

-
-
-
-
-

Perceptron Multi-camadas Treinada por Retropropagação

Aluizio Fausto Ribeiro Araújo
Universidade Federal de Pernambuco
Centro de Informática



-
-
-
-
-

Conteúdo

- Histórico
- Introdução
- Perceptron de Camadas Múltiplas
- Algoritmo de Retropropagação: Descrição, Funções, Capacidades e Exemplos
- Aproximação de Funções
- Arquivo de Treinamento para Generalização.
- Seleção de Modelo
- Protocolo de Treinamento
- Pontos Positivos e Negativos do MLP-BP
- Heurísticas para Melhoria de Desempenho do MLP-BP
- Técnicas para Melhoria do MLP-BP



Histórico

- O perceptron de Rosenblatt: Neurônio com pesos e bias ajustáveis.
- Teorema da convergência do perceptron: Se padrões empregados para treinar o perceptron são extraídos de duas classes linearmente separáveis, então o algoritmo converge e posiciona uma superfície de decisão na forma de hiperplano para separar duas classes.
- Limitações do perceptron:
 - Resolve apenas casos linearmente separáveis.
 - A não-linearidade na ativação do perceptron (função limiar) não é diferenciável, logo não pode ser aplicada a camadas múltiplas.
- Um simples neurônio gera uma classe de soluções que não são gerais para resolver problemas mais complexos, justificando o aparecimento do Perceptron de Camada Múltipla (MLP).

Histórico

- Surgimento do MLP treinado por retropropagação (MLP-BP):
 - Widrow & Hoff (1960) introduzem o algoritmo mínimo quadrado médio (LMS), considerando uma rede de 3 camadas com pesos fixos da entrada para a camada escondida e pesos ajustáveis (regra delta) da camada escondida para a de saída.
 - Desenvolvimento do MLP-RP:
 - Filtro de Kalman (Kalman 1960, Bryson, Denham, Dreyfus 1963, 1969).
 - Werbos: Tese de doutorado (Harvard University, 1974): Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Science.
 - Parker 1982, 1985: "Learning logic: Casting the cortex of the human brain in silicon," TR-47, M.I.T. Center for Computational Research in Economics and Management Science, Cambridge, MA, Feb. 1985.
 - Rumelhart, Hinton, Williams: Learning internal representations by backpropagating errors, Nature 323(99), pp533-536, 1986.

Introdução

- A rede neural supervisionada chamada Perceptron multicamadas (*multilayer perceptron*) utiliza métodos derivados do gradiente no ajustes de seus pesos por retropropagação.
 - Esta rede consiste de uma camada de entrada, uma ou mais camadas escondidas e uma camada de saída. Um sinal de entrada é propagado, de camada em camada, da entrada para a saída.
 - MLP é treinada com um algoritmo de retropropagação do erro.
- Estágios da aprendizagem por retropropagação do erro:
 - Passo para frente: Estímulo aplicado à entrada é propagado para frente até produzir resposta da rede.
 - Passo para trás: O sinal de erro da saída é propagado da saída para a entrada para ajuste dos pesos sinápticos.

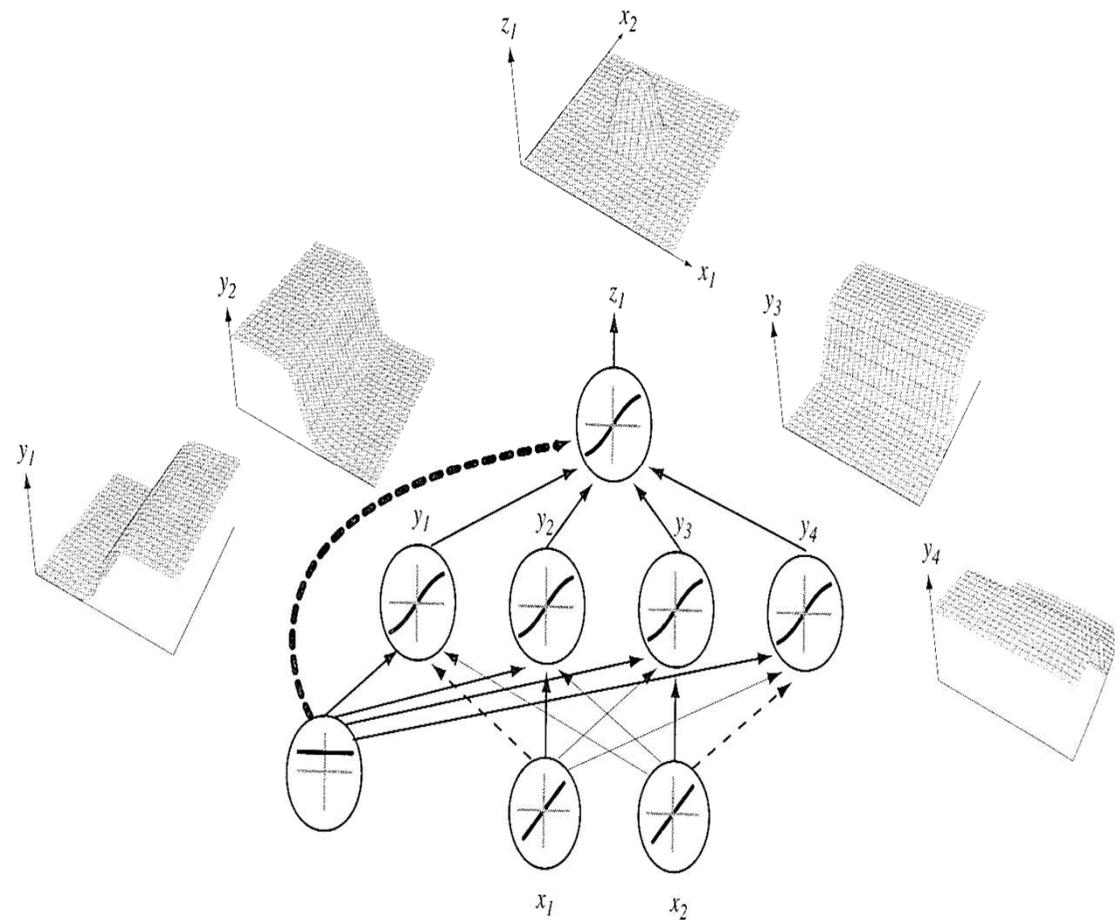
Introdução

- A MLP apresenta três características importantes:
 - Cada unidade de processamento tem função de ativação logística (forma sigmoideal) que é não-linear, suave e diferenciável.
 - Existência de ao menos uma camada escondida que possibilita aprendizagem de tarefas complexas por extração progressiva de características relevantes dos padrões de entrada.
 - O grau de conectividade é alto.
- As limitações de um Perceptron simples devem desaparecer quando se utiliza camadas intermediárias, ou escondidas, entre as camadas de entrada e de saída.
- O emprego de Redes MLP aumentou com a introdução do método retropropagação para aprendizagem.

Introdução

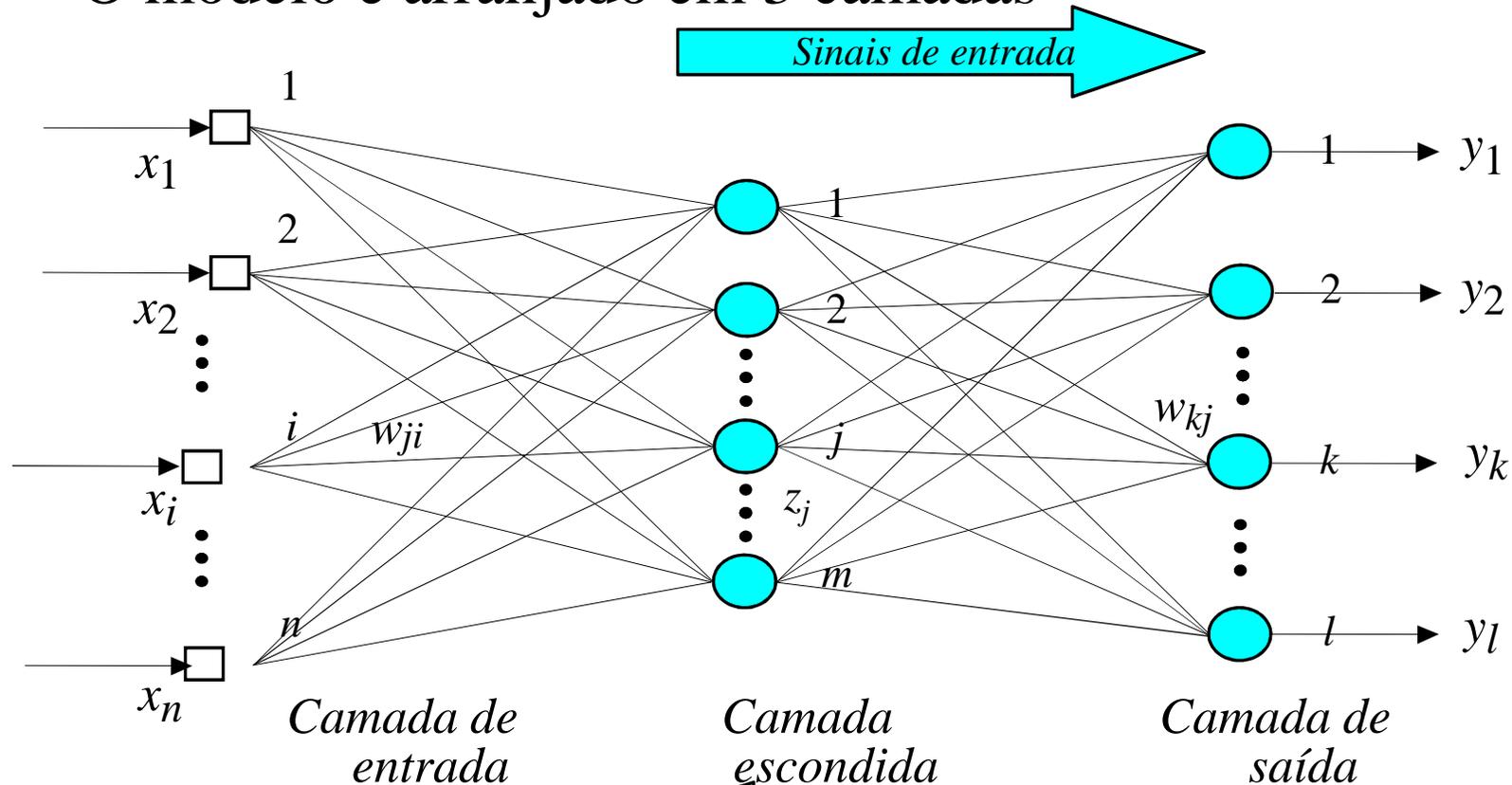
- Sumário da MLP-RP

- Camada de entrada
- Camada escondida
- Camada de saída
- Unidade de bias
- $net_j = \mathbf{w}_j^t \mathbf{x}$
- $y_j = f(net_j)$
- $f(net_k) = \text{Sigm}(net_k)$
- $z_k = f(net_k)$



