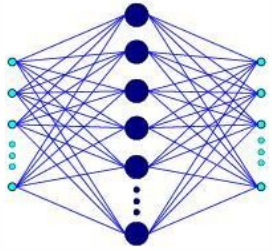
Ciência da Computação

REDE NEURAIS

Semestre: 2010/1

AULA 10



Max Pereira

http://paginas.unisul.br/max.pereira

UNISUL

Aqui seu futuro acontece

Conteúdo

- Redes Competitivas
- Redes Auto-organizáveis (Mapas de Kohonen)
- Exercícios

Treinamento

- Até agora estudamos redes com técnicas de treinamento **supervisionado**, onde há uma saída para cada padrão de entrada, e a rede aprende a produzir as saídas desejadas.
- O foco agora é o treinamento **não-supervisionado**, no qual as redes aprendem a formar a sua própria classificação dos dados de treinamento sem ajuda externa.
- Para que isso seja possível os padrões de entrada devem compartilhar características comuns, e a rede de ser capaz de identificar tais características.

Redes Competitivas

- Uma classe particular de sistemas não-supervisionados é baseada em aprendizado competitivo, no qual os neurônios de saída **competem** entre si para serem ativados.
- Um único neurônio é ativado a cada momento.
- O neurônio ativado é chamado de *winner-takes-all* ou simplesmente neurônio vencedor.
- Tal competição pode ser induzida/implementada através de conexões laterais inibitórias entre os neurônios.
- Como resultado os neurônios são forçados a uma auto-organização.
- Essas redes são conhecidas como mapas auto-organizáveis.

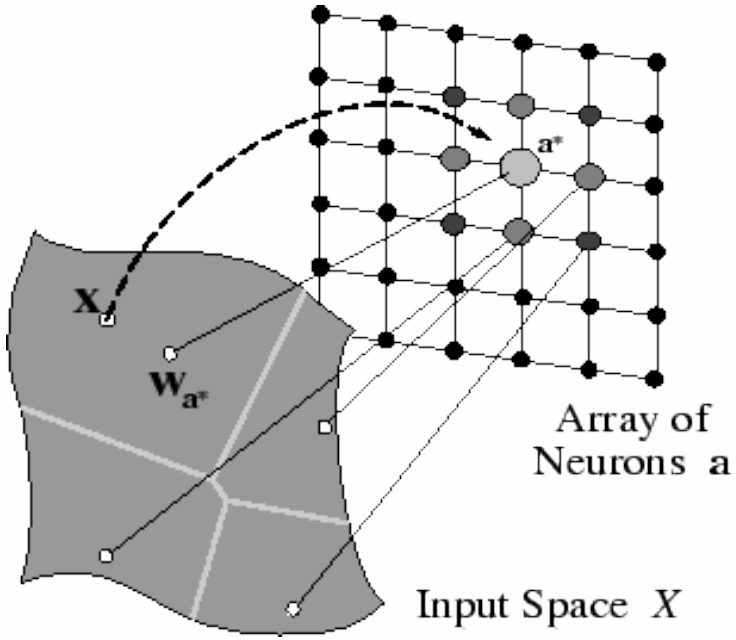
Mapas Topográficos

- Estudos neurobiológicos indicam que diferentes entradas sensoriais (motoras, visuais, auditivas, etc.) são mapeadas em áreas correspondentes no córtex cerebral.
- Essa forma de mapeamento tem duas propriedades importantes:
  - A cada estágio de processamento as partes da informação que são recebidas são mantidas em seu contexto/vizinhança.
  - Neurônios que lidam com partes da mesma informação são mantidos juntos para que se comuniquem via ligações sinápticas curtas.

