International Journal of Hybrid Information Technology
Vol. 9, No.11 (2016), pp. 47-56
http://dx.doi.org/10.14257/ijhit.2016.9.11.05

Second Generation Neural Network for Two Dimensional Problems

Manmohan Shukla[1] and B. K. Tripathi[2],

Associate Professor CSE Dept. MPEC Kanpur, Professor CSE Dept. HBTI Kanpur,
mshukla.psit@gmail.com, abkt.iitk@gmail.com

Abstact

Neurocomputing in complex domain has yielded second generation neural networks. The neural network, which is based on complex value, contains different layers. The attributes of these layers are biases, weights, inputs and outputs. These attributes are also complex numbers. The signal processing, speech processing, learning and prediction of motion on plane are few areas in which complex domain neurocomputing is applied., since in the above said areas, the inputs and outputs are represented by complex values. It has been observed that the neural network with complex value can easily perform the transformation of geometric figures. The examples of transformations are rotation, parallel displacement of straight lines and circles. The neural network can extend to complex domain by the application of transformation. A number in complex domain is composed of different entities i.e. two real numbers and phase information. The two real numbers and phase information of any point on plane is naturally embedded in this number.

Keywords*: Magnitude and phase, Neural Network with real value, Neural Network with complex value, activation functions, complex back-propagation, split activation function, Liouville's theorem, sigmoidal function, Cauchy*-Riemann equation

1. Introduction

In present era, neural networks have proved as a robust technical tool for many tasks *e.g.* classification, function approximation, clustering and prediction [5, 8]. The learning algorithm of back-propagation is one of the best applied neural network model for its training. The learning algorithm provides a way to adjust the parameters of the model which are known as weights and threshold values. The main objective of the neural network is to study a mapping from input provided to output received. Since these weights and the thresholds are real values, therefore the neural network is said to be Real-Valued Neural- Network (RVNN).

RVNN has been worked to different fields such as motion interpretation on plane, speech processing, adaptive signal processing *etc* [3, 6]. These applications generally use signals. These signals carry two kind of knowledge *i.e.* magnitude and phase embedded in it. The signal information *i.e*. the magnitude and the phase are generally shown by two different entities and then the neural network can be trained using these two entities which are basically known as separate inputs to neural network. Although the training is provided to the neural network by these two inputs and it may output desired results, but no model can represent the relationship between phase and magnitude of a signal because of magnitude and phase caries separate information.

To preserve the association of the phase and the magnitude needs complex mathematical representation of the signal. Therefore the representation of the systems by using a model involves the signals with complex values instead of the real values. This proves that the neural- network with complex valued is more applicable for such type of applications.

The neural-network with complex value (CVNN) is the prolongation of neural-network with real-value (RVNN). In the neural-network with complex-value, all the attributes such as weights, biases, inputs and outputs are formed in complex nature [5, 7]. The neural-network with complex valued (CVNN) is useful in domains where complex numbers are generally used to show different parameters, furthermore inputs and outputs. The potential of action in the brain of human being may have distinct pulse patterns. The gap between pulses may also be distinct. This facilitates the initiation of the representation of phase and amplitude through neural networks with complex domain.

The neural-network with complex value (CVNN) may be used for the transformation of [4] the geometric figures.n The rotation, transformation of straight lines and circles are example of geometric transformation. It is not possible to understand these transformations using RVNN. The CVNN also confirms good generalization potential for above said transformations.

2. Neural Network with Complex Valued

The present section concentrates on how the ANN differs from the CVNN. Figure 1 (b), (c) show the ANN and CVNN respectively. It can be observed that the CVNN looks exactly similar the ANN as the neurons and the bias. It can also be observed that how it operates. The architectures are all similar in both the cases. There is only one difference that is the weights. The activation functions are complex valued in the CVNN whereas the activation functions were real valued in standard ANN [4, 7]. To examine the response of the CVNN, it is quit predictable that the concept of variables and functions in complex domains should be pertained.

