

Introdução às Redes Neurais Artificiais

Miguel Ângelo Moreira

Outubro de 1997

Resumo

O presente texto pretende compilar o resultado de um estudo preliminar sobre redes neurais artificiais.

Índice

1	Introdução e Motivação	3
2	Generalidades	3
3	Caracterização das NN	6
3.1	Classificação quanto à arquitectura	6
3.1.1	Redes planares	6
3.1.2	Redes constituídas por camadas	6
3.2	Classificação quanto ao processo de aprendizagem	7
3.2.1	Aprendizagem com supervisão	7
3.2.2	Aprendizagem sem supervisão	8
3.3	Classificação quanto à dinâmica	8
4	Arquitecturas de redes e suas aplicações	9
5	Aprendizagem das redes	9
5.1	Aprendizagem com supervisão	9
5.1.1	Aprendizagem de redes de perceptrões	15
5.1.2	Regra de aprendizagem de Widrow-Hoff	17
5.1.3	Regra de aprendizagem por retropopagação do erro	19
5.1.4	Método do gradiente	19
5.1.5	Método de Levenberg-Marquardt	21
5.1.6	Métodos heurísticos	23
5.2	Aprendizagem sem supervisão	23
5.2.1	Regra de Hebb	23
5.2.2	Regras de Instar e de Outstar	26
5.2.3	Algoritmos de aprendizagem competitiva	28
5.2.4	Regra de Kohonen	30

1 Introdução e Motivação

O sistema nervoso central dos animais recebe do exterior, armazena, processa e transmite informação ao exterior. A observação do desempenho que este apresenta tem revelado uma extraordinária capacidade para executar rápida e eficientemente tarefas de grande complexidade tais como o processamento em paralelo da informação, a memória associativa e a capacidade para classificar e generalizar conceitos. Estes factos têm servido de motivação quer para o estudo detalhado da constituição do cérebro quer para a sua mimetização na concepção de sistemas com as capacidades atrás referidas e designados por *redes neuronais artificiais* (*Artificial Neural Nets*).

As redes neuronais artificiais (abreviadamente, *NN*) têm sido utilizadas na modelação de memória associativa, reconhecimento de padrões, representação de funções booleanas, representação de funções contínuas, previsão de séries temporais, optimização etc. Refiram-se algumas das áreas em que são aplicadas:

- (a) Física de alta energia;
- (b) Processos industriais, robótica, indústria de defesa, indústria aeroespacial, indústria de telecomunicações, indústria electrónica, indústria automóvel, indústria de transportes, indústria de prospecção petrolífera;
- (c) Medicina e biomedicina;
- (d) Reconhecimento de texto, imagem e voz.
- (e) Economia e gestão, análise financeira e banca;
- (f) Seguros;
- (g) Entretenimento.

2 Generalidades

O córtex cerebral é constituído por unidades celulares independentes designadas por *neurónios* que podem comunicar entre si através das chamadas *ligações sinápticas* ou *sinapses*. A comunicação é realizada unidirecionalmente por sinais nervosos eléctricos (predominantemente no interior de cada neurónio) e químicos (nas regiões terminais das ligações sinápticas) de uma forma descontínua, isto é, por impulsos. Estes últimos são desencadeados em cada neurónio sempre que um certo *potencial de activação* é ultrapassado em resultado da recepção de um ou mais sinais nervosos nas *sinapses*.

De referir que os sinais nervosos são, em geral, amplificados (ou pesados) de forma diferenciada ao atravessar as diferentes sinapses de um neurónio.

O primeiro modelo lógico-matemático de um neurónio foi desenvolvido por McCulloch and Pitts em 1943 [15] tendo sido designado por TLU (*Threshold Logic Unit*). Consiste numa unidade computacional que opera com sinais binários (0/1) de acordo com as seguintes regras:

1. Recebe as entradas $x_1, x_2, x_3, \dots, x_n$ em n sinapses de excitação e as entradas $y_1, y_2, y_3, \dots, y_m$ em m sinapses inibidoras;
2. Se pelo menos uma das entradas inibidoras, $y_1, y_2, y_3, \dots, y_m$ é 1 então a unidade computacional é inibida apresentando como saída o valor 0;
3. Se todas as entradas inibidoras, $y_1, y_2, y_3, \dots, y_m$ são 0 então a unidade computacional apresenta o sinal de saída $f(x_1 + x_2 + \dots + x_n - b)$ em que f é a função degrau unitário e a constante b representa um desvio (*bias*).

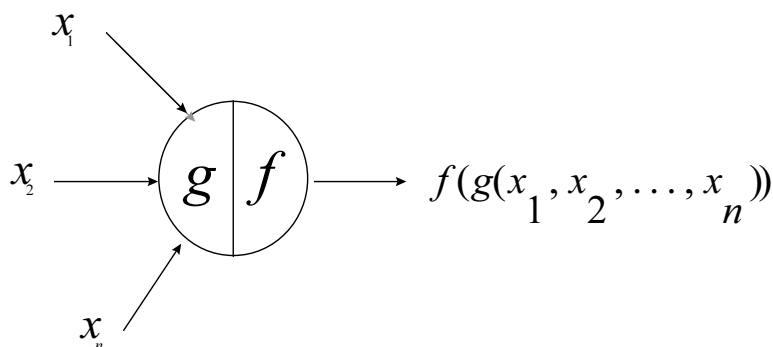


Figura 1: Modelo genérico de um neurónio artificial.

Na figura 1, apresenta-se o modelo de um neurónio artificial em que g representa um campo vectorial de \mathbb{R}^n em \mathbb{R}^p e f um campo escalar de $\mathbb{R}^p \rightarrow \mathbb{R}$. No entanto a pretensão de modelar uma unidade computacional verdadeiramente elementar aconselha a escolha de uma função g mais simples (campo escalar que traduza uma aplicação linear).

Mais formalmente as redes neuronais artificiais podem ser modeladas por um grafo orientado que possui as seguintes propriedades:

1. Uma variável de estado n_i associada a cada nó i ;

2. Um valor real w_{ik} designado por *peso* (*weight*) e associado à aresta (ik) entre os nós i e k ;
3. Um valor real b_i designado por *desvio* (*bias* ou *threshold-shift*) associado a cada nó i ;
4. Uma função de transferência $f_i(n_k, w_{ki}, b_i, [k \neq i])$ associada a cada nó i que determina o estado desse nó em função do seu desvio, dos pesos das arestas a ele ligadas e dos estados dos nós associados às anteriores arestas. Na terminologia habitual diz-se que f é função do *potencial de activação* verificado no nó.

Os *nós* e as *arestas* designam-se, respectivamente, *neurónios* (ou *unidades computacionais*) e *sinapses*. *Neurónios de entrada* são aqueles que não têm sinapses dirigidas para os mesmos. *Neurónios de saída* são aqueles que não possuem sinapses de saída.

De referir que que uma ligeira modificação da rede original (adição de número apropriado de neurónios com a entrada constante do valor 1) permite tratar os desvios acima referidos como se pesos adicionais se tratassem.

As *funções de transferência* (ou de *activação*), f , utilizadas na construção de redes neuronais artificiais têm sido, entre outras, funções em degrau, funções lineares do tipo $f(x) = x$, funções do tipo anterior mas com um contradomínio igual a um intervalo fechado (funções rampa), funções crescentes de classe C^∞ cujo contradomínio se encontra contido num certo intervalo aberto e limitado, *funções Gaussianas* (designadas por funções de *base radial*) ($f(x) = e^{-x^2}$) ou mesmo funções trigonométricas elementares ($\sin x$ e $\cos x$). De referir que as funções de transferência crescentes de classe C^∞ e cujo contradomínio se encontra contido no intervalo $[0, 1]$ (ou $[-1, +1]$) e tais que $\lim_{x \rightarrow +\infty} f(x) = 1$ e $\lim_{x \rightarrow -\infty} f(x) = 0$, (ou $\lim_{x \rightarrow -\infty} f(x) = -1$) são conhecidas por funções *sigmodais*. As funções *tangente hiperbólica* e as funções de *Fermi* ($f(x) = \frac{1}{1+e^{-x}}$) são tipicamente algumas das funções sigmodais mais utilizadas.

Em certas situações que referiremos mais à frente podem ser utilizadas nas unidades de saída de uma rede neuronal as chamadas *funções competitivas*, f_c , que basicamente activam apenas a unidade de saída mais excitada. São funções, normalmente, de contradomínio, $\{0, 1\}$. Para simplificar a notação e se representar-mos por \bar{y} o vector dos potenciais de activação verificadas nas unidades de saída de uma rede neuronal,

$$[f_c(\bar{y})]_i = \begin{cases} 1 & \text{se } y_i \text{ é máximo,} \\ 0 & \text{caso contrário} \end{cases} .$$

As aplicações das redes neuronais artificiais estão intimamente relacionadas com a sua *arquitectura (topologia e propriedades dos neurónios e sinapses)*, *processo de aprendizagem e comportamento dinâmico*.

3 Caracterização das NN

3.1 Classificação quanto à arquitectura

3.1.1 Redes planares

As redes neuronais planares são constituídas por *neurónios escondidos (hidden neurons)* e por *neurónios periféricos (peripheral neurons)* todos eles dispostos numa superfície bidimensional. Estes últimos podem subdividir-se em *neurónios de entrada (input neurons)* e *de saída (output neurons)*. A disposição dos diferentes neurónios pode ou não estar organizada por camadas. Sempre que as respostas dos neurónios de uma rede é binária (0/1 ou $-1/+1$, por exemplo) esta pode designar-se *binária*. Redes neuronais constituídas por neurónios binários e arestas binárias (isto é por onde circulam apenas sinais binários) são conhecidas por *redes McCulloch-Pitts*.

3.1.2 Redes constituídas por camadas

As redes neuronais artificiais planares podem ser constituídas por diferentes *camadas (layers)* dispostas paralelamente. A primeira é designada por *camada de entrada (input layer)* e a última por *camada de saída (output layer)*. As *camadas intermédias (hidden layers)* são designadas por *camadas escondidas*. As redes constituídas por camadas constituem casos particulares das redes planares.

Na contagem do número de camadas de uma rede alguns autores não entram em consideração com a camada de entrada. No entanto no presente texto optaremos por considerar sempre esta camada na operação da sua enumeração. Assim, uma rede com duas camadas será constituída por uma camada de entrada e por uma camada de saída.

Na numeração das camadas atribuiremos o número 0 à camada de entrada e números crescentes consecutivos às camadas seguintes.

Quando a informação de cada camada não constitui informação de entrada de nenhuma camada anterior a rede designa-se por *rede de alimentação directa (feed-forward network)*. Caso contrário designam-se por *redes recorrentes (recurrent networks)*. De referir dois importantes tipos de redes recorrentes: redes de *Elman* [7] e de *Hopfield* [14].

As redes de camadas com alimentação directa e em que a informação de saída é constituída por padrões binários são designadas por alguns autores por redes de *perceptrões*. Mais rigorosamente um *perceptrão* é considerado uma unidade computacional que calcula o *signal* duma *combinação linear* das variáveis de entrada.

