CLASSIFICATION OF SPATIO-TEMPORAL DATA

Jakub Zahradník, Miroslav Skrbek

Czech Technical University in Prague, Faculty of Information Technology,
Department of Computer Systems,
Computational Intelligence Group,
Kolejní 550/2, 160 00 Praha 6, Czech Republic

zahraja6@fel.cvut.cz (Jakub Zahradník)

Abstract

This paper presents a new approach in spatio-temporal data classification. This classification can be used in many branches including robotics, computer vision or medical data analysis. Due to easy transformation of time dimension of spatio-temporal data into the phase of complex number, the presented approach uses complex numbers. The classification is based on a complex-valued neural network with multilayer topology. The paper proposes an extension of complex-valued backpropagation algorithm, which uses activation function applying non-linearity on the amplitude only (preserving the phase) instead of commonly used activation function applying non-linearities on the real and the imaginary part separately. In order to transform the input data into complex numbers, a new coding technique is presented. It encodes the time-dimension into phase of complex number and space-dimensions into amplitude of complex numbers. Another task is to develop output coding, that would allow the classification from complex numbers. It is solved with introduction of one-of-N coding extension into complex numbers, which is used as network's output coding. This approach is verified in application of hand-written character recognition, using the data collected during the writing process. The simulation results of this application are presented in the paper.

Keywords: complex-valued, neural network, spatio-temporal, backpropagation.

Presenting Author's biography

Jakub Zahradník is a Ph.D. student at Czech Technical University in Prague, Faculty of Information Technology, Department of Computer Systems. After getting his master degree in 2009, he has joined the Computational Intelligence Group. His research focuses on neural networks, especially the complex-valued ones.

1 Introduction

The classification of spatio-temporal data is a frequent task in many branches like robotics, computer vision, computational biology, mobile computing, traffic analysis or medical data analysis. There are a number of approaches in neural networks processing spatio-temporal data, e.g. recurrent neural networks or time delay neural networks (TDNN).

In this paper, we present a new method of spatio-temporal data classification based on complex-valued neural network (briefly described in Chapter 2) trained by a complex-valued backpropagation algorithm (Complex-BP). Standard version of Complex-BP [1] uses activation function with non-linearities applied on the real and the imaginary part separately. Our extension of Complex-BP introduces activation function with non-linearity applied on the amplitude, preserving the phase.

Chapters 3 and 4 describe proposed techniques used to encode input and output data. They are based on a new spatio-temporal data coding, which encodes the spatial component of data into the amplitude and the temporal component into the phase of complex number. Classification process itself is described in Chapter 5.

Chapter 6 presents our simulation results obtained from hand-written character recognition based on tablet data.

Chapter 7 concludes presented approaches and results.

2 Complex-valued neural network

2.1 Complex-valued neuron

According to [2], the basic unit in complex-valued neural networks is neuron, which extends the real-valued one to complex domain. In this paper we use the model of complex-valued neuron shown in Fig. 1. All inputs, output, weights and threshold are represented by complex numbers.

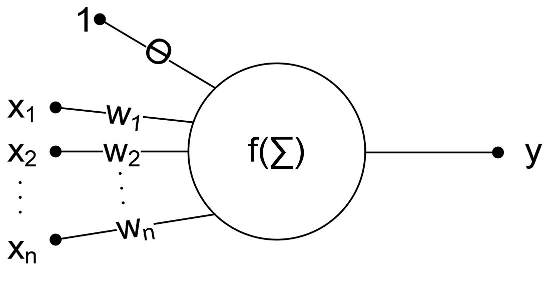

Fig. 1 A model of complex-value neuron. To each input x_i belongs the weight w_i.

