



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 Univer-
sidade 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	viii
Lista de Tabelas	ix
1 Introdução	1
2 SOM para dados relacionais baseados em múltiplas tabelas de dissimilaridade	3
2.1 Introdução	3
2.2 Dados relacionais	3
2.3 Múltiplas matrizes de dissimilaridade	4
2.4 O algoritmo	5
2.4.1 Etapa de representação: determinação dos melhores protótipos .	6
2.4.2 Etapa de ponderação: definição dos melhores vetores de pesos .	7
2.4.3 Etapa de afetação: definição da melhor partição	8
2.4.4 O algoritmo	8
3 Resultados	10
3.1 Base de dados Íris	10
3.2 Base de dados E.coli	11
3.3 Base de dados Thyroid	11
3.4 Base de dados Wine	11
Bibliography	16

Lista de Figuras

Lista de Tabelas

3.1	Base de dados Íris: índices CR , $F - measure$, e $OERC$	10
3.2	Base Íris: Matrizes relevantes ($T_{max} = 16$)	10
3.3	Base Íris: matriz de confusão do algoritmo AB-SOM para múltiplas tabelas de dissimilaridade ($T_{max} = 6$)	11
3.4	Base Íris: Matriz de pesos final do algoritmo AB-SOM para múltiplas tabelas de dissimilaridade ($T_{max} = 6$)	12
3.5	Base E.coli: índices CR , $F - measure$, e $OERC$	12
3.6	Base E.coli: Matrizes relevantes ($T_{max} = 4$)	13
3.7	Base E.coli: matriz de confusão do algoritmo AB-SOM global para múltiplas tabelas de dissimilaridade ($T_{max} = 10$)	13
3.8	Base E.coli: Matriz de pesos final do algoritmo AB-SOM global para múltiplas tabelas de dissimilaridade ($T_{max} = 10$)	13
3.9	Base de dados Thyroid: índices CR , $F - measure$ e $OERC$	14
3.10	Base de dados Thyroid: Matrizes ($T_{max} = 4$)	14
3.11	Base de dados Wine: índices CR , $F - measure$ e $OERC$	14
3.12	Base de dados Wine: Matrizes relevantes $T_{max} = 4$	15

1

Introdução

Métodos de agrupamento organizam um conjunto de itens em grupos de forma que itens localizados em um mesmo grupo possuem alto grau de similaridade, por outro lado, itens de grupos distintos têm alto grau de dissimilaridade. Estes modelos têm sido amplamente aplicados em campos como taxonomia, processamento de imagem, recuperação de informação e mineração de dados. As técnicas mais populares de agrupamento são os métodos hierárquicos e de particionamento ([Qiang Wang and Huang, 2008](#); [Xu and Wunsch, 2005](#)).

Métodos hierárquicos fornecem uma hierarquia completa, isto é, uma sequência de partições aninhadas a partir dos dados de entrada. Métodos hierárquicos podem ser aglomerativos ([Sneath and Sokal, 1973](#); [Zhang *et al.*, 1996](#); [Guha *et al.*, 1998](#); [Karypis *et al.*, 1999](#); [Guha *et al.*, 2000](#)) ou divisivos ([Lance and Williams, 1968](#); [Gowda and Krishna, 1978](#); [Kaufman and Rousseeuw, 1990](#); [Guenoche *et al.*, 1991](#); [Chavent, 1998](#)). Métodos aglomerativos fornecem uma sequência de partições aninhadas iniciando com agrupamento trivial onde cada item está em um único grupo e termina com um agrupamento onde todos os itens estão no mesmo grupo. Um método divisivo inicia com todos os itens no mesmo grupo e executa um procedimento de divisão até que um critério de parada seja alcançado.

Métodos de particionamento buscam obter uma única partição dos dados em um número fixo de agrupamentos. Estes métodos frequentemente buscam uma partição que otimize (geralmente localmente) uma função objetivo. Para melhorar a qualidade do agrupamento, o algoritmo é executado diversas vezes com diferentes inicializações e a melhor configuração obtida do total de execuções é usada como saída do algoritmo de agrupamento. Métodos de particionamento podem ser divididos em agrupamentos HARD ([Forgy, 1965](#); [Huang, 1998](#); [Kanungo *et al.*, 2000](#); [Hansen and Mladenoviae, 2001](#); [Su and Chou, 2001](#)) e agrupamentos fuzzy ([Bezdek, 1981](#); [Hoeppner *et al.*, 1999](#); [Hathaway](#)

et al., 2000; Hung and Yang, 2001; Kolen and Hutcheson, 2002). Agrupamentos HARD fornecem uma partição na qual cada objeto do conjunto de dados é atribuído a um e somente um grupo. Agrupamento fuzzy gera uma partição fuzzy que fornece um grau de atribuição a cada padrão em um dado grupo, que dá flexibilidade para expressar que este objeto pertence a mais de um grupo ao mesmo tempo.

