Agrupamento baseado em SOM com pesos adaptativos para múltiplas tabelas de dissimilaridade

Anderson B. S. Dantas[1], **Francisco de A. T. de Carvalho**[1]

[1]Centro de Informática – Universidade Federal de Pernambuco (CIn/UFPE)
Av. Prof. Luiz Freire, s/n - Cidade Universitária, CEP 50740-540, Recife – PE – Brazil

{absd,fatc}@cin.ufpe.br

Abstract. *This paper introduces a clustering algorithm based on batch Self-organizing map to partition objects taking into account their relational descriptions given by multiple dissimilarity matrices. The presented approach provide a partition of the objects and a prototype for each cluster, moreover the method is capable of learn relevance weights for each dissimilarity matrix by optimizing an adequacy criterion that measures the fit between clusters and the respectives prototypes. These relevance weights change at each iteration and are different from one cluster to another. Experiments using real-world data bases are considered to show the usefulness of the method.*

Resumo. *Este trabalho apresenta um modelo de algoritmo de agrupamento baseado no algoritmo SOM com treinamento batch para classificar objetos levando em consideração suas descrições relacionais dadas por múltiplas matrizes de dissimilaridade. O método apresentado tem como resultado uma partição dos dados de entrada, assim como um protótipo para cada agrupamento, além de adaptar pesos que calculam a relevância de cada matriz de dissimilaridade otimizando um critério de adequação entre os agrupamentos e seus respectivos protótipos. Estes pesos mudam a cada iteração do algoritmo e são diferentes de um grupo para outro. Experimentos com bases de dados reais foram realizado com o objetivo de mostrar a utilidade do algoritmo.*

1. Introdução

Algoritmos de agrupamento (*clutering*) têm sido largamente usados em áreas de aprendizagem de máquina como mineração de dados, recuperação de documentos, segmentação de imagens e classificação de padrões [Jain et al. 1999]. Métodos de agrupamento objetivam organizar um conjunto de dados em grupos (*clusters*), de forma que itens pertencentes a um mesmo grupo (*cluster*) tenham alto grau de similaridade, por outro lado, itens em grupos diferentes possuem alto grau de dissimilaridade.

O mapa auto-organizável de Kohonen (Self-organizing map) [Kohonen 1990] é um tipo especial de rede neural não-supervisionada e possui propriedades de agrupamento e visualização [Gopalakrishnan et al. 2008]. SOM pode ser considerado um algoritmo que faz a projeção de dados multidimensionais em um espaço de uma, duas ou, em casos especiais, três dimensões. Esta projeção possibilita uma partição das entradas em grupos similares ao mesmo tempo em que preserva sua topologia.

Existem duas representações comuns dos objetos em que os algoritmos de agrupamento podem ser baseados: dados caracterizados ou relacionais. Quando cada objeto é

descrito por um vetor de valores quantitativos ou qualitativos, os vetores que descrevem os objetos são chamados dados caracterizados. Quando cada par de objetos é representado por uma relação então temos dados relacionais. O modelo mais comum de dados relacionais é o caso de uma matriz de dissimilaridades $R = [r_{il}]$, onde r_{il} é a dissimilaridade pareada (geralmente uma distância) entre os objetos i e l.

[Golli et al. 2004] propõe uma adaptação do modelo SOM para dados de dissimilaridade e [Conan-Guez et al. 2006] desenvolve diversas modificações para uma implementação mais rápida do algoritmo e para torná-lo mais eficaz. [Lechevallier et al. 2010] apresenta um algoritmo de agrupamento que realiza partição de objetos baseado nas descrições relacionais destes objetos dadas por múltiplas matrizes de dissimilaridade.

Este trabalho propõe um algoritmo baseado em SOM com pesos adaptativos para classificar dados baseados em matrizes de dissimilaridades. Estas matrizes podem ser obtidas através de diferentes conjuntos de variáveis e uma função de dissimilaridade fixa (neste caso a partição final apresenta um consenso entre diferentes conjuntos de variáveis descrevendo os objetos), através de um conjunto fixo de variáveis e diferentes funções de dissimilaridade (consenso entre diferentes funções de dissimilaridade) ou através de diferentes conjuntos de variáveis e funções de dissimilaridade. A relevância de diferentes matrizes de dissimilaridades não é igualitária na definição final dos agrupamentos. Assim, para obter uma partição significativa partindo de todas as matrizes faz-se necessário o uso de pesos adaptativos para cada matriz de dissimilaridade e diferentes para cada agrupamento.

