2
Redes Neurais Auto-Organizáveis

2.1
Introdução

Problemas de *clustering* estão presentes nos mais variados contextos, como por exemplo: classificação de padrões, mineração de dados e recuperação de informações de documentos (por exemplo pode-se citar busca na *internet*). Nesse contexto, *clustering* é considerado como uma das principais etapas da análise de dados, denominada análise de *cluster*, e consiste em classificar os dados de forma não supervisionada. A análise de *cluster* envolve, portanto, a organização de um conjunto de padrões (usualmente representados na forma de vetores de atributos ou pontos em um espaço multidimensional – espaço de atributos) em *clusters* (agrupamentos), de acordo com alguma medida de similaridade. Intuitivamente, padrões pertencentes a um dado *cluster* devem ser mais similares entre si do que em relação a padrões pertencentes a outros *clusters*. Há diversas técnicas utilizadas para a construção destes conglomerados (*clusters*) as quais são classificadas como hierárquicas e não-hierárquicas. De acordo com Mingoti [66], as técnicas hierárquicas, na maioria das vezes, são utilizadas em análise exploratória de dados com o intuito de identificar possíveis agrupamentos e o valor provável do número de grupos g. Com relação às técnicas não hierárquicas, é necessário que o valor do número de grupos já esteja pré-especificado pelo especialista. Conceitos teóricos, algoritmos e exemplos dessas metodologias podem ser encontrados em [26, 49, 60, 66, 67].

2.2
Componentes de uma tarefa de *clustering*

Segundo Jain et al [50], atividades de *clustering* normalmente envolvem os seguintes passos:

1. Representação dos padrões (podendo incluir extração ou seleção de características);
2. Definição de uma medida de similaridade apropriada ao domínio da aplicação;
3. *Clustering* ou Agrupamento;
4. Apresentação dos resultados.

A Figura 2.1 mostra a seqüência típica dos três (3) primeiros passos registrados acima.

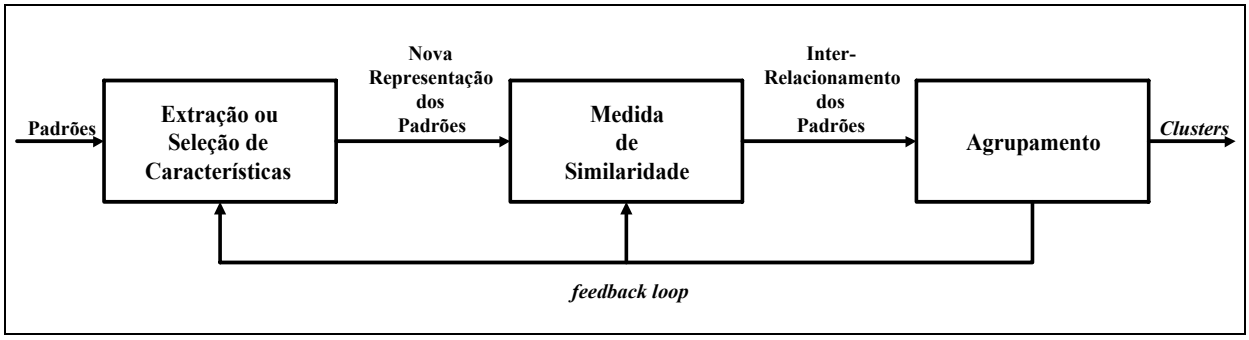

Figura 2.1: Etapas de um processo de *clustering*.

Em geral, pode-se dizer que:

- Representação dos padrões: envolve definição do número, tipo e modo de apresentação dos atributos que descrevem cada padrão;
- Seleção de características: processo de identificação do subconjunto mais efetivo dos atributos disponíveis para descrever cada padrão;
- Extração de características: uso de uma ou mais transformações junto aos atributos de entrada de modo a salientar uma ou mais característica dentre aquelas que estão presentes nos dados;
- Medida de Similaridade: fornecida por uma função de distância definida entre pares de dados ou padrões. É possível incluir na medida de distância, aspectos conceituais (qualitativos) ou então numéricos (quantitativos);
- Agrupamento: os grupos podem ser definidos como conjuntos *crisp* (um padrão pertence ou não-pertence a um determinado grupo) ou *fuzzy* (um padrão pode apresentar graus de pertinência aos grupos).

