MEDWELL PUBLISHING
International Journal of Soft Computing 4 (1): 16-24, 2009
ISSN: 1816-9503

A Novel Neural Learning Algorithm for Separation of Blind Signals

[1]D. Malathi and [2]N. Gunasekaran
[1]School of Computer Science and Engineering, S.R.M Engineering College,
S.R.M University, Tamil Nadu, India
[2]School of Electronics and Communication Engineering, Anna University, Chennai-600 025, India

Abstract: This study proposes, a new learning algorithm for extracting the independent source signals from an artificially mixed signal. An adaptive self-normalized radial basis function neural network is developed and trained by the proposed learning algorithm to model the nonlinearity from the latent variables to the observations. The joint probability density function and marginal probability density functions are used to determine the inverse of the nonlinear mixing matrix, which is assumed to exist and able to be approximated. The centers of the ASN-RBF network are initialized with the weights between input and hidden layer to update the parameters in the generative model. This proposed algorithm is well-suited for nonlinear data analysis problems and theoretically interesting. Minimum 3 signals are considered for simulation. Simulation results show the feasibility of the proposed algorithm. The performance of the proposed network is compared with the Independent Component Analysis (ICA) algorithm and it is illustrated with computer simulated experiments.

Key words: Stochastic gradient descent optimization algorithm, radial basis function neural network, independent component analysis, backpropagation neural network

INTRODUCTION

In many signal and data analysis situations, observed data are known to be some mixture of underlying sources. The mixing process may be linear or nonlinear and while, the structure of the mixing process may be known, the mixture parameters (in the linear case, the mixing matrix) will be unknown. Blind Source Separation (BSS) is a technique, which allows separating a number of source signals from observed mixtures of those sources without a previous knowledge of the mixing process (Cichocki and Amari, 2002). For example, if there are many speakers in a room, then each microphone receives a different mixture of the speaker signals. The task is then to separate the original (unmixed) speaker signals from the mixtures received at the microphones. A numerous attention has been aroused in these techniques in recent years with an increasing number of existing approaches. So, far several authors studied the difficult problem of the nonlinear blind source separation and proposed a few efficient demixing algorithms (Burel, 1992; Deco and Brauer, 1995; Pajunen, 1998; Hyvarinen and Pajunen, 1999). Model-free methods, which used Kohonen's Self-Organizing Map (SOM) have been proposed to extract independent sources from nonlinear mixture, but suffers from the exponential growth of network complexity and interpolation error in recovering continuous sources (Herrmann and Yang, 1996; Pajunen *et al*., 1996; Lin and Grier, 1997). A nonlinear blind source separation algorithm has been proposed using two-layer perceptrons by the gradient descent method to minimize the mutual information (Burel, 1992). Subsequently, backpropagation algorithm has been developed for Burel's model by natural gradient method (Yang *et al*., 1997). In their model cross nonlinearities are included. An entropy-based direct algorithm has been proposed for blind source separation in post nonlinear mixtures (Taleb *et al*., 1995). In addition, the extension of related linear ICA theories to the context of nonlinear mixtures has resulted in the development of nonlinear ICA (Pajunen *et al*., 1996). The so-called nonlinear ICA is to employ a nonlinear function to transform the nonlinear mixture such that the outputs become statistically independent after the transformation. However, this transform is not unique without some specific constraints on the function of nonlinear mixing. If x and y are 2 independent random variables, then f(x) and g(y) are also statistically independent regardless of the nonlinear

Corresponding Author: D. Malathi, School of Computer Science and Engineering, S.R.M Engineering College, S.R.M University, Tamil Nadu, India

functions f and g. Although, there exists many difficulties for this problem, several nonlinear ICA algorithms have been proposed and developed (Pajunen *et al*., 1996; Lee *et al*., 1997). The existence and uniqueness of nonlinear ICA are discussed in detail (Herrmann and Yang, 1996; Pajunen *et al*., 1996; Pajunen, 1996, 1999; Lin *et al*., 1997; Pajunen, 1998) and pointed out that the solution of nonlinear ICA always exists (Hyvarinen and Pajunen, 1999). It can become unique up to a rotation provided that the mixing function is constrained to a conformal mapping for a 2-dimensional problem together with some other assumptions such as bounded support of the probability density function (pdf) (Herault and Jutten, 1986, 1991; Li and Sejnowski, 1995; Makeig and Bell, 1996; Papadias and Paulraj, 1997; Van der Veen *et al*., 1997; Linde *et al*., 1980). Several authors have introduced a class of adaptive algorithms for source separation (Cardoso and Laheld, 1996). A contrast function, which consists of the mutual information and partial moments of the outputs of the separation system is defined to separate nonlinear mixture (Ying *et al*., 2001).

