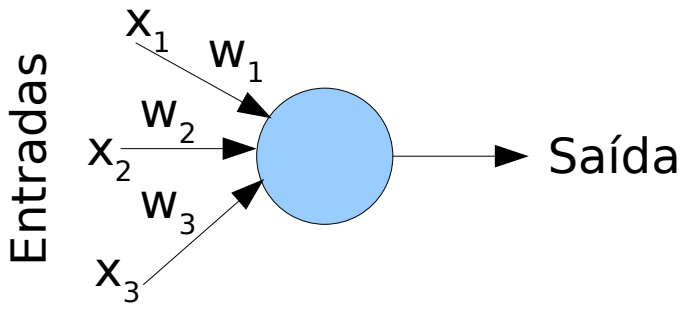
Tema 4. Redes neuronais

- Rede neuronal: combinación de unidades de procesamento local (neuronas)
- Neurona: procesamento simple de varias entradas que da unha saída
- Conexións de entrada con ponderacións (pesos, w_i) das distintas entradas



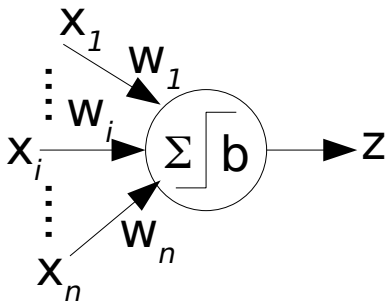
- Función de activación que determina a saída en función exclusivamente das entradas e dos pesos
- Os pesos son persistentes: a memoria da rede neuronal
- As neuronas están agrupadas en capas de neuronas: redes multicapa: capas de entrada, ocultas (1 ou varias) e de saída
- Os pesos calcúlanse para aproximar funcións n-dimensionais

Rede neuronal

- O termo rede neuronal artificial (RNA, ANN en inglés) úsase normalmente para as redes perceptrón multicapa (multi-layer perceptron, MLP)
- Existen outras redes neuronais, supervisadas e non supervisadas
- Entrenamento supervisado: calcúlanse os pesos usando patróns de entrada (patróns de entrenamento) e saídas desexadas (saídas verdadeiras)
- No teste, o patróns de entrada (patróns de teste) propáganse a través da rede para calcular as saídas (saídas preditas)
- Medimos a calidade da aprendizaxe dunha rede neuronal comparando as saídas verdadeiras e preditas para os patróns de teste coas medidas de calidade xa vistas
- As RNA poden ter sobreaprendixaxe

Perceptrón (I)

- Unha única neurona con n entradas e 1 saída, función de activación binaria (0,1) para problemas de clasificación de 2 clases:



Función activación escalón

$$z(\boldsymbol{x}) = f\left(\sum_{i=1}^{n} w_i x_i + b\right) = f\left(\boldsymbol{w}^T \boldsymbol{x} + b\right) \qquad f(t) = \begin{cases} 1 & t > 0 \\ 0 & t \leq 0 \end{cases}$$

- Necesita un umbral b (p.ex. b=0.5) para determinar se a saída é 0 ou 1
- Pesos de conexión $w_1 ... w_n$: calculados iterativamente
- As dúas clases son os dous lados do hiperplano $\mathbf{w}^T\mathbf{x}+b=0$. A clase +1 é os **x** con proxección sobre **w** ($\mathbf{w}^T\mathbf{x}$) > -b; a clase -1 é os **x** con proxección $\mathbf{w}^T\mathbf{x} \leq -b$
- Descenso de gradente, minimiza a suma de erros cadráticos, diferencias entre saída desexada (y) e predita (z), μ<1 é a velocidade de aprendizaxe

$$\boldsymbol{w}(t+1) = \boldsymbol{w}(t) + \mu(y_i - z_i)\boldsymbol{x}_i, i = 1, \ldots, N$$

