4
Redes Neurais Artificiais

"Inteligência computacional pode ser definida como um conjunto de modelos, algoritmos, técnicas, ferramentas e aplicações em sistemas computadorizados que emulem características das habilidades cognitivas do homem" [24]. Inteligência computacional envolve áreas de conhecimento como ciência da computação, matemática, neurofisiologia, lingüística, dentre outras, com o objetivo de desenvolver sistemas capazes de imitar e/ou entender aspectos do pensamento humano. As Redes Neurais Artificiais, empregadas neste trabalho, são técnicas de inteligência computacional que vêm sendo aplicadas a uma enorme gama de problemas com sucesso. Dentre as principais áreas de aplicação das Redes Neurais Artificiais podemos citar sistemas de controle [25], reconhecimento de padrões [26] e aproximação de funções [27]

Redes Neurais Artificiais são sistemas computacionais com processamento altamente paralelo e distribuído e que apresentam a capacidade de aprender e armazenar conhecimento experimental. Estes sistemas computacionais apresentam características similares e análogas as observadas no funcionamento do cérebro humano, destacando-se :

1. Conhecimento é adquirido ou "aprendido" pela rede neural, como no cérebro humano, através de um processo de aprendizado.
2. Pesos existentes nas conexões entre neurônios artificiais, análogos às sinapses em neurônios biológicos, são responsáveis por armazenar o conhecimento aprendido.
3. Apresenta como o cérebro humano, a capacidade de generalização baseada no conhecimento aprendido. Isto significa que uma rede neural consegue produzir saídas adequadas para entradas não observadas no seu processo de treinamento.

4.1.
Modelo de neurônio artificial

O elemento processador básico de uma rede neural artificial é o neurônio artificial. A figura 4.1 apresenta um modelo do neurônio artificial.

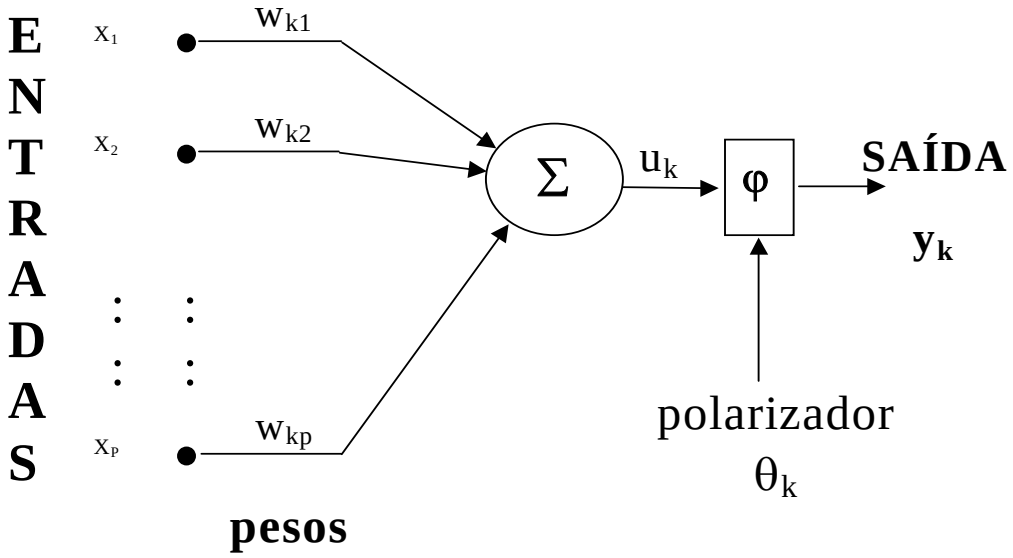

Figura 4-1 : Neurônio Artificial

O modelo do neurônio artificial possui os seguintes componentes principais :

1. Um conjunto de *sinapses* ou conexões de entrada, sendo cada entrada ponderada por um peso sináptico.
2. Uma junção de soma, responsável pela combinação aditiva das entradas ponderadas pelos pesos sinápticos.
3. Uma função de ativação, geralmente não-linear, responsável pela ativação da saída ou resposta do elemento processador.

