Redes Neurais Artificiais

Baseada em:

- Stuart Russel e Peter Norving - “Inteligência Artificial” – seção 20.5
- Jorge M. Barreto – “Introdução às Redes Neurais Artificiais” - http://www.inf.ufsc.br/~barreto/tutoriais/Survey.pdf
- Cassia Yuri Tatibana e Deisi Yuki Kaetsu – “Uma Introdução às Redes Neurais” - http://www.din.uem.br/ia/neurais/#neural

Profa. Josiane M. Pinheiro

outubro/2008

Inteligência Artificial

- Técnicas para simulação de raciocínio
  - Simbólicas
    - Regras e fatos
    - Dedução formal
  - Não-simbólicas
    - redes neurais (sistemas conexionistas)

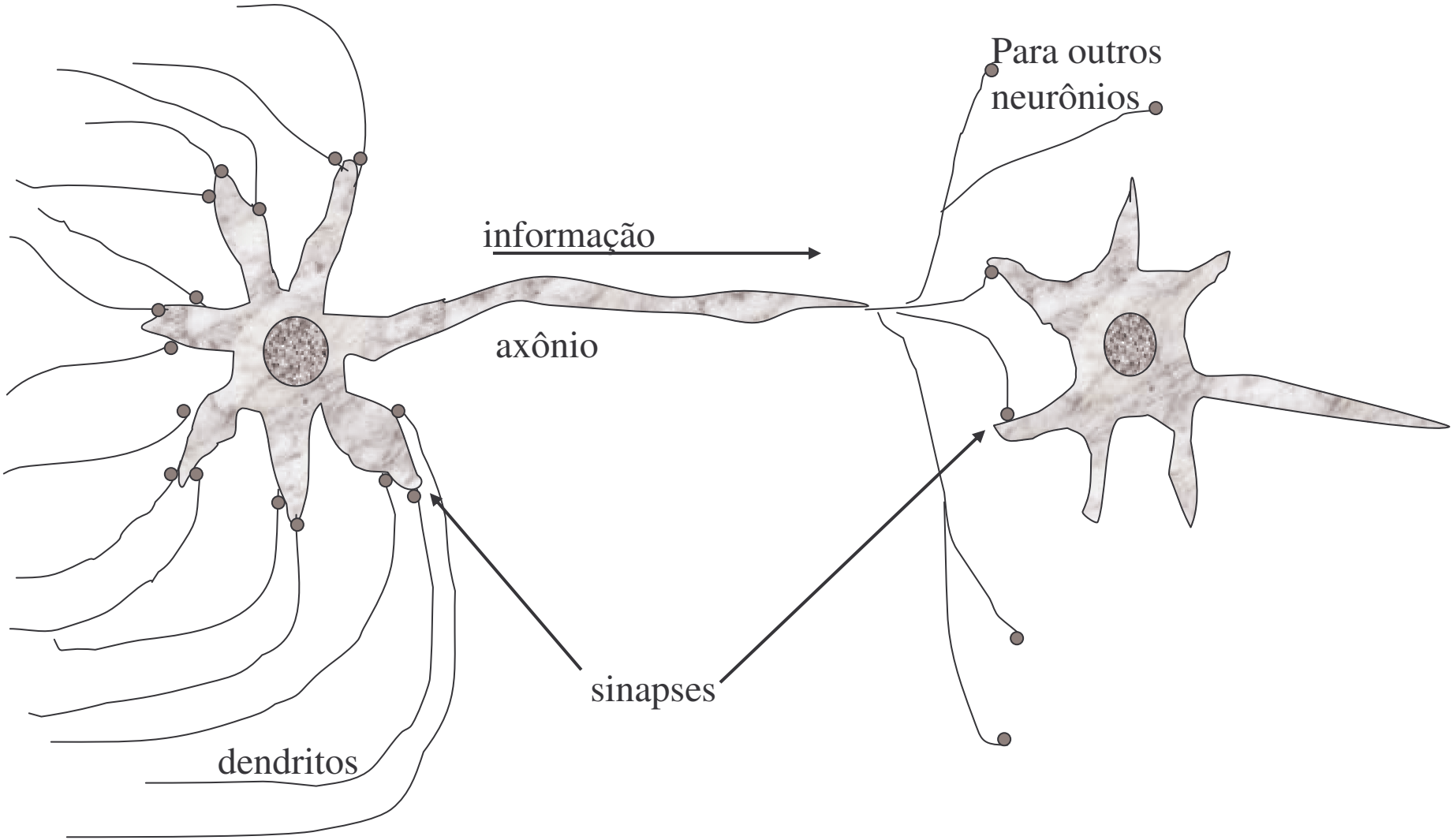
Redes Neurais

- Imitam a estrutura física do cérebro como modelo
- Tem habilidade de:
  - Executar computação distribuída
  - Tolerar entradas ruidosas e
  - Aprender
- Uma das formas mais populares e efetivas de sistemas de aprendizagem

Neurônio Biológico X Neurônio Artificial

- Um neurônio biológico consiste de:
  - Um corpo celular ou soma
  - Ramificações chamadas de dendritos conduzem sinais das extremidades para o corpo celular
  - Uma ramificação única chamada de axônio conduz o sinal do corpo celular para outras extremidades
  - As conexões entre os neurônios são as chamadas de sinapses

Um neurônio biológico



Neurônio Biológico X Neurônio Artificial

- Um neurônio artificial (ou elemento processador)
  - Axônios e dendritos são modelados através de conexões
  - Sinapses são modeladas através de ponderações ou pesos de ajuste entre as conexões
    - Estes pesos de ajuste são modificados enquanto a rede aprende algo
    - São eles que verdadeiramente “guardam” o conhecimento de uma rede neural

Funcionamento de uma rede neural

- Através das sinapses os dendritos recebem sinais excitatórios ou inibitórios
- Esses sinais que chegam ao neurônio através dos dendritos, são somados no corpo celular
- Quando a soma atinge um limiar o neurônio dispara um sinal para os outros (através do axônio)
- As sinapses regulam a forma de como a informação passa pelo neurônio

Propagação da informação

- Não é qualquer informação presente na entrada que se propaga para a saída
- É necessário uma combinação correta entre transmissores e receptores nervosos
- Quando esta combinação acontece, o neurônio dispara (propaga informação)
- A eficácia dessa combinação é determinada pela presença em quantidade suficiente do neurotransmissor na conexão nervosa