Nota que $y_i - z_i \in \{0, \pm 1\}$

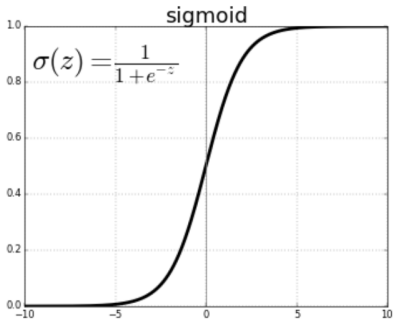
- Repítese ata que $\sum_{i=1}^{N} y_i z_i = 0$. Converxe so se os datos son linearmente separábeis

Perceptrón (II)

- ADALINE é unha versión sen a activación f non linear: resólvese co algoritmo LMS (Least Mean Squares)
- Activación f derivábel: sigmoide ou tanxente hiperbólica:

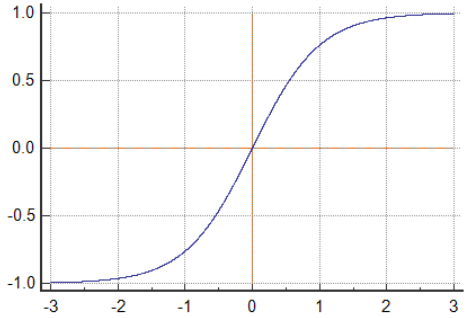
Función sigmoide loxística: 0,1

$$f(t)=\sigma(t)=\frac{1}{1+e^{-at}}$$



Función tanxente hiperbólica: ± 1

$$f(t)=\tanh at=\frac{e^{-at}-e^{at}}{e^{at}+e^{-at}}$$



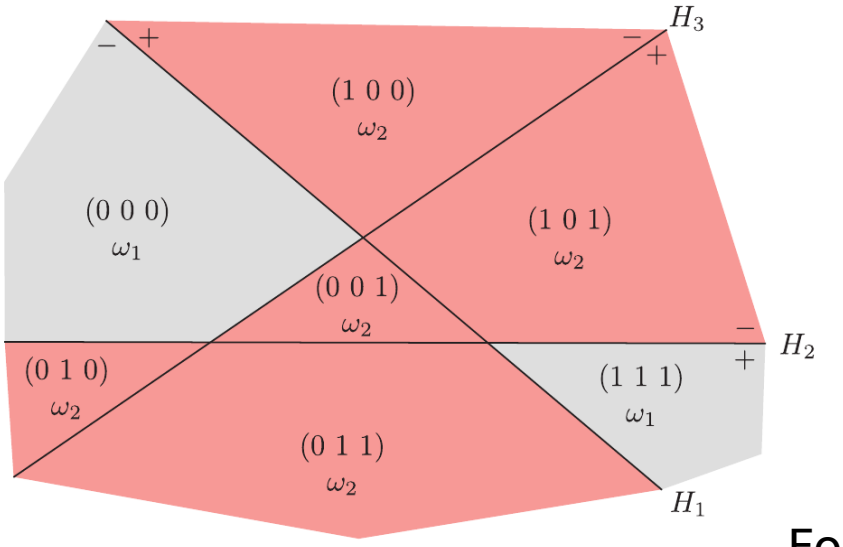
- Kernel perceptron: clasificación linear $z=f(\mathbf{w}^T\Phi(\mathbf{x})+b)$ no espazo oculto multi-dimensional proxectado (f=escalón, Φ: $\mathbb{R}^n \rightarrow \mathbb{R}^m$ con m>n: cerne):