2.1. The Behavior of Algorithm of Learning in Complex Domain

The complex back-propagation (Complex-BP) algorithm for CVNN has contemplated by many scientists in contemporary times [1, 2, 3]. The complex back-propagation (Complex-BP) is a complex domain interpretation of real back-propagation (Real-BP). The objective behind this is imprecision a function. That function will provide the mapping from the inputs to the outputs by employing a definite set of patterns of learning (x, y).

$$y = f(x, w) \tag{1}$$

Here $w \in C^p$,which tends to whole values of weights and all values of thresholds in neural networks. $x \in C^n$, which tends to whole complex valued training input-patterns and $y \in C^m$ corresponds to all complex valued training output-sequences.

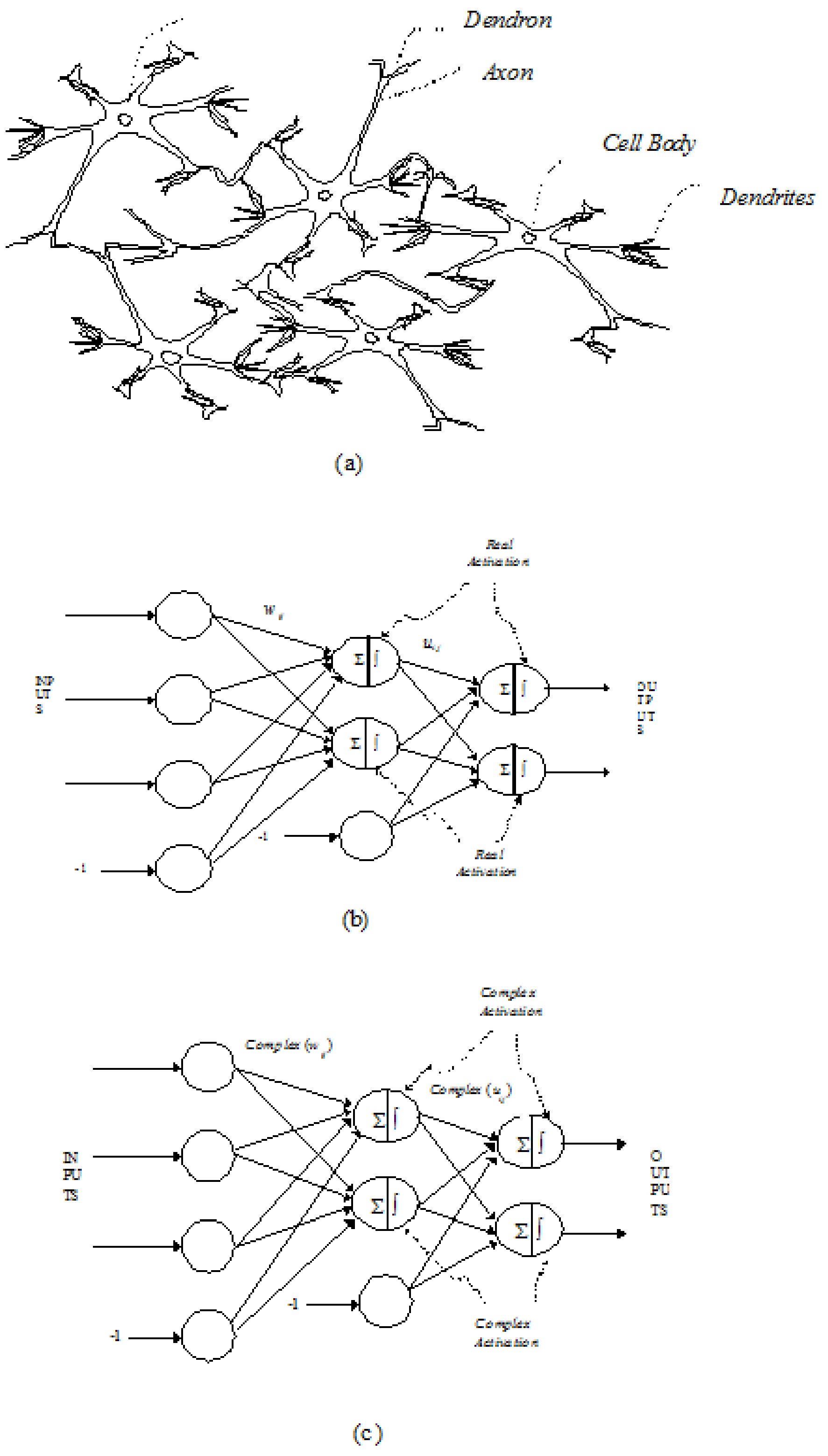

Figure 1. The Figure Shows How the CNN Evolved, (A) The Neurons in the Brain, (B) The ANN Was Developed By Studying the Neurons While, (C) the CNN Came Into Being as an Extension of the ANN

In the Complex-BP, there are two main classes [1, 3, 5]. The first one describes the straight forward extension of activation function. The activation function is mapped complex number to complex number, which is defined through a complex function. While in the second concept, the complex variable has two components-one is real and other is imaginary. The activation function works individually on the components to find out the complex output which consists of real and imaginary parts. The next section attains the Complex-BP by the second approach and this activation function is applied throughout the paper.

2.1.1. Split Activation Function

The Complex-BP with diverge activation function compromises the activation function for its boundary conditions. It can be proved by the Liouville's theorem which proposes that the complex function is not bounded on the entire plane which is complex except it is a constant function.

The error function can be written as,

$$g(y', y) : C^m \times C^m \rightarrow R^+$$

Here, R^+ shows the set of positive real numbers. The split activation function applied on any non linear function in complex domain, here z = x + i y and i is an imaginary unity (i = √(-1)), is

