Pós-Graduação em Ciência da Computação

"SOM para dados relacionais baseados em múltiplas tabelas de dissimilaridade"

Por

Anderson Berg dos Santos Dantas

Dissertação de Mestrado

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE, 2012

Universidade Federal de Pernambuco
Centro de Informática
Pós-graduação em Ciência da Computação

Anderson Berg dos Santos Dantas

"SOM para dados relacionais baseados em múltiplas tabelas de dissimilaridade"

Trabalho apresentado ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientador: *Francisco de Assis Tenório de Carvalho*

RECIFE, 2012

Agradecimentos

...

Resumo

Abstract

Sumário

Lista de Figuras

Lista de Tabelas

1

Mapa auto-organizável

O mapa auto-organizável faz parte do grupo de modelos de aprendizado não-supervisionado de aprendizado competitivo. O objetivo destes modelos é encontrar uma estrutura lógica entre os dados fornecidos, não existe uma resposta esperada nem uma ação determinada que deva ser realizada. Mapas auto-organizáveis foram introduzidos por T. Kohonen em 1981. Os primeiros modelos foram projetados para tratar dados de grandes dimensões. Para realizar esse processamento, a metodologia de visualização topológica é projetada para particionar os dados em agrupamentos (*clusters*) que exibem alguma similaridade.

A característica mais importante dos mapas auto-organizáveis é a possibilidade de comparar agrupamentos. Cada observação é afetada a um agrupamento e cada agrupamento é projetado em um nó do mapa. Observações semelhantes são projetadas no mesmo nó. A dissimilaridade entre as observações projetadas aumenta com a distância que separa os nós.

Classificadores não-supervisionados e mapas auto-organizáveis são classes de métodos que buscam agrupar dados semelhantes. A maioria das aplicações que usam mapas auto-organizáveis são classificadores.

Kohonen projetou um algoritmo auto-organizável que projeta dados em grandes dimensionalidades em um espaço discreto de baixa dimensionalidade. Este espaço é composto de um grafo não-orientado que tem, geralmente 1, 2 ou 3 dimensões, este grafo é denominado mapa. O mapa é formado por neurônios interconectados, as conexões entre os neurônios são as arestas do grafo. A estrutura de grafo permite a definição de uma distância inteira δ no conjunto C de neurônios. Para cada par de neurônios (c,r) do mapa, $\delta(c,r)$ é o tamanho do caminho mais curto em C entre c e r. Para qualquer neurônio c, a distância permite a definição da vizinhança de c de ordem d,

$$V_c(d) = r \in C, \delta(c,r) \leq d. \tag{1.1}$$

2

SOM para dados relacionais baseados em múltiplas tabelas de dissimilaridade

2.1 Introdução

Neste capítulo introduzimos um mapa auto-organizável por lote para dados relacionais baseados em múltiplas matrizes de dissimilaridade. O objetivo do algoritmo apresentado é mapear objetos levando em conta suas descrições relacionais dadas por múltiplas matrizes de dissimilaridade. O modelo aqui proposto é um algoritmo iterativo composto por três etapas, são elas: representação, ponderação e afetação, que serão detalhadas posteriormente. É importante notar que por ser um algoritmo por lote, todo o conjunto de dados é apresentado ao mapa antes que quaisquer alterações sejam realizadas.

O algoritmo SOM por lote para dados relacionais baseados em múltiplas tabelas de dissimilaridade deriva do SOM original por lote e do algoritmo de agrupamento para dados relacionais (De Carvalho *et al.* (5016)). Esta abordagem utiliza diferentes pesos para as matrizes de dissimilaridade, com o objetivo de ponderar a relevância de cada matriz na formação dos agrupamentos. Os pesos mudam a cada iteração do algoritmo, portanto não são definidos absolutamente, além disso, são diferentes de um agrupamento para outro. O cálculo dos vetores de pesos neste algoritmo foi inspirado pela abordagem utilizada para calcular pesos para cada variável em cada agrupamento no algoritmo de agrupamento dinâmico baseado em distâncias adaptativas (REFERÊNCIA).

2.2 Dados relacionais

Diversos métodos de análise de dados se baseiam em dados que podem ser descritos por valores reais, ou seja, por vetores em um espaço dimensional fixo e finito. Entretanto, muitos dados do mundo real requerem estruturas mais complexas para que sejam representados adequadamente. Textos, por exemplo, não são numéricos e possuem uma estrutura interna complexa que é difícil de representar em um vetor.

