Introdução a Redes Neurais Artificiais com a biblioteca Encog em Java

Conference Paper · November 2013

DOI: 10.13140/RG.2.1.3068.5528

1 author:

Raquel Machado
University of São Paulo
4 PUBLICATIONS **10** CITATIONS

Introdução a Redes Neurais Artificiais com a biblioteca Encog em Java

Raquel Machado de Sousa[1]

[1]Laboratório de Sistemas Inteligentes (LSI) – Universidade Federal do Maranhão (UFMA)
Av. dos Portugueses – São Luis – MA – Brasil
rachel.msousanet@gmail.com

Abstract. *This article describes basic concepts of artificial neural networks, ranging from its history of development that spurred the research in this area until the main features involved in the construction of an ANN. Being demonstrated through the development framework Encog used to build RNAs in java, which provides a range of possible application in various sectors.*

Resumo. *Este artigo descreve os conceitos básicos de redes neurais artificiais, que vão desde seu histórico de desenvolvimento que impulsionaram as pesquisas nesta área, até as principais características envolvidas na construção de uma RNA. Sendo demonstrado através do framework de desenvolvimento Encog utilizada para construção de RNAs em java, que oferece uma gama de possibilidade de aplicação em vários setores.*

1. Introdução

As redes neurais artificiais (RNAs) são uma tecnologia que têm sido utilizada para diversos fins, baseada no funcionamento dos neurônios biológicos, possui grande capacidade de aprendizado e generalização diante de exemplos expostos, mesmo com a interferência de ruídos. Conhecida como máquinas de processamento paralelo e por ser uma técnica de estatística não-linear, pode resolver uma grande variedade de problemas.

Segundo Haykin (2001) as RNAs são um processador maciçamente paralelamente distribuído constituído de unidades de processamento simples, que têm a propensão natural para armazenar conhecimento experimental. Podemos dizer que o processamento é paralelo no sentido que todos os neurônios dentro do conjunto ou camada processam as suas entradas simultaneamente e independentemente.

As redes neurais fazem parte de uma área de pesquisa conhecida como Inteligência Artificial, mais precisamente na abordagem conexionista, na qual utiliza de métodos computacionais que possuem e utilizam capacidade racional como os seres humanos na resolução de problemas.

Pelo seu caráter multidisciplinar está nas mais diversas aplicações nas quais podemos citar: avaliação de imagens captadas por satélite, classificação de padrões de escrita e fala, reconhecimento de faces em visão computacional, controle de trens de grande velocidade, previsão de ações no mercado financeiro, identificação de anomalias em imagens médicas. Diante da grande variedade de sua utilização, as tarefas mais comuns realizadas são classificação, regressão numérica, agrupamento, predição e reconhecimento de padrões.

2. Histórico das Redes Neurais Artificiais

Os trabalhos em redes neurais iniciaram no ano de 1943 com a publicação de um artigo por McCulloch & Pitts, no qual os autores realizaram o primeiro modelo matemático inspirado no neurônio biológico, resultando assim na primeira concepção de neurônio artificial. [Silva 2010]

Em 1949, é modelado o primeiro método de treinamento para redes neurais, chamado de regra de aprendizado de Hebb. Ele mostrou que a plasticidade da aprendizagem em redes neurais é conseguida através da variação dos pesos de entrada dos neurônios, propondo uma teoria para explicar o aprendizado em neurônios biológicos baseada no reforço das ligações sinápticas entre os neurônios excitados. [Braga 2007]

Desde então diversos outros pesquisadores continuaram desenvolvendo pesquisas fundamentadas no neurônio biológico e algoritmos de aprendizado. Surgiram modelos da mente, como o Perceptron, de Rosenblatt, em 1958, e o Adaline, de Bernard Widrow, em 1960.

A capacidade do Perceptron de reconhecer padrões simples trouxe muito interesse de pesquisadores para área neurocomputacional, mas em 1969, Minsky e Papert chamaram a atenção para algumas tarefas que o perceptron descrito por Rosenblatt não era capaz de executar. [Braga 2007] Por estar limitado a resolução de problemas linearmente separáveis, o perceptron não é capaz de resolver “problemas difíceis de aprender” como paridade, simetria e conectividade.