Perceptron de Camadas Múltiplas

- O modelo é arranjado em 3 camadas



Algoritmo MLP-BP

- Sinal de erro para a unidade k na iteração n : $e_k(n) = t_k(n) - y_k(n)$

- Erro total:

- C é o conjunto de nodos de saída.

$$E(n) = \frac{1}{2} \sum_{k \in C} e_k^2(n)$$

- Erro quadrado médio:

- Média sobre todo conjunto de treinamento.

- N_{pad} é o número de padrões.

$$E_{av} = \frac{1}{N_{pad}} \sum_{n=1}^{N_{pad}} E(n)$$

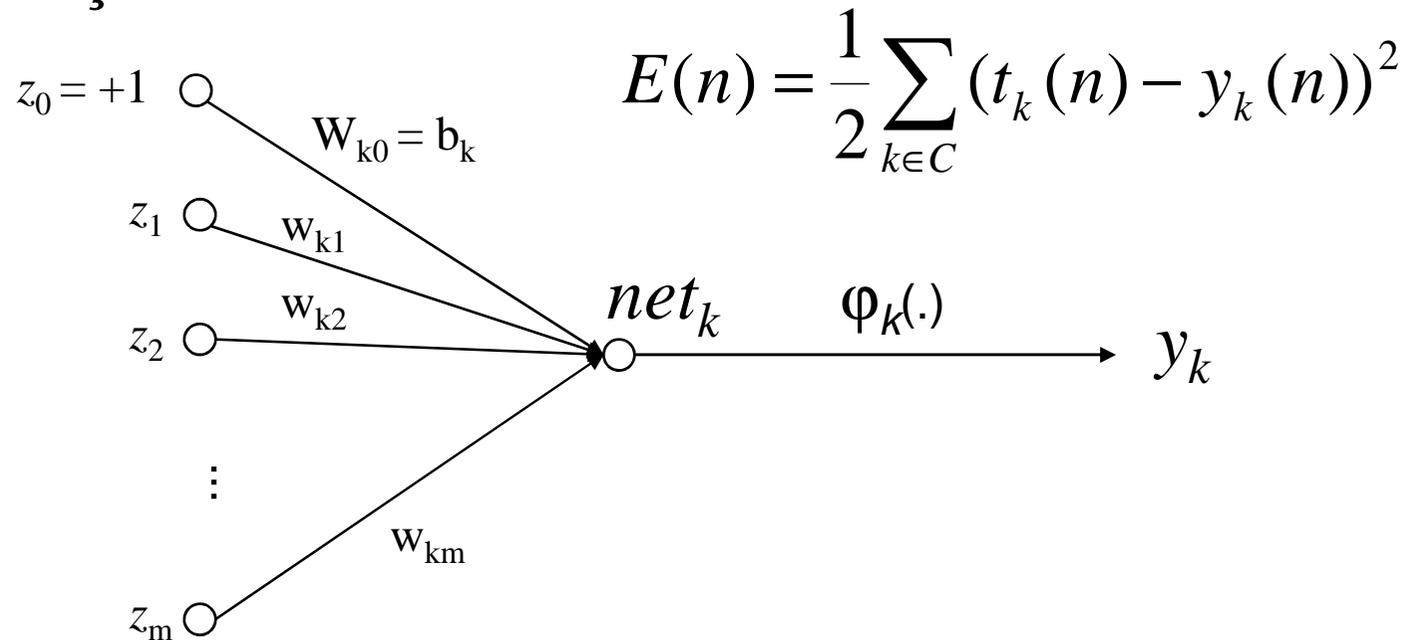
- Processo de aprendizagem: Ajusta parâmetros (pesos sinápticos) para minimizar E_{av} .

- Pesos são atualizados padrão a padrão até completar uma época (apresentação completa do conjunto de treinamento).



Algoritmo MLP-BP

- Notação:



$$net_k(n) = \sum_{i=0}^m w_{ki}(n) z_i(n) \quad y_k(n) = \varphi_k(net_k(n))$$

Algoritmo MLP-BP

- Atualização dos pesos
- Gradiente Descendente
- Por simplicidade de notação, o índice n será retirado.

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \Delta\mathbf{W}(n)$$

$$\Delta w_{pq}(n) = -\eta \frac{\partial E(n)}{\partial w_{pq}(n)}$$

$$\Delta w_{pq} = -\eta \frac{\partial E}{\partial w_{pq}}$$

MLP-BP: Ajuste de pesos $j \rightarrow k$ (nodo $h-o$)

- Gradiente determina a direção de busca no espaço de pesos.

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial net_k} \frac{\partial net_k}{\partial w_{kj}} = -\delta_k \frac{\partial net_k}{\partial w_{kj}} \dots \frac{\partial net_k}{\partial w_{kj}} = \frac{\partial}{\partial w_{kj}} \sum_{q=1}^m w_{kq} z_q = \sum_{q=1}^m \frac{\partial w_{kq}}{\partial w_{kj}} z_q \dots$$

$$\frac{\partial net_k}{\partial w_{kj}} = z_j, \quad \text{pois} \quad \frac{\partial w_{kq}}{\partial w_{kj}} = 0, \quad \frac{\partial w_{kj}}{\partial w_{kj}} = 1$$

- Sensibilidade descreve como o erro total se modifica com a atividade net de cada unidade.

$$\delta_k = -\frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial net_k} = (t_k - y_k) \phi'_k(net_k)$$

- Ajuste para nodo $h-o$: $\Delta w_{kj} = \eta \delta_k z_j = \eta (t_k - y_k) \phi'_k(net) z_j$

MLP-BP: Ajuste de pesos $i \rightarrow j$ (nodo $i-h$)

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial z_j} \frac{\partial z_j}{\partial net_j} \frac{\partial net_j}{\partial w_{ji}}, \quad \text{onde} \quad \frac{\partial E}{\partial z_j} = \frac{\partial}{\partial z_j} \left[\frac{1}{2} \sum_{k=1}^l (t_k - y_k)^2 \right] \therefore$$

$$\frac{\partial E}{\partial z_j} = - \sum_{k=1}^l (t_k - y_k) \frac{\partial y_k}{\partial z_j} = - \sum_{k=1}^l (t_k - y_k) \frac{\partial y_k}{\partial net_k} \frac{\partial net_k}{\partial z_j} = - \sum_{k=1}^l (t_k - y_k) \phi'_k(net_k) w_{kj}$$

$$\frac{\partial z_j}{\partial net_j} = \phi'_j(net_j); \quad \frac{\partial net_j}{\partial w_{ji}} = x_i;$$

$$\frac{\partial E}{\partial w_{ji}} = \left(- \sum_{k=1}^l (t_k - y_k) \phi'_k(net_k) w_{kj} \right) (\phi'_j(net_j))(x_i) = \left(- \sum_{k=1}^l \delta_k w_{kj} \right) (\phi'_j(net_j))(x_i)$$

$$\Delta w_{ji} = \eta \delta_j x_i = \eta \left[\sum_{k=1}^l w_{kj} \delta_k \right] \phi'_j(net_j) x_i$$



MLP-BP: Resumo da Aprendizagem

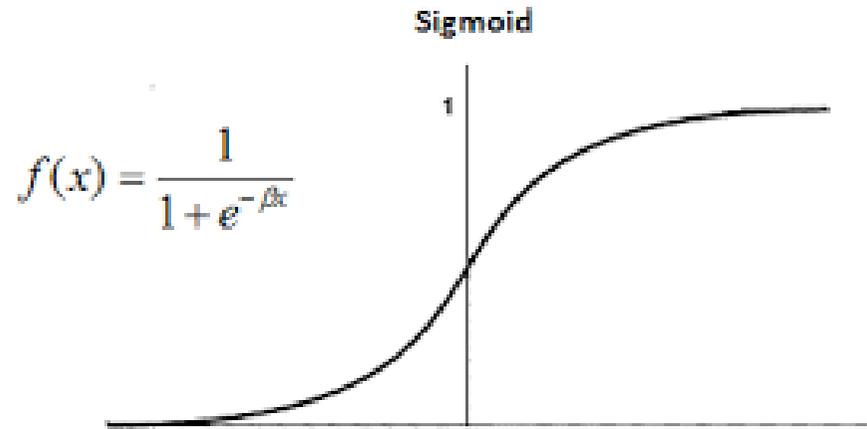
$$\begin{pmatrix} \text{correção} \\ \text{no peso} \\ \Delta w_{ji}(n) \end{pmatrix} = \begin{pmatrix} \text{taxa de} \\ \text{aprendizagem} \\ \eta \end{pmatrix} \cdot \begin{pmatrix} \text{sensibilidade} \\ \delta_j(n) \end{pmatrix} \cdot \begin{pmatrix} \text{sinal de entrada} \\ \text{para nodo } j \\ x_i(n) \end{pmatrix}$$

$$net_j(n) = \sum_{i=0}^m w_{ji}(n) x_i(n) \quad y_j(n) = \varphi_j(net_j(n))$$

- Passagem entrada-saída
- Passagem saída-entrada
 - Compute recursivamente gradiente local δ
 - Da camada de saída para a de entrada
 - Modifique os pesos pela regra delta