Uma maneira de solucionar este problema é criar métodos que possam analisar dados a partir da comparação entre objetos. É possível definir uma medida de similaridade ou dissimilaridade entre os dados e compará-los. Quando cada par de objetos do conjunto de dados é representado por uma relação, então temos o que se chama de dados relacionais. Em geral, algoritmos baseados em dissimilaridades ou similaridades entre dados são mais complexos do que os usuais dados vetorizados, porém são universais e podem ser aplicados a qualquer tipo de dado.

O algoritmo apresentado neste trabalho é capaz de analisar dados levando em consideração a dissimilaridade entre observações. Os dados analisados são matrizes de dissimilaridade contendo a relação entre cada um dos objetos presentes na base de dados. Cada matriz representa uma variável da base e a dissimilaridade é calculada por uma função fixa que é a distância euclidiana entre os objetos. Para encontrar uma partição dos elementos, o método descrito leva em consideração simultaneamente a descrição relacional dos dados dada por múltiplas matrizes de dissimilaridade.

2.3 O algoritmo

Seja $E = \{e_1, \ldots, e_n\}$ o conjunto de n objetos e p matrizes de dissimilaridade $\mathbf{D}_j = [d_j(e_i, e_l)]\,(j = 1, \ldots, p)$, onde $d_j(e_i, e_l)$ denota a dissimilaridade entra dois objetos e_i e $e_l\,(i, l = 1, \ldots, n)$ na matriz de dissimilaridade $\mathbf{D}_j$.

Uma característica importante do modelo introduzido é que este assume que o protótipo G_k do agrupamento C_k é um subconjunto de cardinalidade fixa $1 \leq q << n$ do conjunto E (por questão de simplicidade, geralmente $q = 1$), isto é, $G_k \in E^{(q)} = \{A \subset E : |A| = q\}$.

O algoritmo busca minimizar uma função objetivo dada por:

$$J = \sum_{e_i \in E} \sum_{l=1}^{c} K^T(\delta(\chi(e_i), l)) D_{\lambda_l}(e_i, G_l) \tag{2.1}$$

onde D_{λ_l} é a dissimilaridade global entre um objeto $e_i \in P_l$ e o protótipo do agrupamento $G_l \in E^{(q)}$, parametrizada pelo vetor de pesos $\lambda_l = (\lambda_{l1}, \ldots, \lambda_{lp})$ das matrizes de dissimilaridade $\mathbf{D}_j$ no agrupamento $P_l\,(l = 1, \ldots, c)$.

De acordo com a função de alocação, existem diferentes algoritmos SOM por lote. Neste trabalho consideramos funções de alocação com pesos para cada matriz de dissimilaridade, sendo estes pesos estimados localmente ou globalmente.

Através da função de alocação com pesos estimados localmente, é possível comparar agrupamentos e seus protótipos usando diferentes medidas associadas a cada agrupamento que muda a cada iteração, isto é, a distância não é determinada absolutamente e é diferente de um agrupamento para outro.

A função de alocação parametrizada pelo vetor de pesos $\lambda_l = (\lambda_{l1}, \ldots, \lambda_{lp})$, onde $\lambda_{lj} > 0$ e $\prod_{j=1}^{p} \lambda_{lj} = 1$ e associada ao agrupamento $P_l\,(l = 1, \ldots, c)$ é definida pela seguinte expressão:

$$D_{\lambda_l}(e_i, G_l) = \sum_{j=1}^{p} \lambda_{lj} D_j(e_i, G_l) = \sum_{j=1}^{p} \lambda_{lj} \sum_{e \in G_l} d_j(e_i, e) \qquad (2.2)$$

onde $D_j(e_i, G_l) = \sum_{e \in G_l} d_j(e_i, e)$ representa a dissimilaridade local entre um exemplo $e_i \in P_l$ e o protótipo do agrupamento $G_l \in E^{(q)}$ na matriz $\mathbf{D}_j\,(j = 1, \ldots, p)$.

O princípio da função de alocação definida por um vetor de pesos estimado globalmente para todos os agrupamentos é que há uma distância para comparar os agrupamentos e seus protótipos, distância esta que muda a cada iteração, mas é a mesma para todos os agrupamentos.