Mapa Auto-Organizável

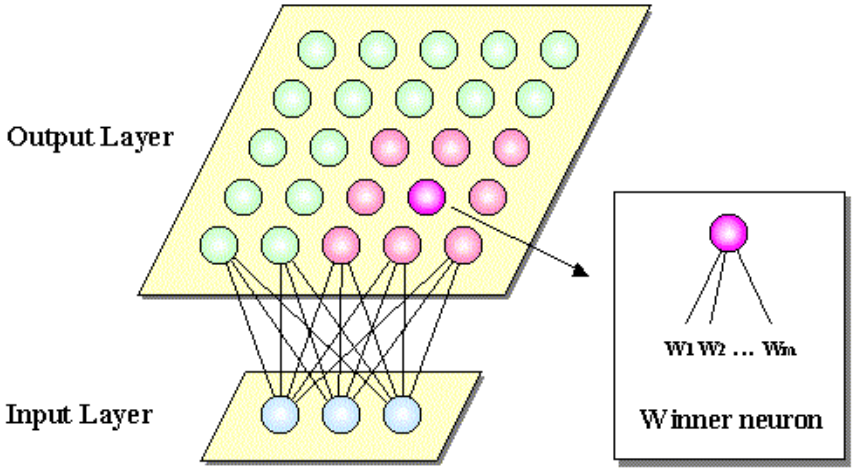
- O objetivo é construir mapas topográficos artificiais que aprendam através da auto-organização, inspirados no modelo neurobiológico.
- Deve-se seguir o *princípio da formação do mapa topográfico*:
  - "A localização espacial de um neurônio de saída, em um mapa topográfico, corresponde a um domínio particular ou característica do espaço de entrada"

Organização do Mapa



Temos pontos **x** no espaço de entrada mapeando para os pontos a^* no espaço de saída

Cada ponto a* no espaço de saída irá mapear para um ponto correspondente W_{a^*} no espaço de entrada



Mapas de Kohonen

- Um tipo particular de mapa auto-organizável. Essa rede tem uma estrutura *feed-forward* com apenas duas camadas.
- O vetor de pesos para um *cluster* serve como exemplar dos padrões de entrada associado com esse *cluster*.
- Padrões de entrada são comparados com todos os neurônios, e o mais próximo é considerado o **neurônio vencedor**.
- Os pesos do neurônio vencedor são atualizados para aproximar-se mais do padrão de dados que representa.

Algoritmo

- Passo 1. Inicializar Pesos W_{ij} (valores randômicos devem ser utilizados)
  - Setar os parâmetros da vizinhança (Raio)
  - Setar a taxa de aprendizado (α)
- Passo 2. Enquanto a condição de parada for falsa, faça

Passo 3. Para cada vetor de treinamento faça

Passo 4. Para cada j calcule:

$$D(j) = \sum_i (w_{ij} - x_i)^2$$ Distância Euclidiana

Passo 5. Encontre o índice *j* onde *D(j)* seja mínimo.

Passo 6. Para todas as unidades *j* em uma vizinhança especificada e para todos os *i* atualize:

$$w_{ij}(new) = w_{ij}(old) + \alpha[x_i - w_{ij}(old)]$$

Passo 7. Alterar taxa de aprendizado

Passo 8. Reduzir *raio*

Passo 9. Testar condição de Parada

Exemplo

- Vetores de entrada

  (1,1,0,0); (0,0,0,1); (1,0,0,0);(0,0,1,1)
- Número máximo de *clusters* = 2
- Taxa de aprendizado (com decréscimo geométrico):

  α=0.6

  descréscimo: $\alpha(t+1)=0.5\ \alpha(t)$

  onde *t* corresponde a época de treinamento.

Exemplo

- Com apenas 2 *clusters* disponíveis, a vizinhança de *j* corresponde a um único neurônio (R = 0).
- Matriz inicial de pesos

$$\begin{bmatrix} 0.2 & 0.8 \\ 0.6 & 0.4 \\ 0.5 & 0.7 \\ 0.9 & 0.3 \end{bmatrix}$$

- Raio inicial: R=0
- Taxa inicial de aprendizado $\alpha(0)=0.6$

Exemplo

- Início do treinamento

$D(1) = (.2 – 1)^2 + (.6 – 1)^2 + (.5 – 0)^2 + (.9 – 0)^2 = 1.86$

$D(2) = (.8 – 1)^2 + (.4 – 1)^2 + (.7 – 0)^2 + (.3 – 0)^2 = 0.98$

O vetor de entrada está mais perto do neurônio de saída 2, ou seja, $j = 2$.