Ajuste das sinapses

- Em uma rede neural biológica
  - Sinapses são ajustadas primeiramente baseadas no genótipo
  - No decorrer da vida as sinapses são ajustadas através do treino ou aprendizado
    - Quanto mais uma conexão é utilizada, maior é a quantidade de neurotrasmissores liberados, e mais forte se torna a conexão
      - Conexões são ajustadas – se tornam mais fracas ou mais fortes
      - Conexões são construídas

Ajuste das sinapses

- Em uma rede neural artificial
  - Sinapses normalmente possuem valores randômicos inicialmente
    - Algum conhecimento prévio do domínio pode ajudar a colocar os pesos sinápticos iniciais
  - Enquanto a rede aprende, as sinapses são ajustadas pra fornecer saídas de melhor qualidade
    - Em algumas tipos de redes apenas são feitos ajustes dos pesos sinápticos já existentes
    - Outros tipos de redes criam neurônios e conexões entre eles para melhorar a qualidade da saída

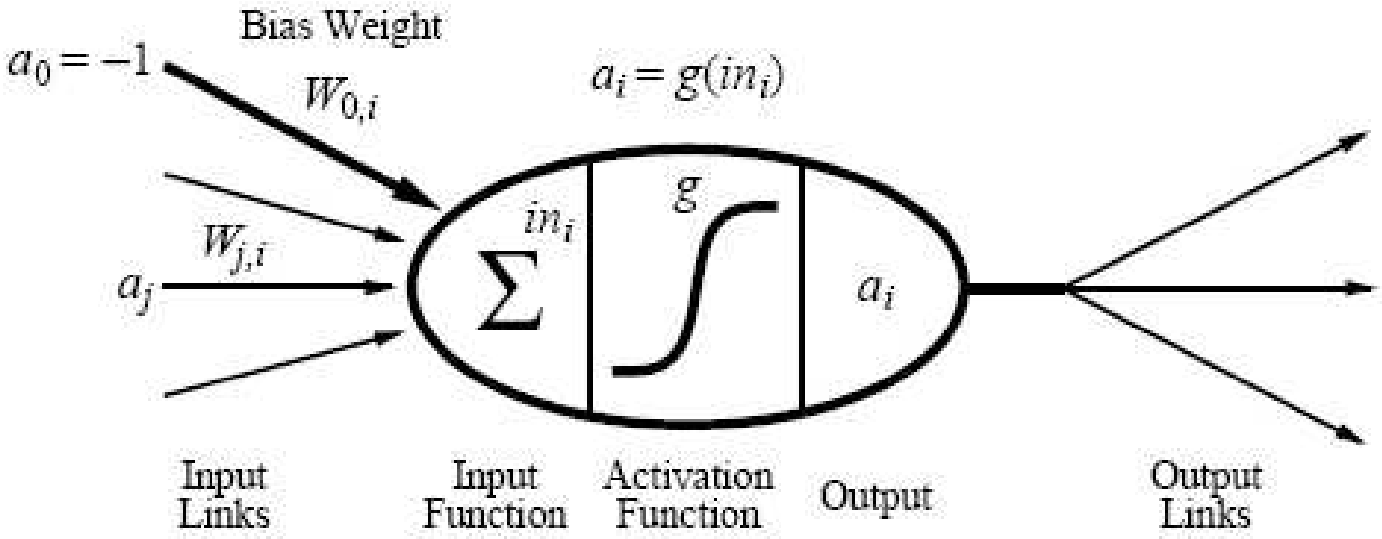
Modelo de uma rede neural artificial

- É caracterizado por:
  - Uma quantidade de neurônios (ou elementos processadores)
  - A forma das conexões entre eles (topologia da rede)
  - Seu esquema de aprendizagem
- Pode ser vista como um grafo orientado composto por:
  - Um certo número de nós (neurônios) que operam em paralelo
  - Cada nó possui um certo número de entradas e somente uma saída (que pode dividir-se em cópias) que se propaga

Modelo de uma rede neural artificial

- Cada entrada de um neurônio está associada a um peso sináptico:
  - Excitátório (positivo) ou
  - Inibitório (negativo)
- O sinal de entrada pode assumir uma variação:
  - Contínua (desde –1 até 1) – grau de veracidade
  - Discreta (valores binários 0 e 1) – falso ou verdadeiro

Neurônio artificial simplificado
(por McCulloch e Pitts – 1943)



A soma ponderadas das entradas

- O neurônio avalia seus sinais de entrada
- Realiza o somatório ponderado de suas entradas através dos pesos sinápticos associadas a cada uma delas

$$in_i = \sum_{j=0}^{n} W_{ji} a_j$$

Onde:

- in_i – a soma ponderada dos n sinais de entrada do neurônio i
- W_{ji} – o valor do peso sináptico associado à conexão entre os neurônios j e i
- a_j – a saída do j-ésimo neurônio

A função de ativação sobre a entrada

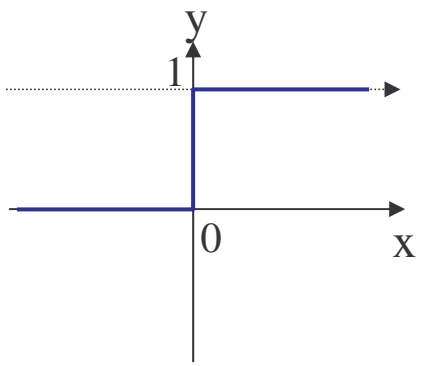
- O sinal de saída:
  - Aplica-se o somatório ponderado das suas entradas numa função de ativação
  - Função de ativação: determina o valor de saída (nível de ativação) do neurônio

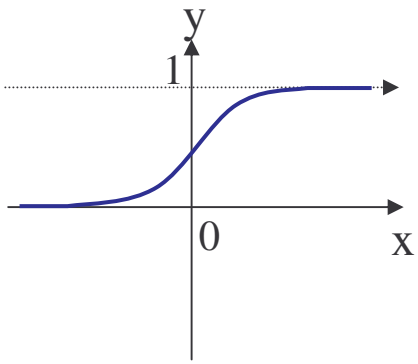
$$a_i = g(in_i) = g\left(\sum_{j=0}^{n} W_{ji} a_j\right)$$

