1° Congresso Brasileiro de Redes Neurais

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

REDE NEURAL LÓGICA

A. L. C. Canella e L. P. Caloba

COPPE / EE / UFRJ
CP 68504 CEP 21945-970, Rio de Janeiro, Brasil
e.mail : caloba@coe.ufrj.br

RESUMO

Neste trabalho é apresentado um algoritimo construtivo para redes neurais com sinais de entrada e saída lógicos. A rede obtida é de duas camadas compostas por neurônios do tipo peceptron com função de ativação do tipo sinal. Todas as sinapses possuem valores inteiros. A rede obtida pode ser convertida num circuito digital com a mesma estrutura obtida pelo algoritimo de Quine-McCluskey.

1. INTRODUÇÃO

Sabe-se que qualquer função lógica pode ser implementada com portas lógicas numa estrutura digital de duas camadas, na qual utiliza-se apenas portas lógicas dos tipos E, OU, E negativo ou OU negativo[1]. Também sabe-se que com apenas um neurônio é possível implementar qualquer uma destas portas lógicas[2]. Logo, é coerente imaginar que qualquer função lógica possa ser implementada utilizando-se uma estrutura neural com apenas duas camadas[2].

O objetivo deste trabalho é apresentar um algoritimo construtivo de treinamento de rede neural com sinais de entrada e de saída lógicos. Este algoritimo monta uma rede neural de duas camadas de modo que cada neurônio da camada intermediária represente um mintermo da função lógica a ser implementada, realizando o neurônio da camada de saída a função lógica OU destes mintermos.

2. NEURÔNIO "E LÓGICO"

Neurônio E lógico é um dispositivo com entradas, saída e parâmetros com valores inteiros cuja saída fica ativa sse a entrada pertencer ao mintermo que o neurônio representa. O modelo de neurônio utilizado em tal finalidade é o do tipo perceptron com a função sinal como função de ativação[2].

Este dispositivo realiza a função lógica E através de uma sutileza matemática. A entrada é formada por n sinais lógicos representados pelos valores inteiros +1 e -1, correspondendo respectivamente aos valores lógicos verdadeiro e falso. As sinapses podem assumir os valores +1, -1 ou 0, com exceção da sinapse de polarização, que pode assumir outros valores inteiros. Cada parcela da soma de produtos que define a variável auxiliar u só pode valer -1, 0 e +1, com exceção da parcela correspondente a sinapse de polarização, que será analisada separadamente. Para que o neurônio j da primeira camada represente uma entrada lógica é suficiente fazer o vetor sinapse $\mathbf{w_j}$ igual ao vetor entrada $\mathbf{x}$. Desta forma, são positivas todas as parcelas da soma que define a variável auxiliar u correspondentes às sinapses ligadas aos sinais de entrada. Definindo-se apropriadamente o valor da sinapse de polarização w_0 é possível, utilizando-se estas características, realizar um E lógico entre o vetor de entrada $\mathbf{x}$ e o vetor sinapse $\mathbf{w_j}$.

O vetor sinapse $\mathbf{w_j}$ que representa os parâmetros do neurônio j da primeira camada é definido pela equação (1), na qual w_0 é a sinapse de polarização e w_i é a sinapse que liga o componente genérico x_i da entrada $\mathbf{x}$ ao neurônio j. O vetor $\mathbf{x}$ que representa a entrada é definido pela equação (2), -1 é o valor fixo da entrada de

polarização x_0 escolhido negativo por conveniência.

(1) $\mathbf{w}_j = \left[w_0\ w_1\ w_2 \cdots w_i \cdots w_{n-1}\ w_n\right]^t$

(2) $\mathbf{x} = \left[-1\ x_1\ x_2 \cdots x_i \cdots x_{n-1}\ x_n\right]^t$

A variável auxiliar u é definida pela equação (3) e a parte referente às parcelas correspondentes às sinapses que ligam os sinais de entrada ao neurônio, representada pela variável q, é definida pela equação (4). O valor desta variável é limitado pela relação definida na equação (5), na qual z é o número de sinapses nulas.

