UTILIZAÇÃO DE UM NEURÔNIO McCULLOCH E PITTS PARA SIMULAÇÃO DE PORTAS LÓGICAS ATRAVÉS DE APRENDIZADO SUPERVISIONADO

Diego SOUZA (1); Bosco MOREIRA (2); Carlos FONSECA (3)

(1) CEFET-RN, Av. Sen. Salgado Filho, 1559, Tirol, Natal-RN – CEP: 59015-000, (84)88858027, fax, e-mail: diegouern@gmail.com
(2) UERN, e-mail: boskyn9@gmail.com
(3) UERN, e-mail: carlosandre@uern.br

RESUMO

As Redes Neurais Artificiais (RNAs) surgiram como uma nova maneira de processar informações. Baseadas no funcionamento do cérebro humano, elas processam informações de maneira paralela. As RNAs têm a habilidade de aprender e generalizar. Devido a esses fatores elas têm grande emprego na área de reconhecimento de padrões, aproximação de funções, controle de processos, entre outras. Neste trabalho foi desenvolvido um neurônio de McCulloch e Pitts (McP), que é um neurônio com peso, capaz de aprender e simular o funcionamento de qualquer uma das portas-lógicas linearmente separáveis (AND, OR, NAND, NOR). O sistema implementado oferece uma interface gráfica completamente personalizável, onde o usuário pode configurar as variáveis iniciais do neurônio de McP. Para a aprendizagem do neurônio são fornecidos pares de treinamento contendo entradas e saídas associadas às entradas, dessa forma, a cada par de entradas apresentadas ao neurônio o programa realiza o cálculo do erro cometido pelo neurônio. A partir do erro e dos sinais de entrada os pesos são reajustados, de tal forma que, ao final da fase de aprendizagem, o neurônio consegue fornecer a resposta correta para cada entrada. Após ter os pesos ajustados, o programa permite ao usuário testar o funcionamento do neurônio, para tanto o usuário fornece pares de entradas para o neurônio e a cada par de entradas fornecido o neurônio responde com a saída. O aplicativo é inteiramente Java e salva seu aprendizado em arquivos de texto. O sistema se reajusta tornando possível emular perfeitamente a saída da porta lógica escolhida, tornado-se um sistema que se mostra dotado de inteligência artificial.

Palavras-chave: RNA, McP, portas-lógicas linearmente separáveis

1. INTRODUÇÃO

O trabalho em redes neurais artificiais, usualmente denominadas “redes neurais”, tem sido motivado pelo fato dessas estruturas terem a capacidade de aprendizado, generalização e adaptação.

Este artigo pretende fornecer uma visão geral sobre RNA, expondo conceitos e também apresentar uma aplicação de um neurônio com peso capaz de emular portas lógicas descritas por funções linearmente separáveis.

Esta aplicação visa auxiliar o ensino de redes neurais, mais especificamente de neurônios artificiais com peso, uma vez que com ela consegue-se configurar facilmente os parâmetros iniciais do neurônio e se desejado o usuário pode visualizar o comportamento do neurônio na fase de aprendizagem em cada iteração e na fase de validação do conhecimento.

2. CONCEITOS FUNDAMENTAIS

2.1. O que são redes neurais artificiais?

RNAs são sistemas paralelos distribuídos compostos por unidades de processamento simples (nodos) que calculam determinadas funções matemáticas (normalmente não-lineares). Tais unidades são dispostas em uma ou mais camadas e interligadas por um grande número de conexões, geralmente unidirecionais. Na maioria dos modelos essas conexões estão relacionadas a peso, os quais armazenam o conhecimento representado no modelo e servem para ponderar a entrada recebida por cada neurônio da rede.

As RNAs tentam reproduzir as funções das redes neurais biológicas, buscando implementar seu comportamento básico e sua dinâmica. Contudo, do ponto de vista físico, as redes artificiais se diferem bastante das redes biológicas, até a fase atual de desenvolvimento das RNAs.

2.2. Neurônio artificial

Um neurônio é uma unidade de processamento de informações fundamental para a operação de uma rede neural. O primeiro modelo artificial de um neurônio biológico foi fruto de dois pioneiros, o psiquiatra e neuroanatomista Warren McCulloch e o matemático na época, recém-graduado, Walter Pitts.