O processo de agrupamento pode ser hierárquico, com um processo recursivo de junções ou separações de grupos, ou não-hierárquico, com o emprego direto de técnicas de discriminação de *clusters*;

- A realimentação (*feedback loop*) do resultado do processo de *clustering* pode levar, se necessário, à redefinição dos módulos de "extração ou seleção de características" e de medida de similaridade.

2.3
Aprendizado por competição

As pesquisas na área de Redes Neurais Artificiais (RNA) têm passado por uma rápida expansão e desfrutado de grande popularidade tanto na comunidade de pesquisa industrial como acadêmica. Este sucesso é dedicado ao fato das RNA serem uma técnica não-linear bastante sofisticada capaz de modelar funções extremamente complexas. Com isso, resultados bastante expressivos estão sendo obtidos em uma gama muito grande de áreas, como exemplo, finanças, medicina, engenharia, geologia e física. De fato, qualquer área onde há problemas de predição, classificação e controle, RNA são apresentadas como uma ferramenta muito interessante [82].

É importante ressaltar que RNA são estruturas maciçamente paralelas inspiradas no conhecimento sobre os sistemas nervosos biológicos. Vale lembrar que alguns modelos de RNA são tão grosseiros, quando comparados a sistemas nervosos biológicos, que qualquer comparação a estes se torna improcedente [67].

De acordo com Haykin [49], as RNA são capazes de aprender a partir de seu ambiente e de melhorar o seu desempenho através da aprendizagem. A melhoria do desempenho ocorre com o tempo de acordo com alguma medida pré-estabelecida. Uma Rede Neural Artificial aprende acerca do seu ambiente através de um processo iterativo de ajustes aplicados a seus pesos sinápticos e níveis de bias. Idealmente, a Rede Neural Artificial se torna mais instruída sobre o seu ambiente após cada iteração do processo de aprendizagem.

No contexto de RNA, aprendizagem pode ser definida como:

"Aprendizagem é um processo pelo qual os parâmetros livres de uma Rede Neural Artificial são adaptados através de um processo de estimulação pelo

ambiente no qual a rede está inserida. O tipo de aprendizagem é determinado pela maneira pela qual a modificação dos parâmetros ocorre."

Um conjunto pré-estabelecido de regras bem-definidas para a solução de um problema de aprendizagem é denominado um algoritmo de aprendizagem.

Nascimento Jr. & Yoneyama [67] mencionam que os diversos métodos de aprendizado podem ser classificados de acordo com a participação do supervisor.

Este trabalho utiliza um tipo de aprendizagem cujo grau de supervisão é muito pequeno sendo, portanto, denominado de supervisão muito fraca. Neste caso, o algoritmo de treinamento da Rede Neural Artificial tenta descobrir categorias entre os dados de entrada e o supervisor participa apenas fornecendo os rótulos para estes agrupamentos. Na literatura, esse tipo de aprendizado é denominado aprendizado não supervisionado ou auto-organizado, um nome que, estritamente falando, não estaria totalmente apropriado, pois ainda ocorre uma pequena (mas efetiva) participação do supervisor no processo de aprendizagem [67]. Para a realização desta aprendizagem, é utilizado a regra de aprendizado competitivo.

Nesse processo de aprendizagem competitivo, como o nome indica, as unidades de saída de um mapeamento competem entre si para se tornar ativos (disparar). É esta característica que torna a aprendizagem competitiva muito adequada para descobrir características estatisticamente similares e, assim sendo, ter a capacidade de fazer classificações em um conjunto de padrões de entrada.

Há na literatura diversas medidas para determinar a similaridade entre entradas. Uma medida de similaridade usada freqüentemente é baseada no conceito de distância euclidiana.

2.4
Rede Neural de Kohonen (RNK)

Rede Neural de Kohonen utiliza aprendizado com supervisão muito fraca. Essa particularidade faz com que sua utilização seja bastante diferente em relação aos outros tipos de RNA em virtude de todos os outros tipos serem projetadas para executar tarefas mediante um processo de aprendizado supervisionado.