$$g(z) = g_R(x) + i g_R(y) \tag{2}$$

The output of the concealed layer neuron is,

$$z_j = f(\sum_{i=1}^{N} w_{ji} x_i) = f(net_j) \tag{3}$$

and the output of the visible layer neuron is,

$$y_k = f(\sum_{j=1}^{L} v_{kj} z_j) = f(net_k) \tag{4}$$

Here, L is the number of concealed neurons.

Here, w_{ji} and v_{kj} are the weights between input-concealed layer and concealed-output layers of neural network with complex value respectively. The result is calculated by using a split activation function as suggested in [4]. The non linear function is used distinctly to the two parts of complex domain. That is the real and imaginary parts of the input of the neuron in split activation function.

$$f(z) = f_R(x) + i f_R(y) \tag{5}$$

here, $$f_R(v) = 1/(1 - e^{(-v)}) \tag{6}$$

Above sigmoidal activation function is applied individually for real and imaginary part. This type of classification shelters that the range of magnitude of real and imaginary part of function f(z) is from 0 to 1. But on another hand, the function f(z) is heretofore holomorphic function, since the Cauchy-Riemann equality is not satisfied *i.e.*,

$$\frac{\partial F(z)}{\partial X} + i \frac{\partial F(z)}{\partial Y} = (1 - F_R(x)) \times F_R(x) + i(1 - F_R(y)) \times F_R(y) \neq 0 \tag{7}$$

Hence, the cogently of the holomorphy is agreeed for the boundedness of its activation function. In straight forward extension of activation function as suggested by Haykin and Leung [1], the nonlinear function maps the complex value to the complex value without splitting into the real and imaginary parts.

$$F(z) = \frac{1}{1 + e^{(-z)}} \tag{8}$$

$here, Z = X + iY$

The complex function f(z) is a holomorphic. But, according to the Liouville's theorem, a function in the complex plane C, which is a bounded holomorphic, is a invariable. To resolve the problems, the authors suggest that the data which are inputted should be taken in the complex numbers to some region. It may be possible that the input data are scaled, but there is no such limit on the values of the complex weights that can take as an input and therefore it is fractious to contrivance.