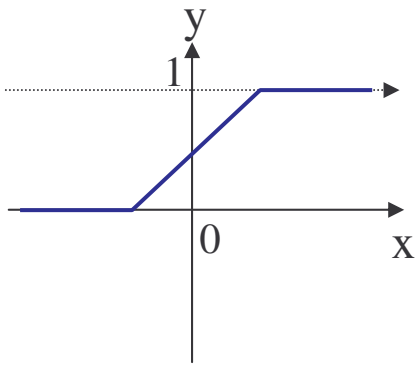
Onde:

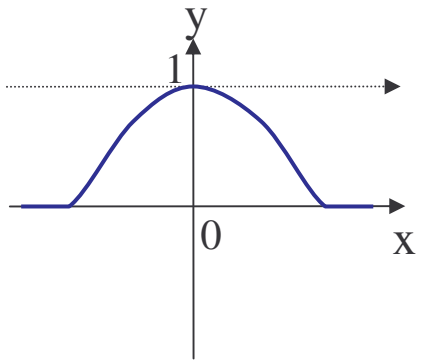
- g – função de ativação do neurônio i
- a_i – saída do neurônio i

Formas de função de ativação (ou função de transferência)

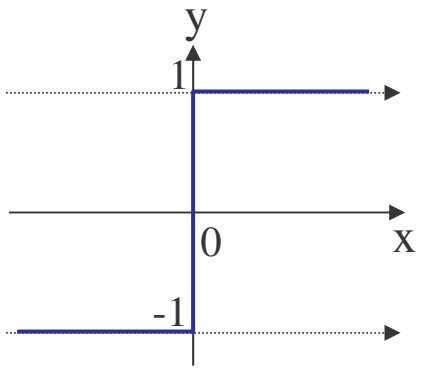




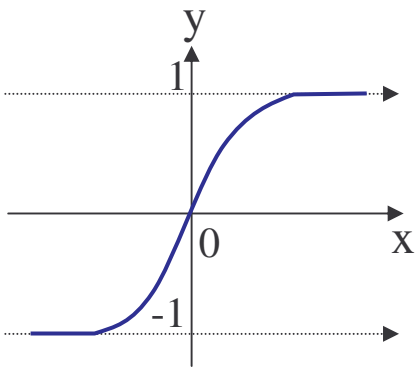




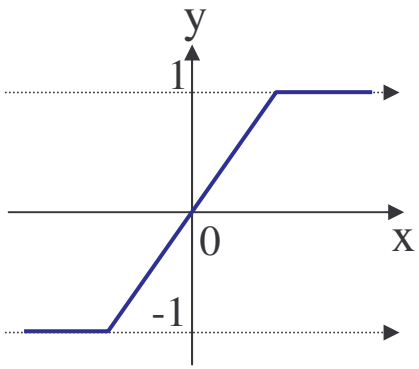
Radial básica



limiar

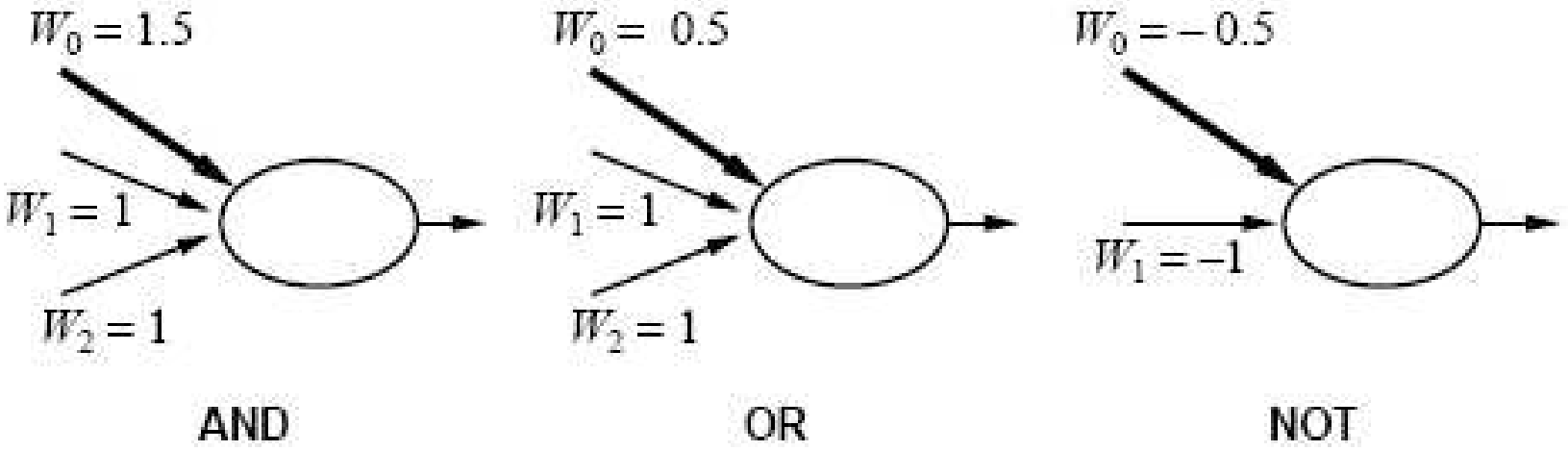


Sigmóide



Rampa-limitada

Neurônios com uma função de ativação de limiar podem atuar como portas lógicas



Funções de ativação

- A função de ativação caracteriza o neurônio em:
  - Linear ou não-linear
- Modelo não-linear mais simples: função de limiar
  - Quando o somatório ponderado das entradas atinge um certo valor (normalmente zero), o neurônio dispara
  - Conhecido como neurônio binário
- Modelo não-linear mais utilizado: função sigmóide
  - Saída é proporcional a soma ponderada das entradas
  - Função de ativação que mais se aproxima de um neurônio real

