3
Redes Neurais Artificiais

3.1.
Introdução

A capacidade de implementar computacionalmente versões simplificadas de neurônios biológicos deu origem a uma sub-especialidade da inteligência artificial, conhecida como Redes Neurais Artificiais (RNAs), que podem ser definidas como sistemas paralelos, compostos por unidades de processamento simples, dispostas em camadas e altamente interligadas, inspiradas no cérebro humano [18].

As RNAs têm sido aplicadas em tarefas de reconhecimento de padrões, processamento de imagem, modelagem empírica, otimização, controle de processos industriais e de processamento de dados. O uso destas redes oferece propriedades e capacidades úteis, tais como a não-linearidade, adaptatividade, tolerância a falhas e mapeamento entrada/saída.

Este capítulo fornece uma rápida revisão sobre redes neurais artificiais, englobando seus tipos, características, treinamento e operação.

3.2.
O Modelo do Neurônio Artificial

A origem da teoria de RNA remonta aos modelos matemáticos e aos modelos de engenharia de neurônios biológicos. A Figura 3 representa o neurônio de uma RNA, onde se destacam os sinais de entrada x_j, os pesos associados às sinapses w_{kj}, o limiar (bias) θ_k, a função de ativação $\varphi(.)$, os sinais de saída y_k e o combinador linear da saída Σ [19]. Em termos matemáticos, pode-se descrever o neurônio *k* através das equações (4) e (5):

$$u_k = \sum_{j=1}^{p} w_{kj}\, x_j + \theta_k \tag{4}$$

$$y_k = \varphi(u_k) \tag{5}$$

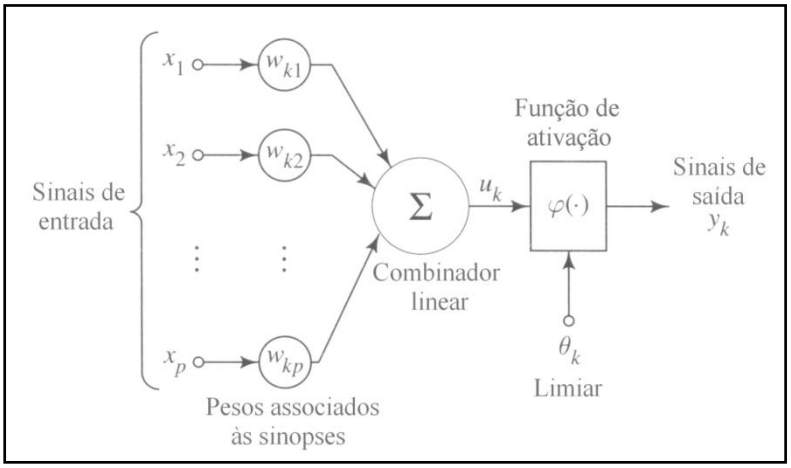

Figura 3 – Modelo matemático do neurônio de uma RNA [19].

A função de ativação é aquela que processa o sinal gerado pela combinação linear das entradas e dos pesos das sinapses, para gerar o sinal de saída do neurônio. São apresentados os três tipos de função de ativação mais utilizados, cujos gráficos são apresentados na Figura 4:

- Função de limiar: a saída do neurônio é igual a zero, quando seu valor for negativo e 1, quando seu valor for positivo.

$$\varphi(v) = \begin{cases} 1 \; se \; v \geq 0 \\ 0 \; se \; v < 0 \end{cases} \tag{6}$$

- Função de limiar por partes: esse tipo de função pode ser visto como uma aproximação de um amplificador não linear.

$$\varphi(v) = \begin{cases} 1 \; se \; v \geq 0{,}5 \\ v/2 \; se + 0{,}5 > v > -0{,}5 \\ 0 \; se \; v \leq -0{,}5 \end{cases} \tag{7}$$

- Função sigmóide: é o tipo de função de ativação mais utilizado em redes neurais artificiais. É definida como uma

função crescente, que apresenta um balanço entre o comportamento linear e não-linear. Um exemplo de função sigmóide é a função tangente hiperbólica, definida pela equação (8):

$$\varphi(v) = \frac{1 - e^{-av}}{1 + e^{+av}} \tag{8}$$

Onde *a* é o parâmetro de inclinação da função.