2.1.2. Learning Rule

In this paper, activation function of split is applied to implement Complex-BP algorithm.The difference between the expected output response and actual output response of the network is,

$$\delta_k = o_k - y_k \qquad k=1,2,\ldots\ldots\ldots,M \tag{9}$$

It is known as the error response.

Here O$_k$ represents the expected response at the kth node of the output and y$_k$ represents the actual output of the kth node. The sum of squared error of the network is given as,

$$E = \sum_{k=1}^{M} |o_k - y_k|^2 \tag{10}$$

The biases and weights are revamped accordingly by applying the descent on the function E. The partial derivative of E with respect to the real and imaginary part of the weights and biases are found distinctly, since this energy function is a non-analytic function.

The weight update for output layer is given as,

$$\mathrm{Re}[v_{kj}(n+1)] = \mathrm{Re}[v_{kj}(n)] - \eta\left(\frac{1}{2}\right)\frac{\partial E}{\partial \mathrm{Re}[v_{kj}(n)]} \tag{11}$$

$$\mathrm{Im}[v_{kj}(n+1)] = \mathrm{Im}[v_{kj}(n)] - \eta\left(\frac{1}{2}\right)\frac{\partial E}{\partial \mathrm{Im}[v_{kj}(n)]} \tag{12}$$

Combining equation (11) and (12),

$$v_{kj}(n+1) = v_{kj}(n) - \eta\frac{1}{2}\left(\frac{\partial E}{\partial \mathrm{Re}[v_{kj}(n)]} + i\frac{\partial E}{\partial \mathrm{Im}[v_{kj}(n)]}\right) \tag{13}$$

The chain rule is applied, to find partial derivative of error with respect to the real and imaginary part.

$$\frac{\partial E}{\partial \mathrm{Re}[v_{kj}]} = \frac{\partial E}{\partial \mathrm{Re}[y_k]}\cdot\frac{\partial \mathrm{Re}[y_k]}{\partial \mathrm{Re}[net_k]}\cdot\frac{\partial \mathrm{Re}[net_k]}{\partial \mathrm{Re}[v_{kj}]} \tag{14}$$

$$\frac{\partial E}{\partial \mathrm{Im}[v_{kj}]} = \frac{\partial E}{\partial \mathrm{Im}[y_k]}\cdot\frac{\partial \mathrm{Im}[y_k]}{\partial \mathrm{Im}[net_k]}\cdot\frac{\partial \mathrm{Im}[net_k]}{\partial \mathrm{Im}[v_{kj}]} \tag{15}$$

Finding the first term in the chain rule for real and imaginary part,

$$\frac{\partial E}{\partial \mathrm{Re}[y_k]} = -2\cdot\mathrm{Re}[\delta_k] \tag{16}$$

$$\frac{\partial E}{\partial \mathrm{Im}[y_k]} = -2\cdot\mathrm{Im}[\delta_k] \tag{17}$$

Finding error derivative w. r.t. the aggregated input of the neuron,

$$\frac{\partial E}{\partial net_k} = \frac{\partial E}{\partial \operatorname{Re}[net_k]} + i\frac{\partial E}{\partial \operatorname{Im}[net_k]} \tag{18}$$

Using equation (16) and (17),

$$\frac{\partial E}{\partial net_k} = \frac{\partial E}{\partial \operatorname{Re}[y_k]} \cdot \frac{\partial \operatorname{Re}[y_k]}{\partial \operatorname{Re}[net_k]} + i\frac{\partial E}{\partial \operatorname{Im}[y_k]} \cdot \frac{\partial \operatorname{Im}[y_k]}{\partial \operatorname{Im}[net_k]}$$

$$= -2\operatorname{Re}[\delta_k]\cdot \operatorname{Re}[y_k]\cdot(1-\operatorname{Re}[y_k]) - 2i\operatorname{Im}[\delta_k]\cdot\operatorname{Im}[y_k]\cdot(1-\operatorname{Im}[y_k]) \tag{19}$$