MLP-BP: Funções de Ativação

- Função Sigmoid

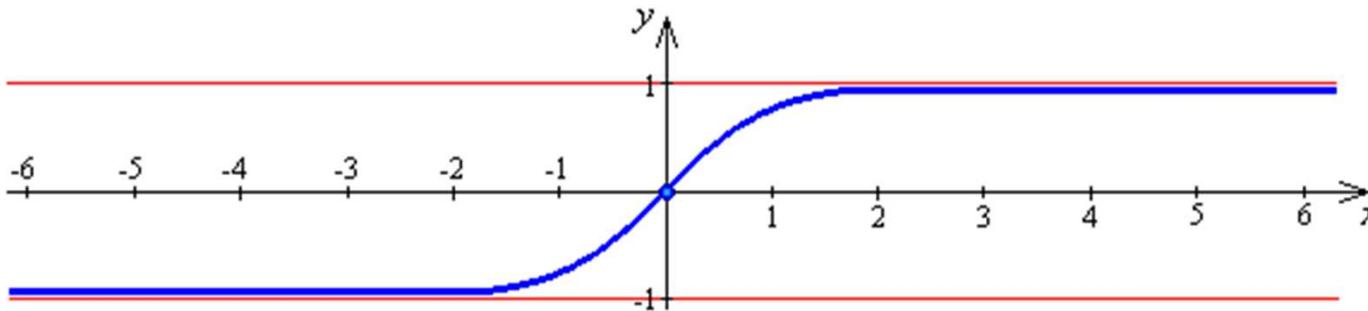


$$y_j = \varphi(\text{net}_j) = \frac{1}{1 + \exp(-a \cdot \text{net}_j)} \quad a > 0 \text{ and } -\infty < \text{net}_j < \infty$$

$$\varphi'(\text{net}_j) = \frac{1}{1 + \exp(-a \cdot \text{net}_j)} \frac{a \cdot [1 + \exp(-a \cdot \text{net}_j) - 1]}{1 + \exp(-a \cdot \text{net}_j)} = a \cdot y_j [1 - y_j]$$

MLP-BP: Funções de Ativação

- Função Tangente hiperbólica



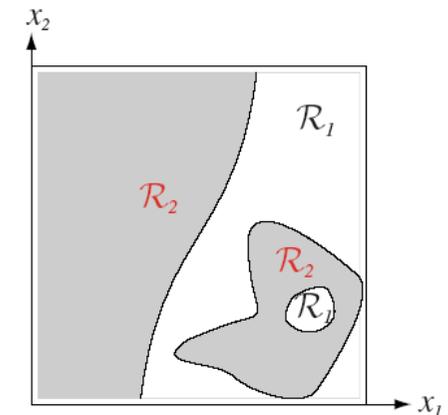
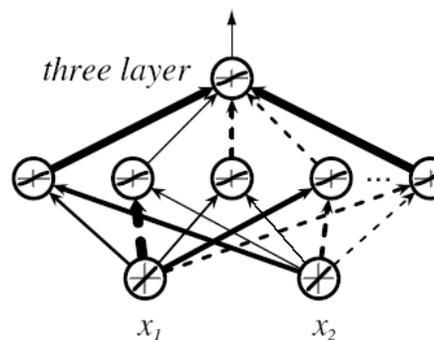
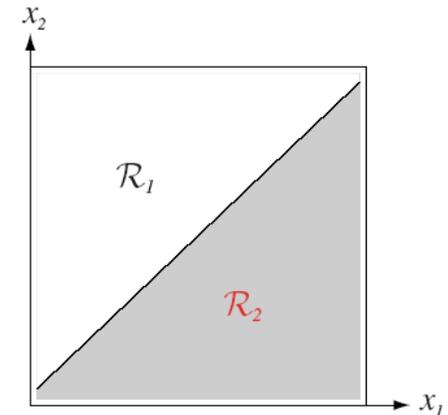
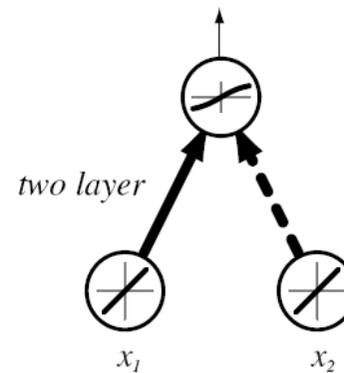
$y = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$\begin{cases} \text{Domínio: } \mathbb{R} \\ \text{Imagem: } (-1, 1) \end{cases}$	$\lim_{x \rightarrow -\infty} \tanh(x) = -1 \text{ e } \lim_{x \rightarrow +\infty} \tanh(x) = 1$
--	--	---

$$\varphi(\text{net}_i) = a \cdot \tanh(b \cdot \text{net}_j) \quad (a, b) > 0$$

$$\varphi'(\text{net}_i) = a \cdot b \cdot \text{sech}^2(b \cdot \text{net}_j) = a \cdot b(1 - \tanh^2(b \cdot \text{net}_j)) = \frac{b}{a} [a - y_j][a + y_j]$$

Poder Expressivo da MLP-PB

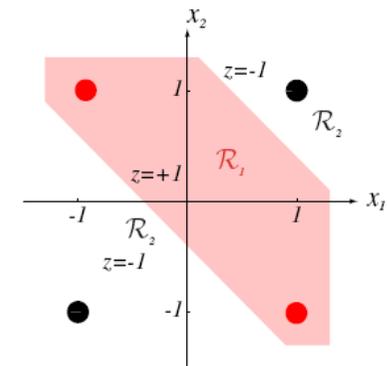
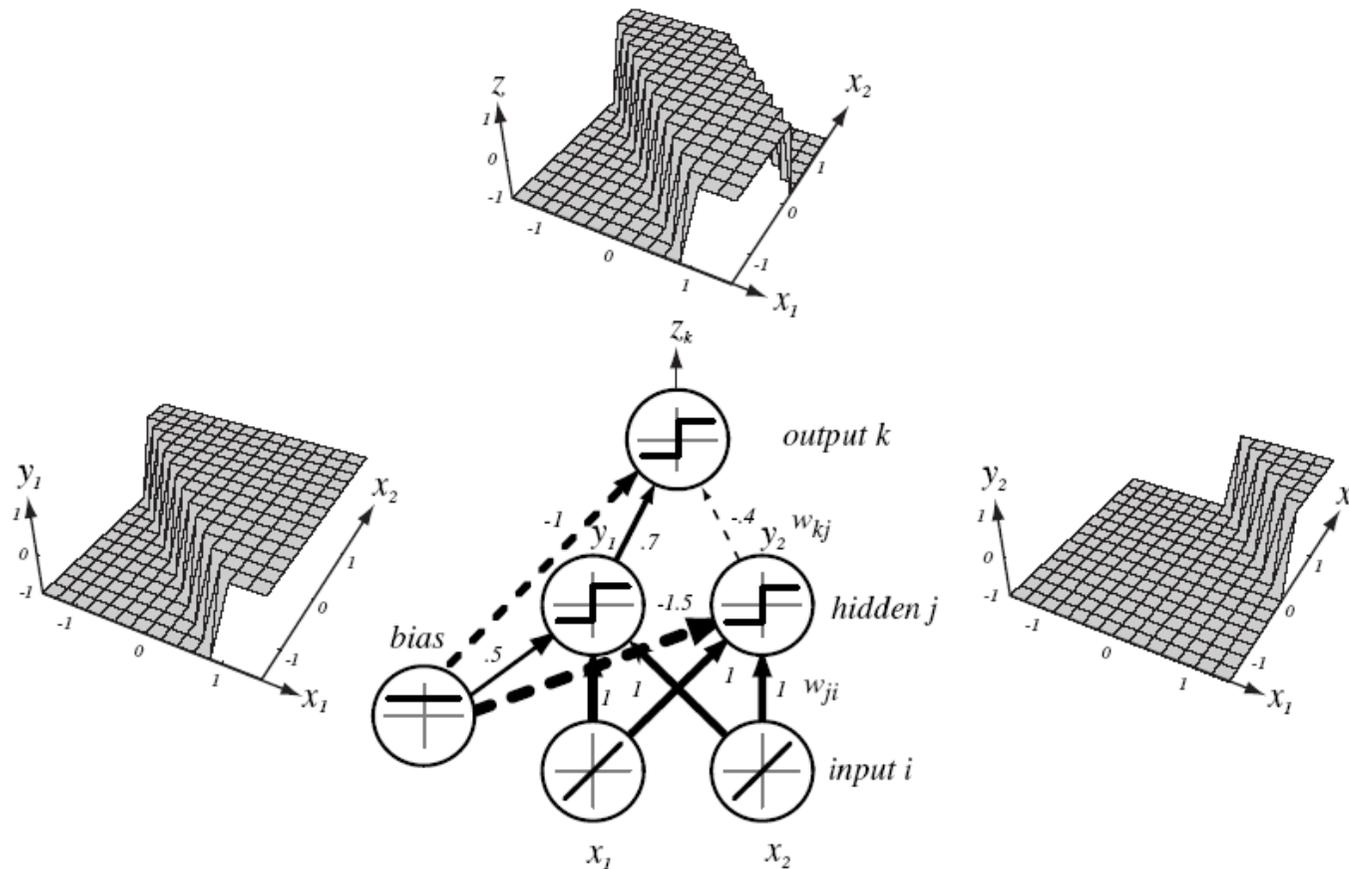
- Qualquer transformação pode ser implementada por 3 camadas?
 - Sim, qualquer função contínua da entrada para a saída pode ser implementada com número suficiente de nodos escondidos.
 - Prova-se pelo teorema de Kolmogorov.



