INSTITUTO FEDERAL DE EDUCAÇÃO,

CIÊNCIA E TECNOLOGIA DE MINAS GERAIS

MARCO AURÉLIO MONTEIRO LIMA

**TOPICOS ESPECIAS EM ENGENHARIA DE COMPUTAÇÃO:**

PERCEPTRONS DE MÚLTIPLAS CAMADAS

BAMBUÍ

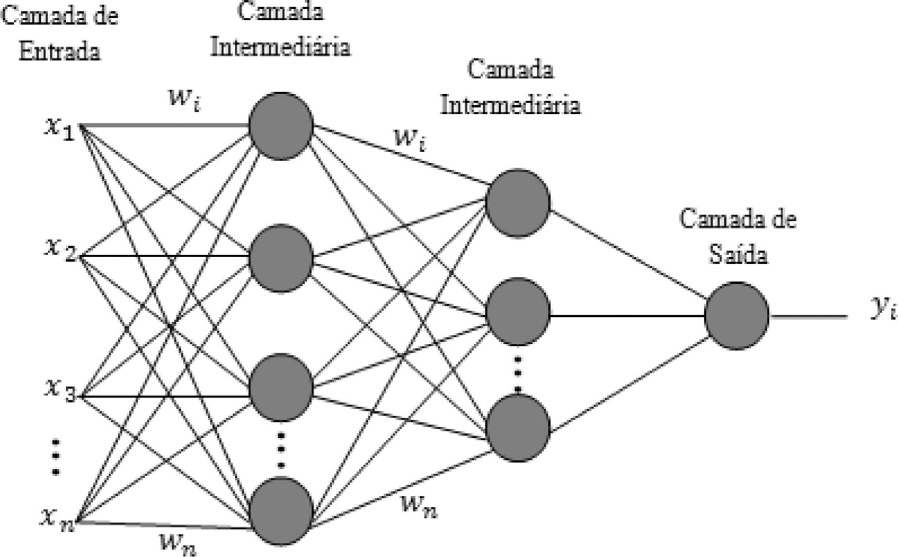
2017

**Perceptrons de Múltiplas Camadas**

É uma rede com uma camada sensorial ou camada de entrada, que possui tantos nós de entrada quanto forem os sinais de entrada, uma ou mais camadas ocultas de neurônios e uma camada de saída com um número de neurônios igual ao número de sinais de saída. O sinal de entrada se propaga para frente através das camadas até a saída ou seja a rede é alimentada para frente.

A MLP é treinada de forma supervisionada pela regra de aprendizagem que minimiza o erro. O algoritmo usado é o de retro propagação de erro.

Figura 1: Exemplo de rede MLP;



A propagação do sinal é feita ao aplicar na camada de entrada o vetor de sinais de entrada, então é calculado o campo local induzido e os sinais de saída para todos os neurônios desta camada. A rede MLP funciona como uma sequência de Perceptrons simples interconectados, onde gera uma propagação de sinais de entrada atravessando todas as camadas até a saída.

Todo o procedimento do treinamento do MLP é baseado no sinal de erro. Na MLP, o sinal de erro do neurônio de saída *j,* na interação *n*, é definido por:

*ej (n) = dj (n) – yj (n).*

Onde:

*yj (n)* é a resposta obtida para o neurônio *j.*

*dj (n)* é a resposta esperada para o neurônio *j.*

O erro de cada neurônio é elevado ao quadrado para evitar que um erro negativo oriundo de um neurônio compense um erro positivo de outro.

Imagina-se uma MLP com função de transferência sigmoide, na qual o seriam arbitrados valores aleatórios aos pesos sinápticos e níveis de bias. O primeiro conjunto de sinais de entrada, resultaria num conjunto de sinais de saída que comparado com os sinais de saída esperados resultaria num erro global instantâneo. Supondo que seja desejado ajustar o erro hipotético w(k,j), ou seja um erro w de uma sinapse de um neurônio k e o neurônio j, pode se escrever a equação de erro global E(n) em função de w(k,j) mantendo os outros pesos constantes.

**O algoritmo de retro propagação de erro:**

Ele pode ser descrito em cinco etapas:

1 – Inicializa valores aleatórios aos pesos e ao bias numa distribuição uniforme de modo que a média seja zero.

2 – Apresenta-se uma época de exemplos a rede. Para cada exemplo é feita a propagação de sinais e a retro propagação dos erros com a correção dos pesos e dos níveis de bias.

3 – Propagação de sinais: aplica-se a camada de entrada da rede o vetor de sinais de entrada x(n) depois é calculado o campo local induzido e o sinal de saída para todos os neurônios. Em seguida calcula-se o sinal de erro para cada neurônio da camada de saída, pela comparação com o vetor de saída esperados. Depois calcula-se o erro instantâneo com este, o erro médio global.

4 – Retro propagação dos sinais de erro: calcula os gradientes locais para todos os neurônios da camada de saída, em seguida calcula-se os ajustes para os pesos e bias daquela camada, e depois somados aos valores atuais. Depois é feito o cálculo do gradiente local para a penúltima camada (camada oculta). Então é calculado os ajustes de pesos e bias para essa camada, e depois somados aos valores atuais.

O processo prossegue para as demais camadas ocultas, e também para a camada de entrada.