Em algumas situações certas camadas possuem neurónios cujas funções transferência são lineares. Neste caso estas camadas designam-se *lineares*. Uma rede apenas com camadas lineares designa-se *rede linear*. As camadas não lineares designam-se normalmente pelo nome da função transferência comum aos diferentes neurónios utilizados nessas camadas. As redes com uma ou mais camadas não lineares podem designar-se *redes não lineares*. Redes com *camadas sigmoidais* ou *Gaussianas* constituem exemplos de redes não lineares. Redes com camadas deste último tipo podem designar-se por *redes de base radial*.

3.2 Classificação quanto ao processo de aprendizagem

3.2.1 Aprendizagem com supervisão

O processo de aprendizagem (isto é, de escolha dos pesos e deslocamentos associados a cada aresta/neurónio) de uma rede neuronal artificial pode ser realizado sob supervisão. Neste tipo de aprendizagem são conhecidas *a priori* as respostas correctas correspondentes a um certo conjunto de dados de entrada. A referir entre outros os seguintes algoritmos de aprendizagem com supervisão:

- (a) Regra de aprendizagem de *Widrow-Hoff* (ou método do gradiente aplicado em redes neuronais lineares);
- (b) Aprendizagem por *retropropagação do erro* (*error backpropagation*) que constitui uma generalização da anterior regra a redes lineares ou não lineares e com três ou mais camadas;
- (c) Método do *gradiente* e seus aperfeiçoamentos. De referir a existência de técnicas destinadas a melhorar a convergência destes métodos tais como a técnica do *momento* e da *taxa adaptativa de aprendizagem*;
- (d) A aprendizagem recorrendo ao método de *Levenberg-Marquardt* aplicável a redes não lineares;
- (e) A aprendizagem recorrendo a técnicas heurísticas, como por exemplo, os *algoritmos evolutivos* (*evolutionary algorithms*) ou a aprendizagem recorrendo ao processo de *simulação do recozimento* (*simulated annealing*).

O algoritmo utilizado pode classificar a rede em que se aplica. De referir que redes não lineares com camadas escondidas e de alimentação directa podem ser conhecidas por *redes de retropropagação e de alimentação directa* (*standart feedforward backpropagation networks*).

3.2.2 Aprendizagem sem supervisão

A aprendizagem sem supervisão é aplicada em sistemas de memória associativa e de reconhecimento de padrões, essencialmente. Nestas redes a aprendizagem é realizada sem se conhecer antecipadamente as respostas consideradas correctas. Podem ser utilizados diferentes algoritmos de aprendizagem sem supervisão, entre outros:

- (a) *Algoritmos de estimulação pela entrada* (*reinforcement algorithms*) também designados (no contexto da aprendizagem sem supervisão) por algoritmos de *aprendizagem associativa* (*associative learning algorithm*). A regra de *Hebb*, [8], as regra de *Instar* e de *Outstar* [6], constituem alguns exemplos deste tipo de algoritmos;
- (b) Algoritmos de *aprendizagem competitiva* tais como a regra de *Kohonen*, [10].

Como anteriormente, algumas redes podem ser classificadas com base no algoritmo de aprendizagem utilizado. De referir, por exemplo, as *redes de Kohonen* e as *redes competitivas*.

Observe-se que a classificação anterior não é completamente estanque na medida em que, por exemplo, a regra de Kohonen pode ser considerado um algoritmo de aprendizagem associativa.

3.3 Classificação quanto à dinâmica

A transferência de informação através das sinapses e entre os diferentes neurónios de uma rede neuronal pode processar-se *sequencialmente* (ou de uma forma *não-síncrona* ou *asíncrona*), isto é, em cada iteração só parte dos neurónios da rede transferem informação aos neurónios seguintes que por sua vez e apenas na iteração seguinte processam e transmitem a informação recebida.

Noutras situações a transferência de informação é realizada *simultaneamente* (ou de um modo *síncrono*) por todos os neurónios da rede.

Noutras situações o mecanismo de activação de cada neurónio não tem uma natureza *determinista* sendo regido por uma lei de natureza *estocástica* (*spin-glass nets*)

4 Architecturas de redes e suas aplicações

Redes de arquitectura planar, com idêntico número de neurónios de entrada e de saída e cuja aprendizagem é realizada sem supervisão são normalmente utilizadas em *sistemas de memória associativa e de reconhecimento de padrões*. A quantidade de padrões armazenados na rede está associada directamente ao número de neurónios utilizados. Podem encontrar-se redes deste tipo com neurónios escondidos. Estas redes são normalmente de dinâmica síncrona.

Na *classificação de padrões* podem ser utilizadas redes binárias de alimentação directa com uma ou duas camadas (redes de perceptrões, [21]). As chamadas *redes competitivas* podem ser igualmente utilizadas com o anterior propósito.

Na representação de qualquer *função Booleana* (ou *função lógica*) podem utilizar-se redes binárias de camadas com alimentação directa e com uma única camada escondida [5].

Na representação de qualquer *função contínua* ([11] e [12]) podem utilizar-se redes de camadas com alimentação directa e com uma camada escondida ([2] e [9]) (em geral a camada escondida pode ser *sigmoidal* e a camada de saída *linear*). Este tipo de rede pode igualmente ser utilizada na *associação e classificação de padrões*. Redes de base radial na camada escondida e lineares na camada de saída têm sido utilizadas com os anteriores propósitos [1].

Redes neuronais com uma única camada (linear) são utilizadas na *aproximação linear de funções*. Este tipo de redes são utilizadas também no *processamento de sinal* [22], *controlo e previsão e identificação de sistemas lineares*. De referir que estas redes podem igualmente ser utilizadas na *associação de padrões*.

Na previsão de *séries temporais* e *séries espaciais* associadas a fenómenos não lineares têm sido utilizadas redes não lineares de camadas com alimentação directa.

5 Aprendizagem das redes

5.1 Aprendizagem com supervisão

No processo de aprendizagem com supervisão de uma rede neuronal artificial pretende-se, conhecendo as respostas/saída correctas da rede a diferentes entradas de dados, escolher os pesos e os desvios das diferentes sinapses/arestas e neurónios que melhor contribuam para a rede modele adequadamente o fenómeno em questão.

Os algoritmos de aprendizagem com supervisão pode ser classificada nos

dois seguintes grupos, Rojas [20]:

- *Algoritmos de estimulação pela entrada (reinforcement algorithms);*
- *Algoritmos de aprendizagem correctiva (corrective learning).*

Nos primeiros, o ajuste dos pesos e desvios é efectuado com base apenas nos valores de entrada na rede ao constatar que a rede não fornece a resposta adequada. Tipicamente as redes de perceptrões utilizam uma aprendizagem com supervisão com estas características, como se verá mais à frente.

Nos segundos, no ajuste dos pesos e desvios interveem, para além do vector de entrada, o conhecimento do erro que lhe está associado.

Seguidamente formularemos o problema da aprendizagem por supervisão de uma rede com unidades diferenciáveis que nos conduzirá ao desenvolvimento de alguma notação e resultados que utilizaremos posteriormente para ilustrar diferentes algoritmos de aprendizagem correctiva.

Seja, $A = \left\{ \left({}^{(1)}\bar{x}, {}^{(1)}\bar{t} \right), \dots, \left({}^{(p)}\bar{x}, {}^{(p)}\bar{t} \right) \right\}$, o conjunto dos pares correspondentes às respostas vectorias correctas (*target*, ${}^{(i)}\bar{t}$) associadas aos diferentes vectores de entrada, ${}^{(i)}\bar{x}$, (também designado por conjunto dos *padrões de treino*), a minimização de uma apropriada função erro, dependente dos pesos e desvios da rede considerada, conduz-nos à formulação natural do problema de aprendizagem da rede.

Chamando, ${}^{(i)}\bar{o}$, ao vector de saída correspondente ao vector de entrada, ${}^{(i)}\bar{x}$, a função de erro cuja minimização é geralmente considerada consiste na semisoma do quadrado das normas Euclidianas dos resíduos, ${}^{(i)}\bar{r} = {}^{(i)}\bar{o} - {}^{(i)}\bar{t}$:

$$E = \frac{1}{2} \sum_{i=1}^p \left\| {}^{(i)}\bar{r} \right\|_2^2 = \frac{1}{2} \sum_{i=1}^p \left\| {}^{(i)}\bar{o} - {}^{(i)}\bar{t} \right\|_2^2 \quad (1)$$

Supondo que os vectores ${}^{(i)}\bar{o} - {}^{(i)}\bar{t}$, têm m componentes a expressão anterior pode escrever-se,

$$E = \frac{1}{2} \sum_{i=1}^p \sum_{j=1}^m \left({}^{(i)}o_j - {}^{(i)}t_j \right)^2 \quad (2)$$

Note-se que muitos autores não utilizam o factor $\frac{1}{2}$, na expressão anterior do erro E .

A razão da utilização generalizada da norma Euclidiana prende-se com o facto desta norma ser diferenciável em todos os pontos com excepção da origem. Repare-se que o erro, E , é função dos pesos e dos desvios associados às diferentes arestas e neurónios.

Consideremos agora uma rede neuronal por camadas, com uma única camada escondida, de alimentação directa e com funções de activação f , de classe C^∞ todas elas idênticas entre si e introduza-se de seguida alguma notação.

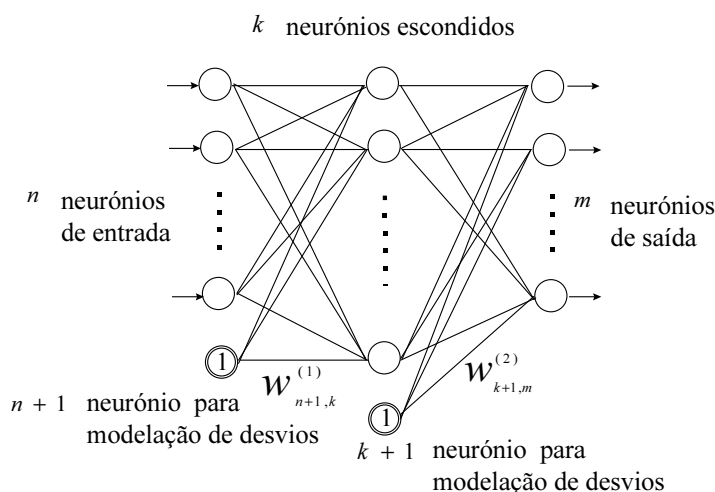


Figura 2: Representação esquemática de uma rede neuronal de três camadas.

A rede neuronal artificial que designaremos por *rede básica*, representada na figura 2, tem como se pode verificar tem n unidades de entrada, k unidades na camada escondida e m unidades de saída. Em cada uma das duas primeiras camadas está representado um neurónio adicional aos quais correspondem entradas constantes iguais ao valor 1. Estas últimas unidades permitem tratar os desvios associados a cada um dos neurónios das camadas escondida e de saída como se se tratassem dos pesos associados às arestas adicionalmente criadas: $w_{n+1,j}^{(1)}$, e $w_{k+1,l}^{(2)}$ em que $j \in \{1, \dots, n\}$ e $l \in \{1, \dots, m\}$. Chamaremos *rede ampliada* à rede básica munida destes neurónios e arestas adicionais.

