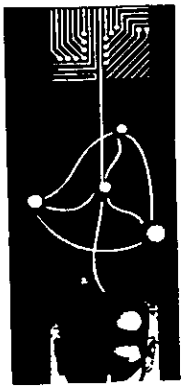

1° Congresso Brasileiro de Redes Neurais

Escola Federal de Engenharia de Itajubá
Itajubá, 24 a 27 de outubro de 1994

UM ACELERADOR GENÉTICO PARA REDES NEURONAIS

Antonius H. M. de Knegt
Mestrado em Eng. Elétrica
CPGEE - UFMG
D.E.T. / USIMINAS

Evandro de O. Araújo
Departamento de Eletrônica
Escola de Engenharia da UFMG
e-mail: earaujo@brufmg.bitnet

Gutemberg de S. Medeiros
Estudante de Eng. Elétrica
UFMG

Resumo

Este artigo apresenta um estudo sobre a utilização de Redes Neuronais em Controle de Processos analisando a viabilidade do emprego dos algoritmos genéticos para acelerar a convergência das redes. O princípio de funcionamento dos algoritmos genéticos é descrito. Resultados da utilização destes algoritmos com uma rede backpropagation para o cálculo da abertura entre os cilindros de um laminador são apresentados. A redução obtida no tempo de processamento desta aplicação nos indica um caminho bastante promissor a ser explorado.

1 Introdução

Estudos de sistemas computacionais inspirados em modelos biológicos remontam aos anos 40 (McCulloch, 1943; Hebb, 1949). Recentemente, a partir dos anos 80, o interesse por estes sistemas conexionistas vem crescendo de maneira bastante sólida, fruto de uma melhor compreensão de suas potencialidades e limitações. O surgimento de novas estruturas para estas redes (HOPFIELD, 1982; LIPMANN, 1987), avanços significativos nos algoritmos de aprendizado e a disponibilidade de processadores cada vez mais rápidos são os fatores que mais contribuiram para a popularização das redes neuronais artificiais. A computação neuronal representa uma maneira efetiva de se tentar incorporar na máquina algumas das potencialidades do cérebro humano tais como aprendizagem com a experiência, raciocínio de senso comum, a capacidade de generalizar e tomar decisões num ambiente de conhecimento incompleto, incerto e impreciso. Uma rede neuronal do tipo "backpropagation" pode ser usada em conjunto com um controlador nebuloso para reduzir os tempos de processamento e atenuar os problemas de memória computacional necessária à aplicação. Uma vez treinada a rede, é muito mais rápido obter os resultados com ela do que consultar uma "look-up table" que pode ser muito grande se o nível de quantização for grande e/ou a partição nebulosa fina (LEE, 1990). As redes neuronais têm também um aspecto de adaptabilidade inerente muito forte, tornando

bastante atraente a utilização destas em sistemas adaptativos. A rede, contudo, padece de alguns inconvenientes. O principal, a nosso ver, é a insegurança quanto à convergência. Os algoritmos genéticos são algoritmos de procura inspirados em modelos genéticos e têm sido pesquisados desde os anos 70 (Holland, 1975). Eles executam uma busca focalizada no espaço de soluções do problema através da acumulação de conhecimento durante a procura, conseguindo assim rapidez e economia de espaço de memória. Eles podem ser aplicados a uma ampla gama de problemas de otimização como, por exemplo, ao problema de seqüenciamento ótimo que se torna rapidamente intratável quando a dimensão cresce. A idéia de integração de um módulo genético com uma rede neuronal aparece como um caminho natural para tentar combinar as boas características destas duas técnicas, como será explicado na seção 4.

2 Redes Neuronais Artificiais

Uma rede neuronal artificial tal como um sistema nebuloso pode ser vista como estimadores sem modelo (KOSKO, 1991). Neste sentido, são aproximadores universais de funções bastante gerais, capazes de mapear vetores de entrada X em vetores de saída Y sem que seja necessária a utilização de um modelo matemático. Uma unidade básica de uma rede neuronal é um neurônio. A figura 1 mostra um neurônio constituído de N entradas ao qual foi acrescentado uma entrada adicional X_0 (polarização), para maior flexibilidade da rede, de valor 1.