Poder Expressivo da MLP-PB

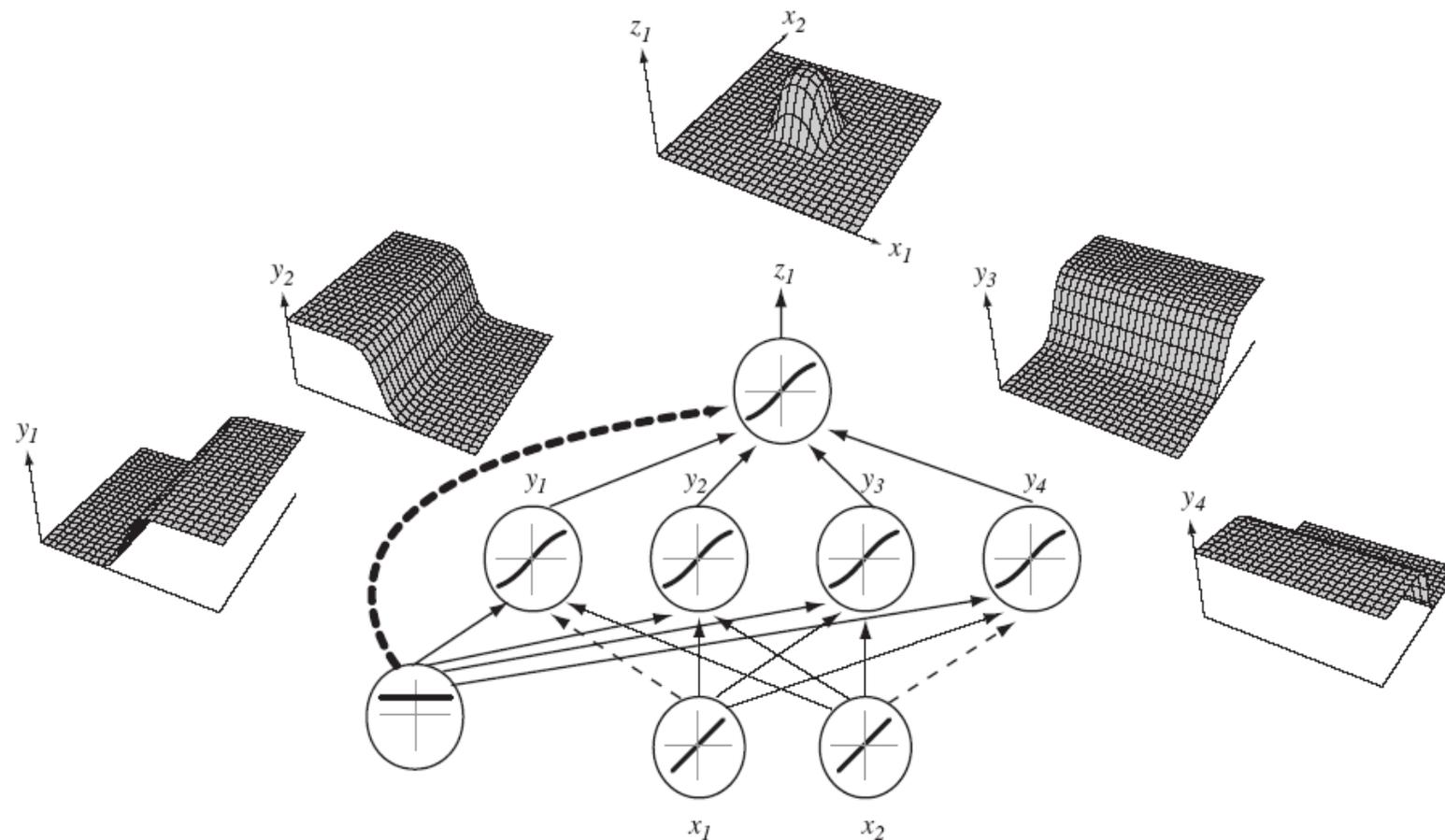
- Porta XOR: Operação da rede

Divisão de classes

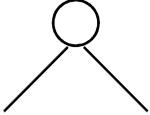
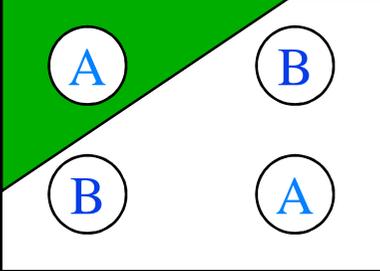
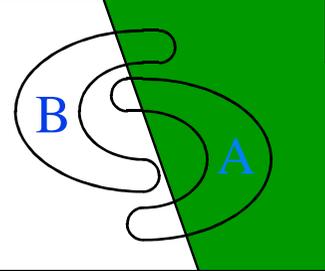
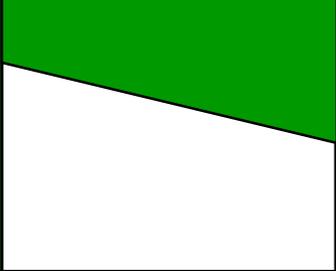
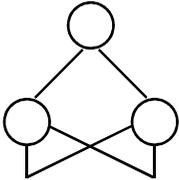
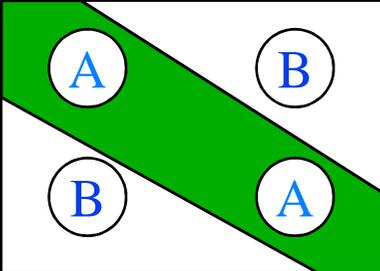
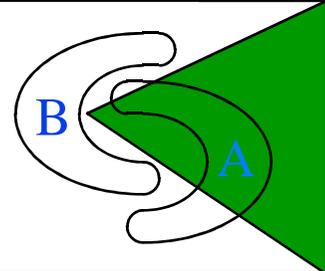
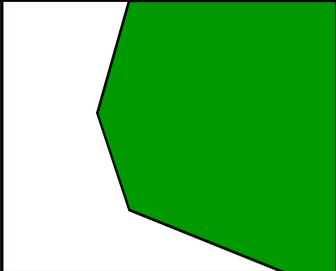
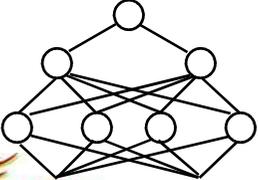
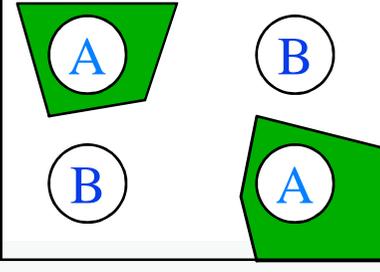
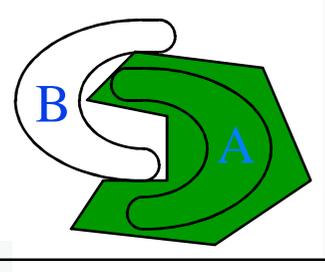
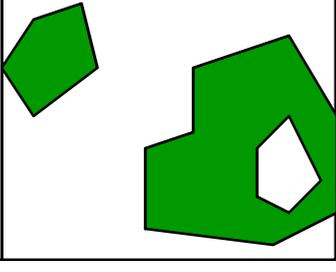


Poder Expressivo da MLP-PB

- Exemplo de operação

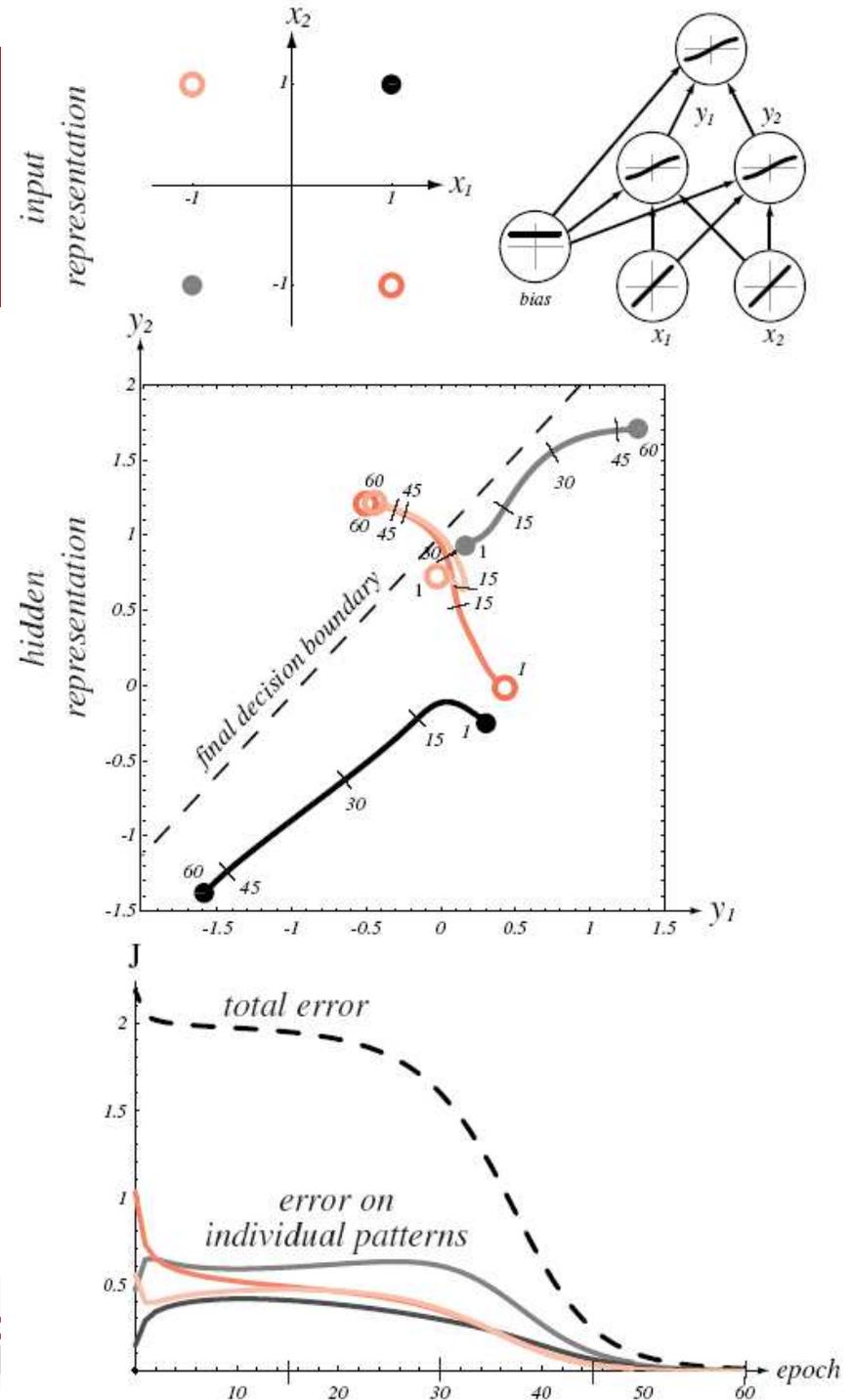


Exemplos de Problemas Não-Linearmente Separáveis

<i>Arquitetura</i>	<i>Tipos de região de decisão</i>	<i>Porta OU-exclusivo</i>	<i>Classes mais complexas</i>	<i>Regiões mais gerais</i>
<i>Camada única</i> 	Semi-plano separado por hiperplano			
<i>Duas camadas</i> 	Regiões convexas abertas ou fechadas			
<i>Três camadas</i> 	Complexidade dependente do problema			