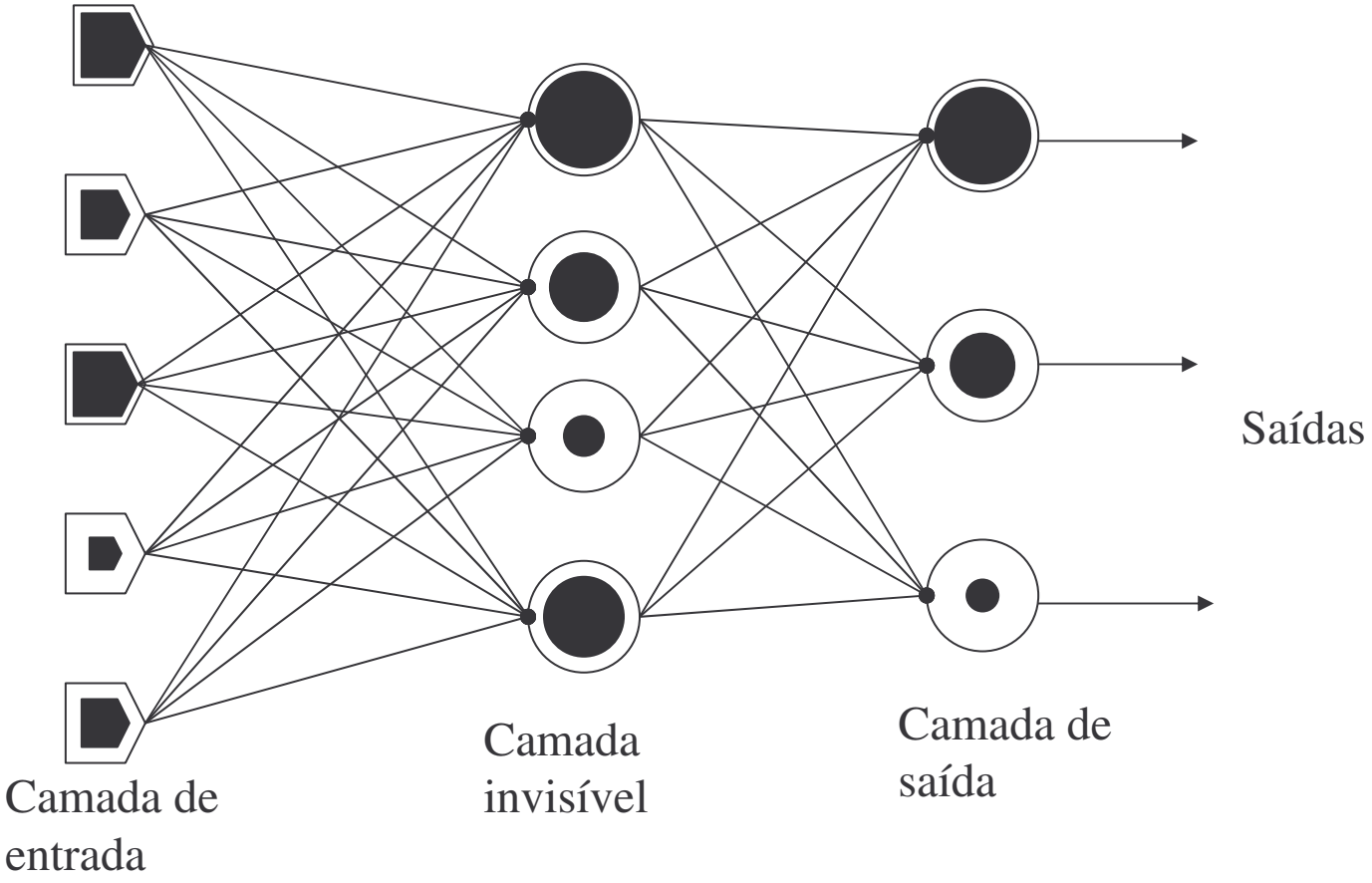
Estruturas das redes neurais artificiais

- Redes acíclicas ou de alimentação direta (*feedforward*)
  - Representa uma função do seu estado atual, não existe nenhum estado interno além dos pesos sinápticos
- Redes cíclicas ou recorrentes (*feedback*)
  - Utiliza sua saída para realimentar suas próprias entradas
  - Os níveis de ativação da rede formam um sistema dinâmico
  - A resposta da rede a uma determinada entrada depende de seu estado inicial, que pode depender de entradas anteriores (memória de curto prazo)
  - São mais próximas do modelo do cérebro, mas são mais difíceis de compreender

Redes de alimentação direta *(feedforward)*

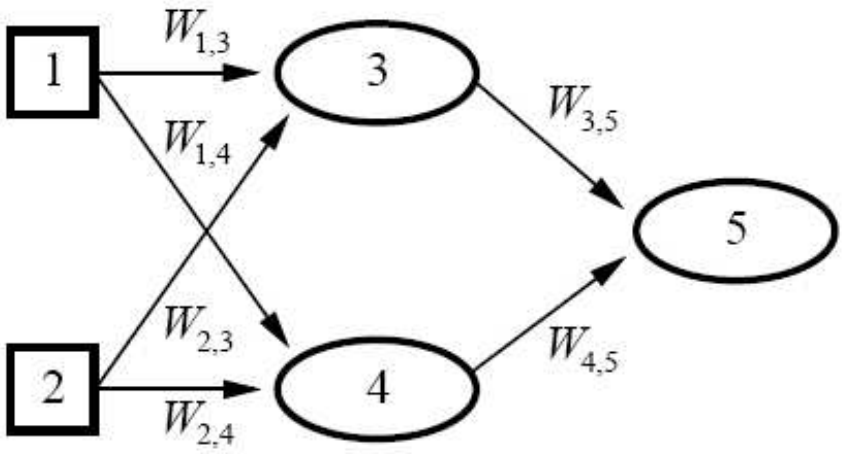
- Normalmente são representadas em camadas:
  - Neurônios que recebem sinais de excitação são chamados de camada de entrada ou primeira camada
  - Neurônios que tem sua saída como saída da rede são a camada de saída ou última camada
  - Neurônios que não pertencem nem à camada de saída e nem a camada de entrada são chamados e neurônios internos (ou *hidden*) podendo se organizar em uma ou mais camadas internas (*hidden layers*)
  - Cada camada recebe apenas entradas de unidades situadas na camada imediatamente precedente