A função básica do neurônio artificial, ou elemento processador, é realizar o somatório, ponderado por fatores denominados pesos sinápticos, dos elementos do vetor de entrada e aplicar este resultado como entrada de uma função não linear denominada função de ativação. Três classes de função de ativação, representadas na figura 4.2, são usualmente utilizadas:

1. Função Sinal : para este tipo de função temos :

   F(x) 1 se x > 0

   0 se x ≤ 0

2. Função linear por partes: para esta função temos :

   F(x) 0 se x ≤ 0

   x + a se -a < x < a

   1 se x $\geq$ a

3. Função Sigmoidal : é a função de ativação mais utilizada em redes neurais artificiais. Definida como uma função monotônica crescente que apresenta propriedades assintóticas e de suavidade. Um exemplo de função simoidal é a função logística definida por :

   $F(x) = \frac{1}{1+e^{-ax}}$

   onde a é o parâmetro de inclinação da função.

   Em muitos casos é interessante que a saída da função sigmoidal varie entre –1 e 1. Nestes casos utiliza-se a função tangente hiperbólica dada por :

   $F(x) = \text{tgh}\ \frac{x}{2} = \frac{1-e^{-x}}{1+e^{-x}}$

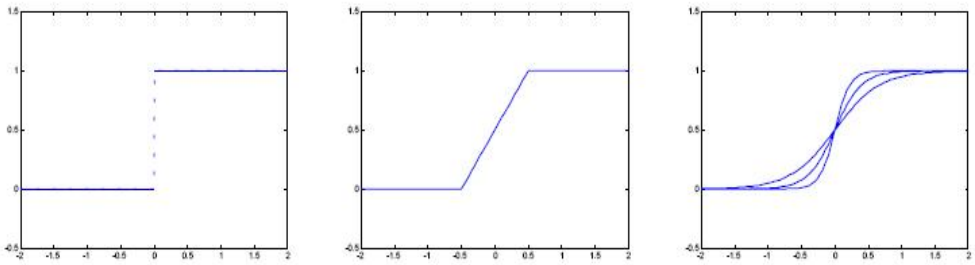

Figura 4-2 : Funções de ativação: (a) Sinal. (b) Linear por partes. (c) Sigmoidal

4.2.
Topologia e Arquitetura de Redes Neurais

4.3.
Redes Neurais Multicamadas

Redes neurais multicamadas são arquiteturas em que neurônios são organizados em duas ou mais camadas. A figura 4.3 representa um esquema típico de uma rede neural artificial com múltiplas camadas. As entradas são conectadas aos elementos processadores básicos, que são por sua vez interconectados com elementos de outras camadas e/ou a saída da rede.

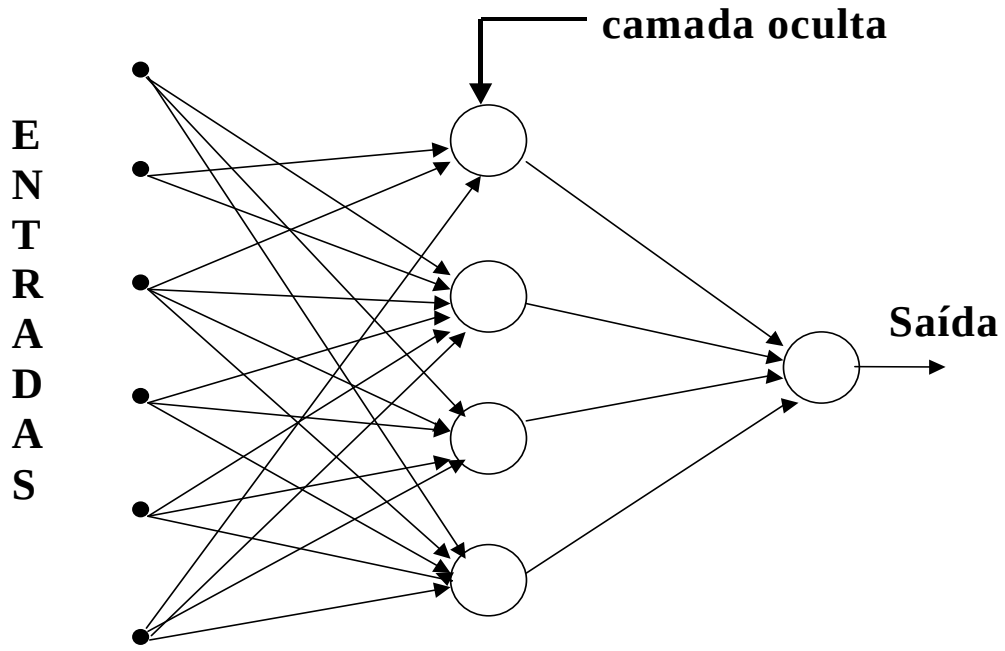

Figura 4-3 : Redes Multicamadas

A camada de entrada é especial pois não realiza qualquer processamento. Ela apenas distribui os valores de entradas para todos os neurônios da 1ª camada de processamento.

