SISTEMAS CONEXIONISTAS EVOLUTIVOS

Daniel F. Leite*, 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, pyramo@pucminas.br, gomide@dca.fee.unicamp.br

Abstract— The aim of this study is to introduce the concept of evolving connectionist systems (ECOS) and to suggest a framework for adaptive processes modeling. The outlined approaches are online neural models constructed from data streams. Basically, ECOS accommodate the possible new knowledge contained in the data in their structures and parameters using one-pass non-recursive incremental learning algorithms. ECOS applications extend to classification, regression, prediction and control problems in continuously changing environments. Their main characteristics include continuous learning, self-organization and adaptation to unknown environments. To illustrate the effectiveness of the modeling approach, the paper considers two types of ECOS, namely evolving Multi-Layer Perceptron (eMLP) and evolving Self-Organizing Map (eSOM), for solving a real electrical machines fault diagnosis problem.

Keywords— Evolving Systems, Neural Networks, Pattern Recognition.

Resumo— O objetivo deste estudo é introduzir o conceito de sistemas conexionistas evolutivos (ECOS) e sugerir abordagens para a modelagem de processos que demandam adaptabilidade incremental. ECOS são modelos neurais *on-line* construídos a partir de fluxos de dados. Basicamente, eles acomodam o possivel novo conhecimento contido nos dados em suas estruturas e parâmetros usando algoritmos de aprendizado incrementais, não-recursivos e de um passo. As aplicações de ECOS se estendem a problemas de classificação, regressão, predição e controle em ambientes sempre em mudança. Suas características principais incluem aprendizado contínuo, auto-organização e adaptação a ambientes desconhecidos. Para ilustrar a efetividade da abordagem, o artigo considera dois tipos de ECOS, Perceptron Multi-Camadas evolutivo (eMLP) e Mapa Auto-Organizável evolutivo (eSOM), para resolução de um problema real de diagnóstico de falta em máquinas elétricas.

Keywords— Sistemas Evolutivos, Redes Neurais, Classificação de Padrões.

1 Introdução

A revolução digital tem facilitado a captura e o armazenamento de dados (Fayyad e Uthurusamy, 1996) (Inmon, 1996). Com o rápido desenvolvimento de hardwares e softwares, grandes quantidades de dados têm sido coletadas e armazenadas em bases de dados. A taxa com que estes dados têm sido armazenados tem crescido rapidamente. Como resultado, algumas técnicas estatísticas e de inteligência computacional para aprendizado de máquina e construção de modelos se tornaram inadequadas ou pouco eficientes (Mitra et al., 2002). Dados brutos são raramente úteis de forma direta. O seu real valor depende da habilidade de extração de informações úteis contidas nos dados para suporte a decisões ou exploração e entendimento do fenômeno que governa a fonte de dados.

Em certos contextos, a análise dos dados é um processo manual. Um especialista pode ser altamente capaz de lidar com certos tipos de dados provenientes de um mesmo fenômeno assim provendo sumários e/ou relatórios a respeito do processo. Nestes casos, o próprio especialista pode ser visto como um processador sofisticado ou um modelo do processo. Entretanto, é evidente que a medida que a quantidade e dimensão dos dados aumentam, esta forma de análise colapsa. Especialistas recorrem então a tecnologias computacionais para automatizar estes processos que lidam com grandes quantidades de dados.

Tem emergido, ao longo de aproximadamente uma década, um novo paradigma no campo da inteligência computacional baseado na construção de modelos a partir de algoritmos incrementais e fluxos de dados. Estes modelos, ditos evolutivos, não somente minimizam o problema do armazenamento de grandes quantidades de dados, mas também oferecem características importantes para a modelagem de processos não-lineares adaptativos. As principais características de modelos evolutivos são aprendizado contínuo, auto-organização e adaptação a ambientes desconhecidos. Geralmente, regras de associação podem ser facilmente extraídas a partir de suas estruturas e parâmetros em qualquer passo durante o processo evolutivo.