(3) $u = -w_0 + x_1w_1 + x_2w_2 + \cdots + x_iw_i + \cdots x_{n-1}w_{n-1}$

(4) $q = x_1w_1 + x_2w_2 + \cdots + x_iw_i + \cdots x_{n-1}w_{n-1} + x_nw_n$

(5) $q \le n - z$

Para que a entrada **x**, apresentada ao neurônio j da primeira camada, pertença ao mintermo por ele representado, todas as parcelas de q devem ser positivas ou nulas. Por outro lado, para que a entrada não pertença a este mintermo, pelo menos uma parcela de q deve ser negativa. As equações (6) e (7) definem matematicamente os dois casos, sendo k o número de parcelas negativas de q.

(6) $q = n - z$

(7) $q = n - z - 2k$

A sinapse de polarização w_0 é definida de tal forma que u seja positivo quando **x** pertencer ao mintermo representado pelo neurônio j e negativo quando não pertencer. A equação que relaciona a variável auxiliar u, a variável q e a sinapse de polarização w_0 é obtida das equações (3) e (4) e é representada pela equação (8).

(8) $u = q - w_0$

Aplicando a equação (6) na equação (8) obtem-se a relação dada pela equação (9), na qual é definida a faixa de valores que w_0 pode assumir para tornar a saída do neurônio j positiva quando **x** pertencer ao mintermo por ele representado. Por outro lado, aplicando a equação (7) na equação (8) obtem-se a relação dada pela equação (10) na qual é definida a faixa de valores que w_0 pode assumir para tornar a saída do neurônio j negativa quando **x** não pertencer ao mintermo por ele representado.

(9) $w_0 < n - z$

(10) $w_0 > n - z - 2k$

Unindo as relações dadas pelas equações (9) e (10) obtem-se a relação dada pela equação (11), na qual é definida a faixa de valores possíveis de w_0 para satisfazer ambas relações. No caso limite, onde k é igual a 1, o único valor inteiro que satisfaz esta relação é indicado pela equação (12).

(11) $n - z - 2k < w_0 < n - z$

(12) $w_0 = n - (z + 1)$

Utilizando-se este valor inteiro para w_0, dependente apenas da ordem da entrada e da ordem do neurônio (número de sinapses nulas), iguais respectivamente a n e z, ao aplicarmos a função sinal sobre a variável auxiliar u, transformamos um simples neurônio do tipo perceptron em um dispositivo que determina a pertinência da entrada a um mintermo.

3. SUPERFÍCIE SEPARADORA

Substituindo na equação (3) o valor inteiro de w_0 definido pela equação (12) e igualando a variável auxiliar u à zero, temos a definição da superfície separadora dada pela equação (13) [2].

(13) $x_1w_1 + x_2w_2 + \cdots + x_iw_i + \cdots x_{n-1}w_{n-1} + x_nw_n + z - n + 1 = 0$

Os pontos de interseção do hiperplano separador com os eixos cartesianos são determinados pela equação (14), pela qual se verifica não haver interseção com os eixos relativos às sinapses nulas e, caso contrário, a interseção ocorre em pontos inteiros com módulo definido pela equação (15).

(14) $x_i = \dfrac{z + 1 - n}{w_i}, \quad w_i \neq 0 \ \text{ e } \ x_j = 0 \ \ \forall \ j \neq i$

(15) $|x_i| = z + 1 - n, \quad w_i \neq 0$

Um mintermo formado por z sinapses nulas é representado por um sub-hipercubo de ordem z contido no hipercubo de ordem n. O número de arestas de ligação *nal* que conectam este sub-hipercubo de ordem z aos demais vértices do

hipercubo de ordem n é o número de arestas cortadas pelo hiperplano separador. Como cada vértice de um hipercubo de ordem n é ligado a n outros vértices e os que formam o sub-hipercubo de ordem z já estão ligados a z vértices dele próprio, conclui-se que cada um dos 2^z vértices do mesmo é ligado a $(n\text{-}z)$ vértices restantes do hipercubo de ordem n, definindo-se desta forma a equação (16) que determina o número *nal*. A forma genérica das entradas representadas por um mintermo realizado por um neurônio E lógico é fornecida pela equação (17), sendo e_i igual a w_i se w_i for diferente de zero e podendo ser ± 1 se w_i for igual a zero.

