Anais do IX Congresso Brasileiro de Redes Neurais / Inteligência Computacional (IX CBRN)
Ouro Preto 25-28 de Outubro de 2009

REDES NEURAIS GRANULARES EVOLUTIVAS EM MODELAGEM DE SISTEMAS

Daniel F. Leite*, Luiz Bergo Jr.†, Pyramo Costa Jr.†, Fernando Gomide*

*Faculdade de Engenharia Elétrica e Computação - Universidade Estadual de Campinas (UNICAMP), Campinas, SP, Brasil

†Programa de Pós-Graduação em Engenharia Elétrica - Universidade Católica de Minas Gerais (PUC-MG), Belo Horizonte, MG, Brasil

Emails: danfl7@dca.fee.unicamp.br, bergojr@hotmail.com, pyramo@pucminas.br, gomide@dca.fee.unicamp.br

Abstract— In this paper, an evolving granular neural network (eGNN) approach is suggested to model unknown, nonstationary dynamic systems. The eGNN is developed under the framework of information granulation and is equipped with incremental data stream-based learning algorithm, primarily designed to work in online mode. The granular neural network develops its structure and refines parameters as data are input. The network is able to model nonstationary dynamic systems using incremental adaptability, the concept of granules, and T-S neurons as basic processing elements. To illustrate the effectiveness of the approach, the paper addresses an instance of modeling moving-train coupling problem using actual data. Experimental results show that eGNN outperforms static, recurrent and evolving modeling alternatives proposed in the literature.

Keywords— Granular Computing, Evolving Systems, Nonlinear System Modeling.

Resumo— Este artigo sugere redes neurais granulares evolutivas (eGNN) como uma abordagem para a modelagem de sistemas dinâmicos não lineares. As eGNN são desenvolvidas utilizando o conceito de granulação de informação junto com algoritmos de aprendizagem incrementais baseado em fluxo de dados. As eGNN são projetadas para funcionar *on-line* desenvolvendo sua estrutura e refinando seus parâmetros a medida que dados são fornecidos. As eGNN utilizam o conceito de grânulos e neurônios T-S como elementos básicos de processamento. Para ilustrar a abordagem, o artigo considera um problema de acoplamento de trens em movimento utilizando dados reais. Resultados experimentais mostram que as eGNN possuem um desempenho superior quando comparado com modelos neurais estáticos, recorrentes, e com abordagens evolutivas sugeridas na literatura.

Palavras-chave— Computação Granular, Sistemas Evolutivos, Modelagem de Sistemas Não Lineares.

1 Introdução

Desde o começo deste século temos testemunhado uma revolução no processamento de informação; atualmente, grandes quantidades de dados são produzidas de forma rápida e crescente como em medições provenientes de sensores em processos industriais avançados, sistemas autônomos, usuários de internet, indústria financeira, mercados de consumo, entre outros. Uns dos desafios enfrentados estão relacionados com a necessidade de lidar com enormes quantidades e processar fluxos de dados em tempo real.

O armazenamento de todos os dados provenientes de um processo e sua análise *off-line* é muitas vezes impossível ou impraticável. Além disto, fluxos de dados freqüentemente são não estacionários e modelos construídos usando dados correntes podem se tornar obsoletos posteriormente.

Modelos de sistemas dinâmicos podem ser obtidos a partir de uma base de dados e algoritmos que operam *on-line*, mas freqüentemente estes algoritmos não modificam a estrutura dos modelos pois podem não capturar novas características presentes no fluxo de dados. Em abordagens evolutivas a estrutura e os parâmetros do modelo não são fixos e ambos se adaptam de forma incremental, *on-line*. Exemplos de modelos evolutivos largamente discutidos na literatura incluem Fuzzy ARTMAP (Carpenter, 1992), Growing Neural Gas (Fritzke, 1995), Unsupervised Evolving Connectionist Systems (Kasabov, 2007), Evolving Fuzzy Neural Network Model (Kasabov, 2001), Evolving Classification Systems (Angelov e Zhou, 2008), entre outros.

