A GAMMA NETWORK APPROACH TO AUTOMATIC SPEECH RECOGNITION

Jianxiong WU† Chorkin CHAN‡ Pengfei SHI†

†Institute of Image Processing and Pattern Recognition,
Shanghai Jiaotong University, P.R.China.

‡Department of Computer Science,
University of Hong Kong, Hong Kong.

ABSTRACT - Gamma neurons [1] are employed in this paper to construct a feed-forward multi-layer network to perform automatic speech recognition. Gamma network can approximate complex time-dependent connection weights with simple network structure. Its structure parameters can be learned in the training phase to determine an optimal structure for dynamics processing to fit the particular application task. An error back-propagation like learning algorithm is derived and experimental results of speaker-independent phoneme recognition are discussed.

1. INTRODUCTION

One of the main problems in applying neural network models to automatic speech recognition is how to process time-varying information in speech signals in an effective way. A possible solution to this problem is to introduce time-delay mechanism for neuron inputs. In this way, the neuron output depends not only on the current input but also on the time-varying feature of the input signal in the past. In general, the system equation for this kind of time-delay neurons can be formulated by the following convolution model:

$$y(t) = \mathcal{F}\left(\vec{w}_0'\vec{x}(t) + \int_0^t \vec{w}'(t-s)\vec{x}(s)ds + \eta\right) \tag{1}$$

Here $y(t)$ is the neuron output, $\vec{x}(t)$ is a vector of inputs, $\vec{w}_0$ is a vector of connection weights for the current input, $\vec{w}(t)$ is a vector of time-dependent connection weights for input signals in the past, η is the threshold value, $\mathcal{F}(\cdot)$ is a monotonically increasing non-negative function and the superscript $'$ denotes vector transposition.

Due to the complexity of the general convolution model, $\vec{w}(t)$ must be simplified in order to construct a network in practice. A common technique is to decompose $\vec{w}(t)$ into a weighted sum of some basis functions, that is

$$\vec{w}(t) = \sum_{\tau=1}^{T} \vec{w}_\tau g_\tau(t) \tag{2}$$

Several research have been carried out along this direction. Waibel *et al* has proposed time-delayed neural network (TDNN) [2] by employing impulse functions as the basis functions:

$$g_\tau(t) = \delta(t - d_\tau) \tag{3}$$

In concentration-in-time neural network (CITN network) of Tank *et al* [3], the integrands of the Γ functions are proposed to serve as the basis functions, i.e.,

$$g_\tau(t) = \left(\frac{t}{d_\tau}\right)^\tau exp\left(\tau(1 - \frac{t}{d_\tau})\right) \tag{4}$$

Gauss functions are chosen as the bases in the TEMP2 neural network proposed by Bodenhausen *et al* [4]

$$g_\tau(t) = \frac{1}{\sqrt{2\pi}\sigma_\tau} exp\left(-\frac{(t-d_\tau)^2}{2{\sigma_\tau}^2}\right) \tag{5}$$

Recently De Vries *et al* [1] suggests Gamma neuron to process temporal patterns which uses Gamma functions as the bases in decomposing $\vec{w}(t)$,

$$g_\tau(t) = \frac{\mu^\tau}{(\tau-1)!} t^{\tau-1} exp(-\mu t) \tag{6}$$

TDNN is quite successful in recognizing English phonemes [2]. However, it is obvious from Eq. (2) and (3) that one cannot achieve good approximation by employing impulse functions as basis functions. The TDNN structure specifies a fixed delay range to process the dynamic feature of the input signal. However, there is no theoretical ground to determine the maximum delay T. The strength of a CITN network is limited by its simple network structure. Only the input layer of a CITN network employs time-delay neurons, and there is no hidden layer in the network. Besides, no comprehensive learning procedure is proposed for a CITN network. The principal advantage of the TEMP2 network is that the structure parameters of the network (i.e., delay time d_τ and delay window width σ_τ in the Gauss function) are adaptive and can be learned from the training data. However, the relatively heavy computation requirement in a TEMP2 network usually results in the fact that only a few basis functions are employed in Eq.(2) which certainly degrades the approximation capability of the network. Beside, there is no easy implementation of the TEMP2 network using delay operators.

