2004 IEEE International Conference on Systems, Man and Cybernetics

Agent Swarm Regression Network ASRN

Chi-kin Chow and Hung-tat Tsui
Visual Signal Processing and Communications Laboratory
Department of Electronic Engineering
The Chinese University of Hong Kong
{ckchow1, httsui}@ee.cuhk.edu.hk

Abstract - *A multi-agent system (MAS), with independent software agents interacting with each other to achieve common goals will complete concurrent distributed tasks under autonomous control. In this paper, novel RBF Regression Network – "Agent Swarm Regression Network ASRN" is proposed and will be trained by a MAS . Each neuron of the ASRN is considered as an agent, which consists of per-defined simple agent behavior set. After a sufficient number of iterations, the weights of neurons can be determined. Two sets of experiment will be examined to observe the effectiveness of the proposed method.*

Keywords: Multi-Agent System, RBF neural network, Regression

1 Introduction

Radial Basis Function (RBF) network is a multi- layers feedforward network using radial basis functions in the solution of the real multivariate interpolation problem. The RBF network consists of three functionally distinct layers. The input layer is simply a set of sensory units. The second layer is a hidden layer of sufficient dimension. It applies a non-linear transformation of the input space to a higher dimension hidden-unit space. The third layer performs a linear transformation from the hidden-unit space to the output space. It has been applied successfully in a number of applications including image processing [3], speech recognition [6, 1, 9], time series analysis and adaptive equalization [7, 4].

Recently, many researchers focus on the training algorithm of the most popular RBF neural network – Support Vector Machine (SVM), which was firstly proposed by V. N. Vapnik [8]. In the structure of SVM, centers and variances of the neurons are fixed while the weights are determined by optimizing a constrained high dimensional quadratic equation. This approach is suffered from the problem of long computational time and large memory requirement. Many modified SVM training algorithms were proposed to tackle this problem. Freitas [5] suggested using Linear Particle Swarm Optimization to improve the scalability. Paquet [2] modeled the SVM training as an sequential process so as to reduce memory size required.

In this paper, we proposed a self-constructed RBF regression network called "*Agent Swarm Regression Network ASRN*". It has the advantages of a SVM: fixing the centers and variances of the RBF so that then parameter space is reduced In addition, by using the MAS based training algorithm, the time for weight determination can be greatly reduced.

This paper is organized as follows. In section 2, we propose a generalized architecture of MAS. The MAS representation of the ASRN is discussed in Section 3. Two sets of regression experiments will be examined so as to illustrate the performance of the proposed algorithm in Section 4. A conclusion is drawn in Section 5.

2 General Architecture of MAS

Various definitions from different disciplines have been proposed for the term Multi-Agent System. As seen from Distributed Artificial Intelligent (DAI), a multi-agent system is a loosely coupled network of problem-solver entities that work together to find answers to problems that are beyond the individual capabilities or knowledge of each entity. More recently, the term multi-agent system has been given a more general meaning. It is now used for all types of systems composed of multiple autonomous components showing the following characteristics:

- Each agent has incomplete capabilities to solve a problem.
- There is no global system control.
- Data is decentralized.
- Computation is asynchronous.

The evolution of an MAS can be decomposed into 3 stages. Initially, an agent collects information from the environment and its neighbor(s) in order to conclude its local environment observation (LEO). Base on the LEO, the agent will draw a decision from its decision-making unit, which benefits to its current situation. Afterward, the agent executes the decision by adjusting its internal parameters. Since the agent swarm keeps interact with the environment, there is no beginning nor terminating stages. In addition, as the swarm undergoes those 3 stages in parallel instead of sequentially, a more complicated swarm behavior is constructed from a set of simple rules.

