Journal of Computer Science 1 (1): 1-6, 2005
ISSN 1549-3636

A New Look to Adaptive Temporal Radial Basis Function Applied in Speech Recognition

[1]Mesbahi Larbi and [2]Benyettou Abdelkader
[1]Department of Computer Science, Dr. Moulay Tahar, Saida University, Algeria
[2]Signal-Image-Parole Laboratory, BP 1505, Department of Computer Science
USTO-MB, Oran, 31000, Algeria

Abstract: This study presents new contribution towards the Adaptive Temporal Radial Basis Function (ATRBF) applied to Continuous speech recognition, in particular the recognition of phonemes like Timit Corpus. ATRBF combines features from Time Delay Neural Network (TDNN) and the advantages of Radial Basis Function (RBF). The capacity to detect the acoustic features and their independent temporal report of the temporal localisation is inspired from the TDNN model. The main use of RBF is both their speed of treatment and few parameters to adjust for the training phase, which encourages to apply this model to new tasks in most delicate cases.

Keywords: ATRBF, TDNN, RBF, Speech Recognition, Temporal Localisation

INTRODUCTION

A successful speech recognition system has to determine features not only present in the input pattern at one point in time, but also features of the input pattern changing over time [1, 2].

The classic methods based on multilayer perceptron use the TDNN network, it is the first model used by Weibel in the speech recognition domain [1]. But the problem was the hard time processing and the adjustment of parameters that become a laborious stain for the new applications.

In the opposite, the RBF networks don't require a special adjustment and the training time becomes shorter with regard to the TDNN. But the problem of RBF is the shift invariant in time[1].

The goal to combine the approach of the RBF with the shift invariance features of the TDNN, can be get a new robust model, this is named Temporal Radial Basis Function (TRBF) , but to be more efficient, we have adapt these networks so that they come more dynamic according to their behaviour and features of the object has study. It can be goes more clearly in the continuous word. Therefore to adapt the TRBF networks, it was necessary to develop an algorithm that permits to solve this type of problem, this algorithm is called (DOLS) which means Dynamic Orthogonal Least Square, presented in this study.

To allow a RBF network to detect features in time, it is necessary to not only present inputs to a point data, but in many passages. In this case, we take a structure of a classic RBF (Fig. 1).

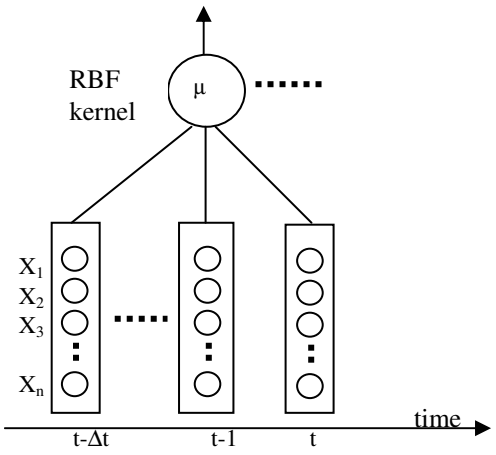

Fig. 1: An RBF Network Sweeping a Temporal Window of Δt Delay

We notice in this network that the input vector have a large window, what burdens the time delay count between centres and inputs.

$$d(t)=\sum_{\tau=0}^{\tau=\Delta t}\sum_{i=0}^{i=n-1}(o_i^{\tau}-\mu_{i,\tau})^2. \quad (1)$$

$$a_i(t)=\varphi_i\left(\frac{-d(t)}{\sigma}\right),\ \varphi_i : kernel(gaussian-triangular) \quad (2)$$

$d(t)$ represent the Euclidian distance between the input (o_i^{τ} : ith component of input vector at τ time) and the

centre ($\mu_{i,\tau}$) of this RBF, $a_i(t)$ represent the activation function (kernel) , which depend on the distance $d(t)$.
Among the major inconveniences of this model we cite the inability of adaptation towards the temporal transfer of the input vector, that is to say if the shape presented to the network already represents a prototype learned but shifted in time, this RBF network doesn't answer appropriately [1]. To solve this type of problem, we propose to prolong the time for the hidden layer (Fig. 2).

