Cairo University

Egyptian Informatics Journal

www.elsevier.com/locate/eij
www.sciencedirect.com

ORIGINAL ARTICLE

Shannon Entropy and Mean Square Errors for speeding the convergence of Multilayer Neural Networks: A comparative approach

Hussein Aly Kamel Rady

El Shorouk Academy, Computer Science Department, El Shorouk – Cairo, P.O. Box 3, El Shorouk, Egypt

Received 18 May 2011; revised 22 September 2011; accepted 26 September 2011
Available online 9 November 2011

KEYWORDS

Shannon Entropy;
Mean Square Error;
Activation function;
Learning rate;
Backpropagation Neural Network

Abstract Improving the efficiency and convergence rate of the Multilayer Backpropagation Neural Network Algorithms is an active area of research. The last years have witnessed an increasing attention to entropy based criteria in adaptive systems. Several principles were proposed based on the maximization or minimization of entropic cost functions. One way of entropy criteria in learning systems is to minimize the entropy of the error between two variables: typically one is the output of the learning system and the other is the target. In this paper, improving the efficiency and convergence rate of Multilayer Backpropagation (BP) Neural Networks was proposed. The usual Mean Square Error (MSE) minimization principle is substituted by the minimization of Shannon Entropy (SE) of the differences between the multilayer perceptions output and the desired target. These two cost functions are studied, analyzed and tested with two different activation functions namely, the Cauchy and the hyperbolic tangent activation functions. The comparative approach indicates that the Degree of convergence using Shannon Entropy cost function is higher than its counterpart using MSE and that MSE speeds the convergence than Shannon Entropy.

© 2011 Faculty of Computers and Information, Cairo University.
Production and hosting by Elsevier B.V. All rights reserved.

E-mail address: dr_Hussein_Rady@yahoo.com

1110-8665 © 2011 Faculty of Computers and Information, Cairo University. Production and hosting by Elsevier B.V. All rights reserved.

Peer review under responsibility of Faculty of Computers and Information, Cairo University.
doi:10.1016/j.eij.2011.09.002

Production and hosting by Elsevier

1. Introduction

Artificial Neural Networks (ANNs) has been a hot topic in recent years in cognitive science, computational intelligence and intelligent information processing [1–7]. They 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 [8,9]. On the other hand, a Neural Network is a well known as one of powerful computing tools to solve optimization problems. Due to massive computing unit neurons and parallel mechanism of neural network approach it can solve the large-scale problem efficiently and optimal solution can be obtained [10]. 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 functional or distributional form for the underlying model. *Second*, they are universal functional approximators in that neural networks can approximate any function with arbitrary accuracy. *Third*, neural networks are nonlinear models, which makes them flexible in modeling real world complex relationships. Finally, neural networks are able to estimate the posterior probabilities, which provides the basis for establishing classification rule and performing statistical analysis.

The feedforward neural network [11–15] is the simplest (and therefore, the most common) ANN architecture in terms of information flow direction. Many of neural network architectures are variations of the feedforward neural network [16]. Backpropagation (BP) is the most broadly used learning method for feedforward neural networks [17,11,18,14]. There are two practical ways to implement the Backpropagation algorithm: batch updating approach and online updating approach. Corresponding to the standard gradient method, the batch updating approach accumulates the weight correction over all the training samples before actually performing the update. On the other hand, the online updating approach updates the network weights immediately after each training sample is fed [1,19].

Information theory is commonly used in coding and communication applications and more recently, it has also been used in classification. In information theoretic classification, a learner is viewed as an agent that gathers information from some external sources. Information theoretic quantities have been widely used for future extraction and selection [20]. As defined in information theory, entropy is a measure of the uncertainty of a particular outcome in a random process [1,21]. The entropy of a random variable is a measure of the uncertainty of the random variable; it is a measure of the amount of information required on the average to describe the random variable. Entropy is a nonlinear function to represent information we can learn from unknown data. In the learning process, we learn some constraints on the probability distribution of the training data from their entropy.

Usually error backpropagation for neural network learning is made using MSE as the cost function [22]. During the learning process, the ANN goes through stages in which the reduction of the error can be extremely slow. These periods of stagnation can influence learning times. In order to resolve this problem, the MSE are replaced by entropy error function [23,8,24]. Simulation results using this error function shows a better network performance with a shorter stagnation period. Accordingly, our purpose is the use of the minimization of the error entropy instead of the MSE as a cost function for classification purposes. Let the error $e(j) = T(j) - Y(j)$ represent the difference between the target T of the j output neuron and its output Y, at a given time t. The MSE of the variable $e(j)$ can be replaced by its EEM counterpart.

MSE has been a popular criterion in the training of all adaptive systems including artificial neural networks. The two main reasons behind this choice are analytical tractability and the assumption that real-life random phenomena may be sufficiently described by second-order statistics. The Gaussian probability density function (pdf) is determined only by its first- and second-order statistics, and the effect of linear systems on low order statistics is well known. Under these linearity and Gaussianity assumptions, further supported by the central limit theorem, MSE, which solely constrains second-order statistics, would be able to extract all possible information from a signal whose statistics are solely defined by its mean and variance [25]. On the other hand, MSE can extract all the information in the data provided that the dynamic system is linear and the noise is Gaussian distributed. However, when the system becomes nonlinear and the noise distribution is non-Gaussian, MSE fails to capture all the information in the error sequences. In this case an alternative criterion is needed in order to achieve optimality. Entropy is a natural extension beyond MSE since entropy is a function of probability density function (pdf), which considers all high order statistics [26]. Various optimization techniques were suggested for improving the efficiency of error minimization process or in other words the training efficiency [27,28].

The rest of the paper is organized as follows. Related work is outlined in Section 2. Section 3 introduces the Multilayer Backpropagation Neural Networks. Section 4 introduces the Mean Square Error. Shannon Entropy was discussed and analyzed in Section 5. Simulated results were discussed in Section 6 for Shannon Entropy and in Section 7 for Mean Square Error. Section 8 compares Shannon Entropy and MSE. Finally Conclusions are outlined in Section 9.

2. Related work

Entropy, which is introduced by Shannon, is a scalar quantity that provides a measure for the average information contained in a given probability distribution function. By definition, information is a function of the pdf; hence, entropy as an optimality criterion extends MSE. When entropy is minimized, all moments of the error pdf (not only the second moments) are constrained. The entropy criterion can generally be utilized as an alternative for MSE in supervised adaptation, but it is particularly appealing in dynamic modeling [25]. MSE can extract all the information in the data provided that the dynamic system is linear and the noise is Gaussian distributed. However, when the system becomes nonlinear and the noise distribution is non-Gaussian, MSE fails to capture all the information in the error sequences. Entropy is a natural extension beyond MSE since entropy is a function of probability density function (pdf), which considers all high order statistics [26].

Many researchers introduces the theoretical concepts of using Error Entropy Minimization as a cost function for artificial Neural Networks. In [26], Xu et al. discusses the information theoretic learning and states that entropy, which measures the average information content in a random variable with a particular probability distribution was previously proposed as a criterion for supervised adaptive filter training and it was shown to provide better neural network generalization compared to MSE. In [22], Alexandre and Sa introduces the Error Entropy Minimization approach to replace the MSE, as the cost function of a learning system, with the entropy of the error. They discusses the theoretical basis of the Renyi's quadratic entropy. In their experimental results, they used three values of Learning rates which are 0.1, 0.2, and 0.3 with MSE and EEM for different smoothing parameters and they calculate the rate of convergence. The methodology of their experimental results differ from our methodology. We used different activation functions as well as different learning rates. They used a recurrent Neural Networks, but I used Backprobagation Neural Networks.