Este artigo sugere uma rede neural granular evolutiva (eGNN) para modelagem *on-line* de sistemas não lineares e não estacionários. A eGNN utiliza uma abordagem de modelagem que gradualmente adapta sua estrutura - grânulos de informação e neurônios T-S - bem como os parâmetros associados utilizando fluxo de dados. A eGNN desenvolve modelos globais usando algoritmos incrementais de um passo sobre os dados e requer quantidades modestas de memória. Além disso, a eGNN não exige nenhum conhecimento *a priori* sobre o sistema e dados correspondentes (distribuição de probabilidade e estatísticas) e inicia a aprendizagem assim que os dados são fornecidos. Para ilustrar seu potencial na resolução de problemas de modelagem complexos, o artigo trata de um problema de modelagem de acoplamento de trens em movimento usando dados reais.

Após esta introdução, o artigo prossegue introduzindo os neurônios T-S na Seção 2. As Seções 3 e 4 detalham a eGNN e seu respectivo algoritmo de aprendizado. A Seção 5 apresenta os resultados do comportamento de eGNN na modelagem de acoplamento de trens em movimento e o compara com abordagens alternativas. A Seção 6 conclui o artigo e sugere itens para investigação futura.

2 Neurônios T-S

Neurônios T-S são implementações neurais de funções de agregação. Neste artigo, enfatizamos *nullnormas* (Calvo, 2001), uma classe especial de operadores de agregação que inclui T-normas e T-conormas (S-normas) como casos limite. *Nullnormas* estendem T-normas e S-normas pois elas oferecem flexibilidade na escolha do elemento neutro da operação de agregação. Em geral, *nullnormas* combinam T-normas e S-normas. Como resultado consegue-se operadores de agregação com propriedades *and* e *or* dos operadores lógicos que ocorrem nestas construções. Neurônios T-S herdam as características de conectivo lógico de normas T e S como uma operação de agregação.

2.1 Nullnormas

T-normas (T) e T-conormas (S) são operadores binários sobre o intervalo unitário que são comutativos, associativos, crescentes, com as seguintes condições de contorno: $T(a,0) = 0$ e $T(a,1) = a$; $S(a,0) = a$ e $S(a,1) = 1$, $a \in [0,1]$. O elemento neutro de T e S normas são $e = 1$ e $e = 0$, respectivamente. Considere uma norma triangular contínua T e uma conorma triangular contínua S. Um operador binário F é chamado de operador de agregação T-S se ele é crescente, comutativo, e satisfaz as condições de contorno $F(a,0) = T(F(1,0),a)$ e $F(a,1) = S(F(1,0),a), \forall a \in [0,1]$.

Nullnormas são uma forma de generalizar normas triangulares. Estas normas são flexíveis com relação à escolha do elemento neutro, admitindo $e \in [0,1]$. Quando $e = 0$, uma *nullnorma* se torna uma T-norma. Por outro lado, quando $e = 1$, a *nullnorma* se torna uma S-norma. Formalmente, uma *nullnorma* V é uma operação $V : [0,1] \times [0,1] \rightarrow [0,1]$ que é comutativa, $V(a,b) = V(b,a)$, monotônica, $V(a,b) \leq V(a,c)$, se $b \leq c$ e associativa, $V(a,V(b,c)) = V(V(a,b),c)$, cujo elemento neutro $e \in [0,1]$ tal que $V(a,e) = e$ e que satisfaz $V(a,0) = a \ \forall \ a \in [0,e]$ e $V(a,1) = a \ \forall \ a \in [e,1]$, onde $a,b,c \in [0,1]$. Limitamos o estudo deste artigo à família de *nullnormas* definida pelo seguinte construtor:

$$V(a,b) = \begin{cases} eS\left(\frac{a}{e},\frac{b}{e}\right) & ,a,b \in [0,e], \\ e + (1-e)\,T\left(\frac{a-e}{1-e},\frac{b-e}{1-e}\right) & a,b \in [e,1], \\ e & \text{caso contrário.} \end{cases}$$

Uma motivação para o uso de *nullnormas* decorre do fato de que T-normas descrevem situações em que ambas as condições a e b são absolutamente necessárias. Se uma destas condições não for satisfeita, então rejeita-se completamente a alternativa correspondente. Com *nullnormas*, se uma das condições não for satisfeita, podemos ainda considerar a contribuição da outra condição na operação de agregação. *Nullnormas* são operadores de agregação T-S que são contínuos.