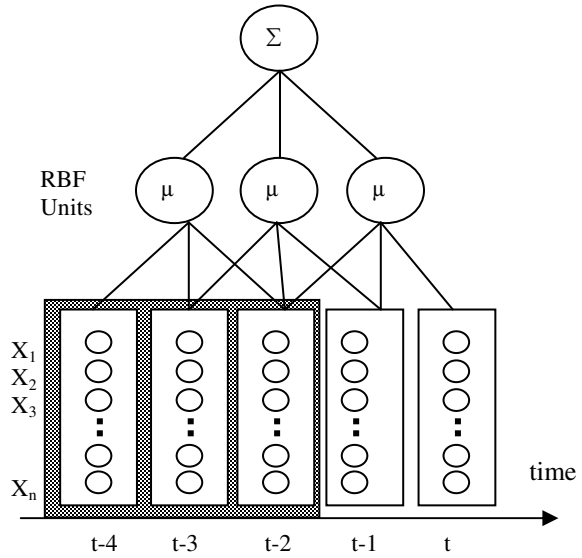

Fig. 2: This T-RBF Network of Shape 3(3) Composed of a Input Window of 5 Units Length with a Block of 3 Units in Time Delay and Generating in Result 3 Hidden Neurons. For Simplicity only one RBF Network is Conceived for Each Class

In this kind of network the RBF units only represent a narrow time window to facilitate the count in the following layer that adds the different processes of these RBF through time. The notation « x(y) » represents sizes of these windows, x indicates the number of delays of input and y represents the size of the window taken for the hidden layer integration.
We remark according to this network, that time limits are fixed and we can not change them during the training. To this effect we propose a variant that permits to solve this problem (Fig. 3).
In this type of architecture, the goal is to get an elevated precision, in addition we can adapt for every shape the network while playing on the number of block in the hidden layer and this is in minimizing the size of the block of input time delay. Therefore the alone parameters to adjust are the size of blocks of input windows and the size of time delays and the number of blocks in the hidden layer. We notice that the taken time delay number for the input depends on he application, generally in recognition of the word, the size of the observation window vectors is in the order of 10 to 15, therefore the number of time delay is understood between 5 to 10.
Concerning the number of blocks of the restrained RBF in the hidden layer is fixed either according to the wanted training rate or following a error threshold.

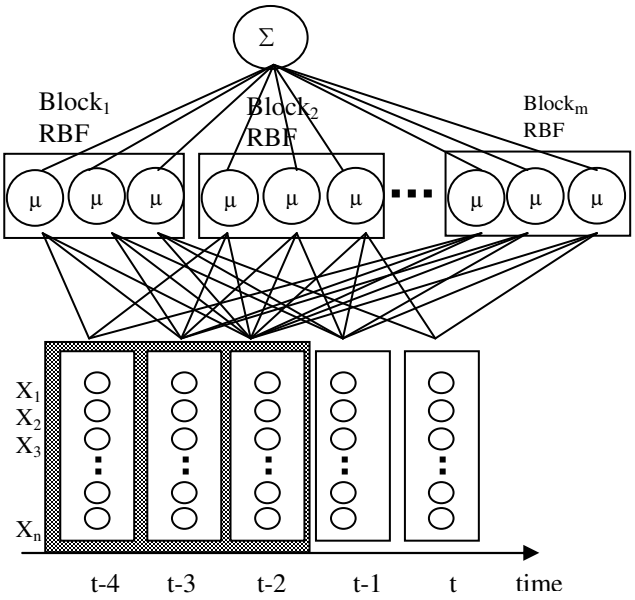

Fig. 3: This ATRBF Network of Shape 3 (3) Composed of a Window of Input Length 5 Units with a Block of Time Delay of 3 Units Generating in result m Hidden Blocks, Each Composed of 3 Hidden Nerons. For Simplicity Only One RBF Network is Conceived for Each Class