The gain of neuron is defined as:

$$g = \sum_{i=1}^{N} w_i \cdot x_i + \Theta, \qquad (1)$$

where $g \in \mathbb{C}$ is the gain, $w_i \in \mathbb{C}$ is weight of i-th input $x_i \in \mathbb{C}$, $\Theta \in \mathbb{C}$ is threshold a N is inputs count. The output is defined as follows:

$$y = f(g), \qquad (2)$$

where y is the output of neuron and $f: \mathbb{C} \to \mathbb{C}$ is the neuron's activation function (discussed in Chapter 2.2). We use the notation $[\cdot]^R$ and $[\cdot]^I$ to separate the real and the imaginary part of complex number. For example the neuron's output can be noted as $y = y^R + i \cdot y^I$, where $y \in \mathbb{C}$, $y^R, y^I \in \mathbb{R}$ and i denotes $\sqrt{-1}$. We also use the polar form of notation: $|[\cdot]|e^{i \cdot \varphi_{[\cdot]}}$. The output of neuron in this notation can be written as $y = |y|e^{i \cdot \varphi_y}$.

2.2 Complex-valued activation function

Assuming the complex-valued backpropagation learning algorithm (as shown in Chapter 2.3), we cannot extend real-valued activation functions to the complex domain from the following reason. We should recall the Liouville's theorem, which says that 'if $f(z)$ is analytic (differentiable) at all $z \in \mathbb{C}$ and bounded, then $f(z)$ is a constant function'. Because activation function $f(z)$ should be bounded, $f(z)$ is constant in the result of Liouville's theorem. That means the analytic functions are not suitable as activation functions in complex-valued neural networks (as discussed in [3]).

According to [2] there are two main classes of activation functions in complex domain, which are used in the most of complex-valued neural networks. The first class divides the complex number into the real and the imaginary part, applies real-valued functions on each part independently and combines the results back into one complex number. We call this class ***Re-Im activation function***. It can be noted as:

$$f_{Re-Im}(g) = f_{Re}(g^R) + i \cdot f_{Im}(g^I), \qquad (3)$$

where $f_{Re-Im}: \mathbb{C} \to \mathbb{C}$ and $f_{Re}, f_{Im}: \mathbb{R} \to \mathbb{R}$. In this class f_{Re} and f_{Im} are non-linear functions, e.g. sigmoid $f(x) = \frac{1}{1+e^{-x}}$ or hyperbolic tangent $f(x) = \tanh(x)$.

The second class is very similar, it divides the given complex number into the amplitude and the phase (uses the polar notation), applies real-valued functions on each part independently and combines the results back into one complex number. We call this class ***Am-Ph activation function***. It can be noted as:

$$f_{Am-Ph}(g) = f_{Am}(|g|) \cdot e^{i \cdot f_{Ph}(\varphi_g)}, \qquad (4)$$

where $f_{Am-Ph}: \mathbb{C} \to \mathbb{C}$ and $f_{Am}, f_{Ph}: \mathbb{R} \to \mathbb{R}$. As discussed in [4], we use $f_{Ph}(\varphi) = \varphi$ and sigmoid or hyperbolic tangent for $f_{Am}(x)$. Fig. 3 shows $f_{Am-Ph}(z) = f(|z|) \cdot e^{i \cdot \varphi_z}$ where $f(x) = \frac{1}{1+e^{-x}}$.

2.3 Complex-valued backpropagation

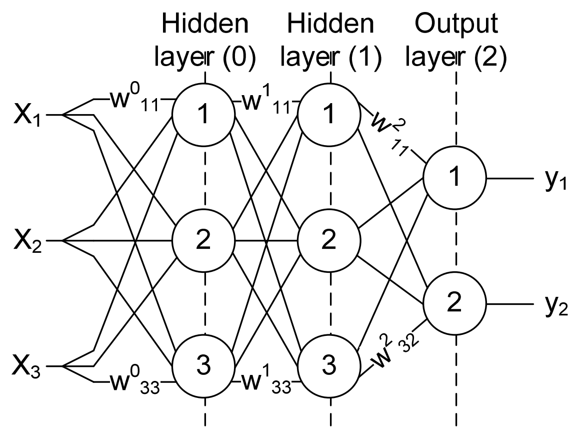

Fig. 2 A multilayer neural network consisting of two hidden layers and one output layer. The weight w_{ij}^k denotes the weight of j-th neuron in k-th layer for i-th input. In other words it is the weight between i-th neuron in (k-1)-th layer and j-th neuron in k-th layer. The threshold Θ_j^k denotes the threshold of j-th neuron in k-th layer.

