Performance Evaluation of Complex Valued Neural Networks Using Various Error Functions

Anita S. Gangal, P. K. Kalra, and D. S. Chauhan

Abstract—The backpropagation algorithm in general employs quadratic error function. In fact, most of the problems that involve minimization employ the Quadratic error function. With alternative error functions the performance of the optimization scheme can be improved. The new error functions help in suppressing the ill-effects of the outliers and have shown good performance to noise. In this paper we have tried to evaluate and compare the relative performance of complex valued neural network using different error functions. During first simulation for complex XOR gate it is observed that some error functions like Absolute error, Cauchy error function can replace Quadratic error function. In the second simulation it is observed that for some error functions the performance of the complex valued neural network depends on the architecture of the network whereas with few other error functions convergence speed of the network is independent of architecture of the neural network.

Keywords—Complex backpropagation algorithm, complex error functions, complex valued neural network, split activation function.

I. INTRODUCTION

In recent years, complex-valued neural networks have widened the scope of application in optoelectronics, imaging, remote sensing, quantum neural devices and systems, spatiotemporal analysis of physiological neural systems, and artificial neural information processing. The generalization of real valued algorithms cannot be simply done as complex valued algorithm. The complex backpropagation algorithm can be applied to multilayered neural networks whose weights, threshold values, inputs and outputs all are complex numbers. Complex version of backpropagation (CVBP) algorithm made its first appearance when Widrow, Mc Cool and Ball [1] announced their complex least mean squares (LMS) algorithm. Kim and Guest [2] published a complex valued learning algorithm for signal processing application. Georgiou and Koutsougeras [3] published another version of CVBP incorporating a different activation function and have shown if real valued algorithms be simply done as complex valued algorithm then singularities and other such unpleasant phenomena may arise. Hirose [4] studied the dynamics of CVNN which was later applied to the problem of reconstructing vectors lying on the unit circle. Benvenuto and Piazza [5] published a different version of CVBP by using different activation function. Wang [6] proposed a complex valued recurrent neural network to solve the complex valued linear equations. A complex activation function for digital VLSI neural networks was implemented by Deville [7] that required lesser hardware than the conventional real valued neural network. A complex valued neural network was used by Smith and Hui[8] to implement a data extrapolation algorithm. Leung and Haykin[9] published the CVBP in which the activation function used was an extended version of sigmoid function and the error function was Quadratic error function. An extensive study of CVBP was reported by Nitta [10]. Decision boundary of a single complex valued neuron consists of two hypersurfaces that intersect orthogonally, and divide a decision region into four equal sections. If both the absolute values of real and imaginary parts of the net inputs to all hidden neurons are sufficiently large, then the decision boundaries for real and imaginary parts of an output neuron in three layered complex valued neural network intersect orthogonally. The average learning speed of complex BP algorithm is faster than that of real BP algorithm. The standard deviation of the learning speed of complex BP is smaller than that of real BP. Hence the complex valued neural network and the related algorithm are natural for learning of complex valued patterns. Werbos and Titus [11] and then Gill and Wright [12] discussed the different consequences of changing error functions in an optimization scheme. Rey [13] has shown that the results could be substantially improved by varying error function in an optimization scheme. He applied absolute error function based optimization to solve a curve fitting problem more efficiently than the standard quadratic error function based optimization. Fernandeze [14] implemented some new error functions for the training of real valued neural network as tools to counter the ill effects of local minima by weighting error functions according to their magnitudes. Matsuoka and Yi [15] used logarithmic error function to eliminate the local minima. Ooyen and Nienhaus [16] have used entropy type error function and have concluded that it’s performance is better than the quadratic error function based backpropagation algorithm for function approximation problems.

II. COMPLEX VALUED NEURAL NETWORK

A three layered complex valued neural network is shown in (1). In this network all the inputs, outputs, weights, and biases are complex values. According to the Liouville’s theorem, a
bounded holomorphic function in the complex plane C is a constant. So the attempt to extend the sigmoidal function to complex plane is met with the difficulty of singularities in the output. To deal with this difficulty Prashant[17] suggested that the input data should be scaled to some region in complex domain. Although the input data can be scaled but there is no limit over the values the complex weights can take hence it is difficult to implement it. To overcome this problem split sigmoidal activation function is used for training the network.

\[ z_j = \phi(u_j) = \frac{1}{1 + e^{-Re(z_j)}} + i \frac{1}{1 + e^{-Im(z_j)}} = Re[z_j] + i Im[z_j] \]  

(2)

Internal potential of output neuron k:

\[ s_k = \sum_{j=1}^{M} (v_{kj} z_j) + \gamma_k = Re[s_k] + i Im[s_k] \]  

(3)