Um dos mais importantes aspectos da inteligência de sistemas recai sobre a habilidade de adaptação de modelos a situações inéditas (Bouchachia et al., 2007). Para alcançar adaptação, estes modelos devem ter suporte de algoritmos de aprendizado que impliquem um melhoramento contínuo ou ao menos a não-degradação do desempenho do sistema em ambientes sempre em mudança. As abordagens sugeridas neste artigo são modelos conexionistas orientados a problemas de classificação, regressão, predição e controle que apresentam adaptabilidade incremental *on-line*. Em particular, pesquisadores estão fazendo frente ao desafio de modelar fluxos de dados usando algoritmos incrementais que adaptam a estrutura e os parâmetros dos modelos baseados nas

possíveis novas informações contidas nos dados.

Um processo evolutivo pode ser definido como um processo que desenvolve, modificando-se de forma contínua ao longo do tempo. Este processo interage com outros processos do ambiente sendo que, a princípio, não é possível determinar o resultado destas interações. Processos evolutivos são geralmente difíceis de se modelar, pois alguns de seus parâmetros podem não ser conhecidos *a priori*, e perturbações ou mudanças inesperadas podem acontecer em qualquer momento durante o desenvolvimento.

Por outro lado, um modelo é dito ser evolutivo se ele possui as seguintes características:

∗ Habilidade de aprendizado *on-line* a partir de fluxos de dados. Uma vez que os dados são processados não há como recorrer a dados históricos;
∗ Nenhum conhecimento anterior sobre a estrutura do sistema é necessário (adaptação estrutural *on-line*);
∗ Nenhum conhecimento prévio sobre as propriedades estatísticas dos dados é requerido (modelo não-paramétrico);
∗ Nenhuma inicialização de protótipos é requerida.

Além das características mencionadas acima, é muitas vezes desejado que modelos evolutivos possuam rápida assimilação de conhecimento, poucos requerimentos de memória e habilidade não-linear de aproximação de funções ou separação de classes.

O projeto de modelos evolutivos é principalmente compreendido pela construção de mecanismos de aprendizado com o intuito de induzir a geração de novo conhecimento sem, no entanto, perder o conhecimento "do passado" (esquecimento catastrófico); além disto, modelos evolutivos tentam refinar o conhecimento existente. O problema pode ser sumarizado em como acomodar novos dados no modelo de forma incremental enquanto manter o sistema em operação.

Processos de modelagem de dados muitas vezes requerem a construção e o teste do modelo de forma simultânea em um ambiente que constantemente evolui no tempo. Se um modelo estático for usado para evolução de fluxos de dados, sua precisão pode decair repentinamente quando, por exemplo, uma característica do ambiente mudar. Modelos pré-treinados *off-line* podem apresentar bom desempenho em vários contextos, entretanto eles necessitam ser re-projetados para novas circunstâncias.

Após esta introdução, o artigo prossegue na Seção II caracterizando os modelos evolutivos abordados. A Seção III mostra o comportamento dos modelos quando resolvendo um problema real de diagnóstico de falta em máquinas elétricas. O artigo conclui com a Seção IV sumarizando sua contribuição principal e sugerindo tópicos para investigações futuras.

2 Sistemas Conexionistas Evolutivos

Um sistema conexionista evolutivo é uma rede neural ou uma coleção de redes que opera continuamente no tempo e adapta sua estrutura e funcionalidades através de interações contínuas com o ambiente e com outros sistemas. Em termos gerais, um sistema conexionista $\{E, P, W, F, R, FO\}$, que é definido pela sua estrutura E, seu conjunto de parâmetros P, os pesos de suas conexões W, sua função F, um procedimento de aprendizado R, e sua função objetivo FO, aprende se o sistema otimizar ao menos parte de sua estrutura E e função F após a observação dos eventos ψ_1, ψ_2, ..., de um problema no espaço Ψ. Através do processo de aprendizado, o sistema melhora sua reação aos eventos observados e captura informações que podem posteriormente representar conhecimento.

ECOS constituem um pequeno subconjunto de todos os modelos conexionistas no contexto da inteligência computacional. Eles evoluem tanto no espaço de dados do problema, como no próprio espaço do sistema. Eles aprendem em qualquer modo de aprendizado: supervisionado, não-supervisionado, por reforço ou por uma combinação destes. Basicamente, o aprendizado em ECOS consiste nas seguintes etapas:

∗ Aquisição de dados;
∗ Pré-processamento e avaliação de características;
∗ Modelagem conexionista;
∗ Aquisição de conhecimento.