A complex-valued backpropagation [1] is an extension of the Backpropagation algorithm to complex numbers. It is supervised method, which changes the weights to minimize the error function. It defines the changes of weights and thresholds as follows:

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i^R} - i \cdot \eta \frac{\partial E}{\partial w_i^I}, \tag{5}$$

$$\Delta \Theta_i = -\eta \frac{\partial E}{\partial \Theta_i^R} - i \cdot \eta \frac{\partial E}{\partial \Theta_i^I}, \tag{6}$$

where w_i is i-th weight of neural network, Θ_i is i-th threshold of neural network, η is learning constant (learning rate) and E is the error function defined as follows:

$$E = \frac{1}{2} \sum_{i=0}^{N} |y_i - \widehat{y_i}|^2, \tag{7}$$

where y_i is i-th output of the neural network, $\widehat{y_i}$ is desired value of i-th output of neural network and N is the count of neural network's outputs. The algorithm is used in multilayer neural networks, where neurons are arranged into layers. Neurons in each layer communicate only with neighbouring layers. Fig. 2 displays the structure of multilayer neural network.

The general complex-valued backpropagation algorithm is described in [1] including convergence discussion using *Re-Im activation function*. Nevertheless the paper does not deal with backpropagation algorithm on *Am-Ph activation function*, which we use.

In order to train a neural network with *Am-Ph activation function*, we introduce a local gradient δ_i^o of i-th neuron in the output layer as follows:

$$\delta_i^o = \begin{array}{c} (y_i - \widehat{y_i})^R \cdot [\alpha_i^o]^R + (y_i - \widehat{y_i})^I \cdot \beta_i^o + \\ + i \cdot ((y_i - \widehat{y_i})^I \cdot [\alpha_i^o]^I + (y_i - \widehat{y_i})^R \cdot \beta_i^o), \end{array} \tag{8}$$

where y_i is output of i-th neuron in the output layer, $\widehat{y_i}$ is the desired output of i-th neuron in the output layer and α_i^o, β_i^o are defined as follows (o denotes output layer):

$$\alpha_i^n = \begin{cases} \left([g_i^n]^{R^2} + i \cdot [g_i^n]^{I^2}\right) \frac{1}{\left|g_i^n\right|^2} \cdot \\ \cdot \left(aF_i^{n\prime}(|g_i^n|) - \frac{aF_i^n(|g_i^n|)}{|g_i^n|}\right) + \\ + (1+i) \frac{aF_i^n(|g_i^n|)}{|g_i^n|}, \end{cases} \tag{9}$$

$$\beta_i^n = \frac{[g_i^n]^R \cdot [g_i^n]^I}{\left|g_i^n\right|^2} \left(aF_i^{n\prime}(|g_i^n|) - \frac{aF_i^n(|g_i^n|)}{|g_i^n|}\right), \tag{10}$$

where g_i^n is gain of i-th neuron in n-th layer and $aF_i^n(x)$ is the function $f_{Am}(x)$ in activation function of i-th neuron in n-th layer (see Eq. (4)).

After computing local gradient of the output layer, we are able to compute local gradients of the remaining layers, where local gradient of the layer is computed from local gradient of the previously computed layer (local gradient of the last hidden layer is computed from local gradient of the output layer and so on). Local gradient of i-th neuron in (n-1)-th hidden layer:

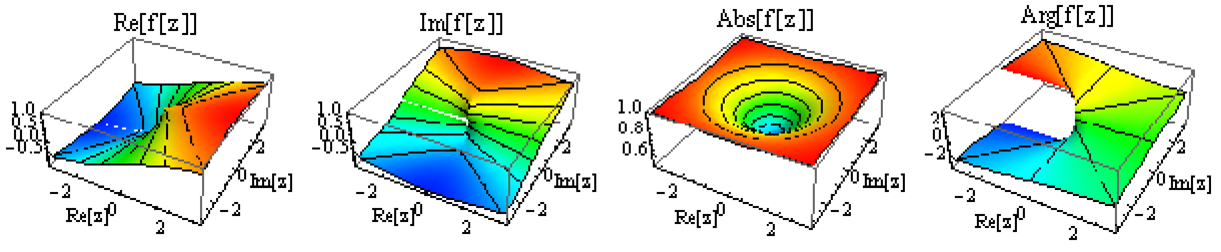

Fig. 3 Am-Ph activation function ($f_{Am-Ph}(z) = f(|z|) \cdot e^{i \cdot \varphi_z}, f(x) = \frac{1}{1+e^{-x}}$). From left to right: real part, imaginary part, absolute value and phase.

$$\delta_i^{n-1} = \begin{array}{c}[\gamma_i^n]^R \cdot [\alpha_i^{n-1}]^R + [\gamma_i^n]^I \cdot \beta_i^{n-1} + \\ + i \cdot ([\gamma_i^n]^I \cdot [\alpha_i^{n-1}]^I + [\gamma_i^n]^R \cdot \beta_i^{n-1}),\end{array} \tag{11}$$

where γ_i^n (back-propagation of the error) is defined as follows:

$$\gamma_i^n = \sum_{k=1}^{N_n} [w_{ik}^n]^* \cdot \delta_k^n, \tag{12}$$

where N_n is count of neurons in n-th layer and $[\cdot]^*$ denotes conjugate of $[\cdot]$. The changes of thresholds and weights are defined as:

$$\Delta\Theta_i^n = -\eta \cdot \delta_i^n, \tag{13}$$

$$\Delta w_{ij}^n = -\eta \cdot \delta_j^n \cdot [y_i^{n-1}]^*, \tag{14}$$

where $\eta \in \mathbb{R}$ is learning constant (learning rate).

To suppress the local minima problem, the momentum is added into the delta rule. It extends Eq. (13) by $+ \alpha \cdot \Delta\Theta_i^n(t-1)$ and Eq. (14) by $+ \alpha \cdot \Delta w_{ij}^n(t-1)$.

The algorithm can run with two weight update modes: per iteration update or per epoch update (the batch mode).

3 Input data coding

The coding of spatio-temporal data is very important for further processing. Our coding approach (called ***Spatio-temporal data coding***) is based on polar form of complex numbers. The data is sequence of events. Each event is identified by its position in space and time of occurance. We call this sequence *a data set*. Each vector $(\vec{x_i}, t_i)$ representing one event, where vector $\vec{x_i} \in \mathbb{R}^n$ denotes the position of event and $t_i \in \mathbb{R}$ denotes time of occurance of event, is encoded into vector $\vec{d_i} \in \mathbb{C}^n$ as follows:

$$d_{ij} = \alpha(x_{ij}) \cdot e^{i \cdot \beta(t_i)}, \tag{15}$$

$$\alpha(x_{ij}) = \frac{x_{ij} - \min_k x_{kj}}{\max_k x_{kj} - \min_k x_{kj}} \in \langle 0; 1 \rangle, \tag{16}$$

$$\beta(t_i) = \frac{t_i - \min_k t_k}{\max_k t_k - \min_k t_k} \cdot 2\pi \in \langle 0; 2\pi \rangle. \tag{17}$$

The whole data set (all the events) is distributed into time interval $\langle 0; 2\pi \rangle$ and each dimension of position is normalized to interval $\langle 0; 1 \rangle$. Fig. 4 shows the proposed coding approach in one space dimension.