(16) $$nal = (n-z)2^z$$

(17) $$\mathbf{e} = \left[-1\ e_1\ e_2 \cdots e_i \cdots e_{n-1}\ e_n\right]$$

Como há z sinapses iguais a zero, existem z entradas "don't care" em **e**, determinando-se desta forma 2^z vetores que satisfazem a equação (17). Aplicando-se o vetor genérico e em u obtem-se a equação (18).

(18)
$$u = e_1 w_1 + e_2 w_2 + \cdots + e_i w_i + \cdots + e_{n-1} w_{n-1} + e_n w_n + z - n + 1 = +1$$

Como existem $(n\text{-}z)$ sinapses não nulas em (18), igualar a zero apenas uma das $(n\text{-}z)$ entradas correspondentes a estas sinapses, significa zerar a variável auxiliar u com uma entrada correspondente à um ponto localizado no centro de uma aresta. Como existem $(n\text{-}z)$ posibilidades e para cada uma há 2^z formas diferentes, devido às entradas "don't care", pode-se concluir que existem no total $(n\text{-}z)2^z$ pontos localizados em centros de arestas que anulam u. Ou seja, o hiperplano cruza $(n\text{-}z)2^z$ centros de arestas. Como o mesmo só cruza $(n\text{-}z)2^z$ arestas do hipercubo, conclui-se que o mesmo só o cruza nos centros das arestas.

4. NEURÔNIO "OU LÓGICO"

A implementação do neurônio OU lógico é bastante simples e deriva diretamente do neurônio E lógico. Imagina-se um hipercubo com n^2 vértices, cria-se um neurônio E lógico que separe apenas o vértice correspondente a todas as entradas negativas, ou seja, todas as entradas com nível lógico falso. Este neurônio E lógico fornecerá saída positiva quando todas as entradas tiverem nível lógico falso e saída negativa quando pelo menos uma entrada possuir nível lógico verdadeiro. Deste modo, é implementado um OU lógico negativo. Invertendo o sinal de saída deste neurônio E lógico, o que representa multiplicar por -1 todos os valores das sinapses, obtem-se um neurônio OU lógico, em que w_i=+1 $\forall$ i e w_0=1-n.

5. TREINAMENTO

O treinamento é um processo iterativo onde apenas as entradas com saídas verdadeiras são consideradas e usadas no treinamento, e apenas uma única vez. Para cada entrada, inicialmente é formado um hiperplano que a separa dos demais vértices, e é iniciado um processo iterativo convergente que determinará todos os hiperplanos que separam o maior número de vértices possivel que esta entrada pode formar com as já treinadas. Este hiperplano inicial é representado por um neurônio que possui a seguinte regra de formação:

(19) $$\mathbf{WB} = [w_0\ w_1\ w_2 \ldots w_i \ldots w_{n-1}\ w_n]$$

fazendo-se,

$w_0 = n\text{-}1$

$w_i = x_i$

onde,

x_i : é a componente i do vetor de entrada;

Os neurônios são divididos em 3 grupos: o grupo A possui todos os neurônios existentes antes da nova entrada ser treinada; o grupo B é inicialmente constituído pelo neurônio WB, equação (19), criado pela nova entrada; o grupo C é formado pelos neurônios gerados quando se compara um neurônio do grupo B com um do grupo A.

5.1 Combinação de hiperplanos.

A seguir é descrito o método utilizado para comparar e gerar os novos hiperplanos.