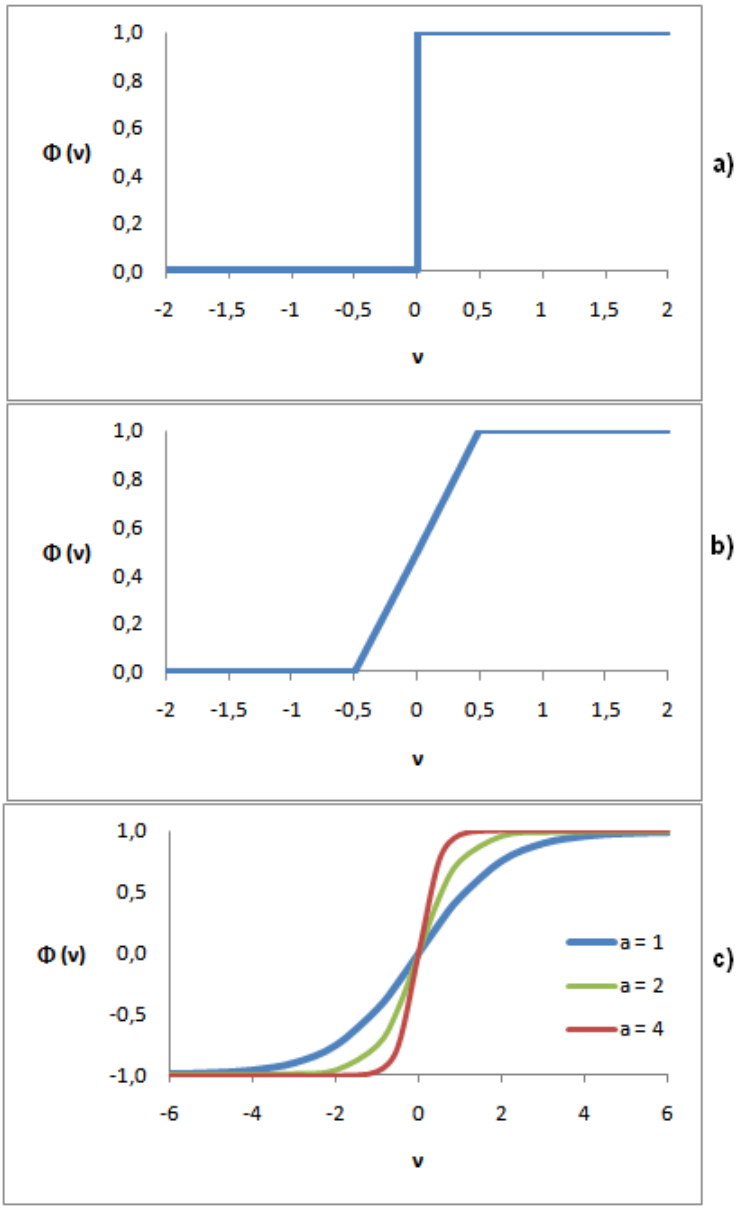

Figura 4 – (a) Função limiar, (b) Função limiar por partes e (c) Função sigmóide com parâmetro de inclinação a variável.

Os neurônios são dispostos em camadas, de forma que a camada que recebe os sinais de entrada e a camada da qual se extraem os sinais de saída são denominadas camadas visíveis, de entrada e de saída, respectivamente, e as demais camadas intermediárias, caso existentes, são denominadas de camadas escondidas [18].

Sob uma perspectiva funcional, as RNAs se distinguem em função da forma como se dá seu treinamento, se com ou sem supervisão, bem como da topologia que apresentam, ou seja, o número de camadas nas quais neurônios são dispostos e de como essas diversas camadas são interligadas.

O treinamento de uma RNA é um método de alterar os pesos associados às sinapses, w_{kj}, de maneira que estes convirjam de forma tal que a RNA possa representar um modelo matemático ou realizar a classificação do objeto em estudo. Esses treinamentos podem ser com ou sem supervisão [18].

3.3. Tipos de Redes Neurais

As redes neurais artificiais vêm sendo desenvolvidas em vários tipos, sendo que as suas características as diferem umas das outras. Elas normalmente costumam ser classificadas de acordo com suas arquiteturas e seus algoritmos de treinamento.