O modelo de neurônio proposto por McCulloch e Pitts é uma simplificação do que se sabia na época sobre o neurônio biológico. Sua descrição resultou em um modelo com m terminais de entrada x_1, x_2... x_m (representando os dendritos no neurônio biológico) e apenas uma saída y (representando o axônio). Para emular o comportamento das sinapses os terminais de entrada têm pesos acoplados w_{k1}, w_{k2}... w_{km} ,onde k diz respeito ao neurônio em questão, e os índices de 1 até m dizem respeito às respectivas entradas. O valor de um peso pode ser positivo ou negativo, dependendo das sinapses, se são inibitórias ou excitatórias .O efeito de uma sinapse em um neurônio pós-sináptico é dado pelo produto da entrada por seu respectivo peso, ou seja, no caso do neurônio artificial da Figura 1 o efeito de sua sinapse é dado por x_i w_i. O neurônio artificial proposto por McCulloch e Pitts possui, ainda, um combinador linear, que é um somador de todas as entradas ponderadas pelos seus respectivos pesos mais o bias, o que resulta no campo local induzido do neurônio, uma função de ativação (φ) é usada para restringir a amplitude da saída do neurônio.

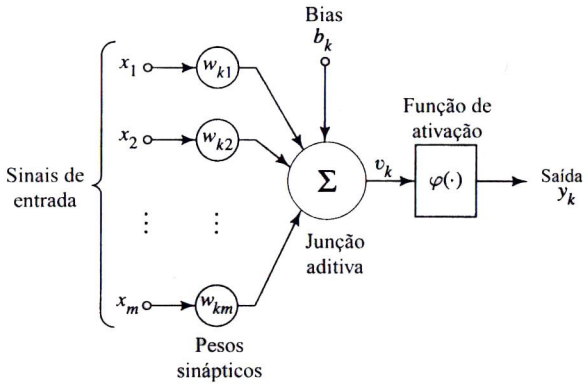

Figura 1 – Neurônio de McCulloch e Pitts

Tabela 1 – Legenda dos Símbolos Utilizados

Símbolo	**Significado**
x	Entrada
y	Saída
w	Peso
b	Bias
φ	Função de Ativação
u	Combinador Linear
η	Eta
e	Erro
v	Campo Local Induzido

Em termos matemáticos, podemos descrever um neurônio *k* escrevendo as seguintes equações:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \qquad \text{[Eq. 01]}$$

$$v_k = u_k + b_k \qquad \text{[Eq. 02]}$$

$$y_k = \varphi(v_k) \qquad \text{[Eq. 03]}$$

2.3. Função de Ativação

Trata-se de uma função matemática que recebe como entrada o campo local induzido do neurônio e fornece uma resposta. Existem diversos tipos de função de ativação, entre elas as mais utilizadas são as sigmóides, limiares e lineares. A escolha da função de ativação a ser utilizada depende da aplicação, pois é ela que irá restringir a saída do neurônio para valores utilizáveis pela aplicação.

No programa implementado, como se deseja que o neurônio aprenda o funcionamento de uma porta-lógica, utilizou-se uma função de limiar, uma vez que ela limita as saídas do neurônio em zero ou um. A saída de

um neurônio assume o valor um se o campo local induzido v_k é um valor positivo ou zero, caso contrário assumirá o valor zero.

$$\varphi(v) = \begin{cases} 1 \; se \; v \geq 0 \\ 0 \; se \; v < 0 \end{cases} \qquad \text{[Eq. 04]}$$

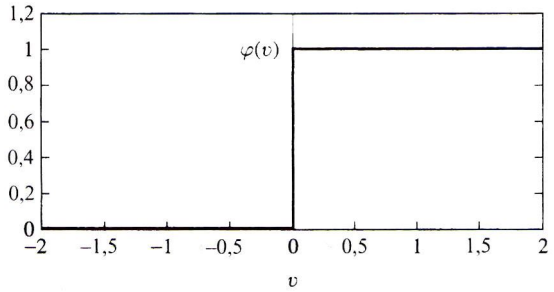

Figura 2 – Função de Ativação Limiar

2.4. Processo de Aprendizagem

A aprendizagem é um procedimento pelo qual os pesos de uma RNA são adaptados através do algoritmo de aprendizagem. Existem basicamente dois tipos de aprendizagem: supervisionada e não-supervisionada. No programa em questão foi utilizado o processo de aprendizagem com professor.