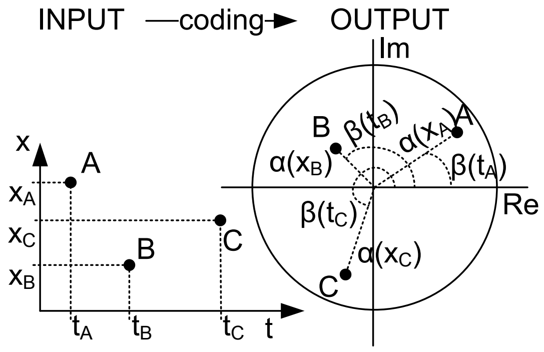

Fig. 4 An example of spatio-temporal data coding. The spatio-temporal input data with one space dimension x and time t on the left (consisting of three points $A[x_A, t_A]$, $B[x_B, t_B]$ and $C[x_C, t_C]$) are encoded into three complex numbers on the right ($d_A = |\alpha(x_A)| \cdot e^{i \cdot \beta(t_A)}$, $d_B = |\alpha(x_B)| \cdot e^{i \cdot \beta(t_B)}$ and $d_C = |\alpha(x_C)| \cdot e^{i \cdot \beta(t_C)}$).

The data set is in chronological order, so the sequence of encoded data phases is an increasing sequence.

In spatio-temporal domain it is useless to be limited only to one event; the essential information is in the sequence of events, called data set.

4 Output data coding

The output of neural network dealing with classification of real-valued problems is based on one-of-N coding. Neural network has as many outputs as the count of classified classes. Each output represents, weather the input belongs to the specific class or not. Unity means, that input belongs to the class, and zero means the exact opposite. This approach is useless in classification of spatio-temporal data, because we want to evaluate the whole data set. It is necessary to bring the time part into it.

We propose a new coding (called ***Spatio-temporal one-of-N coding***) inspired by one-of-N coding, whose output is coded as follows:

$$y_i = |y_i| \cdot e^{i \cdot \varphi_{y_i}}, \tag{18}$$

where $|y_i| \in \langle 0; 1 \rangle$ represents belonging of the submitted input to the i-th class (the higher the value the higher the probability of belonging to i-th class) and $\varphi_{y_i} \in \langle 0; 2\pi \rangle$ should represent the same time as submitted input. Because the essential information is included in data set, further processing is needed to classify the input data set (see Chapter 5).

Fig. 5 shows examples of outputs encoded by this approach.

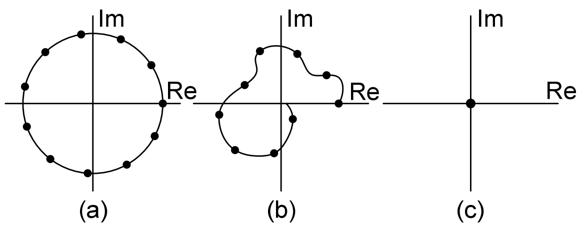

Fig. 5 (a) Ideal case: all the data from time interval $\varphi_{y_i} \in \langle 0; 2\pi \rangle$ belong to class ($|y_i| = 1$); (b) typical case: some data belong to the class more and some less ($|y_i| \in \langle 0; 1 \rangle$); (c) ideal case: none of data belong to class ($|y_i| = 0$).

5 Classification of spatio-temporal data

The classification is done by the complex-valued neural network (using the *Am-Ph activation function*; see Eq. (4)), which is trained using the complex-valued backpropagation (see Chapter 2.3). The input of neural network is encoded in former presented coding approach (see Chapter 3). The output is encoded in spatio-temporal one-of-N coding (see Chapter 4).

To classify spatio-temporal data $(\vec{x_i}, t_i)$ coded as $\vec{d_i}, i \in 1..N$, the following steps are done:

(1) $c_j = 0, \forall j \in 1..C,$

(2) $c_j = c_j + \delta(y_j^i, e^{i \cdot \varphi_{d_{io}}}), \forall i \in 1..N, \forall j \in 1..C,$

(3) data belong to the class with minimal c_j,

where C is number of classes to classify, N is the amount of data, y_j^i is j-th output of neural network when submitted the i-th data and $\delta(y_j^i, e^{i \cdot \varphi_{d_{io}}})$ is the function describing the similarity between the output and the classes ideal output for specified time (phase).