4.3.1.
Redes Neurais Multicamadas Perceptron (MLP)

O modelo de rede neural multicamadas mais simples é aquele em que a camada de entrada – não realiza processamento – se conecta a camada de saída. Este modelo, com apenas um neurônio na camada de saída recebe a denominação de Perceptron. Objeto de intensa pesquisa nas décadas de 50 e 60, o modelo Perceptron tem uso limitado pois, conforme foi provado matematicamente em [28], somente é capaz de solucionar problemas linearmente separáveis. A interconexão das entradas à camada de saída por pelo menos uma camada de neurônios intermediária, tipicamente denominada de camada escondida, remove a limitação do Perceptron e amplia o leque de problemas solucionados por estas redes. Definido um vetor de entradas $\mathbf{x} = [x_0,x_1,x_2,x_3,...x_N]$ e um vetor de saídas $\mathbf{y} = [y_0, y_1, y_{2,...} y_N]$, uma rede neural multicamadas realiza um mapeamento complexo $\mathbf{y} = \Psi(\mathbf{w},\mathbf{x})$, da entrada $\mathbf{x}$ na primeira camada a saída $\mathbf{y}$, parametrizado pelos pesos sinápticos $\mathbf{w}$. Redes multicamadas são, portanto, ferramentas poderosas. Dado um número suficiente de neurônios elas conseguem aproximar qualquer função linearmente contínua.

Uma rede MLP – “Multilayer Perceptron” – apresenta, segundo [27], 3 características principais :

1. O modelo de cada neurônio ou elemento processador da rede possui uma função de ativação não-linear. A função sigmoidal atende esta exigência e é muito utilizada em redes MLP.
2. A rede possui pelo menos uma camada oculta de processamento com neurônios que não fazem parte da entrada ou da saída.
3. A rede possui alto grau de conectividade entre seus elementos processadores. Esta conectividade é definida pelos pesos sinápticos.

Definição do tamanho da rede

Projetar uma rede neural MLP para um problema específico envolve determinar quantas camadas e, principalmente, quantos neurônios artificiais ocultos a rede deve possuir. O tamanho das camadas de entrada e saída será sempre determinado pela natureza do próprio problema. Como exemplo prático e ligado ao problema aqui estudado, uma rede neural MLP desenvolvida para

classificar registros de conexões com características intrínsecas ao TCP/IP apenas em intrusão ou normal terá 9 entradas – correspondentes aos nove campos existentes e apresentados na tabela 5.1 - e apenas 2 saídas. Determinar, quantos neurônios ocultos e quantas camadas uma rede que procure solucionar com precisão este problema é tarefa que não possui resposta exata. Existem, entretanto, soluções aproximadas – heurísticas – que procuram estimar estas variáveis. Estas heurísticas expõem sempre o compromisso entre a convergência e a generalização da rede.

1. Convergência : é a capacidade da rede de aprender todos os padrões de entrada usados no seu treinamento. Uma rede muito pequena – em relação ao problema em análise – não será capaz de aprender os dados de treinamento do problema – ou seja, a rede não possuirá parâmetros ou pesos sinápticos suficientes.
2. Generalização : é a capacidade da rede neural responder adequadamente a padrões jamais vistos. Uma rede muito grande – com número de neurônios muito superior aos necessários para o problema em análise – não responderá corretamente a padrões nunca vistos – perderá a capacidade de generalizar. Isto ocorrerá pois os pesos sinápticos da rede aprenderão os vetores de entrada e também o ruído presente nos dados de treinamento.

A capacidade de generalização de uma rede neural é afetada por três fatores :

1. Tamanho e eficiência dos dados de treinamento.
2. Arquitetura da rede e número de processadores nas camadas ocultas.
3. Complexidade do problema.