Silva et al. in [29], introduces the concepts of Neural Network Classification using Shannon's Entropy with a variable learning rate. They used the EEM algorithm with Shannon Entropy that performed very well when compared to MSE and Cross Entropy. Their results show the effectiveness of entropic criteria, in particular Shannon's Entropy, as cost functions in classification tasks. In [21], Erdogmus et al. discusses the Quadratic Entropy Estimator. In [30], Erdogmus et al. also proposed a cost function that tries to minimize the MSE, while it pays attention to maintaining the variation between consecutive errors small. In [31], William and Hoffman investigates Error Entropy and MSE Minimization for lossless Image. In [25], Erdogmus and Principe, says that, we propose minimization of error entropy as a more robust criterion for dynamic modeling and an alternative to MSE in other supervised learning applications using nonlinear systems such as nonlinear system identification with neural networks. In [32], Bromiley et al. studied and compared Shannon Entropy, Renyi's Entropy, and Information.

3. Multilayer Backpropagation Neural Networks

The Artificial Neural Networks are known as the "universal approximators" and "computational models" with particular characteristics such as the ability to learn or adapt, to organize or to generalize data [33]. Up-to-date designing a (near) optimal network architecture is made by a human expert and requires a tedious trial and error process [16,34,35,7]. On the other hand, they are simplified mathematical approximations of biological neural networks in terms of structure as well as function. In general, there are two aspects of ANN functioning: (1) the mechanism of information flow starting from the presynaptic neuron to postsynaptic neuron across the network and (2) the mechanism of learning that dictates the adjustment of measures of synaptic strength to minimize a selected cost or error function (a measure of the difference between the ANN output and the desired output). Research in these areas has resulted in a wide variety of powerful ANNs based on novel formulations of the input space, neuron, type and number of synaptic connections, direction of information flow in the ANN, cost or error function, learning mechanism, output space, and various combinations of these [16,36].

One of the most commonly used supervised Artificial Neural Network (ANN) model is backpropagation network that uses backpropagation learning algorithm [37–39]. Backpropagation algorithm is one of the well-known algorithms in neural networks. It is one of the most common supervised training methods [40]. Training is usually carried out by iterative updating of weights based on minimizing the Mean Square Error. In the output layer, the error signal is the difference between the desired and the output values. Then the error signal is fed back through the steepest descent algorithm to the lower layers to update the weights of the network. The weights of the network are adjusted by the algorithm such that the error is decreased along a descent direction. Traditionally, two parameters, called learning rate and momentum factor, are used for controlling the weight adjustment along the descent direction and for dampening oscillations. However, the convergence rate of the BP algorithm is relatively slow, especially for networks with more than one hidden layer. The reason for this is the saturation behavior of the activation function used for the hidden and output layers. Since the output of a unit exists in the saturation area, the corresponding descent gradient takes a very small value, even if the output error is large, leading to very little progress in the weight adjustment. The Backpropagation algorithm may be described with the following three steps, which have to be applied several times in an iteration.

1. Forward computation of input signal of training sample and determination of neural network response.
2. Computation of an error between desired response and neural network response.
3. Backward computation of the error and calculation of corrections to synaptic weights and biases [36,30].

3.1. Activation functions

The activation function can adjust the step, position and mapping scope simultaneously, so it has stronger non-linear mapping capabilities [41,33]. Activation function in a backpropagation network defines the way to obtain output of a neuron given the collective input from source synapses. The Backpropagation algorithm requires the activation function to be continuous and differentiable. The following two activation functions were proposed and used in our simulated results [36].

3.1.1. The hyperbolic tangent function

$$F(v)\,tanh(v) = \frac{\exp(v) - \exp(-v)}{\exp(v) + \exp(-v)} \tag{1}$$

The limiting values of this function are -1 and $+1$.

The derivative of $F()$ with respect to v is

$$\begin{aligned} F'(v) &= sech^2(v) = [1 - tanh^2(v)] \\ &= [1 - F^2(v)]\,[1 - F(v)][1 + F(v)] \end{aligned} \tag{2}$$

For a neuron j located in the output layer

$$\delta_j = (1 - O_j)(1 + O_j)(T_j - O_j) \tag{3}$$

For a neuron j located in the hidden layer

$$\delta_j = (1 - O_j)(1 + O_j)\sum_k \delta_k w_{jk} \tag{4}$$

where δ_k is the error gradient at unit k to which a connection points from hidden unit j.

3.1.2. The Cauchy distribution function

The formula for the cumulative distribution function for the Cauchy distribution is:

$$F(v) = 0.5 + \frac{1}{\pi} tan^{-1} v \tag{5}$$

$$F'(v) = \frac{1}{\pi}\left[\frac{1}{1 + tan^2 F(v)}\right] \tag{6}$$

For a neuron j located in the output layer

$$\delta_j = \frac{1}{\pi(1 + tan^2 O_j)}(T_j - O_j) \tag{7}$$

For a neuron j located in the hidden layer

$$\delta_j = \frac{1}{\pi(1 + tan^2 O_j)} \sum_k \delta_j w_{jk} \tag{8}$$

where δ_k is the error gradient at unit k to which a connection points from hidden unit j.

3.2. Learning rates

Learning rate is one of the parameters which governs how fast a neural network learns and how effective the training is. The learning rate (η) is the conventional BP learning rule is a decisive factor in regard to the size of the weights adjustments made at each iteration and hence affects the convergence rate. Nevertheless, the best choice of η is problem dependent and may need some trial-and-error before a good choice is found. If the chosen value of η is too large for the error surface, the search path will oscillate about the ideal path and converge more slowly than a direct descent, on the other hand, if the chosen value of η is too small, the descent will progress in very small steps significantly increasing the total time to convergence [42].

4. Mean Square Errors

In statistics, the Mean Squared Error (MSE) of an estimator is one of many ways to quantify the amount by which an estimator differs from the true value of the quantity being estimated. MSE measures the average of the square of the "error." The error is the amount by which the estimator differs from the quantity to be estimated. The difference occurs because of randomness or because the estimator does not account for information that could produce a more accurate estimate.

The Least Mean Square *LMS* cost function has been used more frequently than any alternative cost function in Neural Networks. It yields good performance with large data bases on real world. *LMS* error cost function is the most often used error function despite being criticized for its lack of convergence speed and a higher possibility of being trapped in a local minima in the network training process [8].

For the general multi-class in Multilayer Neural Networks, let us consider the problem of assigning an input vector $\boldsymbol{x} = \{\boldsymbol{x}_i: i = 1,\ldots,D\}$ to one of M classes $\{c_i: i = 1,2,\ldots,M\}$. Let c_i denote the corresponding class of $\mathbf{x}$, $\{y_i(\boldsymbol{x}): i = 1,2,\ldots,M\}$ the outputs of the network, and $\{d_i: i = 1,2,\ldots,M\}$ the target outputs for all output nodes. With the least mean square cost function, the network parameters are chosen to minimize the following:

$$\Delta = E\left\{\sum_{i=1}^{M}[y_i(\boldsymbol{x}) - d_i]^2\right\} \tag{9}$$

where $E\{\cdot\}$ is the expectation operator. Denoting the joint probability of the input and the ith class by $p(x, c_i)$, we obtain

$$\Delta = \int \sum_{j=1}^{M}\left\{\sum_{i=1}^{M}[y_i(\mathbf{x}) - d_i]^2\right\} p(x, c_i)dx \tag{10}$$