A arquitetura de uma RNA é o modo como todos os neurônios estão estruturados e organizados. O algoritmo de treinamento é um conjunto de regras bem-definidas para a solução do aprendizado da rede neural e melhoria do seu desempenho. Com exceção de algumas redes neurais, a maioria das RNAs tem duas diferentes fases na prática: treinamento e validação.

Na fase de treinamento, a rede ajusta seus pesos através do uso de um algoritmo apropriado, como apresentado na Figura 5. A saída da RNA é comparada com a saída desejada (alvo), e então o resíduo é usado

para ajustar os pesos sinápticos de acordo com o algoritmo de treinamento. Na fase de validação, a RNA calcula a saída com base nas entradas e nos pesos.

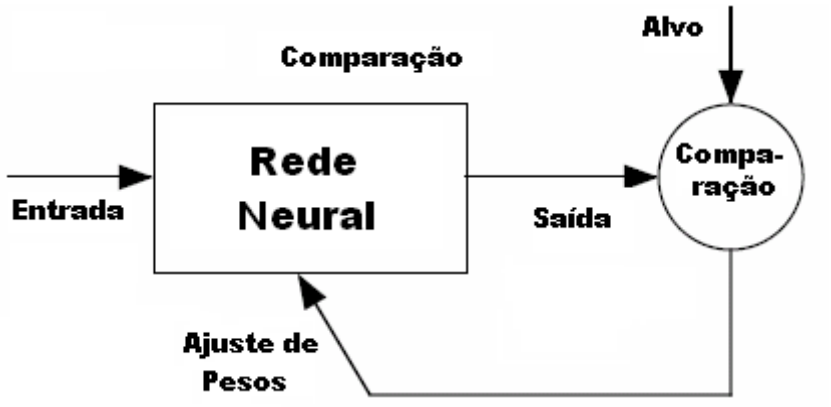

Figura 5 – Ilustração do treinamento de uma RNA.

No geral são três as classes de arquiteturas de rede [18]:

1. Rede com propagação progressiva de uma camada (*Single-Layer Feed Forward Network*): é composta por uma camada de entrada e outra de saída e também é conhecida como Perceptron (Figura 6).

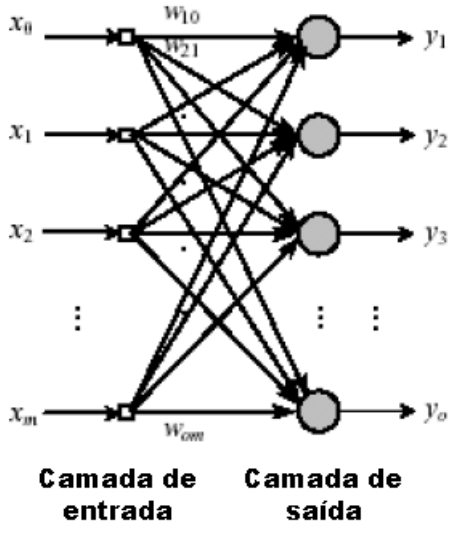

Figura 6 – Redes com propagação progressiva de uma camada (perceptron).

2. Rede com propagação progressiva de múltiplas camadas (*Multilayer Feed Forward Network*): é composta por uma

camada de entrada, uma ou mais camadas ocultas e uma camada de saída.

3. Rede recorrente (*Recurrent Network*): rede com realimentação, ou seja, a saída de um neurônio pode ser entrada para outro de uma camada precedente ou, no caso de auto-realimentação, para o próprio neurônio.

A arquitetura de rede usada neste trabalho foi a de rede com propagação progressiva, também conhecida como *Multilayer Perceptron* (MLP), que será descrita na próxima seção, 3.3.1. A escolha da MLP justifica-se principalmente por este tipo de rede ser considerado um aproximador universal de funções [20,21,22,23]. Maiores detalhes sobre as demais arquiteturas podem ser encontradas em [18].

3.3.1. Redes Multilayer Perceptron (MLP)

As redes do tipo MLP são redes de múltiplas camadas, formadas por uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída, como pode ser visto na Figura 7. Cada neurônio de uma camada recebe os sinais de todos os neurônios da camada anterior e propaga sua saída a todos os neurônios da camada posterior [19].