Estrutura de uma rede *feedforward*



Normalmente os nós entre as camadas são completamente ligados

Uma rede de alimentação direta representa uma função de suas entradas



- Seja o vetor de entrada $\boldsymbol{x} = (x_1, x_2)$, as ativações de entrada são definidas como $(a_1, a_2) = (x_1, x_2)$ e a rede calcula:
- $a_5 = g(W_{3,5}*a_3 + W_{4,5}*a_4)$
- $a_5 = g(W_{3,5}*g(W_{1,3}*a_1 + W_{2,3}*a_2) + W_{4,5}*g(W_{1,4}*a_1 + W_{2,4}*a_2))$

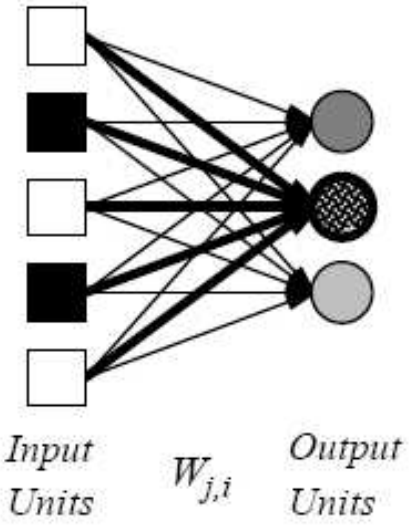
Uma rede de alimentação direta representa uma função de suas entradas

- $a_5 = g(W_{3,5}*g(W_{1,3}*a_1 + W_{2,3}*a_2) + W_{4,5}*g(W_{1,4}*a_1 + W_{2,4}*a_2))$
- A saída da rede como um todo é uma função de suas entradas
- Os pesos da rede atuam como parâmetros dessa função
- Escrevendo-se **W** para os parâmetros a rede calcula uma função $h_{\mathbf{W}}(\boldsymbol{x})$
- Ajustando os pesos, mudamos a função que a rede representa
  - Essa é a forma de aprendizagem das redes neurais

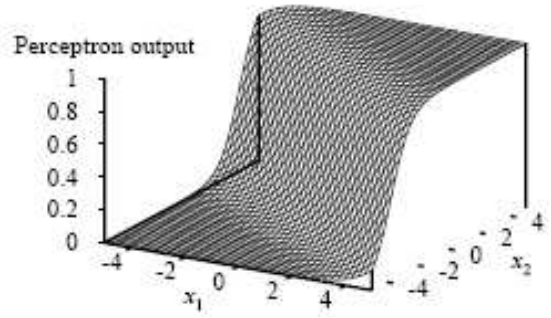
Redes de alimentação direta de uma única camada – Perceptrons

- É uma rede com todas as entradas conectadas diretamente às saídas
- Cada unidade de saída é independente das outras – cada peso afeta apenas uma das saídas

Uma rede de perceptrons com três unidades de saída que compartilham cinco entradas



Saída de um perceptron de duas entradas com uma função de ativação sigmóide

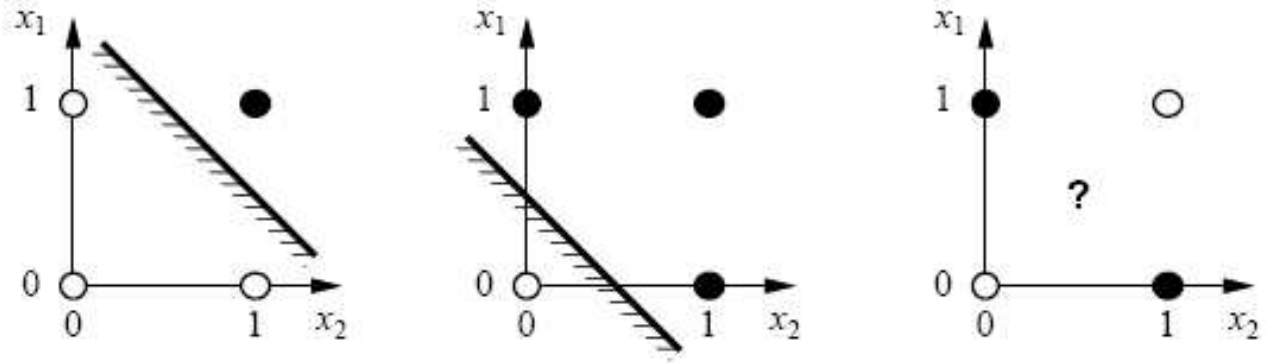


Redes de alimentação direta de uma única camada – Perceptrons

- Uma rede com um única unidade de saída:
  - Com uma função de ativação de limiar podemos visualizar o perceptron como uma representação de uma função booleana (E, OU, NOT, como vimos)
  - Pode representar algumas funções booleanas bastante "complexas" de maneira compacta
    - Exempo: função maioria com cada $W_j = 1$ e limiar $W_0 = n/2$

O que os perceptrons não podem representar

- Podemos ver a equação do perceptron de limiar como uma reta que separa as entradas
- Por esta razão o perceptron é chamado de separador linear
- E pode representar apenas funções linearmente separáveis



(a) x_1 and x_2 (b) x_1 or x_2 (c) x_1 xor x_2

Aprendizagem em redes neurais

- Aprendizado supervisionado
  - São fornecidos pares de entra-saída casados p/ a rede
  - A cada entrada fornecida para a rede a saída é comparada com a saída correta
    - Se a saída obtida for diferente da saída correta os pesos sinápticos são ajustados para minimizar o erro
- Aprendizado não-supervisionado
  - Não possui acesso a saída desejada
  - Aprende por mecanismo de estímulo-reação
    - Não existe ninguém para indicar se a associação feita está correta
    - Processo interno próprio de categorização da informação de entrada

Os algoritmos de aprendizagem para redes neurais

- A idéia por trás da maioria deles é ajustar os pesos da rede para minimizar alguma medida de erro no conjunto de treinamento
  - A medida clássica de erro é a soma dos erros quadráticos
- A aprendizagem é formulada como uma busca de otimização no espaço de pesos
- Os pesos sinápticos variam somente durante a etapa de treinamento
- Treinar uma rede neural = ajustar pesos sinápticos
  - Manualmente ou automaticamente (algoritmos)