Representing the real and the imaginary part of equation (19) by Re[Δ] and Im[Δ],

$$\frac{\partial E}{\partial net_k} = -2(\operatorname{Re}[\Delta] + i\operatorname{Im}[\Delta]) \tag{20}$$

Finding partial derivative of error with respect to weights,

$$\frac{\partial E}{\partial v_{kj}} = \frac{\partial E}{\partial \operatorname{Re}[v_{kj}]} + i\frac{\partial E}{\partial \operatorname{Im}[v_{kj}]}$$

$$= \left(\frac{\partial E}{\partial \operatorname{Re}[net_k]} \cdot \frac{\partial \operatorname{Re}[net_k]}{\partial \operatorname{Re}[v_{kj}]} + \frac{\partial E}{\partial \operatorname{Im}[net_k]} \cdot \frac{\partial \operatorname{Im}[net_k]}{\partial \operatorname{Re}[v_{kj}]} \right)$$
$$+ i\left(\frac{\partial E}{\partial \operatorname{Re}[net_k]} \cdot \frac{\partial \operatorname{Re}[net_k]}{\partial \operatorname{Im}[v_{kj}]} + \frac{\partial E}{\partial \operatorname{Im}[net_k]} \cdot \frac{\partial \operatorname{Im}[net_k]}{\partial \operatorname{Im}[v_{kj}]} \right) \tag{21}$$

Using equation (20) and (21),

$$\frac{\partial E}{\partial v_{kj}} = -2(\operatorname{Re}[\Delta]\cdot\operatorname{Re}[z_j] + \operatorname{Im}[\Delta]\cdot\operatorname{Im}[z_j]) - 2i(\operatorname{Re}[\Delta]\cdot\operatorname{Im}[z_j] + \operatorname{Im}[\Delta]\cdot\operatorname{Re}[z_j])$$

$$= -2\Delta\cdot z_j^*$$

$$= -2z_j^*.(\operatorname{Re}[\delta_k]\cdot\operatorname{Re}[y_k]\cdot(1-\operatorname{Re}[y_k]) + i\operatorname{Im}[\delta_k]\cdot\operatorname{Im}[y_k]\cdot(1-\operatorname{Im}[y_k])) \tag{22}$$

Substituting equation (22) in equation (13),

$$v_{kj}(n+1) = v_{kj}(n) + \eta\cdot z_j^*.(\operatorname{Re}[\delta_k]\cdot\operatorname{Re}[y_k]\cdot(1-\operatorname{Re}[y_k]) + i\operatorname{Im}[\delta_k]\cdot\operatorname{Im}[y_k]\cdot(1-\operatorname{Im}[y_k])) \tag{23}$$

In short, the modification in weight is shown by,

$$v_{kj}(n+1) = v_{kj}(n) + \eta(d_k - y_k)f'(net_k)z_j^* \tag{24}$$

Here, $f'(net_k)$ is given by,

$$f'(net_k) = \operatorname{Re}[\delta_k]\cdot\operatorname{Re}[y_k]\cdot(1-\operatorname{Re}[y_k]) + i\operatorname{Im}[\delta_k]\cdot\operatorname{Im}[y_k]\cdot(1-\operatorname{Im}[y_k]) \tag{25}$$