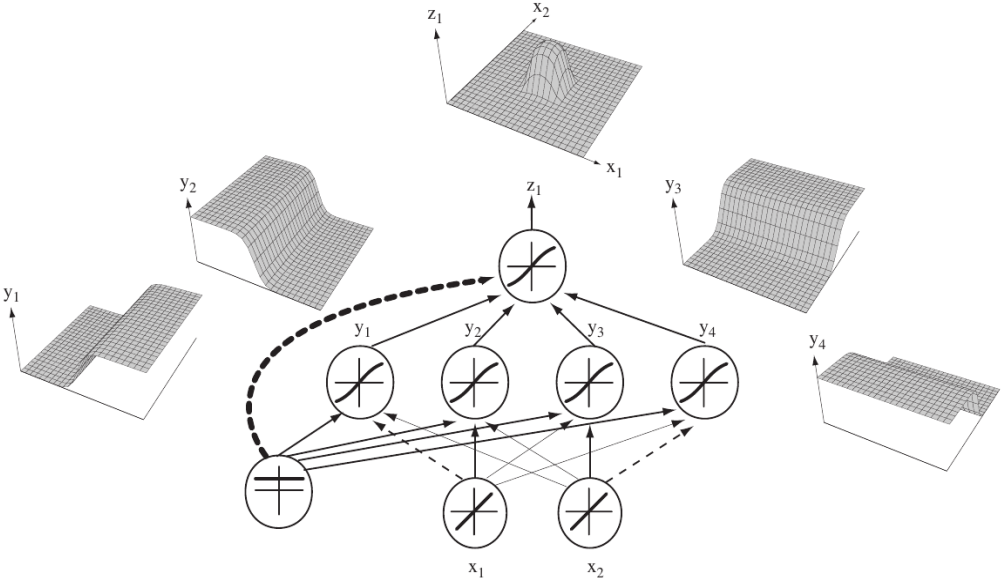
$$\boldsymbol{w}=\sum_{i=1}^{N} a_i y_i \boldsymbol{\Phi}(\boldsymbol{x}_i) \qquad b=\sum_{i=1}^{N} a_i y_i$$

- Inicio: $a_i=0$, i=1..N; $a_i=a_i+1$ se $y_i z_i<0$, ata que $y_i z_i>0$, $\forall$i=1..N

Rede neuronal multicapa

- Cada neurona da capa oculta é un clasificador linear do espazo de entrada (hiperplano en n dimensións)
- Separa o espazo en dous semiespazos, con saídas 0,1 ou ± 1
- Xuntando varias neuronas nunha capa, divídese o espazo en rexións con fronteiras lineares a cachos
- Cada rexión da unha combinación de saídas distinta na capa oculta

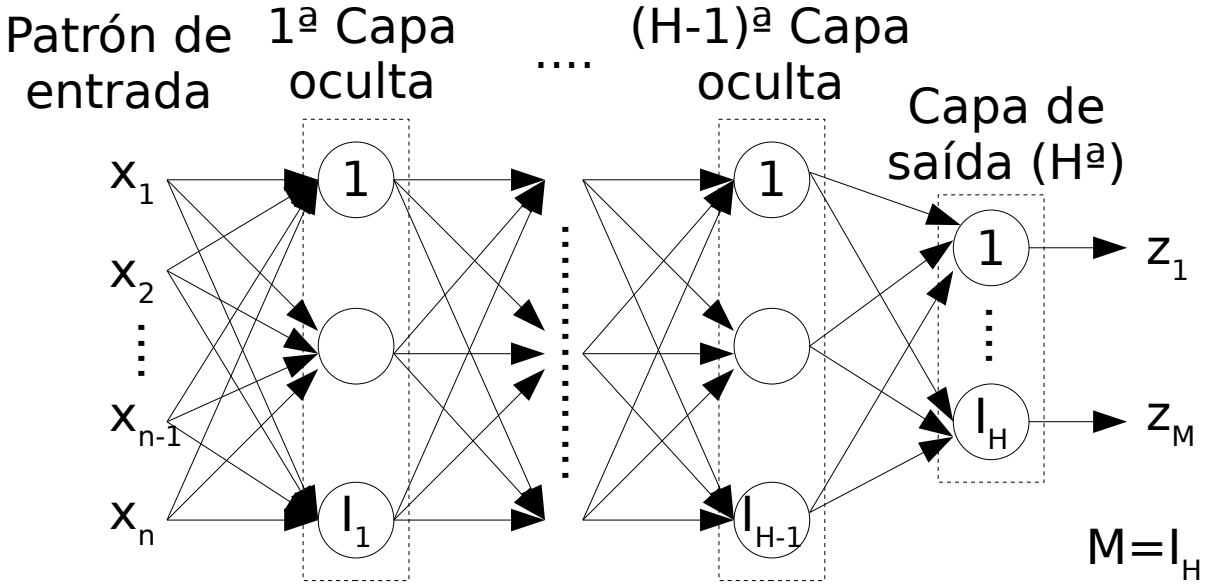




Fonte: Haykin, Neural networks. A comprehensive foundation, Prentice Hall, 1988

Rede perceptrón multicapa (MLP)