Esta Rede Neural é formada por grades de unidades uni ou bidimensionais (maiores dimensões são possíveis, mas pouco comuns), que modificam seus pesos

sinápticos em um processo de aprendizagem competitivo, formando sobre a grade de saída um sistema de coordenadas significativas para diferentes características de entrada (mapa topográfico dos padrões de entrada). A Figura 2.2 ilustra o caso de uma grade bidimensional, que é mais usual. Se duas unidades estão topologicamente próximas nesse espaço, elas são ditas unidades vizinhas [67]:

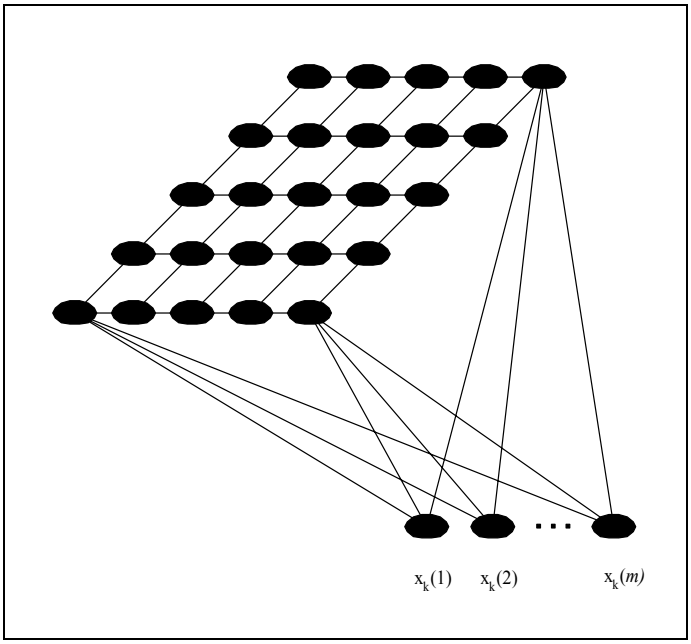

Figura 2.2: Rede de Kohonen com as unidades de saída organizadas como grade bi-dimensional.

O objetivo da Rede Neural de Kohonen é obter um mapeamento entrada-saída, onde a topologia do espaço de entrada seja preservada no espaço de saída. Assim, as entradas que são próximas no espaço de entrada serão mapeadas para saídas próximas no espaço de saída, ou seja, ativarão unidades de saída que estão fisicamente próximas.

Uma conseqüência interessante da Rede de Kohonen é a alocação das unidades de saída de acordo com a distribuição de probabilidade das entradas no espaço de entrada. Por exemplo, se não for usado durante o treinamento da Rede Neural nenhum ponto de uma determinada região do espaço de entrada, nenhuma unidade será modificada de forma a aumentar a similaridade de seu vetor de pesos com as entradas desta região. Analogamente, mais unidades de saída serão atraídas para as regiões do espaço de entrada com mais pontos de treinamento.

2.5
Exemplo ilustrativo da Rede Neural de Kohonen

Este tópico tem como objetivo elucidar a implementação de uma Rede Neural de Kohonen conforme registrado em Souza [82]. Para tal, seja uma rede com 9 unidades (neurônios), 20 padrões (vetores de treinamento). É importante assinalar que cada padrão possui 3 entradas (atributos).

A Tabela 2.1 fornece o centro de cada *cluster* com seus respectivos pontos de treinamento:

Tabela 2.1 - Centro dos *clusters* com seus respectivos pontos de treinamento.