O objetivo dos mapas auto-organizáveis é encontrar uma estrutura nos dados fornecidos.

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.* \(2011\)](#)). 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.

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.

Existem basicamente dois tipos de dados em que se baseiam os algoritmos de agrupamento. O mais comum é que se tenha uma matriz contendo valores de medições, onde as linhas da matriz correspondem aos objetos e as colunas às variáveis. Alternativamente, outra maneira de representar os dados é através de uma matriz de dissimilaridade entre os objetos. Tais dissimilaridades podem ser obtidas de diversas formas. Esses tipos de dados são chamados dados relacionais. Agrupamento de dados relacionais é mais utilizado em situações onde os dados não podem ser descritos por características numéricas, também é mais prático quando a distância possui alto grau de complexidade computacional ou quando grupos de objetos similares não podem ser representados eficientemente por um único protótipo.

Muitos algoritmos foram adaptados para analisar dados relacionais. [Kaufman and Rousseeuw \(1987\)](#) apresenta uma adaptação do k-means para dados relacionais. Ainda, [Golli *et al.* \(2004\)](#) apresenta um modelo de mapa auto-organizável por lote baseado em dados relacionais. 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 Múltiplas matrizes de dissimilaridade

Em diversas situações, dados relacionais são descritos por múltiplas tabelas de dissimilaridade. Como apontado por [Frigui *et al.* \(2007\)](#) muitas aplicações podem se beneficiar de algoritmos de agrupamento baseados em múltiplas matrizes de dissimilaridade. Na categorização de imagens, pode-se ter uma matriz com informações de cor, outra matriz com informação de textura e outra com informação de estrutura.

Porém, diferentes matrizes não são igualmente importantes, algumas podem ser redundantes, outras irrelevantes, ou ainda, podem interferir negativamente na formação dos agrupamentos. Para que haja uma formação adequada dos agrupamentos faz-se necessário o uso de pesos para cada matriz de dissimilaridade, pesos estes que dependem de cada agrupamento. O objetivo da ponderação sobre as matrizes é encontrar graus

de relevância e identificar quais características descrevem melhor os dados, tornando o agrupamento mais significativo.

A ponderação de características deriva da seleção de características e tem sido um tópico de pesquisa importante em algoritmo de aprendizado não-supervisionado. Em (?), os autores introduzem um algoritmo fuzzy k-means que tem a vantagem de trabalhar com ponderação para objetos e variáveis simultaneamente. [Grozavu et al. \(2009\)](#) desenvolveu dois modelos usando mapas auto-organizáveis (SOM), que realizam simultaneamente agrupamento e ponderação de variáveis. [Frigui et al. \(2007\)](#) propõe um algoritmo de agrupamento baseados em múltiplas tabelas de dissimilaridade (CARD) que calcula pesos relacionados à relevância de cada matriz de dissimilaridade sobre cada agrupamento. Em outro trabalho, [Frigui and Nasraoui \(2004\)](#) apresenta uma abordagem que realiza agrupamento e ponderação de variáveis simultaneamente.

O modelo aqui apresentado utiliza diferentes pesos adaptativos para cada matriz de dissimilaridades. Esses pesos mudam a cada iteração do algoritmo e, além disso, são diferentes de um agrupamento para outro, ou seja, cada matriz possui uma influência diferente sobre a formação de cada agrupamento. 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 ([Diday and Govaert \(1977\)](#)).

2.4 O algoritmo

Seja $E = \{e_1, \dots, e_n\}$ o conjunto de n objetos e p matrizes de dissimilaridade $\mathbf{D}_j = [d_j(e_i, e_l)]$ ($j = 1, \dots, p$), onde $d_j(e_i, e_l)$ denota a dissimilaridade entre dois objetos e_i e e_l ($i, l = 1, \dots, n$) 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 $1 \leq q \ll n$ 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:

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

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 $\lambda_l = (\lambda_{l1}, \dots, \lambda_{lp})$ das matrizes de

dissimilaridade \mathbf{D}_j no agrupamento P_l ($l = 1, \dots, c$).

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 $\lambda_l = (\lambda_{l1}, \dots, \lambda_{lp})$, onde $\lambda_{lj} > 0$ e $\prod_{j=1}^p \lambda_{lj} = 1$ e associada ao agrupamento P_l ($l = 1, \dots, c$) é definida pela seguinte expressão:

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

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 ($j = 1, \dots, p$).