Sejam W_1 e W_2 as matrizes dos pesos da rede básica correspondentes respectivamente às arestas entre as duas camadas iniciais e entre as duas camadas finais e \widehat{W}_1 e \widehat{W}_2 as correspondentes matrizes da rede ampliada. Designando por $w_{i,j}^{(1)}$ e $w_{i,j}^{(2)}$ os elementos genéricos das matrizes W_1 e W_2 que correspondem aos pesos das sinapses que estabelecem a conexão da unidade i para a unidade j conclui-se que W_1 e W_2 são matrizes respectivamente do tipo $n \times k$ e $k \times m$. Um raciocínio idêntico permite concluir que as matrizes \widehat{W}_1 e \widehat{W}_2 são respectivamente do tipo $(n + 1) \times k$ e $(k + 1) \times m$ e que diferem das primeiras apenas na linha de pesos adicionais que cada uma regista. Representaremos também por $w_{i,j}^{(1)}$ e $w_{i,j}^{(2)}$ os pesos das matrizes ampliadas

notando que as matrizes W_1, \widehat{W}_1 e W_2, \widehat{W}_2 só diferem na última linha sendo fácil reconhecer pelo índice de linha de cada peso se este está associado a uma matriz ampliada.

Refira-se que muitos autores consideram no desenvolvimento das ideias anteriores as correspondentes matrizes transpostas das matrizes anteriores facto que é normalmente facilmente reconhecível observando a ordem pela qual se efectua o produto matricial entre vectores e estas últimas.

Designemos por 0, 1, e 2, respectivamente, a primeira, segunda e terceira camada da rede e por $\bar{x}^{(0)}, \bar{y}^{(0)}, \bar{x}^{(1)}, \bar{y}^{(1)}$ e $\bar{x}^{(2)}, \bar{y}^{(2)}$ os vectores (coluna) de entrada e de saída de cada uma das referidas camadas.

Se $\bar{x} = (x_1, x_2, \dots, x_n)^T$, for um vector de entrada de dados na rede então naturalmente $\bar{x}^{(0)} = \bar{x} = \bar{y}^{(0)}$ já que a função de activação associada à primeira camada é a função identidade.

Designemos por $\widehat{x} = (x_1, x_2, \dots, x_n, 1)^T$ o vector ampliado que está associado ao vector anterior e $\bar{o} = (o_1, o_2, \dots, o_m)^T$, ao correspondente vector de saída de dados da rede. Naturalmente $\bar{y}^{(2)} = \bar{o}$.

Designemos por, $\bar{x}^{(1)} = (x_1^{(1)}, x_2^{(1)}, \dots, x_k^{(1)})^T$ e $\bar{y}^{(1)} = (y_1^{(1)}, y_2^{(1)}, \dots, y_k^{(1)})^T$ os vectores de entrada e de saída na camada escondida. Assim, $y_i^{(1)} = f(x_i^{(1)})$, $1 \leq i \leq k$. Representemos por $\widehat{y}^{(1)} = (y_1^{(1)}, y_2^{(1)}, \dots, y_k^{(1)}, 1)^T$, o correspondente vector ampliado.

Se, $\bar{x}^{(2)} = (x_1^{(2)}, x_2^{(2)}, \dots, x_m^{(2)})^T$, representar o vector de entrada na terceira camada (última camada neste caso) então com algum abuso de linguagem, $\bar{o} = \bar{y}^{(2)} = (f(x_1^{(2)}), f(x_2^{(2)}), \dots, f(x_m^{(2)}))$, pode representar-se por, $\bar{o} = \bar{y}^{(2)} = f(\bar{x}^{(2)})$.

Calculemos agora a relação entre as componentes do vector de saída de cada camada e as componentes do vector de entrada na camada seguinte.

$$x_\beta^{(1)} = \sum_{\alpha=1}^n x_\alpha w_{\alpha,\beta}^{(1)} + w_{n+1,\beta}^{(1)}, \beta \in \{1, \dots, k\} \text{ ou } \bar{x}^{(1)} = \widehat{W}_1^T \times \widehat{x}. \quad (3)$$

$$x_\gamma^{(2)} = \sum_{\beta=1}^k y_\beta^{(1)} w_{\beta,\gamma}^{(2)} + w_{k+1,\gamma}^{(2)}, \gamma \in \{1, \dots, m\} \text{ ou } \bar{x}^{(2)} = \widehat{W}_2^T \times \widehat{y}^{(1)}. \quad (4)$$

Simbolicamente e recorrendo aos abusos de linguagem já referidos, podemos escrever,

$$\begin{aligned} \bar{o} = f(\bar{x}^{(2)}) &= f(\widehat{W}_2^T \times \widehat{y}^{(1)}) = f(\widehat{W}_2^T \times f(\widehat{x}^{(1)})) = \\ &= f(\widehat{W}_2^T \times f(\widehat{W}_1^T \times \widehat{x})). \end{aligned} \quad (5)$$

Uma forma natural de minimizar o erro, E , e que está na base de uma grande parte dos algoritmos de aprendizagem com supervisão consiste, como veremos mais adiante, em procurar (iterativamente ou analiticamente) os pesos e desvios da rede tais que $\nabla E = 0$ (condição necessária para que as variáveis em questão sejam minimizantes de E).

Tendo em vista obter a expressão de ∇E que utilizaremos posteriormente na exposição de algumas técnicas de aprendizagem, comecemos por calcular, $\frac{\partial E}{\partial w_{\alpha,\beta}^{(2)}}$ e $\frac{\partial E}{\partial w_{\alpha,\beta}^{(1)}}$ considerando para tal a presença de um único padrão de treino, (\bar{x}, \bar{t}) .

Suponha-se que $\alpha \neq k + 1$:

$$\begin{aligned}
\frac{\partial E}{\partial w_{\alpha,\beta}^{(2)}} &= \frac{1}{2} \frac{\partial \|\bar{o} - \bar{t}\|_2^2}{\partial w_{\alpha,\beta}^{(2)}} = \frac{1}{2} \frac{\partial \left(\sum_{i=1}^m (o_i - t_i)^2 \right)}{\partial w_{\alpha,\beta}^{(2)}} \\
&= \frac{1}{2} \sum_{i=1}^m \frac{\partial (o_i - t_i)^2}{\partial w_{\alpha,\beta}^{(2)}} = \sum_{i=1}^m (o_i - t_i) \frac{\partial o_i}{\partial w_{\alpha,\beta}^{(2)}} \\
&= (o_\beta - t_\beta) \frac{\partial o_\beta}{\partial w_{\alpha,\beta}^{(2)}} = (o_\beta - t_\beta) \frac{\partial f(x_\beta^{(2)})}{\partial w_{\alpha,\beta}^{(2)}} \\
&= (o_\beta - t_\beta) \frac{\partial f \left(\sum_{\zeta=1}^k f(x_\zeta^{(1)}) w_{\zeta,\beta}^{(2)} \right)}{\partial w_{\alpha,\beta}^{(2)}} \\
&= (o_\beta - t_\beta) \frac{df(x_\beta^{(2)})}{dx_\beta^{(2)}} \times f(x_\alpha^{(1)}), 1 \leq \alpha \leq k.
\end{aligned}$$

Se $\alpha = k + 1$:

$$\frac{\partial E}{\partial w_{k+1,\beta}^{(2)}} = (o_\beta - t_\beta) \frac{df(x_\beta^{(2)})}{dx_\beta^{(2)}}.$$

Calculemos agora, $\frac{\partial E}{\partial w_{\alpha,\beta}^{(1)}}$, e suponha-se que $\alpha \neq n + 1$:

$$\begin{aligned}
\frac{\partial E}{\partial w_{\alpha,\beta}^{(1)}} &= \frac{1}{2} \frac{\partial \|\bar{o} - \bar{t}\|_2^2}{\partial w_{\alpha,\beta}^{(1)}} \\
&= \frac{1}{2} \frac{\partial \left(\sum_{i=1}^m (o_i - t_i)^2 \right)}{\partial w_{\alpha,\beta}^{(1)}} = \frac{1}{2} \sum_{i=1}^m \frac{\partial (o_i - t_i)^2}{\partial w_{\alpha,\beta}^{(1)}} \\
&= \sum_{i=1}^m (o_i - t_i) \frac{\partial o_i}{\partial w_{\alpha,\beta}^{(1)}} = \sum_{i=1}^m (o_i - t_i) \frac{\partial f(x_i^{(2)})}{\partial w_{\alpha,\beta}^{(1)}}
\end{aligned}$$

$$\begin{aligned}
&= \sum_{i=1}^m (o_i - t_i) \frac{df(x_i^{(2)})}{dx_i^{(2)}} \frac{\partial x_i^{(2)}}{\partial w_{\alpha,\beta}^{(1)}} = \\
&= \sum_{i=1}^m (o_i - t_i) \frac{df(x_i^{(2)})}{dx_i^{(2)}} \frac{\partial \left(\sum_{j=1}^k f \left(\sum_{\zeta=1}^n x_\zeta w_{\zeta,j}^{(1)} + w_{n+1,j}^{(1)} \right) w_{j,i}^{(2)} + w_{k+1,i}^{(2)} \right)}{\partial w_{\alpha,\beta}^{(1)}} \\
&= \sum_{i=1}^m \sum_{j=1}^k (o_i - t_i) \frac{df(x_i^{(2)})}{dx_i^{(2)}} \frac{\partial \left(f \left(\sum_{\zeta=1}^n x_\zeta w_{\zeta,j}^{(1)} + w_{n+1,j}^{(1)} \right) w_{j,i}^{(2)} \right)}{\partial w_{\alpha,\beta}^{(1)}} + \\
&\quad + \sum_{i=1}^m (o_i - t_i) \frac{df(x_i^{(2)})}{dx_i^{(2)}} \frac{\partial w_{k+1,i}^{(2)}}{\partial w_{\alpha,\beta}^{(1)}} \\
&= \sum_{i=1}^m \sum_{j=1}^k (o_i - t_i) \frac{df(x_i^{(2)})}{dx_i^{(2)}} \frac{\partial \left(f(x_j^{(1)}) w_{j,i}^{(2)} \right)}{\partial w_{\alpha,\beta}^{(1)}} \\
&= \sum_{i=1}^m \sum_{j=1}^k (o_i - t_i) \frac{df(x_i^{(2)})}{dx_i^{(2)}} \frac{d \left(f(x_j^{(1)}) w_{j,i}^{(2)} \right)}{dx_j^{(1)}} \frac{\partial x_j^{(1)}}{\partial w_{\alpha,\beta}^{(1)}} \\
&= \sum_{i=1}^m \sum_{j=1}^k (o_i - t_i) \frac{df(x_i^{(2)})}{dx_i^{(2)}} \frac{w_{j,i}^{(2)} df(x_j^{(1)})}{dx_j^{(1)}} \frac{\partial \left(\sum_{\zeta=1}^n x_\zeta w_{\zeta,j}^{(1)} + w_{n+1,j}^{(1)} \right)}{\partial w_{\alpha,\beta}^{(1)}} \\
&= \sum_{i=1}^m \sum_{j=1}^k \sum_{\zeta=1}^n (o_i - t_i) \frac{df(x_i^{(2)})}{dx_i^{(2)}} \frac{w_{j,i}^{(2)} df(x_j^{(1)})}{dx_j^{(1)}} \frac{\partial \left(x_\zeta w_{\zeta,j}^{(1)} \right)}{\partial w_{\alpha,\beta}^{(1)}} + \\
&\quad + \sum_{i=1}^m \sum_{j=1}^k (o_i - t_i) \frac{df(x_i^{(2)})}{dx_i^{(2)}} \frac{w_{j,i}^{(2)} df(x_j^{(1)})}{dx_j^{(1)}} \frac{\partial w_{n+1,j}^{(1)}}{\partial w_{\alpha,\beta}^{(1)}} \\
&= \sum_{i=1}^m (o_i - t_i) \frac{df(x_i^{(2)})}{dx_i^{(2)}} \frac{w_{\beta,i}^{(2)} df(x_\beta^{(1)})}{dx_\beta^{(1)}} x_\alpha, 1 \leq \alpha \leq n.
\end{aligned}$$