A função de alocação parametrizada pelo vetor de pesos boldmathλ_l $=\lambda = (\lambda_1, \ldots, \lambda_p)\,(l = 1, \ldots, c)$, onde $\lambda_j > 0$ and $\prod_{j=1}^{p} \lambda_j = 1$, é expressa da seguinte forma:

$$D_{\lambda}(e_i, G_l) = \sum_{j=1}^{p} \lambda_j D_j(e_i, G_l) = \sum_{j=1}^{p} \lambda_j \sum_{e \in G_l} d_j(e_i, e) \qquad (2.3)$$

onde $D_j(e_i, G_l)$, novamente representa a dissimilaridade local entre um exemplo $e_i \in P_l$ e o protótipo do agrupamento $G_l \in E^{(q)}$ na matriz $\mathbf{D}_j\,(j = 1, \ldots, p)$.

Quando T é fixo, a minimização da função J é realizada iterativamente em três etapas: representação, ponderação e afetação.

2.3.1 Etapa de representação: determinação dos melhores protótipos

Na etapa de representação, a partição $P^{(t-1)} = (P_1^{(t-1)}, \ldots, P_c^{(t-1)})$ e os vetores de pesos ${\lambda_l}^{(t-1)}\,(r = 1, \ldots, c)$ são fixos. A função objetivo J é minimizada de acordo com os protótipos.

Proposição 2.3.1. *O protótipo é atualizado de acordo com a função de alocação usada:*

1. *Se a função de alocação é definida pela equação 2.2, calcule o protótipo* $G_l^{(t)} = G^* \in E^{(q)}$ *do agrupamento* $P_l^{(t-1)}$ $(l = 1, \ldots, c)$ *segundo a expressão:*

$$G^* = argmin_{G \in E^{(q)}} \sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{j=1}^{p} \lambda_{lj}^{(t-1)} \sum_{e \in G} d_j(e_i, e) \qquad (2.4)$$

2. *Se a função de alocação é definida pela equação 2.3, calcule o protótipo* $G_l^{(t)} = G^* \in E^{(q)}$ *do agrupamento* $P_l^{(t-1)}$ $(r = 1, \ldots, c)$ *de acordo com a equação seguinte:*

$$G^* = argmin_{G \in E^{(q)}} \sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{j=1}^{p} \lambda_{j}^{(t-1)} \sum_{e \in G} d_j(e_i, e) \qquad (2.5)$$

2.3.2 Etapa de ponderação: definição dos melhores vetores de pesos

Durante a etapa de ponderação, a partição $P^{(t-1)} = (P_1^{(t-1)}, \ldots, P_c^{(t-1)})$ e os protótipos $G_l^{(t)} \in E^{(q)}\,(l = 1, \ldots, c)$ são mantidos fixos. A função objetivo J é minimizada de acordo com os vetores de pesos.

Proposição 2.3.2. *Os vetores de pesos são atualizados de acordo com a função de alocação utilizada:*

1. *Se a função de alocação é definida pela equação 2.2, os vetores de pesos* ${\lambda_l}^{(t)} = (\lambda_{l1}^{(t)}, \ldots, \lambda_{lp}^{(t)})\,(l = 1, \ldots, c)$, *com* $\lambda_{lj}^{(t)} > 0$ *e* $\prod_{j=1}^{p} \lambda_{lj}^{(t)} = 1$, *têm seus pesos* $\lambda_{lj}^{(t)}\,(j = 1, \ldots, p)$ *calculados de acordo com a equação:*

$$\lambda_{lj}^{(t)} = \frac{\left\{ \prod_{h=1}^{p} \left[\sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{e \in G_l^{(t)}} d_h(e_i, e) \right] \right\}^{\frac{1}{p}}}{\left[\sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{e \in G_l^{(t)}} d_j(e_i, e) \right]} \qquad (2.6)$$

2. *Se a função de alocação é definida pela equação 2.3, o vetor de pesos* $\lambda^{(t)} = (\lambda_1^{(t)}, \ldots, \lambda_p^{(t)})$ *, com* $\lambda_j^{(t)} > 0$ *e* $\prod_{j=1}^{p} \lambda_j^{(t)} = 1$*, tem seus pesos* $\lambda_j^{(t)}\,(j = 1, \ldots, p)$ *calculados segundo a equação:*

$$\lambda_j^{(t)} = \frac{\left\{ \prod_{h=1}^{p} \left[\sum_{r=1}^{c} \left(\sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{e \in G_l^{(t)}} d_h(e_i, e) \right) \right] \right\}^{\frac{1}{p}}}{\sum_{r=1}^{c} \left[\sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{e \in G_l^{(t)}} d_j(e_i, e) \right]} \tag{2.7}$$