In this study, we propose a novel learning algorithm to extract independent components from the mixture signal. The ASN-RBF neural network is developed by MATLAB and trained by the proposed algorithm to minimize the objective function, which is the difference between the joint probability density function (pdf) and the product of marginal pdfs of the output vectors. The training continues until a stable value of weight vector is obtained.

FORMULATION OF THE PROBLEM

Blind Source Separation (BSS) is a technique, which allows separating a number of source signals from observed mixtures of those sources without a previous knowledge of the mixing process (Comon *et al*., 1991; Lin *et al*., 1997). Suppose that the two audio signals are generated by two sources simultaneously. Assume that the two signal generators are placed at 2 different locations and the 2 signal observers (e.g., microphones) receive the nonlinear mixture of these 2 signals. Each of these observed signals is a weighted sum of the 2 source signals generated by 2 sources. We denote them as m_1 and m_2, respectively. Now, the 2 observed signals can be represented as:

$$o_1 = w_{11}m_1 + w_{12}m_2 \quad (1)$$

$$o_2 = w_{21}m_1 + w_{22}m_2 \quad (2)$$

where, w_{11}, w_{12}, w_{21}, w_{22} are the parameters whose value depends on the locations of the 2 sources. These values can be represented as a linear mixing matrix, say W as given in Eq. 3.

The observed signal can be represented in the form of a vector as given in Eq. 4:

$$W = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} \quad (3)$$

$$O = W.M \quad (4)$$

where, $M = [m_1\ m_2]^T$

Now, the blind source separation problem can be defined as an estimation of the two original source signals m_1 and m_2 by receiving only the observed signals O_1 and O_2. This problem is also called as cocktail party problem as already been mentioned by many researches.

Actually, if we knew the mixing matrix W, then we can easily separate the two source signals by finding the inverse of the mixing matrix, i.e. W^{-1}, which can also be called as demixing matrix.

$$\text{Therefore, } M = W^{-1}\ O \quad (5)$$

But, the problem is considerably more difficult since we do not know the mixing matrix W a priori. Blind means that we know very little if anything on the mixing matrix and make little assumptions on the source signals.

The block diagram of mixing and non-mixing matrix is shown in Fig. 1. The three signals M_1, M_2 and M_3 (which are assumed to be independent) are mixed by the random mixing network and they are observed as inputs to the microphones. The output from the microphone is given as inputs to ASN-RBF neural network. The weights are updated by the Stochastic Gradient descent Algorithm and the training continues until there is no change in the weight values in the consecutive iterations.

This problem can be solved by the classical method, Independent component Analysis (ICA), which assumes that the 2 sources are statistically independent of each other and non-Gaussian. ICA was originally developed to deal with problems that are closely related to cock-tail party problem. Since, the recent increase of interest in ICA, it has become clear that this principle has a lot of other interesting applications as well.