Os pesos do neurônio vencedor são ajustados:

$$\mathbf{w_{i2}(new) = w_{i2}(old) + 0.6[x_i – w_{i2}(old)]}$$

Exemplo

- A matriz de pesos após o ajuste:

$$\begin{bmatrix} 0.2 & 0.92 \\ 0.6 & 0.76 \\ 0.5 & 0.28 \\ 0.9 & 0.12 \end{bmatrix}$$

- Para o segundo vetor (0,0,0,1):

$$D(1) = (.2 - 0)^2 + (.6 - 0)^2 + (.5 - 0)^2 + (.9 - 1)^2 = 0.66$$

$$D(2) = (.92 - 0)^2 + (.76 - 0)^2 + (.28 - 0)^2 + (.12 - 1)^2 = 2.2768$$

O vetor de entrada está mais perto do neurônio de saída 1, ou seja, $j = 1$.

Exemplo

- A matriz de pesos após o ajuste:

$$\begin{bmatrix} 0.08 & 0.92 \\ 0.24 & 0.76 \\ 0.20 & 0.28 \\ 0.96 & 0.12 \end{bmatrix}$$

- Para o terceiro vetor (1,0,0,0):

$$D(1) = (.08 - 1)^2 + (.24 - 0)^2 + (.2 - 0)^2 + (.96 - 0)^2 = 1.8656$$

$$D(2) = (.92 - 1)^2 + (.76 - 0)^2 + (.28 - 0)^2 + (.12 - 0)^2 = 0.6768$$

O vetor de entrada está mais perto do neurônio de saída 2, ou seja, $j = 2$.

Exemplo

- A matriz de pesos após o ajuste:

$$\begin{bmatrix} 0.08 & 0.968 \\ 0.24 & 0.304 \\ 0.20 & 0.112 \\ 0.96 & 0.048 \end{bmatrix}$$

- Para o quarto vetor (0,0,1,1):

$$D(1) = (.08 - 0)^2 + (.24 - 0)^2 + (.2 - 1)^2 + (.96 - 1)^2 = 0.7056$$

$$D(2) = (.968 - 0)^2 + (.304 - 0)^2 + (.112 - 1)^2 + (.048 - 1)^2 = 2.724$$

O vetor de entrada está mais perto do neurônio de saída 1, ou seja, $j = 1$.

Exemplo

- A matriz de pesos após o ajuste:

$$\begin{bmatrix} 0.032 & 0.968 \\ 0.096 & 0.304 \\ 0.680 & 0.112 \\ 0.984 & 0.048 \end{bmatrix}$$

- Redução da taxa de aprendizado:

  $\alpha = 0.5\ (0.6) = 0.3$

Condição de parada = número de iterações

Exercício

- Número de iterações 2
- Topologia: 2 entradas ligadas com 4 neurônios de saída
- Taxa de aprendizado $\alpha(t)$ será 0.9 durante todo o treinamento da rede.
- O neurônio terá apenas ele próprio como vizinho (R = 0).

Exercício

- Matriz inicial de pesos:

$$\begin{bmatrix} 2 & 3 & 2 & 1 \\ 2 & 2 & 1 & 3 \end{bmatrix}$$

- Entradas:

(0,3)

(7,2)

Trabalho

- Construir duas redes de Kohonen para agrupar as letras com fontes diferentes.
- Utilizar 25 *clusters*
- Taxa de aprendizado inicial α=0.6, com redução linear até 0.01.
- Na primeira rede não deve haver estrutura topológica, na segunda rede utilizar estrutura linear.
- Na estrutura linear (com R=1), o neurônio vencedor *j* e os seus vizinhos topológicos (*j* + 1 e *j* – 1) devem ser atualizados em cada iteração.
- Analisar os resultados e apresentar o número de *clusters* formados, para cada rede, com as respectivas letras.