3 RBF Network as MAS

In this section, we model the training algorithm of the ASRN as an MAS. By considering each training sample as a neuron, a RBF network with optimal size can be achieved if:

1. The number of zero-magnitude neurons is minimized.
2. The sum of regression error is minimized.

However, there is a contradiction between the conditions. Therefore, the algorithm aims at constructing an optimal size network such that each sample's regression error is less than a given threshold, rather than the one with minimum total regression error. Instead of applying a passive approach that determines the set of zero-magnitude neurons, we demand each neuron actively to suppress its magnitude as much as possible. Therefore, the neurons can be regarded as an agent swarm, while the criteria listed above can be treated as the behavior of the agents.

3.1 Parameter of ASRN agent

For a MAS-based application, the features of an agent are described by its abilities. In this application, the ASRN agent consists of 8 parameters. 4 of them are adjustable during the iterations while the remaining 4 parameters are predefined at the beginning of swarm evolution. The follows are the details of the agent parameters.

i. Magnitude M – It characterizes the magnitude of an agent, and hence the weight affects to the whole regression network. The magnitudes of the whole swarm are assigned as one initially.

ii. Reduced Magnitude ∂M – Quantity of magnitude to be reduce at the current evolution

iii. Position vector μ – Position of the training sample.

iv. Desired Output Y – Desired Output of the training sample.

v. Variance σ – This parameter describes the correlation among the agent with its neighbor(s). To simplify the training process, all neurons of the network are set to be the same value. Instead of predefining the value by the user, it is suggested by the following equation:

$$\sigma = p^{-\frac{1}{D}}(-8\ln 0.8)^{-\frac{1}{2}}$$

where p is the swarm size and D is the swarm dimension

vi. Local Environment Observation (LEO) O – The regression value at μ.

vii. Neighbor Linkage $\vec{N}$ – It is a P by 1 vector, i.e. $N_i = [n_{i,1}\ n_{i,2}\ \dots\ n_{i,P}]$. $n_{i,j}$ equals to 1 if Agent A_j is the neighbor of A_i. Otherwise, $n_{i,j}$ is set to be zero. Therefore, $N_i \bullet N_i^T$ represents the number of neighbors that A_i communicates with.

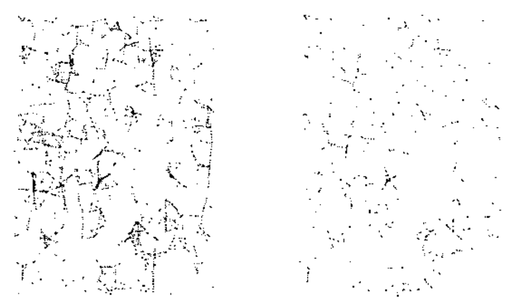
viii. Magnitude Decay Rate α – This parameter controls the magnitude decrement rate of an agent. Instead of a constant decay rate along the iterations, the rate of decrement is changed adaptively. The adaptive strategy will be discussed at the Section 3.4

3.2 Neighbor of ASRN agent

Different from DAI that every agent can collect information from all the others, the MAS agents can only communicate with limited amount of agents inside the swarm. We named the agents that can be communicated by agent A_i as the neighbors of A_i. It is obvious that as the size of neighbor increase, the more information can be collected and hence a more accurate decision can be drawn. On the other hand, more time is needed to analysis the collected information. Therefore, for a robust MAS, the agents should achieve the global objective by communicating with minimum number of neighbors. In the case of ASRN, the neighbors of an agent A_i are defined as follows:

1. Calculate the *Contribution* of agent A_j: $\{G_{i,j}\}$, i.e. $G_{i,j} = exp(-\|\mu_i - \mu_j\| / 2\sigma^2)$ for all $j \neq i$.
2. Sort $\{G_{i,j}\}$ in the descending order.
3. Assume $R(j)$ be the rank of A_j.
4. By giving the decentralized error threshold β, agent A_k is the neighbor of A_i if:

$$\sum G_{i,k} \le (1-\beta)\sum_{m=1}^{P} G_{i,m} \text{ for } R(k) \ge R(h)$$



(a) β = 0.005 (b) β = 0.01

Fig. 1 The swarm linkage examples.