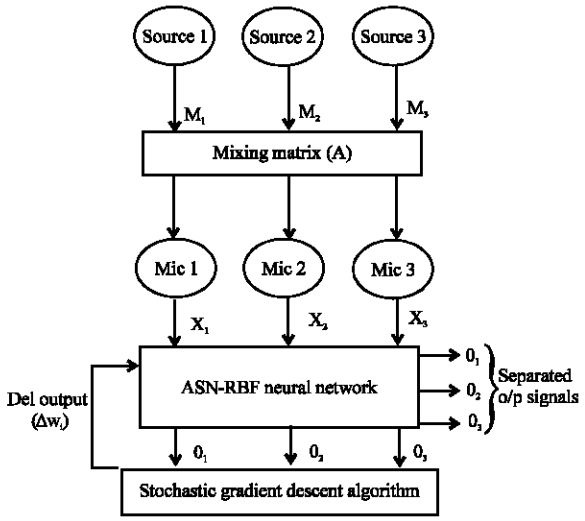

Fig. 1: This figure shows the block diagram of mixing and non-mixing matrix

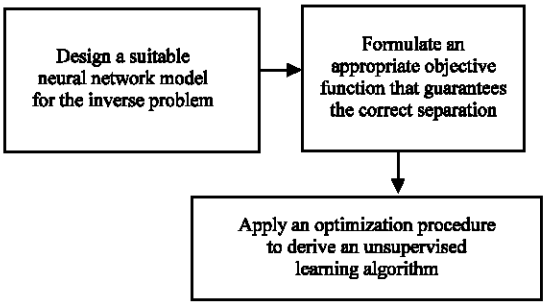

Fig. 2: This block diagram shows the design method to solve BSS problem

Without loss of generality, we can assume that both the mixture variables and the independent components have zero mean. If this is not true, then the observed variables can always be centered by subtracting the sample mean, which makes the model zero-mean (Hyvärinen, 1999).

The ICA can solve BSS problem under the following assumptions (Comon, 1994).

- The sources m_i are statistically independent.
- The sources must have non-Gaussian distributions.

But, by using ICA, the variances and the order of the independent sources can not be determined.

As shown in Fig. 2, it is necessary to design a suitable neural network for the blind source separation problem. In our research, we have developed ASN-RBF neural network since it exhibits fast training, simplicity and good generalization. To extract independent components from mixture signal, an appropriate objective function is chosen and it is minimized by the unsupervised learning algorithm.

STOCHASTIC GRADIENT DESCENT ALGORITHM (SGDA)

To separate independent components from the observed signal, we require an objective function. The objective function is chosen such that it should give original signals when it is minimized (Yogesh and Rai, 2002). In signal processing, when the components of the output vector become independent, its joint probability function factorizes to marginal pdfs, which is given in Eq. 6:

$$f(m) = \prod_{i=1}^{k} f_i(m_i) \tag{6}$$

where, m_i is the i^{th} component of the output signal. The pdf of m parameterized by W can be written as given in Eq. 7 (Papoulis, 1991):

$$f(m, W) = \frac{f_o(O)}{|J|} \tag{7}$$

where, |J| is determinant of the Jacobian matrix J. It can be defined as:

$$J = \begin{vmatrix} \frac{\partial m_1}{\partial o_1} & \frac{\partial m_1}{\partial o_2} & \cdots & \frac{\partial m_1}{\partial o_k} \\ \frac{\partial m_2}{\partial o_1} & \frac{\partial m_2}{\partial o_2} & \cdots & \frac{\partial m_2}{\partial o_k} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial m_k}{\partial o_1} & \frac{\partial m_k}{\partial o_2} & \cdots & \frac{\partial m_k}{\partial o_k} \end{vmatrix} \tag{8}$$

From Eq. 4, each element in Eq. 8 can be represented in terms of w as:

$$\frac{\partial m_i}{\partial o_j} = w_{ij} \tag{9}$$

Therefore, Eq. 8 can now be written as:

$$W = \begin{vmatrix} w_{11} & w_{12} & \cdots & w_{1k} \\ w_{21} & w_{22} & \cdots & w_{2k} \\ \cdots & \cdots & \cdots & \cdots \\ w_{k1} & w_{k2} & \cdots & w_{kk} \end{vmatrix} \tag{10}$$

Now Eq. 7 can be written as:

$$f(m,W)=\frac{f_o(O)}{|W|} \tag{11}$$

To extract independent components from the observed signal, the difference between the joint pdf and product of marginal pdfs has to be determined. When the components become independent, the difference becomes zero. This can be represented as:

$$f(m,W)-\prod_{i=1}^{k} f_i(m_i)=0 \tag{12}$$

Since, logarithm provides computational simplicity, taking logarithm on both sides of Eq. 12, we get

$$\log(f(m,W))-\sum_{i=1}^{k}\log\left(f_i(m_i)\right)=0 \tag{13}$$

Substituting the value of f(m,W) from Eq. 11 in Eq. 13, we get

$$\log\left(\frac{f_x(X)}{|W|}\right)-\sum_{i=1}^{k}\log\left(f_i(m_i)\right)=0 \tag{14}$$

Because, the pdf of the input vector is independent of the parameter vector W, the objective function for optimization becomes

$$\Phi(W)=-\log\left(|W|\right)-\sum_{i=1}^{k}\log\left(f_i(m_i)\right) \tag{15}$$