Equation (2.1.24) gives the formulae for weight modification in the output layer, proceeding on the similar lines, the weight modification for input layer is described as,

$$w_{ji}(n+1) = w_{ji}(n) + \eta\cdot x_i^*.\{\operatorname{Re}[z_j]\cdot(1-\operatorname{Re}[z_j])$$

$$\times\sum_{k=1}^{N}(\operatorname{Re}[\delta_k]\cdot\operatorname{Re}[y_k]\cdot(1-\operatorname{Re}[y_k])\cdot\operatorname{Re}[v_{kj}]$$

$$+\operatorname{Im}[\delta_k]\cdot\operatorname{Im}[y_k]\cdot(1-\operatorname{Im}[y_k])\cdot\operatorname{Im}[v_{kj}])$$
$$-i\operatorname{Im}[z_j]\cdot(1-\operatorname{Im}[z_j])\times\sum_{k=1}^{N}(\operatorname{Re}[\delta_k]\cdot\operatorname{Re}[y_k]\cdot(1-\operatorname{Re}[y_k])\cdot\operatorname{Im}[v_{kj}]$$
$$-\operatorname{Im}[\delta_k]\cdot\operatorname{Im}[y_k]\cdot(1-\operatorname{Im}[y_k])\cdot\operatorname{Re}[v_{kj}])\} \qquad (26)$$

3. Geometrical Transformations

3.1. Rotation Transformation

Here, the aim is to show that the CVNN is capable of learning and generalizing a transformation where each point $I_k(=r_k e^{(i\theta_k)})$ is mapped into $I_k e^{(i\alpha)}(=r_k e^{(i\theta_k+\alpha)})$. The magnitude of the two points is same but the angle is updated by a factor $e^{(i\alpha)}$. In training step, a complex pattern of eight values is presented to the network. These input points represent a straight line in the complex plane. The output values are the input points rotated counterclockwise over Π/4 radians. Architecture of 1-6-1 is used and learning rate is set to 0.5. The training is carried out till the error converged to 0.1.

In Figure 2, triangles represents the input points and squares represents the output points. After training with these points, the neural network was presented with points representing alphabet Z and O. The Figure 3 shows the output of rotation when the network was presented with alphabet Z.

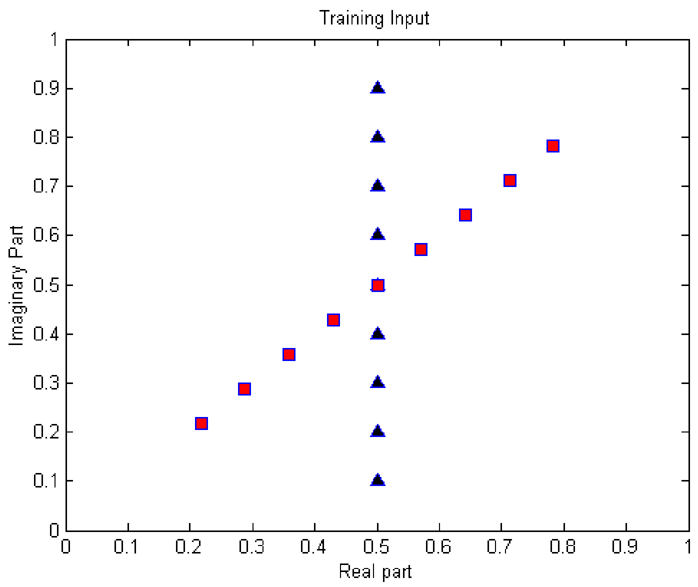

Figure 2. Training Points for Rotation Transformation. Triangle Represents the Input Points and Squares Represent the Output Points

Triangles represent the input point, squares represent the coveted output and the circles represent the real output of the network. The test results show that the network has learned the generalization of the rotation transformation. The same notation is followed in the Figure 4 which shows the test results of the network with input representing alphabet O. It should be noted that, while CVNN successfully learns and generalizes the rotation transformation, the RVNN do es not.