Fig. 1 illustrates 2 examples of agent neighbor linkages with different values of β. Circles indicate the position of agents and the straight lines represent the neighborhood linkage among the swarm. It is found that more neighborhood linkages are established at the case of smaller β.

3.3 LEO of ASRN agent

Theoretically, the observed regression value is defined as:

$$O_i = M_i + \sum_{j=1}^{N_i} M_{I_{i,j}} e^{-\frac{|\mu_i - \mu_{I_{i,j}}|}{2\sigma^2}} \quad (1)$$

where $I_{i,j}$ is the agent index of j^{th} neighbor of i^{th} agent. However, since the positions and the variances of all agents are fixed, the LEO can be rewritten as a linear equation in which the training process is accelerated.

3.4 Decision Making Unit of ASRN agent

Based on the constraints of an ideal network described at Section 3, the behavior of an agent can be formulated at below such that the swarm converges to an optimal network:

1. Determine the regression error E_i, i.e. $E_i = Y_i - O_i$
2. *If* Maximum Error Threshold $(E_{max}) \geq |E_i|$ *then*
   $\alpha_i \leftarrow 2\alpha_i$; $\partial M_i \leftarrow \alpha_i \times M_i$
   Else
   $\alpha_i \leftarrow 0.5\alpha_i$; $\partial M_i \leftarrow M_i - E_i$
   EndIf

3.5 Action Unit of ASRN agent

After determining the quantity of magnitude reduced, the agent take an action by simply decreasing it magnitude, i.e. $M_i \leftarrow M_i - \partial M_i$.

4 Experimental Results

4.1 Regression of 2D Sinc function

In the first experiment presented here, we apply the proposed system to a two-inputs, one-output function approximation problem. The function chosen is

$$y = \sin(d)/d \text{ where } d = \sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2}$$

over the interval [0, 1]. This function is chosen since the input-output complexity varies over the allowed range. A swarm with size from 2500 to 2000 are presented in the allowed interval (fig. 2a) where σ and E_{max} are chosen as 1 and 0.05 respectively. In addition, the positions of the agents are random assigned in the allowed range with the corresponding desired output value are provided. The neighbors of each agent are defined by the formulation described at Section 3.2b with $c = 0.8$ and $\beta = 0.01$. Fig. 2a shows the distribution of the 200 samples trial. Fig. 2b illustrates the corresponding resultant magnitude of the swarm. The agent with higher intensity indicates its relative high magnitude. Fig. 2c shows the reconstructed mapping.

Table 1 Results of Experiment 4.1

Sample Size	Processing Time (sec.)	No. of non-zero magnitude agent	Mean Square Error
2,500	8	42	0.058
5,000	20	49	0.037
10,000	51	50	0.031
20,000	132	57	0.026

4.2 Image Representation

Chow [3] suggests representing an image I(x,y) as an mixture of Gaussians:

$$I(x,y) = \sum_i M_i \exp(-\frac{(\alpha_i - x)^2 + (\beta_i - y)^2}{2\sigma^2}) + b$$

By treating each pixel as a training sample, *Chow* uses the Support Vector Regression to determine the weight of each Gaussian function, and hence the continuous representation of an image.

In this section, we repeat the experiment at the Ref. [3] by replacing SVM with ASRN to determine the weights. The target 256 × 256 image in this experiment is shown at fig. 3a. This experiment was performed at the PC platform with 1.7GHz CPU and 256MB memory. It takes approximately 48 sec. to obtain the resultant weights at fig. 3b, and the corresponding reconstructed image is shown at fig. 3c. Fig. 4 illustrates the swarm's PSNR over the iterations. We observe that the global behavior of the agent swarm can be divided into 2 stages. At the first stage, the agents try to minimize their regression error, i.e. the value of PSNR increase along with the iterations. However, as the PSNR of the swarm reach to its maximum value, all agents change to suppress their magnitudes under the constraint that their regression errors are smaller than a given threshold.