On the other hand, a Gamma neuron can take advantage of the nice properties of Gamma functions to realize the required computation in a very effective way. Moreover it is shown that any time-dependent function $\vec{w}(t)$ can be well approximated for a sufficiently large T using Eq. (2) and (6) if $\vec{w}(t)$ exponentially decays to zero as t approximates to infinity [5]. In addition, static neuron model, TDNN and CITN can all be seen as special cases of a Gamma neuron [1]. So Gamma neurons are very attractive as elements of a neural network which is capable of processing time-varying information of complex input patterns such as speech signals.

A multi-layer network structure with Gamma neurons in both its input layer and hidden layers (called Gamma network hereafter) are proposed and tested in an automatic speech recognition experiment in this paper. Section 2 summaries the implementation of a Gamma neuron originally proposed by De Vries *et al* [1]. Section 3 discusses the training method for a multi-layer Gamma network. An error back-propagation algorithm is proposed to learn the connection weights of a Gamma network $\vec{w}_0, \vec{w}_1, \ldots, \vec{w}_T$ as well as the network structure parameter μ's. With such an algorithm, a Gamma network can automatically determine its delay structure and adjust the dynamics processing capability for each Gamma neuron in the network from a given set of training data to achieve an optimal performance. Finally, experimental results of recognizing speaker-independent phonemes (/b/,/d/,/g/) by means of a Gamma network are presented in Section 4.

2. THE STRUCTURE OF MULTI-LAYER GAMMA NETWORK

Defining Gamma state variables $\vec{x}_\tau(t)$ as

$$\vec{x}_0(t) = \vec{x}(t) \tag{7}$$

$$\vec{x}_\tau(t) = \int_0^t g_\tau(t-s)\vec{x}(s)ds \qquad \tau \geq 1 \tag{8}$$

and substituting Eq. (2), (7), (8) into (1), one gets the simplified system equation for a Gamma neuron:

$$y(t) = \mathcal{F}\left(\sum_{\tau=0}^{T} \vec{w}_\tau^t \vec{x}_\tau(t) + \eta\right) \tag{9}$$

Using the derivative property of Gamma functions in Eq. (6), it is easy to show that the

derivatives of the Gamma state variables $\vec{x}_\tau(t)$ should follow:

$$\frac{\partial \vec{x}_\tau(t)}{\partial t} = -\mu \vec{x}_\tau(t) + \mu \vec{x}_{\tau-1}(t) \qquad \tau \geq 1 \tag{10}$$

Under the assumption that the sampling frequency, f_s (corresponding to the frame rate of a speech recognition system), of the acoustic features, $\vec{x}(t)$, of the input signal is higher than the Nyquist frequency, the discrete system equation of a Gamma neuron is

$$\vec{x}_0(n) = \vec{x}(n) \tag{11}$$

$$\vec{x}_\tau(n) = (1-\mu)\vec{x}_\tau(n-1) + \mu \vec{x}_{\tau-1}(n-1) \qquad \tau \geq 1 \tag{12}$$

$$y(n) = \mathcal{F}\left(\sum_{\tau=0}^{T} \vec{w}_\tau \vec{x}_\tau(n) + \eta\right) \tag{13}$$

The above operation can be effectively realized by a Gamma node as illustrated in Fig. 1 (D denotes a delay operator in the figure).