- Capa oculta: activación sigmoidal ou tanh
- Capa de saída: escalón (clasificación) ou linear (saídas reais, regresión)
- Necesitamos: conxunto de entrenamento, función de coste, activación derivábel



- $\mathbf{w}_j^k$: peso de neurona j=1..l_k en capa k=1..H
- $a_{ij}^k=(\mathbf{w}_j^k)^T\mathbf{h}_i^{k-1}+b_j^k$, i=1..N (patrón), k=1..H (capa), j=1..l_k (neurona)
- $\mathbf{h}_i^{k-1}$: saída de capa k-1, con l_{k-1} valores, para patrón $\mathbf{x}_i$
- $\mathbf{h}_i^k$: saída de capa k para patrón $\mathbf{x}_i$: $h_{ij}^k=f(a_{ij}^k)$ con j=1..l_k
- y_{ij} =saída verdadeira de neurona de saída j e patrón $\mathbf{x}_i$

Retropropagación (I)

- Descenso de gradente: $\Delta \boldsymbol{w}_j^k = -\mu \frac{\partial J}{\partial \boldsymbol{w}_j^k}, \Delta b_j^k = -\mu \frac{\partial J}{\partial b_j^k}, k = 1, \ldots, H$
- Erro cadrático J_i : avaliado na capa de saída para patrón i: $J_i = \frac{1}{2} \sum_{j=1}^{I_H} \left(y_{ij} - h_{ij}^H \right)^2 = \frac{|\boldsymbol{y}_i - \boldsymbol{h}_i^H|^2}{2}, J = \sum_{i=1}^{N} J_i$
- A derivada é: $\frac{\partial J_i}{\partial \boldsymbol{w}_j^k} = \frac{\partial J_i}{\partial a_{ij}^k} \frac{\partial a_{ij}^k}{\partial \boldsymbol{w}_j^k}, \frac{\partial J_i}{\partial b_j^k} = \frac{\partial J_i}{\partial a_{ij}^k} \frac{\partial a_{ij}^k}{\partial b_j^k}$

$a_{ij}^k = (\mathbf{w}_j^k)^T \mathbf{h}_i^{k-1} + b_j^k$

- Defínese: $\delta_{ij}^k \equiv \frac{\partial J_i}{\partial a_{ij}^k}$. Ademais: $\frac{\partial a_{ij}^k}{\partial \boldsymbol{w}_j^k} = \boldsymbol{h}_i^{k-1}, \frac{\partial a_{ij}^k}{\partial b_j^k} = 1$
- De modo que: $\Delta \boldsymbol{w}_j^k = -\mu \sum_{i=1}^{N} \delta_{ij}^k \boldsymbol{h}_i^{k-1}; \Delta b_j^k = -\mu \sum_{i=1}^{N} \delta_{ij}^k$
- Para a capa de saída (k=H): $\delta_{ij}^H = \left(y_{ij} - h_{ij}^H \right) f'\left(a_{ij}^H \right) = \varepsilon_{ij}^H f'\left(a_{ij}^H \right)$

$h_{ij}^k = f(a_{ij}^k)$

$\varepsilon_{ij}^H = y_{ij} - h_{ij}^H$

$$\Delta \boldsymbol{w}_j^H = -\mu \sum_{i=1}^{N} \varepsilon_{ij}^H f'\left(a_{ij}^H \right) \boldsymbol{h}_i^{H-1}$$

$$\Delta b_j^H = -\mu \sum_{i=1}^{N} \varepsilon_{ij}^H f'\left(a_{ij}^H \right)$$

Retropropagación (II)

