Proceedings of the VI Brazilian Conference on Neural Networks - VI Congresso Brasileiro de Redes Neurais
pp. 97–102, June 2–5, 2003 - Centro Universitario da FEI, Sao Paulo - SP - Brazil

Incorporação de Recorrência em Estruturas Neurofuzzy

Ivette Luna[1], Rosangela Ballini[2], Fernando Gomide[1]
[1]Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica e de Computação
13081-970 Campinas - SP - Brasil
[2]Universidade Estadual de Campinas
Instituto de Economia
13083-970 Campinas - SP - Brasil
E-mails: {iluna, gomide}@dca.fee.unicamp.br, ballini@eco.unicamp.br

Abstract

Hybrid neurofuzzy networks are addressed in this paper. The network models have two basic structures, a fuzzy neural inference system and a neural network. The fuzzy system contains fuzzy neurons modeled through logic and *and* or *operations processed via t-norms and s-norms, respectively. The neural network has nonlinear elements placed in series with the previous logical elements. The neural fuzzy inference system encodes a set of if-then rules and its recurrent multilayered structure performs fuzzy inference. Learning is based on an associative reinforcement learning to update second layer weights, and gradient search for output layer weights. The recurrent fuzzy neural network is particularly suitable to model nonlinear dynamic processes. Computational experiments with modeling of an unknown nonlinear process suggest that the hybrid fuzzy neural models are simpler, learning is faster, and that approximation errors are lower when compared with its counterparts.*

1. Introdução

Nestes últimos anos, sistemas neurofuzzy tornaram-se uma alternativa atrativa para aplicações em áreas distintas da ciência e engenharia, devido a sua capacidade de tratar imprecisão e incerteza presentes na informação disponível e na dinâmica dos próprios sistemas.

Sistemas neurofuzzy combinam dois grandes paradigmas, redes neurais artificiais e teoria de conjuntos fuzzy, respectivamente. Em particular, sistemas neurofuzzy estáticos se mostram promissores em identificação de sistemas, controle, previsão de séries temporais, classificação de padrões, etc. Contudo, o uso de redes neurofuzzy estáticas ainda é restrito, devido à sua estrutura não recorrente e à carência de algoritmos eficientes de aprendizagem para estruturas neurais com realimentação. Embora análise e síntese de redes neurais recorrentes sejam mais complexas, estes modelos têm proporcionado desempenho superior em diversas aplicações. Devido à sua estrutura, redes recorrentes são mais difíceis de serem analisadas, o processo de treinamento é mais trabalhoso, e os algoritmos de aprendizagem mais complexos e lentos [1]. A incorporação de recorrência em sistemas neurofuzzy faz com que estes sistemas sejam capazes de tratar problemas que envolvem relações temporais, já que a presença de realimentação permite a criação de representações internas e os mecanismos de memória necessários para processar e armazenar este tipo de informação.

Este trabalho analisa e compara o efeito da recorrência em uma classe de redes neurofuzzy, partindo-se de uma estrutura neural estática como base para duas estruturas multicamadas híbridas recorrentes. Dois tipos de recorrência são analisados: recorrência interna local e recorrência interna global. Estas três estruturas tem duas partes em comum. A primeira parte constitui um sistema de inferência fuzzy para o processamento dos sinais de entrada. Além de neurônios que fornecem o grau de pertinência das entradas à conjuntos nebulosos que granularizam os respectivos universos na primeira camada, neurônios lógicos do tipo *and* [2] compõem a segunda camada. As relações temporais são induzidas por uma realimentação local e global nos neurônios lógicos desta camada intermediária, fornecendo assim, elementos de memória necessários para a representação temporal. A segunda parte é constituída por neurônios não lineares clássicos organizados em uma camada que atua como uma função não linear de agregação [3].