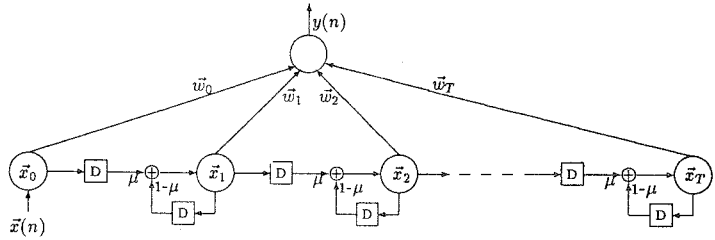

Figure.1 The Structure of a Gamma neuron

The recursive memory structure in Eq. (12) is stable for $0 \leq \mu \leq 2$ and the interesting memory structure is obtained for only $0 \leq \mu \leq 1$ [1]. For $\mu = 0$, a Gamma neuron is reduced to a normal static neuron. A Gamma neuron works as a discrete time CITN if $0 < \mu < 1$. It becomes a TDNN when μ is set to 1. The strength of the Gamma neuron is the structure parameter μ can be learned in the training phase, so, the optimal temporal structure is determined to fit the problem to be solved.

Multi-layer Gamma network is purely a feed-forward network. Every processing units in its every layer is a Gamma neuron. The output of each Gamma neuron is the input of all the Gamma neurons in its next succeeding layer. Assuming K layers in a Gamma network (including the input layer) with M^k Gamma neurons in the k^{th} layer, one can represent the system equation of the Gamma network as

$$y_j^k(n) = \mathcal{F}\left(\sum_{i=1}^{M^{k-1}} \sum_{\tau=0}^{T} w_{ji\tau}^k x_{ji\tau}^k(n) + \eta_j^k\right) \qquad k = 1, 2, \ldots, K \tag{14}$$

$$x_{ji\tau}^k(n) = (1-\mu_{ji}^k) x_{ji\tau}^k(n-1) + \mu_{ji}^k x_{ji,\tau-1}^k(n-1) \tag{15}$$

$$x_{ji0}^k(n) = y_i^{k-1}(n) \qquad i = 1, 2, \ldots, M^{k-1} \tag{16}$$

Here $y_j^k(n)$ and $x_{ji\tau}^k(n)$ are the output value and the value of the i^{th} component of the τ^{th} Gamma state variable, respectively, of the j^{th} Gamma neuron in the k^{th} layer at time n. $w_{ji\tau}^k$

is the connection weight between the output and the i^{th} component of the τ^{th} Gamma state variable of the j^{th} Gamma neuron in the k^{th} layer. μ_{ji}^k is the structure parameter for the i^{th} input component of j^{th} Gamma neuron in the k^{th} layer, η_j^k is the threshold value of j^{th} Gamma neuron in the k^{th} layer. $y_j^0(n) = x_j(n)$ is the input value in the j^{th} input node of the network at time n.

In Eq. (14), all Gamma neurons in the network have the same number of Gamma state variables, i.e., the parameter T is common to all the Gamma neurons. Since every Gamma neuron has its individual structure parameter μ_{ji}^k for every input component of it, the capability of processing time-varying information for each input component in every Gamma neuron is different. In this manner, the network is highly powerful to process complex dynamic information. Moreover, since the structure parameters μ_{ji}^k's are learned from the training data, the network structure is adaptive to the particular application task.

3. TRAINING A MULTI-LAYER GAMMA NETWORK

It is quite straightforward to generalize the error back-propagation algorithm [6] of Rumelhart *et al* to train a Gamma network. Let $\vec{x}(0), \vec{x}(1), \ldots, \vec{x}(N)$ be a training sequence and $\vec{d}(0), \vec{d}(1), \ldots, \vec{d}(N)$ be the corresponding desired network output ($\vec{x}(n)$ and $\vec{d}(n)$ are M^0-dimensional and M^K-dimensional vectors, respectively). The network parameter set $\Theta = \{w_{ji\tau}^k, \mu_{ji}^k \eta_j^k\}$ can be obtained through the following stochastic optimization process:

$$\Theta^* = \min_{\Theta} E = \min_{\Theta} \frac{1}{2} \sum_{n=0}^{N} \sum_{j=1}^{M^K} \left(y_j^K(n) - d_j(n)\right)^2 \tag{17}$$