Table 2 Results of Experiment 4.2

	ASRN	SVM
Processing Time (sec.)	112	10431
No. of non-zero weight neurons	32356	32987
PSNR	27.65	27.54

To verify the performance of proposed algorithm, we sorted the neuron with ascending order of magnitude. By reconstructing the target image with various percentages of sorted neurons (fig. 5), we observe the corresponding PSNR has a slightly decrease at the first 40%. Therefore, it can be concluded that the proposed algorithm determines 40% percentage of agents (neuron/pixel) is redundant, which is similar to the result found at Ref. [1]. Fig. 6 show the reconstructed image by using the largest 25%, 40%, 55%, 70%, 85% and 100% magnitude agents.

5 Conclusion

It shown that a fixed-center RBF network can be modeled by a MAS effectively. We propose to decompose the training algorithm to a parallel and straightforward agent behavior, which makes the algorithm simple to be implemented. In addition, the neighborhood concept of MAS speed up the training process by comparing with SVM. Accurate and scalable training results were illustrated so as to demonstrate the acceptable performance of the algorithm.

References

[1] Changxue Ma, Randolph M.A., Drish J., "A support vector machines-based rejection technique for speech recognition", *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing,* 2001 (ICASSP '01). 2001, Volume 1, 7-11 May 2001 Page(s): 381 -384 vol.1

[2] de Freitas N., Milo M., Clarkson P., Niranjan M., Gee A., "Sequential support vector machines", *Neural Networks for Signal Processing IX,* 1999. Proceedings of the 1999 IEEE Signal Processing Society Workshop, 23-25 Aug. 1999, Page(s): 31 -40

[3] Kai Tik Chow D., Tong Lee, "Image approximation and smoothing by support vector regression", *Proceedings of International Joint Conference on Neural Networks 2001,* Volume: 4, 15-19 July 2001, Page(s): 2427 - 2432, vol. 4.

[4] Kumar P.C., Saratchandran, P., Sundararajan N., "Communication channel equalisation using minimal radial basis function neural networks", *Neural Networks for Signal Processing VIII,* 1998. Proceedings of the 1998 IEEE Signal Processing Society Workshop, 31 Aug.-2 Sept. 1998, Page(s): 477 -485.

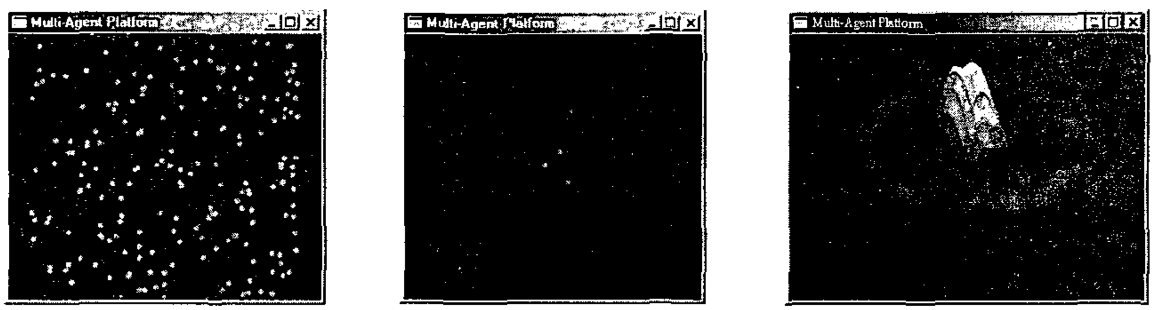
[5] Sankar R., Sethi N.S., "Robust speech recognition techniques using a radial basis function neural network for mobile applications", *Proceedings of IEEE Southeastcon* '97. 'Engineering New Century', 12-14 April 1997, Page(s): 87 -91.

[6] Sebald D.J., Bucklew. J.A., "Support vector machine techniques for nonlinear equalization", *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Volume: 48 Issue: 11, Nov. 2000, Page(s): 3217 -3226.