Substituting $p(\boldsymbol{x}, c_i) = p(c_i|\boldsymbol{x})p(\boldsymbol{x})$ in Eq. (10) gives:

$$\Delta = \int \left\{\sum_{j=1}^{M}\sum_{i=1}^{M}[y_i(\boldsymbol{x}) - d_i]^2 p(c_i|\boldsymbol{x})\right\} p(\mathbf{x})dx$$
$$= E\left\{\sum_{j=1}^{M}\sum_{i=1}^{M}[y_i(\boldsymbol{x}) - d_i]^2 p(c_j|\boldsymbol{x})\right\} \tag{11}$$

$$= E\left\{\sum_{j=1}^{M}\sum_{i=1}^{M}[y_i^2(x)p(c_j|x) - 2y_i(x)d_i p(c_j|x) + d_i^2 p(c_j|x)]\right\} \tag{12}$$

But since, $y_i^2(\boldsymbol{x})$ is a function only of $\boldsymbol{x}$ and $\sum_{j=1}^{M} p(c_j|x) = 1$, we obtain

$$\Delta = E\left\{\sum_{i=1}^{M}[y_i^2(\boldsymbol{x}) - 2y_i(\boldsymbol{x})\sum_{j=1}^{M} d_i p(c_j|\boldsymbol{x}) + \sum_{j=1}^{M} d_i^2 p(c_j|\boldsymbol{x})]\right\}$$
$$= E\left\{\sum_{i=1}^{M}[y_i^2(x) - 2y_i(x)E\{d_i|x\} + E\{d_i^2|x\}]\right\} \tag{13}$$

$$= E\left\{\sum_{i=1}^{M}[y_i(\boldsymbol{x}) - E\{d_i|\boldsymbol{x}\}]^2\right\} \tag{14}$$

$$= E\left\{\sum_{i=1}^{M}[y_i(\boldsymbol{x}) - E\{d_i|\boldsymbol{x}\}]^2\right\} + E\left\{\sum_{i=1}^{M} var\{d_i|\boldsymbol{x}\}\right\} \tag{15}$$

where $var\{d_i|\boldsymbol{x}\}E\{d_i^2|\boldsymbol{x}\} - E^2\{d_i|\boldsymbol{x}\}$ is a conditional variance of Δ is achieved by choosing network parameters to minimize the first term of Eq. (15) which is simply the mean-squared error between the network output $y_i(\boldsymbol{x})$ and the conditional expectation of the target outputs. Thus, when network parameters are chosen to minimize a *LMS* cost function, outputs estimate the conditional expectation of the target outputs so as to minimize the mean squared error [8].

5. Shannon's entropy error function

In 1948 Shannon introduced a general uncertainty measure on random variables which takes different probabilities among states into account. For a discrete random variable x, Shannon's entropy is defined as [29,32]:

$$H(x) = -\sum_x f(x)\log_2 f(x) \tag{16}$$

here f is the probability distribution of x. For a continuous random variable x, Shannon Entropy is defined as [43]:

$$H(x) = -\int_{-\infty}^{\infty} f(x)\log f(x)dx \tag{17}$$

where f is the probability density function of x.

In order to compute the Shannon Entropy of the error one must choose a value for the smoothing parameter in the Parzen Window method, that is best suited for a specific data set [30,44,27]. The value of the smoothing parameter is always experimentally selected. One of the problems of pdf estimation using the Parzen Window method, besides the choice of the kernel, is the choice of the smoothing parameter h. The Parzen window estimator does not assume any functional form of the unknown pdf, as it allows its shape to be entirely determined by the data without having to choose a location of the centers. The pdf is estimated by placing a well-defined kernel function on each data point and then determining a common width denoted as the smoothing parameter. In Parzen windowing, the pdf is approximated by a sum of even, symmetric kernels whose centers are translated to the sample points. A suitable and commonly used Kernel function is the Gaussian. The Gaussian function is preferable because it is continuously differentiable, and therefore the sum of Gaussian functions is continuously differentiable on the space of real vectors of any dimension [31,45].

5.1. Shannon Entropy for multilayer perceptrons

Consider a Multilayer Perceptron (MLP) with one hidden layer with output y and a target variable (class membership for each example in the dataset), t. for each example we measure the error using $e(n) = t(n) - y(n)$, $n = 1, 2, \ldots, N$ where N is the total number of examples. We only consider the two-class problem; thus we set $t \in \{-1, 1\}$. The proposed Back-propagation algorithm does not use expression (17) directly as a cost function, but, instead it uses a Shannon's entropy estimator with mean square consistency given by

$$\hat{H}(E) = -\frac{1}{N}\sum_{n=1}^{N} \log \hat{f}(e(n)) \tag{18}$$

where E is the error (difference) random variable. For the estimation of $f(x)$ we use the nonparametric kernel estimator

$$\hat{f}(e(n)) = \frac{1}{Nh}\sum_{n=1}^{N} K\left(\frac{e(n) - e(l)}{h}\right) \tag{19}$$

where h is the smoothing parameter (bandwidth) of the Standard Gaussian Kernel K given by

$$K(x) = \frac{1}{\sqrt{2\pi}} exp\left(-\frac{1}{2}x^2\right) \tag{20}$$

from (18) and (19) we find that

$$\hat{H}(E) = -\frac{1}{N}\sum_{n=1}^{N} \log \frac{1}{Nh}\sum_{l=1}^{N} K\left(\frac{e(n) - e(l)}{h}\right) \tag{21}$$

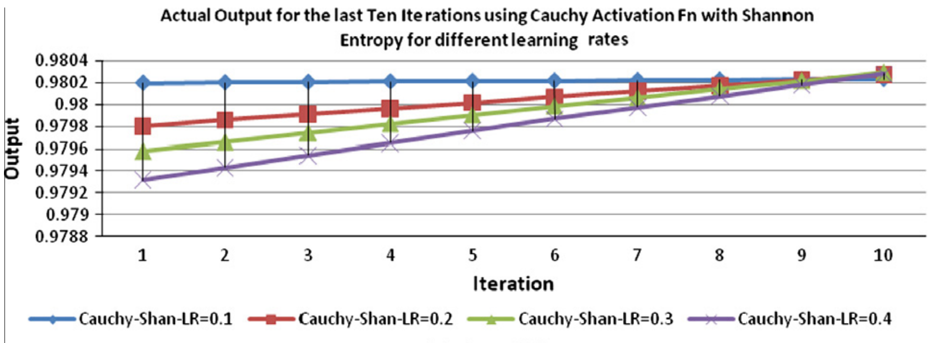
From (20) and (21) we find that:

$$\hat{H}(E) = -\frac{1}{N}\sum_{n=1}^{N} \log \frac{1}{Nh\sqrt{2\pi}}\sum_{l=1}^{N} e^{\left(-\frac{1}{2}\left(\frac{e(n)-e(l)}{h}\right)^2\right)} \tag{22}$$

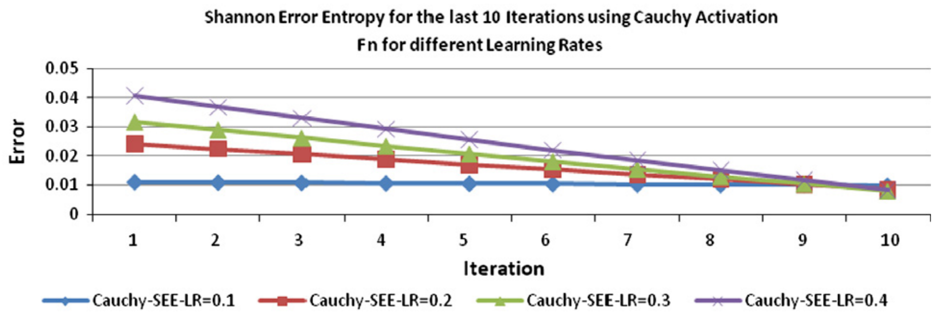
6. Simulated results for Shannon Entropy

In this section, the Shannon Entropy was used as a cost function. Two different activation functions and different learning rates were used to compare the results as follows.