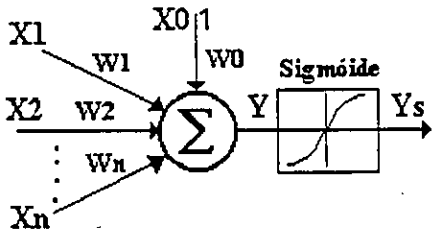

Figura 1

Na figura acima, $[W_0 \ldots W_N]$ é o vetor de pesos. Representando por **sgm(.)** uma função de ativação do neurônio, a operação realizada pelo neurônio sobre um vetor de entradas pode ser expressa por:

$$y_s = sgm\left(\sum_{k=0}^{N} X_k W_k\right)$$

Optamos, neste trabalho, por uma função de ativação monotônica crescente e derivável em qualquer ponto da forma:

$$y_s = \frac{1}{1+e^{-y}}$$

Os neurônios podem ser interligados, as saídas de uns compondo as entradas de outros, formando diversos níveis ou camadas. Utilizamos, aqui, estruturas neuronais não realimentadas ("feedforward") gerando por isto sempre sistemas estáveis.

2.1 Treinamento das redes neuronais

Treinar uma rede neuronal é encontrar um vetor **W** de pesos que possibilite a rede emular uma função qualquer, com uma precisão estipulada. A rede é treinada utilizando-se subconjuntos de dados, extraídos judiciosamente dos universos de entrada e saída. Após o treinamento, a rede é capaz de "generalizar", isto é, capaz de gerar valores aproximadamente corretos para um vetor de entradas arbitrário que não faz parte do subconjunto de treinamento. As redes neuronais podem ser treinadas através de um processo *supervisionado* ou *não supervisionado*. No primeiro método, os valores de saída da rede para cada apresentação das entradas são comparados com a correspondente saída desejada, gerando um vetor de erro que é então utilizado como orientador do processo de treinamento dos pesos. No caso do treinamento não supervisionado as entradas são agrupadas em classes, usualmente por similaridade ou

proximidade geométrica dos vetores de entrada. Os padrões podem ser definidos antes do início do treinamento ou a própria rede pode criar estas classes durante o processo de classificação.

2.2 A rede backpropagation

Trabalhamos com uma rede "feedforward" de 3 camadas (uma camada oculta) usando o modo de treinamento supervisionado. O esquema de aprendizado, chamado de "error backpropagation", baseia-se na minimização do gradiente do erro quadrático entre a saída desejada e a saída calculada pela rede. O treinamento no método backpropagation começa com a apresentação do vetor de entradas **X** à camada de entrada da rede. Este vetor é propagado através da rede gerando o vetor de saída **Y** e um vetor erro é calculado a partir do valor da saída desejada . Este erro é propagado para trás ("backpropagation") em direção à camada de entrada. A adaptação dos pesos se dá em função da contribuição das diversas sinapses para este erro (FREEMAN, 1991). O valor do passo de aprendizado μ, usualmente em torno de 0.25, determina a rapidez e a estabilidade da convergência. Um valor de μ muito pequeno torna muito lento o treinamento, enquanto um valor muito grande pode, mesmo, impedir a convergência. Uma boa estratégia é começar com um valor alto para μ e reduzi-lo à medida que o treinamento avança. O tempo necessário para o treinamento da rede depende fortemente dos pesos iniciais utilizados. Na seção 4, descreveremos um método para acelerar o treinamento, usando um algoritmo genético.