Os pesos da camada de saída tanto da rede neurofuzzy estática, como das estruturas recorrentes analisadas, são ajustados utilizando o método do gradiente. Os pesos da camada intermediária das três estruturas são ajustados utilizando um método de treinamento por reforço associativo. Neste caso, os pesos da camada intermediária são ajustados usando um mecanismo de recompensa e punição inspirado em [4] e [5].

Neste trabalho, as estruturas neurofuzzy estática

(**RNF1**), neurofuzzy recorrente local (**RNFR1**) e neurofuzzy recorrente global (**RNFR2**) são analisadas comparando o desempenho destas estruturas com as correspondentes estruturas de redes neurais multicamada (**MLP**) [6], recorrente (**RNR2**) [7], e fuzzy recorrente (**RNFR3**), [1]. A comparação é feita considerando um problema de identificação de sistema dinâmico não linear.

O artigo está organizado da seguinte forma. Após esta introdução, a próxima seção detalhada as estruturas híbridas a serem analisadas; a seção 3 apresenta os procedimentos utilizados para aprendizagem. Os resultados, análise e síntese das simulações realizadas são apresentados na seção 4. Finalmente, a seção 5 conclui o trabalho.

2. Estruturas híbridas

Esta seção detalha as estruturas de redes neurofuzzy estática e recorrentes de interesse neste trabalho. Estas estruturas são compostas por duas partes: um sistema de inferência fuzzy e uma rede neural clássica conforme sugere a Figura 1, no caso, uma rede com recorrência global. Redes com recorrência parcial ou local tem estrutura análoga, porém com realimentação de saída apenas. A rede estática também possui a mesma estrutura, a menos das conexões laterais e de realimentação de saída dos neurônios da segunda camada. Note que a recorrência ocorre nos neurônios da camada intermediária.

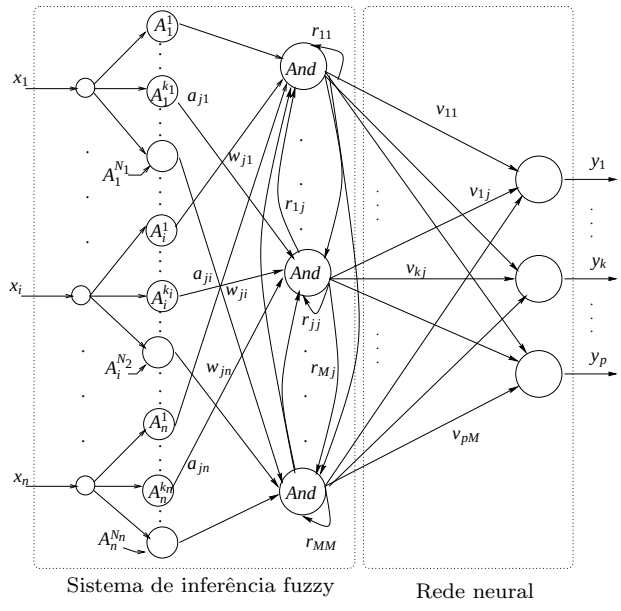

Figura 1: Rede neurofuzzy com recorrência global

Esta camada é constituída por neurônios lógicos do tipo *and* (figura 2), cujo processamento se dá através de operadores da classe das $t-$normas e $s-$normas [2]. Já a camada de saída é formada por neurônios não lineares clássicos. Seu propósito é realizar uma combinação das saídas dos neurônios da camada anterior via uma função de agregação não linear.

O sistema de inferência é composto pelas camadas de entrada e intermediária. A camada de entrada contém neurônios cujas funções de ativação são as funções de pertinência dos conjuntos fuzzy que formam a partição do espaço de entrada. Isto é, para cada dimensão $x_i(t)$ do vetor de entrada $n-$dimensional $\mathbf{x}(t)$ existem N_i conjuntos fuzzy $A_i^{k_i}, k_i = 1, \ldots, N_i$ cujas funções de pertinência são as correspondentes funções de ativação dos neurônios da camada de entrada. A variável t denota o tempo discretizado, isto é, $t = 1, 2, \ldots$ e será omitida no decorrer do artigo para simplificar a notação. Deste modo, os graus de pertinência associados aos padrões de entrada são $a_{ji} = \mu_{A_i^{k_i}}(x_i),\ i = 1, \ldots, n$ e $j = 1, \ldots, M$; onde M é o número de neurônios da camada intermediária.