Normalmente não se tem controle sobre o 3º fator apresentado (complexidade do problema). O problema pode ser investigado de duas maneiras :

1. Fixa-se arquitetura e tamanho da rede e determina-se qual o tamanho dos dados de treinamento necessário.

2. Tamanho do conjunto de treinamentos é fixo e deve se achar a melhor arquitetura e tamanho da rede neural para o problema em questão.

Para nosso estudo temos um conjunto de dados de treinamento extenso porém fixo. Iremos então, utilizar as seguintes heurísticas para determinar o tamanho das redes empregadas:

1. Hecht-Nielsen [29] : demonstraram, a partir do teorema de Kolmogorov que qualquer função de *n* variáveis pode ser representada por 2*n* + 1 funções de uma variável. Tem-se que :

   $$\mathbf{N_{hidden} = 2N_{in} + 1}$$

   onde N_{hidden} = número de neurônios ocultos e N_{in} = número de entradas.
2. Upadhyaya-Eryurek [29] : qualquer padrão binário P pode ser representando por $\log_2 P$ parâmetros. Logo :

   $$\mathbf{w \leq N_{in} \log_2 P}$$

   onde w = número de pesos sinápticos e P = número de padrões para treinamento.
3. Baum-Haussler [29] : heurística que associa o erro esperado da rede com o tamanho da rede :

   $$\mathbf{N \geq} \frac{w}{\varepsilon}$$

   onde ε = erro esperado nos testes da rede.

Na prática, as heurísticas apresentadas são utilizadas em conjunto com séries de tentativas e ajustes em arquiteturas e definições de redes.

4.4. Algoritmos de Aprendizado

Existem diversas maneiras de se classificar uma rede neural artificial. Uma das mais importantes é a classificação pelo processo ou algoritmo de aprendizado empregado para treiná-la, que pode ser supervisionado ou não-supervisionado.

4.4.1.
Aprendizado Supervisionado

No aprendizado supervisionado – representado na figura 4.4 - a rede recebe uma série de padrões ou vetores de entradas com sua respectiva resposta ou saída desejada. A comparação entre a saída desejada e a saída real gerada pela rede é utilizada para alterar os parâmetros internos (ou pesos sinápticos) da mesma de modo a aproximar sua resposta a saída desejada. Este procedimento é repetido até que a diferença entre as saídas geradas pela rede para os diversos padrões apresentados e a resposta esperada seja menor que um determinado limiar predefinido. O processo de treinamento ou aprendizado supervisionado de uma rede neural consiste, essencialmente, em minimizar o erro entre a saída da rede para um determinado padrão de entrada e a resposta esperada para aquele mesmo padrão.

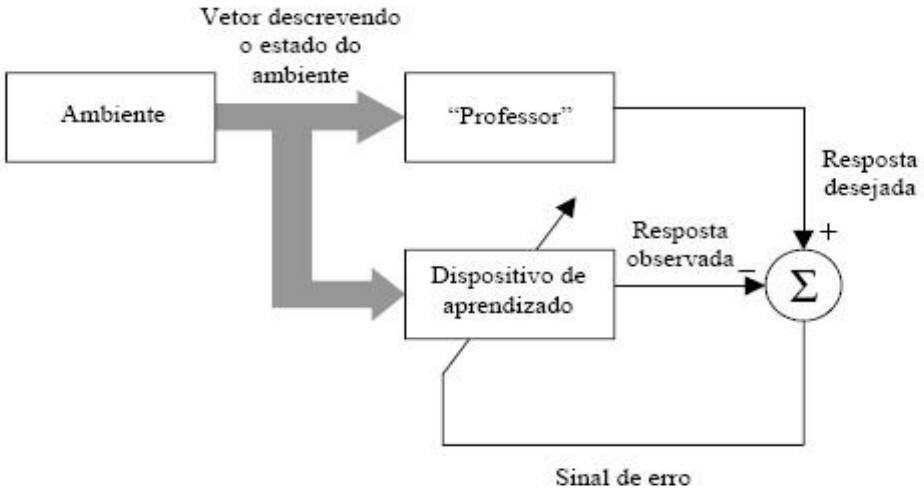

Figura 4-4 : Treinamento Supervisionado

Pontos chaves a serem destacados para o aprendizado supervisionado :

1. A rede neural é estimulada pelo ambiente através da apresentação de padrões na sua camada de entrada.
2. Sua estrutura interna é alterada em função destes padrões e da saída desejada.