2.2 Neurônios T-S

Neurônios T-S são modelos de neurônios artificiais que utilizam *nullnormas* como operação de agregação. Formalmente, seja $x = \{x_1, x_2, ..., x_n\}$ um vetor de entrada, $x \in [0,1]^n$, e $w = \{w_1, w_2, ..., w_n\}$ o correspondente vetor de pesos, $w \in [0,1]^n$. Adotando o produto algébrico (em geral pode ser uma T-norma, mas este artigo utiliza o produto) no processamento sináptico e uma *nullnorma* na agregação, a saída $o \in [0,1]$ de um neurônio T-S é:

$$o = V(x_1 w_1, x_2 w_2, ..., x_n w_n).$$

Esta expressão será denotada por $o = V(x,w)$. Estruturas com neurônios T-S apresentam uma diversidade de características não lineares que o mapeamento entre as entradas e saídas pode assumir dependendo da escolha de w e e. Em particular, o valor do elemento neutro permite operações de agregação com características intermediárias entre T-normas e S-normas.

3 Rede Neural Granular Evolutiva

O conceito de redes neurais granulares (GNN) foi introduzido por Pedrycz e Vukovich (2001) enquanto que o conceito de eGNN foi sugerido por Leite, Costa Jr. e Gomide (2009). O aprendizado de GNNs e eGNNs seguem um princípio comum que envolve duas etapas: **i)** granulação do espaço e/ou dados de entrada. Neste passo forma-se uma coleção de grânulos que particiona o universo das variáveis de entrada; **ii)** Construção/Refinamento da rede neural. Aqui, o aprendizado é baseado nos grânulos de informação ao invés dos dados originais. Como conseqüência, a rede neural não utiliza, necessariamente, todos os dados originais, em geral muito mais numerosos. Modelos eGNN se diferem de modelos GNN principalmente devido à sua característica de adaptação incremental utilizando fluxos de dados. Em eGNN as etapas **i)** e **ii)** são executadas sempre que um novo dado é fornecido.

Originalmente, a eGNN foi proposta como um classificador evolutivo. Ela superou o desempenho de modelos não lineares alternativos em problemas clássicos de reconhecimento de padrões como *Iris* e *Wine*. Este artigo apresenta um novo modelo eGNN para a modelagem de sistemas não lineares.

Modelos eGNN são construídos a partir do processamento de fluxos de dados usando algoritmos de aprendizado incrementais. As redes eGNN criam granulações de informação usando grânulos e neurônios T-S como elementos de processamento. As características principais destes modelos são aprendizado contínuo, auto-organização e adaptação. Regras do tipo SE-ENTÃO podem ser extraídas de sua estrutura e parâmetros durante o processo de evolução. Grânulos são incrementalmente criados ou atualizados durante a evolução.

Em particular, quando a entrada atual é percebida pelos grânulos existentes, três tipos de situações ocorrem: **i)** dois ou mais grânulos existentes estão aptos a se adaptarem a nova entrada. Neste caso, aquele que apresentar o maior valor produzido pela operação de agregação é selecionado para acomodar a entrada. A acomodação consiste no ajuste dos parâmetros dos grânulos e dos neurônios T-S para estimarem os coeficientes de funções locais em passos futuros; **ii)** a entrada atual pertence aos limites de apenas um grânulo. Aqui, o grânulo, seu correspondente neurônio T-S e a função local associada ao grânulo são atualizados para acomodar a entrada; e **iii)** nenhum dos grânulos existentes acomoda a entrada. Neste caso um novo grânulo é criado.

4 Aprendizagem em eGNN

Modelos eGNN aprendem a partir de um fluxo de dados $(x, y)^{[h]}$, $h = 1, 2, ...$, onde o valor de saída $y_k^{[h]}, k = 1, ..., m$ é assumido ser conhecido dado o vetor de entrada correspondente $x_j^{[h]}, j = 1, ..., n$. eGNN acomoda o novo conhecimento contido no dado criando um novo grânulo de informação ou adaptando os grânulos e neurônios T-S existentes. A cada grânulo γ^i de uma coleção finita de grânulos $\gamma = \{\gamma^1, \gamma^2, ..., \gamma^c\}$, definida no espaço de entrada $X \subseteq \Re^n$, são associadas funções

$$\bar{y}_k^i = a_{0k}^i + \sum_{j=1}^{n} a_{jk}^i x_j, \quad \forall i, k$$

definidas no espaço de saída $Y \subseteq \Re^m$. Sempre que γ^i está ativo para algum $x_j^{[h]} \forall j$, estima-se $a_{jk}^i \ \forall j, k$, usando o método dos mínimos quadrados.