Se $\alpha = n + 1$:

$$\frac{\partial E}{\partial w_{n+1,\beta}^{(1)}} = \sum_{i=1}^m (o_i - t_i) \frac{df(x_i^{(2)})}{dx_i^{(2)}} \frac{w_{\beta,i}^{(2)} df(x_\beta^{(1)})}{dx_\beta^{(1)}}.$$

Em resumo,

$$\frac{\partial E}{\partial w_{\alpha,\beta}^{(2)}} = \begin{cases} (o_\beta - t_\beta) \frac{df(x_\beta^{(2)})}{dx_\beta^{(2)}} \times f(x_\alpha^{(1)}), 1 \leq \alpha \leq k. \\ (o_\beta - t_\beta) \frac{df(x_\beta^{(2)})}{dx_\beta^{(2)}}, \alpha = k + 1, \end{cases}, 1 \leq \beta \leq m. \quad (6)$$

e

$$\frac{\partial E}{\partial w_{\alpha,\beta}^{(1)}} = \begin{cases} \left(\sum_{i=1}^m (o_i - t_i) \frac{df(x_i^{(2)})}{dx_i^{(2)}} w_{\beta,i}^{(2)} \right) \frac{df(x_\beta^{(1)})}{dx_\beta^{(1)}} x_\alpha, 1 \leq \alpha \leq n. \\ \left(\sum_{i=1}^m (o_i - t_i) \frac{df(x_i^{(2)})}{dx_i^{(2)}} w_{\beta,i}^{(2)} \right) \frac{df(x_\beta^{(1)})}{dx_\beta^{(1)}}, \alpha = n + 1. \end{cases}, 1 \leq \beta \leq k. \quad (7)$$

De notar que as derivadas parciais (6) e (7) foram calculadas tomando em consideração apenas um único padrão de treino, (\bar{x}, \bar{t}) . Facilmente se deduz que no caso em que o conjunto dos padrões de treino é

$$A = \left\{ \left({}^{(1)}\bar{x}, {}^{(1)}\bar{t} \right), \dots, \left({}^{(p)}\bar{x}, {}^{(p)}\bar{t} \right) \right\},$$

as expressões das derivadas parciais necessárias ao cálculo de ∇E , serão:

$$\begin{aligned} \frac{\partial E}{\partial w_{\alpha,\beta}^{(2)}} &= \sum_{\mu=1}^p \frac{\partial^\mu E}{\partial w_{\alpha,\beta}^{(2)}} e \\ \frac{\partial E}{\partial w_{\alpha,\beta}^{(1)}} &= \sum_{\mu=1}^p \frac{\partial^\mu E}{\partial w_{\alpha,\beta}^{(1)}} \end{aligned} \quad (8)$$

com

$$\frac{\partial^\mu E}{\partial w_{\alpha,\beta}^{(2)}} = \begin{cases} (\mu o_\beta - \mu t_\beta) \frac{df(\mu x_\beta^{(2)})}{d\mu x_\beta^{(2)}} \times f(\mu x_\alpha^{(1)}), 1 \leq \alpha \leq k. \\ (\mu o_\beta - \mu t_\beta) \frac{df(\mu x_\beta^{(2)})}{d\mu x_\beta^{(2)}}, \alpha = k + 1, \end{cases}, 1 \leq \beta \leq m. \quad (9)$$

e

$$\frac{\partial^\mu E}{\partial w_{\alpha,\beta}^{(1)}} = \begin{cases} \left(\sum_{i=1}^m (\mu o_i - \mu t_i) \frac{df(\mu x_i^{(2)})}{d\mu x_i^{(2)}} w_{\beta,i}^{(2)} \right) \frac{df(\mu x_\beta^{(1)})}{d\mu x_\beta^{(1)}} x_\alpha, 1 \leq \alpha \leq n. \\ \left(\sum_{i=1}^m (\mu o_i - \mu t_i) \frac{df(\mu x_i^{(2)})}{d\mu x_i^{(2)}} w_{\beta,i}^{(2)} \right) \frac{df(\mu x_\beta^{(1)})}{d\mu x_\beta^{(1)}}, \alpha = n + 1. \end{cases}, 1 \leq \beta \leq k. \quad (10)$$

5.1.1 Aprendizagem de redes de perceptrões

As funções de transferência das unidades computacionais das redes de perceptrões são do seguinte tipo:

$$f(x) = \begin{cases} 1 & \text{se } x \geq 0, \\ 0 & \text{se } x < 0. \end{cases} \quad (11)$$

Como este tipo de função não é diferenciável as técnicas de aprendizagem com supervisão a utilizar não poderão basear-se na procura de pontos de estacionaridade.

Consideremos uma rede neuronal artificial com apenas duas camadas (uma camada de entrada com n unidades e uma camada de perceptrões de saída com m unidades) e suponha-se que o conjunto dos padrões de treino é,

$$A = \left\{ \left({}^{(1)}\bar{x}, {}^{(1)}\bar{t} \right), \dots, \left({}^{(p)}\bar{x}, {}^{(p)}\bar{t} \right) \right\},$$

em que cada ${}^{(i)}\bar{t}$ é uma sequência de zeros e uns. Naturalmente os vectores coluna ${}^{(i)}\bar{x}$ e ${}^{(i)}\bar{t}$ têm n e m componentes respectivamente.

Modelemos os desvios através da criação de uma unidade de entrada adicional à qual é imposto um valor de entrada igual a um, tal como foi descrito relativamente à rede neuronal da figura 2 e utilizemos a notação habitual para representar as diferentes variáveis da rede ampliada e da rede básica. Desta forma o conjunto dos padrões de treino associados à rede ampliada será,

$$\hat{A} = \left\{ \left({}^{(1)}\hat{x}, {}^{(1)}\bar{t} \right), \dots, \left({}^{(p)}\hat{x}, {}^{(p)}\bar{t} \right) \right\},$$

em que cada ${}^{(i)}\hat{x}$ é um vector de entrada ampliada.

Chamemos \widehat{W} à matriz dos pesos e \bar{o} ao vector de saída correspondente à rede ampliada. Assim,

$$f \left(\left[\widehat{W}^T \hat{x} \right]_i \right) = o_i, 1 \leq i \leq m.$$

O algoritmo básico de aprendizagem deste tipo de rede pode ser descrito da seguinte forma:

$$\begin{cases} \widehat{W}^T(k+1) = \widehat{W}^T(k) + \Delta \widehat{W}^T, \\ \Delta \widehat{W}^T(k) = (\bar{t} - \bar{o}) \hat{x}^T = \bar{e} \hat{x}^T. \end{cases} \quad (12)$$

Designando por e_j o erro verificado na unidade de saída j (componente j de \bar{e}) e chamando x_i à componente i do vector de entrada, a expressão anterior pode reescrever-se da seguinte forma mais sugestiva:

$$\begin{cases} w_{ij}(k+1) = w_{ij}(k) + \Delta w_{ij}(k) \\ \Delta w_{ij}(k) = x_i e_j, 1 \leq i \leq n+1, 1 \leq j \leq m. \end{cases} \quad (13)$$

Em resumo, se x_i for a componente i do vector \hat{x} , e tendo em conta a equação 13:

1. $e_j = 0 \Rightarrow \Delta w_{ij} = 0$;

2. $e_j = 1 \Rightarrow \Delta w_{ij} = x_i$;
3. $e_j = -1 \Rightarrow \Delta w_{ij} = -x_i$.

Note-se que a interpretação anterior torna clara a razão pela qual este método se enquadra no conjunto daqueles que se designam por *algoritmos de estimulação pela entrada*.

De notar igualmente que uma bem sucedida rede deste tipo ao associar uma dada sequência m -dimensional de zeros e uns (do conjunto $\{(1)\bar{t}, \dots, (p)\bar{t}\}$) a cada sequência n -dimensional (do conjunto $\{(1)\hat{x}, \dots, (p)\hat{x}\}$) pode ser considerada um dispositivo de classificação. No entanto pode não ser possível promover uma aprendizagem bem sucedida se algum dos conjuntos,

$$B_j = \{(i)\hat{x} \in \hat{A}_1 : (i) t_j = 1\} \text{ e } C_j = \{(i)\hat{x} \in \hat{A}_1 : (i) t_j = 0\}, 1 \leq j \leq m,$$

com $\hat{A}_1 = \{(1)\hat{x}, \dots, (p)\hat{x}\}$, não for *linearmente separável*¹.

Em Rojas [20] pode encontrar-se uma esclarecedora discussão destes tópicos.

5.1.2 Regra de aprendizagem de Widrow-Hoff

Suponha-se que a rede neuronal artificial em aprendizagem com supervisão tem apenas duas camadas² e é do tipo linear, isto é, só possui funções de activação do tipo $f(x) = x$. Nestas circunstâncias a correspondente função erro a minimizar, tendo em conta que $(i)\bar{o} = (i)\hat{x}\widehat{W}_1$ assumirá a forma,

$$E = \frac{1}{2} \sum_{i=1}^p \|(i)\bar{r}\|_2^2 = \frac{1}{2} \sum_{i=1}^p \|\widehat{W}_1^T (i)\hat{x} - (i)\bar{t}\|_2^2.$$

De notar que a matriz \widehat{W}_1 é do tipo $(n+1) \times k$, assim, a minimização de E consistirá na determinação dos $(n+1) \times k$ elementos da matriz \widehat{W}_1 que satisfazem o requisito referido. De notar que se $p \geq n+1$ (isto é se o número de padrões utilizados no treino for maior do que o número de neurónios artificiais da camada de entrada) o problema anterior é claramente um problema de *mínimos quadrados lineares*,

$$\text{minimizar } \|B\bar{w} - \bar{b}\|_2^2, \quad (14)$$

¹Dois conjuntos são *linearmente separáveis* se existir um hiperplano que passa pela origem que "separe" entre si os conjuntos referidos.