Sejam **WA**, **WB** e **WC**, respectivamente, um neurônio do grupo A, um do grupo B e o a ser gerado no grupo C. Cujos vetores representativos são os seguintes:

$$\mathbf{WA} = [\ wa_0\ wa_1\ wa_2 \ldots wa_i \ldots wa_{n-1}\ wa_n\ ]$$
$$\mathbf{WB} = [\ wb_0\ wb_1\ wb_2 \ldots wb_i \ldots wb_{n-1}\ wb_n\ ]$$
$$\mathbf{WC} = [\ wc_0\ wc_1\ wc_2 \ldots wc_i \ldots wc_{n-1}\ wc_n\ ]$$

O algorítimo de comparação e associação é o seguinte:

0) $z = 0$
1) se $wb_0 < wa_0$, então:
sai;
2) repetir sequencialmente para i igual a 1 até n
2.a) se $wa_i = 0$ e $wb_i \neq 0$, então:
sai;
2.b) $wc_i = wa_i + wb_i$
se $wc_i > 0$, então $wc_i = +1$;
se $wc_i < 0$, então $wc_i = -1$;
se $wc_i = 0$, então $z = z + 1$;
2.c) se $z > 1$, então:
sai;
3) se $z \neq 1$, então:
sai;
4) $wc_0 = wa_0 - 1$
5) se $wa_0 = wb_0$, então:
elimina-se o neurônio **WA**;
6) incluir o novo neurônio **WC**.

5.2 Algorítimo de Treinamento

Cada neurônio do grupo B é comparado com todos os neurônios do grupo A e os neurônios gerados nesta comparação formam o grupo C. Após todos os neurônios do grupo B serem comparados com os do grupo A, os neurônios restantes do grupo B são unidos aos restantes do grupo A. E os gerados no grupo C passam a formar o grupo B. Este processo iterativo continua até que não haja a formação de novos neurônios no grupo C.

O algoritimo consiste no seguinte:

0) verificar se ainda há entradas a serem treinadas: se houver então continuar, caso contrário, ir para o passo 10;
1) considerar a próxima entrada;
2) se a saída correspondente for falsa, então retornar para o passo 0, caso contrário, continuar;
3) gerar um neurônio que forme um hiperplano que separe o vértice correspondente a esta entrada das demais;
4) iniciar o grupo B com este neurônio;
5) verificar se há neurônios no grupo B: se houver continuar, caso contrário, retornar ao passo 0;
6) comparar os neurônios do grupo B com os do grupo A e formar novos neurônios, se possível, no grupo C;
7) formar um novo grupo A com os neurônios restantes dos grupos A e B;
8) formar um novo grupo B com os novos neurônios gerados no grupo C;
9) retornar ao passo 5;
10) iniciar a etapa de secagem;

Durante o treinamento é necessário realizar simultaneamente as seguintes secagens (eliminação de neurônios supérfluos):

a) após um neurônio do grupo B ter sido comparado com todos os neurônios do grupo A, ele deve ser eliminado caso tenha gerado algum novo neurônio no grupo C;
b) ao se comparar um neurônio do grupo B com um do grupo A e houver a geração de um novo neurônio no grupo C, o neurônio do grupo A deve ser eliminado caso a sua ordem seja igual a do neurônio do grupo B;
c) só deve ser gerado um novo neurônio no grupo C caso o mesmo ainda não tenha sido gerado neste grupo;

Com o objetivo de acelerar o processo, os neurônios do grupo B só devem ser comparados com os do grupo A que tiverem ordem igual ou superior.

5.3 Secagem

O método consiste na formação de um grupo I a partir dos elementos do grupo II que inicialmente possui todos os neurônios obtidos na etapa de treinamento, isto é, os neurônios do grupo A. O objetivo é formar o grupo I com apenas os elementos indispensáveis do grupo II para a representação da função lógica a ser implementada.

O algorítimo é bastante simples e eficiente. Consiste no transporte para o grupo I de todos os neurônios do grupo II cujos hiperplanos separem pelo menos um vértice não separado pelos outros hiperplanos. Quando não houver nenhum neurônio com esta caracteristica, elimina-se o neurônio de menor significância do grupo II. Se persistir em não haver nenhum neurônio do grupo II habilitado para ser transportado para o grupo I, elimina-se outro neurônio de menor significância do grupo II. Quando houver algum neurônio do grupo II que possa ser transportado para o grupo I, realiza-se a transferência. Este processo segue sucessivamente até não haver mais neurônios no grupo II.