A estrutura eGNN é ilustrada na Figura 1. A rede consiste de cinco camadas. A camada de entrada apresenta os vetores $x_j^{[h]}$, $j = 1, ..., n$, à rede. A camada evolutiva consiste de grânulos de informação γ^i, $i = 1, ..., c$, extraídos a partir do fluxo de dados. Grânulos γ^i são definidos por funções de pertinência A_j^i, $j = 1, ..., n$, nos respectivos universos, cada uma associada à uma componente do vetor de entrada. Na camada de agregação se encontram os neurônios T-S, $TSn^i \forall i$. Estes processam entradas normalizadas $\tilde{x}_j^{i[h]} \forall j, i$, provenientes do processo de granulação, e agregam resultados para gerar a saída $o^i \forall i$. A saída pode ser vista como uma medida de compatibilidade entre o dado e os grânulos existentes. A camada de decisão compara os valores da agregação e o grânulo com o maior valor, o grânulo vencedor γ^ν, produz os resultados gerados pelos polinômios $\bar{y}_k^\nu \ \forall k$ associados a γ^ν. A camada de saída compara a saída m-dimensional estimada pela rede $\bar{y}_k^{\nu[h]}$ com o vetor de saída desejado $y_k^{[h]} \forall k$. Esta operação resulta no erro de estimação ϵ_k, $k = 1, ..., m$. Em modelos eGNN, nenhum grânulo e neurônio T-S existe anteriormente ao aprendizado. Estes são criados e adaptados durante o processo evolutivo sempre que novas informações são encontradas no fluxo de dados. Os pesos das conexões $w_j^i \forall j, i$, associam diferentes graus de relevância a diferentes atributos (*wrapper* incremental), enquanto os pesos $\delta^i \forall i$, dependem da quantidade de dados pertencendo a γ^i. Em geral, δ^i é visto como um mecanismo para indicar regiões de dados mais densas ou esparsas no espaço de entrada.

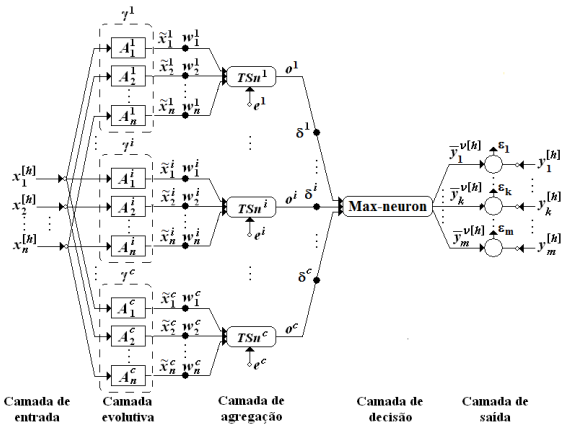

Figura 1: Rede neural granular evolutiva

Seja $\rho \in X \subseteq \Re^n$ o tamanho máximo que um grânulo pode assumir; escolhas adequadas de ρ são muito importantes pois ele está diretamente relacionado com a transparência e precisão do modelo. Grânulos com valores grandes de ρ podem resultar em perda de especificidade, enquanto que valores muito pequenos acarretam em modelos mais complexos. O valor de ρ dita a granularidade do espaço de entrada. Geralmente, quanto maior ρ, menor o número de grânulos criados e, apesar de a interpretação ser mais simples, menor é a capacidade de aproximar não-linearidades. Por outro lado, valores pequenos de ρ podem levar ao sobre-ajuste do modelo e a perda de interpretabilidade. Um valor adequado de ρ é aquele que representa um compromisso aceitável entre complexidade e interpretabilidade.

4.1 Criação e Atualização de Grânulos

Funções de pertinência $A_j^i \forall j$ associadas aos grânulos γ^i são definidas em diferentes domínios. Assumimos funções trapezoidais ou triangulares para A_j^i de γ^i. Logo, as funções são definidas por quatro parâmetros: $j-$ésimo limite inferior, l_j^i; $j-$ésimo limite superior, L_j^i; e valores intermediários, λ_j^i e Λ_j^i. Para funções trapezoidais $\lambda_j^i < \Lambda_j^i$, e para funções triangulares $\lambda_j^i = \Lambda_j^i$.