We use the following function to compare the output to the classes ideal output:

$$\delta_{Abs}(y, c) = |y - c|. \qquad (19)$$

It is the distance of the values y and c in complex plane. In case of using this function, the winner class is the one with the minimal value of following expression:

$$c_j = \sum_{i=1}^{N} |y_j^i - e^{i \cdot \varphi_{d_{io}}}|. \qquad (20)$$

6 Simulation results

To verify the proposed approach, the simulator of complex-valued neural network was implemented. The simulator is implemented in Java and it supports the multilayer feedforward networks. The simulator uses the model of neuron stated in Chapter 2.1 with both the *Re-Im* and *Am-Ph activation functions*. The presented extension of complex-valued backpropagation algorithm is implemented.

The chosen problem is to recognize single hand-written character based on shape and pencil's velocity during writing. The characters are chosen, because they are easily repeatable. To classify the data set, we use complex-valued neural network with two hidden layers. Let C be the count of classes to classify, then the output layer consists of C neurons, the second hidden layer (the neighbour of the output layer) consists of $2C$ neurons and the first hidden layer consists of $3C$ neurons. In every neuron we use the *Am-Ph activation function* shown in Eq. (4) with $f_{Am}(x)$ defined as follows:

$$f_{Am}(x) = \frac{1}{1 + e^{10(0.5 - x)}}. \qquad (21)$$

The constants 10 and 0.5 are needed because we obtain only $x \in \mathbb{R}^+$ so we want to transpose the curve and make it steeper (presuming to obtain only small numbers).

Testing data consists of 5 classes (characters 'a', 'b', 'd', 'k' and 'l'), in each class 100 data sets obtained by tablet, written by only one person. The characters were chosen with respect to the similarity between 'a' and 'd' and between 'b' and 'l'. From each class 5 random data sets are used to train the network and the other 95 to evaluate the classification. Training data sets are time-quantizied to achieve 64 equidistant input vectors per data set (one character). We do the quantization to prevent the network preferring more frequent phases.

The initial thresholds and weights are randomly generated; real and imaginary part in interval $\langle -0.3; 0.3 \rangle$. The network is trained in batch mode with learning rate of 0.00015 and momentum 0.01.

Tab. 1 displays the accuracy of classification on training and testing sets. In rows are submitted characters and in columns classified characters. Notation a (b) means that a input data from training set (5 instances of each class) were classified as the column header and b input data from testing set (95 instances of each class) were classified as the column header.

Tab. 1 A character classification on training and testing sets.

	a	**b**	**d**	**k**	**l**
a	5 (95)	0 (0)	0 (0)	0 (0)	0 (0)
b	0 (0)	5 (93)	0 (0)	0 (0)	0 (2)
d	0 (0)	0 (0)	5 (95)	0 (0)	0 (0)
k	0 (0)	0 (0)	0 (0)	5 (95)	0 (0)
l	0 (0)	0 (0)	0 (0)	0 (0)	5 (95)

The training set was classified with 100% accuracy, but in the testing set were two characters ‘b’ wrongly classified as character ‘l’.

Fig. 6 displays the error while training the network, Fig. 7 shows one data set of character ‘k’ and its reduction to 64 time-equidistant points, Fig. 8 displays a character ‘k’ encoded in spatio-temporal data coding (see Chapter 3) and Fig. 9 shows an example of neurons outputs.

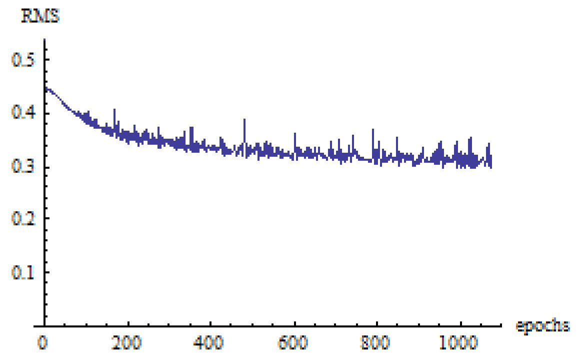

Fig. 6 An RMS error during the training process. The experiment was repeated many times with similar results. The fluctuation is caused mainly by the momentum of training.