²Estamos agora a supor que tem n unidades de entrada e k unidades de saída.

em que B é uma matriz do tipo $l \times [(n+1) \times k]$ e \bar{b} um vector coluna do tipo $l \times 1$, $l \geq [(n+1) \times k]$ que contém a informação associada aos diferentes padrões utilizados na aprendizagem e \bar{w} é um vector coluna do tipo $[(n+1) \times k] \times 1$ associado aos diferentes elementos de \widehat{W}_1 que se pretendem conhecer. Como se sabe o problema (14) pode ter solução exacta se a matriz B tiver as colunas linearmente independentes recorrendo à resolução das chamadas *equações normais* (pg. 391, [18]).

No entanto diversos factores podem recomendar a utilização de métodos iterativos na resolução deste problema, tais como, o peso computacional exagerado associado ao processo de inversão da matriz dos coeficientes das referidas equações normais ou a necessidade de dispôr de um algoritmo que possa ser utilizado na aprendizagem de redes com mais do que duas camadas. Note-se que a identificação das diferentes matrizes de pesos e desvios ($\widehat{W}_1, \widehat{W}_2$, etc) de uma rede com três ou mais camadas a partir da resolução das equações normais pode não ser viável.

Nestas circunstâncias o algoritmo de aprendizagem geralmente utilizado consiste num *método de descida* do tipo *método do gradiente*. Este método consiste em procurar iterativamente os pesos e desvios que satisfazem a condição $\nabla E = 0$ (condição necessária para que esses pesos e desvios sejam minimizantes da função E), ajustando na direcção e sentido oposto ao do crescimento máximo de E as variáveis referidas, isto é fazendo,

$$\Delta w = -\varepsilon \nabla E, \varepsilon \in \mathbb{R}^+,$$

em que ε representa a chamada taxa de aprendizagem.

Mais detalhadamente,

$$w_{\alpha,\beta}^{(i)}(k+1) = w_{\alpha,\beta}^{(i)}(k) - \varepsilon \frac{\partial E}{\partial w_{\alpha,\beta}^{(i)}(k)}, \varepsilon \in \mathbb{R}^+, k \in \mathbb{N}_0. \quad (15)$$

No caso linear o método referido é designado por *algoritmo ou regra de aprendizagem de Widrow-Hoff* e assume a seguinte forma quando aplicado ao exemplo da figura 2:

$$\frac{\partial E}{\partial w_{\alpha,\beta}^{(2)}} = \begin{cases} (o_\beta - t_\beta) \times x_\alpha^{(1)}, 1 \leq \alpha \leq k. \\ (o_\beta - t_\beta), \alpha = k+1, \end{cases}, 1 \leq \beta \leq m. \quad (16)$$

e

$$\frac{\partial E}{\partial w_{\alpha,\beta}^{(1)}} = \begin{cases} \left(\sum_{i=1}^m (o_i - t_i) w_{\beta,i}^{(2)} \right) x_\alpha, 1 \leq \alpha \leq n. \\ \left(\sum_{i=1}^m (o_i - t_i) w_{\beta,i}^{(2)} \right), \alpha = n+1. \end{cases}, 1 \leq \beta \leq k. \quad (17)$$

5.1.3 Regra de aprendizagem por retropropagação do erro

Suponha-se que a rede em aprendizagem por supervisão tem três ou mais camadas. Nesta situação, o cálculo das derivadas parciais $\frac{\partial E}{\partial w_{\alpha,\beta}^{(1)}}$, $\frac{\partial E}{\partial w_{\alpha,\beta}^{(2)}}$, etc, cujo conhecimento é necessário para a implementação do método do gradiente tem de efectuar-se recursivamente. Para verificar este facto consideremos a rede correspondente à figura 2 e consideremos a seguinte representação alternativa das equações (6) e (7):

$$\frac{\partial E}{\partial w_{\alpha,\beta}^{(2)}} = \begin{cases} \Delta_{\beta}^{(2)} \times y_{\alpha}^{(1)}, 1 \leq \alpha \leq k. \\ \Delta_{\beta}^{(2)}, \alpha = k + 1, \end{cases} , 1 \leq \beta \leq m. \quad (18)$$

em que $\Delta_{\beta}^{(2)} = (o_{\beta} - t_{\beta}) \frac{df(x_{\beta}^{(2)})}{dx_{\beta}^{(2)}}$, $y_{\alpha}^{(1)} = f(x_{\alpha}^{(1)})$ e

$$\frac{\partial E}{\partial w_{\alpha,\beta}^{(1)}} = \begin{cases} \Delta_{\beta}^{(1)} x_{\alpha}, 1 \leq \alpha \leq n. \\ \Delta_{\beta}^{(1)}, \alpha = n + 1. \end{cases} , 1 \leq \beta \leq k. \quad (19)$$

em que $\Delta_{\beta}^{(1)} = \left(\sum_{i=1}^m \Delta_i^{(2)} w_{\beta,i}^{(2)} \right) \frac{df(x_{\beta}^{(1)})}{dx_{\beta}^{(1)}}$.

Repare-se que estas últimas expressões explicitam o carácter recursivo que envolve o cálculo das derivadas parciais referidas: para calcular $\frac{\partial E}{\partial w_{\alpha,\beta}^{(1)}}$ torna-se necessário calcular previamente $\frac{\partial E}{\partial w_{\alpha,\beta}^{(2)}}$. Note-se igualmente que este processo pressupõe o cálculo sucessivo dos termos $\Delta_{\beta}^{(2)}$, $\Delta_{\beta}^{(1)}$ (que se podem interpretar como *erros*), num sentido contrário ao da propagação da informação através da rede.

A regra de aprendizagem por *retropropagação do erro* consiste, assim, num algoritmo que generaliza a aplicação do *método do gradiente* a redes neuronais com três ou mais camadas e que implementa o sistema de cálculo sucessivo das derivadas parciais numa direcção contrária à da normal propagação da informação através da rede.

5.1.4 Método do gradiente

Mais formalmente, os *métodos de descida* são algoritmos iterativos desenvolvidos para resolver sistemas de equações lineares do tipo $A\bar{x} = \bar{b}$ e que se baseiam na minimização de uma norma apropriada do erro (ou equivalentemente do resíduo), tal como se expõe em Pina, [18], pg. 357. Se,

$$\bar{w}(k+1) = \bar{w}(k) + \epsilon(k)\bar{p}(k), \epsilon(k) \in \mathbb{R}, k \in \mathbb{N}_0 \quad (20)$$

representar o referido processo iterativo as escolhas do factor $\epsilon(k)$ e da direcção $\bar{p}(k)$ são efectuadas de acordo com o critério referido. O *método de gradiente* enquadra-se neste tipo de métodos e caracteriza-se pelo facto de se escolher em cada iteração a direcção $\bar{p}(k)$ que corresponde à maior descida da função erro e que coincide com a direcção do gradiente de uma norma do erro.

Supondo que uma dada rede neuronal artificial tem n sinapses com os correspondentes pesos, w_1, w_2, \dots, w_n , o método do gradiente pode ser representado pelo seguinte algoritmo,

$$w_i(k+1) = w_i(k) - \epsilon \frac{\partial E}{\partial w_i(k)}, 1 \leq i \leq n, k \in \mathbb{N}_0, \epsilon \in \mathbb{R}^+ \quad (21)$$

com E a apropriada função erro (1). O parâmetro ϵ designa-se *taxa de aprendizagem (learning rate)*.

A convergência do método do gradiente puro é em geral lenta. Na figura 3 ilustra-se uma situação em tal pode acontecer.

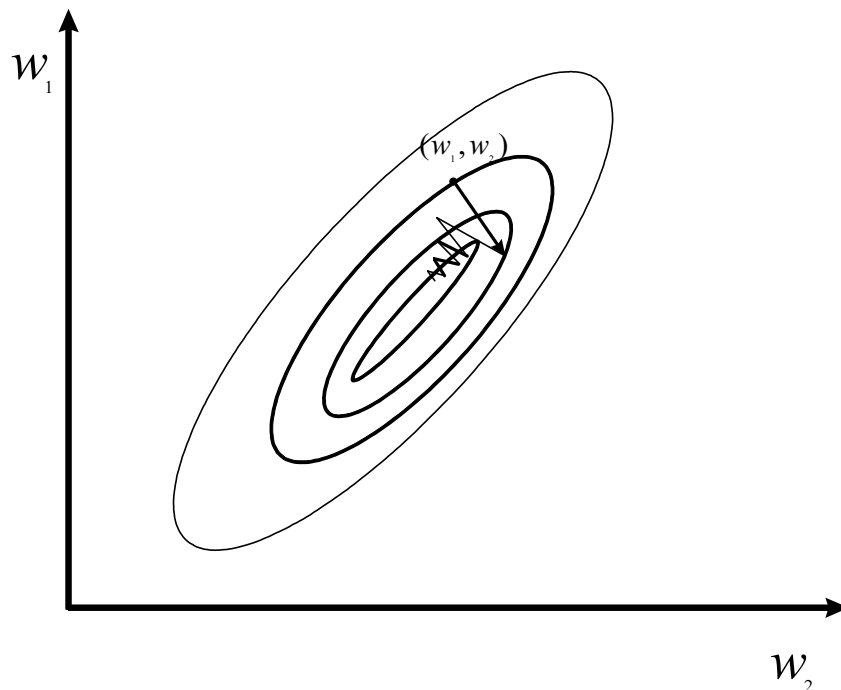


Figura 3: Oscilações na convergência do método do gradiente

A convergência demasiado lenta deste método originou o desenvolvimento de diferentes técnicas para a sua aceleração tais como o *método dos gradientes conjugados* que se caracteriza por uma escolha mais apropriada das direcções

$\bar{p}(k)$, (pg. 363, Pina [18]). O método do gradiente com *momento* é um algoritmo desta classe caracterizado pela seguinte expressão,

$$\begin{aligned} w_i(k+1) &= w_i(k) - \varepsilon \frac{\partial E}{\partial w_i(k)} + \mu \Delta w_i(k), \\ 1 &\leq i \leq n, k \in \mathbb{N}_0, \varepsilon, \mu \in \mathbb{R}^+ \end{aligned} \quad (22)$$

com $\Delta w_i(k) = w_i(k) - w_i(k-1)$. O parâmetro μ designa-se por *momento* (*momentum rate*).

De referir que esta técnica introduz pequenas correcções na direcção de descida escolhida, correcções estas que são influenciadas directamente pela direcção escolhida na iteração anterior tal como se o avanço na pesquisa dos diferentes pesos no domínio do problema se caracterizasse por possuir uma certa quantidade de movimento, momento ou memória.