Now, the Edgeworth series has been used to expand the second term in Eq. 15. The first three terms of this expansion is

$$f_i(m_i)\approx\Psi(o)-\frac{\lambda_3\Psi^{(3)}(o)}{6\sqrt{n}}+\frac{1}{n}\left[\frac{\lambda_4\Psi^{(4)}(o)}{24}+\frac{\lambda^2{}_3\Psi^{(6)}(o)}{72}\right]+O\left(1/n^{3/2}\right) \tag{16}$$

Here, the random variables O_j have mean μ, variance σ^2 and higher cumulants $k=\sigma^r\lambda$. $\Psi^{(i)}(i)$ is the j^{th} derivative of $\Psi(m)$ with respect to m. Cumulants can also be expressed in terms of moments. The rth order cumulants are expressed in terms of moments as follows:

$$k_{i,3}=m_{i,3} \tag{17}$$

$$k_{i,4}=m_{i,4}-3m_{i,2}^2 \tag{18}$$

After simplification, the Gradient descent of Eq. 15 now becomes

$$-\frac{\partial\Phi(w)}{\partial w}=w^{-T}-\Psi(m)o^T \tag{19}$$

The Stochastic gradient descent algorithm for weight updation can now be written as:

$$w(t+1)=w(t)-\eta\frac{\partial\Phi(w)}{\partial w} \tag{20}$$

Substituting the gradient of the cost from Eq. 19, the weight update rule can now be written as:

$$w(t+1)=w(t)+\eta(t)\left[w^{-T}-\Psi(m(t))\right]o^T(t) \tag{21}$$

The advantage of Edgeworth series is that the error is controlled, so that it is a true asymptotic expansion.

Algorithm description

Step 1: Initialize the parameter.

- Assign weights between input and hidden layer.
- Assign weights between hidden and output layer.
- Setη = 0.99, σ = 0.09.

Step 2: Apply the input.

Step 3: Compute the output.

Step 4: Update weights between hidden and output layer

```
d1 = inv(det(hou_old));
for k = 1:output_neurons
```

$$d_2(k)=3\times y(k)^3\ 4\times y(k)^5+\frac{35}{12}\times y(k)^7$$
$$-5\times y(k)^9+\frac{41}{12}\times y(k)^{11}-\frac{7}{9}\times y(k)^{13}+\frac{1}{18}\times y(k)^{15}$$

```
end
deloutput=d1-(d2*mixtures');
w (t + 1) = w(t) + lrp* deloutput (k) + 0.05;
```

Step 5: Evaluate O(t)=W(t)*x(t).

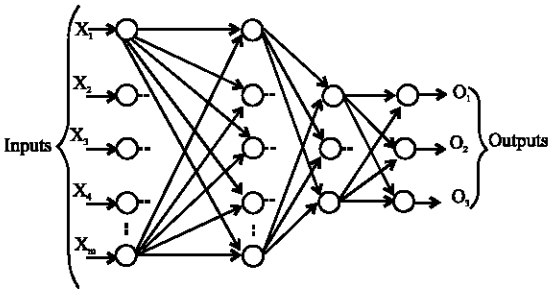

Fig. 3: This figure is the architecture of adaptive self-normalized radial basis function neural architecture

Step 6: Repeat steps 2-5 until a stable value of w (t) is obtained.

ASN-RBF neural architecture: To choose an architecture for a given problem, the generalization error has to be minimum and to get quantitative measures for it, we have to consider the characteristics of the network such as number of layers, number of nodes in hidden layer and connectivity: A priori information about the problem may be included here. The radial basis function neural network is proposed for BSS problem since the network exhibits rapid training, simplicity and generality. In recent years, there has been an increasing interest in using Radial Basis Function Neural Networks for many problems. Like Backpropagation and Counter propagation neural networks, it is a feedforward neural network that is capable of performing nonlinear relationship between the input and output vector spaces. The network consists of three layers: an input layer, a single layer of nonlinear processing hidden neurons and an output layer.

The ASN-RBF neural network architecture is shown in Fig. (3). The 1000 samples from input signals are given as inputs to input layer. The input layer behaves as fan-in, fan-out since it does not perform any computation (i.e.,) it does not process the inputs. The outputs from the input layer are given as inputs to hidden layer.