MLP-BP: Exemplo

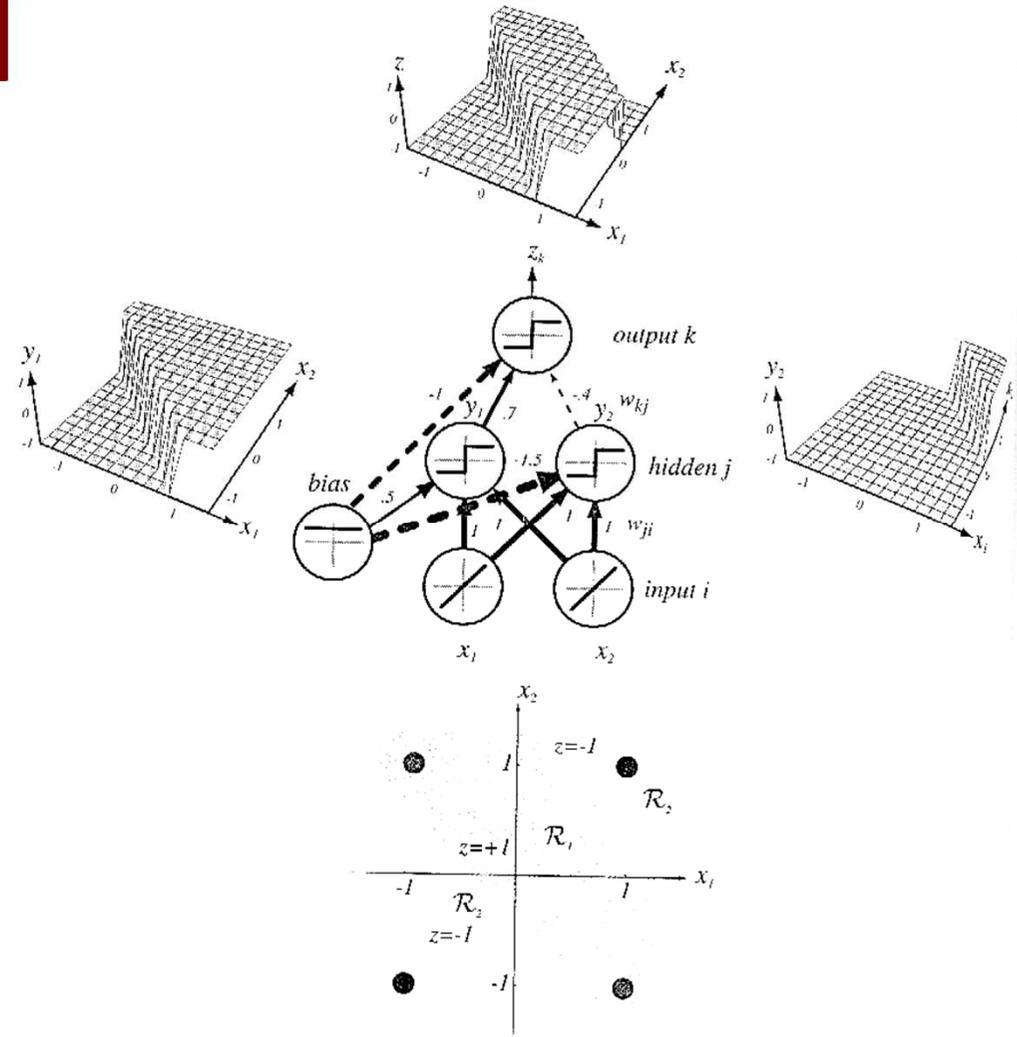
-
-
- **Detector de Características:**
 - Unidades escondidas atuam como detectores de características.
 - O progresso da aprendizagem leva as unidades escondidas a descobrirem gradualmente característica salientes dos dados de treinamento.
 - Transformação não-linear do espaço de entrada para o de características.
 - Semelhança com o discriminante linear de Fisher.



MLP-BP: Exemplo

- **Porta XOR:**

- $(x_1 \text{ OR } x_2) \text{ AND NOT } (x_1 \text{ AND } x_2)$
- $y_1 : x_1 + x_2 + 0.5 = 0$
- $> 0 \rightarrow y_1 = 1$, de outro modo -1
- $y_2 : x_1 + x_2 - 1.5 = 0$
- $> 0 \rightarrow y_2 = 1$, de outro modo -1



Aproximação de Funções

- Uma MLP-BP pode ser vista como um mecanismo para construir um mapeamento entrada-saída não-linear.
 - Questão fundamental: O número de camadas escondidas na MLP para se construir um mapeamento contínuo.
- Teorema da Aproximação Universal:
 - Definido como um teorema de existência no sentido que ele provê a justificativa matemática para aproximação de uma função arbitrária e contínua em contraposição a uma representação exata.
 - A equação ($F(.)$) que fundamenta o teorema generaliza aproximações através de uma série de Fourier.

Aproximação de Funções

Teorema da aproximação universal : Seja $\varphi(\cdot)$ uma função não constante, limitada, monotonicamente crescente e contínua. Seja I_{m_0} o hipercubo unitário m_0 - dimensional $[0,1]^{m_0}$. O espaço de funções contínuas sobre I_{m_0} é denotado por $C(I_{m_0})$. Então, dada uma função f , com $C(I_{m_0}) \ni f$ e $\varepsilon > 0$, existe um inteiro M e conjuntos de constantes reais α_i, b_i e w_{ij} , onde $i = 1, \dots, m_1; j = 1, \dots, m_0$ tal que pode - se definir :

$$F(x_1, x_2, \dots, x_{m_0}) = \sum_{i=1}^{m_1} \alpha_i \varphi \left(\sum_{j=1}^{m_0} w_{ij} x_j + b_i \right)$$
 como uma realização aproxima-

da da função $f(\cdot)$, isto é $\left| F(x_1, x_2, \dots, x_{m_0}) - f(x_1, x_2, \dots, x_{m_0}) \right| < \varepsilon$

x_1, x_2, \dots, x_{m_0} contido no espaço de entrada.

Aproximação de Funções

- O teorema é diretamente aplicável a uma Perceptron MLP pois
 - A função logística é não-constante, limitada, monotonicamente crescente e contínua.
 - A equação ($F(\cdot)$) representa a saída da rede se for suposto que:
 - A rede neural tem m_0 unidades de entrada, m_1 unidades escondidas na única camada escondida e sua entrada é denotada por x_1, x_2, \dots, x_{m_0} .
 - Cada neurônio tem pesos sinápticos $w_{i_1}, w_{i_2}, \dots, w_{m_0}$ e bias b_i .
 - A saída da rede é a combinação linear das saídas das unidades escondidas com $\alpha_1, \alpha_2, \dots, \alpha_{m_1}$ definindo os pesos sinápticos da camada de saída.

Aproximação de Funções

- Teorema da Aproximação Universal
 - Existência de aproximação de função contínua arbitrária.
 - Um camada escondida simples é suficiente para a MLP-BP computar uma aproximação uniforme ϵ para um conjunto de treinamento.
 - Uma camada escondida única é a melhor opção pois tem baixo tempo de treinamento, é fácil de implementar e generalizar.
- Fronteira de Erros de Aproximação de Nodo Escondido:
 - Quanto maior for o número de nodos escondidos, mais acurada é a aproximação.
 - Quanto menor for o número de nodos escondidos, mais acurada é o ajuste (interpolação) empírico.

Arquivo de Treinamento para Generalização

- Uma rede neural generaliza se seu mapeamento entrada-saída for completo ou aproximadamente correto para os dados de teste.
 - O processo de treinamento pode ser visto como a solução de um problema de ajuste de curva para interpolação não-linear dos dados de entrada.
 - Uma RN deixa de generalizar quando está sobretreinada, isto é, quando aprende muitos pares entrada-saída e passa a considerar características não gerais para o conjunto de padrões. Assim, ocorre memorização analogamente a uma tabela *look-up*.
 - É importante buscar mapeamentos não-lineares suaves (*smooth*) para demandar menos esforço computacional.

Arquivo de Treinamento para Generalização

- Occam's Razer
 - Encontra a função mais simples entre várias que satisfazem condições desejadas.
- Fatores que influenciam generalização :
 - Conjunto de treinamento: Tamanho e sua representatividade do ambiente por este conjunto deve ser compatível com a arquitetura fixa da rede neural.
 - Arquitetura da rede neural: Ela deve ser boa o suficiente para generalizar, pressupondo um conjunto de treinamento fixo.
 - Complexidade física do problema tratado. Este fator independe do usuário ou pesquisador.