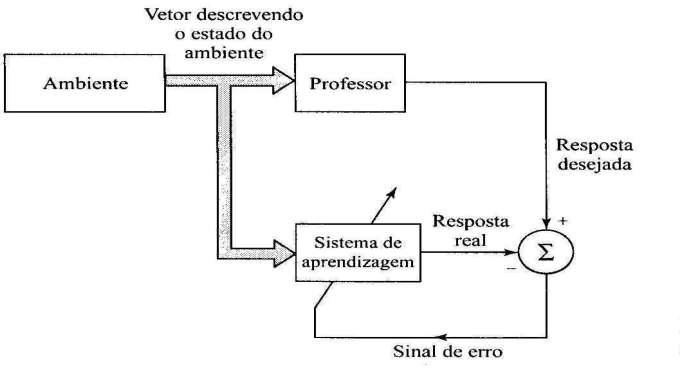

Figura 3 – Diagrama de aprendizagem com professor

2.4.1. Processo de Aprendizagem com Professor

É o tipo de aprendizado mais utilizado. Consiste em ajustar parâmetros da rede, de forma a encontrar uma ligação entre os pares entrada e saída fornecidos. Um professor indica explicitamente se as saídas dos neurônios estão adequadas. A rede tem sua saída comparada com a saída desejada gerando o sinal de erro da sua resposta atual, que será utilizado no ajuste dos pesos conforme mostrado na equação [Eq. 07]. A soma dos erros quadráticos de todas as saídas [Eq. 06] é utilizada como medidor de desempenho da rede e como condição de parada se o resultado chegar a zero. O erro, e, é a diferença entre a saída desejada e a saída obtida [Eq. 05], onde a saída obtida y_k é a soma ponderada das entradas pelos pesos [Eq. 01], somada ao bias b_k [Eq. 02], passadas pela função de ativação [Eq. 03].

$$e(j) = d(j) - y(j) \qquad \text{[Eq. 05]}$$

$$md = \sum e^2 \qquad \text{[Eq. 06]}$$

$$w_k(j+1) = w_{kj} + \eta e_j x_{kj} \quad \text{[Eq. 07]}$$

onde w_{kj} é o peso atual somado ao produto da entrada x_{kj} pela taxa de aprendizado η e o erro.

3. PROGRAMA

A motivação para o desenvolvimento deste programa adveio da necessidade de visualizar o funcionamento de um neurônio artificial, do modo como ocorre o aprendizado do neurônio, da influência que cada parâmetro tem nesta fase, para facilitar a compreensão desta técnica da inteligência artificial.

Objetivou-se, com este programa, a obtenção de uma ferramenta capaz de auxiliar o ensino de RNA, para tanto, criou-se uma interface gráfica de forma a tornar simples ao usuário a visualização do processo de aprendizagem, a influência de cada fator nesta etapa, e a validação do conhecimento de um neurônio artificial.

O Programa e dividido basicamente em duas partes: fase de aprendizagem e fase de reconhecimento. Na fase de aprendizagem, o usuário indica todas as entradas necessárias para que o aplicativo faça os cálculos e ajustes dos pesos, de tal forma que, na fase de reconhecimento o sistema possa emular a porta lógica proposta.

3.1. Interface

O programa foi desenvolvido em linguagem Java, com uma interface amigável e intuitiva, como se pode observar na Figura 4. Através da interface todos os dados do programa podem ser configurados, bem como se pode verificar o resultado de execuções anteriores. O usuário pode, também, acompanhar o desenvolvimento do aprendizado através da aba saída, controlando a velocidade de execução para uma melhor análise dos cálculos e desenvolvimento da situação. Após o aprendizado, o usuário pode verificar a eficiência do programa através da tela de simulação Figura 5.