We make the remark on the speed of training that characterizes our type of network in comparison to other architectures as networks based on back-propagation.
The classic RBF can be formed to accomplish tasks of the pattern recognition with no linear and complex Contours [3], they are limited to treat some static models, rather than to treat shapes that are temporal nature. The training of such network requires some techniques to adapt the hidden neurons with passages of input windows so that it is an adequate integration of information according to the temporal interval.
Several methods of training of this network are proposed e.g. A first method permits to create centres of unsupervised manner follow-up of the weight count between the hidden and output layers [4].
A second method characterised by incremental training, which means that the creation of a centre is followed by the count of his correspondent weight [1, 4, 5].
The Temporal RBF, like ATDNN [6], LSTM are proposed to defeat the static limitation [7]. Networks

with this capacity can play an important role in applications domain, possessing some dynamic properties in pattern recognition e.g., the no stationary signals and the dynamic shapes. Also to take part of classic RBF advantages in approximation and recognition, the objective come closer toward a behaviour wanted by a collection of functions, named kernels [1]. A kernel is characterised here by a centre and a receptor field r, these kernels can be chosen by k-means clustering or the quantification techniques.
All these parameters can be taken in consideration by integrating a Bayesian neuronal classifier or to be optimised with the neuro-genetic techniques. In addition, we can combine this approach with other techniques as the Hidden Markov Models "HMM" while using the generated probabilities. In the following part we describe the algorithm of training that makes part of incremental method.

Algorithm of OLS: For this algorithm OLS (Orthogonal Least Square), we suppose that the kernel function ϕ is fixed and that is the same for every hidden cell, the initial set of this centres must be fixed also. Therefore this algorithm permits to make an incremental training [6]:

* First it makes the linear separation between the input layer and the hidden layer, it creates the hidden neurons automatically while applying the Gram-Schmidt orthogonalisation, that permits to eliminate redundancies of information. Other methods consist in using the genetic algorithms to minimize the number of hidden neurons with a good generalization.
* Secondly to make the training between the hidden layer and the output layer, using the least square method while calculating synaptic weights.

The OLS algorithm, conceived in origin for no linear system identification, can apply the RBF network that can be considered like a particular case of the linear regression model definite by :

$$d(t) = \Sigma P_i(t)\ \theta_i + \varepsilon(t) \qquad (3)$$

when $d(t)$: the desired output at t time
θ_i : are the estimated parameters
$\varepsilon(t)$: the mistake of $d(t)$ approximation
P_i: fixed functions of $x(t)$ (Regressor)
$P_i(t) = P_i(x(t))$

The function calculated by RBF network is the same as calculated by the formula (3), the analogy is the following form:

d(t) is the desired output of the network for the tth example, $P_i(t)$ is the activation of the i^{th} hidden cell for the tth example. The constant w_0 (bias) can be gotten by the corresponding definition:

$$P_i(t) = 1 \qquad (4)$$

Thereafter, we are going to consider the following notations:
Ne: The number of examples of the training basis
M: The initial number of centres
d: The vector of desired outputs: $d = [d(1)\ldots \ldots d(Ne)]^T$
P: The matrix of hidden layer activation: $P = [P_1 \ldots\ldots P_M]$
P_i: The vector of exit of the ith hidden cell: $P_i = [P_i(1)\ldots. P_i(Ne)\ ]^T$
θ : The weight vector of the output layer: $\theta = [\theta_1 .. \theta_M]$
E: The error vector between the calculated output and desired output: $E = [\varepsilon(1) .. \varepsilon(Ne)]^T$
With the above definite notations, the equation (3) can be written:

$$d = (P\ \theta + E) \qquad (5)$$

The resolution of the system equation (5) is a trivial problem. The vector of solutions θ can be defined by the least square method.

Dynamic OLS Variant: The OLS with its classic version can not adapt our ATRBF network, therefore the original idea of the Dynamic OLS method resides at each iteration, the creation of a hidden centre block and not only one centre, as the size of every block is expressed like suit:
If we consider that the input vector is composed of n characteristic on a temporal input window of Nfe length and if we take the value of input time delay equal to Nde, such as: Nfe ≥Nde.Then the number of neurons composing every block is equal to: Nfe-Nde+1.
The size of every hidden centre is equal to: n x Nde, (Fig. 4).
With this representation, we can follow the following steps: by leaving the eq. 5:

$$d = (P\ \theta + E)$$

The orthogonalisation of the columns Pi can be gotten by the decomposition of the P matrix in two matrices W and A as:

$$P = W\ A \qquad (6)$$

Where W: of size Ne x M, is the orthogonal image of the P matrix.