Este trabalho está organizado da seguinte forma: a seção 2 traz uma introdução sobre o algoritmo SOM e sua definição formal. A seção 3 trata do algoritmo SOM com treinamento batch aplicado a dados de dissimilaridade. O algoritmo SOM com pesos adaptativos aplicado a dados representados por matrizes de dissimilaridade é apresentado na seção 4. Os experimentos realizados com bases de dados reais são apresentados na seção 5. Finalmente, a seção 6 traz as conclusões do trabalho.

2. Self-organizing map (SOM)

A classe de métodos chamados *self-organizing maps* [Kohonen 1990] (mapas auto-organizáveis) segue uma série de procedimentos para associar um número finito de vetores de objetos (entradas) com um número finito de pontos representativos, tal que as relações de semelhança entre as entradas são respeitadas por esses pontos. O algoritmo SOM representa uma abordagem de aprendizado não-supervisionado, todas as propriedades de cada grupo são estimados ou aprendidos sem o uso de informação a priori [Murtagh and Hernández-Pajares 1995].

O SOM consiste em um conjunto de neurônios organizados topologicamente em uma grade, geralmente com uma, duas, ou três dimensões, chamada mapa. De modo mais formal, o mapa é descrito por um grafo não-orientado (C, Γ). C é um conjunto de m neurônios interconectados com topologia definida por Γ. A estrutura de grafo permite que se defina uma função de distância entre dois neurônios no mapa. Para cada par de neurônios (c,r) no mapa, $\delta(c, r)$ é o comprimento do caminho mais curto entre c e r no grafo C. Essa função de distância permite que haja uma relação de vizinhança entre neurônios.

O conceito de vizinhança é incorporado ao algoritmo através do uso de uma função *kernel* K positiva e tal que $\lim_{|x|\to\infty} K(x) = 0$ [Badran et al. 2005]. As distâncias $\delta(c, r)$ entre os neurônios *c* e *r* do mapa permitem a definição da influência relativa dos neurônios sobre os objetos. $K(\delta(c, r))$ quantifica essa influência. Os neurônios e os padrões de entrada são ambos associados a vetores m-dimensionais. O neurônio com melhor correspondência será atualizado, como também os neurônios na vizinhança ao redor deste "vencedor". A região ao redor do neurônio que melhor representa um dado objeto de entrada é modificada para se aproximar mais do objeto apresentado. O resultado é que os neurônios no mapa ficam ordenados, neurônios vizinhos possuem vetores referência (ou protótipos) similares.

Considere um vetor $\mathbf{w}_i(\mathbf{w}_i \in \mathbb{R}^p)$ associado a cada elemento i do mapa. O vetor do elemento i na iteração t é $\mathbf{w}_i^{(t)}$. Atribua $t = 0$.

Etapa 0: Escolha um vetor $\mathbf{x}(\mathbf{x} \in \mathbb{R}^p)$ aleatoriamente do conjunto de entradas.

Etapa 1: Determine o neurônio $c = i$ tal que $\| \mathbf{x} - \mathbf{w}_i^{(t)} \|$ é mínimo. Onde $\| \cdot \|$ representa uma medida de dissimilaridade que pode ser expressa em termos de uma função de distância, por exemplo, a distância Euclidiana seria uma escolha plausível.

Etapa 2: Para todos os elementos i do mapa:

$$\begin{aligned} \mathbf{w}_i^{(t+1)} &= \mathbf{w}_i^{(t)} + \alpha^{(t)}(\mathbf{x} - \mathbf{w}_i^{(t)}) \text{ se } i \in N_c(t) \\ &= \mathbf{w}_i^{(t)} \text{ se } i \notin N_c(t) \end{aligned}$$

onde $\alpha^{(t)}$ é uma pequena taxa usada para controlar a convergência e $N_c(t)$ é a vizinhança do elemento c do mapa.

Etapa 3: Incremente t; retorne ao Passo 0, a menos que uma condição de parada seja atingida.

É importante notar que neste tipo de algoritmo somente um indivíduo é apresentado por vez (Etapa 0), os neurônios do mapa se ajustam às entradas a medida em que elas são apresentadas.

