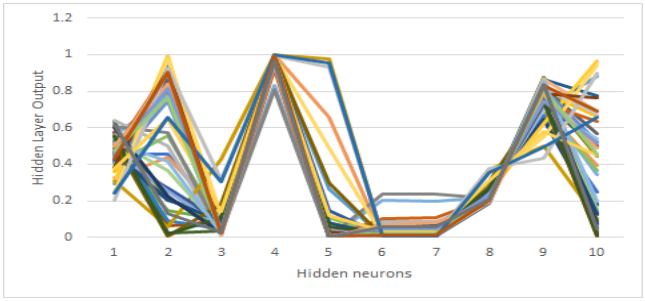
In the proposed method, a portion of randomly selected training data are used for training the network. In back propagation training, maximum epoch and learning gain are set to be 5,000 and 0.01, respectively. 20 initial training data are randomly selected for each training data set. As a result of the training, the hidden layer output matrix is clustered according to the data classes. As the number of training data grows, the number of hidden neurons in ELM structure grows and the computational burden in computing inverse matrix grows more significantly, accordingly. In order to achieve efficient training with more efficient neural network structure, the proposed method in this paper utilized the clustered information of hidden neurons' outputs for different classes resulting from back



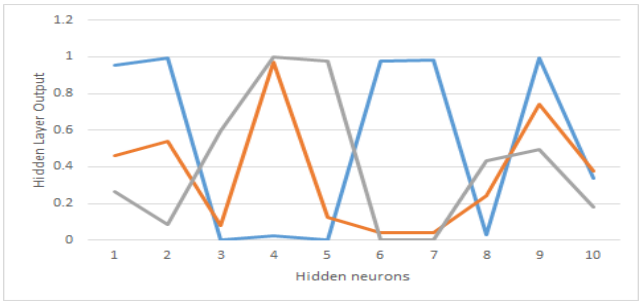
(a)



(b)



(c)



(d)

Fig. 3 Output pattern of hidden layer neurons trained with BP on Iris data for : (a) Class 1, (b) Class 2, (c) Class 3. (d) Average value for the outputs for 3 classes

propagation training with sampled training data. In general, the basic ELM requires NxMxN multiplication for $H^T H$ for *H* matrix with the size of NxN and 2MN(M-1) multiplication for matrix inversion with the Gaussian Elimination method. It is obvious that the number of multiplication can be reduced significantly when the number of data, M, can be replaced with the number of classes as proposed in this paper because M is much larger than the number of classes in data.

V. Experiments and Results

In order to evaluate the proposed Average ELM, several UCI datasets as shown in Table I are used for experiments [10]. Experiments are performed with Intel Core i5-3570 CPU, 8GB RAM, and C language for Visual Studio 2013. Experiments are performed two parts. In the first part, the stability on feature mapping H(x) is tested for both cases: BP training weighted ELM and basic ELM. In the second part, the proposed Average ELM and basic ELM are compared in terms of training speed and classification accuracy. From the data sets in Table I, 90% data in each class are randomly chosen for training classifiers while the remaining data are used for evaluating classifiers. This training data and test data combination is collected 10 times. That is, there exist 10 different combinations of randomly chosen training data and test data sets. The accuracy of different algorithms used in experiments is reported by mean and variance from these 10 data sets.

A. Stability of BP Trained Weight-based ELM

In this experiment, the number of hidden neurons is set to be 20 for all cases. This training and test results are summarized in Table II for different data sets. As can be seen from Table II, the input layer weights with BP training give more stable results in terms of classification accuracy when compared with the basic ELM. This result shows that the optimized input layer weights with BP training is more feasible than randomly assigned weights. Similar results on the problem of randomly assigned weights are reported [8].

B. Training Speed and Classification Accuracy of Average ELM

In this experiment, the proposed Average ELM is evaluated with the basic ELM in terms of training speed and classification accuracy. In all experiments, the number of hidden neurons, the number of training epochs, and learning rate are set to be 20, 5,000, and 0.01, respectively. The results are summarized in Table III. The results shown in Table III indicate that the proposed Average ELM requires more training time for BP training and its classification accuracy was not improved when compared with the basic ELM. This result is somewhat anticipated since the average ELM is not so significant with the number of training data is small and the network architecture is not large.

C. Application of Average ELM to Convolutional Neural Network

In the input layer as shown in Fig. 1, the original image is converted to gray image and resized to our standard 80×80 size [6]. In order to produce maps, size of 76×76, convolutions using 5×5 kernels and biases are performed by using convolution layer (C1). Spatial sub-sampling of each map in C1 is performed in sub-sampling layer (S2). Note that 5 weights and 5 biases are used to produce five 19×19 maps. Convolutions between all possible combinations of two difference maps in S2 and 5×5 kernels are performed to produce one map in C3. Note that layer C3 consists of 10 15×15 maps. With the same fashion, 10 5×5 maps of layer S4 are obtained and 100 1×1 maps of layer C5 are also obtained. The output layer is constructed with the full connection of layer C5 and the output neurons whose dimension is the same as the number of target classes. Note that there are 25,000 (100×10×5×5) parameters in layer C5 and these parameters are usually optimized by using the error-back-propagation algorithm which shows slow in convergence speed.