Sempre que um novo dado $x_j^{[h]} \notin [l_j^i, L_j^i] \forall j, i$, cria-se um novo grânulo. O novo grânulo γ^{c+1} é construído usando funções triangulares $A_j^{c+1} \forall j$, com centro $x^{[h]} \in \Re^n$. Seus parâmetros assumem $l_j^{c+1} = x_j^{[h]} - \frac{\rho_j}{2}$; $\lambda_j^{c+1} = \Lambda_j^{c+1} = x_j^{[h]}$; e $L_j^{c+1} = x_j^{[h]} + \frac{\rho_j}{2}$; $\forall j$. Caso $x^{[h+\Delta]}$, onde Δ é um

inteiro positivo, pertença aos limites de γ^i, então os parâmetros λ_j^i e $\Lambda_j^i \forall j$ são atualizados acomodando $x_j^{[h+\Delta]} \forall j$. Basicamente, a adaptação consiste em ajustar $\lambda_j^i = x_j^{[h+\Delta]}$, se $x_j^{[h+\Delta]} \in [l_j^i, \lambda_j^i]$; ou $\Lambda_j^i = x_j^{[h+\Delta]}$, se $x_j^{[h+\Delta]} \in [\Lambda_j^i, L_j^i]$. Polinômios $\overline{y}_k^i$, $k = 1, ..., m$ são associados aos grânulos γ^i. Seus coeficientes a_{jk}^i, $j = 0, ..., n$; $k = 1, ..., m$, são estimados usando mínimos quadrados sempre que γ^i é vencedor para um dado $x^{[h]}$.

4.2 Ajuste dos Pesos da Camada Evolutiva

Os pesos da camada evolutiva $w_j^i \ \forall j, i$, ponderam a importância da componente j na saida da rede. Inicialmente, todos w_j^i são 1 e durante o aprendizado os valores de w_j^i são ajustados de acordo com os dados.

Novos dados $(x, y)^{[h]}$ podem causar a revisão de γ^{i_1} se $o^{i_1} > 0$, mas $\exists\ o^{i_2}$ tal que $o^{i_2} > o^{i_1}$. O seguinte procedimento é usado para comprimir γ^i reduzindo sua compatibilidade com $x^{[h]}$. Duas situações são de interesse: **i)** se A_j^i, $i = 1, ..., c$, para algum j resulta em $\tilde{x}_j^{i[h]} \in]0, 1[$, então faça $l_j^i = x_j^{[h]}$ se $x_j^{[h]} < \lambda_j^i$, ou faça $L_j^i = x_j^{[h]}$ se $x_j^{[h]} > \Lambda_j^i$. Este procedimento comprime a função de pertinência A_j^i de γ^i; e **ii)** se $A_j^i \forall i$ é tal que $\tilde{x}_j^{i[h]} = 1$ para algum j, então os parâmetros de A_j^i são mantidos e o peso associado w_j^i é ajustado:

$$w_j^i(\text{novo}) = \beta w_j^i(\text{velho}) \quad ,$$

onde $\beta \in]0, 1[$ é uma constante de decaimento. O procedimento de atualização é justificado porque, nesta situação, A_j^i não contribui satisfatoriamente com o processo de granulação.

4.3 Ajuste dos Elementos Neutros

O modelo eGNN sugerido usa neurônios T-S anddominados para processar dados. O $i-$ésimo neurônio TSn^i processa os n atributos da entrada normalizada $\tilde{x}_j^{i[h]}$, $j = 1, ..., n$, associada ao grânulo γ^i através de $V(\tilde{x}_j^{i[h]}, w_j^i)\ \forall j$. O resultado é um valor de saída único o^i que pode ser visto como a compatibilidade entre $x^{[h]}$ e γ^i. Um procedimento para inicializar e ajustar o elemento de absorção de neurônios T-S é o seguinte: escolher uma T-norma e uma S-norma para os neurônios T-S; quando um grânulo γ^{c+1} é criado, ajustar seu elemento de absorção fazendo $e^{c+1} = 0$. Isto induz uma T-norma. Durante a evolução, alguns e^i, $i = 1, ..., c$, podem aumentar seus valores dependendo dos dados. Por exemplo, caso um ou poucos atributos de $x_j^{[h]} \forall j$ não ative as funções $A_j^i \forall j$ ainda podemos considerar γ^i como candidato para acomodar $(x, y)^{[h]}$ aumentando o elemento neutro de seu respectivo neurônio T-S como segue

$$e^i(\text{novo}) = e^i(\text{velho}) + \chi(1 - e^i(\text{velho})) \quad ,$$

onde $\chi \in]0, 1[$ é uma taxa de adaptação. Ajustando e^i durante a evolução, valores baixos de pertinência são compensados por valores maiores usando processamento baseado em S-norma. Caso poucos atributos do dado atual não pertençam a γ^i, então o dado pode ainda assim ser compatível com γ^i com um certo grau aumentando e^i. Este ajuste geralmente evita a criação de grânulos muito similares e ajuda a manter uma quantidade razoável de grânulos na estrutura do modelo.