Este algoritimo é de facil implementação. Cada iteração consiste basicamente de quatro

etapas sequênciais. São realizadas quantas iterações forem necessárias para eliminar ou transferir para o grupo I todos os neurônios do grupo II.

A primeira etapa consiste em determinar para cada neurônio do grupo II três parâmetros. O primeiro, denominado de P1, é o número de vértices separados pelo hiperplano correspondente ao neurônio que é igual a 2^{-z}, onde z é a ordem ou o número de sinapses nulas. O segundo, denominado de P2, é o número de vértices, entre os separados pelo hiperplano correspondente ao neurônio em questão, não separados por nenhum dos hiperplanos correspondentes aos neurônios do grupo I. O terceiro, denominado de P3, é o número de vértices, entre os separados pelo hiperplano correspondente ao neurônio em questão, não separados pelos hiperplanos correspondentes aos neurônios do grupo I e aos demais neurônios do grupo II.

O algoritimo que determina os valores dos parâmetros P1, P2 e P3 é bastante simples e direto. Sabendo-se o número de sinapses nulas e quais são elas, a determinação de quantas e quais são as entradas englobadas pelo neurônio em questão é direta. Deste modo, a determinação do valor do parâmetro P1 é direta e é igual a 2^{-z}. Para se determinar os valores dos parâmetros P2 e P3 deve-se considerar duas redes neurais. A rede I é formada pelos neurônios do grupo I e a rede II é formada pelos neurônios que formam o grupo II com exceção do neurônio em questão. Realiza-se um loop com P1 iterações, onde é apresentada a ambas redes as P1 entradas englobadas pelo neurônio em questão. O valor do parâmetro P2 é determinado pelo número de entradas que não ativam a rede I, ou seja, pelo número de entradas ainda não englobadas por algum dos neurônios do grupo I. O valor do parâmetro P3 é determinado pelo número de entradas que não ativam simultaneamente as redes I e II, ou seja, pelo número de entradas não englobadas pelos neurônios dos grupos I e II.

A segunda etapa consiste simplesmente na eliminação de todos os neurônios do grupo II que possuam o parâmetro P2 nulo, pois, estes não possuem nenhuma informação nova a ser acrescentada ao grupo I.

A terceira etapa é o transporte para o grupo I de todos os neurônios artificias do grupo II com o parâmetro P3 não nulo, pois, estes possuem alguma informação nova a ser acrescentada no grupo I não contida em nenhum outro neurônio do grupo II.

A quarta etapa consiste na eliminação do neurônio de menor significância do grupo II, caso não tenha ocorrido nenhuma transferência para o grupo I. Ou seja, elimina-se um dos neurônios que obtiverem o menor parâmetro P1 entre os neurônios do grupo II com o menor parâmetro P2.

Este algoritimo é repetido sucessivamente até que não haja mais neurônios no grupo II. Como em cada iteração pelo menos um neurônio do grupo II é eliminado ou transferido para o grupo I, este é um processo iterativo convergente, pois o número de neurônios do grupo II é finito.

6. ENTRADAS "DON'T CARE"

Será realizada a construção de duas redes neurais simultaneas. Na rede principal serão treinadas as entradas com saída verdadeira e entradas "don't care" com saída considerada como verdadeira. A rede auxiliar será treinada apenas com entradas "don't care" com saída considerada como verdadeira. Desta forma, teremos na rede principal a representação neural de todos os maiores mintermos possíveis de serem formados com entradas cuja saída seja verdadeira e com entradas "don't care" com saída considerada como verdadeira. Também, estará representado na rede principal todos os maiores mintermos possíveis de serem formados apenas com entradas "don't care" com saída considerada como verdadeira. Na rede auxiliar estará representado todos os maiores mintermos que as entradas "don't care" com saída considerada com verdadeira podem formar.