Pode encontrar-se uma discussão do problema da escolha óptima dos parâmetros ε e μ em Rojas [20] (pg. 183).

5.1.5 Método de Levenberg-Marquardt

Consideremos o problema da minimização de 2,

$$E = \frac{1}{2} \sum_{i=1}^p \sum_{j=1}^m \left({}^{(i)}o_j - {}^{(i)}t_j \right)^2,$$

em que m e p são respectivamente o número de unidades de saída e o número de padrões de uma rede neuronal artificial não linear de camadas e com alimentação directa e suponha-se o problema referido reduzido a um problema com a forma,

$$E(\bar{w}) = \frac{1}{2} \bar{R}(\bar{w})^T \bar{R}(\bar{w}) = \frac{1}{2} \sum_{i=1}^{p \times m} r_i(\bar{w})^2,$$

em que \bar{w} é um vector coluna cujas componentes são os pesos e desvios da rede e \bar{R} é um campo vectorial cujas componentes escalares são os resíduos r_i associados ao problema original. De notar que estes campos escalares são funções não lineares de \bar{w} .

Consideremos agora uma aproximação de segunda ordem de E ,

$$E(\bar{w}_0 + \bar{h}) \simeq E(\bar{w}_0) + \nabla E(\bar{w}_0)^T \bar{h} + \frac{1}{2} \bar{h}^T \nabla^2 E(\bar{w}_0) \bar{h} \quad (23)$$

e com base nesta expressão calculemos uma aproximação linear do gradiente de E em \bar{w}_0 ,

$$\nabla E(\bar{w}_0 + \bar{h}) \approx \nabla E(\bar{w}_0) + \nabla^2 E(\bar{w}_0) \bar{h}. \quad (24)$$

A minimização de (23) através da procura dos seus pontos de estacionaridade (zeros de (24)) conduz-nos ao *método de Newton*:

$$\begin{aligned}\bar{h} &\approx - \left(\nabla^2 E(\bar{w}_0) \right)^{-1} \nabla E(\bar{w}_0) \Rightarrow \\ \bar{w}(k+1) &= \bar{w}(k) - \left(\nabla^2 E(\bar{w}(k)) \right)^{-1} \nabla E(\bar{w}(k)), k \in \mathbb{N}_0\end{aligned}\quad (25)$$

Notando que,

$$\nabla E(\bar{w}) = \sum_{i=1}^{p \times m} r_i(\bar{w}) \cdot \nabla r_i(\bar{w}) = \bar{J}(\bar{w})^T R(\bar{w})$$

com $\bar{J}(\bar{w})^T = (\nabla r_1(\bar{w}), \nabla r_2(\bar{w}), \dots, \nabla r_{p \times m}(\bar{w}))$, e que

$$\begin{aligned}\nabla^2 E(\bar{w}) &= \sum_{i=1}^{p \times m} \nabla r_i(\bar{w}) \cdot \nabla r_i(\bar{w}) + \sum_{i=1}^{p \times m} r_i(\bar{w}) \cdot \nabla^2 r_i(\bar{w}) \\ &= \bar{J}(\bar{w})^T \bar{J}(\bar{w}) + S(\bar{w})\end{aligned}$$

com $S(\bar{w}) = \sum_{i=1}^{p \times m} r_i(\bar{w}) \cdot \nabla^2 r_i(\bar{w})$, o método de Newton pode rescrever-se da seguinte forma,

$$\begin{cases} \bar{w}(k+1) = \bar{w}(k) - \left(\bar{J}(\bar{w}(k))^T \bar{J}(\bar{w}(k)) + S(\bar{w}(k)) \right)^{-1} \bar{J}(\bar{w}(k))^T R(\bar{w}(k)), \\ k \in \mathbb{N}_0. \end{cases}\quad (26)$$

O *método de Levenberg-Marquardt* consiste em aproximar $S(\bar{w})$ por μI , em que I , representa a matriz identidade e μ um parâmetro maior do que zero,

$$\begin{cases} \bar{w}(k+1) = \bar{w}(k) - \left(\bar{J}(\bar{w}(k))^T \bar{J}(\bar{w}(k)) + \mu(k)I \right)^{-1} \bar{J}(\bar{w}(k))^T R(\bar{w}(k)), \\ k \in \mathbb{N}_0, \mu(k) > 0. \end{cases}\quad (27)$$

De observar que quando o parâmetro μ se aproxima de zero o método anterior aproxima-se do *método de Gauss-Newton*. Mais formalmente o método de Levenberg-Marquardt pode considera-se uma modificação do método de Gauss-Newton efectuada com base nos algoritmos das regiões de confiança. Em [4] pode encontrar-se uma exposição bastante completa deste tipo de algoritmos.

5.1.6 Métodos heurísticos

Nem sempre as funções de activação associadas às diferentes unidades são diferenciáveis. Por outro lado a arquitectura complexa da rede pode tornar impraticável a aplicação de técnicas convencionais de aprendizagem. Estes factos têm motivado a utilização dos chamados *métodos heurísticos* de optimização tais como os *algoritmos evolutivos* (como por exemplo os *algoritmos genéticos*) e a *simulação do recozimento*.

A necessidade de dispor de técnicas que permitam a optimização da topologia de uma rede neuronal artificial destinada a uma aplicação específica tem também motivado a aplicação dos métodos heurísticos com este objectivo.

Em Rojas [20] pg. 428, pode encontrar-se um excelente resumo de algumas destas técnicas, das suas vantagens e principais problemas.

5.2 Aprendizagem sem supervisão

Na aprendizagem sem supervisão não se conhecem à partida as saídas correctas que correspondem a entradas conhecidas na rede neuronal.

Os algoritmos de aprendizagem podem ser classificados em duas grandes classes, Rojas [20]:

- Algoritmos *estimulação pela entrada* (*reinforcement algorithms*) que também podem ser designados por algoritmos de *aprendizagem associativa* (*associative learning rule*);
- Algoritmos de *aprendizagem competitiva* (*competitive learning*).

Nos primeiros, a entrada de cada vector na rede estimula um reajuste (reforço) dos pesos favorável a uma saída com certas características.

Nos segundos, as unidades computacionais de saída da rede competem entre si pelo direito de ser activadas (isto é fornecerem uma dada resposta) quando uma dada entrada é fornecida. Em geral só a activação de uma única unidade de saída é autorizada.

5.2.1 Regra de Hebb

A regra de Hebb insere-se na classe dos algoritmos de estimulação pela entrada ou de aprendizagem associativa. Através do exemplo seguinte procuraremos introduzir a sua aplicação.

Consideremos a rede representada na figura 4 e que possui duas camadas com idêntico número de unidades, n unidades de entrada e n unidades de

saída supondo adicionalmente que as componente dos vectores de entrada na rede pertencem ao conjunto $\{-1, +1\}$.

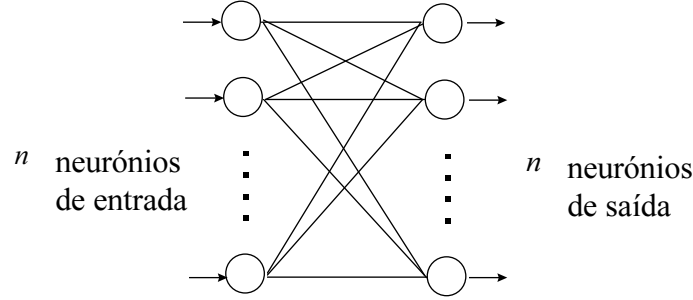


Figura 4: Ilustração da regra de Hebb

Suponha-se que as funções de activação associadas à camada de saída são assim definidas,

$$f(x_i^{(1)}) = \text{sgn}(x_i^{(1)}) \equiv \begin{cases} +1, & \text{se } x_i^{(1)} \geq 0 \\ -1, & \text{se } x_i^{(1)} < 0 \end{cases}, 1 \leq i \leq n. \quad (28)$$

Seja $\bar{\sigma} = (\sigma_1, \sigma_2, \dots, \sigma_n)$ um dado padrão (ou vector) e escolham-se os pesos da rede referida de acordo com a seguinte regra,

$$w_{ij} = \frac{1}{n} \sigma_i \sigma_j. \quad (29)$$

Consideremos o seguinte padrão (vector) de entrada na rede,

$$\bar{x} = (x_1, x_2, \dots, x_n) \equiv \begin{cases} -\sigma_i, & \text{se } 1 \leq i \leq k \\ \sigma_i, & \text{se } k+1 \leq i \leq n \end{cases}. \quad (30)$$

Repare-se que o padrão (29) difere do padrão (30) apenas em k posições. Naturalmente, $x_i^{(1)} = \sum_{j=1}^n w_{ji} x_j = \frac{1}{n} \sigma_i \sum_{j=1}^n \sigma_j x_j$ e $o_i = y_i = \text{sgn}(x_i^{(1)})$, utilizando a notação habitual.

Calculemos qual o padrão de saída correspondente à entrada referida,

$$\begin{aligned} o_i &= y_i = \text{sgn}(x_i^{(1)}) = \text{sgn}\left(\sum_{j=1}^n w_{ji} x_j\right) \\ &= \text{sgn}\left(\frac{1}{n} \sigma_i \sum_{j=1}^n \sigma_j x_j\right) = \text{sgn}\left(\frac{1}{n} \sigma_i (-k + n - k)\right) \\ &= \text{sgn}\left(\sigma_i \left(1 - \frac{2k}{n}\right)\right) = \sigma_i \text{ se } k < \frac{n}{2}. \end{aligned}$$

Note-se que a saída \bar{o} é, assim, idêntica ao padrão $\bar{\sigma}$ subjacente aos pesos escolhidos (isto é armazenados em "memória") de acordo com a regra (29) se o padrão de entrada não diferir deste último significativamente, facto que motiva a utilização deste fenómeno na concepção de sistemas de memória associativa e de reconhecimento de padrões.

Quando se pretende "memorizar" na rede diferentes padrões a generalização da anterior regra pode escrever-se da seguinte forma,

$$w_{ij} = \frac{1}{n} \sum_{\mu=1}^p {}^{(\mu)}\sigma_i {}^{(\mu)}\sigma_j \quad (31)$$

em que p representa o número de padrões do conjunto referido,

$$\{ {}^{(1)}\bar{\sigma}, {}^{(2)}\bar{\sigma}, \dots, {}^{(p)}\bar{\sigma} \}.$$

As regras anteriormente referidas constituem formas da chamada *regra Hebb*.

É importante observar que se pretendermos que a resposta (associativa) da rede anterior à entrada $\bar{\sigma}$ seja por exemplo \bar{v} , a regra 29 deve escrever-se da seguinte maneira:

$$w_{ij} = \frac{1}{n} \sigma_i v_j. \quad (32)$$

Repare-se que a equação anterior estabelece sinapses com pesos positivos sempre que os potenciais de activação entre as unidades de entrada e saída forem do mesmo sinal (concordantes) e pesos negativos caso contrário.