Devido a estas descobertas a abordagem conexionista ficou adormecida até meados dos anos 80 quando ressurgiram novamente pelas descobertas feitas por Hopfield com sua abordagem de energia em 1982 e o algoritmo de aprendizagem backpropagation para redes perceptron múltiplas camadas (*multilayer feed-forward networks*) primeiramente proposto por Werbos, reinventada diversas vezes e popularizada por Rumelhart et al. em 1986. [Jain, Mao e Mohiuddin 1996]

O marco histórico das principais publicações na área de RNAs é apresentado no quadro abaixo:

Tabela 1. Marco histórico da pesquisa em redes neurais artificiais

Ano	Autores
1943	McCulloch e Pitts
1948	Wiener
1949	Hebb
1957	Rosenblatt
1958	Widrow e Hoff
1969	Minsky e Papert
1960 - 1980	Kohonen, Grossberg, Widrow, Arderson, Caianiello, Fukushima e Igor Aleksander
1974	Werbos
1982	Hopfield
1986	Rumelhart e McClelland

3. Neurônio Biológico x Neurônio Artificial

3.1. Neurônio Biológico

O centro do sistema nervoso humano é o cérebro, representado pela rede neural (nervosa), que recebe continuamente informação, percebe e toma decisões apropriadas. [Haykin 2001] Através dos impulsos elétricos, também chamadas de sinapses, que transportam os sinais de informação, o cérebro tem o poder de atuar e responder a estímulos a todo momento.

O cérebro é formado por cerca de 10^{11} neurônios, cada um destes processa e se comunica com milhares de outros continuamente e paralelamente. Os neurônios biológicos são divididos, de maneira simplificada, em três seções: o corpo celular, os dendritos e o axônio, cada um com funções específicas, porém complementares. [Braga 2007]

Os dendritos são filamentos prolongados responsáveis pela captação, de forma contínua, dos estímulos vindos de diversos outros neurônios. O corpo celular recebe os estímulos advindos dos dendritos, onde é processado as informações, a fim de produzir um potencial de ativação que indicará se o neurônio poderá disparar um impulso elétrico ao longo do axônio. O axônio que é formado por um único prolongamento, têm por função conduzir os impulso elétricos para os outros neurônios conectores chegando até os dendritos.

As sinapses são as unidades estruturais que se configuram como as conexões que viabilizam a transferência de impulsos elétricos do axônio de um neurônio para os dendritos dos outros. [Silva 2010] Podemos dizer que uma sinapse converte um sinal elétrico pré-sináptico em um químico e então de volta, em um sinal elétrico pós-sináptico. Na figura abaixo podemos visualizar o modelo simplificado do neurônio biológico e a propagação dos sinais sinápticos.

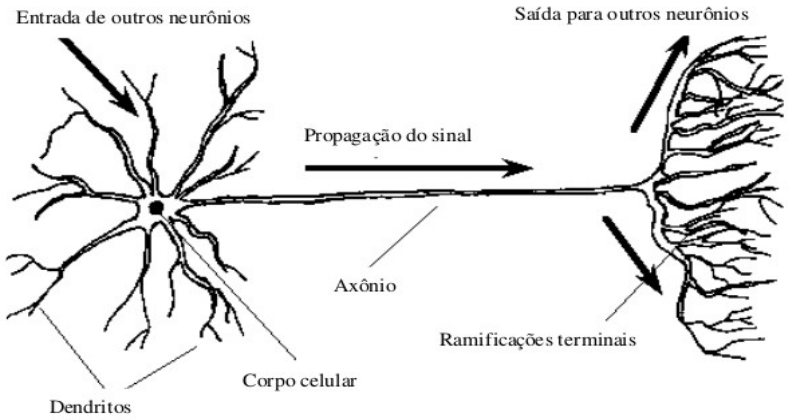

Figura 1. Modelo da célula neural biológica

3.2. Neurônio Artificial

A analogia entre o comportamento do neurônio biológico e o neurônio artificial foi o que possibilitou e deu base para as pesquisas em redes neurais. Fazendo a correlação com o modelo do neurônio mais simples de McCulloch & Pitts (1943), os sinais de entrada representados pelo conjunto $\{ x_1, x_2, \ldots, x_n \}$, equivalem aos impulsos elétricos

externos captados pelos dendritos.

As ponderações exercidas pelas junções sinápticas do modelo biológico são representadas no neurônio artificial pelo conjunto de pesos sinápticos $\{ w_1, w_2, \ldots, w_n \}$. Obtendo - se a soma ponderada dos sinais de entrada $\{ x_n \}$ e seus respectivos pesos sinápticos $\{ w_n \}$ obtemos nosso potencial de ativação, denotado por u, e representado no neurônio biológico pelo corpo celular que gera o sinal de saída propagado pelo axônio. Esse sinal de saída é representado por y no neurônio artificial.