A etapa de secagem, além de ser responsável por eliminar os mintermos dependentes, será também responsável pela eliminação dos termos formados apenas por entradas "don't care" com saída considerada como verdadeira. Desta forma, restará na rede principal apenas os maiores mintermos que possuam alguma entrada com saída definida. Para que isto ocorra, a etapa de secagem deve ser alterada. Esta alteração ocorre na primeira etapa do algoritimo iterativo responsável pela seleção e eliminação dos elementos do grupo II. Acrescenta-se um grupo III formado pelos neurônios da rede auxiliar. O cálculo do parâmetro P1 é o mesmo, mas, os cálculos dos

parâmetros P2 e P3 devem ser alterados. O parâmetro P2 passa a ser o número de vértices, entre os separados pelo hiperplano correspondente ao neurônio do grupo II em questão, não separados pelos hiperplanos correspondentes aos neurônios dos grupos I e III. O parâmetro P3 passa a ser o número de vértices, entre os separados pelo hiperplano correspondente ao neurônio do grupo II em questão, não separados pelos hiperplanos correspondentes aos neurônios dos grupos I e III e pelos demais neurônios do grupo II.

7. EXEMPLOS

São apresentados dois exemplos de [1], onde as funções lógicas são definidas por um somatório de mintermos: m indica entradas com saídas lógicas verdadeiras e d indica entradas "don't care". Os resultados obtidos são apresentados nas tabelas abaixo.

$F(A,B,C,D,E) = \Sigma$ m(0,6,8,10,12,14,17,19,20, 22,25, 27,28,30)

w0	w1	w2	w3	w4	w5	mint.
3	-1	-1	-1	0	-1	$\overline{A}\,\overline{C}\,\overline{D}\,\overline{E}$
2	-1	0	0	1	-1	$\overline{A}B\overline{E}$
2	1	0	-1	0	1	$A\overline{C}E$
2	-1	1	1	0	0	$CD\overline{E}$
2	-1	0	1	0	1	$AC\overline{E}$

$F(A,B,C,D,E) = \Sigma$ m(1,4,7,14,17,20,21,22,23) + d(0,3,6,19,30)

w0	w1	w2	w3	w4	w5	mint.
2	-1	1	1	0	0	$CD\overline{E}$
2	1	0	-1	-1	0	$\overline{B}\,\overline{C}E$
2	-1	0	1	-1	0	$\overline{B}C\overline{E}$
2	0	1	1	-1	0	$\overline{B}CD$
2	0	0	1	-1	1	$A\overline{B}C$

8. CONCLUSÕES

O resultado final obtido pelo algoritmo apresentado é o mesmo que seria obtido pelo algoritmo de Quine-McCluskey caso a rede obtida fosse implementada com portas lógicas. Existem outros algoritimos construtivos que podem obter estruturas de redes neurais mais compactas. Entretanto, com relação a implementação, não são tão simples e confiáveis quanto aos circuitos digitais[3]. A estrutura neural proposta é simples de ser implementada e tão confiável e compacta quanto um circuito digital obtido pelo uso do algoritimo de Quine-McCluskey.

A diferença básica entre os dois algoritmos é que enquanto o de Quine-McCluskey opera sobre toda a massa de dados simultaneamente, necessitando a utilização de grandes tabelas, o algoritmo apresentado neste trabalho processa cada entrada individualmente e um só vez. Este processamento dos dados de entrada um a um é usualmente um requisito dos sistemas neurais. Assim, pode-se supor que o algoritimo de treinamento proposto se aproxima muito mais de um modelo neural natural que um algoritimo de treinamento diretamente originado do algoritimo de Quine-McCluskey, alem da necessitar de muito menos memória.

9. REFERÊNCIAS

[1] F. F. Hill e G. R. Peterson, "Introduction to Switching Theory & Logical Design", John Willey & Sons, 1981.

[2] J. Hertz, A. Krogh e R. G. Palmer, "Introduction to the Theory of Neural Computation", Addison Wesley, 1990.

[3] H. M. A. Andree, G. T. Barkema, W. Lourens e A. Taal, "A Comparison Study of Binary Feedforward Neural Networks and Digital Circuits", Neural Networks, Vol. 6, pp. 785-790, 1993.