A discussão do comportamento de redes neuronais reguladas por regras deste tipo face a questões como a convergência, a quantidade de informação armazenada e outros tópicos relacionados, pode encontrar-se em Müller e al [16], pg. 24.

A aplicação natural da regra de Hebb a redes neuronais de perceptrões em que a aprendizagem é promovida sem supervisão pode escrever-se da seguinte forma,

$$\begin{cases} w_{ij}(k+1) = w_{ij}(k) + \Delta w_{ij}(k) \\ \Delta w_{ij}(k) = \varepsilon \sigma_i o_j. \end{cases} \quad (33)$$

em que $0 < \varepsilon < 1$, representa a chamada *taxa de aprendizagem*. A aplicação sucessiva deste processo de ajuste reforçará positivamente a intensidade dos pesos das sinapses que promovem uma saída de sinal idêntico à entrada e inibirá os restantes casos fazendo desenvolver pesos negativos nestas situações, mimetizando assim, a regra correspondente à equação 32.

Este algoritmo de aprendizagem pode igualmente ser aplicado a redes de perceptrões cujas funções de activação correspondem à função degrau unitário

e cujos vectores de entrada são elementos genéricos de \mathbb{R}^n , permitindo gerar redes que "aprendem" o padrão dos vectores do conjunto de treino.

Diferentes variantes do algoritmo 33 podem ser encontradas em aplicações. De referir a chamada *regra de Hebb com decaimento*:

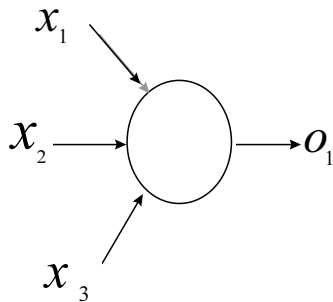
$$\begin{cases} w_{ij}(k+1) = w_{ij}(k) + \Delta w_{ij}(k) \\ \Delta w_{ij}(k) = \varepsilon x_i o_j - \lambda w_{ij}(k). \end{cases} \quad (34)$$

em que x_i representa a apropriada componente de um dos vectores do conjunto de treino e $0 < \lambda \ll 1$, um factor de decaimento cuja existência impede que os pesos assumam valores absolutos arbitrariamente grandes no decurso da aprendizagem.

5.2.2 Regras de Instar e de Outstar

As regras de Instar e de Outstar constituem variantes da regra de Hebb e estão associadas a neurónios com a mesma designação e desenvolvidos por Grossberg[6]. Basicamente as unidades de Instar produzem informações de reconhecimento e as unidades de Outstar reproduzem, por estimulação adequada, os padrões armazenados. De referir que as designações "Instar" e "Outstar" são especialmente sugestivas. A figura 5 permite comprovar este facto.

Unidade de INSTAR



Unidade de OUTSTAR

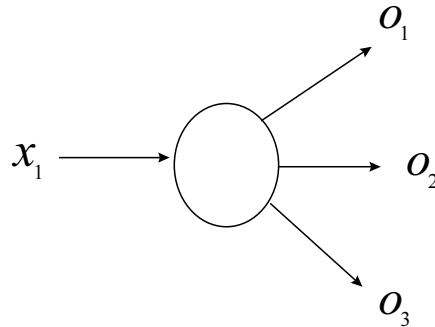


Figura 5: Exemplos de unidades de Instar e de Outstar

A regra de Instar pode descrever-se da seguinte forma,

$$\begin{cases} w_{i1}(k+1) = w_{i1}(k) + \Delta w_{i1}(k) \\ \Delta w_{i1}(k) = \varepsilon x_i o_1 - \varepsilon o_1 w_{i1}(k) = \varepsilon o_1 (x_i - w_{i1}(k)). \end{cases} \quad (35)$$

em que $o_1 \in \{0, 1\}$, $x_i \in \mathbb{R}$ é a componente i de um vector do padrão de treino e ε representa a taxa de aprendizagem.

Reparando que $o_1 \in \{0, 1\}$ pode concluir-se que este algoritmo é indêntico à regra de Hebb com decaimento supondo o factor de decaimento igual à taxa de aprendizagem. De notar que o peso w_{ij} só é ajustado quando a unidade se apresenta activa ($o_1 = 1$) e quando a diferença $x_i - w_{i1} \neq 0$, assim pode concluir-se que o processo de aprendizagem prossegue até que a unidade fique inactiva ou que $x_i = w_{i1}$. Desta forma a aquisição por parte da rede das características dos vectores permite a sua utilização como dispositivo para *reconhecimento* de padrões em particular. Nesta situação, como veremos, este algoritmo é muito semelhante à regra de Kohonen.

De referir que o algoritmo anterior pode também ser utilizado na situação em que a função de activação da unidade de saída é linear e portanto, $o_1 \in \mathbb{R}$. Aplicações com esta última característica podem encontrar-se na toolbox de redes neurais do programa MATLAB, [3].

A regra de Outstar pode descrever-se da seguinte forma,

$$\begin{cases} w_{1j}(k+1) = w_{1j}(k) + \Delta w_{1j}(k) \\ \Delta w_{1j}(k) = \varepsilon x_1(o_j - w_{1j}(k)) \end{cases}, \quad (36)$$

em que $o_j \in \mathbb{R}$ (neste caso a função de activação da unidade de saída é, normalmente, linear), $x_1 \in \{0, 1\}$ e ε representa a taxa de aprendizagem, como habitualmente. Repare-se que a aprendizagem só se verifica se $x_1 = 1$ e prossegue até que $o_j = w_{1j}(k)$ facto que possibilita a concepção de redes cuja resposta seja o padrão "armazenado" quando as suas entradas são estimuladas adequadamente.

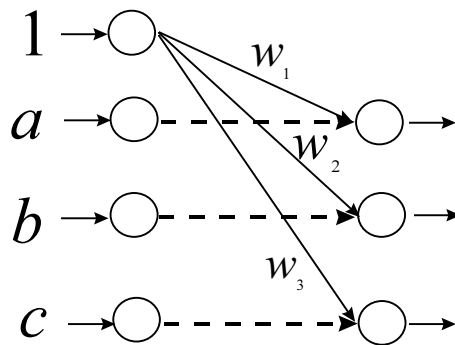


Figura 6: Rede neuronal com uma unidade de Outstar

Na figura 6 representamos uma rede com as características anteriores e que dispõe de uma unidade de Outstar. Refira-se que as sinapses representadas em traço interrompido possuem pesos permanentemente iguais a 1. Os

pesos iniciais da unidade de Outstar são $w_1 = w_2 = w_3 = 0$. A condução do processo de aprendizagem efectuada com o padrão,

$$\bar{x} = \begin{pmatrix} 1 \\ a \\ b \\ c \end{pmatrix}$$

conduz os pesos da unidade de Outstar aos valores, $w_1 = a, w_2 = b, w_3 = c$. Naturalmente e após a aprendizagem, se a rede for estimulada com a entrada, $(1, 0, 0, 0)^T$, a sua saída será, $(a, b, c)^T$, como se pretendia.

5.2.3 Algoritmos de aprendizagem competitiva

Os algoritmos de *aprendizagem competitiva* (*reinforcement and competitive learning*) podem promover a aprendizagem sem supervisão, de redes destinadas a identificar *acumulações de pontos* (*cluster points*) num espaço n -dimensional permitindo assim construir naturalmente um dispositivo de classificação. O exemplo seguinte clarifica esta ideia.

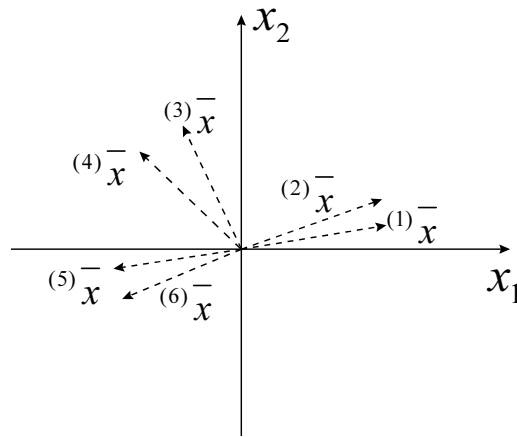


Figura 7: Seis vectores distribuidos por três zonas

Consideremos a distribuição de p pontos (*vetores de norma igual a um para simplificar*) em \mathbb{R}^2 exemplificada na figura 7 e consideremos a rede representada na figura 8.

Designemos por \bar{w}_1, \bar{w}_2 e \bar{w}_3 os pares ordenados constituídos pelos pesos associados às sinapses ligadas respectivamente às unidades de saída 1, 2 e 3, isto é,

$$\bar{w}_i = (w_{1i}, w_{2i}), 1 \leq i \leq 3.$$

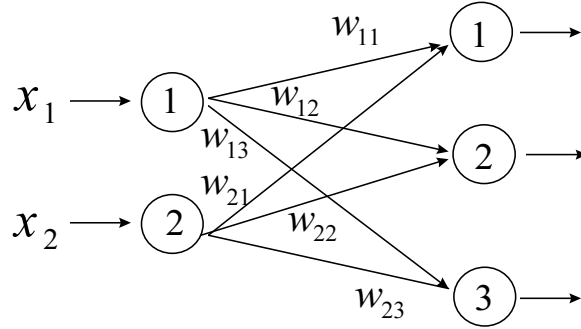


Figura 8: Rede destinada ao reconhecimento de distribuições de vectores

De referir que os vectores de norma igual a um que consideraremos na aprendizagem sem supervisão e que consideraremos distribuídos por três zonas de acumulação, serão,

$$A = \left\{ {}^{(1)}\bar{x}, {}^{(2)}\bar{x}, \dots, {}^{(p)}\bar{x} \right\},$$

cujas componentes, ${}^{(i)}\bar{x}_i = \left({}^{(i)}x_1, {}^{(i)}x_2 \right)$, representam as coordenadas da posição de cada ponto.

Defina-se o seguinte algoritmo de aprendizagem competitiva:

1. Atribuição aleatória de valores normalizados aos pesos \bar{w}_1, \bar{w}_2 e \bar{w}_3 ;
2. Aprendizagem:
 - (a) Escolha aleatória de ${}^{(j)}\bar{x} \in A$;
 - (b) Escolha de $\bar{w}_\alpha = \max \left\{ \langle {}^{(j)}\bar{x}, \bar{w}_i \rangle, 1 \leq i \leq 3 \right\}$;
 - (c) Fazer $\bar{w}_\alpha(k+1) = \bar{w}_\alpha(k) + {}^{(j)}\bar{x}$ e normalizar $\bar{w}_\alpha(k+1)$;
 - (d) Continuação da aprendizagem ou paragem após verificação de um adequado critério;
3. Paragem.