Aprendizagem de um perceptron com função de ativação sigmóide

- O erro quadrático para um único exemplo de treinamento $\boldsymbol{x}$ e saída verdadeira $\boldsymbol{y}$ é escrito como:

$$E = \frac{1}{2} Err^2 \equiv \frac{1}{2} (y - h_{\mathbf{w}}(\mathbf{x}))^2$$

- O gradiente descendente é usado para reduzir o erro quadrático
- Calculando a derivada parcial de *E* em relação a cada peso temos:

$$\frac{\partial E}{\partial W_j} = -Err * g'(in) * x_j$$

- Atualizamos o peso como a seguir:

$$W_j = W_j + \alpha * Err * g'(in) * x_j$$

- Onde α é a taxa de aprendizagem

função APRENDIZAGEM-DE-PERCEPTRON (*exemplos*, *rede*) **retorna** uma hipótese de perceptrons

entrada: *exemplos*, um conjunto de exemplos, cada um com entrada **x** = $x_1,..., x_n$ e saída *y*

rede, um perceptron com pesos W_j, $j = 0...n$ e função de ativação *g* (diferenciável)

repita

para cada *e* **em** *exemplos* **faça**

$$in \leftarrow \sum_{j=0}^{n} W_j x_j[e]$$

$$Err \leftarrow y[e] - g(in)$$

$$W_j \leftarrow W_j + \alpha * Err * g'(in) * x_j[e]$$

até que algum critério de parada seja satisfeito

retornar HIPÓTESE-DA-REDE-NEURAL(*rede*)

- Para perceptrons de limiar onde *g'(in)* é indefinido, a regra do perceptron original simplesmente omite este valor.
- Para perceptrons que utilizam a função sigmóide, *g' = g*(1-g)*

Regra de Aprendizado de Widrow-Hoff (ou Delta) para perceptrons de limiar

- Ignora a função de ativação fazendo *g(in) = in*

$$in \leftarrow \sum_{j=0}^{n} W_j x_j[e]$$

$$Err \leftarrow y[e] - in$$

$$W_j \leftarrow W_j + \alpha * Err * x_j[e]$$

- Uma rápida análise:
  - Quando o erro é positivo, a saída da rede é pequena demais, e os pesos devem ser aumentados
  - Quando o erro é negativo, a saída da rede é grande demais e os pesos devem ser diminuídos

- Depois de encontrar os pesos que minimizam o erro quadrático, podemos reinserir a função de limiar para produzir os valores necessários

Regra de Aprendizado Delta Generalizada para perceptrons com função sigmóide

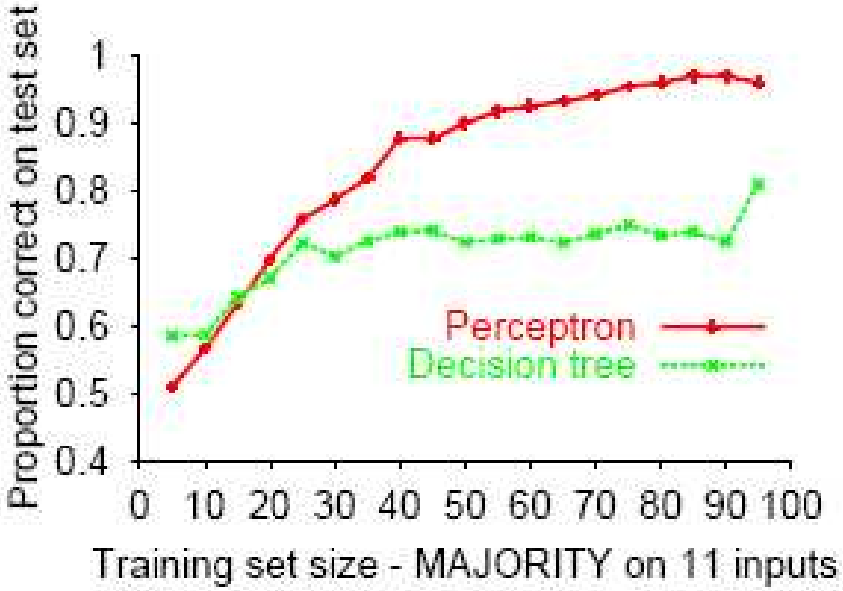
- A derivada da função sigmóide *g'(in) = g(1-g)* resultando em:

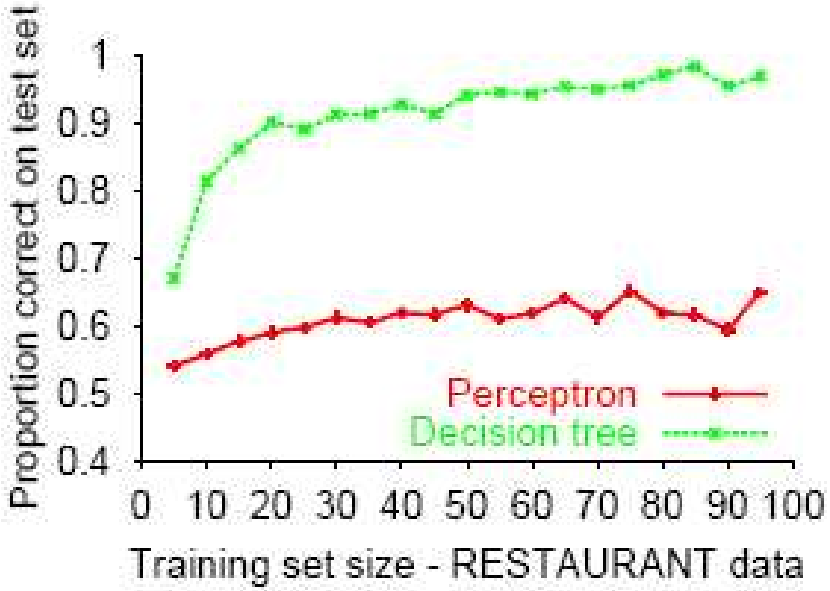
$$in \leftarrow \sum_{j=0}^{n} W_j x_j[e]$$