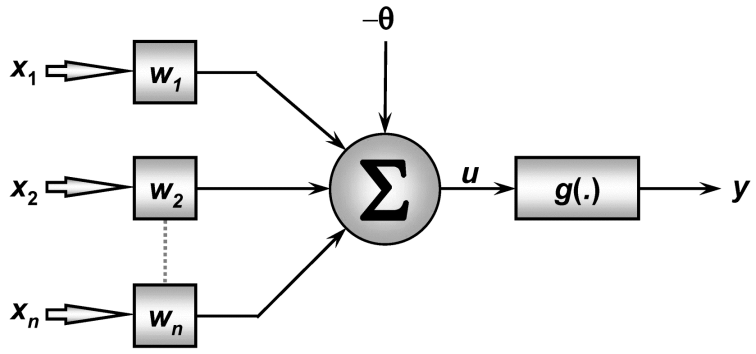

Figura 2. Modelo de um neurônio artificial

Considerando a figura 2, identifica-se os principais elementos que constituem o neurônio artificial, que de acordo com Silva (2010) são 7:

- Sinais de entrada $\{ x_1, x_2, \ldots, x_n \}$:

  São sinais advindos do meio externo e que representam os valores assumidos pelas variáveis de uma aplicação específica. Esses sinais geralmente são normalizados visando melhorar a eficiência computacional dos algoritmos de aprendizagem;

- Pesos sinápticos $\{ w_1, w_2, \ldots, w_n \}$

  São os valores que servirão para ponderar cadas uma das variáveis de entrada da rede, permitindo-se quantificar a sua relevância em relação à funcionalidade do respectivo neurônio;

- Combinador linear $\{ \sum \}$:

  Sua função é agregar todos os sinais de entrada e pondera-los com os respectivos pesos sinápticos a fim de produzir um valor de potencial de ativação;

- Limiar de ativação $\{\theta\}$:

  É uma variável que especifica qual será o patamar apropriado para que o resultado produzido pelo combinador linear possa gerar um valor de disparo em direção a saída do neurônio;

- Potencial de ativação $\{ u \}$:

  É o resultado produzido pela diferença do valor produzido entre o combinador linear e o limiar de ativação. O resultado de u é representado por:

$$u = \sum_{i=0}^{n} w_i . x_i - \theta$$

- Função de ativação { g }:

  A função limita a saída do neurônio dentro de um intervalo de valores razoáveis a serem assumidos pela sua própria imagem funcional.

- Sinal de saída { y }:

  É o valor final produzido pelo neurônio em relação a um determinado conjunto de sinais de entrada, podendo ser também utilizado por outros neurônios que estão sequencialmente interligados. $y = g(u)$

4. Funções de Ativação

Como já citado acima a função de ativação é responsável por gerar uma saída $y = g(u)$ a partir dos valores de entrada e seus respectivos pesos. Dependendo do domínio empregado as redes devem apresentar uma não – linearidade em sua saída, que seja suave, portanto, diferenciável. Então, quando propriedades dinâmicas estão envolvidas na definição do estado de ativação, equações diferenciais ou as diferenças são empregadas. [Silva 1998]

As principais funções utilizadas são [Silva 2010]:

a) Função linear

Também chamada de função identidade, a função linear, produz resultados de saída idênticos aos valores do potencial de ativação { u }, sua expressão matemática é definida por:

$$g(u) = u$$

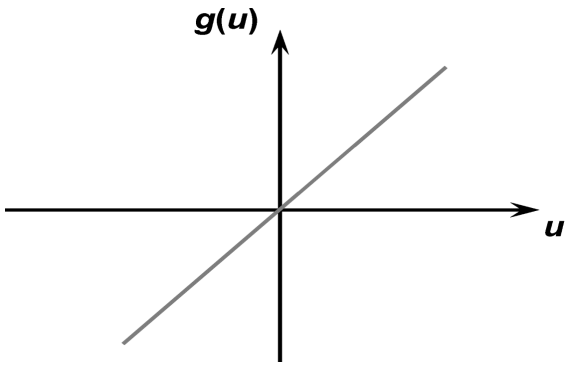

Figura 3. Função de ativação linear

b) Função degrau

O resultado produzido assumirá valores unitários positivos quando o potencial de ativação for maior que ou igual a zero e valores nulos caso seja o potencial seja menor que zero.