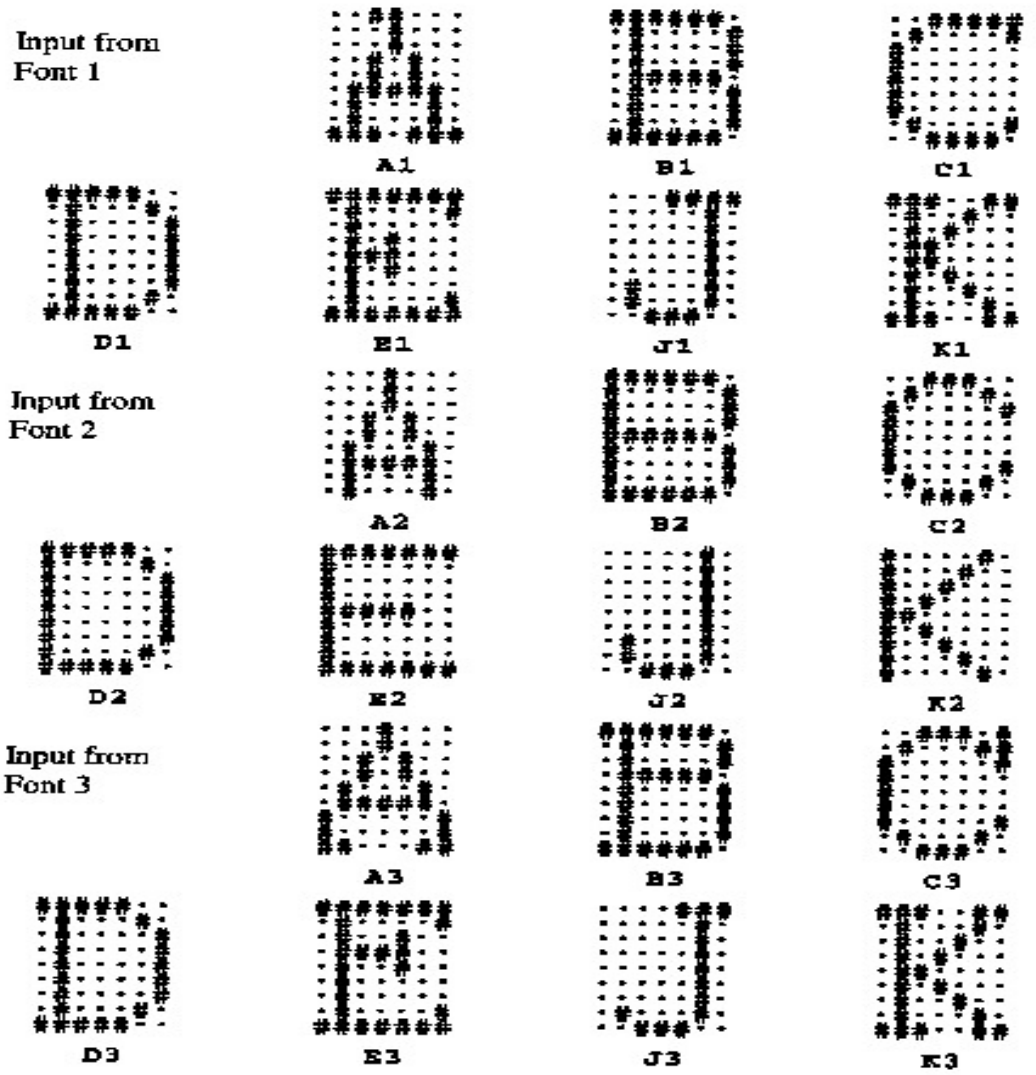


Figure 4.9 Training input patterns for character recognition examples.