A saída z_j de um neurônio da camada intermediária é definida por:

$$z_j = \mathop{\mathbf{T}}_{i=1}^{n+D} (w_{ji}\, \mathbf{s}\, a_{ji}) \tag{1}$$

Note que para $D = 0$ obtém-se as saídas z_j da rede **RNF1** enquanto que para $D = 1$, as saídas z_j são aquelas da rede **RNFR1**, fazendo $w_{jn+1} = r_{jj} = r_j$ e $a_{jn+1} = q^{-1} z_j$. Para $D = M$ obtém-se as saídas z_j da rede **RNFR2**. Neste caso $w_{jn+l} = r_{jl}$, e $a_{jn+l} = q^{-1} z_j$, $l = 1, \ldots, M$, onde q^{-1} é o operador de atraso.

É importante também verificar que cada uma das estruturas analisadas gera um conjunto de regras do tipo *se-então* $R = \{R_j, j = 1, \ldots, M\}$. Por exemplo, para o caso da rede **RNF1** estas tem a seguinte forma:

$$\begin{aligned} R_j \quad : \quad & \text{Se } (x_1 \text{ é } A_1^{k_1} \text{ com certeza } w_{j1}) \ldots \\ & \textit{and } (x_i \text{ é } A_i^{k_i} \text{ com certeza } w_{ji}) \ldots \\ & \textit{and } (x_n \text{ é } A_i^{k_n} \text{ com certeza } w_{jn}) \\ & \text{então, } z \text{ é } z_j. \end{aligned}$$

Isto significa que existe uma dualidade entre o sistema de inferência fuzzy neural e um sistema baseado em regras fuzzy. No caso das estruturas recorrentes, as regras R_j incorporam as suas próprias saídas através do operador de atraso q^{-1}, atuando estas como variáveis internas de um sistema de inferência fuzzy recorrente equivalente [8].

A segunda parte da estrutura é uma rede neural clássica, composta por neurônios não lineares e não recorrentes. A saída y_k corresponde à agregação dos valores de z_j e v_{kj}, $j = 1, \ldots, M$ e $k = 1, \ldots, p$, isto é:

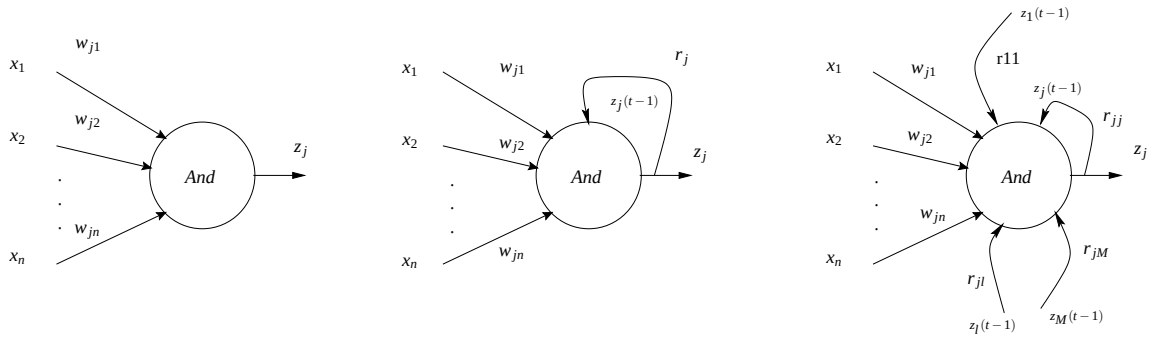

Figura 2: Neurônio lógico *And*. (a) estático, (b) recorrência interna local, (c) recorrência interna global.

$$y_k = \psi(u) = \psi\left(\sum_{j=1}^{M}(v_{kj}\cdot z_j)\right) \quad (2)$$

onde, $\psi : \Re^n \to [0,1]$ é uma função monotônicamente crescente. Neste trabalho considera-se a função sigmóide: $\psi(u) = 1/(1+\exp(-u))$.