3. Batch self-organizing map aplicado em dados de dissimilaridade

3.1. SOM batch para dados de dissimilaridade

O algoritmo SOM batch baseado em dados de dissimilaridade conforme proposto por [Golli et al. 2004] também é descrito por um grafo (C, Γ). A principal diferença está nos dados que serão classificados. Estes dados são representados por uma relação de dissimilaridade. O algoritmo de treinamento do tipo batch é um algoritmo iterativo onde todo o conjunto de exemplos (E) é apresentado ao mapa antes que qualquer ajuste seja realizado. A seguir é dada a definição formal deste modelo.

Seja $E = x_1, \ldots, x_n$ um conjunto de n objetos. Cada neurônio c é representado por um protótipo $a_c = x_{j_i},\ x_{j_i} \in E$. A distância entre um objeto x_i e o protótipo a_c de um neurônio é descrita pela dissimilaridade d^T de $E \times P(E)$ em $\mathbb{R}^+$ definida por:

$$d^T(x_i, a_c) = \sum_{l \in C} K^T(\delta(c, l))\, d^2(x_i, a_l) \tag{1}$$

Tal dissimilaridade é baseada na função kernel positiva K, como descrita na seção anterior. K é usada para definir uma família de funções K^T parametrizadas por T, onde $K^T(\delta) = K(\frac{\delta}{T})$. O parâmetro T controla o tamanho da vizinhança dos neurônios: se o valor de T for pequeno, poucos neurônios serão atingidos pela vizinhança. Um simples exemplo para K^T é definido por $K^T(\delta) = e^{-\frac{\delta^2}{T^2}}$.

Durante a etapa de aprendizado, a seguinte função custo (ou função de adequação) E é minimizada alternando-se o passo de afetação e o passo de representação:

$$E(f, a) = \sum_{i=1}^{n} d^T(x_i, a_{f(x_i)}) = \sum_{i=1}^{n} \sum_{l \in C} K^T(\delta(f(x_i), l)) d^2(x_i, a_l) \tag{2}$$

Esta função calcula o ajuste entre a partição calculada pela função de afetação e os protótipos a do mapa.

Durante a etapa de afetação, a função f afeta cada indivíduo x_i ao neurônio mais próximo. O Grupo C_r que minimiza a função de adequação é definida da seguinte forma:

$$r = f(x_i) = arg \min_{c \in C} d^T(x_i, a_c) = arg \min_{c \in C} \sum_{l \in C} K^T(\delta(c, l))\, d^2(x_i, a_l) \tag{3}$$

Durante a etapa de representação, são selecionados novos protótipos que representam o conjunto de observações. Esta etapa de otimização pode ser realizada independentemente para cada neurônio. O protótipo a_l^* do Grupo C_l, que minimiza o critério de adequação E é calculado segundo a equação:

$$a_l^* = argmin_{x \in E} \sum_{i=1}^{n} \exp\{-\frac{(\delta((f(x_i))^{(t-1)}, l))^2}{2T^2}\} d^2(x_i, x) \tag{4}$$

3.2. O Algoritmo

Cada iteração do algoritmo alterna entre duas fases (afetação e representação) objetivando minimizar a função de adequação.

Inicialização: iteração $k = 0$; selecione os protótipos iniciais a^0; fixe $T = T_{max}$ e o número total de iterações N_{iter}
Iteração: Na iteração k, o conjunto de protótipos da iteração anterior a^{k-1} é conhecido. Calcule o novo valor de T:

$$T = T_{max} * (\frac{T_{min}}{T_{max}})^{\frac{k}{N_{iter}-1}}$$

- **etapa de afetação**: Afetar cada indivíduo X_i ao protótipo como definido na equação 3.
- **etapa de representação**: determinar os novos protótipos a^k* que minimizam a função $E(f_{a^k}, a)$, a_c^k* é definido pela equação 4.

Repetir **Iteração** até $T = T_{min}$.

4. Self-organizing map adaptativo para múltiplas tabelas de dissimilaridade

4.1. Introdução

Nesta seção apresentamos um algoritmo adaptativo baseado no algoritmo SOM para dados de dissimilaridade [Golli et al. 2004] e no algoritmo de agrupamento para múltiplas tabelas de dissimilaridade de [Lechevallier et al. 2010]. O objetivo do modelo apresentado é mapear objetos levando em conta suas descrições relacionais dadas por múltiplas matrizes de dissimilaridade. O algoritmo é capaz de associar dados de entrada com pontos representativos de um mapa formando grupos de objetos. O método utiliza pesos adaptativos diferentes de um grupo para outro com o intuito de ponderar sobre as matrizes de dissimilaridades e medir sua relevância na formação de cada agrupamento.