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 $\lambda_l = \lambda = (\lambda_1, \dots, \lambda_p)$ ($l = 1, \dots, c$), onde $\lambda_j > 0$ and $\prod_{j=1}^p \lambda_j = 1$, é expressa da seguinte forma:

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

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 ($j = 1, \dots, p$).

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

2.4.1 Etapa de representação: determinação dos melhores protótipos

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

protótipos.

Proposição 2.4.1. *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 $P_l^{(t-1)}$ ($l = 1, \dots, c$) segundo a expressão:

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

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

$$G^* = \operatorname{argmin}_{G \in E^{(q)}} \sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{j=1}^p \lambda_j^{(t-1)} \sum_{e \in G} d_j(e_i, e) \quad (2.5)$$

2.4.2 Etapa de ponderação: definição dos melhores vetores de pesos

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

Proposição 2.4.2. *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 $\lambda_l^{(t)} = (\lambda_{l1}^{(t)}, \dots, \lambda_{lp}^{(t)})$ ($l = 1, \dots, c$), com $\lambda_{lj}^{(t)} > 0$ e $\prod_{j=1}^p \lambda_{lj}^{(t)} = 1$, têm seus pesos $\lambda_{lj}^{(t)}$ ($j = 1, \dots, p$) calculados de acordo com a equação:

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

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

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

2.4.3 Etapa de afetação: definição da melhor partição

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

Proposição 2.4.3. *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:

$$m = (\chi^{(t)}(e_i))^{(t)} = \operatorname{argmin}_{1 \leq r \leq c} \sum_{l=1}^c K^T(\delta(r, l)) \sum_{j=1}^p \lambda_{lj}^{(t)} \sum_{e \in G_l^{(t)}} d_j(e_i, e) \quad (2.8)$$

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:

$$m = (\chi^{(t)}(e_i))^{(t)} = \operatorname{argmin}_{1 \leq r \leq c} \sum_{l=1}^c K^T(\delta(r, l)) \sum_{j=1}^p \lambda_j^{(t)} \sum_{e \in G_l^{(t)}} d_j(e_i, e) \quad (2.9)$$

2.4.4 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 $1 \leq q \ll n$ dos protótipos G_l ($l = 1, \dots, c$);

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 $G_l^{(0)} \in E^{(q)}$ ($l = 1, \dots, c$);

Determine $\lambda_l^{(0)} = (1, \dots, 1)$ ($l = 1, \dots, c$);

Determine o mapa $L(c, \mathbf{G}^0)$, onde $\mathbf{G}^0 = (G_1^{(0)}, \dots, G_c^{(0)})$

Aloque cada objeto e_i ao protótipo mais próximo para obter a partição $P^{(0)} = (P_1^{(0)}, \dots, P_c^{(0)})$ de acordo com as equações (2.8) e (2.9)

2. *Etapa de representação: cálculo dos melhores protótipos.*

Determine $t = t + 1$;

Calcule $T = T_{max} \left(\frac{T_{min}}{T_{max}} \right)^{\frac{t}{N_{iter}-1}}$

A partição $P^{(t-1)} = (P_1^{(t-1)}, \dots, P_c^{(t-1)})$ e $\lambda_l^{(t-1)}$ ($l = 1, \dots, c$) são mantidos fixos.

Calcule o protótipo $G_l^{(t)} = G^* \in E^{(q)}$ do agrupamento $P_l^{(t-1)}$ ($l = 1, \dots, c$) de acordo com as equações (2.4) e (2.5)

3. *Etapa de ponderação: cálculo dos melhores pesos.*

A partição $P^{(t-1)} = (P_1^{(t-1)}, \dots, P_c^{(t-1)})$ e os protótipos $G_l^{(t)} \in E^{(q)}$ ($l = 1, \dots, c$) são mantidos fixos.

Calcule os vetores de pesos λ_l ($l = 1, \dots, c$) de acordo com as equações (2.6) e (2.7)

4) *Etapa de afetação: definição da melhor partição.*

Os protótipos $G_l^{(t)} \in E^{(q)}$ ($l = 1, \dots, c$) e $\lambda_l^{(t)}$ ($l = 1, \dots, c$) 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 múltiplas tabelas de dissimilaridade ($T_{max} = 6$)

Agrupamentos	Classes		
	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 ($T_{max} = 6$)

Agrupamentos	Matriz			
	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 ($T_{max} = 10$)

Agrupamentos	Classes							
	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