A dinâmica da rede neurofuzzy pode ser resumida da seguinte maneira:

1. N_i é o número de conjuntos fuzzy que constitui a partição da $i-$ésima entrada;
2. A variável j indexa os neurônios *and*. Para as estruturas neurofuzzy analisadas, j é determinada utilizando a seguinte expressão:

$$j = f(K) = k_n + \sum_{i=2}^{n}(k_{(n-i+1)} - 1)\left(\prod_{\tau=1}^{i-1} N_{(n+1-\tau)}\right) \quad (3)$$

onde, $K = (k_1, \ldots, k_i, \ldots, k_n)$, sendo k_i o índice associado à regra ativada pelo componente x_i do padrão de entrada $\mathbf{x}$;

3. $a_{ji} = \mu_{A_i^{k_i}}(x_i)$ é o grau de pertinência de x_i no conjunto fuzzy $A_i^{k_i}$, sendo a_{ji} a entrada para o neurônio j da camada intermediária;
4. z_j é a $j-$ésima saída da camada intermediária definida por (1);
5. y_k é a $k-$ésima saída da rede conforme (2);
6. w_{ji} é o peso entre o $j-$ésimo neurônio *and* e o $i-$ésimo neurônio da camada de entrada;
7. v_{kj} é o peso entre a saída y_k da rede e o $j-$ésimo neurônio *and*;
8. No caso da rede **RNFR1** $r_j = r_{jj}$ é o peso da recorrência e no caso da rede **RNFR2** r_{jl} são os pesos das conexões laterais entre o $j-$ésimo neurônio *and* e o $l-$ésimo neurônio da segunda camada.

As arquiteturas apresentadas tem como vantagens a geração automática da topologia da rede durante a aprendizagem, flexibilidade quanto a utilização de diversas normas triangulares, e a possibilidade de extrair regras fuzzy diretamente da topologia [5] e [9].

3. Método de Aprendizagem

O método de aprendizagem é resumido a seguir:

1. Geração das funções de pertinência;
2. Inicialização dos pesos;
3. Até satisfazer a condição de parada:
   - 3.1 Apresentar um padrão $\mathbf{x}$ à rede, geralmente escolhido aleatoriamente;
   - 3.2 Efetuar a fuzzificação;
   - 3.3 Determinar os neurônios *and* ativos;
   - 3.4 Atualizar os pesos w_{ji}, r_{jl} e v_{kj};
   - 3.5 Testar a condição de parada (máximo erro permitido ou número máximo de iterações).

Dois conceitos diferentes são utilizados para ajustar os pesos: aprendizagem por reforço associativo, no caso dos pesos da camada intermediária, e o método do gradiente para os pesos da camada de saída.

Os passos do método de aprendizagem são detalhados nas próximas seções.

3.1 Geração das funções de pertinência

Assume-se que funções de pertinência são funções triangulares normais e complementares no intervalo $[x_{imin},\ x_{imax}]$, $i = 1, ..., n$. Neste caso existem duas possíveis partições do universo de entrada: partição uniformemente distribuída e partição não uniformemente distribuída, respectivamente.

Para gerar uma partição não uniformemente distribuída, técnicas de agrupamento são utilizadas

para determinar os centros c_{is}, $s = 1, \ldots, N_i$. Neste artigo adota-se um algoritmo de agrupamento baseado em uma rede neural auto-organizada LVQ (*Learning Vector Quantization*) proposto em [5].