Fig. 1a shows the actual outputs using the different learning rates shown in the figure as an increasing sequences using Shannon Entropy and Cauchy activation function for the last 10 iterations. At the beginning of these iterations, we find that degree of convergence at learning rate (LR) = 0.1 is larger than the degree of convergence at LR = 0.2 which is greater than LR = 0.3 which is greater than its counterpart using LR = 0.4. At the last iteration, we find that the four points nearly coincide. Fig. 1b shows the Shannon Error Entropy (SEE) using Cauchy activation function at different learning rates for the last 10 iterations. They represent a decreasing sequences towards the stopping criterion = 0.01. At the



(a) Actual Output.



(b) Shannon Error Entropy.

Figure 1 Actual output and Shannon Error Entropy using different learning rates and Cauchy activation function for the last 10 iterations. Smoothing parameter $h = 0.01$.

Table 1 Comparing degree of convergence, number of iterations and Shannon Error Entropy for different learning rates using Tanh and Cauchy activation functions.

Learning rate	Comparing Tanh and Cauchy activation function using Shannon Error Entropy					
	Degree of convergence %		Number of iterations		Shannon Error Entropy	
	Tanh	Cauchy	Tanh	Cauchy	Tanh	Cauchy
0.1	98.026186	98.0231	343	2782	0.00891643	0.009919
0.2	98.0266689	98.0273	81	335	0.008757602	0.008549
0.3	98.0468748	98.029	56	190	0.002146134	0.007977
0.4	98.1310499	98.0278	62	161	−0.024664148	0.00838

beginning of these last 10 iterations, we find that at the first iteration, SEE at learning rate = 0.1 < SEE at learning rate = 0.2 < SEE at learning rate = 0.3 < SEE at learning rate = 0.4.

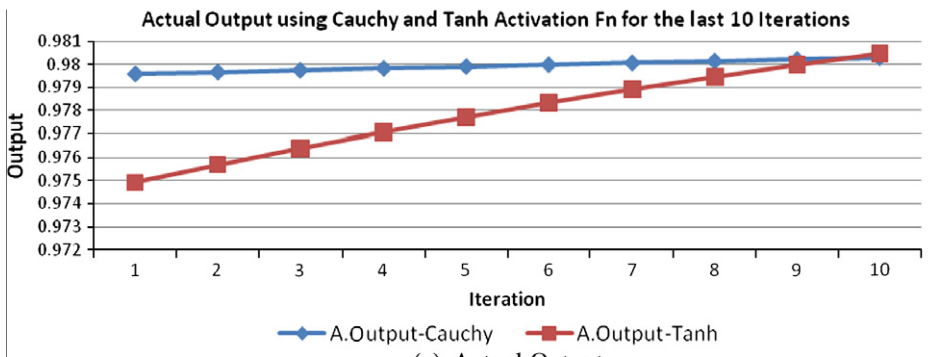
Table 1 compares the degree of convergence, number of iterations and Shannon Error Entropy for different learning rates shown in the table with Tanh and Cauchy activation functions. The table shows that:

1. The maximum degree of convergence = 98.1310499 using Tanh activation function for learning rate = 0.4, witch speeds the convergence rate (number of iterations = 62).
2. For learning rate = 0.1, the Cauchy activation function produces maximum number of iterations (slowing the convergence rate).
3. For different learning rates, the Tanh activation function speeds the convergence rate than using Cauchy activation function.

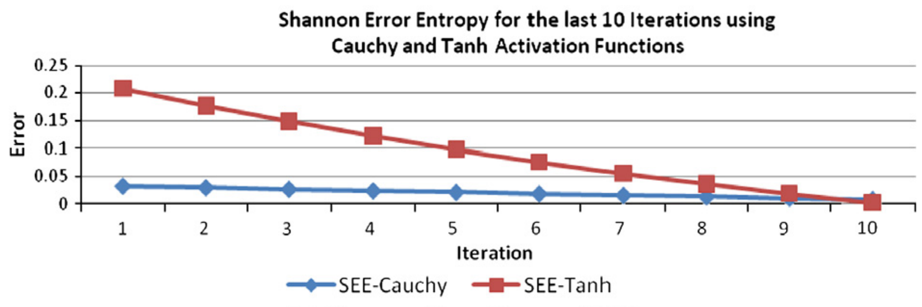
Fig. 2a shows an increasing sequence of the actual outputs for the last 10 iterations using Cauchy and Tanh activation functions. At the beginning of these iterations, the actual output using Cauchy activation function is more converged than using Tanh activation function until the last iteration were the two actual outputs nearly coincide. Fig. 2b shows a decreasing sequence for Shannon Error Entropy using both Cauchy and Tanh activation functions until the stopping criteria met. At the beginning of these iterations, the SEE using Cauchy activation function is low than its counterparts using Tanh activation function.

Table 2 shows an increasing sequence of the degree of convergence for learning rate = 0.1 and learning rate = 0.2 for Cauchy activation function. At the last iteration, the degree of convergence for learning rate = 0.2 is higher than its counterpart using learning rate = 0.1.

Table 3 shows the degree of convergence and Shannon Error Entropy for the last 10 iterations for learning rate = 0.3 and smoothing parameter = 0.01. From the table, we find that



(a) Actual Outputs.



(b) Shannon Error Entropy(SEE).

Figure 2 Actual outputs and SEE using Cauchy and Tanh Activation Functions for the last 10 iterations – Shannon Entropy for learning rate = 0.3, h = 0.01.

Table 2 Degree of convergence and Shannon Error Entropy using learning rate = 0.1, 0.2. Smoothing parameter h = 0.01 and Cauchy activation function.

Using Cauchy activation function – Shannon Entropy					
Degree of convergence %		Degree of convergence %		Shannon Error Entropy	
IT. No.	LR = 0.1	IT. No.	LR = 0.2	LR = 0.1	LR = 0.2
2772	98.0195453	325	97.9802508	0.011104711	0.024203534
2773	98.0199458	326	97.9856377	0.010972519	0.022392591
2774	98.0203461	327	97.9909837	0.010840435	0.020600173
2775	98.0207461	328	97.9962893	0.010708457	0.018826004
2776	98.0211459	329	98.0015551	0.010576586	0.017069812
2777	98.0215455	330	98.0067814	0.010444823	0.01533133
2778	98.0219448	331	98.0119688	0.010313165	0.013610299
2779	98.0223439	332	98.0171178	0.010181615	0.011906461
2780	98.0227427	333	98.0222287	0.010050171	0.010219566
2781	98.0231413	334	98.0273022	0.009918833	0.008549368

Table 3 Degree of convergence and Shannon Entropy Error using Cauchy and Tanh activation functions for the last 10 iterations where LR = 0.3 and the smoothing parameter = 0.01.