$$g(u) = \begin{cases} 1, se\ u \geqslant 0 \\ 0, se\ u < 0 \end{cases}$$

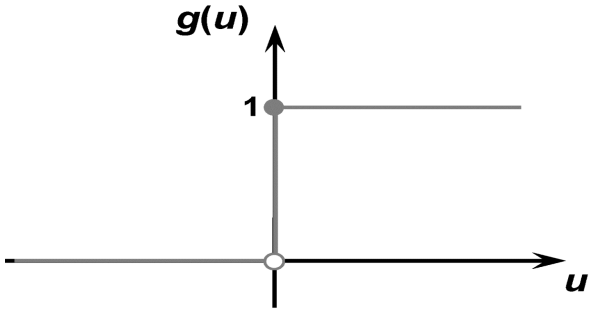

Figura 4. Função de ativação degrau

c) Função degrau bipolar

O resultado desta função assumirá valores unitários positivos quando o potencial de ativação do neurônio for maior que zero, valor nulo quando o potencial for nulo e valores negativos quando o potencial for menor que zero.

$$g(u)=\begin{Bmatrix} 1, se\ u>0 \\ 0, se\ u=0 \\ -1, se\ u<0 \end{Bmatrix}$$

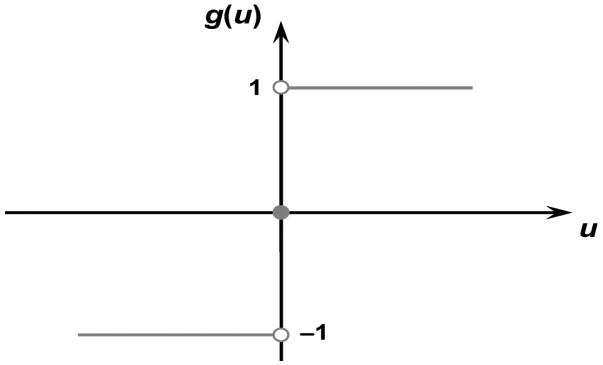

Figura 5. Função de ativação degrau bipolar

d) Função logística

O resultado de saída produzido pela função logística assumirá valores entre zero e um, de acordo com a seguinte expressão:

$g(u)=\frac{1}{1+e^{-\beta . u}}$, onde β é uma constante real associada ao nível de inclinação da função logística frente ao seu ponto de inflexão.

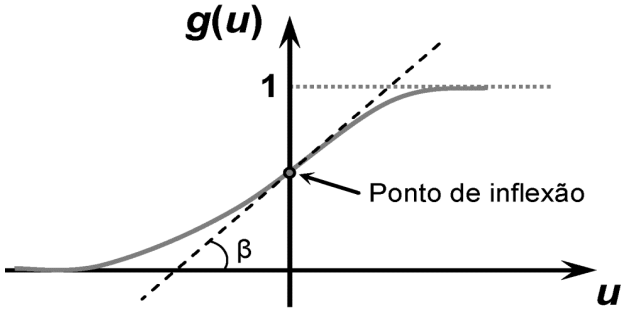

Figura 6. Função de ativação logística

e) Função tangente hiperbólica

O resultado de saída sempre assumirá valores reais entre -1 e 1, cuja expressão matemática é definida por:

$g(u)=\frac{1-e^{-\beta.u}}{1+e^{-\beta.u}}$, β aqui também está associado ao nível de inclinação em relação ao ponto de inflexão da função.

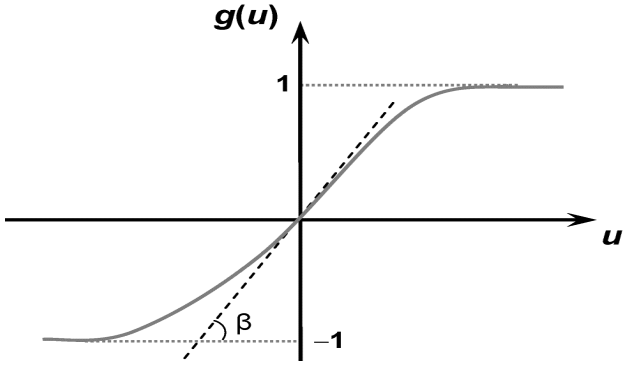

Figura 7. Função de ativação tangente hiperbólica

f) Função gaussiana

A saída da função gaussiana produzirá resultados iguais para aqueles valores de potencial de ativação $\{u\}$ que estejam posicionados a uma mesma distância do seu centro(média), sendo que a curva é simétrica em relação a este. A função é dada pela seguinte equação :

$g(u)=e^{-\frac{(u-c)^2}{2\sigma^2}}$, onde c é um parâmetro que define o centro da função gaussiana e σ denota o desvio padrão associada a mesma.