4.4 Ajuste dos Pesos da Camada de Decisão

As conexões da camada de decisão são ponderadas por δ^i. Este representa a quantidade de dados no contexto de $\gamma^i \forall i$. Em geral, os pesos δ^i são uma forma de identificar regiões densas e esparsas de dados. Quanto maior o valor de δ^i, maior a chance de ativação de γ^i em passos subseqüentes.

Inicialmente, faz-se $\delta^i = 1, \forall i$. Durante a evolução, os valores de alguns δ^i podem reduzir ou aumentar dependendo de critérios pré-estabelecidos. Por exemplo, um critério simples é o seguinte: se γ^i não ativar após um certo número de passos, então reduzir δ^i fazendo

$$\delta^i(\text{novo}) = \zeta^i \delta^i(\text{velho}) \quad ,$$

onde $\zeta^i \in]0, 1[\ \forall i$. Caso contrário, se γ^i ativa com freqüência, aumentar δ^i usando

$$\delta^i(\text{novo}) = \delta^i(\text{velho}) + \zeta^i(1 - \delta^i(\text{velho})) \quad .$$

Uma conseqüência deste procedimento é que, em ambientes não estacionários, grânulos inativos durante um número de iterações são suprimidos. Isto significa que o processo atual mudou e a exclusão de grânulos significa $\delta^i \to 0^+$. Isto é justificado para manter uma quantidade razoável de informação atualizada disponível. Caso a aplicação requeira a memorização de eventos raros, então o procedimento de exclusão é proibitivo.

4.5 O Neurônio Max

O neurônio max do modelo eGNN processa sua entrada c-dimensional, onde cada componente representa o grau de compatibilidade entre $\gamma^i \forall i$ e $x^{[h]}$, e determina o grânulo vencedor γ^ν para $x^{[h]}$. O neurônio max apresenta na saída da rede os valores $\overline{y}_k^\nu(x^{[h]})$ associados a γ^ν. O erro de estimação da rede é computado por

$$\epsilon_k = |y_k^{[h]} - \overline{y}_k^\nu(x^{[h]})|, \quad \forall k.$$

Caso $\epsilon_k < \wp\ \forall k$, onde $\wp$ é um limiar, refinam-se os parâmetros da rede. Caso $\epsilon_k \geq \wp$ para algum k, os mecanismos de compressão de grânulos e ajuste de pesos procedem.

5 Acoplamento de Trens em Movimento

Descrição do problema: O acoplamento de trens em movimento é uma alternativa para melhorar a

produtividade de malhas ferroviárias. Um dos motivos para se acoplar trens em movimento é a necessidade de um elemento trator adicional, denominado *Helper*, para superar rampas críticas. Basicamente, o *Helper* aproxima-se de um trem pela parte traseira e usa sensores de distância e velocidade para realizar frenagem. Esta manobra deve ser precisa o suficiente para assegurar um acoplamento suave entre os trens. Dados reais de curvas de frenagem de um *Helper* são utilizados neste experimento.

Características da base de dados: Número de exemplos: 1730; Número de atributos: 4 entradas quantitativas especificamente, distância (dam) e velocidade relativa entre trens (km/h), velocidade atual do Helper (km/h) e pressão de freio momentânea (psi), e uma saída representando a pressão do freio no instante seguinte; Dados de treino/teste: 60/100 %. Para comparação de performance, consideramos os modelos: redes perceptron multi-camadas (MLP) e Elman, extended e evolving Takagi-Sugeno (xTS) (eTS), e eGNN.

Resultados: Para avaliar o efeito de diferentes parametrizações, eGNN adota dois conjuntos de parâmetros: eGNN-1 usa $\rho_1 = 0.7$, $\beta_1 = \zeta_1 = 0.9$, $\chi_1 = 0.1$; e eGNN-2 emprega $\rho_2 = 0.2$, $\beta_2 = \zeta_2 = 0.95$, $\chi_2 = 0.05$. Cada experimento é processado cinco vezes com os mesmos parâmetros. Em cada realização, os dados são apresentados seqüencialmente apenas uma vez a eGNN. eGNN inicia a aprendizagem com uma rede vazia e sem pré-treinamento. A Tabela I mostra a performance de eGNN e dos demais modelos segundo o erro médio quadrático de aproximação (NRMSE). A Figura 2 ilustra o valor da pressão aplicada no freio do Helper.