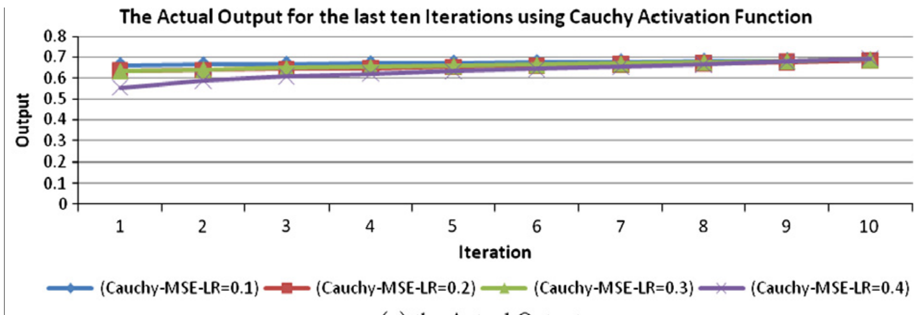
Shannon Entropy for the last 10 iterations with learning rate = 0.3					
Degree of convergence %		Degree of convergence %		Shannon Error Entropy	
IT. No.	Cauchy	IT. No.	Tanh	Cauchy	Tanh
180	97.9577778	46	97.4930825	0.031810609	0.207974248
181	97.9660596	47	97.568692	0.028997422	0.176859505
182	97.9742467	48	97.6401721	0.026227664	0.148320265
183	97.9823408	49	97.7078228	0.023500361	0.122094312
184	97.9903435	50	97.7719152	0.020814569	0.097951467
185	97.9982566	51	97.8326948	0.018169369	0.075688957
186	98.0060815	52	97.8903851	0.01556387	0.055127547
187	98.01382	53	97.94519	0.012997204	0.036108302
188	98.0214736	54	97.9972962	0.01046853	0.018489861
189	98.0290437	55	98.0468748	0.007977032	0.002146134

Table 4 Statistics for different learning rates using Shannon Entropy and Tanh activation function for the last 10 iterations.

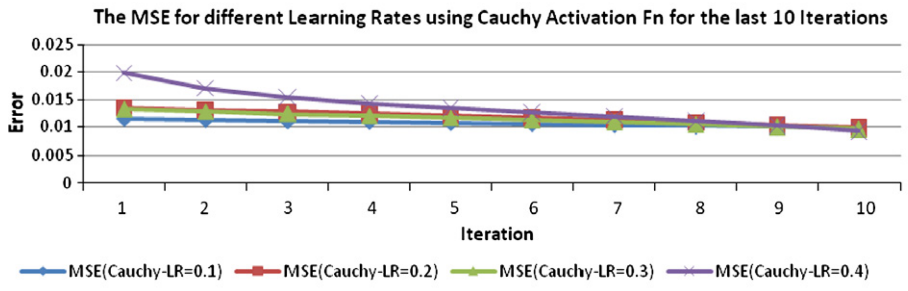
	Actual output using Shannon Entropy and Tanh activation function			
Statistics	LR = 0.1	LR = 0.2	LR = 0.3	LR = 0.4
Mean	0.979981818	0.978820582	0.977894126	0.965115363
STD	0.000189985	0.001016079	0.001860713	0.009937387
Max	0.98026186	0.980266689	0.980468748	0.981310499
Min	0.979697098	0.977244561	0.974930825	0.954287507

Table 5 Statistics for the actual output using different learning rates (Shannon Entropy and Cauchy activation functions).

	Actual output using Shannon Entropy and Cauchy activation function			
Statistics	LR = 0.1	LR = 0.2	LR = 0.3	LR = 0.4
Mean	0.980213447	0.980040116	0.979939444	0.979807461
STD	1.20972E-05	0.000158277	0.000239721	0.000323086
Max	0.980231413	0.980273022	0.980290437	0.980278168
Min	0.980195453	0.979802508	0.979577778	0.979317609



(a) the Actual Outputs.



(b) MSE

Figure 3 The actual outputs and MSE for the last 10 iterations using MSE and Cauchy activation function for different learning rates.

Table 6 Comparing the no. of iterations, degree of convergence and Mean Square Error using Cauchy and Tanh activation function.

Learning rate	Degree of convergence %		No. of Iterations		MSE	
	Cauchy	Tanh	Cauchy	Tanh	Cauchy	Tanh
0.1	68.5662297	68.4983055	72	19	0.009880819	0.009923568
0.2	68.449192	68.527031	18	11	0.009954535	0.009905478
0.3	68.947557	68.9775886	13	3	0.009642542	0.0096239
0.4	69.3911665	69.6624284	10	2	0.009369007	0.009203682

the degree of convergence is an increasing sequence but Shannon Error Entropy is a decreasing sequence until the stopping criterion met. The last iteration shows that the degree of convergence using tanh is higher than the degree of convergence using Cauchy activation function.

Table 4 shows a statistic including the mean, Standard Deviation, Max and Min for the different learning rates shown in the table using Tanh activation function. It shows that:

1. The mean value of the actual outputs using learning rate = 0.1 is the largest while for learning rate = 0.4 it is the lowest.
2. The largest maximum value of the actual output occurs when learning rate = 0.4 and it has also the slowest minimum value.

Table 5 shows a statistic contains Mean, Standard Deviation Max and Min. the table shows that:

1. The Mean value for learning rate = 0.1 is the largest.
2. The maximum value for learning rate = 0.3 is the largest.
3. The minimum value for learning rate = 0.4 is the lowest.

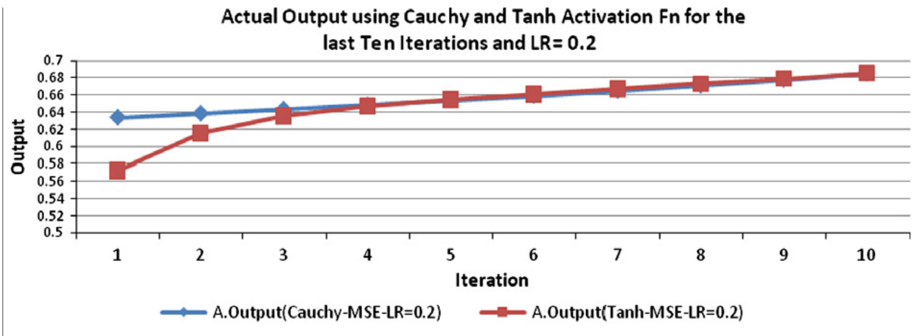
7. Simulated results for Mean Square Error

In this section, the Mean Square Error was used as a cost function. Two different activation functions and different learning rates were used to compare the results as follows.

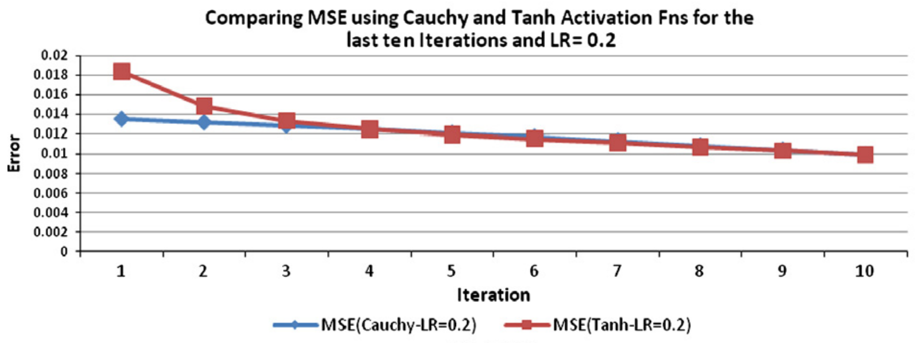
Fig. 3a shows the actual output using the last 10 iterations and Cauchy activation function for different learning rates shown in the figure. All represents an increasing sequence of the actual outputs until the last iteration. At the beginning of these last 10 iterations, we find that for learning rate = 0.1 the actual output is the nearest to the desired output than all the others. At the last iteration all the four curves are nearly coincide. Fig. 3b showing a decreasing sequences until the stopping criteria met. At the beginning of these iterations, we find that the MSE using learning rate = 0.1 is the lowest. At the last iteration all the errors nearly coincide.