2.3.3 Etapa de afetação: definição da melhor partição

Durante a etapa de afetação os protótipos $G_l^{(t)} \in E^{(q)}\,(l = 1, \ldots, c)$ e os vetores de pesos ${\lambda_l}^{(t-1)}\,(r = 1, \ldots, c)$ são mantidos fixos. A função objetivo J é minimizada de acordo com a função de afetação.

Proposição 2.3.3. *Cada exemplo* $e_i \in E$ *é alocado ao neurônio mais próximo de acordo com a função de alocação utilizada:*

1. *Se a função de alocação é definida pela equação (2.2), alocar o exemplo* $e_i \in E$ *no agrupamento* C_m *segundo a equação:*

$$m = (\chi^{(t)}(e_i))^{(t)} = argmin_{1 \le r \le c} \sum_{l=1}^{c} K^T(\delta(r, l)) \sum_{j=1}^{p} \lambda_{lj}^{(t)} \sum_{e \in G_l^{(t)}} d_j(e_i, e) \tag{2.8}$$

2. *Se a função de alocação é definida pela equação (2.3), alocar o exemplo* $e_i \in E$ *no agrupamento* C_m *segundo a expressão:*

$$m = (\chi^{(t)}(e_i))^{(t)} = argmin_{1 \le r \le c} \sum_{l=1}^{c} K^T(\delta(r, l)) \sum_{j=1}^{p} \lambda_j^{(t)} \sum_{e \in G_l^{(t)}} d_j(e_i, e) \tag{2.9}$$

2.3.4 O algoritmo

O algoritmo SOM em lote para dados relacionais baseados em múltiplas matrizes de dissimilaridade pode ser resumido como segue:

1. Inicialização
   Fixe o número c de agrupamentos;

Fixe a cardinalidade $1 \leq q << n$ dos protótipos $G_l\,(l = 1,\ldots,c)$;
Fixe δ; Fixe a função kernel K
Fixe o número de iterações N_{iter}
Fixe T_{min}, T_{max}; Determine $T \leftarrow T_{max}$; Determine $t \leftarrow 0$;
Selecione aleatoriamente c protótipos distintos $G_l^{(0)} \in E^{(q)}\,(l = 1,\ldots,c)$;
Determine ${\lambda_l}^{(0)} = (1,\ldots,1)\,(l = 1,\ldots,c)$;
Determine o mapa $L(c,\mathbf{G}^0)$, onde $\mathbf{G}^0 = (G_1^{(0)},\ldots,G_c^{(0)})$
Aloque cada objeto e_i ao protótipo mais próximo para obter a partição $P^{(0)} = (P_1^{(0)},\ldots,P_c^{(0)})$ de acordo com as equações (2.8) e (2.9)

2. *Etapa de representação: cálculo dos melhores protótipos.*
Determine $t = t+1$;
Calcule $T = T_{max}(\frac{T_{min}}{T_{max}})^{\frac{t}{N_{iter}-1}}$
A partição $P^{(t-1)} = (P_1^{(t-1)},\ldots,P_c^{(t-1)})$ e ${\lambda_l}^{(t-1)}\,(l = 1,\ldots,c)$ são mantidos fixos.
Calcule o protótipo $G_l^{(t)} = G^* \in E^{(q)}$ do agrupamento $P_l^{(t-1)}$ $(l = 1,\ldots,c)$ de acordo com as equações (2.4) e (2.5)

3. *Etapa de ponderação: cálculo dos melhores pesos.*
A partição $P^{(t-1)} = (P_1^{(t-1)},\ldots,P_c^{(t-1)})$ e os protótipos $G_l^{(t)} \in E^{(q)}\,(l = 1,\ldots,c)$ são mantidos fixos.
Calcule os vetores de pesos λ_l $(l = 1,\ldots,c)$ de acordo com as equações (2.6) e (2.7)