Centro dos *Clusters*	Pontos de Treinamento
$\mathbf{C}_1 = [0 \quad 0 \quad 0]^{\mathbf{T}};$	$\boldsymbol{x}_1 = [0{,}1033 \quad -0{,}3161 \quad -1{,}1968]^{\mathbf{T}};$ $\boldsymbol{x}_2 = [-0{,}5450 \quad -0{,}4317 \quad 0{,}0667]^{\mathbf{T}};$ $\boldsymbol{x}_3 = [0{,}2247 \quad 0{,}5830 \quad -1{,}1134]^{\mathbf{T}};$ $\boldsymbol{x}_4 = [0{,}8058 \quad -0{,}2275 \quad -0{,}2512]^{\mathbf{T}};$ $\boldsymbol{x}_5 = [1{,}6383 \quad -0{,}8553 \quad 0{,}1821]^{\mathbf{T}};$
$\mathbf{C}_2 = [0 \quad 10 \quad 0]^{\mathbf{T}};$	$\boldsymbol{x}_6 = [-0{,}0510 \quad 8{,}2008 \quad 0{,}5728]^{\mathbf{T}};$ $\boldsymbol{x}_7 = [-0{,}8182 \quad 11{,}0073 \quad 0{,}9793]^{\mathbf{T}};$ $\boldsymbol{x}_8 = [-0{,}5798 \quad 10{,}1410 \quad 1{,}8104]^{\mathbf{T}};$ $\boldsymbol{x}_9 = [-1{,}6670 \quad 9{,}0699 \quad 0{,}4962]^{\mathbf{T}};$ $\boldsymbol{x}_{10} = [0{,}1776 \quad 10{,}9324 \quad -0{,}5245]^{\mathbf{T}};$
$\mathbf{C}_3 = [10 \quad 0 \quad 0]^{\mathbf{T}};$	$\boldsymbol{x}_{11} = [10{,}3432 \quad -0{,}6592 \quad 1{,}1919]^{\mathbf{T}};$ $\boldsymbol{x}_{12} = [9{,}1674 \quad -1{,}4384 \quad 1{,}0644]^{\mathbf{T}};$ $\boldsymbol{x}_{13} = [9{,}6810 \quad -0{,}8515 \quad 0{,}2207]^{\mathbf{T}};$ $\boldsymbol{x}_{14} = [9{,}1564 \quad -1{,}7950 \quad 0{,}7760]^{\mathbf{T}};$ $\boldsymbol{x}_{15} = [9{,}5935 \quad 0{,}2390 \quad 1{,}2727]^{\mathbf{T}};$
$\mathbf{C}_4 = [10 \quad 10 \quad 10]^{\mathbf{T}};$	$\boldsymbol{x}_{16} = [8{,}9785 \quad 9{,}2988 \quad 8{,}8607]^{\mathbf{T}};$ $\boldsymbol{x}_{17} = [10{,}8186 \quad 8{,}8467 \quad 9{,}8218]^{\mathbf{T}};$ $\boldsymbol{x}_{18} = [10{,}7659 \quad 9{,}9824 \quad 10{,}3016]^{\mathbf{T}};$ $\boldsymbol{x}_{19} = [8{,}7124 \quad 9{,}9586 \quad 10{,}3646]^{\mathbf{T}};$ $\boldsymbol{x}_{20} = [9{,}0056 \quad 8{,}6034 \quad 9{,}3408]^{\mathbf{T}};$

Continuando, tem-se a disposição das unidades em uma grade bi-dimensional:

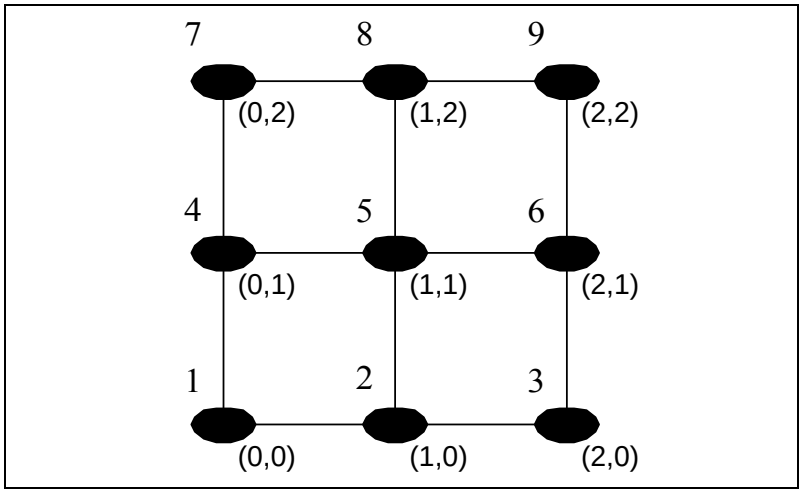

Figura 2.3: Disposição bi-dimensional das unidades.

Antes de mostrar a disposição da grade das unidades no espaço das entradas, é necessário atribuir, aleatoriamente, valores iniciais para os pesos da rede. Dessa forma, tem-se

$$\mathbf{W}_1^0 = [1 \quad 1 \quad 3]^{\mathbf{T}}, \quad \mathbf{W}_2^0 = [2 \quad 3 \quad 1]^{\mathbf{T}}, \quad \mathbf{W}_3^0 = [3 \quad 0 \quad 5]^{\mathbf{T}}, \quad \mathbf{W}_4^0 = [3 \quad 2 \quad 2]^{\mathbf{T}},$$

$$\mathbf{W}_5^0 = [1 \quad 4 \quad 3]^{\mathbf{T}} \quad \mathbf{W}_6^0 = [5 \quad 1 \quad 1]^{\mathbf{T}}, \quad \mathbf{W}_7^0 = [5 \quad 5 \quad 5]^{\mathbf{T}}, \quad \mathbf{W}_8^0 = [3 \quad 1 \quad 0]^{\mathbf{T}},$$

$$\mathbf{W}_9^0 = [4 \quad 1 \quad 3]^{\mathbf{T}}$$

A Figura 2.4 ilustra este processo.