Seja $E = \{e_1, \ldots, e_n\}$ um conjunto de n objetos e p o número de matrizes de dissimilaridade $\mathbf{D}_j = [d_j(e_i, e_l)](j = 1, \ldots, p)$, onde $d_j(e_i, e_l)$ é a dissimilaridade entre os objetos e_i e $e_l(i, l = 1, \ldots, n)$ na matriz de dissimilaridades $\mathbf{D}_j$. Assuma que o protótipo g_l do grupo (*cluster*) C_l é um elemento do conjunto de objetos E, i.e., $g_l \in E \; \forall l = 1, \ldots, L$.

O algoritmo SOM ponderado para múltiplas tabelas de dissimilaridade calcula uma partição $P = (C_1, \ldots, C_l)$ de E em L grupos e o respectivo protótipo g_l representando o grupo C_l em P tal que um determinado critério de adequação (função objetivo) que mede o ajuste entre os grupos e seus repectivos protótipos é localmente otimizado. O critério de adequação é definido como:

$$\begin{aligned} E &= \sum_{i=1}^{n}\sum_{l=1}^{c} K(\delta(f(e_i), l), T) D_{\lambda_l}(e_i, g_l) \\ &= \sum_{i=1}^{n}\sum_{l=1}^{c} \exp\{-\frac{(\delta((f(e_i)), l))^2}{2T^2} \sum_{j=1}^{p} \lambda_{lj} d_j(e_i, g_l) \end{aligned} \quad (5)$$

onde K é uma função kernel positiva, tal que $\lim_{|\delta|\to\infty} K(\delta) = 0$. $\delta(c, r)$ calcula a distância entre dois neurônios no mapa. A matriz de pesos λ é composta por c vetores de pesos $\lambda_k = (\lambda_k^1, \ldots, \lambda_k^j, \ldots, \lambda_k^p)(k = 1, \ldots, c)$, muda a cada iteração do algoritmo, ou seja, os vetores não possuem valores determinados absolutamente e são diferentes para cada matriz de dissimilaridades. O objetivo da matriz de pesos é ponderar sobre cada matriz de dissimilaridade avaliando sua relevância na minimização do critério de adequação. O modelo segue três etapas definidas (representação, atualização de pesos e afetação):

Etapa 1: Definição dos melhores protótipos

Nesta etapa, a partição $P^{(t-1)} = (P_1^{(t-1)}, \ldots, P_c^{(t-1)})$ e $\lambda_l^{(t)}(l = 1, \ldots, c)$ são fixos. O protótipo $g_l^{(t)} = g^* \in E$ do grupo C_l, que minimiza o critério de adequação E é calculado segundo a equação:

$$g^* = argmin_{e \in E} \sum_{i=1}^{n} \exp\{-\frac{(\delta((f(e_i))^{(t-1)}, l))^2}{2T^2}\} \sum_{j=1}^{p} \lambda_{lj}^{(t-1)} d_j(e_i, e) \quad (6)$$

Etapa 2: Definição dos melhores pesos

Nesta etapa, a partição $P^{(t-1)} = (P_1^{(t-1)}, \ldots, P_c^{(t-1)})$ e os protótipos $g_l^{(t)} \in E(l = 1, \ldots, c)$ são fixos. O elemento j do vetor de pesos $\lambda_l = (\lambda_k^1, \ldots, \lambda_k^p)$, que minimiza o critério de adequação E é calculado segundo a expressão:

$$\lambda_{lj}^{(t)} = \frac{\left\{\prod_{h=1}^{p}(\sum_{i=1}^{n}\exp\{-\frac{\delta^2(f^{(t-1)}(x_i),l)}{2T^2}\}d_h(e_i, g_l^{(t)}))\right\}^{\frac{1}{p}}}{\sum_{i=1}^{n}\exp\{-\frac{\delta^2(f^{(t-1)}(x_i),l)}{2T^2}\}d_j(e_i, g_l^{(t)})} \tag{7}$$