- Para a capa k<H:

$$\delta_{ij}^{k}=\frac{\partial J_i}{\partial a_{ij}^{k}}=\sum_{l=1}^{I_{k+1}}\frac{\partial J_i}{\partial a_{il}^{k+1}}\frac{\partial a_{il}^{k+1}}{\partial a_{ij}^{k}}=\sum_{l=1}^{I_{k+1}}\delta_{il}^{k+1}\frac{\partial a_{il}^{k+1}}{\partial a_{ij}^{k}}$$

- Como $a_{il}^{k+1}=(\mathbf{w}_l^{k+1})^T\mathbf{h}_i^k+b_l^{k+1}$ e $h_{ij}^k=f(a_{ij}^k)$, temos que:

$$a_{il}^{k+1}=\sum_{m=1}^{I_k} w_{lm}^{k+1} f\left(a_{im}^{k}\right)+b_l^{k+1} \qquad \frac{\partial a_{il}^{k+1}}{\partial a_{ij}^{k}}=w_{lj}^{k+1} f'\left(a_{ij}^{k}\right)$$

- Obtendo unha expresión recursiva de δ_{ij}^k en función de δ_{ij}^{k+1} e $\mathbf{w}_l^{k+1}$:

$$\delta_{ij}^{k}=\sum_{l=1}^{I_{k+1}}\delta_{il}^{k+1}w_{lj}^{k+1}f'\left(a_{ij}^{k}\right)=f'\left(a_{ij}^{k}\right)\sum_{l=1}^{I_{k+1}}\delta_{il}^{k+1}w_{lj}^{k+1},k=H-1,\ldots,1$$

- Denotando $\varepsilon_{ij}^{k}=\sum_{l=1}^{I_{k+1}}\delta_{il}^{k+1}w_{lj}^{k+1}$, obtemos: $\boxed{\delta_{ij}^{k}=\varepsilon_{ij}^{k}f'\left(a_{ij}^{k}\right)}$
- A derivada é f ’(t)=af(t)[1-f(t)] e f ’(t)=a[1-f ^{2}(t)] para sigmoide e tanxente hiperbólica, respectivamente

Retropropagación (III)

repeat # bucle das épocas
  for i=1:N
    for k=1:H # propagación directa (entrada)
      for j=1:I_k
        $a_{ij}^k=(\mathbf{w}_j^k)^T\mathbf{h}_i^{k-1}+b_j^k$; $h_{ij}^k=f(a_{ij}^k)$
      endfor
    endfor
    for j=1:I_H
      $\varepsilon_{ij}^H=y_{ij}-h_{ij}^H$; $\delta_{ij}^H=\varepsilon_{ij}^H f'(a_{ij}^H)$
    endfor
    for k=H-1:-1:1 # retropropagación do erro
      for j=1:I_k
        $\varepsilon_{ij}^k=\sum_{l=1}^{I_{k+1}}\delta_{il}^{k+1}w_{lj}^{k+1}$; $\delta_{ij}^k=\varepsilon_{ij}^k f'(a_{ij}^{k+1})$
      endfor
    endfor
  endfor
  for k=1:H # actualización dos pesos
    for j=1:I_k
      $\Delta\boldsymbol{w}_j^k=-\mu\sum_{i=1}^{N}\delta_{ij}^k\boldsymbol{h}_i^{k-1}$; $\Delta b_j^k=-\mu\sum_{i=1}^{N}\delta_{ij}^k$
      $\mathbf{w}_j^k=\mathbf{w}_j^k+\Delta\mathbf{w}_j^k$; $b_j^k=b_j^k+\Delta b_j^k$
    endfor
  endfor
until criterio de remate

Proceso por lotes

- Pesos ($\mathbf{w}_j^k$, b_j^k) iniciais aleatorios baixos
- Época=presentación do conxunto de entrenamento
- **Criterio de remate**: 1) J ou gradente de J inferior a umbral; ou 2) máximo nº épocas
- Velocidade μ intermedia: evita lentitude e oscilacións
- Caída en mínimos locais que poden ter J elevado: seleccionar a mellor de entre varias inicializacións
- Actualización de pesos patrón a patrón (online): pode evitar mínimos locais, converxe antes

Melloras e consellos (I)