The output of the ASN-RBF neural network is obtained by the Eq. 22.

$$y_i = f_i(x) = \sum_{k=1}^{N} w_{ik}\phi_k(x, c_k) = \sum_{k=1}^{N} w_{ik}\phi_k\left(\|x - c_k\|\right) \quad (22)$$

for i =1,2,3, ……, m

where, $x \in R^{n+1}$ is an input vector and $\Phi_k(.)$ is a radial basis function, which is given by $e^{-D2_i}/(2\sigma)^2$ where, $D_i^2 = (X - W_{ik})^T (X - W_{ik})$, σ is the spread factor, which controls the width of the Radial Basis Function, w_{ik} are the weights in the output layer, N is the number of neurons in the hidden layer and $c_k \in R^{n\times1}$ are the RBF centers in the input vector space. For each neuron in the hidden layer, the Euclidean distance between its associated center and the input to the network is computed (Haykin, 1994). The output of the neuron in a hidden layer is a nonlinear function of the distance. Finally, the output of the network is computed as a weighted sum of the hidden layer outputs given by Output_of_outputn(b) = (input_to_outputn(b)/α), where, α is the scaling parameter, which determines the convergence of the learning algorithm. During training, if it is very low, the total error becomes NaN. It is increased gradually, so that for a particular value, the network converges and the error was reduced to acceptance value.

The centers c_k are defined points that are assumed to perform an adequate sampling of the input vector space. They are usually chosen as subset of the input data.

The weight vector W_{ik} determines the value of X, which produces the maximum output from the neuron. The response at other values of X drops quickly as X deviates from W, becoming negligible in value when X is far from W.

PERFORMANCE MEASURES AND EXPERIMENTAL RESULTS

To analyze the performance of an algorithm, we must know the following.

- When has something been learned?
- When is the network good?
- How long does learning take?

Learning means fitting a model to a set of training data, such that for a given set of input patterns, the desired output patterns are reproduced (Poggio and Girosi, 1989). Learning criterion: When we have binary output units, we can define that an example has been learned if the correct output has been produced by the network. In general, we need a global error measure (E) and we define a critical error E_c. The condition is that not only should the global error at every output node exceed a certain value; all patterns should have been learned to a certain extent. If the output units are continuous valued, e.g. within the interval [0,1], then we might define anything <0.4 as 0 and anything >0.6 as 1; whatever lies between 0.4-0.6 is considered as incorrect. In this way, a certain tolerance is possible. We also have

to distinguish between performance on the training and test set. So, we need a quantitative measure of generalization ability.

The performance of the proposed algorithm can be analyzed by the parameters given below.

Convergence: It minimizes the objective function. Minimizing the objective function can be visualized using error surfaces. The metaphor used is that the system moves on the error surface to a local minimum. The error surface or landscape is typically visualized by plotting the error as a function of two weights. Error surfaces represent a visualization of some parts of the search space. i.e., the space, in which the weights are optimized. Weight spaces are typically high-dimensional, so what is visualized is the error corresponding to just two weights. The error function depends not only on the data to be learned but also on the activations.

Convergence rates with stochastic gradient algorithm are typically faster than backpropagation algorithm. There is a lot of literature about improvements. In order to increase the convergence rate, the learning rate parameter η can also be changed over time:

$$\Delta\eta = \begin{cases} +c & \text{if } \Delta E < 0 \\ -d\eta & \Delta E > 0 \quad \text{for several steps} \\ 0 & \text{Otherwise} \end{cases} \tag{23}$$

where, c and d are parameters whose values are chosen between 0 and 1. There are various ways, in which the learning rate can be adapted. Newton, Steepest descent, Conjugate gradient and Quasi-Newton are all alternatives.

Local minima: One of the problems with all gradient descent algorithms is that they may get stuck with in local minima. There are various ways, in which they can be escaped. Noise can be introduced by shaking the weights. Shaking the weights means that a random variable is added to the weights. Alternatively, the algorithm can be run again using a different initialization of the weights. It has been argued that because the space is so high dimensional, there is always a 'ridge' where an escape from a local minimum is possible (Lappalainen and Giannakopoulos, 1999; Lee *et al*., 1997). Because error functions are normally only visualized with very few dimensions, one gets the impression that a backpropagation algorithm is very likely to get in a local minimum. This seems not to be the case with large dimensions. Since the proposed algorithm is stochastic, it does not up with local minima.