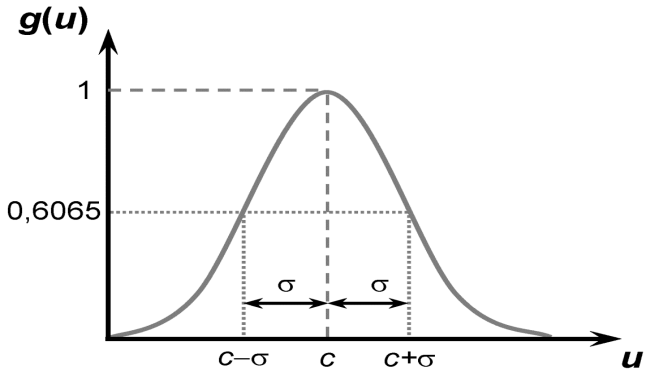

Figura 8. Função de ativação gaussiana

5. Estruturas de RNAs

A maneira pela qual os neurônios de uma rede neural são estruturados está intimamente ligada com o algoritmo de aprendizagem usado para treinar a rede. Podemos, portanto, falar de algoritmos de aprendizagem utilizados no projeto de redes neurais como sendo estruturados. [Haykin 2001]

Assim, o desempenho de uma rede neural depende não só da função de ativação

usada pelos neurônios, mas também de como esses neurônios são arranjados, isto é, estruturados uns com os outros, e dos algoritmos de aprendizado utilizados no processo de treinamento da rede.

Jain, Mao e Mohiuddin (1996) consideram apenas dois tipos de arquiteturas de RNA: redes *feedforward* e redes recorrentes. Haykin (2001) e Silva (2010) dividem a arquitetura de uma RNA em três tipos: redes *feedforward* de camada simples, redes *feedforward* de camada múltiplas e redes recorrentes. Aqui consideraremos estes três tipos de arquitetura.

5.1. Redes *feedforward* de camada simples

Este tipo de estrutura é formada por apenas uma camada de entrada e uma única camada de neurônios, que é a mesma camada de saída. Esta rede é estritamente do tipo alimentada adiante ou acíclica, e é chamada de rede de camada única, sendo que a designação "camada única" se refere à camada de saída de nós computacionais (neurônios). Não contamos a camada de entrada de nós, porque lá não é realizada qualquer computação. [Haykin 2001]

Exemplos de redes desse tipo de arquitetura são as redes Perceptron e a Adaline.

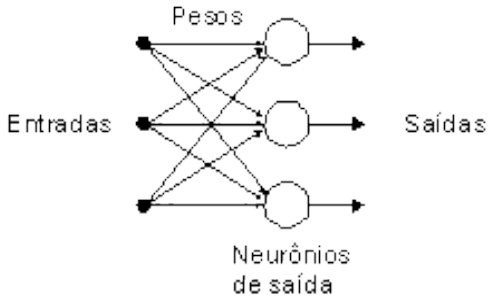

Figura 9. Rede feedforward de camada simples

5.2. Redes *feedforward* de camada múltipla

As redes *feedforward* de camadas múltiplas são constituídas pela presença de uma ou mais camadas escondidas de neurônios. Com os neurônios ocultos é possível intervir entre a camada de entrada e a saída da rede de uma maneira útil tornando a rede capaz de extrair estatísticas de ordem elevada. Como exemplos desse tipo de rede temos as Perceptron multicamadas (*multilayer Perceptron – MLP*) e as redes de base radial (*radial basis function – RBF*)

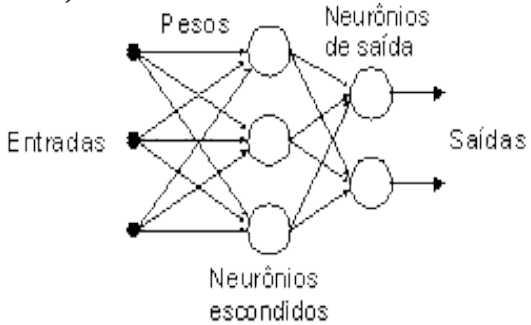

Figura 9. Rede feedforward de camada múltipla

5.3. Redes recorrentes

São redes em que as saídas dos neurônios são realimentadas como sinais de entrada para outros neurônios. Estas redes possuem uma característica de realimentação que as qualifica para o processo dinâmico de informações.[Silva 2010]

Um exemplo para esse tipo de rede são as redes Hopfield, que pode ser ilustrado na figura.