[7] Paquet U., Engelbreebt A.P., "Training support vector machines with particle swarms", *Proceedings of the International Joint Conference on Neural Networks,* 2003. Volume:2, July 20 -24, 2003, Page(s): 1593 -1598

[8] V. N. Vapnik, C. Cortes, "Support Vector Networks", *Machine Learning,* 20(3), 273-297, 1995.

[9] Yue Xicai, Ye Datian, Liu Ming, "Text-independent speaker identification by genetic clustering radial basis function neural network", *Proceedings of the 23rd Annual International Conference of the Engineering in Medicine and Biology Society IEEE,* 2001, Volume: 2, 25-28 Oct. 2001, Page(s): 1777 -1780 vol.2.



(a) (b) (c)

Figure 2 (a) Position of the agent swarm. (b) Magnitude plot of resultant agent swarm. (c) Resultant regression surface.

(a) (b) (c)

Figure 3 (a) Target Image – "Pepper". (b) Resultant magnitude plot of swarm (c) Resultant reconstructed image.

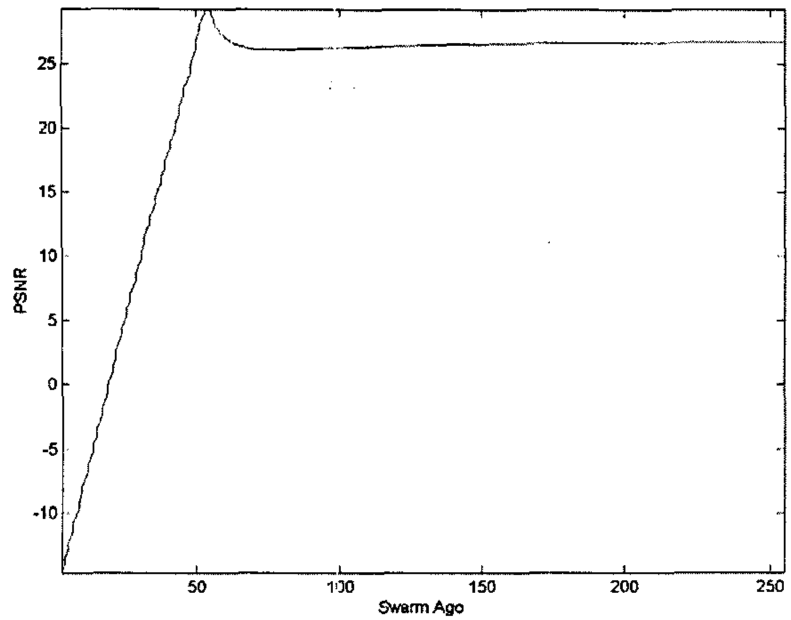


Figure 4 PSNR against the swarm ago

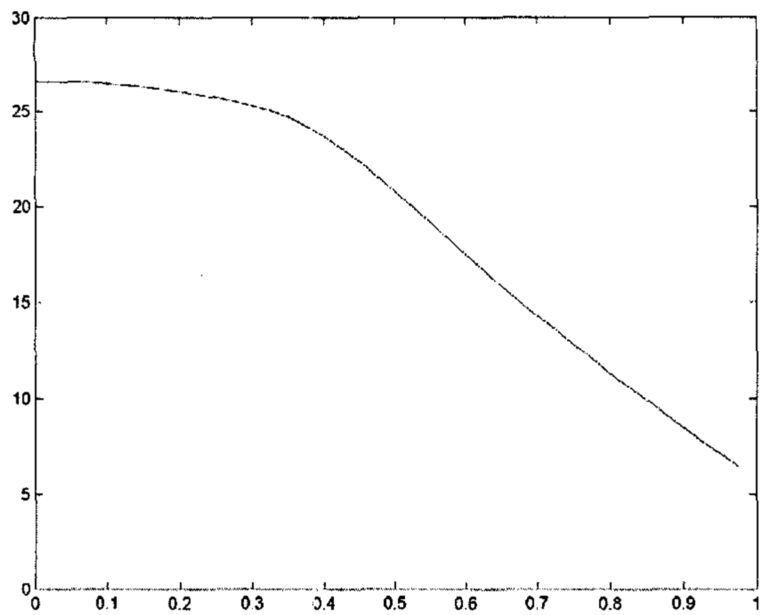


Figure 5 Resultant regression PSNR against various percentages of deactivated agents.