Performance index: After the separating matrix W has been computed by the stochastic gradient descent algorithm, the separation quality can be measured by the measure, so called performance index, (PI) (Cichocki and Amari, 2002) as:

$$\begin{aligned} PI(t) = &\frac{1}{m}\sum_{i=1}^{m}\left\{\sum_{j=1}^{m}\frac{|s_{ij}|^2}{\max_q |s_{iq}|^2} - 1\right\} \\ &+ \frac{1}{m}\sum_{j=1}^{m}\left\{\sum_{i=1}^{m}\frac{|s_{ij}|^2}{\max_q |s_{qj}|^2} - 1\right\} \end{aligned} \tag{24}$$

where, s_{ij} denotes the i^{th} row and j^{th} column of P, (P = A.W)

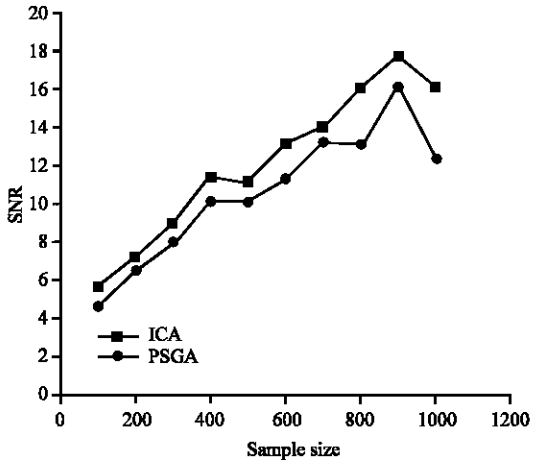

Fig. 4: Comparision of SNR for ICA and SGDA algorithm

Table 1: SNR of separated signals using ICA

Sample size (X)	100	200	300	400	500	600	700	800	900	1000
SNR	5.7	7.2	9.0	11.4	11.0	13.1	14.0	16.0	17.8	16.0

Table 2: SNR of separated signals using SGDA

Sample size (X)	100	200	300	400	500	600	700	800	900	1000
SNR	4.6	6.5	8.0	10.1	10.0	11.2	13.2	13.0	16.0	12.3

Signal to noise ratio: Another measure of separation quality used was the Signal to Noise Ratio (SNR) of the separated outputs, given by the Eq. (25) (Guillermo *et al*., 2003).

$$SNR = 10\log_{10}\left(\frac{\sum\left[m(t)^2\right]}{\sum\left[n(t)^2\right]}\right) \quad (25)$$

where, m(t) is the desired signal and n(t) = o(t)-m(t) is the noise indicating the undesired signal. o(t) is the estimated source signals.

From Quantitative analysis of Table 1 and 2, it has been observed that the Signal to Noise (SNR) of the separated signals by the Proposed Stochastic Gradient Descent Algorithm is much lower than that of ICA algorithm and its comparison for 1000 sampled data is shown in Fig. 4.

EXPERIMENTAL RESULTS

The three audio signals, which are given below in Table 3 are nonlinearly mixed by the mixing matrix A given in Eq. 26, which is generated by the matlab function rand().

$$A = \begin{bmatrix} 0.9501 & 0.4860 & 0.4565 \\ 0.2311 & 0.8913 & 0.0185 \\ 0.6068 & 0.7621 & 0.8214 \end{bmatrix} \quad (26)$$

The learning rate parameter was initially chosen as 0.99 and it is varied by the Eq. 23 during training. The spread factor for radial basis function is set at 0.09. The centre of the basis functions are initially set to the weight matrix between input and hidden layer. The original,