Table 6 compares the degree of convergence, the number of iterations and the Mean Square Error for both Cauchy and Tanh activation functions. It shows that:

1. Tanh activation function produces number of iterations less than its counterpart using Cauchy activation function for all learning rates.



(a) The Actual Outputs.



(b) MSE

Figure 4 Actual outputs and MSE using Cauchy and Tanh activation functions for the last 10 Iterations when learning rate = 0.2.

Table 7 Actual output and MSE for Cauchy and Tanh activation functions for the last 10 iterations when the learning rate = 0.2.

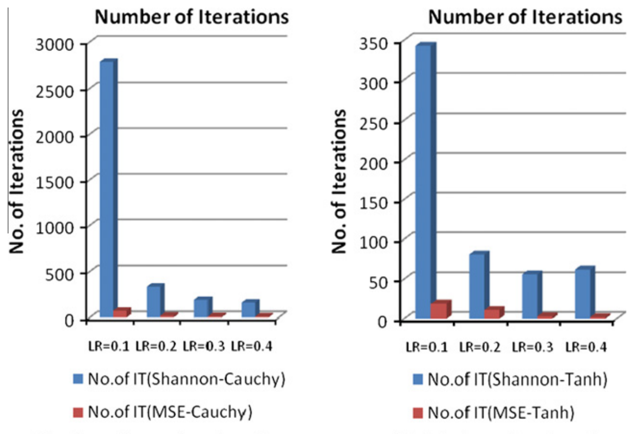
Learning rate = 0.2					
Actual output		Actual output		MSE	
IT. No.	Cauchy	IT. No.	Tanh	Cauchy	Tanh
7	0.633448978	0	0.572145876	0.013435965	0.018305915
8	0.638023957	1	0.615592769	0.013102666	0.014776892
9	0.642779943	2	0.635498227	0.012760617	0.013286154
10	0.647773512	3	0.646757653	0.01240635	0.012478016
11	0.653048635	4	0.654527756	0.012037525	0.011935107
12	0.658638736	5	0.6609408	0.011652751	0.011496114
13	0.664567775	6	0.666932166	0.011251478	0.011093418
14	0.670850646	7	0.672903447	0.01083393	0.010699215
15	0.677493174	8	0.679005457	0.010401065	0.01030375
16	0.68449192	9	0.68527031	0.009954535	0.009905478

2. The learning rate = 0.4 speeds the convergence rate than all other learning rates.
3. When the learning rate increases, the number of iterations decreases for both activation functions.
4. The degree of convergence is the largest when learning rate = 0.4 using Tanh activation function.

Fig. 4a shows an increasing sequences for the actual outputs using both Cauchy and Tanh activation functions for the last 10 iterations and learning rate = 0.2. At the beginning of these last 10 iterations, we find that Cauchy activation function causes a more convergence to the actual output to the desired output than using Tanh activation function. At the last iteration the actual output nearly coincide to each other. Fig. 4b shows the Mean Square Error using Cauchy and Tanh activation functions. At the beginning of these iterations, we find that the MSE using Cauchy is less than the MSE using Tanh activation function. At the last iteration they are nearly coincide.

Table 7 shows the actual output and Mean Square Error using learning rate = 0.2 for the last 10 iterations. It follows that:

1. The actual outputs represents an increasing sequence until the last iteration.
2. The Mean Square Error represents a decreasing sequence until the stopping criterion met.



(a) Cauchy activation Fn. (b) Tanh activation Fn.

Figure 5 Comparing the number of iterations using Shannon Entropy and MSE.

3. The last iteration indicates that the degree of convergence using Tanh activation function is greater than its counterpart using Cauchy activation function.
4. The MSE using Tanh activation function is less than the MSE using Cauchy activation function.

8. Results for comparing Shannon Entropy and Mean Square Error

In this section, the Shannon Entropy and MSE were used as a cost function. Two different activation functions and different learning rates were used to compare the results as follows.

Fig. 5a compares the number of iterations using Cauchy activation function between the Shannon Entropy and MSE and shows that the number of iterations using MSE is less than the number of iterations using Shannon Entropy. While Fig. 5b compares the number of iterations using Tanh activation function between the Shannon Entropy and MSE and shows that the number of iterations using MSE also less than the number of iterations using Shannon Entropy.

Table 8 compares the number of iterations using Shannon Entropy and Mean Square Error for different learning rates with Cauchy and Tanh activation functions. It shows that:

1. Generally the number of iterations decreases when the learning rates in the table increases except for learning rate = 0.4 using Shannon Entropy.
2. MSE speeds the rate of convergence than Shannon Entropy for both Cauchy and Tanh activation functions for each learning rate.
3. For learning rate = 0.1, Shannon Entropy produces the largest number of iterations.

Table 10 Comparing the degree of convergence and the error using Cauchy activation function for learning rate = 0.1 for the last 10 iterations.

Cauchy activation function using learning rate = 0.1			
Degree of convergence (%)		Error	
Shannon	MSE	Shannon	MSE
98.0195453	66.116323	0.011104711	0.011481036
98.0199458	66.3846572	0.010972519	0.011299913
98.0203461	66.6543839	0.010840435	0.011119301
98.0207461	66.9253177	0.010708457	0.010939346
98.0211459	67.1972758	0.010576586	0.010760187
98.0215455	67.4700779	0.010444823	0.010581958
98.0219448	67.7435467	0.010313165	0.010404788
98.0223439	68.0175081	0.010181615	0.010228798
98.0227427	68.2917914	0.010050171	0.010054105
98.0231413	68.5662297	0.009918833	0.009880819

Table 8 Comparing the number of iterations for different learning rates using Cauchy and Tanh activation functions.

Comparing the number of iterations using Shannon Entropy and MSE				
Learning rate	Cauchy		Tanh	
	Shannon Entropy	MSE	Shannon Entropy	MSE
0.1	2782	72	343	19
0.2	335	18	81	11
0.3	190	13	56	3
0.4	161	10	62	**2**

Table 9 Comparing the degree of convergence using Cauchy and Tanh activation functions (Shannon Entropy and MSE) for different learning rates.

Comparing the degree of convergence (%) using Shannon Entropy and MSE				
Learning rate	Cauchy		Tanh	
	Shannon Entropy	MSE	Shannon Entropy	MSE
0.1	98.0231	68.5662297	98.026186	68.4983055
0.2	98.0273	68.449192	98.0266689	68.527031
0.3	98.029	68.947557	98.0468748	68.9775886
0.4	98.0278	69.3911665	98.1310499	69.6624284