2.3 Redes Neuronais no Controle de Processos

As redes neuronais como os controladores nebulosos podem ser usados para representar o mapeamento entrada/saída de processos e desta forma constituem uma técnica bastante simples e eficaz de controle de processos não lineares complexos. O tempo de processamento é sempre um fator limitante em aplicações de tempo-real. Nos controladores nebulosos, este problema pode ser atenuado fazendo-se uso das tabelas de consulta ("look-up table"). Quando o número de antecedentes das regras e a partição nebulosa é muito fina, estas tabelas se tornam muito grandes, ocupando um excessivo espaço de memória. As redes neuronais, uma vez treinadas, fornecem de imediato a resposta para o sistema com uma exigência mínima de memória. Estas duas características, rapidez e pouca memória requerida, tornam as redes neuronais uma opção interessante para uma aplicação como os controladores nebulosos de tráfego (ARAÚJO, 1994). Nesta aplicação, procura-se determinar a extensão do tempo de verde da via com direito de passagem em função da taxa de chegada de veículos nesta via e da fila que se forma na outra via. A partir de dados gerados pelo controlador nebuloso de tráfego, uma rede neuronal backpropagation é treinada . As entradas são *Chegadas* e *Fila* de veículos. A saída é a *Extensão* de tempo do sinal verde. Depois de treinada, a rede pode substituir o módulo de inferência nebulosa. Os resultados que obtivemos foram muito bons. Os tempos de processamento foram bastante reduzidos e a fase de treinamento foi relativamente rápida para o conjunto de dados utilizados.

Numa outra aplicação recente (PATARO, 1993), uma rede neuronal backpropagation é utilizada para o cálculo da abertura entre os cilindros de um laminador. As entradas são a carga prevista, a largura da chapa e a espessura desejada. A saída da rede é a abertura entre os cilindros. A utilização da rede simplificou sobremaneira o cálculo da

abertura, tornando desnecessária a obtenção do módulo de rigidez.

3 Algoritmos Genéticos

São algoritmos de procura de soluções ótimas de finalidade geral. Eles utilizam mecanismos similares aos que regem a evolução das populações de seres vivos [GREFENSTETTE, 1993]. A rede neuronal backpropagation funciona bem desde que o espaço de solução não seja muito convoluído, pois neste caso existirão muitos mínimos locais, os quais retardam ou paralisam o processo de obtenção da solução ótima global. Os algoritmos genéticos não envolvem minimização de gradiente. Eles não são, pois, afetados por mínimos locais e constituem-se numa alternativa atraente para solução desta classe de problemas. Apresentaremos, abaixo, um método para acelerar a fase de treinamento das redes neuronais usando um algoritmo genético para inicializar os pesos da rede.

3.1 Descrição do método

Geramos aleatoriamente "**n**" vetores de pesos dentro de um intervalo onde supomos estar a solução para o problema. O conjunto de pares de teste entrada/saída é então aplicado à rede com cada um dos "**n**" vetores de pesos, e o erro quadrático é armazenado em cada caso. Os vetores de pesos são dispostos em ordem crescente do erro associado. Os 25% melhores vetores são duplicados, os 25% piores são eliminados e os 50% restantes, mantidos. O total dos vetores permanece o mesmo mas os 25% melhores dobraram sua incidência. Em seguida, algumas partes dos vetores são modificadas em um processo conhecido como "**crossover**". O número de vetores submetidos ao "crossover" é um parâmetro arbitrado, não existindo uma regra geral para definição de seu valor. Em nosso trabalho, trocamos dois pesos em posições aleatórias dentro de cada vetor [JANSON, 1993]. A fase final, chamada de "**mutação**", consiste em substituir, aleatoriamente, dentro da matriz de pesos alguns de seus valores. A taxa de mutação é também um parâmetro arbitrário. O processo de mutação injeta informação nova na população, o que é conveniente, pois não existe garantia de que a solução do problema esteja no universo dos pesos vigente. Por ser aleatória, a mutação pode também destruir um bom vetor de pesos antes que este possa ser duplicado. Na prática, observa-se que uma taxa elevada de mutação provoca uma oscilação nos valores dos erros a cada geração. O processamento genético dos dados a cada geração envolve as segintes fases:

- gerar aleatoriamente uma matriz de pesos dentro de um intervalo fixado;
- avaliar o desempenho de cada linha da matriz de pesos;
- duplicar as melhores e eliminar as piores.
- executar o "crossover";
- aplicar a mutação.

Repete-se o processo do passo 2 ao passo 5 até atingir o número fixado de gerações ou a precisão estipulada.