$$Err \leftarrow y[e] - g(in)$$

$$W_j \leftarrow W_j + \alpha * Err * g * (1-g) * x_j[e]$$

Curva de aprendizagem perceptrons X árvores de decisão



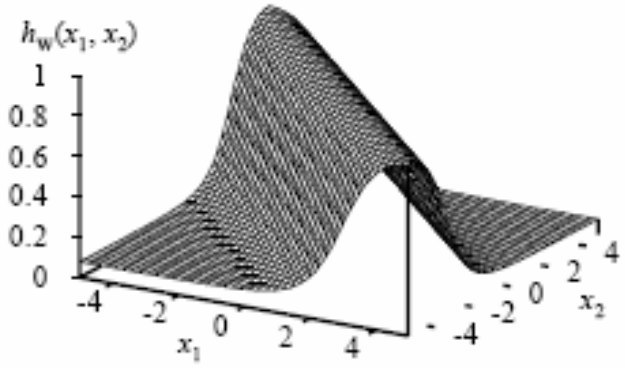


Exercício

- Considere um perceptron com duas unidades de entrada com valores reais e uma unidade de saída
- Todos os pesos iniciais e o desvio (bias) são iguais a 0,5.
- A saída deve ser 1 para os valores de entrada $x_1 = 0,7$ e $x_2 = -0,3$
- Mostre como a regra de aprendizado e Widrow-Hoff altera os pesos de perceptron para que a rede tenha a saída correta para este exemplo
- Considere $\alpha = 0,3$

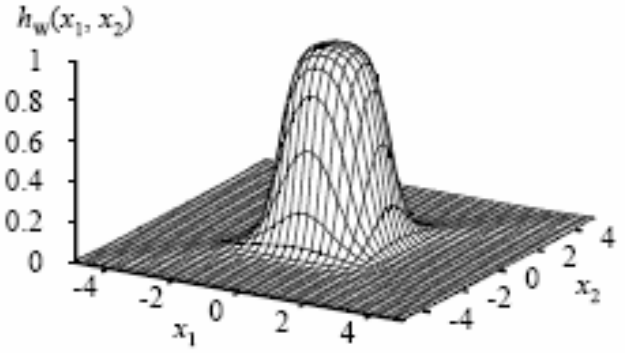
Redes de alimentação direta de várias camadas

- A vantagem em adicionar camadas ocultas é que ela aumenta o espaço de hipóteses que a rede pode representar



Combinação de duas funções de limiar oposta para obter um cume

(Uma camada oculta que combina a saída de dois perceptrons)



Combinação de dois cumes para formar uma coluna

(Uma segunda camada oculta que combina a saída da camada anterior

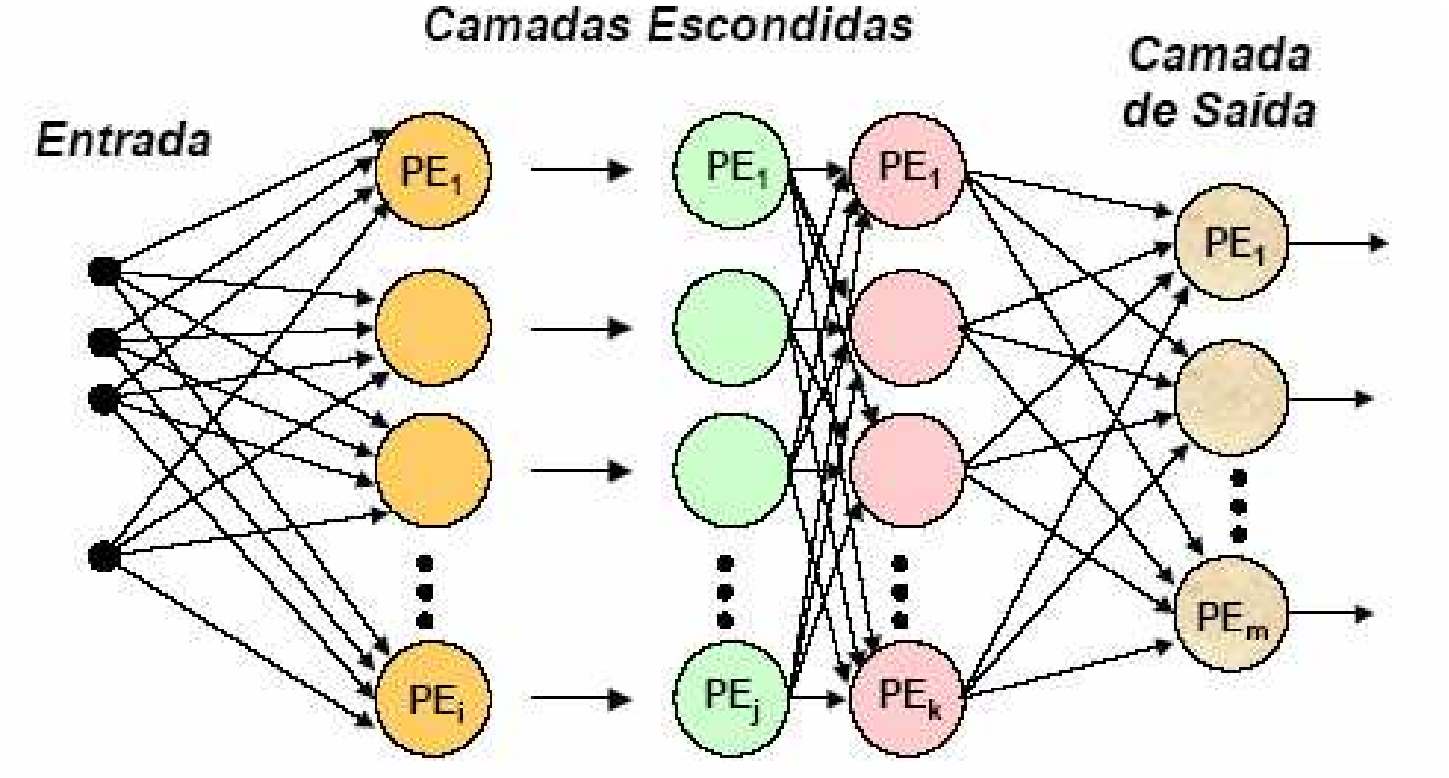
Aprendizagem em redes de várias camadas

- Algoritmo de aprendizagem *back propagation*
  - É o mais difundido em redes multicamadas
  - Aprendizado supervisionado
  - Durante a operação nenhuma informação é retropropagada
  - Na fase de aprendizagem o sinal de erro da saída é retropropagado pela rede modificando os pesos sinápticos para minimizar o erro
  - Depois do treinamento o processo de aprendizado se encerra e os pesos são armazenados
- Muitas vezes chamadas de redes *back propagation*, mas *o* termo *back propagation* se refere ao método de aprendizagem da rede