3. Sua resposta futura ao estímulo do ambiente será diferente devido a alteração da estrutura interna.

Diversos algoritmos para ajuste dos parâmetros da rede, baseados neste conceito, existem como : *adaline*, *madaline*, retropropagação do erro (“backpropagation”) dentre outros.

Algoritmo de Retropropagação do Erro

O algoritmo de retropropagação do erro é o processo supervisionado mais utilizado para treinamento de redes neurais MLP. A seguir apresentamos uma versão macro do seu funcionamento.

1. Inicialização dos pesos e parâmetros da rede neural com valores aleatórios.
2. Apresentar um vetor **x** na entrada da rede neural e observar o vetor de saída **y** gerado.
3. Comparar o vetor de saída **y** gerado pela rede com o vetor **d** de resposta desejada.
4. Adaptar os pesos **w** através do algoritmo recursivo :
    a. Calcular, para cada camada da rede, começando da camada de saída e seguindo em direção a camada de entrada : $\mathbf{w_{ij}}(t+1) = \mathbf{w_{ij}}(t) + \eta\, \delta_j\, \mathbf{x'}_i$ onde :
        i. $\mathbf{w_{ij}}(t)$ é o valor do peso sináptico do neurônio j no tempo t.
        ii. η é um termo de ganho, que permite regular a velocidade dos ajustes.
        iii. Se o neurônio pertencer a camada de saída da rede neural temos :

            $\delta_j = y_j(1 - y_j)(d_j - y_j)$ onde d_j representa a saída desejada e y_j representa a saída real.
        iv. Se o neurônio pertencer a uma camada oculta temos:

            $\delta_j = x'_j\,(1 - x'_j) \sum \delta_k w_{jk}$ onde k representa todos os elementos processadores acima de j.

5. Voltar ao passo 2 e repetir procedimento até que a condição de parada seja alcançada.

O parâmetro η regula a taxa de atualização dos pesos sinápticos. Se η for pequeno o treinamento demorará mais e, dependendo dos valores de inicialização dos pesos, o processo de treinamento pode ficar preso em um mínimo local. Uma solução possível para esta situação é utilizar η adaptativo. Variações do algoritmo tradicional de retropropagação do erro utilizam esta técnica. Por outro lado, se η for muito grande as variações nos pesos poderão oscilar impedindo que o mínimo global seja alcançado. Neste caso, é empregado um termo de momento α, que reduz a possibilidade de oscilações, da seguinte forma : $\mathbf{w}_{ij}(t+1) = \mathbf{w}_{ij}(t) + \eta\, \delta_j\, \mathbf{x'}_i + \alpha(\mathbf{w}_{ij}(t) - \mathbf{w}_{ij}(t-1))$

Mais detalhes sobre o algoritmos de retropropagação do erro e suas variantes podem ser obtidos em [27].

Validação Cruzada

A essência do aprendizado da rede neural através do algoritmo de retropropagação do erro consiste em realizar um mapeamento, através dos ajustes dos pesos sinápticos, de pares entrada-saída de vetores. Espera-se, paralelamente, que a rede aprenda o suficiente com esta associação para ser capaz de encontrar associações válidas para vetores de entradas desconhecidos. A rede deve, portanto, ter capacidade de generalizar o conhecimento aprendido aplicando-o em novas situações com sucesso.

A validação cruzada é uma técnica estatística clássica [30] que é útil em determinar, durante o treinamento, a capacidade de generalização de uma rede neural. Os dados de treinamento devem ser subdivididos em dois conjuntos distintos :

1. Conjunto de Treinamento (estimação) : usado para treinar a rede neural.
2. Conjunto de Validação : usado para testar ou validar a rede treinada. Deve ser composto por entre 10% e 25% das amostras disponíveis.

Após o treinamento da rede neural por um número predefinido de épocas – a apresentação de todos os padrões de treinamento disponível corresponde a uma época - o treinamento é interrompido e a rede é testada com os dados de validação. Repete-se este processo até que o desempenho da rede com os dados de validação se estabilize em um valor considerado aceitável para o problema em análise.

A motivação para esta divisão é validar o modelo em um conjunto de dados diferentes do usado para adaptar os pesos sinápticos. Evita-se também, com o emprego desta técnica, que ocorra o fenômeno denominado “overtraining” (treinamento excessivo) da rede neural. Uma rede treinada em excesso aprende fortemente os dados de treinamento e apresenta péssima capacidade de generalizar este conhecimento.