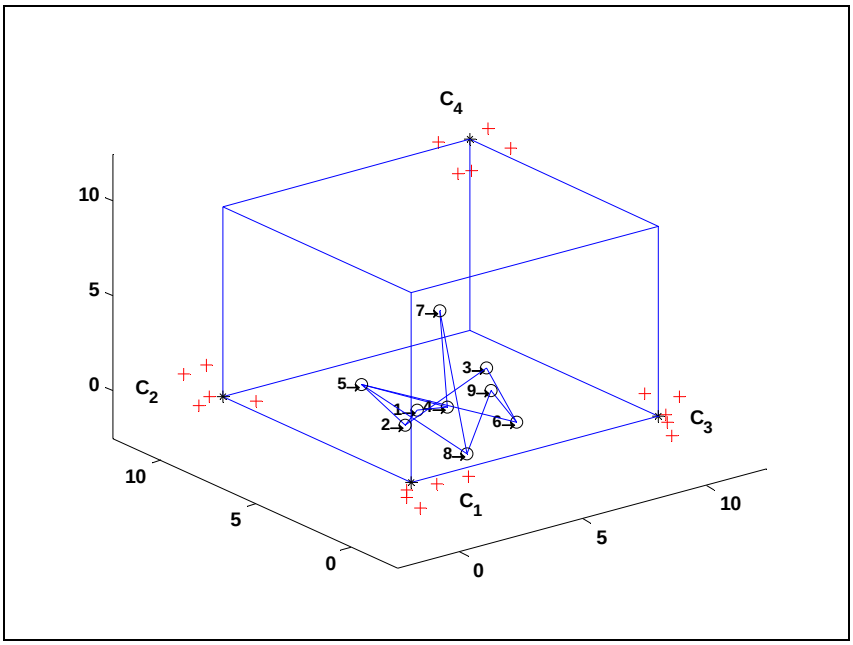

Figura 2.4: Disposição das unidades no espaço das entradas.

Neste ponto, é importante descrever como é feito o algoritmo de treinamento:

1. Inicie os pesos da rede $\mathbf{W}_i^0$, $(i = 1, \ldots, 9)$, Figura 2.4;

   Defina o raio inicial da vizinhança , $R_{vo} = 3\sqrt{2}$;

   Defina o número de iterações , Niter = 50;

   Defina a taxa de aprendizado inicial $\eta_0 = 0{,}5$;

   Defina o η mínimo durante o aprendizado, $\eta_{\min} = 0{,}05$;

   Defina $\eta = \eta_0$;

2. *Loop* em k (número de iterações) de 1 a 50;
3. Reordene, de forma aleatória, os 20 vetores de treinamento para esta iteração;
4. *Loop* em j (número do vetor de treinamento na seqüência aleatória) de 1 a 20;
5. Calcule a distância euclidiana entre $\boldsymbol{x}_j^k$ e todos os pesos da rede:

   $d\left(\boldsymbol{x}_j^k, \mathbf{W}_i^k\right) = \left\| \boldsymbol{x}_j^k - \mathbf{W}_i^k \right\|$ para j fixo e $i = 1, \ldots, 9$;

6. Determine o índice v da unidade vencedora, a unidade com a maior similaridade em relação ao vetor $\boldsymbol{x}_j^k$, ou seja: $\mathbf{W}_v^k = \min_i \left\{ \left\| \boldsymbol{x}_j^k - \mathbf{W}_i^k \right\| \right\}, i = 1,...,9$;
7. Atualize os pesos da unidade vencedora $\mathbf{W}_v^k$ e da sua vizinhança, usando o critério da distância, ou seja:
   loop em i ($i = 1,\ldots,9$);
   calcule $d_{iv} = \|\mathbf{p}_i - \mathbf{p}_v\|$
   se $d_{iv} < \mathrm{R_v}$ então faça: $\mathbf{W}_i^{k+1} = \mathbf{W}_i^k + \eta\left(\boldsymbol{x}_j^k - \mathbf{W}_i^k\right)$
   fim do *loop* em i;
8. Fim do *loop* em j. Retorne ao passo 4 até que todos os 20 pontos de treinamento sejam apresentados 1 única vez em seqüência aleatória;
9. Reduza o tamanho da vizinhança, $\mathrm{R_v} = \mathrm{R_{vo}}(\mathrm{Niter} - k + 1)/\mathrm{Niter}$;
10. Reduza a constante de aprendizado η, $\eta = \eta_{\mathbf{min}} + \eta_0 . (\mathrm{Niter} - k + 1)/\mathrm{Niter}$;
11. Fim do *loop* em k (*loop* de iterações)