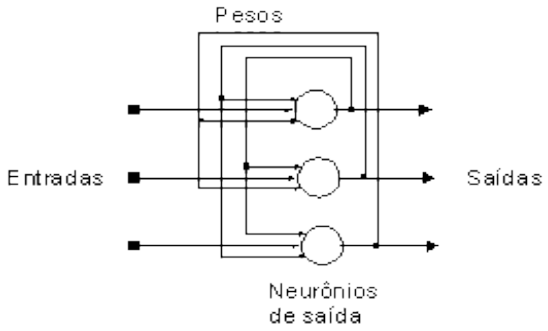

Figura 10. Rede recorrente

6. Aprendizado em Redes Neurais

Aprendizagem é uma das características mais relevantes no processo de treinamento de uma rede neural. A definição geral de aprendizagem segundo Haykin (2001) é expressa da seguinte forma:

"A aprendizagem é um processo pelo qual os parâmetros livres de uma rede neural são adaptados através de um processo de estimulação pelo ambiente no qual a rede está inserida. O tipo de aprendizagem é determinado pela maneira pela qual a modificação dos parâmetros ocorre."

A aprendizagem ocorre no momento que a rede é estimulada pelo seu ambiente, de forma a sofrer modificações nos seus parâmetros livre, isto é, seus pesos sinápticos, como resultado desta estimulação; e por fim, responde de uma nova maneira ao ambiente por causa da modificação ocorrida na sua estrutura interna.

O treinamento ou aprendizado de um RNA ocorre após a preparação dos dados e da escolha do modelo e da arquitetura de RNA a serem utilizados para os dados. O treinamento de uma RNA não possui tempo determinado de execução e, na maioria dos casos, acontece com um subconjunto de exemplos. [Barone 2003]

Há vários algoritmos diferentes para treinamento de redes neurais, Haykin (2001) e Braga (2007) agrupam em dois paradigmas principais: aprendizado supervisionado e aprendizado não – supervisionado. Jain, Mao e Mohiuddin (1996) considera três grupos: aprendizado supervisionado, aprendizado não-supervisionado e aprendizado híbrido. Aqui consideraremos a literatura de Haykin para esta classificação. Apesar de ainda não se chegar a um consenso se a aprendizagem híbrida ou aprendizagem por reforço faz parte do paradigma supervisionado ou não-supervisionado.

6.1. Aprendizado supervisionado

No aprendizado supervisionado é considerado em cada amostra sinais de entrada e sinais de saída desejadas no processo. Também chamada de aprendizagem com professor, pois podemos considerar o professor como tendo conhecimento sobre o

ambiente, sendo o conhecimento representado pelo conjunto de exemplos de entrada-saída.

Segundo Haykin (2001) o professor é capaz de fornecer à rede neural uma resposta desejada, que representa a ação ótima a ser realizada pela rede neural. Os parâmetros da rede são ajustados sob a influência combinada do vetor de treinamento e do sinal de erro. O sinal de erro é definido como a diferença entre a resposta desejada e a resposta real da rede. O ajuste é então realizado com o objetivo da rede atingir a solução ótima no sentindo estatístico. Quando a condição é alcançada, o professor deixa de atuar sobre a rede neural e lida por si mesma com o ambiente.

6.2. Aprendizado não – supervisionado

O aprendizado não-supervisionado, chamado também de aprendizagem sem professor, somente os padrões de entrada estão disponíveis para a rede, ao contrário do aprendizado supervisionado, cujo o conjunto de treinamento possui pares de entrada e saída. [Braga 2007]

Haykin (2001) divide esse paradigma em ainda duas seções:

a) Aprendizagem por reforço

   Em muitas literaturas este tipo de aprendizagem é considerada como um tipo de aprendizagem supervisionada ou como um terceiro paradigma de aprendizagem. A aprendizagem por reforço é realizado através de um crítico externo que procura maximizar o reforço das boas ações executadas pela rede.

   O processo de treinamento da rede é realizado tipicamente por tentativa e erro, pois a única resposta disponível para determinada entrada é se esta é satisfatória ou não. Se for considerada satisfatória, incrementos nos pesos sinápticos e limiares são então gradualmente efetuados visando recompensar esta condição comportamental. [Silva 2010]

b) Aprendizagem não-supervisionada

   Neste tipo de aprendizagem não existe qualquer professor ou crítico que ajude no treinamento da rede. Em vez disso, são dadas condições para a rede se auto organizar, de acordo com uma medida de representação e os parâmetros livres da rede são otimizados em relação a esta medida.