Uma vez determinados os centros $c_{is}, i = 1, \ldots, n$ e $s = 1, \ldots, N_i$, onde N_i é o número de funções de pertinência para o componente de entrada x_i, os graus de pertinência $\mu_{A_i^s}(x_i)$ podem ser determinados para cada padrão de entrada $\mathbf{x}$.

3.2 Inicialização dos pesos

Os pesos da camada de saída v_{kj} são inicializados aleatoriamente com valores no intervalo $[-1, 1]$; os pesos da recorrência r_j para a rede **RNFR1** e r_{jl} para **RNFR2** assim como os pesos w_{ji}, foram inicializados com valores aleatórios entre $[0, 1]$, $i = 1, \ldots, n$, $j = 1, \ldots, M$ e $k = 1, \ldots, p$.

3.3 Determinação dos neurônios *and* ativos

Para cada padrão de entrada $\mathbf{x}$ existe no máximo dois conjuntos fuzzy com graus de pertinência diferentes de zero, denominados conjuntos ativos. Conjuntos ativos, por sua vez, definem os neurônios *and* ativos.

Dado um padrão de entrada $\mathbf{x} = [x_1, \ldots, x_i, \ldots, x_n]$, o número de neurônios *and* ativos são encontrados combinando os índices da função de pertinência para os quais os graus de pertinência são diferentes de zero.

Assim, de M neurônios *and*, no máximo 2^n estarão ativos para cada padrão apresentado à rede. Desta forma, o tempo de processamento da rede é independente do número de conjuntos fuzzy da partição do espaço de entrada [5], [10].

3.4 Fuzzificação

Nesta etapa, determinam-se os graus de pertinência do padrão de entrada aos conjuntos fuzzy ativos. Seja k_i^1 e k_i^2 os índices das funções de pertinência com graus de pertinência não nulos. Se $k_i^1 \neq k_i^2$, então $\mu_{k_i^2}(x_i) = 1 - \mu_{k_i^1}(x_i)$, devido à complementaridade das funções de pertinência.

3.5 Atualização dos pesos

O método do gradiente é utilizado para ajustar os pesos da rede neural clássica v_{kj}, enquanto que os pesos do sistema de inferência fuzzy e entre os neurônios da camada intermediária são ajustados utilizando o método de treinamento por reforço associativo. Uma vez avaliada as saídas de cada neurônio para um determinado padrão de entrada, utilizando as expressões (1) e (2), o objetivo do processo de treinamento supervisionado é minimizar o erro médio entre a saída atual da rede e a saída desejada, isto é, minimizar:

$$\varepsilon = \frac{1}{p} \sum_{k=1}^{p} (y_k - \hat{y}_k)^2 \tag{4}$$

onde, $\hat{y}_k$ é o valor da $k-$ésima saída no instante t e y_k é a $k-$ésima saída desejada para o correspondente padrão $\mathbf{x}$. Assim, se v_{kj} é um peso conectado às unidades de saída, então:

$$\Delta v_{kj} = \eta (y_k - \hat{y}_k)\ \psi'(u)\ z_j \tag{5}$$

onde $\psi'(u) = \psi(u)(1 - \psi(u))$ é a derivada da função de ativação da camada de saída avaliada em u, e η é a taxa de aprendizagem.

Os pesos da camada intermediária são atualizados utilizando aprendizado por reforço. Para tal é um sinal de reforço é necessário. Em [4], um sinal de reforço $\delta = 1 - \varepsilon$ é proposto. Valores grandes de δ correspondem a uma melhor aproximação entre a saída da rede e o sinal desejado. O mecanismo de treinamento aqui utilizado foi inicialmente proposto em [10]. O algoritmo combina os procedimentos de atualização dos pesos sugerido em [4] e [5]. Assim, os pesos das unidades intermediárias são atualizados da seguinte forma:

$$\Delta w_{ji} = \delta \alpha_1 [1 - w_{ji}] - (1 - \delta) \alpha_2 w_{ji} \tag{6}$$

$$\Delta r_{jl} = \delta \alpha_3 [1 - r_{jl}] - (1 - \delta) \alpha_4 r_{jl} \tag{7}$$

onde, $0 < \alpha_1 << \alpha_2 < 1$ e $0 < \alpha_3 << \alpha_4 < 1$ são as taxas de aprendizagem, $j = 1, \ldots, M$ e $i = 1, \ldots, n$.