No passo 7, $\mathbf{p}_i$ e $\mathbf{p}_v$ indicam, respectivamente, os vetores com a localização das unidades i e v no arranjo geométrico das unidades de saída. Como exemplo, sejam: $\mathbf{p}_1 = [0 \quad 0]^{\mathbf{T}}$, $\mathbf{p}_2 = [1 \quad 0]^{\mathbf{T}}$, $\mathbf{p}_5 = [1 \quad 1]^{\mathbf{T}}$, $\mathbf{p}_9 = [2 \quad 2]^{\mathbf{T}}$; assumindo a grade com distância horizontal e vertical unitária.

A identificação dos *clusters* ocorre no espaço das entradas. Em contrapartida, todo o processo referente ao critério de vizinhança como seus respectivos ajustes, se dão no espaço topológico das unidades verificado na Figura 2.3.

Após implementar o algoritmo em Matlab (www.mathworks.com), tem-se o seguinte resultado:

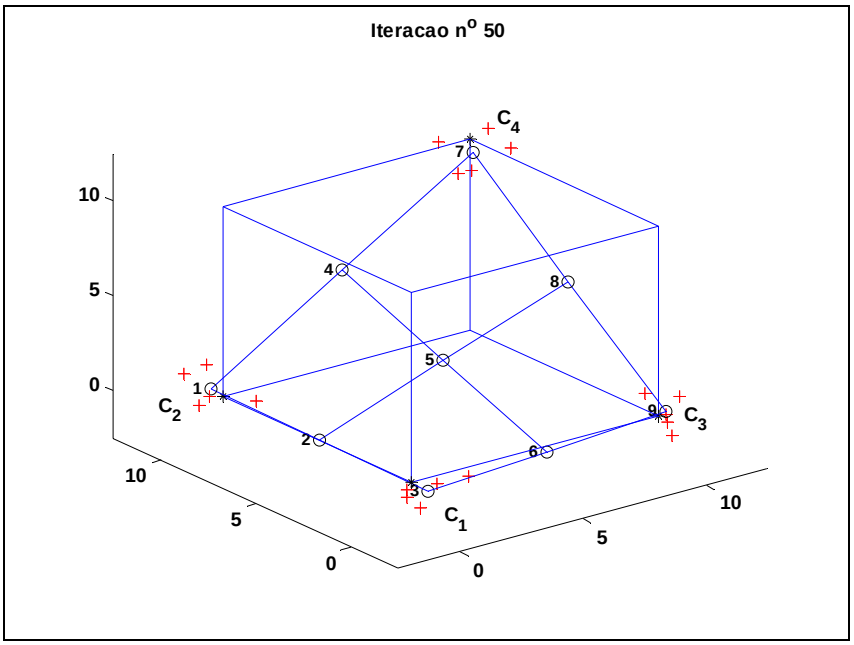

Figura 2.5: Resultado do processo de identificação dos *clusters*.

Através da Figura 2.5, pode-se perceber claramente que a rede identificou os 4 *clusters* registrados na Tabela 2.1:

- Unidade 1 ↔ *cluster* 2;
- Unidade 3 ↔ *cluster* 1;
- Unidade 7 ↔ *cluster* 4;
- Unidade 9 ↔ *cluster* 3 .

2.6
Rede Neural de Kohonen via simulação de Monte Carlo (MCRNK)

É pertinente enfatizar que os pesos sinápticos, da Rede Neural de Kohonen, muito influenciam nos resultados dos *clusters* encontrados. Afinal, como eles (pesos sinápticos) são inicializados com valores aleatórios diferentes, a cada simulação podem-se encontrar *clusters* bem distintos. Esta conjectura foi comprovada por Souza [82] que propôs fazer uma simulação de Monte Carlo utilizando-se dos resultados obtidos a partir da Rede Neural de Kohonen. É importante salientar que, procedendo desta forma, resultados estatisticamente

mais significativos são obtidos. Por esta razão, esta tese faz uso desta técnica a qual foi denominada de Rede Neural de Kohonen via simulação de Monte Carlo (MCRNK).