   Durante processo de aprendizado os padrões de entrada são apresentados continuamente à rede, e a existência de regularidades nesses dados faz com que o aprendizado seja possível. Regularidade e redundância nas entradas são características essenciais para haver aprendizado não-supervisionado. [Braga 2007]

7. Regras de Aprendizagem

As regras de aprendizagem definem como ocorrerá o ajuste dos pesos entre os sucessivos ciclos de treinamento de uma rede neural. [Basheer e Hajmeer 2000]

Os tipos básicos de regra de aprendizagem são: aprendizagem por correção de erro; aprendizagem baseada em memória; aprendizagem hebbiana; aprendizagem competiviva e a aprendizagem de boltzman.

A aprendizagem por correção de erro utiliza o paradigma de aprendizagem

supervisionada. Seu princípio básico é usar o sinal de erro para modificar os pesos a fim de reduzir o erro gradualmente. A regra de aprendizagem do perceptron e do backpropagation é baseado neste princípio. [Jain, Mao e Mohiuddin 1996]

Na aprendizagem baseada em memória todas as experiências passadas são armazenadas explicitamente em uma grande memória de exemplos de entrada-saída. O método utilizado é o método do vizinho mais próximo.

A aprendizagem hebbiana trouxe uma grande avanço para a área de redes neurais, nesta aprendizagem se dois neurônios em ambos os lados de uma sinapse são ativados simultaneamente, então a força daquela sinapse é seletivamente aumentada. Se dois neurônios em ambos os lados de uma sinapse são ativados assincronamente, então aquela sinapse é seletivamente enfraquecida ou eliminada. [Haykin 2001]

Quando falamos em aprendizagem competitiva, a ideia é que, neste caso, dado um padrão de entrada, fazer com que as unidades de saída disputem entre si para serem ativadas. Existe uma competição entre as unidades de saída para decidir qual delas será vencedora e, consequentemente terá sua saída ativada. A unidade vencedora tem seus pesos atualizados no treinamento. [Braga 2007]

A aprendizagem de boltzman é um algoritmo de aprendizagem estocástico derivado de ideias enraizadas na mecânica estatística. Uma rede neural projetada com base na regra de aprendizagem de boltzman é denominada uma máquina de boltzman.

O objetivo de boltzman aprendizagem é ajustar os pesos de conexão para que os estados das unidades visíveis satisfaçam uma determinada distribuição de probabilidade desejada. [Jain, Mao e Mohiuddin 1996]

8. Construindo RNAs em java com Encog

O Encog é um framework da Inteligência Artificial para Java que suporta não só redes neurais como outras áreas da IA. É uma estrutura de aprendizagem de máquina avançada que suporta uma variedade de algoritmos avançados, bem como métodos de apoio para normalizar e processar dados. Algoritmos de aprendizagem de máquina, como Support Vector Machines, Redes Neurais Artificiais, programação genética, redes Bayesianas, Hidden Markov modelos e algoritmos genéticos são suportados. [Heaton 2011]

8.1. Instalação do Encog

Para instalar é necessário baixar a última versão do Encog no seguinte link:

http://www.heatonresearch.com/encog/

Depois é só extrair os seguintes arquivos em um local desejado e incluir o caminho das bibliotecas ao usar uma IDE.

- The Encog Core
- The Encog Examples
- The Encog Workbench

Para mais informações acesse: http://www.heatonresearch.com/wiki/Getting_Started

8.2. Construindo uma rede neural

As redes neurais no Encog são constituídas por camadas, sinapses, propriedades e uma classe lógica neural. Uma camada é uma coleção de neurônios semelhantes; a sinapse liga uma camada a outra; as propriedades de uma rede neural definem as qualidades únicas que um tipo de rede neural pode ter; a classe lógica neural define como a saída da rede neural deve ser calculada; e as funções de ativação medem a saída de uma camada antes de atingir a camada seguinte.

Para exemplificarmos, usaremos o problema do XOR na criação de uma rede neural simples. Aqui usaremos uma rede feedforward com o algoritmo backpropagation para treinamento da rede neural.

O algoritmo de treinamento backpropagation segue os seguintes passos para ajuste dos pesos:

1. Apresentação de um padrão X à rede, o qual fornece uma saída Y;

2. Cálculo do erro (diferença entre o valor desejado e a saída) para cada saída;

3. Determinação do erro retropropagado pela rede associado à derivada parcial do erro quadrático.

4. Ajuste dos pesos de cada elemento;

5. Por fim, um novo padrão é apresentado a rede e o processo é repetido até que ocorra a convergência, ou seja, (erro < tolerância estabelecida) ou o número de interações corresponda a um valor máximo estabelecido.