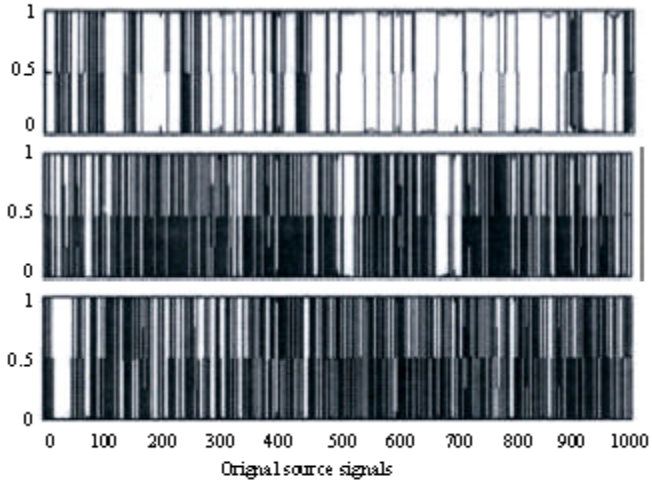

Fig. 5a: Original source signals

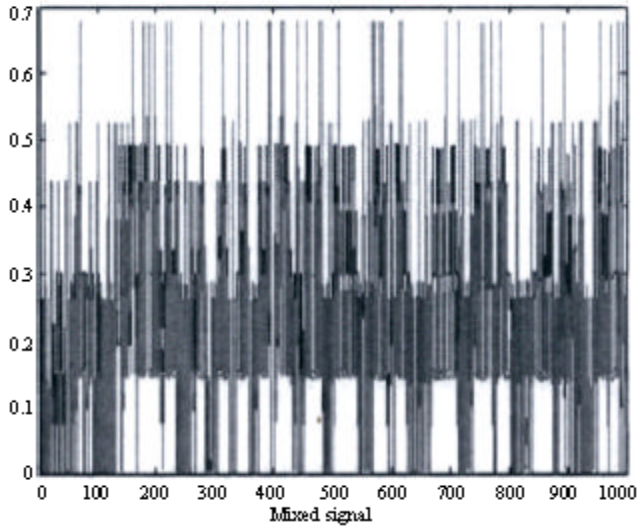

Fig. 5b: Mixture signal

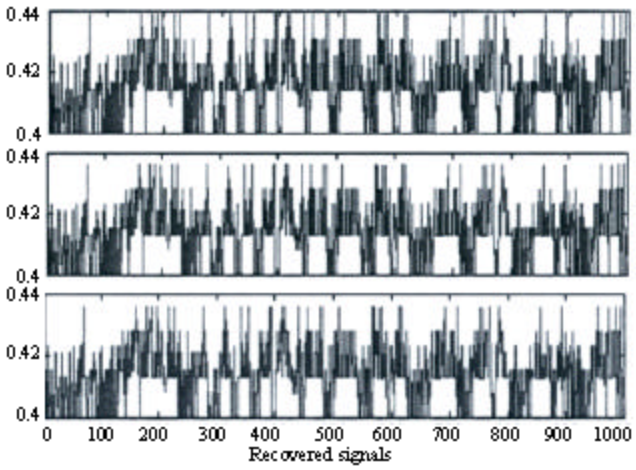

Fig. 5c: Recovered source signal

Table 3: Three source signals used for simulation

Parameter	Source 1 (beans.wav)	Source 2 (wsong2.wav)	Source 3 (ssong1.wav)
Bit rate	64 Kbps	48 Kbps	176 Kbps
Sample size	8-bit mono	8-bit mono	8-bit mono
Sample rate	8 Khz	6 Khz	22 Khz
No.of samples	1000	1000	1000

mixture and separated output signals are shown in Fig. 5. Figure 5a shows the signals, which are generated artificially and the Fig. 5b shows the mixture signal after the three source signals are mixed by the mixing matrix 'A' and the Fig. 5c represents the recovered signal from ASN-RBF neural network.

CONCLUSION

This study proposes a novel neural learning algorithm for extracting independent sources from an artificially mixed signal. An adaptive self-normalized Radial Basis Function neural network has been developed and trained by the proposed stochastic gradient descent optimization algorithm with fixed centers to update the parameters in the generative model. Many different approaches have been attempted by numerous researches using neural networks, each claiming various degrees of success. But, the separation of signals in real environments is still to be improved. The performance of the proposed network is compared with the ICA algorithm and it is illustrated with computer simulated experiments.

REFERENCES

Burel, G., 1992. Blind separation of sources: A nonlinear neural algorithm. Neural Networks, 5: 937-947.

Cardoso, J.F. and B. Laheld, 1996. Equivariant adaptive source separation. IEEE. Trans. Signal Proc., 43: 3017-3029.

Cichocki, A. and S.I. Amari, 2002. Text book titled. Adaptive Blind Signal and Image Processing. Wiley.

Comon, P. and C. Jutten *et al*., 1991. Blind separation of sources, part II: Problem statement. Signal Proc., 24: 11-20.

Comon, P., 1994. Independent component analysis: A new concept?, Signal Proc., 36: 287-314.

Deco, G. and W. Brauer, 1995. Nonlinear higher-order statistical decorrelation by volume-conserving neural architectures. Neural Networks, 8: 525-535.