Em geral, suas principais características incluem: ECOS

∗ Evoluem em domínios abertos;
∗ Aprendem continuamente durante toda a sua existência;
∗ Usam aprendizado construtivo e possuem estruturas evolutivas;
∗ Dividem o espaço do problema permitindo uma rápida adaptação.

Nas próximas seções são apresentados fundamentos e propostas de algoritmos de aprendizado dentro deste contexto.

2.1 Medição de Distância

A medição de distância é um passo fundamental em métodos de clusterização, quantização e prototipação (Kasabov, 2007). A distância entre dois pontos no espaço Euclidiano $\Re^n$ é geralmente dada pela distância Euclidiana (norma-2), que é uma generalização do teorema de Pitágoras para mais de duas coordenadas e também a idéia intuitiva de distância. Outras distâncias, baseadas em outras normas são algumas vezes consideradas.

Formalmente, para um ponto (x_1, ..., x_n) e um ponto (z_1, ..., z_n), a distância de Minkowski D de ordem p (norma-p) é definida:

$$D = (\sum_{i=1}^{n} |x_i - z_i|^p)^{\frac{1}{p}},\ p \geq 1.$$

No caso especial em que $p = \infty$ (norma-infinito) tem-se:

$$\begin{aligned} D &= lim_{p\to\infty}(\sum_{i=1}^{n} |x_i - z_i|^p)^{\frac{1}{p}} = \\ &= max(|x_i - z_i|, ..., |x_n - z_n|). \end{aligned}$$

Além da norma-2, a norma-1 (também conhecida como distância de Manhattan) e a norma-infinito (distância de Chebyshev) são as mais usadas, sendo as demais raramente consideradas. Note que p não precisa ser necessariamente um número inteiro, podendo assumir valores reais, desde que não sejam menores que 1 de acordo com o teorema da inequação do triângulo para todos os espaços Euclidianos (Deza e Deza, 2006).

2.2 Perceptron Multi-Camadas Evolutivo

O modelo eMLP foi originalmente proposto em (Kasabov, 2003). Neste artigo sugerimos uma variação do algoritmo de aprendizado eMLP que resulta uma maior eficiência da abordagem em termos de velocidade de processamento e requerimentos de memória. Além disso, o algoritmo de aprendizado sugerido apresenta melhorias que acreditamos tornar eMLP uma abordagem mais efetiva e competitiva.

eMLP é uma rede neural evolutiva que consiste de três camadas de neurônios i, j e k denominadas nesta ordem: camada de entrada, com funções lineares ou não-lineares; camada evolutiva; e camada de saída, com funções de ativação lineares saturadas. A camada evolutiva é a camada que se adapta aos dados e é também a camada a qual o aprendizado está mais relacionado. Admita um fluxo de pares de entrada/saída de dados $(x, y)^{[h]}$, $h = 1, 2, ...$, dirigido a um modelo eMLP. O nível de ativação de um neurônio ϕ^ν da coleção finita de neurônios existentes na camada evolutiva $\Phi = \{\phi^1, ..., \phi^\nu, ..., \phi^c\}$ considerando função de ativação linear é obtido a partir de:

$$A^\nu = 1 - D^\nu,$$

onde D^ν é a distância normalizada entre o vetor de entrada atual $x^{[h]} \in \Re^n$ e o vetor de pesos das conexões da entrada do neurônio ϕ^ν, $W_{ij}^\nu \in \Re^n$. A medida D^ν pode ser calculada, por exemplo, a partir da distância Euclidiana entre $x^{[h]}$ e $W_{ij}^\nu \forall \nu$.