Seleção de Modelo

- Seleção de MLP-BP com o melhor número de pesos dados N amostras de treinamento. Deve-se encontrar r : para minimizar erro de classificação do modelo treinado pelo conjunto de estimação e testado pelo conjunto de validação.

$$r = \frac{\text{tamanho do conjunto de validação}}{\text{tamanho do conjunto de estimação} + \text{tamanho do conjunto de validação}}$$

- Sugestão: $r = 0.2$
- Kearns(1996) levantou as propriedades qualitativas de r ótimo:
 - Análise com Dimensão VC.
 - Em problemas de baixa complexidade (# de respostas desejadas comparado ao número de amostras), desempenho da validação cruzada é insensível a r .
 - Um único r pode ser adequado para um conjunto grande de funções.

Protocolo de Treinamento

- Treinamento estocástico:
 - Seleção aleatória de amostras.
- Treinamento *Batch*
 - Época (Epoch): Apresentação simples de todos padrões de treinamento.
 - Pesos são atualizados apenas uma vez por época:

$$\Delta w_{ji}(n) = -\eta \frac{\partial E_{av}(n)}{\partial w_{ji}(n)} = -\frac{\eta}{N_{pad}} \sum_{n=1}^{N_{pad}} e_j(n) \frac{\partial e_j(n)}{\partial w_{ji}(n)}$$

Protocolo de Treinamento

- Modo Seqüencial:
 - Atualiza-se pesos para cada amostra de treinamento.
 - Menos armazenamento, maior velocidade de convergência.
 - Risco: Ordem seqüencial pode levar a convergências para ótimos locais.
- Treinamento on-line:
 - Dados de treinamento abundantes.
 - Alto custo para armazenamento.

Protocolo de Treinamento

- Maximização do conteúdo de informação: Todo exemplo propiciar mais pesquisa no espaço de pesos:
 - Uso de um exemplo que resulta no maior erro (exemplo difícil) com respeito às iterações prévias.
 - Uso de um exemplo radicalmente deferente dos demais.
 - Emprego de ordem aleatória para apresentação de exemplos para assegurar propriedade de pertinência a uma dada classe.
 - Ênfase em tratar mais amostras difíceis que fáceis, tendo cuidado com a distorção da distribuição dos exemplos e com presença de “outliers”.
- Valores alvos devem ser tratáveis pela função de ativação, isto é, o valor do padrão desejado deve ser atingido pela função.
 - Quando isto não acontece, os pesos podem tender a infinito.

Algoritmo MLP-BP

- Inicialize aleatoriamente pesos e bias;
- Enquanto E_{av} for insatisfatório e houver disponibilidade computacional:
 - Para cada padrão de entrada:
 - Compute a soma ponderada dos nodos escondidos;
 - Compute a resposta dos nodos escondidos;
 - Compute a soma ponderada dos nodos de saída;
 - Compute a resposta dos nodos de saída;
 - Modifique os pesos que chegam à camada de saída;
 - Modifique os pesos que chegam a cada uma das camadas escondidas;
 - Fim-do-Para
- Fim-do-Enquanto

Pontos Positivos de MLP-BP

- Alto Poder de Representação
 - Qualquer função L2 (função que sobre um intervalo finito possua número finito de descontinuidades) pode ser representada por MLP-BP.
 - Muitas destas funções podem ser aproximadas por uma MLP-BP.
- Larga Aplicabilidade
 - Requer conjunto de treinamento representativo;
 - Não requer significativos conhecimento a priori ou do domínio (problema mal estruturado).
 - Tolera dados ruidosos ou faltosos entre as amostras de treinamento.

Pontos Positivos de MLP-BP

- Facilidade de implementar a aprendizagem:
 - O algoritmo é de simples implementação;
 - Funções empregadas e suas derivadas existem.
- Capacidade de Generalização:
 - Produz bons resultados para padrões não treinados.

Pontos Negativos de MLP-BP

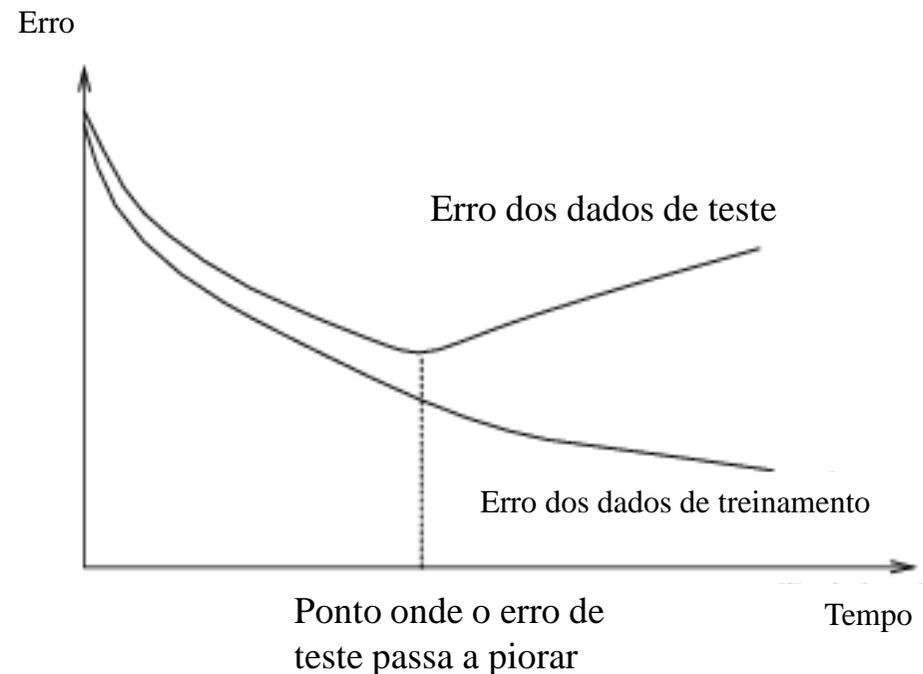
- Aprendizagem de transformações complexas pode demorar para convergir.
- Visualização da MLP-BP como uma caixa preta:
 - Constrói mapeamento mas não justifica as associações.
 - Não explica resultados intuitivamente pois a aprendizagem na camada escondida não tem significado claro no espaço do problema.
- Ausência de maneira teoricamente fundamentada (como em métodos estatísticos) para avaliar a aprendizagem:
 - Nível de confiança que alguém pode ter depois de treinar uma MLP-BP e atingir um determinado erro?
 - Nível de confiança da resposta para uma entrada particular empregando a rede treinada.

Pontos Negativos de MLP-BP

- Limitações decorrentes do gradiente descendente:
 - Garante a redução do erro para mínimo local.
 - Pode cair em mínimos locais, dependendo da superfície de erro:
 - Superfícies com vales e poços podem levar a ótimos locais.
 - Nem toda função que é representável pode ser aprendida, isto é, os erros podem não convergir para o desejado.
 - Possíveis correções:
 - Redes com diferentes número de unidades escondidas que podem levar a superfícies menos inadequadas.
 - Pesos iniciais diferentes para iniciar a otimização a partir de pontos iniciais distintos na superfície.
 - Mecanismo de perturbação aleatória, e.g., cozimento simulado, para escapar de mínimos locais.

Pontos Negativos de MLP-BP

- A generalização não é garantida:
 - Problema de sobre-treinamento: A rede treinada atende com perfeição os padrões treinados não garante respostas acuradas para entradas não-treinadas.



Pontos Negativos de MLP-BP

- A generalização não é garantida:
 - Possíveis soluções:
 - Mais e melhores amostras;
 - Emprego de rede menor;
 - Uso de fronteira de erro para forçar término mais rápido;
 - Adição de ruído nas amostras;
 - Mudar de (x_1, \dots, x_n) para $(x_1 \alpha_1, \dots, x_n \alpha_n)$ onde α_n são pequenos deslocamentos aleatórios.
 - Validação cruzada:
 - Separe amostras (cerca de 10%) para dado de teste;
 - Verifique o erro nos dados de teste periodicamente;
 - Interrompa a aprendizagem quando erro passar a crescer.

Pontos Negativos de MLP-BP

- Paralisação da rede com função de ativação tipo sigmoide

$$S(x) = 1/(1 + e^{-x}), \quad \text{logo } S'(x) = S(x)(1 - S(x)) \rightarrow 0$$

assim $x \rightarrow \pm\infty$ então $S'(x) \rightarrow 0$.

Se x cair na região de saturação, $S(x)$ não altera seu valor de modo significativo independentemente do crescimento da magnitude of x .

– Entrada para um nó pode cair dentro de uma região de saturação quando alguns pesos se tornam muito grandes durante a aprendizagem. A implicação disto é que os pesos param de crescer pois a derivada da função de ativação tende a zero.

– Possíveis soluções:

- Uso de funções que não saturem;
- Normalização periódica dos pesos.

$$w_{kj} = w_{kj} / \sqrt{\sum_{i=1}^{ncama} (w_{kj})^2}$$



Técnicas para Melhoraria em MLP-BP

- Determinação de função de ativação
- Escalonamento de valores entrada/saída
- Treinamento com dados ruidosos ou artificiais
- Determinação do número de unidades escondidas
- Inicialização dos pesos
- Termo de Momento
- Taxa de Aprendizagem
- Decaimento dos pesos
- Poda da rede
- Aprendizagem a partir de pistas
- Critério de parada

Determinação da Função de Ativação

- Propriedades da função de ativação:
 - Não linearidade;
 - Saturação para valores máximos e mínimos;
 - Continuidade e suavidade;
 - Monotonicidade, que não é essencial.
- A função sigmóide tem estas propriedades.
 - O número de iterações de treinamento tende a diminuir quando a função sigmoide é antiassimétrica ($f(-v)=-f(v)$) ao invés de não-simétrica.
 - A centralização em zero facilita o treinamento.
 - O alcance (*range*) e a inclinação devem ser compatíveis com as amostras de entrada e saída.