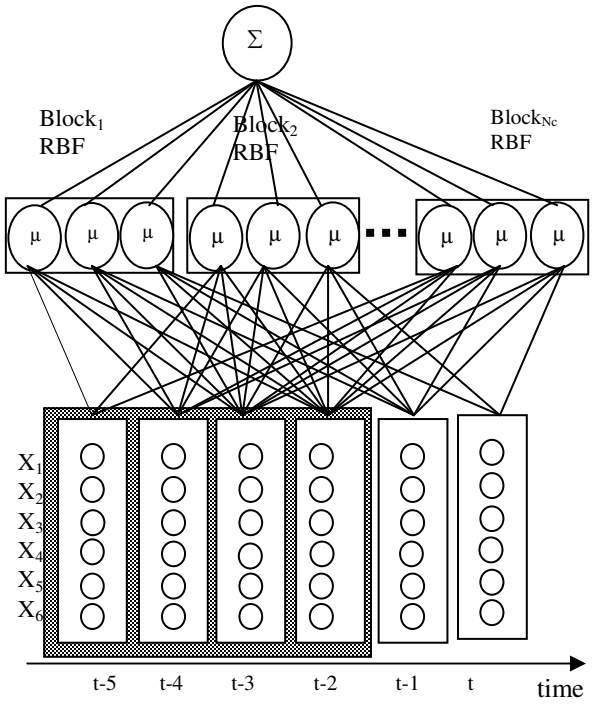

Fig. 4: Parameters of This ATRBF Network are: n number of Features Equal to Dim (X)=6. Nde: Time delay Equal to 4, Nfe: Size of the Input Window Equal to 6, Nnc: Number of Hidden Neurons by Block Equal to, Nbc: Represents the Number of Centres in Hidden Block, μ: Represents a Centre of Size Equal to Nde x n

A: of size M x M, is a superior triangular matrix containing orthogonal coefficients.

The A matrix is defined in Table 1.

Table 1: Representation of A Matrix Containing the Orthogonal Coefficients

1	$\alpha_{1,2}$				α_{1M}
0	1				$\alpha_{2,M}$
0	0	1			
				$\alpha_{M-2,M-1}$	$\alpha_{M-2,M}$
0			0	1	$\alpha_{M-1,M}$
0			0	0	1

The space generated by the vectors P_i is the same space generated by the vector W_i and the system of equation (6) can be written as suit:

$$d = W*G + E \quad (7)$$

Where G=A. θ is the search solution.
Now, Let's note:

$$H = W^T. W. E \quad (8)$$

Since the columns of the matrix W are orthogonal one by one, H is a diagonal matrix with h_i elements as:

$$hi = w_i^t.w_i = \sum_{i=1}^{Ne} w_i(t) .w_i(t) , 1 \le i \le M \quad (9)$$

This property which gives the Dynamic OLS method very interesting, and this is for following reason : the orthogonal solution G is calculated by:

$$G = H^{-1} . W^T d \quad (10)$$

Which can be written by:

$$Gi = w_i^T d / (w_i^T .w_i) , 1 \le i \le M \quad (11)$$

It means that the G_i elements of the orthogonal solution G depend only on w_i column, in other mean by the orthogonal image for each calculated outputs of each centre. This part defines the quotient of the reduction of the approximation mistake introduced by every vector w_i:

$$[err]_i = G_i^2 . w_i^T .w_i /(d^T. d) ;1 \le i \le M \quad (12)$$

This equation is used for construction iterative of the ATRBF network as criteria of selection. Hence of an initial whole of M centers, the network is constructed to every iteration, by adding the center that possesses the $[\text{maximal error}]_i$ value, and once we take the corresponding Gi.
At each iteration we calculate the elements of A and W by:

$$\alpha_{jk}^i = W_j^t * p_i / W_j^t * W_j \quad (13)$$

$$W_k^i = p_i - \sum^{k-1} \alpha_{jk}^i * W_j \quad (14)$$

The criterion of iteration stop here is not based merely on the Akaike criterion:

$$1 - \sum_{i=1}^{i=M} err_i \le \varepsilon \quad (15)$$

But also with the following criterion :

$$\text{Modulo M on } Nnc=0, \text{ where } M \neq 0 \quad (16)$$

In end of iterations, we calculate the synaptic weights according to the system:

$$G = A*\theta \quad (17)$$

The developed approach has been achieved on a subset of the TIMIT data base [8, 9] organized of 6 vowels, 6 fricatives and 6 plosives. For our survey we reduced the

space of study for the case of plosives (Table 2). Signals have been sampled to 16 KHZS with an analysis cepstral under the Mel ladder, takes all 20 ms in Hamming windows of 25 ms giving each 12 MFCCS coefficients and the corresponding residual energy.