4. Simulações e Comparações

Esta seção apresenta os resultados de simulação considerando a identificação de um sistema dinâmico não linear. O exemplo adotado foi sugerido em [6]. O sistema não-linear é descrito pela equação à diferenças de terceira ordem:

$$y(t+1) = f[y(t), y(t-1), y(t-2), u(t), u(t-1)] \tag{8}$$

onde $u(t)$ é a entrada e a função $f[\cdot]$, suposta desconhecida, tem a forma:

$$f[x_1, x_2, x_3, x_4, x_5] = \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_3^2 + x_2^2} \tag{9}$$

A rede **RNF1** utilizou as cinco variáveis $y(t)$, $y(t-1)$, $y(t-2)$, $u(t)$ e $u(t-1)$ como entradas, enquanto que as redes recorrentes **RNFR1** e **RNFR2** utilizaram somente $u(t)$ como entrada.

Durante a aprendizagem, considerou-se valores aleatórios uniformemente distribuídos no intervalo

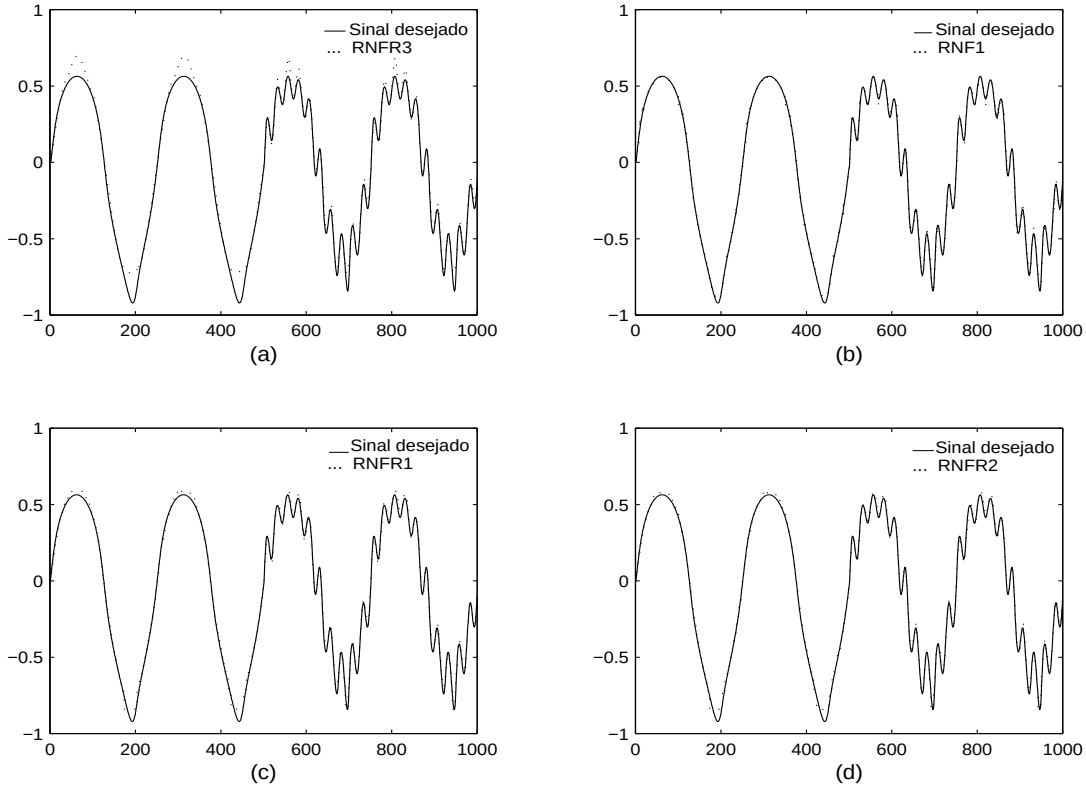

Figura 3: Exemplo de Identificação. (a) RNFR3, (b) RNF1, (c) RNFR1, (d) RNFR2.