- **Momento** (α): inercia no aprendizaxe: $\Delta \boldsymbol{w}_j^k(t+1)=\alpha \Delta \boldsymbol{w}_j^k(t)-\mu \sum_{i=1}^{N} \delta_{ij}^k \boldsymbol{h}_i^{k-1}$

  t=iteración; $\alpha \in [0.7\text{-}0.95]$; $\Delta \mathbf{w}_j^k$ redúcese aprox. en 1-α

- **RMSProp** (root mean square propagation): divide o incremento nos pesos por un termo que depende do seu cadrado; μ,β son hiperparámetros

  $S_w=0; S_b=0$

  for epoca=1:nepocas

  calcula $\Delta \mathbf{w}$ e Δb

  $S_w=\beta S_w+(1-\beta)|\Delta \mathbf{w}|^2; S_b=\beta S_b+(1-\beta)\Delta b^2$

  $$\boldsymbol{w}(t+1)=\boldsymbol{w}(t)-\frac{\mu \Delta \boldsymbol{w}}{\varepsilon+\sqrt{S_w}}; b(t+1)=b(t)-\frac{\mu \Delta b}{\varepsilon+\sqrt{S_b}}$$

  endfor

Usa o cadrado de gradente para escalar o incremento nos pesos

- **Preprocesamento**: estandarización, entradas con media 0 e desviación 1, debe coincidir co rango da activación f(t)

- **Activación simétrica** (tanh) no canto de sigmoide

Melloras e consellos (II)

- **Velocidade de aprendizaxe adaptativa**: $\mu(t=0)\in[0.03\text{-}0.1]$
  - $\mu(t+1)=(1+\varepsilon_i)\mu(t)$ se $J(t+1)<J(t)$
  - $\mu(t+1)=(1-\varepsilon_d)\mu(t)$ e $\alpha=0$ se $J(t+1)>(1+\varepsilon_c)J(t)$,
  - $\varepsilon_i=0.05$, $\varepsilon_d=0.3$ e $\varepsilon_c=0.04$
- **Reducción da velocidade**: t=época; μ_0,β,k: hiperparámetros

$$\mu(t)=\frac{\mu_0}{1+t\beta} \qquad \mu(t)=0.95^t\mu_0 \qquad \mu(t)=\frac{k\mu_0}{\sqrt{t}}$$

- **Velocidade distinta para cada peso**: increméntase μ cando o gradente dese peso ten o mesmo signo en dúas iteracións
- **Saídas desexadas** y_{ij} con valores no rango da función de activación (linear para regresión, sigmoide ou tanxente hiperbólica para clasificación, etc.)

Melloras e consellos (III)

- Se $y_{ij} \in [0,1]$, pode ser interpretada como probabilidade e pódese usar a **entropía cruzada** como función de custe no canto de RMS:

$$J = -\sum_{i=1}^{N} \sum_{j=1}^{I_H} \left[y_{ij} \ln h_{ij}^H + \left(1 - y_{ij}\right) \ln \left(1 - h_{ij}^H\right) \right]$$

- Outra función de custe é a **diverxencia de Karhunen-Loewe** ou **entropía relativa**, usando activación softmax para a saída:

$$J = -\sum_{i=1}^{N} \sum_{j=1}^{I_H} y_{ij} \ln \frac{h_{ij}^H}{y_{ij}} \qquad h_{ij}^H = \frac{\mathrm{e}^{a_{ij}^H}}{\sum_{l=1}^{I_H} \mathrm{e}^{a_{il}^H}}$$

- O nº H de capas e o nº de neuronas I_k en cada capa k=1..H debe ser decidido de antemán: hiperparámetros sintonizábeis
- Se hai moitas neuronas (e pesos $\mathbf{w}_j^k$, que son parámetros libres) en relación co número de patróns de entrenamento, prodúcese **sobreaprendizaxe**

Melloras e consellos (IV)