Guillermo Bedoya and Sergio Bermejo *et al*., 2003. Comparison of Neural Algorithms for Blind Source Separation in Sensor Array Applications. ESANN proceedings-European Symposium on Artificial Neural Networks, Bruges (Belgium), pp: 131-136.

Haykin, S., 1994. Neural networks. A Comprehensive Foundation: Prentice-Hall, pp: 256-290.

Herault, J. and C. Jutten, 1986. Space or time adaptive signal processing by neural network models. In: Proc. AIP Conf. Snowbird, UT, pp: 206-211.

Herrmann, M. and H.H. Yang, 1996. Perspectives and limitations of self-organizing maps in blind separation of source signals. In: Progress in Neural Information Processing: Proc. ICONIP, pp: 1211-1216.

Hyvärinen, A., 1999. New approximations of differential entropy for independent component analysis and projection pursuit. In: Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press, 10: 273-279.

Hyvarinen, A. and P. Pajunen, 1999. Nonlinear independent component analysis: Existence and uniqueness results. Neural Networks, 12: 429-439.

Jutten, C. and J. Herault, 1991. Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture. Signal Proc., 24: 1-20.

Lappalainen, H. and X. Giannakopoulos, 1999. Multilayer perceptrons as nonlinear generative models for unsupervised learning: A Bayesian treatment. In: Proc. Int. Conf. Artificial Neural Networks (ICANN), Edinburgh, U.K., pp: 19-24.

Lee, T.W. and B. Koehler *et al*., 1997. Blind source separation of nonlinear mixing model. In: Proc. IEEE Workshop, Neural Networks for Signal Proc. VII, pp: 405-415.

Li, S. and T.J. Sejnowski, 1995. Adaptive separation of mixed sound sources with delays by a beamforming Herault-Jutten network. IEEE J.Oceanic Eng., 20: 73-79.

Lin, J.K. and D.G. Grier *et al*., 1997. Source separation and density estimation by faithful equivariant SOM, in Advances in Neural Inform. Proc. Syst. Cambridge, MA: MIT Press, 9: 536-541.

Linde, Y. and A. Buzo *et al*., 1980. An algorithm for vector quantizer design. IEEE. Trans. Commun., 28: 84-95.

Lindgren, U.A. and H. Broman, 1998. Source separation using a criterion based on second-order statistics. IEEE Trans. Signal Proc., 46: 1837-1850.

Makeig, S. and A. Bell *et al*., 1996. Independent Component Analysis in Electroencephalographic Data. Advances in Neural Information Processing Systems. In: Mozer, M. *et al.* (Eds.). Cambridge, MA: MIT Press, 8: 145-151.

Pajunen, P., 1998. Blind source separation using algorithmic information theory. In: Neurocomput. Elsevier, 22: 35-48.

Pajunen, P. and A. Hyvarinen *et al*., 1996. Nonlinear blind source separation by self-organizing maps. In: Progress in Neural Information Processing: Proc. ICONIP, New York, 2: 1207-1210.

Papadias, C.B. and A. Paulraj, 1997. A constant modulus algorithm for multi-user signal separation in presence of delay spread using antenna arrays. IEEE Signal Proc. Lett., 4: 178-181.

Papoulis, A., 1991. Probability, Random Variables and Stochastic Processes. 3rd Edn. McGraw-Hill International Edition.

Poggio, T. and F. Girosi, 1989. A theory of networks for approximation and learning. MIT, Technol. Rep. AI, 1140: 11-17.

Taleb, A. and C. Jutten *et al*., 1998. Source separation in post nonlinear mixtures: An entropy-based algorithm. In: Proc. ESANN, pp: 2089-2092.

Van der Veen, A.J. and S. Talvar *et al*., 1997. A subspace approach to blind space-time signal processing for wireless communication systems. IEEE Trans. Signal Proc., 45: 173-190.

Yang, H.H and S. Amari *et al*., 1997. Information backpropagation for blind separation of sources from nonlinear mixture. In: Proc. IEEE ICNN, TX, pp: 2141-2146.

Ying, T. and J. Wang *et al*., 2001. Non linear blind source separation using Radial Basis Function Neural Network. IEEE Trans. Neural Networks, 12 (1): 124-134.

Yogesh, S. and C.S. Rai, 2002. A neural approach to independent component analysis. J. CSI, 31 (4): 19-23.