(a)

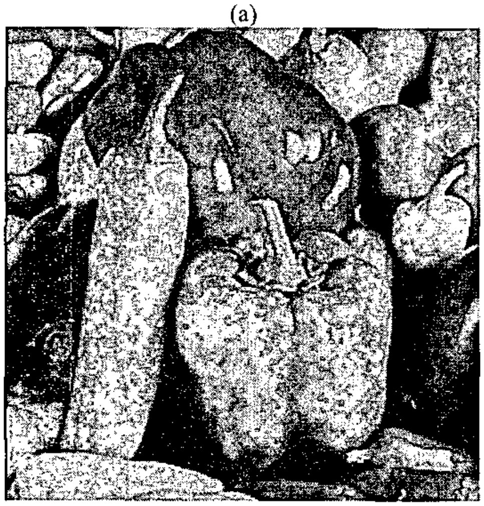
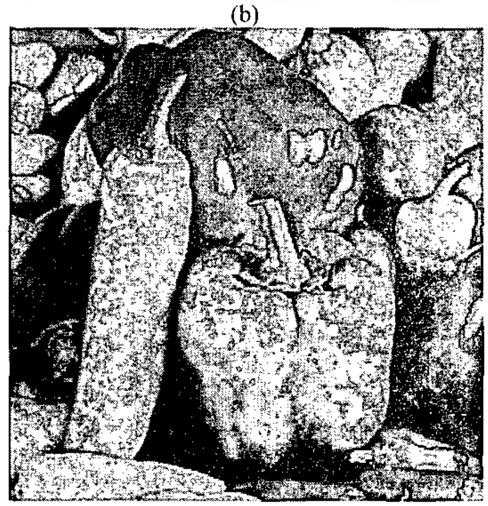
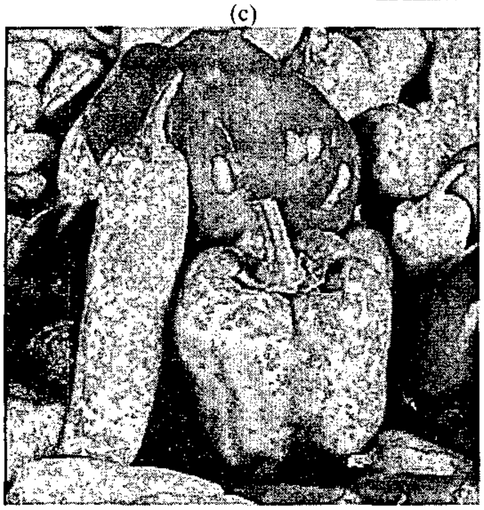
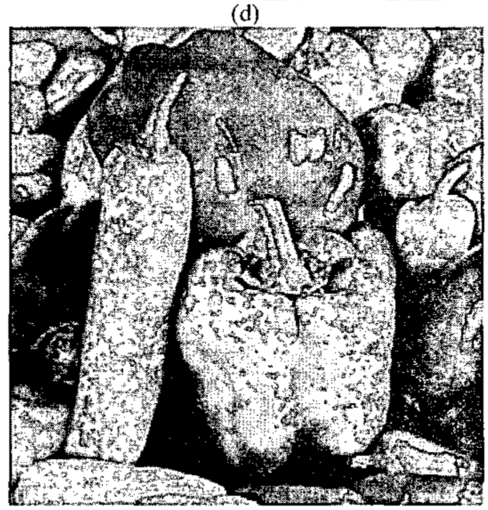
(b)

(c)

(d)

(e)

(f)

Figure 6 Reconstructed image by using the largest (a) 25%, (b) 40%, (c) 55%, (d) 70% (e) 85% and (f) 100% magnitude agents.