O aprendizado eMLP é baseado na acomodação de novos exemplos de treinamento $x^{[h]}$ nos parâmetros e estrutura do modelo através do ajuste dos pesos sinápticos $W_{ij}^\nu\ \forall \nu$, e/ou da adição de novos neurônios ϕ^{c+1}, ϕ^{c+2}, ..., estendendo a coleção atual de neurônios Φ. Note que $\Phi = \emptyset$ anteriormente ao aprendizado. Em particular, o modelo eMLP gera protótipos quando o dado é suficientemente dissimilar aos protótipos existentes. Caso contrário, um refinamento de alguns protótipos existentes é conduzido. A decisão de associar novos dados a protótipos existentes ou a novos protótipos é baseada no valor limiar ρ^ν que determina a área de atuação de um neurônio ϕ^ν. Por exemplo, caso $D^\nu \leq \rho^\nu$ para algum ν, então ajusta-se os parâmetros W_{ij}^ν correspondentes; caso $D^\nu > \rho^\nu$, então cria-se ϕ^{c+1}.

Quando um neurônio ϕ^{c+1} é criado, seu respectivo vetor de pesos W_{ij}^{c+1} é ajustado exatamente para o vetor de entrada $x^{[h]}$, enquanto que seu vetor de pesos de saída W_{jk}^{c+1} é ajustado exatamente para o vetor de saída desejado $y^{[h]}$.

Um neurônio ϕ^ν é dito vencedor quando ele apresenta a maior ativação A^ν para a entrada atual $x^{[h]}$. Os pesos W_{ij}^ν do neurônio vencedor são ajustados de acordo com:

$$W_{ij}^\nu(\textbf{novo}) = W_{ij}^\nu + \eta_1(x^{[h]} - W_{ij}^\nu),$$

onde η_1 é a taxa de aprendizado 1. O vetor de pesos da saída do neurônio vencedor é ajustado de acordo com:

$$W_{jk}^\nu(\textbf{novo}) = W_{jk}^\nu + \eta_2(A^\nu E^\nu),$$

onde η_2 é a taxa de aprendizado 2 e E^ν é um vetor de erro obtido a partir de:

$$E^\nu = y^{[h]} - \bar{y}^{[h]},$$

onde $y^{[h]} \in \Re^m$ é a saída desejada e $\bar{y}^{[h]}$ é a saída estimada pela rede.

O procedimento de aprendizagem $eMLP$ é sumarizado abaixo:

INÍCIO
Definir ρ, η_1, η_2, $c = 0$;
Fazer (1)
  {Propagar $x^{[h]}$, $h = 1, 2, ...$, na rede;
   Se $h = 1$ //Primeiro par de dados
     Adicionar um neurônio ϕ^{c+1} associado a $y^{[h]}$; $c = c + 1$;
   Caso contrário
     {Se ($D^\nu > \rho^\nu$, $\nu = 1, ..., c$, ou o neurônio vencedor ϕ^ν
      não é associado a $y^{[h]}$, $h = 1, 2, ...$)
        Criar ϕ^{c+1} e associá-lo a $y^{[h]}$; $c = c + 1$;
      Caso contrário
        Atualizar W_{ij}^ν e W_{jk}^ν de ϕ^ν; } }
FIM

2.3 O Operador Agregação

A agregação de neurônios da camada evolutiva de redes eMLP é um mecanismo sugerido para manter controle sobre a quantidade de neurônios se desenvolvendo durante o processo de aprendizado. Basicamente, este operador agrega dois neurônios se desenvolvendo de forma similar em um único neurônio. O operador de agregação pode ser aplicado em todas as iterações do algoritmo de aprendizado ou após um certo número de iterações. Em geral, este operador melhora a capacidade de generalização da rede e possibilita manter uma quantidade razoável de informação ativa na memória.

Isto pode facilitar a interpretação da rede por parte de especialistas. O procedimento de agregação é sumarizado como segue:

INÍCIO
Fazer para ϕ^{ν}, $\nu = 1, ..., c$;
{Encontrar subconjuntos de neurônios $R \in N^m$ tais que as distâncias entre dois neurônios ϕ^{r1}, $\phi^{r2} \in R$, $D(W_{ij}^{r1}, W_{ij}^{r2})$ e $D(W_{jk}^{r1}, W_{jk}^{r2})$, sejam inferiores a um limiar W_{th};
Mesclar neurônios de R em ϕ^{c+1} com parâmetros:

$$W_{ij}^{c+1} = \frac{1}{m-1} \sum_{i=1}^{m-1} D(W_{ij}^{r_i}, W_{ij}^{r_{i+1}}),$$

$$W_{jk}^{c+1} = \frac{1}{m-1} \sum_{i=1}^{m-1} D(W_{jk}^{r_i}, W_{ik}^{r_{i+1}});$$

Excluir todos os neurônios em R; }
FIM

A agregação de neurônios é uma regularização importante que pode trazer melhorias de desempenho de uma rede eMLP em certas aplicações.