4) *Etapa de afetação: definição da melhor partição.*
Os protótipos $G_l^{(t)} \in E^{(q)}\,(l = 1,\ldots,c)$ e ${\lambda_l}^{(t)}\,(l = 1,\ldots,c)$ são mantidos fixos
$P^{(t)} \leftarrow P^{(t-1)}$

para $i = 1$ até n faça
encontre o agrupamento $C_{m^*}^{(t)}$ ao qual e_i pertence
encontre o agrupamento vencedor $C_m^{(t)}$ segundo as equações (2.8) e (2.9)
se $m^* \neq m$
$C_m^{(t)} \leftarrow C_m^{(t)} \cup \{e_i\}$
$C_{m^*}^{(t)} \leftarrow C_m^{(t)} \setminus \{e_i\}$

4) *Critério de parada.*
Se $T = T_{min}$ (ou se $t = N_{iter} - 1$) então PARE; senão vá para 2 (Etapa de representação).

3 Resultados

3.1 Base de dados Íris

Tabela 3.1 Base de dados Íris: índices *CR*, $F-measure$, e *OERC*

Índices	T_{max}	B-SOM	AB-SOM	global AB-SOM
CR	6	0.4017	0.4847	0.4599
	7	0.3999	0.5067	0.5157
	9	0.3979	0.3978	0.4913
	16	0.3958	0.4653	0.3889
$F-measure$	6	0.4931	0.5785	0.5501
	7	0.5229	0.6071	0.5575
	9	0.5394	0.5383	0.5490
	16	0.5340	0.5742	0.5250
OERC	6	2.67%	4.00%	4.67%
	7	2.67%	4.67%	4.00%
	9	4.67%	3.33%	5.49%
	16	2.00%	4.67%	3.33%

Tabela 3.2 Base Íris: Matrizes relevantes ($T_{max} = 16$)

Modelo	Matriz mais importante	Matriz menos importante
AB-SOM	3-Petal length	1-Sepal length
global AB-SOM	3-Petal length	2-Sepal width

Tabela 3.3 Base Íris: matriz de confusão do algoritmo AB-SOM para mútiplas tabelas de dissimilaridade ($T_{max} = 6$)

	Classes		
Agrupamentos	1-Iris setosa	2-Iris versicolour	3-Iris virginica
0,0	0	16	0
0,1	0	12	0
0,2	0	1	10
0,3	0	4	0
0,4	0	0	12
0,5	0	0	3
0,6	0	0	0
0,7	38	0	0
1,0	0	2	10
1,1	0	9	0
1,2	0	3	0
1,3	0	0	6
1,4	0	0	4
1,5	0	3	3
1,6	0	0	2
1,7	12	0	0

3.2 Base de dados E.coli

3.3 Base de dados Thyroid

3.4 Base de dados Wine

Tabela 3.4 Base Íris: Matriz de pesos final do algoritmo AB-SOM para múltiplas tabelas de dissimilaridade ($T_{max} = 6$)

	Matriz			
Agrupamentos	1-Sepal length	2-Sepal width	3-Petal length	4-Petal width
0,0	0.5878	0.5075	**2.1617**	**1.5507**
0,1	0.1438	0.7925	**1.5381**	**5.7034**
0,2	0.1193	**47.9332**	0.5631	0.3104
0,3	0.7568	0.9139	**1.7698**	0.8169
0,4	0.3581	0.3508	**3.3944**	**2.3451**
0,5	**1.7980**	0.2266	**2.7677**	0.8866
0,6	**1.0037**	0.3404	**1.6919**	**1.7297**
0,7	0.2591	0.0672	**5.2699**	**10.8944**
1,0	0.4403	0.3797	**4.1288**	**1.4484**
1,1	0.2676	0.2180	**4.7063**	**3.6428**
1,2	0.1780	**2.1141**	**1.1214**	**2.3694**
1,3	**1.7573**	0.0686	**3.2489**	**2.5531**
1,4	0.3215	0.1939	**4.4941**	**3.5684**
1,5	**1.4477**	0.1896	**1.0308**	**3.5350**
1,6	**2.0289**	0.7747	0.6483	0.9812
1,7	0.4077	0.1109	**5.3766**	**4.1132**