$[-2,2]$ para a entrada $u(t)$; adotou-se o produto algébrico como $t-$norma e a soma probabilística como a $s-$norma.

As taxas de aprendizagem para o ajuste dos pesos foram $\eta = 0.25$, $\alpha_1 = 0.0001$, $\alpha_3 = 0.001$, $\alpha_2 = 0.001$ e $\alpha_4 = 0.01$. A Figura 3 mostra a saída do sistema real e as saídas fornecidas pelas redes **RNFR3**, **RNF1**, **RNFR1** e **RNFR2** para a seguinte entrada $u(t)$:

$$u(t) = \begin{cases} sen(2\pi t/250) & 0 < t \leq 500, \\ 0.8sen(2\pi t/250) + 0.2sen(2\pi t/25) & \text{t>500.} \end{cases}$$

A Tabela 1 caracteriza os parâmetros que definem a complexidade das estruturas avaliadas, assim como o número de parâmetros ativos por iteração. Nesta tabela, n é o número de entradas, p o número de saídas, $Np = N_i$ o número de partições para cada componente x_i do vetor de entrada $\mathbf{x}$ e Na_{max} é o número máximo de funções de pertinência ativas por iteração. Parâmetros ativos são aqueles correspondentes aos neurônios ativos.

A Tabela 2 mostra os erros quadráticos médios (EQM) de treinamento, de teste, e os tempos de processamento (T_{proc}, em segundos) para o treinamento de cada uma das redes consideradas neste trabalho. A análise das tabelas 1 e 2 sugere que, em termos de um compromisso entre complexidade e desempenho, as redes neurofuzzy recorrentes híbridas são tão ou mais eficientes que as redes correspondentes **RNFR3**, **RNR2** e **MLP**.

Por exemplo, a rede **RNFR3** [1] utiliza duas entradas, 30 conjuntos nebulosos em cada partição destas entradas, significando um número maior de parâmetros a ajustar por iteração. Além disso, seu EQM de teste (Tabela 2) foi maior que o EQM de teste das redes **RNF1**, **RNFR1** e **RNFR2**. Por outro lado, embora o EQM de teste obtido pela rede **MLP** seja menor que os obtidos pelas redes propostas, cabe considerar que o tempo de processamento da **MLP** foi maior, assim como o número de parâmetros e o número de entradas (cinco, conforme sugere a expressão (8)). Claramente, o número de parâmetros no caso das redes neurofuzzy propostas é substancialmente menor.

Resumindo, as redes neurofuzzy híbridas analisadas neste trabalho produzem modelos mais simples e parcimoniosos pois contém um número reduzido de entradas e de parâmetros, requer um tempo baixo de aprendizagem, além de proporcionar menores erros de aproximação do que redes com estruturas similares sugeridas na literatura. Isto significa maior eficiência computacional em termos de exigências de memória e tempo de processamento, além de modelos mais transparentes e com erros aceitáveis de aproximação.

Tabela 1: Parâmetros característicos das estruturas.

Estrutura	n	p	Np	Na_{max}	Parâmetros / iteração	Parâmetros
RNF1	5	1	[7,7,7,6,6]	32	$Na_{max} \cdot (n+p)$	192
RNFR1	1	1	[30]	2	$Na_{max} \cdot (n+p+1)$	6
RNFR2	1	1	[30]	2	$Na_{max} \cdot (n+p+Na_{max})$	8
RNR2[1]	1	1	[50]	50	$Na_{max} \cdot (n+p+2)+p$	201
RNFR3	2	1	[50,50]	50	$3 \cdot n \cdot Na_{max} + Na_{max} \cdot p$	350
MLP[2]	5	1	[30,20]	[30,20]	$M_1 \cdot (n+1) + M_2 \cdot (M_1+1) + (M_2+1) \cdot p$	821

[1] Para RNR2, $Np = Na_{max}$ é o número de neurônios da camada intermediária.