2.4 Método de Clusterização Evolutiva - ECM

ECM é um algoritmo incremental de um passo usado para clusterização dinâmica de dados onde não há um número pré-definido de clusters (Kasabov e Song, 2002). ECM se baseia na distância entre novos exemplos e os centros dos clusters existentes para evoluir o modelo do processo.

Inicialmente, ECM possui uma coleção de clusters vazia. O aprendizado se inicia após a chegada do primeiro dado de um fluxo. Vetores $x^{[h]}$, $h = 1, 2, ...$, são determinados mais próximos do centro de um cluster C^{ν} da coleção atual de clusters $C = [C^1, ..., C^{\nu}, ...C^c]$ a partir de alguma medida de distância D^{ν}. Caso esta distância seja maior que um limiar ρ^{ν} pré-definido, cria-se um novo cluster C^{c+1} aumentando a coleção de clusters existentes. O centro de C^{c+1} é denominado Cc^{c+1} e seu raio Rd^{c+1} é definido inicialmente como sendo zero. Por outro lado, caso D^{ν} seja menor que ρ^{ν} para algum $\nu \in [1, ..., c]$, então o centro Cc^{ν} e o raio Rd^{ν} são ajustados no sentido de deslocar o cluster no domínio de entrada e/ou aumentar seu raio de alcance. Qual e quanto um cluster será modificado, depende do exemplo atual e dos clusters existentes. O raio de um cluster Rd^{ν} não é mais atualizado quando ele atinge o limiar máximo de expansão ρ^{ν}.

A Fig. 1 ilustra um exemplo de clusterização ECM onde são modelados o fluxo de dados $x^{[h]}$, $h = 1, ..., 9$. Note que:

* $x^{[1]}$ causa a criação de C^1,
* $x^{[2]}$ atualiza C^1,
* $x^{[3]}$ produz C^2,
* $x^{[4]}$ é descartado,
* $x^{[5]}$ atualiza C^1,
* $x^{[6]}$ é descartado,
* $x^{[7]}$ causa o ajuste de C^2,
* $x^{[8]}$ gera C^3,
* $x^{[9]}$ causa o ajuste de C^1.