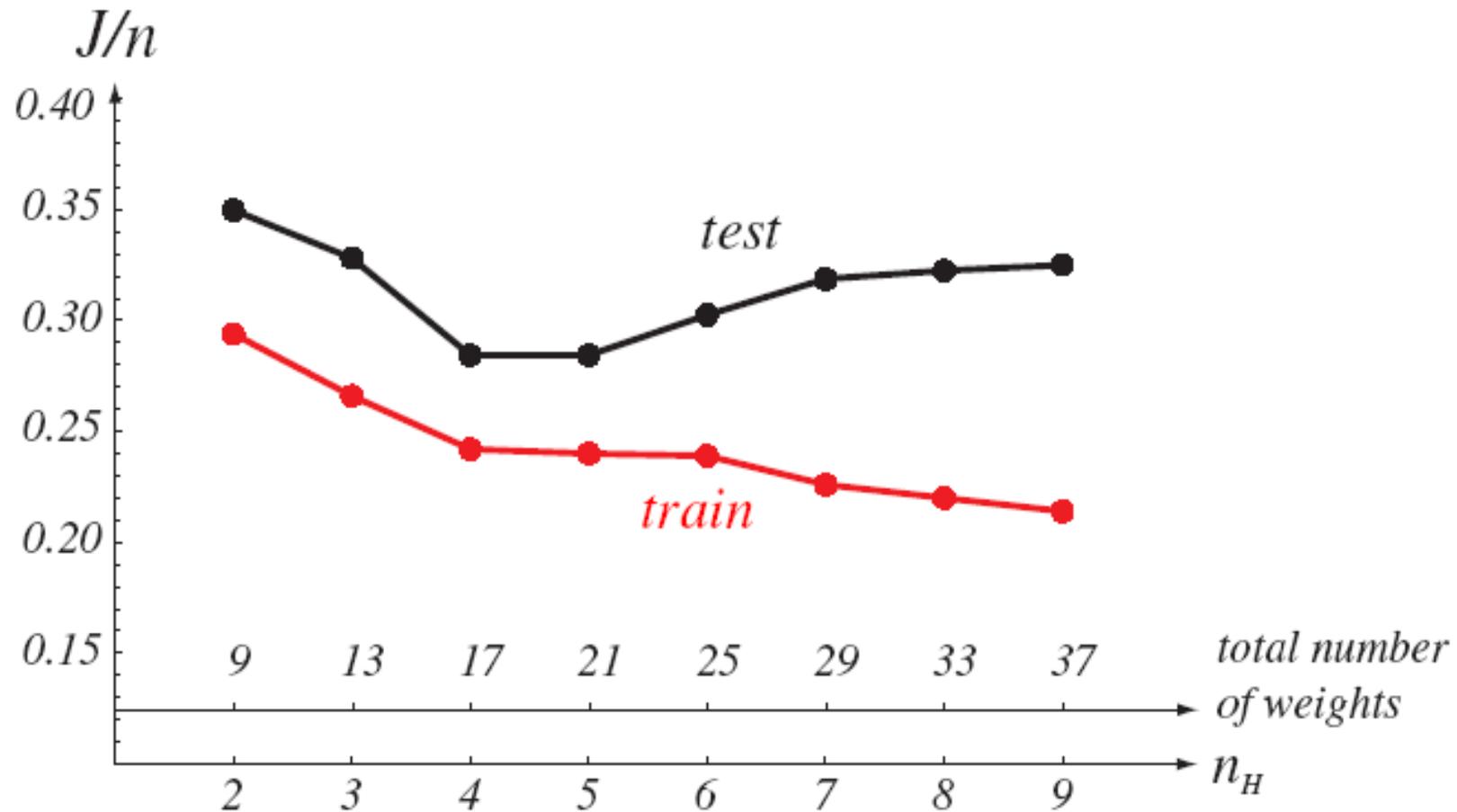
Escalonamento de Valores de Entrada/Saída

- Deve-se padronizar:
 - Diferenças altas de escala pode provocar redução de erros apenas nos valores mais altos de escala.
- Normalização das entradas: Pré-processamento das entradas para obter média pequena se comparada a seu desvio padrão, e variância unitária.
 - Para acelerar treino, usa-se entradas não-correlacionadas e escalonadas para se obter covariâncias aproximadamente iguais.
- Valores para alvos: Estes valores devem estar dentro do alcance da função de ativação:
 - Nenhuma saída deve atingir a parte saturada da função de ativação.
 - Sugestão para representação local: Binária (+1 positivo; -1 negativo).

Treinamento com Dados Ruidosos ou Artificiais

- Para conjunto de treinamento pequeno, pode-se gerar padrões de treinamento adicionais através de:
 - Adição de ruído Gaussiano d -dimensional aos verdadeiros padrões de treinamento (ausência de informações do problema) com variância menor que um.
 - Geração artificial de padrões (existência de informações sobre o problema).
- O número de unidades escondidas deveria ser menor que o número total de amostras para treinamento (N_{pad}). Uma sugestão inicial é $N_{pad}/10$.

Determinação do Número de Unidades Escondidas



Inicialização de Pesos

- Não se deve inicializar todos os pesos em zero pois não há mapeamento inicial restringindo a busca à vizinhança da origem.
- Pesos inicialmente altos tendem a saturar as unidades. Valores inicialmente baixos levam à operação em região muito plana em torno da origem da superfície de erro.
 - Valores de pesos sinápticos cujo desvio padrão do campo local induzido se situe entre as partes linear e saturada da sigmoideal.
- Seleção de sementes adequadas para aprendizagem rápida e uniforme.

Inicialização de Pesos

- Para dados padronizados:
 - Escolha aleatória de distribuição única.
 - Utilize igualmente pesos positivos e negativos $-\omega < w < \omega$
 - Para ω pequeno, então *net* é linear;
 - Para ω grande, então *net* satura rapidamente.
- Recomendações de valores iniciais:
 - Pesos entrada-escondida $-\frac{1}{\sqrt{n}} < w_{ji} < +\frac{1}{\sqrt{n}}$
 - Pesos escondida-saída: $-\frac{1}{\sqrt{m}} < w_{kj} < +\frac{1}{\sqrt{m}}$

Termo de Momento

- Pode ajudar a que um processo de aprendizagem não fique preso em um mínimo local.

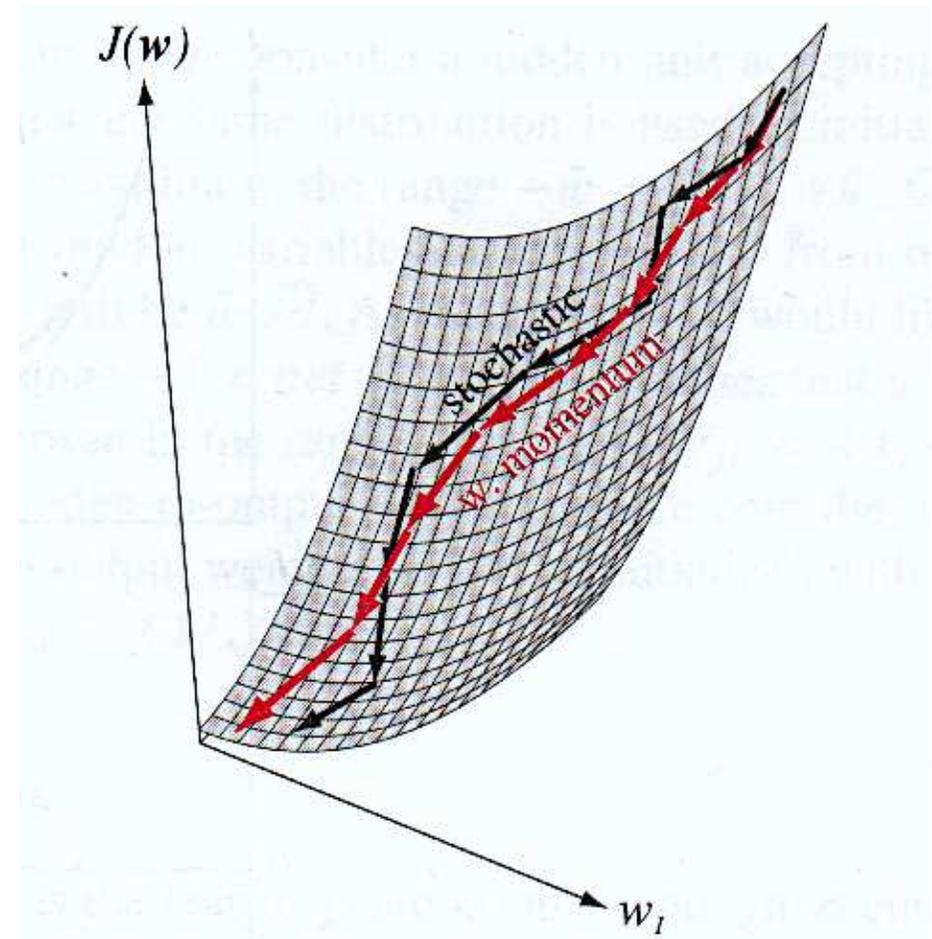
$$\mathbf{w}(m+1) = \mathbf{w}(m) + (1-\alpha)\Delta\mathbf{w}_{bp}(m) + \alpha\Delta\mathbf{w}(m-1)$$

– onde α é a constante do momento que para convergir $0 \leq |\alpha| \leq 1$.

- Valor típico = 0.9
- Se a derivada parcial tiver mesmo sinal em iterações consecutivas, o momento aumenta o ajuste dos pesos que estiverem na mesma direção do ajuste atual, acelerando a convergência.
 - Caso contrário, momento leva à diminuição do ajuste.

Termo de Momento

- O termo de momento:
 - Permite que a rede aprenda mais rapidamente em regiões de planícies na superfície de erro.
 - Impede variações súbitas de direção durante a atualização dos pesos.



Taxa de Aprendizagem

- Taxas de aprendizagem pequenas levam a ajustes mais sutis mas demoram mais a convergir.
 - Taxa de aprendizagem fixa deve ser menor que 1.
 - Inicie com taxa alta e gradualmente diminua seu valor.
 - Inicie com taxa baixa e periodicamente dobre seu valor até o erro começar a aumentar.
 - Atribua taxas mais altas para padrões pouco representados.
 - Determine máximo passo para cada estágio de aprendizagem para evitar sobrevalor de erro.
- Taxa da última camada deve ser menor que das anteriores.
 - Última camada tem gradiente local mais alto.
 - Permite ajustes sutis.