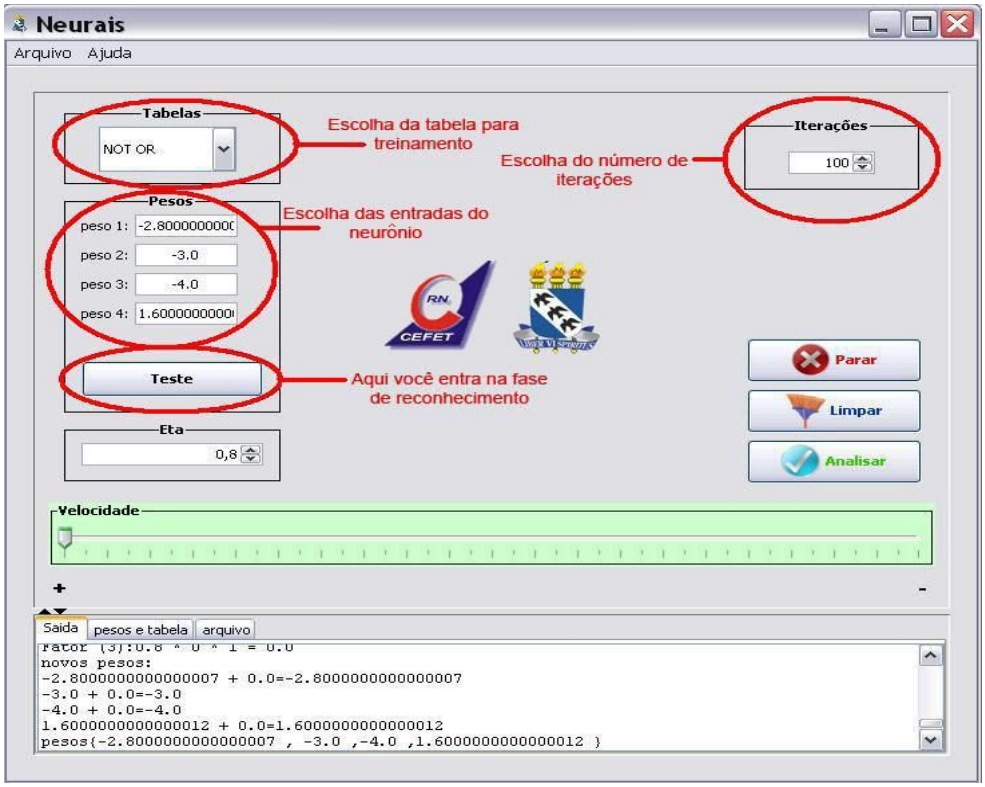

Figura 4 – Fase de treinamento

Figura 5 – Fase de reconhecimento (NOT OR)

3.2. Configurando as Entradas

São ajustáveis pelo usuário a partir da interface gráfica: a porta-lógica desejada para o aprendizado do neurônio, os valores dos pesos iniciais e o peso do bias (peso 4), o valor do coeficiente de aprendizagem (Eta) e o número de máximo de iterações da fase de aprendizagem. O usuário pode, ainda, configurar a velocidade de execução do programa para acompanhar a evolução do aprendizado.

3.3. Saídas

A área de saída do programa disponibiliza três tipos de saídas diferentes, cada uma em uma aba. A aba principal, chamada “Saída”, mostra os dados da execução atual. A aba “pesos e tabela” mostra os pesos iniciais, a tabela escolhida e os pesos ajustados. A aba “arquivo” mostra como os pesos foram armazenados no arquivo texto correspondente àquela tabela que está sendo aprendida.

3.4. Persistência

O aplicativo trabalha com arquivos texto, isso significa que um treinamento previamente realizado pode ser restaurado, e que um treinamento atual pode ser salvo para uso futuro.

4. Conclusão

Uma grande falha acadêmica contemporânea é o ensino teórico sem a devida complementação prática. Este trabalho tem como principal objetivo auxiliar e facilitar o estudo do neurônio com peso. Com o intuito de integrar a teoria à prática, o programa já apresentado foi desenvolvido. Durante seu desenvolvimento vários conceitos, antes nebulosos na teoria, foram completamente esclarecidos. Além disso, pôde-se verificar uma utilidade prática para os conhecimentos adquiridos, o que evita uma sensação de lacuna após o aprendizado teórico.

REFERÊNCIAS

HAYKIN, S. **Redes Neurais.** 2. ed. São Paulo: Bookman, 2001.

BRAGA, A.P.; CARVALHO, A.C.; LUDERMIR, T.B. **Redes Neurais Artifciais.** 1. ed. Rio de Janieiro: LTC, 2000.

C.A.G. Fonseca, F.M.U. de Araújo, A.V. Medeiros, e A.L. Maitelli, "Algoritmos Genéticos para Otimização de um Controlador Nebuloso para Supressão de Vibrações", Anais do VI SBAI, Bauru, São Paulo, 2003, pp. 959-963.

A.P.L.F. Carvalho **Redes Neurais Artificiais.** Disponível em:
< http://www.icmc.usp.br/~andre/research/neural/index.htm> Acesso em: 09 ago2008.

AGRADECIMENTOS

Deixamos expressos nossos sinceros agradecimentos às seguintes instituições e pessoas, sem as quais o presente trabalho teria sido impossível:

- Universidade do Estado do Rio Grande do Norte, no apoio técnico e a disponibilidade da sua estrutura física para o desenvolvimento deste trabalho.

- Centro Federal de Educação Tecnológica do Rio Grande do Norte, no apoio técnico e incentivo no desenvolvimento.

- Nossos amigos, Flávio Henrique e Waldney Andrade, que nos ajudou bastante no apoio ao desenvolvimento do aplicativo.