$\sigma_j^k(n)$ and $\delta_j^k(n)$ are defined as the net input and error variable for the j^{th} Gamma neuron in the k^{th} layer at time n, i.e.,

$$\sigma_j^k(n) = \sum_{i=1}^{M^{k-1}} \sum_{\tau=0}^{T} w_{ji\tau}^i x_{ji\tau}^k(n) + \eta_j^k \tag{18}$$

$$\delta_j^k(n) = \frac{\partial E(n)}{\partial \sigma_j^k(n)} = \frac{\partial E(n)}{\partial y_j^k(n)} \cdot \frac{\partial y_j^k(n)}{\partial \sigma_j^k(n)} = \mathcal{F}'\left(\sigma_j^k(n)\right) \cdot \frac{\partial E(n)}{\partial y_j^k(n)} \tag{19}$$

where

$$E(n) = \frac{1}{2} \sum_{j=1}^{M^K} (y_j^K(n) - d_j(n))^2 \tag{20}$$

is the network output error at time n. The following recursive formula for computing $\delta_j^k(n)$ can be derived if a sigmoid function is employed for $\mathcal{F}(\cdot)$:

$$\delta_j^K(n) = y_j^K(n) \cdot (1 - y_j^K(n)) \cdot (y_j^K(n) - d_j(n)) \tag{21}$$

$$\delta_i^k(n) = \mathcal{F}'\left(\sigma_i^k(n)\right) \sum_{j=1}^{M^{k+1}} \delta_j^{k+1}(n) \cdot \sum_{\tau=0}^{T} w_{ji\tau}^{k+1} I_{ji\tau}^{k+1}(n) \qquad 0 < k < K \tag{22}$$

where $I_{ji\tau}^{k+1}(n) = \dfrac{\partial x_{ji\tau}^{k+1}(n)}{\partial y_i^k(n)}$ can be recursively computed from Eq.(7) and (8) by

$$I_{ji0}^k(n) = 1 \tag{23}$$

$$I_{ji\tau}^k(n) = I_{ji,\tau-1}^k(n) - J_{ji,\tau-1}^k(n) \tag{24}$$

$$J_{ji0}^k(n) = exp(-\mu_{ji}^k n f_s) \tag{25}$$

$$J_{ji\tau}^k(n) = \frac{\mu_{ji}^k}{\tau} \cdot n f_s \cdot J_{ji,\tau-1}(n) \tag{26}$$

consonant	No. of training tokens	No. of testing tokens
/b/	1207	534
/d/	1219	390
/g/	720	285
total	3146	1209

Table.1 Number of training and testing tokens

Using the following derivatives, the network parameters can be determined by the well-known steepest descent searching method or any other optimization procedure:

$$\frac{\partial E}{\partial \eta_j^k} = \sum_{n=0}^{N} \delta_j^k(n) \tag{27}$$

$$\frac{\partial E}{\partial w_{ji\tau}^k} = \sum_{n=0}^{N} \delta_j^k(n) x_{ji\tau}^k(n) \tag{28}$$

$$\frac{\partial E}{\partial \mu_{ji}^k} = \frac{1}{\mu_{ji}^k} \sum_{n=0}^{N} \delta_j^k(n) \cdot \sum_{\tau=1}^{T} \tau \cdot w_{ji\tau}^k \cdot \left(x_{ji\tau}^k(n) - x_{ji,\tau+1}^k(n)\right) \tag{29}$$

4. PHONEME RECOGNITION EXPERIMENTS

English phoneme recognition experiments with a subset of the DARPA TIMIT speech database [7] are performed to evaluate the performance of a Gamma network trained by the proposed learning algorithm. Three consonants (/b/,/d/,/g/) in different phonetic contexts are extracted for recognition from continuously spoken sentences according to the phoneme boundaries provided by the database. Since these three consonants are very short in duration and their characteristics are also reflected in the surrounding phonemes, a $16ms$ segment of signals is included both before the onset and after the offset of these consonants.