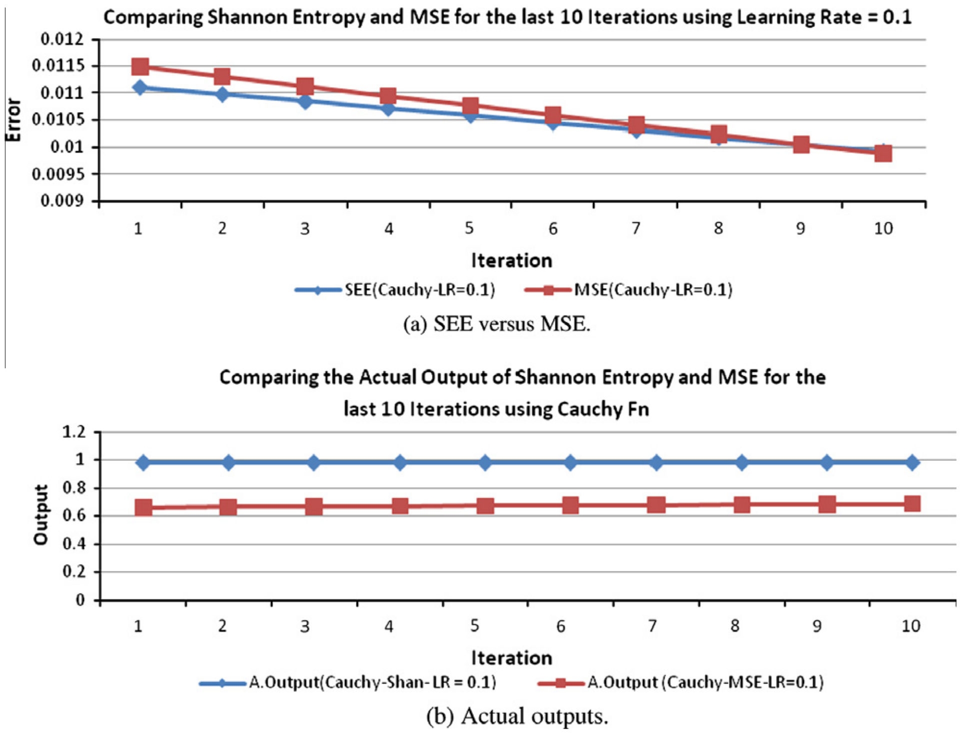


Figure 6 Comparing the actual outputs and SEE versus MSE using Cauchy activation function for learning rate = 0.1.

4. For learning rate = 0.4, MSE produces the smallest number of iterations.

Table 9 compares the degree of convergence for Shannon Entropy and MSE using the two activation functions Tanh and Cauchy. It shows that:

1. For all learning rates the degree of convergence using Shannon Entropy is greater than its counterpart using MSE.
2. The largest degree of convergence occurred by Shannon Entropy when learning rate = 0.4 using Tanh activation function.

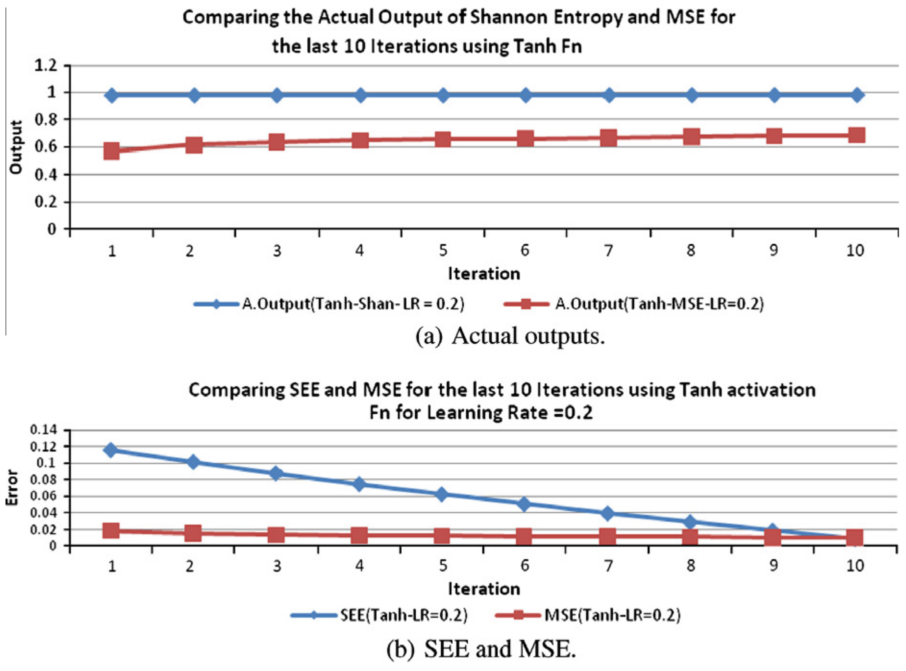


Figure 7 Comparing the actual outputs and SEE versus MSE using Tanh activation function for learning rate = 0.1.

3. For MSE, the smallest degree of convergence occurs when learning rate = 0.1 using Tanh activation function.

Table 10 compares the degree of convergence and the error using Shannon Entropy and MSE for the last 10 iterations using Cauchy activation function and learning rate = 0.1. The table shows that:

1. The degree of convergence represented as an increasing sequence until the last iteration met.
2. The Error represented as a decreasing sequence until the stopping criteria met.
3. The degree of convergence using Shannon Entropy is more than its counterpart using MSE.

Fig. 6a represents a decreasing sequence for SEE and MSE using Cauchy activation function and learning rate = 0.1. At the beginning of these iterations, we find that the SEE is less than the MSE. At the last iteration they represent two coincide points. Fig. 6b shows that the actual output using Shannon Entropy is larger than its counterpart using MSE for every iteration.

Fig. 7a shows that the actual output using Shannon Entropy is larger than its counterpart using MSE for every iteration. Fig. 7b represents a decreasing sequence for SEE and MSE using Tanh activation function and learning rate = 0.2. At the beginning of these iterations, we find that the MSE is less than the SEE. At the last iteration they represent nearly two coincide points.

Table 11 compares the degree of convergence and the error using Shannon Entropy and MSE for the last 10 iterations using Tanh activation function and learning rate = 0.2. The table shows that:

1. The degree of convergence represented as an increasing sequence until the last iteration met.
2. The Error represented as a decreasing sequence until the stopping criteria met.
3. The degree of convergence using Shannon Entropy is more than its counterpart using MSE.

Table 11 the degree of convergence and SEE versus MSE using Shannon Entropy and MSE for the last 10 iterations using learning rate = 0.2.

Tanh activation function for learning rate = 0.2			
Degree of convergence %		Error	
Shannon	MSE	SEE	MSE
97.7244561	57.2145876	0.115762983	0.018305915
97.7626102	61.5592769	0.101414051	0.014776892
97.7994916	63.5498227	0.087774391	0.013286154
97.8351606	64.6757653	0.074798755	0.012478016
97.869674	65.4527756	0.062445377	0.011935107
97.903085	66.09408	0.050675664	0.011496114
97.935444	66.6932166	0.03945391	0.011093418
97.9667983	67.2903447	0.02874705	0.010699215
97.9971926	67.9005457	0.01852443	0.01030375
98.0266689	68.527031	0.008757602	0.009905478

9. Conclusions

Slow convergence and long training times are still the disadvantages mentioned using neural networks. In this paper, we present a comparative study between Shannon Entropy and Mean Square Errors for improving the training efficiency (degree of convergence and convergence rate) of Multilayer Backpropagation Neural Network Algorithms. Minimizing the Error using MSE was derived. Shannon Entropy Estimator was also derived. Several concluding remarks were obtained in Sections 6–8. We also outline the following conclusions:

1. Mean Square Error speeds the convergence than Shannon Entropy.
2. The degree of convergence using Shannon Entropy is higher than the degree of convergence using MSE.
3. Generally speaking, all the number of iterations using Shannon Entropy is greater than the number of iterations using MSE for all activation functions and all learning rates.
4. The actual output using Shannon Entropy is greater than the actual output using MSE for arctangent and Cauchy activation functions and all learning rates.
5. The arctangent activation function increases the convergence rate (speeds the convergence) than the Cauchy activation function using both Shannon Entropy and MSE, since the number of iterations using arctangent activation function is less than the number of iterations using Cauchy activation function for Shannon Entropy and MSE and all learning rates.
6. When the learning rate increases, the number of iterations decreases for Shannon Entropy and MSE cost functions and also for arctangent and Cauchy activation functions. On the other hand, increasing the learning rate speeds the convergence of the Backprobagation Neural Network for the cost and activation functions.
7. For the same learning rate, the number of iterations using arctangent activation function is less than the number of iterations using Cauchy activation function.