Output of output neuron k:

\[ o_k = \phi(s_k) = \frac{1}{1 + e^{-Re(s_k)}} + i \frac{1 + e^{-Im(s_k)}}{1 + e^{-Im(s_k)}} = Re[y_k] + i Im[y_k] \]  

(4)

Error

\[ e_k = o_k - d_k \]  

(5)

With the help of this error \( e_k \) using different complex error functions the error \( E \) is obtained. Then we derive the gradient of \( E \) w.r.t. both the real and imaginary part of the complex weights.

\[ \nabla_{w_{jk}} E = \frac{\partial E}{\partial Re[w_{jk}]} + i \frac{\partial E}{\partial Im[w_{jk}]} \]  

(6)

During training the network cost function \( E \) is minimized by recursively altering the weight coefficient based on gradient descent algorithm, given by

\[ w_{jk}(p+1) = w_{jk}(p) + \Delta w_{jk}(p) = w_{jk}(p) - \eta \nabla_{w_{jk}} E \]  

(7)

Where ‘\( p \)’ is the number of iterations and ‘\( \eta \)’ is the learning rate constant.

III. VARIOUS ERROR FUNCTIONS

It must be noted that in the description of Error functions, the function’s form has been retained to that of real error functions forms while extending to complex domain. This was done to make sure that the error computed kept the same formula even while operating in the complex domain. This also makes sure that the surface plot of the function is close to the plane plot of the same. We have studied the performance of the complex valued neural networks by using following error functions:

A. The Absolute Error Function

Absolute error is one of several robust functions that display less skewing of error due to the outliers. A small numbers of outliers are less likely to affect the total error and so they do not affect the learning algorithm as severely as mean squared error.

Complex absolute error function is defined to be

\[ E = \sum_{n} abs(e_i) = \sum_{n} \sqrt{(e_i e_i^*)} \]  

(8)

B. Andrew Error Function

Complex Andrew error function is defined as

\[ E = \sum_{n} 1/\pi^2 * \cos(\pi * abs(e_i)); if abs(e_i) \leq 1 \]  

\[ = 0; \text{ else} \]  

(9)
The Andrew error function has a discontinuity at the origin. The point beyond which the function can be suppressed can be chosen as desired while the dynamics of update operate according to a sinusoid by definition to take complex values. The surface plot of the complex definition reveals that the function is rotationally symmetrical about the z-axis.

C. Bipolar Hyperbolic Squared Error Function

Complex Bipolar Hyperbolic Squared error function is defined as

\[ E = \sum_{n} \ln(1 + e_{i}^*) \]  

(10)

The surface is characterized by a unique maximum and rotational symmetry. The surface is differentiable with respect to real and imaginary parts of the complex variables that now appear as argument of the function.

D. Cauchy Error Function

Complex Cauchy error function is given as

\[ E = \sum_{n} \left(1 + \frac{c^2}{e_{i}^*} \right) \]  

(11)

The surface plot shows a unique point of global minimum. This surface is differentiable through the real plane and is rotationally symmetrical about the z-axis. The surface is characterized by changing convexity as the radius vector increases.

E. Fair Error Function

The complex Fair error function is rotationally symmetrical and has one global minimum. The convexity with respect to the xy-plane is maintained everywhere. Complex Fair error function is defined as

\[ E = \sum_{n} c^2 \left[\frac{(abs(e_{i})/c) - \ln(1 + abs(e_{i})/c)}{}\right] \]  

(12)

Where, c is the tuning constant. This function is defined continuous derivatives except at the origin. The tuning constant, c is usually set as 1.3998.

F. Fourth Power Error Function

This function is useful when dealing with the data known to be free from outliers, or in cases where it is important to minimize the worst-case error, rather than the average error. This error function increases more rapidly for errors more than unity as compared to Quadratic error function. The surface is smooth for the derivatives of all orders that exist and is rotationally symmetrical about the z-axis. The complex Fourth Power error function is defined as

\[ E = \sum_{n} \frac{1}{2} \left(e_{i}^* \right)^4 \]  

(13)

G. German McClure Error Function

The German-McClure error function is defined to suppress large errors near the origin. The asymptotes of the function suppress the outliers. This function approximates to Quadratic function for smaller values of error as the denominator can be approximated as unity. The complex German McClure error function is given by

\[ E = \frac{\frac{1}{4} c e_{i}^*/2}{1 + e_{i}^*} \]  

(14)

H. Huber Error Function

The complex Huber error function is given by

\[ E = \frac{\frac{1}{r} c e_{i}^*/2}{1 + e_{i}^*} \]  

if \( |e_{i}| < c \)  

(15)

\[ E = \frac{\frac{1}{r} c (abs(e_{i}) - c / 2)}{1 + e_{i}^*} \]  

if \( |e_{i}| \geq c \)

Where n is the number of outputs and c is the tuning constant. A typical value for c is 1.345. When dealing with noisy data, the training values may contain outliers with unusual deviation from the true underlying function. Huber function can be used to ignore these outliers, or at least reduce the ill effect they have on learning. The function has good effects of Quadratic and Absolute error function.