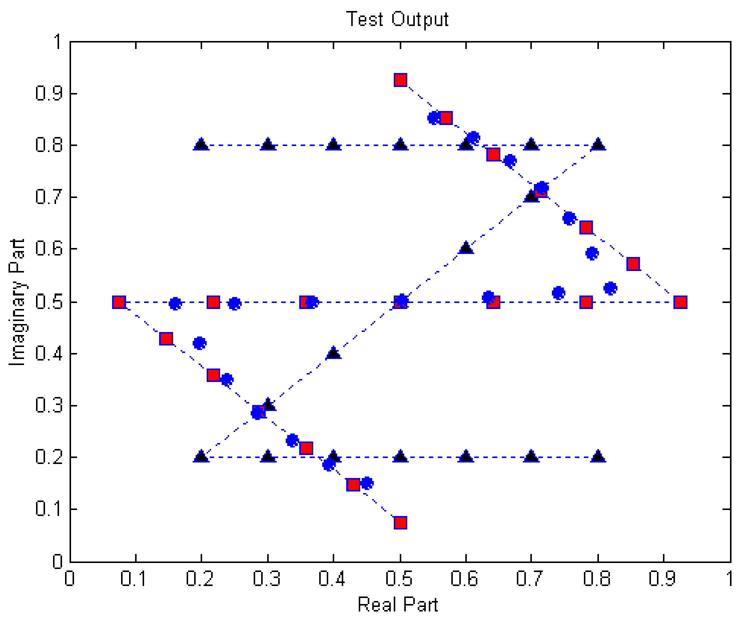

Figure 3. Rotation of Alphabet Z

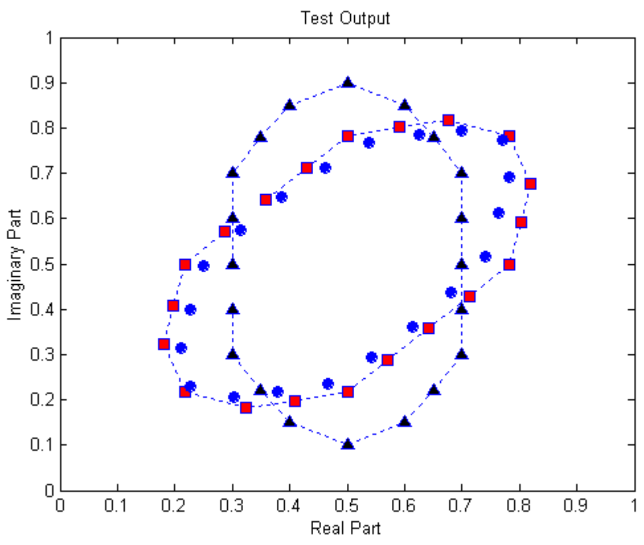

Figure 4. Rotation of Alphabet O

3.2. Similarity Transformation

In similarity transformation, the complex input pattern is scaled down by 0.5. The scaling is in terms of magnitude only, the angle is preserved. The training input pattern consists of a set of complex values which represents a line. This line lies in the complex plane. The pattern which is outputted is the scaled pattern.

The training error convergence is 0.1. The generalization ability of the CVNN is checked by giving alphabet Z as an input. As seen from Figure 6, the network is able to generalize the similarity transformation.

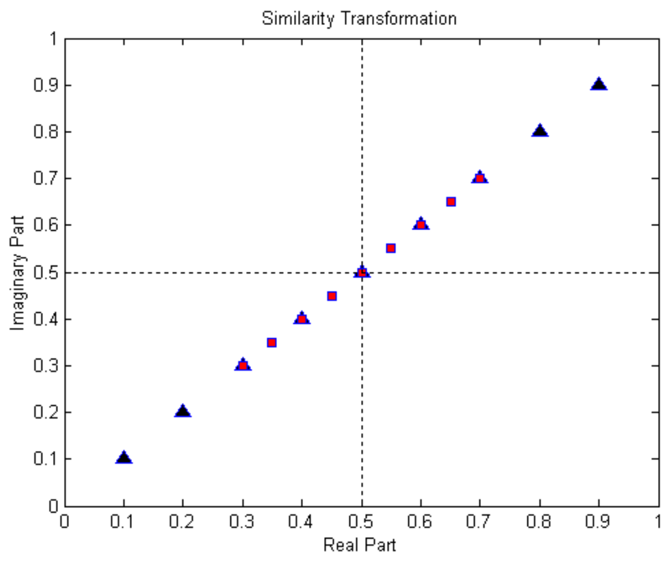

Figure 5. Similarity Transformation: Training Input (Triangles) and Output (Squares)