- Na práctica demostrouse que a retropropagación non consegue boas solucións usando varias capas ocultas
- Demostrouse que o MLP cunha soa capa oculta é un aproximador universal de calquera función continua
- O erro de entrenamento pódese reducir ata que se quera aumentando o nº de neuronas ocultas
- Isto en xeral aumenta o erro de teste: sobreaprendizaxe
- **Regularización do MLP**: súmase ao erro J normal un termo cadrático cos pesos ponderado por un hiper-parámetro de regularización λ: $J'(\mathbf{W})=J(\mathbf{W})+\lambda|\mathbf{W}|^2$, $\mathbf{W}=\{\mathbf{w}_j^k\}$, $k=1..H, j=1..I_k$
- Alternativa a $|\mathbf{W}|^2$: $\sum_{l=1}^{N_w} \frac{\theta_l^2}{\theta_h^2+\theta_l^2}$ sendo θ_l o peso l-ésimo ($l=1..N_w$) e θ_h un umbral: tende a suprimir os pesos l con $\theta_l<\theta_h$

Melloras e consellos (V)

- **Análise de sensibilidade**: suprímense periódicamente os pesos θ_l con saliencia $s_l = h_{ll}\theta_l^2/2$ baixa (mide efecto en J da supresión de θ_l)

$$h_{ll} = \frac{\partial^2 J}{\partial \theta_l^2}$$

- **Parada temprana** (early stopping): en cada época, avalía a rede sobre un conxunto de validación (distinto ao de entrenamento)

  Detén o entrenamento cando o erro neste conxunto comeza a aumentar (síntoma de sobreaprendizaxe)

- **Compartición de pesos**: fórzase a determinadas conexións a ter pesos iguais para garantir certas invarianzas

  Ex: traslación, rotación e escalado de imaxes

  Alternativa: usar como entradas características invariantes ante estas transformacións

Limitacións do MLP

- Entrenamento lento, rematando en mínimos locais non óptimos, máis acusado con varias capas ocultas
- Moitos parámetros sintonizábeis: nº de capas ocultas (H), nº de neuronas en cada capa ($I_1..I_H$), velocidade de aprendizaxe (μ), momento (α), etc.
- Por un tempo abandonáronse as redes multicapa porque unha RNA cunha capa oculta é aproximador universal
- O nº de neuronas para aprender un problema cunha soa capa oculta é meirande que con varias capas
- A retropropagación baséase en f ’(a_{ij}^{k+1}): a función sigmoide ou tanh ten derivada nula en case todo o seu dominio
- Isto leva a gradentes nulos e detén o entrenamento, provocando moitos problemas

Extreme Learning Machine (ELM)

- Rede con H=2 capas de propagación: 1 capa oculta, 1 capa de saída, propagación directa, non usa retropropagación
- Pesos de entrada $\mathbf{W}^1=\{w_{jl}^1\}$ e sesgos $\{b_j\}$ con j=1..I_1 e l=1..n, con valores aleatorios no rango [-1,1]
- Pesos de saída $\mathbf{W}^2=\{w_{jl}^2\}$ e $\{b_j\}$ con j=1..I_2 e l=1..I_1 calculados usando a pseudoinversa (pinv) da matriz **H** de actividades na capa oculta e as saídas desexadas **Y** como $\mathbf{W}^2=\mathbf{H}^+\mathbf{Y}$
- Método directo e eficiente para datos pequenos
- Demóstrase teóricamente que o erro de entrenamento redúcese cando $I_1 \rightarrow N$
- Capa de saída: activación f sigmoide (clasificación) ou linear (regresión):

$$y_{ij}=h_{ij}^2=w_{jl}^2 f\left(\sum_{m=1}^{n} w_{lm}^1 x_{im}+b_j^1\right)$$

$$i=1,\dots N\,;\, j=1\dots I_2$$

- So sintoniza o número I_1 de neuronas ocultas

Outros tipos de redes neuronais

- Redes de funcións de base radial (radial basis function, RBF)
- Redes con realimentación das entradas ás saídas: p.ex. redes recursivas
- Mapas auto-organizados (Self-Organized Maps, SOM): aprendizaxe non supervisado
- Learning vector quantization (LVQ): SOM ampliado con aprendizaxe supervisada
- Redes neuronais celulares (CNN)
- Máquinas de Boltzmann