References

[1] Wu W, Wang J, Cheng M, Li Z. Convergence analysis of online gradient method for BP neural networks. Neural Networks 2011;24:91–8.

[2] Shihab K. A backpropagation neural network for computer network security. J Comput Sci 2006;2(9):710–5.

[3] Bodyanskiy Y, Pliss I, Slipchenko O. Growing neural networks using nonconventional activation functions. Int J Inform Theor Appl 2007;14.

[4] Shrestha R, Theobald S, Nestmann F. Simulation of flood flow in a river system using artificial neural networks. Hydrol Earth Syst Sci 2005;9(4):313–21.

[5] Zweiri Y. Optimization of a three-term backpropagation algorithm used for neural network learning. Int J Inform Math Sci 2007.

[6] Ventresca M, Tizhoosh H. Improving the convergence of backpropagation by opposite transfer functions. In: International Joint Conference on Neural Networks, Vancouver, BC, Canda, July 16–21, 2006.

[7] Zainuddin Z, Mahat N, Hassan Y. Improving the convergence of the backpropagation algorithm using local adaptive techniques. World Acad Sci Eng Technol 2005;1.

[8] Rady H. Classification of multilayer neural networks using cross entropy and mean square errors. El Shorouk J ACM – Adv Comput Sci 2008.
[9] Zhang G. Neural networks for classification: a survey. IEEE Trans Syst Man Cyber – Part C: Appl Rev 2000;30(4).
[10] Abd El-Wahed W, Zaki E, El-Refaey A. Artificial immune system based neural networks for solving multi-objective programming problems. Egyptian Inform J 2010;11(2):56–65.
[11] Shao H, Zheng G. Boundedness and convergence of online gradient method with penalty and momentum. Neurocomputing 2011;74:765–70.
[12] Sun J. Local coupled feedforward neural network. Neural Networks 2009.
[13] Rajapandian V, Gunaseeli N. Modified standard backpropagation algorithm with optimum initialization for feedforward neural networks. Int J Imaging Sci Eng 2007;1(3).
[14] Ahlawat A, Pandey S. A variant of back-propagation algorithm for multilayer feed-forward network. In: International conference (information research & applications); 2007.
[15] Alsmadi M, Omar K, Noah S. Back propagation algorithm: the best algorithm among the multi-layer perceptron algorithm. IJCSNS Int J Comput Sci Network Security 2009;9(4).
[16] Ghash-Dastidar S, Adeli H. A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection. Neural Networks 2009;22:1419–31.
[17] Shao H, Zheng G. Convergence analysis of a back-propagation algorithm with adaptive momentum. Neurocomputing 2011;74:749–52.
[18] Iranmanesh S, Mahdavi M. A differential adaptive learning rate method for back-propagation neural networks. World Acad Sci Eng Technol 2009;50.
[19] Krissilov A, Krissilov D, Oleshko N. Application of the sufficiency principle in acceleration of neural networks training. Int J Inform Theor Appl 1997;10.
[20] Meynet J, thiran J. Information theoretic combination of pattern classifiers. Pattern Recogn 2010;43:3412–21.
[21] Erdogmus D, Principe J, Kim S, Sanchez J. A recursive Renyi's entropy estimator. IEEE; 2002.
[22] Alexandre L, Sa J. Error entropy minimization for LsTM training. In: ICANN. Berlin, Heidelberg: Springer-Verlag; 2006. p. 244–53.
[23] Nasr G, Badr E, Joun C. Cross entropy error function in neural networks: forecasting gasoline demand. In: Proceedings FLAIRS-02. American Association for Artificial Intelligence; 2002. <www.aaai.org>.
[24] Erdogmus Santamaria D, Principe J. Entropy minimization for supervised digital communications channel equalization. IEEE Trans Signal Process 2002;50(5).
[25] Erdogmus D, Principe J. An error-entropy minimization algorithm for supervised training of nonlinear adaptive systems. IEEE Trans Signal Process 2002;50(7).
[26] Xu J, Erdogmus D, Principe J. Minimum error entropy Luenberger observer. In: American control conference, Portland, OR, USA, June 8–10, 2005.
[27] Nawi N, Ghazali R, Salleh M. An approach to improve back-propagation algorithm by using adaptive gain. Biomed Soft Comput Human Sci 2010;16(2):125–34.
[28] Al Bayati A, Sulaiman N, Sadiq G. A modified conjugate gradient formula for back propagation neural algorithm. J Comput Sci 2009;5(11):849–56.
[29] Silva L, Sa M, Alexandre L. Neural network classification using Shannon's Entropy. In: Proceedings of the European symposium on artificial neural networks, ESANN' 2005, Bruges, Belgium, 27–29 April 2005.
[30] Ismail Z, Jamaluddin F. A backpropagation method for forecasting electricity load demand. J Appl Sci 2008;8(13):2428–34.
[31] William P, Hoffman M. Error entropy and mean square error minimization for lossless images compression. IEEE; 2006.
[32] Bromiley P, Thacker N, Thacker E. Shannon entropy, Renyi entropy, and information. Tina 2004.
[33] Mellitus. Improved gradient descent back propagation neural networks for diagnoses of type II diabetes. Global J Comput Sci Technol 2010;9(5 Ver 2.0):94–110.
[34] Kiranyaz S, Ince T, Yildirim A, Gabbouj M. Evolutionary artificial neural networks by multi-dimensional particle swarm optimization. Neural Networks 2009;22:1448–62.
[35] Srinivasan V, Eswaran C, Sriraam N. Approximate entropy-based epileptic EEG detection using artificial neural networks. IEEE Trans Inform Technol Biomed 2007;11(3).
[36] Rady H. A comparative approach to accelerate backpropagation neural network learning using different activation functions. El Shorouk J ACM – Adv Comput Sci 2009.
[37] Gross B, Hanna D. Artificial neural networks capable of learning spatiotemporal chemical diffusion in the cortical brain. Pattern Recogn 2010;43:3910–21.
[38] Rady H. Different aspects for enhancing the backpropagation neural networks. El Shorouk J ACM – Adv Comput Sci 2007.
[39] Nawi N, Ransing R, Ransing M. An improved conjugate gradient based learning algorithm for back propagation neural networks. Int J Comput Intell 2007;4(1).
[40] Kandil M, Mohamed F, Saleh F, Fayek M. A new approach for optimizing backpropagation training with variable gain using Pso. In: GVIP 05 Conf. Cairo (Egypt): CICC, 19–21 December 2005.
[41] Debes K, Koenig A, Gross H. Transfer functions in artificial neural networks. Brains Minds Media 2005.
[42] Yu C, Liu B. A backprobagation algorithm with adaptive learning rate and momentum coefficient. IEEE; 2002.
[43] Erdogmus D, Rao Y, Principe J. Recursive least squares for an entropy regularized MSE cost function. In: European symposium on artificial neural networks, Bruges, Belgium, 23–25 April 2003. p. 451–6.
[44] Espindola G, Pantaleao E. Parzen windows and nonparametric desnsity estimation applied in high resolution imagery classification. In: Anais XIV Simoio Brasileiro de Sensoriamento Remoto, Natal, Brasil, 25–30 abril. INPE. p. 6869–73.
[45] Shwartz S, Zibulevsky M, Schechner Y. Fast kernel entropy estimation and optimization. Signal Process 2005;85:1045–58.