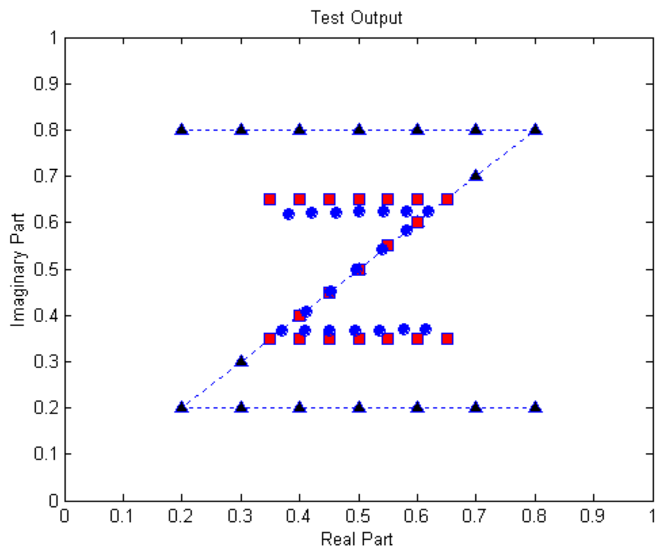

Figure 6. Test Results for Similarity Transformation: Test Input (Triangles), Desired Outputs (Squares), Actual Outputs (Circles)

4. Conclusions

The domain of neural-network with complex value has given the second generation of neurocomputing. The complex-valued back-propagation is defined and its performance is tested for motion on plane. The generalized mapping performed in our experiments, which confirm the ability of complex-BP with output function defined in Eq.(2), for the processing of amplitude and phase of signals accurately. The complex-valued neural network may understand and postulate the both amplitude and phase of each point on plane while traditional real valued neural network can not. This research work has future scope to extend this work from two dimensions (complex number) to four dimensions (quaternion). The quaternion can be used for easy analysis of three and four dimension image processing problems, computer vision and robotics.

References

[1] H. Leung and S. Haykin, "The Complex Backpropagation Algorithm", IEEE Trans. On Signal Processing, vol. 39, no. 9, **(1991)**.

[2] N. Benvenuto and F. Piazza, "On the Complex Backpropagation Algorithm", IEEE Trans. On Signal Processing, vol. 40, no. 4, **(1992)**.

[3] G. M. Georgiou and C. Koutsougeras, "Complex Domain Backpropagation", IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing, vol 39, no. 5, **(1992)**.

[4] T. Nitta, "An Extension of the Back-Propagation Algorithm to Complex Numbers", Neural Networks, vol. 10, no. 8, **(1997)**.

[5] B K Tripathi, "On Efficient Learning Machine with Root Power Mean Neuron in Complex Domain", ISSN: 1045-9227, IEEE Transaction on Neural Network, vol. 22, no. 05, **(2011)**, pp. 727-738.

[6] B K Tripathi, "Complex Generalized-Mean Neuron Model and its Applications", DOI: 10.1016/j.asoc.2009.12.038, Applied Soft Computing, Elsevier Science, vol. 11, no. 01, **(2011)**, pp. 768 -777.

[7] B K Tripathi, "The Novel Aggregation Function Based Neuron Models in Complex Domain", ISSN 1432-7643, Soft Computing, Springer, vol. 14, no. 10, **(2010)**, pp. 1069 - 1081.

[8] B K Tripathi, "Hybrid computation model for intelligent system design by synergism of modified EFC with neural network", DOI: 10.1142/S0219622014500813, International Journal of Information Technology & Decision Making, World Scientific, vol. 13, **(2014)**.