Tabela 1: Acoplamento de Trens em Movimento

Modelo	No. grânulos/neurônios	NRMSE^{+}	NRMSE^{++}
MLP	21	0.0282	0.0308
Elman	21	0.0270	0.0301
eTS*	9	0.0279	0.0284
xTS*	6	0.0275	0.0279
eGNN-1*	5	0.0156	0.0160
eGNN-2*	18	0.0134	0.0143

* Modelos *on-line*; $^{+}$ Melhor; $^{++}$ Média.

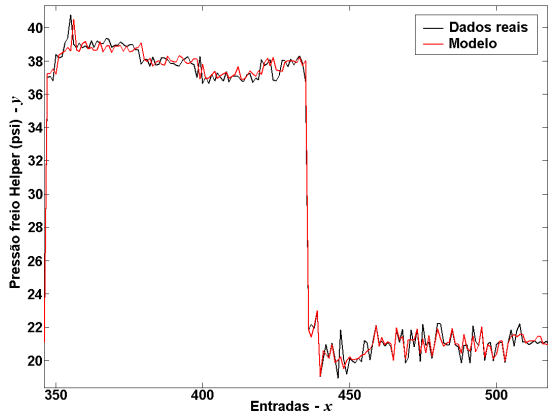

Figura 2: Saída do sistema real e do modelo eGNN

Os resultados mostram que os modelos eGNN obtiveram as melhores performances neste problema. Além disso, eles podem operar em tempo real usando estruturas compactas. Os fatores de maior influência neste estudo são as peculiaridades de cada algoritmo de aprendizado e as diferentes considerações estruturais. No caso de eGNN salienta-se o maior grau de flexibilidade dos neurônios T-S, sua estrutura adaptativa e a abordagem baseada em grânulos. Modelos eGNN mostraram um forte potencial para a resolução de problemas de modelagem que demandam adaptabilidade incremental, o que é particularmente atrativo para o caso de sistemas não estacionários.

6 Conclusão

Uma rede neural granular evolutiva, eGNN, foi sugerida como um mecanismo de modelagem adaptativa de sistemas não lineares. A abordagem eGNN evolui modelos a partir de grânulos de informação e neurônios T-S. Isto provê à modelos eGNN a habilidade de desenvolver estruturas altamente flexíveis e mecanismos robustos para acompanhar a evolução. Experimentos com um problema de acoplamento de trens em movimento mostraram que eGNN é competitiva quando comparada à técnicas de modelagem não lineares alternativas. Trabalhos futuros deverão considerar problemas de controle adaptativo.

Agradecimento

O primeiro autor é grato a CAPES pelo suporte. O último autor agradece ao CNPq processo 304857/2006-8.

Referências

Angelov, P.; Zhou, X. (2008). "Evolving Fuzzy Rule-Based Classifiers from Data Streams". *IEEE Trans. on Fuzzy Systems*, Vol.16-6, pp:1462-1475.

Calvo, T.; De Baets, J.; Fodor, J. (2001). "The functional equations of Frank and Alsina for uninorms and nullnorms". *Fuzzy Sets and Systems*, Vol. 120, pp: 385-394.

Carpenter, G. A.; et al. (1992). "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning". *IEEE Transactions on Neural Networks*, Vol. 3-5, pp: 698-713.

Fritzke, B. (1995). "A Growing Neural Gas Network Learns Topologies". *Advances in Neural Information Processing Systems*, pp: 625-632.

Kasabov, N. (2007). Evolving Connectionist Systems. Springer-Verlag London, 2^{nd} edition.

Kasabov, N. (2001). "Evolving fuzzy neural networks for supervised/unsupervised online knowledge-based learning". *IEEE Transactions on SMC - Part B*, Vol. 31-6, pp: 902-918.

Leite, D. F.; Costa Jr., P.; Gomide, F. (2009). "Evolving Granular Classification Neural Networks". *International Joint Conference on Neural Networks*, 8p (Forthcoming).

Pedrycz, W.; Vukovich, W. (2001). "Granular Neural Networks". *Neurocomp.*, V.36, pp:205-224.