4 Um acelerador genético para a rede backpropagation

Testes realizados com o algoritmo genético mostraram que nos primeiros ciclos a adaptação é mais rápida do que no método backpropagation. O tempo gasto a cada ciclo no caso do genético é constante e corresponde aproximadamente ao produto de três fatores: o tempo gasto para uma

apresentação, o número de padrões de teste e o número de linhas da matriz de pesos. Após um avanço inicial rápido, o algoritmo genético se torna muito lento com melhorias ocorrendo após longos intervalos e devidas ao processo de mutação. No backpropagation, os tempos necessários para treinamento de cada par são variáveis, muito longos no início, decrescendo exponencialmente com a evolução do treinamento dos pesos. Estes fatos sugerem a aplicação de um método misto de treinamento no qual o método genético é utilizado nos primeiros ciclos, agilizando assim o treinamento da rede backpropagation. Em nossa aplicação, os pesos obtidos após 10 ciclos do algoritmo genético foram passados para o algoritmo backpropagation. Os resultados são mostrados, a seguir.

5 Resultados

O treinamento de uma rede backpropagation para o problema do controle de tráfego foi relativamente rápido. Julgamos mais interessante, para avaliar o desempenho do algoritmo genético, a aplicação da rede backpropagation para o cálculo da abertura entre os cilindros de um laminador. Os dados foram extraídos de [PATARO, 1993]. Fixamos a taxa de mutação em 0,5 e o passo de aprendizado em 0,25 e realizamos uma série de simulações para avaliar o ganho em se acoplar um algoritmo genético a uma rede backpropagation. Os resultados tabelados abaixo indicam o bom desempenho do algoritmo misto (genético e neuronal). Em todas as simulações realizadas houve uma redução significativa do número de iterações e do tempo de processamento. Outras aplicações vão ser utilizadas visando uma melhor compreensão da influência dos diversos parâmetros.

Série	Algoritmo misto	Backpropagation
1	7229 iterações	46484 iterações
2	9354 iterações	40481 iterações
3	21212 iterações	48530 iterações
4	23007 iterações	45450 iterações
5	2156 iterações	38649 iterações

Tabela 1

6 Conclusões

Os bons resultados obtidos com a utilização dos algoritmos genéticos para obtenção dos pesos iniciais de uma rede backpropagation, embora não definitivos, representam um passo importante para a viabilização das redes neuronais em aplicações tempo-real. As potencialidades dos algoritmos genéticos merecem ser mais cuidadosamente exploradas. O número de gerações e a taxa de mutação são apenas alguns dos parâmetros cuja influência deve ser investigada.

7 Referências Bibliográficas

FREEMAN, 1991 "Neural Networks". FREEMAN, J.A & SKAPURA, D.M. Addison Wesley, 1991.

GREFENSTETTE, 1993 "A Brief Tutorial". GREFENSTETTE, J.J. IEEE Expert, pp 6 - 9, 10/93.

HEBB, D.O., 1949 "The Organization of Behaviour". HEBB, D.O. John Wiley, N.Y., 1949.

HOLLAND, 1975 "Adaptation in Natural and Artificial Systems". HOLLAND, J. H. MIT Press, Cambridge, MA, 1975.

HOPFIELD, 1982 "Neural Networks and Physical Systems with emergent collective computational abilities". HOPFIELD, J.J. Proc. of the Nat. Acad. Sciences, Vol. 79, pp. 2554 - 2558, 1982.

JANSON, 1993 "Training Product Unit Neural Networks with Genetic Algorithms". JANSON, D. J. & FRENZEL, J. F. IEEE Expert, pp 26 - 33, 10/93.

KOSKO, 1992 "Neural Networks and Fuzzy Systems". KOSKO, B. Prentice Hall, 1992.

LEE, 1990 "Fuzzy Logic in Control Systems: Fuzzy Logic Controller". LEE, C.C. IEEE Transactions on Systems, Man and Cybernetics vol. 20, pp 419-434, 1990.

LIPPMAN, 1987 "An Introduction to Computing with Neural Nets". LIPPMANN, R.P. IEEE ASSP Magazine, pp. 4-22, 04/87.

McCULLOCH, 1943 "A Logical calculus of the ideas immanent in nervous activity". McCULLOCH, W.S. & PITTS. W. Bulletin of Math. Biophysics, vol. 4, pp 115-133, 1943.

PATARO, 1993 "Posicionamento Automático de Cilindros em Laminadores Empregando Treinamento com Redes Neuronais". PATARO, C. D. M. & SCHMIDT, W. G. & RESENDE, P. & HELMAN, H. III Seminário de Automação na Indústria, ABM, 1993.