Taxa de Aprendizagem

- Sugestão de LeCun: A taxa de aprendizagem é inversamente proporcional à raiz quadrada do número de conexões sinápticas da camada.
- Taxa de aprendizagem adaptativa (método delta-bar-delta)
 - Cada peso w_{kj} tem sua própria taxa η_{kj}
 - Se Δw_{kj} mantém a direção, aumenta a taxa pois E tem curva suave na vizinhança do peso atual.
 - Se Δw_{kj} muda de direção, diminua a taxa pois E tem curva acentuada na vizinhança do peso atual.

Decaimento do Peso

- Heurística: Manutenção do peso pequeno.
 - Simplifica rede e evita sobre-treinamento.
- Possibilidade de não haver algum tipo de normalização: Inicie com pesos altos e decaia seus valores durante o treinamento:

$$\mathbf{w}^{new} = \mathbf{w}^{old} (1 - \epsilon)$$

- Pesos pequenos são eliminados.

Poda da Rede

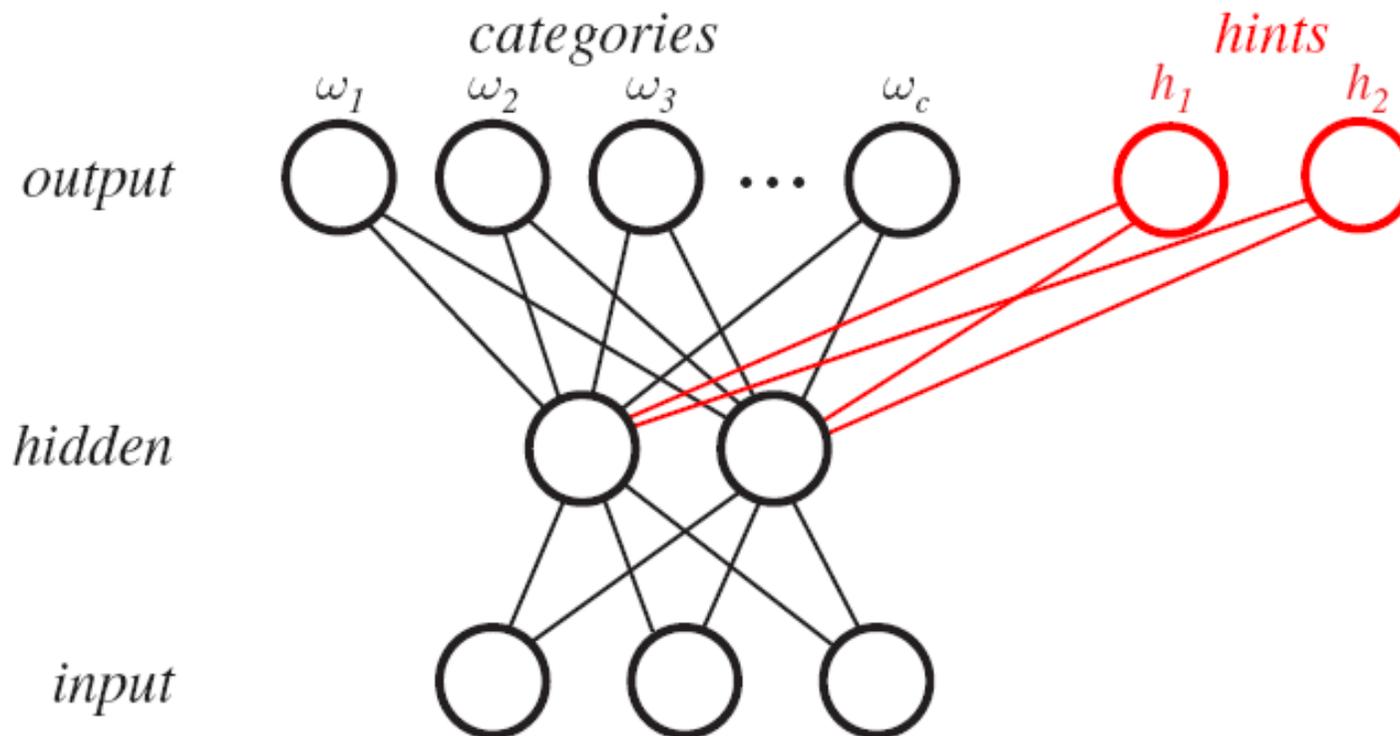
- Minimização da rede tende a melhorar generalização
 - Diminui probabilidade de aprender peculiaridades ou ruídos.
- Poda da rede elimina pesos sinápticos com magnitude pequena.
- Complexidade-regularização
 - Compromisso entre confiabilidade dos dados de treinamento e desempenho do modelo.
 - Aprendizagem supervisionada minimizando função de risco:
$$R(w) = E_s(W) + \lambda E_c(W)$$
onde $E_s(W)$ é a medida de desempenho
 $E_c(W)$ é a penalização por complexidade

Aprendizagem a Partir de Pistas

- Emprego de informação a priori para ser incluída no processo. Tal informação inclui propriedades invariantes, simetrias, outros conhecimentos sobre a função de ativação.
- Adição de unidades de saída para resolver problema secundário
 - O problema secundário é diferente mas relacionado ao original.
- Treinamento com problema original e secundário simultaneamente.
- Descarte unidades de pistas após treinamento.
- Vantagens
 - Melhora na seleção de características;
 - Melhora representação de unidades escondidas.

Aprendizagem a Partir de Pistas

- Interessante se houver dados para treinamento que sejam insuficientes para chegar a precisão desejada.

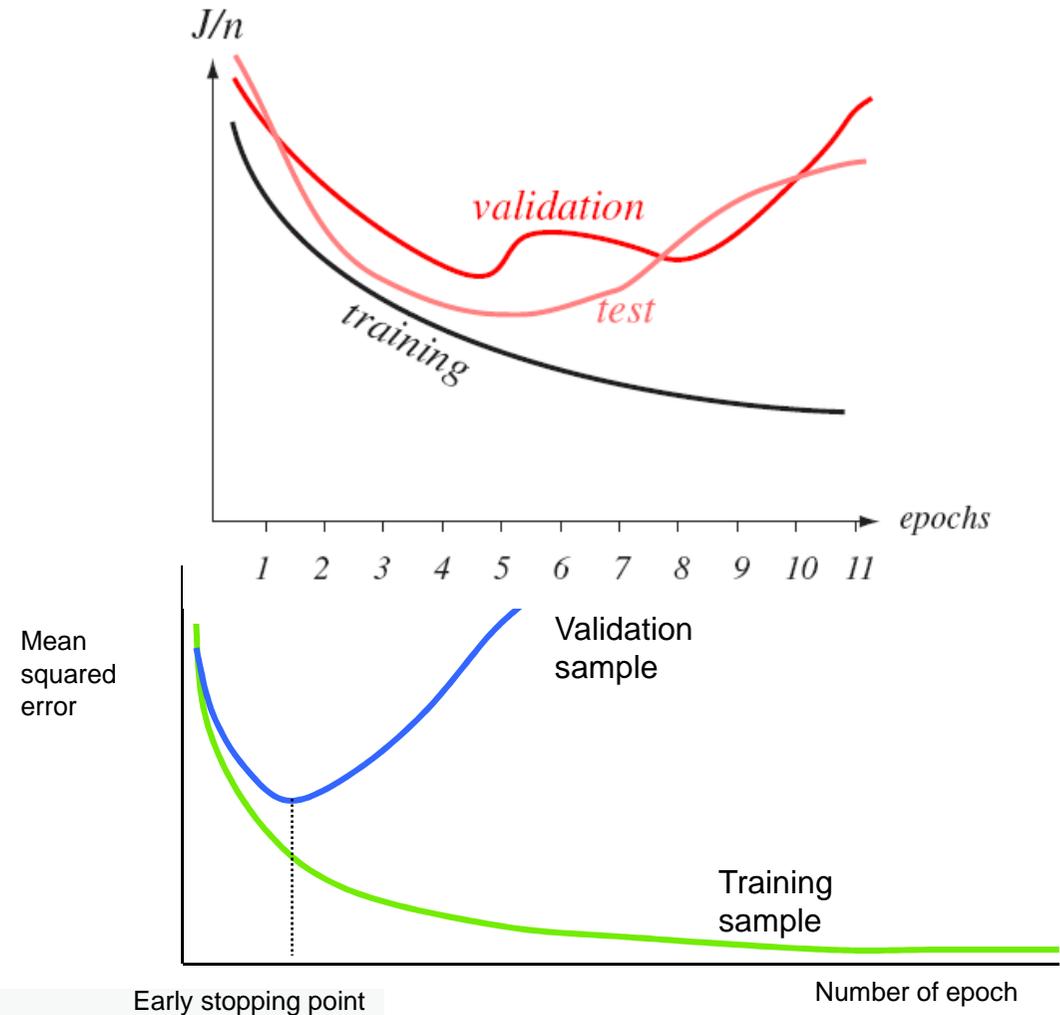


Critério de Parada

- Alguns possíveis critérios de parada:
 - Vetor de gradiente nulo, indicando mínimo local ou global.
 - Medida de erro estacionária.
 - Medida de erro dentro de faixa de aceitação.
- Progresso de treinamento a partir de pesos iniciais pequenos:
 - Início: Linearidade.
 - Avançado: A não linearidade é alcançada.
 - Portanto, término prematuro do treinamento se parece com decaimento de pesos.

Parada com Conjunto de Validação Separado

- Método de antecipação de parada:
 - Computar os pesos sinápticos depois de parte do treinamento, tomando valores fixos de pesos.
 - Reassumir treinamento depois de calcular erro de validação.



Referências

- Hassoun, M. H. (1995). *Fundamentals of Artificial Neural Networks*. Cambridge: The MIT Press.
- Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation*. IEEE Press.