I. Hyperbolic Squared Error Function

Hyperbolic Squared error needs normalization while running training with one of the backpropagation algorithm or its variants. The complex Hyperbolic Squared error function is given by

\[ E = \frac{\frac{1}{r} c \ln(1 + e_{i}^*)}{1 + e_{i}^*} \]  

(16)

Hyperbolic Squared error is similar in shape to the Bipolar Hyperbolic Squared error function.

J. Log Cosh Error Function

The complex Log-Cosh error function is defined as

\[ E = \frac{\frac{1}{r} c \ln(cosh(e_{i}^*))}{1 + e_{i}^*} \]  

(17)

K. Mean Median Error Function

This has the advantage of both the Mean error function and Median error function. Hence reduces the influence of large errors but at the same time retains its convexity. Complex Mean meridian function is defined as

\[ E = \frac{\frac{1}{r} 2 c \left(\sqrt{1 + e_{i}^*/2} - 1\right)}{} \]  

(18)

L. Minkowski Error Function

The complex Minkowski error function is given as

\[ E = \frac{\frac{1}{r} c \left(abs(e_{i})\right)^\gamma}{1 + e_{i}^*} \]  

(19)

Where n is the number of outputs and typical value of r is chosen as 0.4.

M. Quadratic Error Function

This is the standard error function. Complex Quadratic error function is defined as
\[ E = \sum_{n} \frac{3}{2} e^{r_{i}} e^{i} \]  \hspace{1cm} (20)

N. *Sinh Error Function*

This function is steeper than Quadratic error function and is
symmetrical about the origin hence the update involves
two parts, the first is the gradient in the first quadrant and the
second is the gradient in the third quadrant. In both cases, the
gradient is directed towards the origin.

The complex Sine-Hyperbolic function is given by:
\[ E = \sum_{n} \frac{3}{2} \text{Sinh}(|e|) \] \hspace{1cm} (21)

O. *Turkey Biweight Error Function*

Turkey Biweight error function reduces the effect of large
errors and suppresses the outliers. Hence the contribution of
an outlier to this error function is smaller. The complex
Turkey Biweight error function is defined as
\[ E = \sum_{n} \frac{c^{2}}{6} \left[ 1 - \left( 1 - \left( |e|^2 / c^2 \right) \right) \right] \] \hspace{1cm} (22)

Where \( c \) is the tuning constant and its typical value is 4.6851.

P. *Welsch Error Function*

The complex Welsch error function is defined as
\[ E = \sum_{n} \frac{c^{2}}{2} \left[ 1 - \exp \left( -i(e_{i}^{*} / c^2) \right) \right] \] \hspace{1cm} (23)

This function reduces the effect of large errors. The typical
value of the tuning constant \( c = 2.9846 \).

IV. SIMULATION

A. *Simulation 1*

For the first experiment we have taken three layers complex
valued neural network with architecture 2-5-1. Nitta [9] has
defined the complex XOR gate in complex domain as:
1. The real part of the output is unity if the first input is
equal to the second input else it is zero
2. The imaginary part of the output is unity if the
second input is equal to unity else it is zero

We trained the complex valued neural network with the
first eight patterns of Table I. Then for the testing we have
done simulation on all the sixteen patterns. The experiments
were run twice and average values of the epochs with various
error functions are shown in Table II. The learning rate
parameter is chosen 1.3 for all the simulations and the target
error is set to 0.001.

In case of absolute error function the absence of index (the
power, unlike the quadratic error function) is a distinguishing
feature of this error function as this enables smoothing out the
ill-effects of the outlier points that would otherwise have
offset the best-fit of the optimization scheme. It can also be
noted that the function form in fact is the quadric cone. The
update rule for the complex valued neural network steers the
real part and the imaginary part of the weights to the minima
separately. The problem of local minima exists in general with
this error function based algorithm. The initial weights and the
learning parameter decide how the training should progress.
The dynamics of real part depends not only on the real part of
the weights but also on the imaginary parts as the updates
(6),(7) of the real and imaginary parts are dependent on each
other. This complex error function is not differentiable at the origin.