O uso da validação cruzada é altamente recomendável quando se tem um problema com volume de dados de treinamento grande e se deseja obter boa capacidade de generalização da rede neural. São características presentes no problema aqui estudado – detecção de intrusos em redes TCP/IP – e por isto a validação cruzada foi empregada em todos os procedimentos de treinamento das redes neurais deste trabalho.

Condição de parada

É preciso estabelecer qual o critério ou condição de parada do algoritmo de treinamento. Idealmente, um algoritmo como o de retropropagação do erro procura encontrar um valor mínimo global e esta deveria ser sua condição de encerramento. Na prática isto nem sempre é possível e o algoritmo pode encontrar apenas um valor mínimo local na superfície de erro. Critérios de parada tipicamente empregados na prática :

1. Execução do algoritmo de treinamento por um número de interações predeterminado.
2. Parada quando o erro médio quadrático (MSE – Mean Squared Error) atingir determinado valor.
3. Parada quando a taxa de decréscimo no erro médio quadrático entre interações ficar menor do que um determinado valor.
4. Parada quando o erro encontrado no processo de validação (quando se utiliza validação cruzada) aumenta. A figura 4.5 ilustra esta situação.

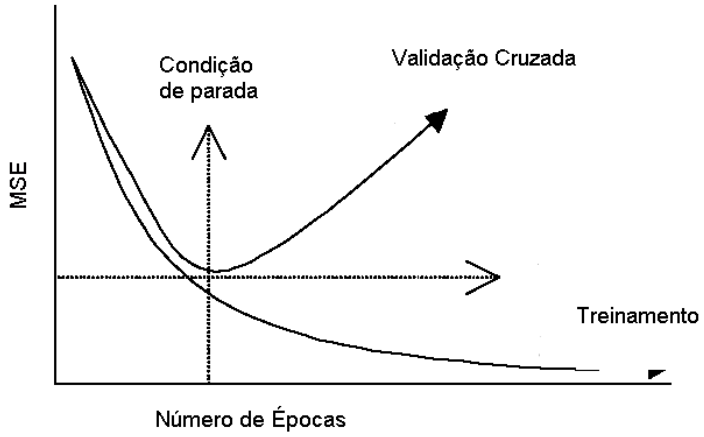

Figura 4-5 : Condição de Parada

4.4.2. Aprendizado Não-supervisionado

O aprendizado não-supervisionado não apresenta a saída desejada para a rede. Padrões são apresentados à rede neural e sua saída irá refletir características ou "conhecimento" por ela aprendido sobre os padrões de entrada. Mapa de Kohonem ou "Self-Organizing Maps", Hopfield e Memória Associativa Bidirecional são exemplos de técnicas não supervisionadas de treinamento de redes neurais. Redes treinadas com algoritmos de aprendizado não supervisionado são empregadas em problemas de agrupamento ("clustering") e mineração ("mining") de dados.

4.4.3. Aprendizado em lote ("batch")

O aprendizado das redes neurais pode ser ainda classificado de acordo com a maneira em que os dados de treinamento são apresentados na entrada da rede e os pesos são alterados.

No aprendizado em lote ou “batch” todos os vetores de entrada são apresentados a rede – completando uma época – e os pesos somente são alterados ao final deste processo. O erro passa a ser então uma função de custo formada pelo somatório do erro médio quadrático de cada padrão individual, como em :

$$E_{avg} = \frac{1}{2N}\sum_{n=1}^{N}\sum_{j\in C} e_j^2(n)$$

Onde N corresponde ao número de vetores dos dados de treinamento, $e_j(n)$ corresponde ao erro para o elemento processador *j* quando o vetor *n* é aplicado na entrada da rede.

4.4.4. Aprendizado Sequencial

Quando cada vetor de entrada é aplicado individualmente, o erro calculado e os pesos sinápticos alterados, temos um aprendizado seqüencial ou vetor a vetor. Esta modalidade, apesar de mais difícil de ter o processamento paralelizado em comparação com o aprendizado em lote, é muito utilizada devido aos seguintes benefícios :

1. Algoritmo é simples de programar.
2. Algoritmo requer menos memória / armazenamento local.
3. Apresenta vantagens quando os padrões de treinamento possuem redundâncias.