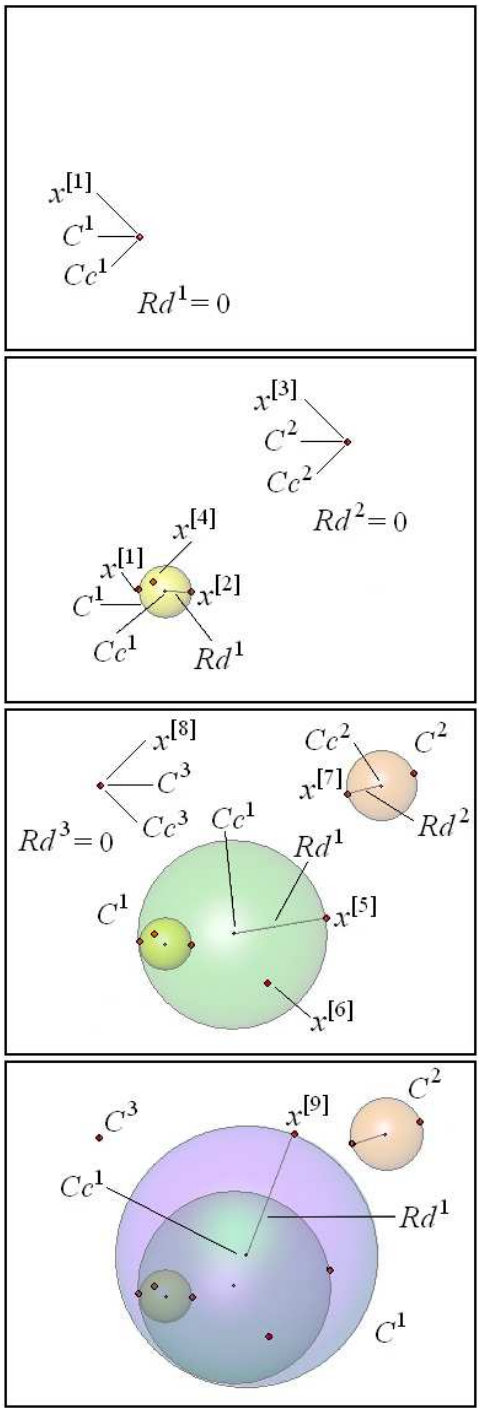

Figura 1: Processo de clusterização evolutiva

Em ECOS, os centros de clusters ECM são representados pelos pesos das conexões dos neurônios, enquanto os raios dos clusters são representados pelos limiares ρ^{ν} $\forall\nu$. O ECM é o princípio que rege o funcionamento de eSOM. Em eSOM, clusters são representados por neurônios no espaço de saída. Em contraste, neurônios eSOM possuem relação de vizinhança. Isto os oferecem parâmetros adicionais e flexibilidade ao aprendizado.

2.5 Mapa Auto-organizável Evolutivo

O modelo eSOM foi originalmente proposto em (Da Deng e Kasabov, 2000). Sugerimos neste artigo uma variação de algoritmo de aprendizado eSOM mais objetiva e efetiva em termos de tempo de processamento e aplicabilidade em tempo real.

eSOM desenvolve os princípios de mapeamento de SOM, porém nenhuma restrição topológica é estabelecida *a priori* ao aprendizado. eSOM consiste de duas camadas i (entrada) e j (saída), sendo a última a camada que evolui de acordo com o fluxo de dados. eSOM permite a evolução dos neurônios no espaço de dados original e ao mesmo tempo desenvolve e mantém uma representação topológica. Em contraste ao SOM, a vizinhança dos neurônios evoluídos não é pré-definida. Ela é definida durante o aprendizado de acordo com a distância entre os neurônios existentes.

Inicialmente admite-se um mapa nulo. Gradualmente, neurônios são gerados a partir de novas informações contidas no fluxo de dados. Caso um novo exemplo seja dissimilar a um certo grau dos neurônios existentes, um novo neurônio é criado. Por outro lado, caso o exemplo seja relacionado com um neurônio existente, então refina-se os parâmetros associados. Formalmente, a partir de um fluxo de dados $x^{[h]}$, $h = 1, 2, ...$, a ativação do neurônio ϕ^ν da coleção de neurônios existentes $\Phi = \{\phi^1, ..., \phi^\nu, ..., \phi^c\}$ é dada por:

$$A^\nu = e^{(\frac{-D(x^{[h]} - W_{ij}^\nu)^p}{\epsilon^p})},$$

onde ϵ é um raio em torno de W_{ij}^ν e p é a ordem da medida de distância D. A seguinte função para minimização do erro de aproximação estocástico *on-line* é admitida:

$$\zeta^\nu = \sum_{i=1}^{c} A^\nu D(x^{[h]} - W_{ij}^\nu),$$

onde c é o número de neurônios do $eSOM$ antes da chegada de $x^{[h]}$. Para minimizar a função acima, os vetores $W_{ij}^\nu \forall \nu$, são atualizados pelo método do gradiente descendente. Assumindo $p = 2$ temos:

$$\frac{\partial \zeta^\nu}{\partial W_{ij}^\nu} = A^\nu (W_{ij}^\nu - x^{[h]}) + D(x^{[h]} - W_{ij}^\nu)^2 \frac{\partial A^\nu}{\partial W_{ij}^\nu}.$$

Por praticidade, assumimos que a variação da ativação será menor a cada vez que os pesos forem ajustados. Assim, A^ν, $\nu = 1, ..., c$, são constantes e a seguinte relação para ajuste de pesos surge:

$$\Delta W_{ij}^\nu = \eta A^\nu (x^{[h]} - W_{ij}^\nu),$$

onde η é a taxa de aprendizado. A probabilidade de que um exemplo $x^{[h]}$ seja associado ao ν-ésimo neurônio W_{ij}^ν é dada por:

$$P_i(x^{[h]}, W_{ij}^\nu) = A^\nu (\sum_{\nu=1}^{c} A^\nu)^{-1}.$$

Durante o aprendizado, caso neurônios se tornem inativos por um longo tempo, estes são removidos do mapa. O procedimento de modelagem $eSOM$ é sumarizado como segue:

INÍCIO
Definir ρ, η, Δ, $c = 0$;
Fazer (1)
{Propagar $x^{[h]}$, $h = 1, 2, ...$, na rede;
Encontrar um conjunto de neurônios S que são próximos a $x^{[h]}$ a um grau pré-determinado ρ;
Se S é nulo
Criar um novo neurônio ϕ^{c+1} e conectá-lo aos dois vizinhos mais próximos, formando um conjunto S;
Calcular a ativação A^ν e ajustar W_{ij}^ν dos neurônios em S;
Recalcular as conexões entre neurônios a partir de alguma medida de distância;
Após $h + \Delta$ iterações, excluir as conexões mais fracas e/ou neurônios isolados; }
FIM

---

3 Detecção de Falta em Máquinas Elétricas

3.1 Descrição Básica do Problema

Atualmente, máquinas elétricas estão entre os mais importantes equipamentos industriais. Geralmente, elas são componentes críticos em processos de automação industrial requerendo certos níveis de confiabilidade, segurança, eficiência e performance. Por estas razões, questões relacionadas à proteção destas máquinas contra faltas têm recebido considerável atenção por engenheiros de manutenção e especialistas.

Sistemas de detecção de falta têm sido desenvolvidos e aperfeiçoados para o propósito de monitoramento das condições de operação de máquinas elétricas (Leite et al., 2007) (Nandi e Toliyat, 2005). Estes sistemas podem trazer diversos benefícios às industrias relativos à redução de períodos de parada e de desconexões desnecessárias da planta, agendamento da disponibilidade de pessoal para manutenção, redução de custos de reparo e de peças em estoque, minimização de perdas em produção, entre outros.

Em particular, um dos tipos de faltas mais comuns em motores de indução é o curto-circuito entre espiras dos enrolamentos do estator. Esta falta primária acontece após o rompimento do isolamento entre espiras. Após a falta, o processo de degradação da máquina se intensifica e falhas mais danosas podem surgir, resultando em danos irreversíveis ao motor. Entretanto, se a falta de espira para espira for detectada por sistemas de diagnóstico em estagio incipiente, por exemplo, o enrolamento da fase faltosa pode ser substituído, reduzindo significantemente perdas financeiras e aumentando a segurança operacional.

3.2 Características da base de dados

∗ Número de exemplos (observações): 3500;
∗ Número de atributos: 13 variáveis de entrada quantitativas denominadas tensões abc, correntes

Tabela 1: Detecções corretas de falta em máquinas elétricas

Modelo	No. protótipos	Épocas de treinamento	Tempo requerido (s)	% Precisão (Melhor)	% Precisão (Média)
MLP	[13-35-35-3]	10000	821,7	94,8	92,3
Elman	[13-35-35-3]	10000	1280,5	94,8	93,6
SOM	[13-70]	5000	350,0	86,1	81,9
FCM	80	56	16,8	75,0	72,7
$eMLP_1$	54	1	0,36	98,5	96,3
$eMLP_2$	69	1	0,50	99,1	96,6
$eSOM_1$	43	1	0,20	97,8	95,1
$eSOM_2$	60	1	0,26	98,4	96,3

abc do estator, defasamentos tensão/corrente abc, potência ativa, deslize do rotor, valor absoluto da corrente de seqüência negativa, e ângulo da corrente de seqüência positiva; e 3 variáveis de saída nomeadas porcentagem de espiras em curto-circuito nos enrolamentos abc do estator;
$*$ Dados de treinamento/teste: 60/100%.

Para comparação de performance considerou-se os seguintes modelos: redes neurais Perceptron Multi-Camadas (MLP) e Elman, Mapa Auto-Organizável (SOM), Fuzzy C-Means (FCM), além de eMLP e eSOM.