TABLE I

<table>
<thead>
<tr>
<th>Input 1</th>
<th>Input 2</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>i</td>
<td>i</td>
</tr>
<tr>
<td>i</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>i</td>
<td>i</td>
<td>1+i</td>
</tr>
<tr>
<td>i</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1+i</td>
</tr>
<tr>
<td>1+i</td>
<td>i</td>
<td>i</td>
</tr>
<tr>
<td>1+i</td>
<td>1+i</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>i</td>
</tr>
<tr>
<td>0</td>
<td>1+i</td>
<td>0</td>
</tr>
<tr>
<td>i</td>
<td>1+i</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0+i</td>
<td>i</td>
</tr>
<tr>
<td>1</td>
<td>1+i</td>
<td>0</td>
</tr>
<tr>
<td>1+i</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1+i</td>
<td>1</td>
<td>i</td>
</tr>
</tbody>
</table>

The complex Bipolar hyperbolic has unique maxima, and
hence the training process should steer the network so as to
attain the maxima of the function. While implementing the
sign of the function is reversed so the update runs in
accordance with accepted conventions. While implementing
the Complex Andrew error function, the training was directed
towards the minima. In case of Cauchy error function, the
training steers the weights so as to reach the minimum of the function.

For fourth power error function, the weight update is more
rapid for error values greater than unity, and the rate of
training is diminished for fractional errors, lying in the
interval [0,1]. The cube term that results from the form of the
error function enhances the update if the error is greater than
unity and suppresses it if error is fractional. Complex German-
McClure error function is just the Quadratic function for
smaller values of the errors, while for the large values the
denominator comes into play, and the function deviates from
being quadratic. The Huber error function is defined piece-
wise. The characteristic feature of the function is that it
involves both the Quadratic error function and the Absolute
error function. The parameter \( c \) is the point of demarcation to
assign a domain of operation for each error function. The function enables one to optimally choose error functions. If the data is prone to outliers and if their scatter is biased to one side, then an obvious choice would be to suppress the influence of these spurious points by assigning an Absolute error function to this side, and set Quadratic function to operate on the other side. It was found that in statistical analysis such choice improved the results as a judicial assignment was proven to be effective. The function is a paraboloid of revolution for the part of definition that is quadratic and for the part that is Absolute function it is a cone.

For complex Hyperbolic Squared error function the surface is characterized by a unique maximum at the origin to which the training process should steer the network error. During implementation of the function a negative sign was prefixed to the error function, and the usual gradient was developed for the function. The complex Mean-Median error function behaves like the quadratic error function for smaller complex errors, and as absolute error function for large errors. This function finds best application for data that are prone to outliers and if their scatter is biased to one side, then an obvious choice would be to suppress the influence of these spurious points by assigning an Absolute error function to this side, and set Quadratic function to operate on the other side. It was found that in statistical analysis such choice improved the results as a judicial assignment was proven to be effective. The function is a paraboloid of revolution for the part of definition that is quadratic and for the part that is Absolute function it is a cone.

### TABLE II

<table>
<thead>
<tr>
<th>Error function</th>
<th>Average Epochs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Absolute error</td>
<td>12100</td>
</tr>
<tr>
<td>Andrew error</td>
<td>28000</td>
</tr>
<tr>
<td>Cauchy error</td>
<td>17600</td>
</tr>
<tr>
<td>Fourth order error</td>
<td>26300</td>
</tr>
<tr>
<td>Fair error</td>
<td>21000</td>
</tr>
<tr>
<td>German McClure error</td>
<td>27700</td>
</tr>
<tr>
<td>Huber error</td>
<td>24900</td>
</tr>
<tr>
<td>Hyperbolic Squared</td>
<td>27600</td>
</tr>
<tr>
<td>Bipolar Hyperbolic</td>
<td>27000</td>
</tr>
<tr>
<td>Log Cosh error</td>
<td>14500</td>
</tr>
<tr>
<td>Mean Median error</td>
<td>15000</td>
</tr>
<tr>
<td>Minkowski error</td>
<td>17600</td>
</tr>
<tr>
<td>Quadratic error</td>
<td>31300</td>
</tr>
<tr>
<td>Sinh error</td>
<td>16900</td>
</tr>
<tr>
<td>Tukey error</td>
<td>22100</td>
</tr>
<tr>
<td>Welsch error</td>
<td>24300</td>
</tr>
</tbody>
</table>

As clear from the results shown in Table III complex valued neural networks with Absolute error function, Andrew error function, Fourth power error function, Logarithmic error function, Turkey error function, and Welsch error function are sensitive to the architecture. The number of iterations for convergence during training varies widely with the architecture. On the other hand, the Cauchy error function, Huber error function, Log-Cosh error function, Mean-Median error function, Quadratic error function, and Sinh error function have been robust and have shown little dependence on the architecture. Hence, the extra neurons and extra weights are not necessary as smaller architecture shows the similar performance and could solve the problem.
V. CONCLUSION

The above simulations reveal that the error functions can indeed be treated as a parameter for training complex valued neural network. Absolute error, Cauchy error, Fair error, Mean-Median error, Fourth error functions can replace Quadratic error function depending on the applications and requirements. With a proper choice of the parameter $c$ the Huber function has the features of both Quadratic error function and absolute error function. The Huber error function consistently works well. Hence, Quadratic error function can be completely replaced by Huber error function.

REFERENCES