[2] Para MLP, $Np = Na_{max} = [M_1,\ M_2]$, sendo M_1 e M_2 o número de neurônios das camadas intermediárias.

Tabela 2: Erros quadráticos médios (EQM) para o exemplo de identificação.

Estrutura	Épocas	T_{proc} (s)	EQM. Treinamento $(\times 10^{-6})$	EQM. Teste $(\times 10^{-3})$
RNF1	25	28	0.2525	0.1394
RNFR1	800	281	0.1000	0.3839
RNFR2	800	315	0.1000	0.3180
RNR2	600	329	1.2654	0.4358
RNFR3	500	343	7.5300	1.6750
MLP	1000	563	74.2830	0.0664

5. Conclusões

Neste trabalho três diferentes estruturas de redes neurofuzzy foram consideradas. As estruturas destas redes contém um sistema de inferência fuzzy composto por neurônios lógicos que utilizam normas triangulares para o processamento da informação. O sistema de inferência fuzzy está conectado em série com uma rede neural clássica, constituindo assim uma rede neural híbrida. A aprendizagem combina métodos de gradiente com um mecanismos de reforço associativo, recompensa e punição.

Experimentos computacionais, considerando um problema de identificação de processo não linear, sugerem que, quando comparadas com estruturas similares propostas na literatura, as redes neurofuzzy híbridas são eficazes para aproximar modelos de sistemas dinâmicos não lineares. Estas redes fornecem modelos mais simples, com erros pequenos de aproximação e recursos computacionais modestos.

Agradecimentos

Os autores agradecem ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) processos $\sharp 300729/86-3$ e $\sharp 132544/2001-6$, pelo auxilio.

Referências

[1] C. Lee and C. Teng. Identification and control of dynamic systems using recurrent fuzzy neural networks. *IEEE Transactions on Fuzzy Systems*, 4(8):349–366, 2000.

[2] W. Pedrycz and F. Gomide. *An Introduction to Fuzzy Sets: Analysis and Design*. MIT Press, Cambridge, MA, 1998.

[3] I. Luna, R. Ballini, and F. Gomide. Rede Neurofuzzy Recorrente para Identificação e Controle de Sistemas Dinâmicos Discretos. *Anais do XIV Congresso Brasileiro de Automação, Natal - Brasil*, pages 353–358, Setembro 2002.

[4] A. Barto and M. Jordan. Gradient following without backpropagation in layered networks. *in Proceedings of the IEEE First International Conference on Neural Networks*, (2):629–636, 1987.

[5] W. Caminhas, H. Tavares, F. Gomide, and W. Pedrycz. Fuzzy sets based neural networks: Structure, learning and applications. *Journal of Advanced Computational Intelligence*, 3(3):151–157, 1999.

[6] K. Narendra and K. Parthasarathy. Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, 1(1):4–27, 1990.

[7] Y. Cheng, T. Karjala, and D. Himmelblau. Closed loop nonlinear process identification using internally recurrent nets. *Neural Networks*, 10(3):573–586, 1997.

[8] H. Bersini and V. Gorrini. Recurrent fuzzy systems. Technical Report TR/IRIDIA/94-11, Institut de Reserches Interdisciplinaires et de Développements en Intelligence Artificielle, 1994.

[9] E. Iyoda. Inteligência computacional no projeto automático de redes neurais híbridas e redes neurofuzzy heterogêneas. Master's thesis, FEEC - Unicamp, Brasil, Janeiro 2000.

[10] R. Ballini and F. Gomide. Learning in recurrent, hybrid neurofuzzy networks. *IEEE International Conference on Fuzzy Systems*, pages 785–791, 2002.