No Encog é preciso definir o vetor de entrada e de saída desejada da rede neural, a camada de entrada da rede é definida por um vetor *double* onde o tamanho significa a quantidade de neurônios.

```
//Entrada necessária para o XOR
public static double XOR_INPUT[][] = {{1.0,0.0}, {0.0,0.0}, {0.0,1.0}, {1.0,1.0}};
// Dados ideais necessários para XOR
public static double XOR_IDEAL[][] = {{1.0}, {0.0}, {1.0}, {0.0}};
```

Depois de definido os dados do treinamento, uma rede neural básica é criada através das classes *BasicNetwork* e *BasicLayer.*

```
//Cria a rede neural
BasicNetwork network = new BasicNetwork();
network.addLayer(new BasicLayer(null,true,2));
network.addLayer(new BasicLayer(new ActivationSigmoid(), true, 2));
network.addLayer(new BasicLayer(new ActivationSigmoid(),true,1));
network.getStructure().finalizeStructure();
network.reset();
```

Aqui na estrutura da rede neural chamada *network* são adicionados a camada de entrada com 2 neurônios, uma camada oculta com 2 neurônios e a camada de saída com 1 neurônio, através do método *addLayer*. Por fim, é informada a estrutura da rede que não serão mais adicionados camadas na rede através do método *finalizeStructure*. O

método *reset* torna aleatório os pesos das redes entre as camadas.

O treinamento é feito através da classe *MLDataSet*, o objeto irá conter o conjunto de treinamento. A implementação do treinamento então é feito através do algoritmo Backpropagation que pode ser implementado da mesma maneira pela interface *Train.* Em seguida é inicializado o número de épocas e a interação de treinamento da rede. O teste da rede neural é realizado pela classe *MLDataPair* que contém os pares de entrada e saída do treinamento. A classe *MLData* computa a saída da rede, dado a rede treinada e os respectivos dados de entrada somente. Por fim , são exibidos os dados de saída ideal e os dados reais recebido pelo objeto *MLData.*

```
//Cria dados de treinamento
MLDataSet trainingSet = new BasicMLDataSet(XOR_INPUT, XOR_IDEAL);
//Treinamento da rede neural
final Backpropagation train = new Backpropagation(network,trainingSet,
0.7, 0.3);
//Inicializo o numero de épocas
int epoch = 1;

//Interação de treinamento da rede
do {
  train.iteration();
  System.out.println("Epoch #" + epoch +"Error: " + train.getError());
  epoch++;
} while(train.getError() > 0.01);

//Teste da rede neural
System.out.println("Neural Network Results:");
for (MLDataPair pair: trainingSet){
   final MLData output = network.compute(pair.getInput());
   System.out.println(pair.getInput().getData(0) + ","
                      + pair.getInput().getData(1) + ", actual="
                      + output.getData(0) + ", ideal="
                      + pair.getIdeal().getData(0));
}
Encog.getInstance().shutdown();
```

Referencias

Basheer, I. A., & Hajmeer, M. (2000). Artificial neural networks: fundamentals, computing, design, and application. *Journal of microbiological methods*, *43*(1), 3-31.

Barone, D.A.C. (2003), Sociedades artificiais: a nova fronteira da inteligência nas máquinas, Bookman.

Braga, A. P. (2007), Redes neurais artificiais: teoria e aplicações, LTC, 2ª edição.

Jain, A. K., Mao, J., & Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, *29*(3), 31- 44.

Heaton, J. (2011), Programming Neural Networks with Encog3 in Java, Heaton Research.

Haykin, S. (2001), Redes neurais: princípios e prática, Bookman, 2ª edição.

Silva, I.N. (2010), Redes neurais artificiais: para engenharia e ciências aplicadas, Artliber.

Silva, L.N.C. (1998). "Análise e síntese de estratégias de aprendizado para redes neurais artificiais." Dissertação (Mestrado em Engenharia Elétrica). Universidade Estadual de Campinas.

Tafner, M.A. (1999). "Estagiamento Automático do Sono Utilizando um Canal de EEG e uma Rede Neural Artificial com Alta Representação Cortical." Tese (Doutorado em Engenharia de Produção). Universidade Federal de Santa Catarina, http://www.eps.ufsc.br/teses99/tafner/, Outubro.