Table 2: Subset of TIMIT Base Containing the /b/, /d/, /g /

	Train examples	Test examples
/b/	399	182
/d/	1371	526
/g/	1337	546

This work was achieved on a Pentium 4 microcomputer 1.7 GHz with 256 Mo of RAM, developed by the C++ builder and Matlab 6.5 programming language. Concerning the training data basis, it has been quantified with a Kohonen Self Organising Map of size 15x15, generating a basis of 225 phoneme examples, therefore the size of training basis is about 675 examples, every example is normalized on an input length window equal to 4. The size of the test basis is 450 example at rate of 150 examples by phoneme.

Parameters of our ATRBF are: Ne=675, Nfe=4 , Nde can take the values between 1 and 4, Nnc=Nfe-Nde+1, Nbc varies according to the phoneme basis training and the wanted precision, the used kernel is gaussian with receiving field equal to 1, the threshold error is fixed to 0.04 and finally the data are normalized by center-reduce method.

Effect of Changes on Input Time Delay: The global rate accuracy is about 98% in learning and 83% in test, the Fig. 5 shows us the influence of the time delay change on the performance of the network.

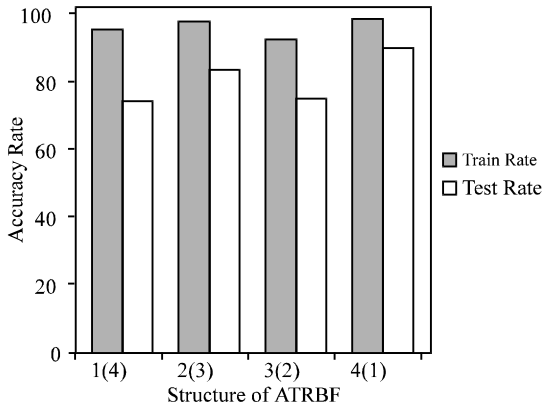

Fig. 5: Performance According to Size of Windows x(y), xNde, yNde

If the number of the time delay is narrow the performance degrades seen that it has less space time to browse all characteristic.

On the other hand if the time delay is large, there is a risk that the system becomes not shift invariant over time. The best case is choosing the time delay in the median

Comparison with TRBF, SVM, TDNN: A comparison was made between different temporal approach in accuracy rate (Fig. 6).

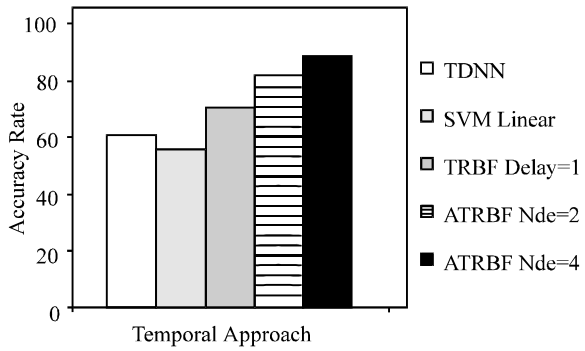

Fig. 6: Comparison between the Different Temporal Approach in Accuracy Rate

The phonemes /b/, /d/, /g/ represents an obstacle for the SVM approach [10], yet it has some best rates in the other phonemes, it is due to difficulty towards consonant detection, it comes back to the overlap in the training data basis [10]. Concerning the TRBF its problem is in the shift invariant in time as leaving of the time delay equals to 1 [11]. For the TDNN, results gotten on basis containing /b/ /d/ /dh/ /g / of TIMIT and implemented on a Digital WorkStation 433 MHz with three day of count in spite of data basis containing 3793 examples, the gotten rate was not competitor [9], what it has taken to hybrid with the HMM approach [9]. Finally the model proposed in this study gave good results, it comes back has hybridisation between advantages of the TDNN approach and networks RBF [1].