3.3 Resultados

Para avaliar o resultado de diferentes parametrizações, as abordagens eMLP e eSOM consideraram dois limiares de alcance dos neurônios que são $\rho_1 = 0.3$ e $\rho_2 = 0.2$. Cada experimento foi realizado cinco vezes com os mesmos parâmetros. Em cada experimento, os dados foram misturados para induzir diferentes ordens no fluxo de dados. Os dados foram apresentados aos modelos seqüencialmente apenas uma vez para emular um fluxo. eMLP e eSOM começam a aprender com uma base de regras vazia e sem pré-treinamento. Isto é feito para simular um processo evolutivo real. A Tabela 1 mostra o desempenho dos modelos considerados.

Os resultados da Tabela 1 mostram que o modelo eMLP sugerido foi o mais preciso para a classificação de padrões de falta em máquinas elétricas. Usando uma estrutura compacta, 43 protótipos, o modelo eSOM proposto atingiu a performance de $97,8\%$. eSOM se apresentou como a alternativa mais rápida e com menos requerimentos de memória para a modelagem de processos evolutivos. Note que eMLP e eSOM usam algoritmos de um passo, o que naturalmente apela por suas aplicações na resolução de problemas em tempo real que demandam adaptabilidade incremental.

4 Conclusão

Variações evolutivas de algoritmos de aprendizado de sistemas conexionistas foram sugeridas. O aprendizado evolutivo provê redes neurais com habilidade de modelagem de processos adaptativos em tempo real a partir de fluxos de dados e algoritmos incrementais. Experimentos em detecção de falta em máquinas elétricas têm confirmado a efetividade da abordagem evolutiva quando comparada a técnicas não-lineares de aprendizado de máquina. Trabalhos futuros incluem o desenvolvimento de algoritmos evolutivos para induzir o aprendizado de modelos baseados em novas estruturas conexionistas e em grânulos de informação. Aplicações em séries temporais e controle são também direções de desenvolvimento.

Agradecimento

O primeiro autor agradece a CAPES pelo apoio financeiro. O segundo autor é grato a CEMIG pelo suporte P&D178. O último autor agradece ao CNPq pelo suporte 304857/2006-8.

Referências

Bouchachia, A.; Gabrys, B.; Sahel, Z. (2007). "Overview of some incremental learning algorithms". *IEEE Fuzzy Systems Conf.*, pp: 1-6.

Da Deng; Kasabov, N. (2000). "ESOM: an algorithm to evolve self-organizing maps from online data streams". *IEEE International Joint Conference on Neural Networks*, Vol. 6, pp: 3-8.

Deza, M.; Deza, E. (2006). Dictionary of Distances. Elsevier Science, 1^a edição, pp: 412p.

Fayyad, U.; Uthurusamy, R. (1996). "Data mining and knowledge discovery in databases". *Communications ACM*, Vol. 39, pp: 24-27.

Inmon, W. H. (1996). "The data warehouse and data mining". *Communications ACM*, Vol. 39, pp: 49-50.

Kasabov, N.; Song, Q. (2002). "DENFIS: Dynamic evolving neuro-fuzzy inference system and its application for time-series prediction". *IEEE Tran. on Fuzzy Systems*, Vol. 10-2, pp: 144-154.

Kasabov, N. (2003). Evolving Connectionist Systems. Springer Verlag, Londres, 1^a ed. pp: 307p.

Kasabov, N. (2007). Evolving Connectionist Systems: The Knowledge Engineering Approach. Springer Verlag, Londres, 2^a edição, pp: 465p.

Leite, D. F.; et al. (2007). "Real-Time Model-Based Fault Detection and Diagnosis for Alternators and Induction Motors". *IEEE Electric Machines and Drives Conf.*, Vol. 1, pp: 202-207.

Mitra, S.; Pal, S. K.; Mitra, P. (2002). "Data mining in soft computing framework: a survey". *IEEE Tran. on Neural Nets.*, Vol. 13-1, pp: 3-14.

Nandi, S.; Toliyat, A. (2005). "Condition Monitoring and Fault Diagnosis of Electrical Motors - A Review". *IEEE Transaction on Energy Conversion*, Vol. 20, No. 4, pp: 719-729