- Bezdek, J. C. (1981). Pattern recognition with fuzzy objective function algorithms. *Plenum Press, New York*.
- Chavent, M. (1998). A monotetic clustering method. *Pattern Recognition Letters*, **19**, 989–996.
- De Carvalho, F., Lechevallier, Y., and de Melo, F. (2011). Partitioning hard clustering algorithms based on multiple dissimilarity matrices. *Pattern Recognition*.
- Diday, E. and Govaert, G. (1977). Classification automatique avec distances adaptatives. *R.A.I.R.O. Informatique Comput. Sci.*, **11**, 329–349.
- Forgy, E. (1965). Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. *Biometrics*, **21**, 768–780.
- Frigui, H. and Nasraoui, O. (2004). Unsupervised learning of prototypes and attribute weights. *Pattern Recognition*, **37**, 567–581.
- Frigui, H., Hwanga, C., and Rhee, F. C.-H. (2007). Clustering and aggregation of relational data with applications to image database categorization. *Pattern Recognition*, (40 (11)), 3053–3068.
- Golli, A., Conan-Guez, B., and Rossi, F. (2004). A self-organizing map for dissimilarity data. In *Classification, Clustering, and Data Mining Applications*, pages 61–68. Springer Berlin Heidelberg.
- Gowda, K. and Krishna, G. (1978). Disaggregative clustering using the concept of mutual nearest neighborhood. *IEEE Transactions on Systems, Man, and Cybernetics*, **8**, 888–895.
- Grozavu, N., Bennani, Y., and Lebbah, M. (2009). From variable weighting to cluster characterization in topographic unsupervised learning. *Proceedings of the International Joint Conference on Neural Networks*, pages 1005–1010.
- Guenoche, A., Hansen, P., and Jaumard, B. (1991). Efficient algorithms for divisive hierarchical clustering. *Journal of Classification*, **8**, 5–30.
- Guha, S., Rastogi, R., and Shim, K. (1998). Cure: An efficient clustering algorithm for large databases. In *Proc. ACM SIGMOD Int. Conf. Management of Data*, pages 73–84.

- Guha, S., Rastogi, R., and Shim, K. (2000). Rock: A robust clustering algorithm for categorical attributes. *Information Systems*, **25** (5), 345–366.
- Hansen, P. and Mladenoviae, N. (2001). J-means: A new local search heuristic for minimum sum of squares clustering. *Pattern Recognition*, **34**, 405–413.
- Hathaway, R., Bezdek, J., and Hu, Y. (2000). Generalized fuzzy c-means clustering strategies using lp norm distances. *IEEE Transactions on Fuzzy Systems*, **8** (5), 576–582.
- Hoepfner, F., Klawonn, F., and Kruse, R. (1999). Fuzzy cluster analysis: Methods for classification, data analysis, and image recognition. *Wiley, New York*.
- Huang, Z. (1998). Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, **2**, 283–304.
- Hung, M. and Yang, D. (2001). An efficient fuzzy c-means clustering algorithm. *Proc. IEEE Int. Conf. Data Mining*, pages 225–232.
- Kanungo, T., Mount, D., Netanyahu, N., Piatko, C., Silverman, R., and Wu, A. (2000). An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions in Pattern Analysis Machine Intelligence*, **24** (7), 881–892.
- Karypis, G., Han, E., and Kumar, V. (1999). Chameleon: Hierarchical clustering using dynamic modeling. *IEEE Computer*, **32** (8), 68–75.
- Kaufman, L. and Rousseeuw, P. (1990). Finding groups in data. *Wiley, New York*.
- Kaufman, L. and Rousseeuw, P. J. (1987). Clustering by means of medoids. *Statistical data analysis based on the L1-norm and related methods*, pages 405–416.
- Kolen, J. and Hutcheson, T. (2002). Reducing the time complexity of the fuzzy c-means algorithm. *IEEE Transactions on Fuzzy Systems*, **10** (2), 263–267.
- Lance, G. and Williams, W. (1968). Note on a new information statistic classification program. *The Computer Journal*, **11**, 195–197.
- Qiang Wang, Y. Y. and Huang, J. Z. (2008). Fuzzy k-means with variable weighting in high dimensional data analysis. *The Ninth International Conference on Web-Age Information Management*, pages 365–372.
- Sneath, P. and Sokal, R. (1973). Numerical taxonomy. *Freeman, San Francisco*.
-

- Su, M. and Chou, C. (2001). A modified version of the k-means algorithm with a distance based on cluster symmetry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **23** (6), 674–680.
- Xu, R. and Wunsch, D. (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, **16** (3), 645–678.
- Zhang, T., Ramakrishnan, R., and Livny, M. (1996). Birch: An efficient data clustering method for very large databases. In *Proc. ACM SIGMOD Conf. Management of Data*, pages 103–114.