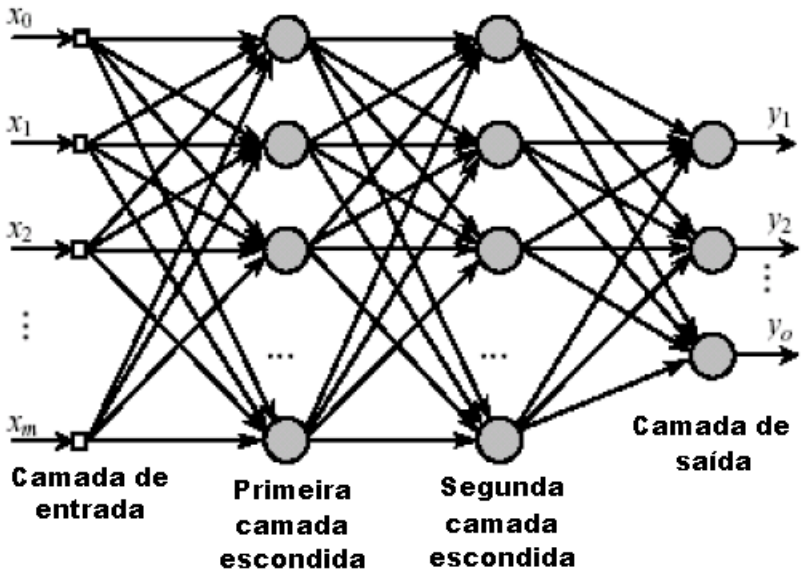

Figura 7 – Rede do tipo MPL [19].

A camada de entrada é utilizada para receber os dados de entrada da rede. Essa camada, portanto, não efetua qualquer tipo de processamento, servindo apenas para receber e armazenar o vetor de entrada. O número de neurônios nessa camada corresponde ao vetor de entrada.

A camada de saída tem a função de armazenar as respostas obtidas pela rede. Além disso, esta camada também pode proporcionar não-linearidade. O número de neurônios nessa camada corresponde ao vetor de saída.

Entre a camada de entrada e a de saída, pode-se ter uma ou mais camadas ocultas. As camadas ocultas proporcionam complexidade e não-linearidade para a rede. Não existe um método que determine o número ideal de camadas ocultas e de neurônios em cada uma dessas camadas [18].

3.4. Algoritmos de Treinamento

O objetivo principal do treinamento de uma RNA é fazer com que a aplicação de um conjunto de entradas produza um conjunto de saídas desejadas ou, no mínimo, um conjunto de saídas consistentes [19].

O treinamento é realizado pela aplicação seqüencial dos vetores de entradas (e em alguns casos também os de saída), enquanto os pesos da rede são ajustados de acordo com um procedimento de treinamento pré-determinado. Durante o treinamento, os pesos da rede gradualmente convergem para determinados valores, de maneira tal que a aplicação dos vetores de entrada produzam as saídas necessárias. Os procedimentos de treinamento das RNAs podem ser separados em duas classes:

- Supervisionado;
- Não supervisionado.

O treinamento supervisionado necessita de um vetor de entrada e um vetor de saída, conhecido como vetor alvo. Esses dois vetores são então utilizados para o treinamento da RNA. O procedimento de treinamento funciona da seguinte maneira: o vetor de entrada é aplicado, a saída da rede é calculada e comparada com o correspondente vetor alvo. O erro encontrado então é realimentado através da rede e os pesos são atualizados de acordo com um algoritmo determinado a fim de minimizar este erro. Este processo de treinamento é repetido até que o erro para os vetores de treinamento alcance valores pré-determinados [19].

O treinamento não-supervisionado, por sua vez, não requer vetor alvo para as saídas e, obviamente, não faz comparações para determinar a resposta ideal. O conjunto de treinamento modifica os pesos da rede de forma a produzir saídas que sejam consistentes.

O processo de treinamento de ambas as classes extrai as propriedades estatísticas do conjunto de treinamento e agrupa os vetores similares em classes, onde a aplicação de um vetor de uma determinada classe à entrada da rede produzirá um vetor de saída específico.