Redes *back propagation*

- Requer no mínimo 3 camadas
  - De entrada: recebe as entradas
  - Invisível ou intermediária: processam os dados
  - De saída: apresenta os resultados da rede
  - Os nós da camada de entrada se conectam somente com os nós da camada invisível
  - E os nós da camada invisível somente com os nós da camada de saída

Rede *back propagation* - Arquitetura



O algoritmo de aprendizado *Back Propagation*

- Duas fases:
  - *Feed-Forward* → as entradas se propagam pela rede, da camada de entrada até a camada de saída
  - *Feed-Backward* → os erros se propagam na direção contrária ao fluxo de dados, indo da camada de saída até a primeira camada escondida
- Repetido, até que, para todos os processadores da camada de saída e para todos os padrões de treinamento, o erro seja menor do que o especificado

O algoritmo de aprendizado

- Exige que a função de transferência seja contínua e diferenciável
- Deve ser também assintótica para valores infinitamente positivos e negativos
- É conhecida como regra Delta generalizada que utiliza mínimo erro médio quadrático
- Equações de erros operam sobre funções diferenciais
- Baseadas em uma heurística de gradiente descendente

Cada neurônio da camada evolui seu erro

$$E_{pj} = (T_{pj} - O_{pj})$$

Onde:

- E_{pj} – erro linear na saída *j* para o padrão *p*
- T_{pi} – saída desejada para a saída *j*, padrão *p*
- O_{pj} – saída *j* computada para o padrão *p*

- Isto é feito até que $O_{pj} \approx T_{pj}$

Como atualizar os pesos

- A qualidade da aproximação realizada pode ser medida pelo erro quadrático da saída
- O erro quadrático "instantâneo", para o padrão *p* neurônio *j* é dado por:

$$E_{pj} = (e_{pj})^2 = (T_{pj} - O_{pj})^2$$

- Este valor é utilizado para atualizar os pesos da rede para minimizar o erro
- Através do método do gradiente descendente, deslocando o vetor de sinapses em uma direção

Como atualizar os pesos

- A cada passo de treinamento, cada sinapse w_{ij} sofre uma atualização:

$$w_{ij,novo} = w_{ij,anterior} + \eta \Delta w_{ij}$$

$$\text{Onde } \eta > 0 \text{ é o passo do treinamento}$$

- O acréscimo correspondente em cada sinapse corresponde a:

$$\Delta w_{ij} = -\frac{\partial E_{pj}}{\partial w_{ij,anterior}}$$

Gradiente do valor esperado de erro

O processo de treinamento

- Durante o treinamento espera-se que o seu erro diminua de forma que os pesos w_{ij} alcancem melhores valores (mínimo global)
- Mas a rede pode entrar em um estado de paralisia causado pelo próprio processo de otimização de erros pelo gradiente
- w_{ij} é deslocado numa região de gradiente pequeno (mínimo local)
- E o treinamento praticamente pára onde não é desejável
- *Simulated annealing* pode ser usado para fugir deste problema

Monitoração do processo de aprendizagem

- É importante durante o treinamento para evitar o sobreaprendizado
  - Situação em que a rede perde sua capacidade de generalização (memoriza os pares de entrada/saída)

- Para reduzir o sobreaprendizado
  - Acompanhar o desempenho da rede sobre um conjunto separado de teste
  - Depois de passar por um conjunto de treino, o conjunto de teste é apresentado à rede e os resultados obtidos são avaliados, mas não propagados

Monitoração do processo de aprendizagem

- Um critério deve ser estabelecido como ponto de parada, antes que se alcance o sobreaprendizado
- A eficiência da rede para o conjunto de teste e treino vai aumentado devido ao ajuste dos pesos
- Até que esta eficiência atinge um platô – em torno do qual os pesos ficam oscilando
- A rede está entrando no sobreaprendizado, e o processo de aprendizagem deve parar

Fatores que influenciam o aprendizado

- O número de camadas invisíveis
- O número de neurônios nas camadas invisíveis
- O formato do conjunto de treino

Número de camadas invisíveis

- Camadas invisíveis são:
  - Detectores de características
  - Unidades internas de representação
- A rede acumula conhecimento nestas camadas abstraído do conjunto de treino
- Pode ser treinada para detectar a presença de características nos padrões de entrada
  - Capacidade para lidar com dados incompletos
- Para se descobrir as características que estão sendo representadas na camada invisível – análise de sensibilidade

Número de camadas invisíveis

- O treinamento é iniciado com apenas uma camada invisível, e verificando-se o seu desempenho
- Aumenta-se o número de camadas gradativamente, monitorando-se o desempenho da rede
- A capacidade de mapeamento de padrões complexos aumenta com o aumento das camadas
- Mais camadas invisíveis demandam maior tempo de treinamento
- Para grande parte das aplicações
  - Redes neurais *back propagation* com uma camada invisível é o suficiente

Formato do conjunto de treino

- Deve ter qualidade, refletir as informações desejadas
- Pode ser preparado seguindo uma dist. randômica
- Ou seguindo uma mesma proporcionalidade para se evitar o aprendizado preferencial
  - Preferência para os padrões mais freqüentes
- Deve ser completamente representativo das entradas utilizadas para a rede

Sistemas tradicionais X conexionistas

- Sistemas tradicionais
  - Problema de aquisição de conhecimento
- RN
  - Necessita de uma boa quantidade de exemplos
  - Não exige que se defina regras específicas
  - Desenvolve suas próprias regras
  - Útil quando não existem regras estabelecidas, ou elas são muito complexas
  - Podem extrapolar fatos e gerar conclusões

Sistemas tradicionais X conexionistas

- RN
  - Habilidade em trabalhar com dados incompletos ou com ruído (incerteza)
  - Requerem um número substancial de exemplos
  - Não é fácil obter uma explicação sobre os resultados obtidos
  - Conhecimento na forma de pesos sinápticos
  - Explicações sobre os resultados envolvem árdua tarefa de análise de sensibilidade