Após algumas épocas de aprendizagem cada um dos pesos \bar{w}_1, \bar{w}_2 e \bar{w}_3 irá deslocar-se para uma zona de acumulação de vectores. Reparando que $-1 \leq \langle \bar{x}, \bar{w}_i \rangle \leq 1, 1 \leq i \leq 3$, sempre que $|\bar{x}| = 1$, não é difícil verificar que se as unidades de saída estiverem munidas de funções de activação do tipo degrau unitário com um desvio de, por exemplo, $-0,9$, a rede da figura indicada responderá com saídas do tipo, $(0, 0, 0)^T, (1, 0, 0)^T, (0, 1, 0)^T$ e

$(0, 0, 1)^T$, correspondentes a entradas de norma um afastadas de todas as zonas de acumulação de vectores ou próximas de alguma delas.

De referir a possibilidade de utilização, neste tipo de redes, das chamadas *funções de activação competitivas* associadas às unidades de saída da rede anterior. Este tipo de funções permitem que só a unidade de saída mais excitada seja activada.

Em Rojas [20], pode encontrar-se a discussão da convergência deste tipo de algoritmos.

5.2.4 Regra de Kohonen

As redes de Kohonen podem designar-se redes *autoajustáveis* (*self-organizing networks*) cujo algoritmo de aprendizagem sem supervisão se insere quer no seio dos algoritmos de estimulação pela entrada ou de aprendizagem associativa quer no seio dos algoritmos de aprendizagem competitiva.

Chamando f_c à função competitiva e a \bar{y} ao vector cujas componentes são os potenciais de activação verificados nas unidades de saída, a regra de Kohonen pode assim descrever-se numa das suas formas mais simplificadas,

$$\begin{cases} w_{ij}(k+1) = w_{ij}(k) + \Delta w_{ij}(k) \\ \Delta w_{ij}(k) = \varepsilon d(i, j) o_j (x_i - w_{ij}(k)). \end{cases} \quad (37)$$

em que $o_j = [f_c(\bar{y})]_j \in \{0, 1\}$, $x_i \in \mathbb{R}$, ε representa a chamada taxa de aprendizagem e finalmente $d(i, j)$ uma apropriada distância entre a unidade de entrada i e a unidade de saída "vencedora", j . Note-se que este algoritmo apresenta muitas semelhanças relativamente à regra de Instar.

Para ilustrar mais facilmente as características autoajustáveis das redes de Kohonen e caracterizar a aplicação da regra com a mesma designação consideremos a rede da figura (9) e cujas unidades de saída estão munidas da função competitiva. Com esta rede pretende-se identificar vectores do conjunto, $A = \{^{(1)}\bar{x}, ^{(2)}\bar{x}, \dots, ^{(p)}\bar{x}\}$, distribuídos bidimensionalmente em \mathbb{R}^2 com uma linha segmentada com m vértices. Para tal definam-se os seguintes vectores de pesos associados a cada uma das unidades de saída,

$$\bar{w}_i = (w_{1i}, w_{2i}), 1 \leq i \leq m,$$

e defina-se a função distância da seguinte forma:

$$d(i, j) = \begin{cases} 1 & \text{se } |i - j| \leq r \\ 0 & \text{caso contrário} \end{cases}, \quad (38)$$

em que r é um inteiro positivo apropriado.

Execute-se o seguinte algoritmo:

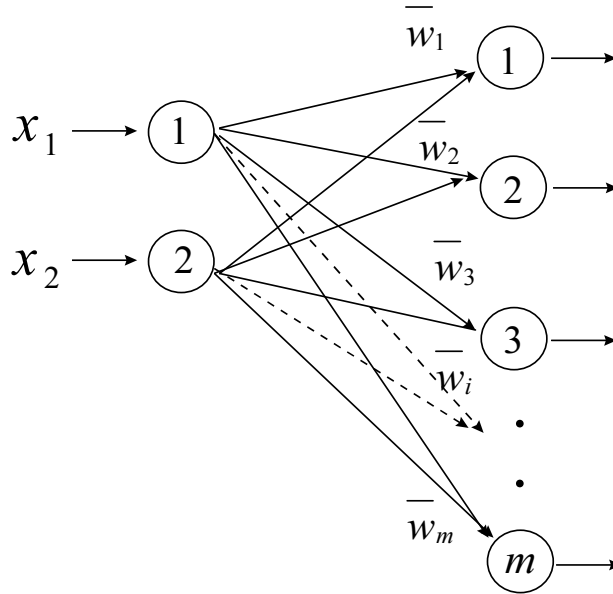


Figura 9: Ilustração de uma rede de Kohonen

1. Escolha aleatória dos vectores de pesos $\bar{w}_i = (w_{1i}, w_{2i}), 1 \leq i \leq m$;
 - (a) Escolha aleatória de um vector ${}^{(\alpha)}\bar{x} \in A$;
 - (b) Escolha da unidade de saída k que verifica a maior excitação determinada pela entrada ${}^{(\alpha)}\bar{x}$;
 - (c) Aplicar a regra de Kohonen 37,

$$\begin{cases} \bar{w}_i(k+1) = \bar{w}_i(k) + \Delta \bar{w}_i(k) \\ \Delta \bar{w}_i(k) = \varepsilon d(i, k) ({}^{(\alpha)}\bar{x} - \bar{w}_i(k)) \end{cases}, 1 \leq i \leq m;$$

- (d) Continuação da aprendizagem modificando ε e d de acordo com um plano prévio ou paragem após verificação de um adequado critério;

2. Paragem.

De referir que o plano prévio mencionado no algoritmo pressupõe uma progressiva diminuição do valor r associado à função distância.

Na execução do algoritmo pode verificar-se que os diferentes vectores de pesos (associados aos diferentes neurónios de saída) se distribuem pelas acumulações de vectores do conjunto A de uma forma mais ou menos ordenada. Note-se que pelo facto de cada vector de pesos estar associado a cada

neurónio de saída é possível estabelecer uma identificação entre os vectores do conjunto A e a "linha" com m neurónios (vértices) promovendo, no caso de uma aprendizagem bem sucedida, uma aplicação entre A e $\{1, 2, \dots, m\}$ que preserva a topologia do primeiro conjunto (isto é as relações de *vizinhaça*): se \bar{x} for um vector na vizinhaça de ${}^{(i)}\bar{x} \in A$ só a unidade de saída que lhe está associada será activada.

O fenómeno acima descrito motiva a designação deste tipo de redes por *autoajustáveis*.

Redes mais complexas permitem promover a projecção de conjuntos de vectores n -dimensionais em regiões m -dimensionais, permitindo, por exemplo, identificar a dimensão real da variedade que contém o conjunto dos vectores de treino.

Em Rojas [20] pg. 390, pode encontrar-se a discussão do problema da convergência deste tipo de algoritmos.

Referências

- [1] Chen, S., Cowan, C. F. N. e Grant, P. M., *Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks*, IEEE Transactions on Neural Networks, vol. 2, no. 2, pp. 302-309, Março 1991.
- [2] Cybenko, G., *Approximation by superpositions of sigmoidal functions*, Math. of Control, Signals, and Systems 2, 303, 1989
- [3] Demuth, H. e Beale, M., *Neural Network Toolbox-For Use with MATLAB*, The MathWorks, Inc, 1994.
- [4] Dennis, J. E. e Schnabel, R. B., *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, SIAM, 1996.
- [5] Denker, J.S., Schwartz, D., Wittner, B., Solla, S., Howard, R., Jackel, L. e Hopfield, J., *Complex Systems* 1, (877) 1987.
- [6] Grossenber, S., *Studies of the Mind and Brain*, Drodrecht, Holland: Reidel Press, 1982.
- [7] Elman, J. L., *Finding structure in time*, Cognitive Science, vol. 14, pp. 179-211, 1990.
- [8] Hebb, D., *The Organization of Behaviour*, New York: John Wiley, 1949.
- [9] Hornik, K., Stinchcomb, M. e White, H., *Multilayer FeedForward Networks are Universal Approximators*, Neural Networks 2, 359, 1989.

- [10] Kohonen, T., *Self-Organization and Associative Memory*, 2nd Edition, Springer, 1987.
- [11] Lapedes, A. S., e Farber, R. M., *Nonlinear Signal Processing using Neural Networks: Prediction and System Modelling*, Los Alamos technical report LA-UR-87-2662, 1987.
- [12] Lapedes, A. S., e Farber, R. M., *How Neural Networks Work*, in [13], pp.331.
- [13] Lee, Y. S., *Evolution, Learning and Cognition*, World Scientific, Singapore, 1988.
- [14] Li, J., Michel, A. N. e Porod, W., *Analysis and synthesis of a class of neural networks: linear systems operating in a closed hypercube*, IEEE Transactions on Circuits and Systems, vol. 36, no. 11, pp. 1405-1422, 1989.
- [15] McCulloch, W. W. and Pitts, W., *A logical calculus of the ideas immanent in nervous activity*, Bull. Math. Biophys., no. 5, pp. 115-133, 1943.
- [16] Müller, B., Reinhard, J., Strickland, M. T., *Neural Networks-An Introduction*, Springer, 1995.
- [17] Neelakanta, P. S. e De Groff, D. F., *Neural Network Modelling*, CRC Press, 1994.
- [18] Pina, Heitor, *Métodos Numéricos*, McGraw Hill, 1995.
- [19] Ripley, B. D., *Pattern Recognition and Neural Networks*, Cambridge University Press, 1996.
- [20] Rojas, R., *Neural Networks-A Systematic Introduction*, Springer, 1996.
- [21] Rosenblatt, F., *Principles of Neurodynamics*, Washington D. C.: Spartan Press, 1961.
- [22] Widrow, B. e Sterns, S. D., *Adaptative Signal Processing*, New York:Prentice-Hall, 1985.

Índice Remissivo

- acumulações de pontos, 28
- algoritmos
 - de aprendizagem associativa, 23
 - de aprendizagem competitiva, 8, 23
 - de aprendizagem correctiva, 10
 - de estimula
 - de estimulação pela entrada, 8, 10
- aprendizagem competitiva, 28
- arestas, 5
- competitivas
 - redes, 8
- decaimento
 - regra de Hebb com, 26
- desvio, 4
- função
 - lógica, 9
- funções
 - competitivas, 5
 - de activação, 5
 - de activação competitivas, 30
 - de base radial, 5
 - de Fermi, 5
 - de transferência, 5
 - Gaussianas, 5
 - sigmoidais, 5
- Kohonen
 - redes de, 8
- ligações sinápticas, 3
- linearmente separável, 17
- método
 - de Gauss-Newton, 22
 - de Levenberg-Marquardt, 22
 - de Newton, 22
- nós, 5
- neurónios, 3
 - de entrada, 5
 - de saída, 5
 - periféricos, 6
- perceptrão, 7
- peso, 5
- potencial de activação, 5
- rede
 - ampliada, 11
 - básica, 11
 - binária, 6
 - de alimenta
 - de Elman, 6
 - de Hopfield, 6
 - de McCulloch-Pitts, 6
 - recorrente, 6
- redes
 - autoajustáveis, 30
 - de Kohonen, 30
- regra
 - de Hebb, 25
 - de Instar, 26
 - de Outstar, 26
- sinapses, 3, 5
- taxa de aprendizagem, 18, 20, 25, 30
- TLU, 4
- unidades computacionais, 5