O algoritmo de retropropagação é o utilizado para o treinamento das RNAs desenvolvidas neste trabalho e será apresentado a seguir.

3.4.1. Algoritmo de Retropropagação

O algoritmo de retropropagação é um algoritmo supervisionado que utiliza pares (entrada e saída desejada) para, por meio de um mecanismo de correção, ajustar os pesos sinápticos da rede neural. O treinamento ocorre em duas fases distintas conhecidas como *forward* e *backward.* Na fase *forward*, é definida a saída da rede para um dado padrão de entrada e, na fase *backward*, a diferença entre a saída desejada e a saída atual é utilizada pela rede para atualizar os pesos de suas conexões.

A implementação computacional do algoritmo de retropropagação apresenta os seguintes passos [24]:

Passo 1: Inicialização de todos os pesos e parâmetros.

Passo 2: Fase *forward*, que consiste em:

- Dadas as entradas, calcular as saídas para todas as camadas da rede;
- Calcular o erro de saída da rede.

Passo 3: Fase *backward*, que consiste em:

- Efetuar o cálculo das atualizações dos pesos entre camadas da rede, iniciando a partir da última camada, até chegar à camada de entrada.

O algoritmo de retropropagação consiste na utilização dos pesos, considerando a propagação do erro de saída da rede para a sua entrada. Este algoritmo também é conhecido como regra delta generalizada [18,21].

Uma descrição mais aprofundada do algoritmo de retropropagação e de outros algoritmos de treinamento podem ser encontrados em [18].

3.5. Desempenho de uma RNA

De forma a avaliar os resultados de uma rede neural, o erro médio quadrático (MSE – *Mean Square Error*) é normalmente usado como indicador de desempenho da rede. A equação (9) expressa esta função:

$$MSE = \frac{1}{N}\sum_{j=1}^{M}\sum_{k=1}^{N}(y_{jk} - y_{d\,jk})^2 \qquad (9)$$

Onde *M* é o número de saídas da rede, *N* é o número de amostras de treinamento ou validação, y_j é a saída da rede, $y_{d\,j}$ é o valor de saída

desejado e *MSE* é o erro médio entre a saída da rede y_j e os valores desejados y_{dj}.

Durante o treinamento, o *MSE* normalmente começa com valores altos e então decresce conforme aumenta o número de épocas. Ao final, o *MSE* se mantém praticamente constante e a rede neural converge.

A capacidade de generalização de uma rede pode ser avaliada através dos exemplos de validação que não foram usados durante o treinamento. Quando uma rede é treinada excessivamente, a mesma pode perder a sua capacidade de generalização. Este fenômeno é denominado super treinamento (*overfitting*), aonde o erro durante os treinos são bastante pequenos, porém podem ser grandes quando novos dados são apresentados à rede [9].

De forma a projetar uma RNA com bom poder de generalização, pode-se usar um conjunto de validação durante o processo de treinamento de forma a determinar qual o momento exato de se interromper o mesmo. Este método é conhecido como o Método da Parada Antecipada (*Early Stopping*), o qual testa a rede com os dados de validação após cada época. O ponto de parada antecipada é dado na época aonde o MSE de validação começa a crescer, como ilustrado na Figura 8.

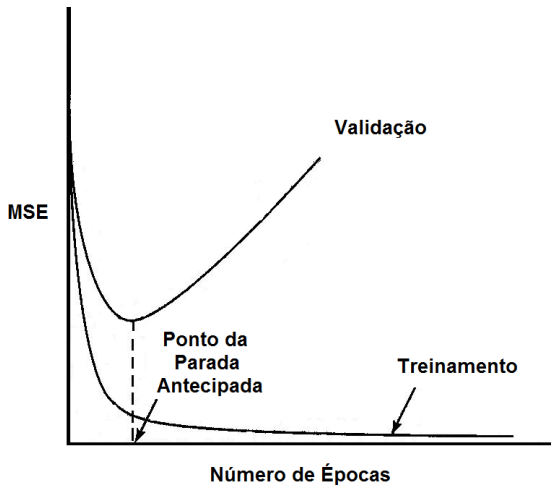

Figura 8 – Método da para antecipada com conjunto de validação [9].