Tabela 3.5 Base E.coli: índices *CR*, $F-measure$, e *OERC*

Índices	T_{max}	B-SOM	AB-SOM	global AB-SOM
CR	4	0.3035	0.3213	0.2739
	4.5	0.3462	0.2899	0.3165
	6	0.3264	0.2867	0.3555
	10	0.3303	0.2998	0.2925
$F-measure$	4	0.4473	0.4629	0.4082
	4.5	0.5486	0.4153	0.4680
	6	0.4631	0.4036	0.5265
	10	0.5187	0.4385	0.4460
OERC	4	21.43%	17.56%	24.40%
	4.5	16.37%	25.30%	20.54%
	6	16.37%	24.11%	14.88%
	10	15.77%	24.70%	24.40%

Tabela 3.6 Base E.coli: Matrizes relevantes ($T_{max} = 4$)

Modelo	Matriz mais importante	Matriz menos importante
AB-SOM	5	3
global AB-SOM	4	3

Tabela 3.7 Base E.coli: matriz de confusão do algoritmo AB-SOM global para múltiplas tabelas de dissimilaridade ($T_{max} = 10$)

	Classes							
Agrupamentos	1	2	3	4	5	6	7	8
0,0	0	18	0	0	16	1	0	0
0,1	0	10	0	1	24	0	0	0
0,2	0	6	1	0	17	1	0	0
0,3	6	0	0	0	1	4	0	0
0,4	22	0	0	0	2	0	0	0
1,0	1	1	0	1	6	5	5	2
1,1	0	0	0	0	10	0	0	0
1,2	10	0	0	0	0	1	0	0
1,3	15	0	0	0	0	1	0	0
1,4	22	0	0	0	0	0	0	0
2,0	2	0	1	0	0	29	0	9
2,1	0	0	0	0	0	10	0	9
2,2	6	0	0	0	1	0	0	0
2,3	25	0	0	0	0	0	0	0
2,4	34	0	0	0	0	0	0	0

Tabela 3.8 Base E.coli: Matriz de pesos final do algoritmo AB-SOM global para múltiplas tabelas de dissimilaridade ($T_{max} = 10$)

Matriz				
1	2	3	4	5
0.7636	0.7101	0.4876	**2.1144**	**1.7887**

Tabela 3.9 Base de dados Thyroid: índices *CR*, $F-measure$ e *OERC*

Índices	T_{max}	B-SOM	AB-SOM	AB-SOM global
CR	4	0.3662	0.4539	0.5700
	5	0.3642	0.3689	0.3604
	6	0.2618	0.3183	0.5882
	10	0.3784	0.3660	0.3654
$F-$ *measure*	4	0.4961	0.5841	0.5978
	5	0.4870	0.4788	0.4678
	6	0.4019	0.4850	0.6401
	10	0.5023	0.5048	0.4885
OERC	4	9.77%	3.25%	4.65%
	5	9.30%	5.58%	4.19%
	6	11.16%	4.19%	7.44%
	10	9.30%	3.72%	5.12%

Tabela 3.10 Base de dados Thyroid: Matrizes ($T_{max} = 4$)

Modelo	Matriz mais importante	Matriz menos importante
AB-SOM	4-TSH	1-T3-resin uptake test
global AB-SOM	5-maximal absolute difference in TSH	1-T3-resin uptake test

Tabela 3.11 Base de dados Wine: índices *CR*, $F-measure$ e *OERC*

Índices	T_{max}	B-SOM	AB-SOM	global AB-SOM
CR	4	0.2775	0.3560	0.3498
	5	0.2857	0.3449	0.3248
	6	0.2967	0.3589	0.3489
	10	0.2940	0.3773	0.3339
$F-$ *measure*	4	0.4163	0.4897	0.4737
	5	0.4286	0.4735	0.4357
	6	0.4343	0.5245	0.4801
	10	0.4427	0.5332	0.4637
OERC	4	26.40%	7.30%	2.25%
	5	26.97%	3.37%	4.49%
	6	26.97%	4.49%	5.06%
	10	26.97%	6.17%	6.74%

Tabela 3.12 Base de dados Wine: Matrizes relevantes $T_{max} = 4$

Modelo	Matriz mais importante	Matriz menos importante
AB-SOM	2	3
global AB-SOM	7	3

Referências Bibliográficas

De Carvalho, F., Lechevallier, Y., and de Melo, F. (2011, doi: 10.1016/j.patcog.2011.05.016). Partitioning hard clustering algorithms based on multiple dissimilarity matrices. *Pattern Recognition*.