The speech material was spoken by 630 speakers, each spoke five randomly selected sentences from the MIT acoustic-phonetic corpus (there are in total 450 sentences in the corpus). 3146 tokens of three consonants from 420 speakers are employed as the training set and 1209 tokens from the rest of speakers are served as the testing set. Table.1 shows the number of training and testing tokens for each consonant.

The speech signal is sampled at 16KHz. After Hamming windowing, a 256-point discrete Fourier transformation is applied every $4ms$. The acoustic features employed in the experiment are 16 mel-scaled cepstrum coefficients [8] computed from the logarithm output of a bank of 20 critical bandpass filters [9].

A Gamma network with one hidden layer is employed in the experiment. The network consists of 16 Gamma units in the input layer, 20 Gamma units in the hidden layer and 3 units in the output layer. Each unit in the input layer receives a mel-scaled cepstrum coefficient each $4ms$. Every output unit corresponds to one consonant class. Gamma units in the input layer have 4 Gamma state variables and those in the hidden layer have 6 Gamma state variables. All network parameters are initialized with random values. After the training process terminated, a correct recognition rate of 82.4% is obtained for the testing consonants.

For the purpose of comparison, a TDNN with the similar architecture, i.e., 16, 20 and 3 units in the input, hidden and output layers, 4 and 6 delay operators for units in the input and hidden layers respectively, is constructed and trained by the standard error back-propagation algorithm. When tested with consonants in the testing set, a correct recognition rate of 78.6% is obtained.

	Gamma network			TDNN		
	/b/	/d/	/g/	/b/	/d/	/g/
/b/	466	37	31	441	24	69
/d/	44	303	43	45	266	79
/g/	26	32	227	28	14	243

Table.2 Confusion matrices for a Gamma network and a TDNN

Table.2 shows the confusion matrices for both the Gamma network and the TDNN.

5. CONCLUSIONS

This paper presents a Gamma neural network architecture for recognition of patterns with temporal structures. A learning algorithm based on the principle of error back-propagation is derived. Speaker-independent English phoneme recognition experiments are performed to verify the proposed network architecture and the associated learning algorithm. An improved recognition performance is observed when a Gamma network is compared with a TDNN.

References

[1] De Vries, B. & Principe, J.C. (1991) *A theory of neural networks with time delays,* In R.P. Lippmann, J.E. Moody & D.S. Touretzky, editors, Advances in Neural Information Processing Systems: Volume 3, 162–168, (Morgan Kaufmann Pub.).

[2] Waibel, A., Hanazawa, T., Hinton, G., Shikano, K. & Lang, K.J. (1989) *Phoneme recognition using time-delay neural networks,* IEEE Trans. on Acoustics, Speech and Signal Processing, ASSP-37(3):328–339.

[3] Tank, D.W. & J.J. Hopfield, J.J. (1987) *Concentrating information in time: analog neural networks with applications to speech recognition problems,* Proceedings of IEEE First International Conference on Neural Networks.

[4] Bodenhausen, U. & Waibel, A. (1991) *Learning the architecture of neural networks for speech recognition,* Proceedings of 1991 IEEE International Conference on Acoustics, Speech and Signal Processing, 117–120.

[5] Cohen *et al.* (1979) *Stable oscillations in single species growth models with hereditary effects,* Mathematical Biosciences, 44:255–268.

[6] Rumelhart, D.E., Hinton, G.E. & Williams, R.J. (1986) *Learning internal representation by error propagation,* in D.E. Rumelhart and J.L. McClelland, editors, Parallel and Distributed Processing, Volume I: Foundations, (MIT Press).

[7] Lamel, L.F, Kassel, R.H. & Seneff, S. (1986) *Speech database development: design and analysis of the acoustic-phonetic corpus,* Proceedings of the DARPA Speech Recognition Workshop, 100–110.

[8] O'Shaughnessy, D. (1987) *Speech Communication: human and machine,* (Addison-Wesley Publishers).

[9] Zwicker, E. (1961) *The subdivision of the audible frequency range into critical bands,* Journal of the Acoustics Society of America, 33:248.