5 – Interação: iteram-se as computações novas épocas de exemplos de treinamento para rede de forma aleatória de época para época, até que seja satisfeito o critério de parada, que pode ser o número máximo de interações ou um valor limite para o erro global médio da rede.

**Analise da Implementação:**

A matriz de saída:

[,1] [,2] [,3] [,4]

[1,] -1 1 -0.9356076 0.9335540

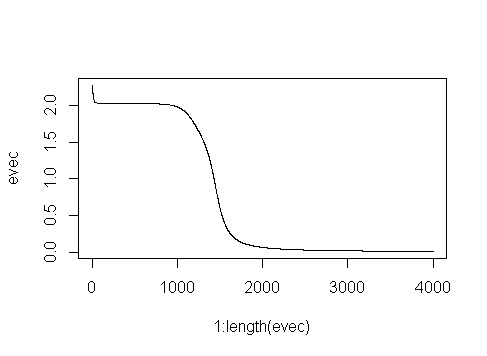
[2,] 1 -1 0.9211483 -0.9187666

[3,] 1 -1 0.9186985 -0.9161373

[4,] -1 1 -0.9433776 0.9414895

A coluna um e dois mostra os valores desejados da rede, a coluna três e quatro mostram os valores obtidos. Arredondando os valores obtidos a um pode-se concluir, que a rede obteve um bom desempenho alcançou valores bem próximos do esperado para a rede.

Gráfico 1: Convergência do erro;



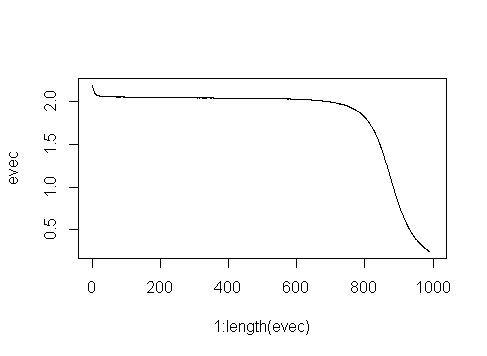
Valores iniciais:

|  |  |  |
| --- | --- | --- |
| Taxa de aprendizagem | Máximo de épocas | Tolerância de erro |
| 0.01 | 4000 | 0 |

Alterando os valores da taxa de aprendizagem, máximo de épocas e tolerância do erro:

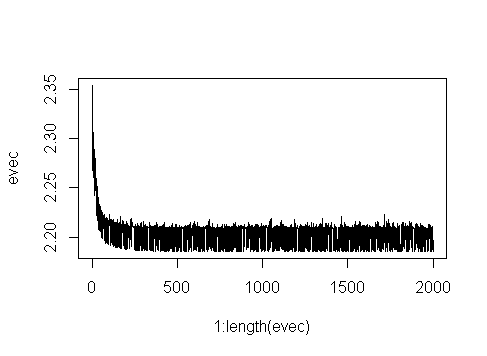
|  |  |  |
| --- | --- | --- |
| Taxa de aprendizagem | Máximo de épocas | Tolerância de erro |
| 0.02 | 1000 | 0.25 |
| 0.1 | 2000 | 0.5 |
| 0.01 | 9000 | 0 |
| 0.5 | 6000 | 2 |

Gráfico 2: Gráfico de convergência para 1000 épocas;



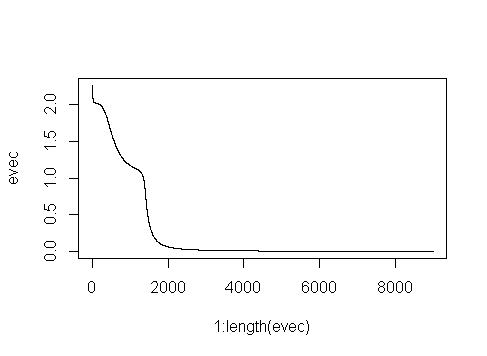
Nesse primeiro foi alterado mais drasticamente a tolerância do erro o que fez o erro demorar um pouco mais para convergir.

Gráfico 3: Gráfico de convergência para 2000 épocas;



Neste gráfico foi alterado a taxa de aprendizagem o que fez o gráfico oscilar muito.

Gráfico 4: Gráfico de convergência para 9000 épocas;



Aumentando o número de épocas percebe-se que o erro convergiu bem mais rápido do que o estipulado, mas também não mudou muito.

Executando os códigos algumas vezes com os valores inicias, pode-se notar que o gráfico não é o mesmo para todas as execuções, e as vezes varia muito de um para outro. Isso deve-se porque os valores dos pesos estão sendo aleatorizados todas as vezes que executa o código, então o gráfico não será o mesmo pois alguns vão convergir mais rapidamente, enquanto outros não.

Comparando a mlp com a perceptron e a adaline, pode-se notar que o erro atual na mlp é sempre menor que o erro anterior, o que faz ela convergir mais rápido que a perceptron e adaline. Outro fato é que o erro não fica oscilando como na perceptron.

**Bibliografia**

JR, Oswaldo; MONTGOMERY, Eduard; ***Redes Neurais***: Fundamentos e Aplicações com Programas em C. Rio de Janeiro: Editora Ciência Moderna Ltda. 2007. 45p.