In CNN training, the above mentioned number of parameters are huge and its training time required, for example, for 30 image data per class and 5 class problem are over 140 hours [6]. In order to reduce this training time for CNN, Average ELM can be applied to the final layer of CNN. In this case, the matrix inversion for Average ELM depends on the final C5 and the number of classes. Initial experiments on the data set used in [6] show that the proposed method can save the training time for CNNs as much as 30 % while maintaining the classification accuracy. More intensive experiments on this subject are under way and the results will be reported in future.

VI. Conclusion

In order to reduce the computational loads involved in training convolutional neural networks, the concept of Average ELM is proposed in this paper. The architecture and training procedure of CNNs are reviewed and the pros and cons of CNNs are also presented in this paper. In order to enhance the applicability of CNNs to practical problems, a solution to fast training of CNNs is required. One option to the problem can be the ELM since ELM was introduced to solve the slow training problem in BP algorithm that is adopted in training CNNs. This paper proposes the Average ELM that can reduce training time with large data set. This paper also proposes a method to apply Average ELM concept to the final layer of CNNs. When applied to sets of UCI data classification problem, the proposed method shows promising results in terms of training speed and classification accuracy. With further experiments on larger training data sets, the training speed can be measured and its advantage will be witnessed further.

Acknowledgment

This work was supported by the IT R&D program of The MKE/KEIT (10040191, The development of Automotive Synchronous Ethernet combined IVN/OVN and Safety control system for 1Gbps class).

References

[1] N. Dala and B. Triggs, "Histograms of Oriented Gradients for Human Detection," *Proc. of IEEE Conf. on CVPR*, vol. 1, pp. 886-893, 2005.

[2] Y.L. Huang and R.F. Chang, "Texture Features for DCT-Coded Image Retrieval and Classification," *Proc. of IEEE ICASSP*, vol. 6, pp. 3013–3016, Mar, 1999.

[3] Y. Lecun, P. Haffner, L. Bottou, and Y. Bengio, "Gradient Based Learning Applied to Document Recognition," *Proc. of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov, 1998.

[4] Fu Hie Huang and Y. Lecun, "Large-scale Learning with SVM and Convolutional for Generic Object Categorization," *Proc. of IEEE Conf. on CVPR*, pp. 284-291, 2006.

[5] *Pattern Analysis and Machine Learning*, vol. 26, no. 11, pp. 1408–1423, Nov, 2004.

[6] Thao Nguyen, Kheon-Hee Lee, Miso Jang, Dong-Chul Park, Soo-Young Min, and Yunsik Lee, "Classifier Integration Model with Convolution Neural Networks," *Proc. of 4th ICCEIT*, vol.1, pp. 140-143, Dec. 2014.
[7] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundarajan, " A fast and accurate Online Sequential Learning Algorithm for Feed-forward Networks," *IEEE Tr. on Neural Networks*, vol. 17, No. 6, pp. 1411- 1423, 2006
[8] W.-Y. Deng, Q.-H. Zheng, L. Chen, and X.-B. Xu, " Research on extreme learning of neural networks," Chinese Journal of Computers, vol. 33, no. 2, pp. 279-287, 2010.
[9] X. Liu, L. Wang, G.-B. Huang, J. Zhang, and J. Yin, "Multiple kernel extreme learning machine," *Neurocomputing*, vol. 149, ParT a, PP. 253-264, 2015
[10] https://archive.ics.uci.edu/ml/datasets.

Kheon-Hee Lee received the B.S. degree in Electronics Engineering from MyongJi University, Korea, in 2013. He is pursuing his M.S. degree in Electronics Engineering at Intelligent Computing Research Lab at MyongJi University. His research interests include pattern recognition, deep learning, neural networks, and object recognition.

Miso Jang received the B.S. degree in Electronics Engineering from MyongJi University, Korea, in 2012. She is pursuing his M.S. degree in Electronics Engineering at Intelligent Computing Research Lab at MyongJi University. Her research interests include intelligent computing, neural networks, and pattern recognition.

Dong-Chul Park (M'90-SM'99) received the B.S. degree in electronics engineering from Sogang University, Seoul, Korea, in 1980, the M.S. degree in electrical and electronics engineering from the Korea Advanced Institute of Science and Technology, Seoul, Korea, in 1982, and the Ph.D. degree in electrical engineering, with a dissertation on system identifications using artificial neural networks, from the University of Washington (UW), Seattle, in 1990. From 1990 to 1994, he was with the Department of Electrical and Computer Engineering, Florida International University, The State University of Florida, Miami. Since 1994, he has been with the Department of Electronics Engineering, MyongJi University, Korea, where he is a Professor. From 2000 to 2001, he was a Visiting Professor at UW. He is a pioneer in the area of electrical load forecasting using artificial neural networks. He has published more than 130 papers, including 40 archival journals in the area of neural network algorithms and their applications to various engineering problems including financial engineering, image compression, speech recognition, time-series prediction, and pattern recognition. Dr. Park was a member of the Editorial Board for the IEEE TRANSACTIONS ON NEURAL NETWORKS from 2000 to 2002.

Soo-Young Min received the B.S. degree in Electronics Engineering from Inha University, Incheon, Korea, in 1987. Since 1993, he has been with Software Device Research Center at Korea Electronics Technology Institute, Song Nam, Korea, where he is a senior researcher. He has been involved in numerous research projects on wireless communication protocol and system software. His research interests include vehicular communication network, wireless communication protocol, and system software design.