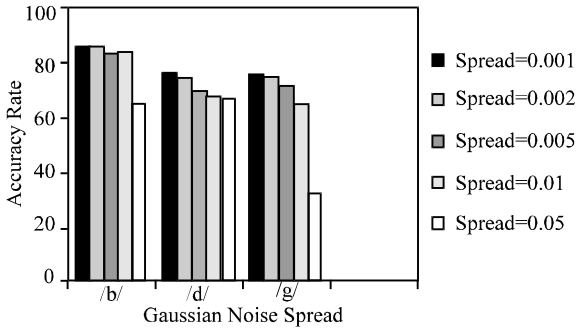

Fig. 7: Accuracy Rate Characterising the Tolerance of the Approach Towards a Guassian Noise of Different Spread σ

Noise Effect: While noticing the Fig. 7, we can deduct that the ATRBF among so many other methods [12], can adjust well with a word dived in a noised middle, by adding to the test data basis a gaussian noise of spread belongs to 0.001 until 0.05.

We see that the classifier resists good until a threshold spread σ equal to 0.005 where the test rate overcome 70% but from σ passing the value of 0.01 the rate degrades, it can even notice itself at the human being, we doesn't sometimes manage to distinguish passages of words some if we are in a very noisy environment.

DISCUSSION

In this study we have presented a new approach based on the adaptive temporal radial basis function, applied to speech recognition. The main advantage with regard to other neural architectures, it is well the time won in the training, in addition we have few parameters to adjust. The ATRBF combines advantages of the TDNN that are shift invariants in the time and their capacity of temporal feature recognition and advantages of the RBF in their speed.

It is shown that our ATRBF has a good rate of training and test. We hope that this work can be generalized and tested in different domains like the pattern recognition and more especially in applications covering the spatio-temporal basis, as the mobile robotics, detection of targets, economic fluctuations etc.

REFERENCES

1. Berthold, M.R., 1994. A Time Delay Radial Basis Function for Phoneme Recognition. Proc. Int. Conf. on Neural Network, Orlando, USA.
2. Benyettou, A., 1995. Acoustic Phonetic Recognition in the Arabex System. Int. Work Shop on Robot and Human Communication, ATIP95.44, Japan.
3. Zheng , G. and S. Billings, 1996. Radial Basis Function Network Using Mutual Information and the Orthogonal Least Square Algorithms. Neural Network, 9: 619- 637.
4. Karyianis, N. B., 1999. Reformulated Radial Basis Function Neural Network Trained By Gradient Descent. IEEE Transaction on Neural Network , 10: 657- 669.
5. Haykin, S., 1999. Neural Network A Comprehensive Foundation. Prentice Hall Upper Saddle River, New Jersey, USA. pp: 256-265.
6. Day, S. P. and M. R. Davenport, 1993. Continuous-time Temporal Back Propagation with Adaptable Time Delay. IEEE Transaction on Neural Network, 4: 348-354.
7. Xu, Z. B., H. Qiao, J. Peng and B. Zheng, 2004. A Comparative Study of Two Modeling Approachs in Neural Network. Neural Network, 17: 73-85.
8. Yak, D., Q. Lin, C. Che., L. Jin and J. Flanagan, 1995. Environment-Independent Continuous Speech Recognition. IEEE Automatic Speech Recognition Workshops, Utah, 151-152.
9. Keng, T. and Y. Colin, 2000. Speaker Adaptive Phoneme Recognition Using Time Delay Neural Networks. Thesis, National University of Singapore.
10. Juneja, A. and E. C. Wilson, 2002. Segmentation of Continuous Speech Using Acoustic Phonetic Parameters and Statistical Learning. In Proceeding of the ICNIP.
11. Mesbahi, L. and A. Benyettou, 2003. A New Contribution Towards a Temporal Radial Basis Function Applied To Mackey-Glass Time Series. 3rd ICNNAI , Minsk, Belarus, pp: 72-78.
12. Barbier, L. and G. Chollet, 1991. Robust Speech Parameters for Word Recognition in Noise Using Neural Network. IEEE International Conference on Acoustic Speech and Signal Processing, 1: 145-148.