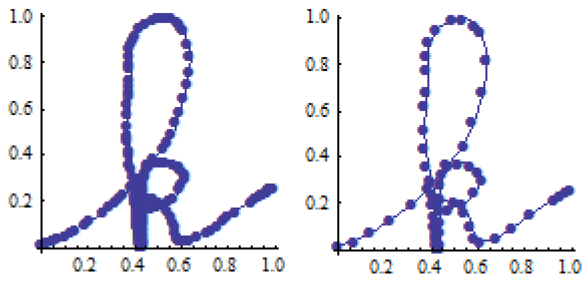

Fig. 7 One data set representing character ‘k’. Original data set on the left is reduced to 64 time-equidistant points on the right. This reduction is required for training process, otherwise the network can prefer the part of data set with higher time density of points.

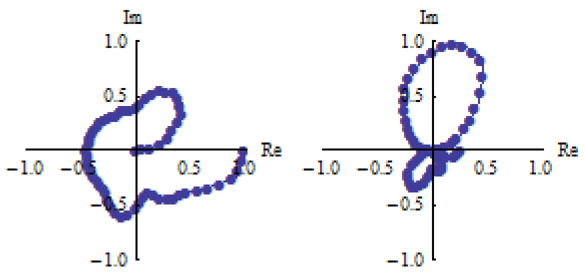

Fig. 8 Data set representing character ‘k’ encoded with spatio-temporal data coding. Encoded x-dimension on the left and y-dimension on the right.

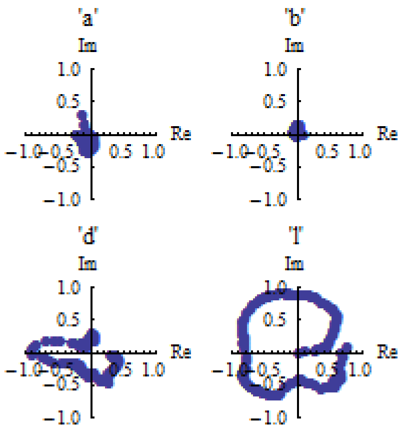

Fig. 9 A neurons outputs for classes representing characters ‘a’, ‘b’, ‘d’ and ‘l’ encoded in spatio-temporal one-of-N coding (see Chapter 4). Tested is not-trained character ‘l’.

7 Conclusion

We have presented the process of classification of spatio-temopral data. The process is based on complex-valued neural network learnt by algorithm of complex-valued backpropagation. In order to train a network using *Am-Ph activation function*, we have extended the complex-valued backpropagation algorithm. For classification purposes we have introduced two new coding techniques: spatio-temporal data coding and spatio-temporal one-of-N coding. We have implemented an application in Java and simulated the identification of hand-written character based on presented approach.

8 Acknowledgement

This work has been supported by the research program "Transdisciplinary Research in the Area of Biomedical Engineering II" (MSM6840770012) sponsored by the Ministry of Education, Youth and Sports of the Czech Republic.

9 References

[1] T. Nitta, An extension of the Back-Propagation Algorithm to Complex Numbers, *Neural Networks*, Vol. 10, No.8, pp. 1391 – 1415, 1997.

[2] A. Hirose, Complex-Valued Neural Networks. Springer Berlin. 2006.

[3] G. M. Georgiou and C. Koutsougeras, Complex Domain Backpropagation, *IEEE Transactions on Circuits and Systems-II*, Vol. 39, No. 5, pp. 330–334, 1992.

[4] Y. Kuroe, M. Yoshida and T. Mori, On Activation Functions for Complex-Valued Neural Networks – Existence of Energy Functions – , *Artificial Neural Networks and Neural Information Processing*, LNCS 2714, pp. 985–992, 2003.