Etapa 3: Definição da melhor partição

Nesta etapa, os protótipos $g_l^{(t)} \in E(l = 1, \ldots, c)$ e $\lambda_l^{(t)}(l = 1, \ldots, c)$ são fixos. O grupo C_r que minimiza o critério E é determinado de acordo com a equação:

$$r = (f(e_i))^{(t)} = argmin_{1\leq h\leq c}\sum_{l=1}^{c}\exp\{-\frac{(\delta(h,l))^2}{2T^2}\}\sum_{j=1}^{p}\lambda_{lj}^{(t)}d_j(e_i, g_l^{(t)}) \tag{8}$$

4.2. O algoritmo

1. *Inicialização*
   Fixe o número c de grupos;
   Fixe δ;
   Fixe a função kernel K;
   Fixe o número de iterações N_{iter};
   Fixe T_{min}, T_{max}; Atribua $T \leftarrow T_{max}$; Atribua $t \leftarrow 0$;
   Selecione c protótipos aleatoriamente $g_l^{(0)} \in E(l = 1, \ldots, c$;
   Configure $\lambda_l^{(0)} = (\lambda_{l1}^{(0)}, \ldots, \lambda_{lp}^{(0)}) = (1, \ldots, 1)(l = 1, \ldots, c)$;
   Configure o mapa $L(c, \mathbf{G}^0)$, onde $\mathbf{G}^0 = (g_1^{(0)}, \ldots, g_c^{(0)})$.
   Cada objeto e_i é afetado ao protótipo mais próximo com o objetivo de obter a partição $P^{(0)} = (P_1^{(0)}, \ldots, P_c^{(0)})$ de acordo com o seguinte critério:

$$(f(e_i))^{(0)} = argmin_{1\leq r\leq c}\sum_{l=1}^{c}\exp\{-\frac{(\delta(r,l))^2}{2T^2}\}\sum_{j=1}^{p}\lambda_{lj}^{(0)}d_j(e_i, g_l) \tag{9}$$

2. *Passo 1: Representação*
   Atribua $t = t + 1$;
   Atribua $T = T_{max}(\frac{T_{min}}{T_{max}})^{\frac{t}{N_{iter}-1}}$
   A partição $P^{(t-1)} = (P_1^{(t-1)}, \ldots, P_c^{(t-1)})$ e $\lambda_l^{(t)}(l = 1, \ldots, c)$ são fixos.
   Calcule o protótipo $g_l^{(t)} = g^* \in E$ do grupo $P_l^{(t-1)}(l = 1, \ldots, c)$ de acordo com a equação 6.
3. Passo 2: Atualização dos pesos
   A partição $P^{(t-1)} = (P_1^{(t-1)}, \ldots, P_c^{(t-1)})$ e os protótipos $g_l^{(t)} \in E(l = 1, \ldots, c)$ são fixos.
   Atualize o vetor de pesos de acordo com a equação 7.

4. *Passo 3: Afetação*
   Os protótipos $g_l^{(t)} \in E(l = 1, \ldots, c)$ e $\lambda_l^{(t)}(l = 1, \ldots, c)$ são fixos
   $P^{(t)} \leftarrow P^{(t-1)}$
   para $i = 1$ até n faça:
   encontre o grupo $C_m^{(t)}$ ao qual o objeto e_i pertence
   encontre o grupo vencedor $C_r^{(t)}$, onde r é obtido pela equação **??**

   se $r \neq m$:
   $C_r^{(t)} \leftarrow C_r^{(t)} \cup \{e_i\}$
   $C_m^{(t)} \leftarrow C_m^{(t)} \setminus \{e_i\}$

5. *Critério de parada*
   Se $T == T_{min}$ então PARE; senão volte ao passo 1

5. Experimentos e resultados

Para mostrar a utilidade do algoritmo proposto, foram realizados experimentos utilizando bases de dados obtidas do repositório UCI Machine Learning Repository [Frank and Asuncion 2010]. No algoritmo SOM adaptativo para múltiplas tabelas, cada um dos atributos que compõem as bases é representado por uma matriz de dissimilaridade. Para o modelo SOM batch não-adaptativo é considerada somente uma matriz de dissimilaridade. Com a finalidade de avaliar os resultados dos agrupamentos realizados pelo método proposto neste trabalho foi considerado um índice externo: o índice de Rand corrigido (CR) [Hubert and Arabie 1985], a medida F-*measure* [van Rijsbergen 1979] e a taxa de erro global de classificação (OERC - overall error rate of classification) [Breiman et al. 1984].

O índice CR avalia o grau de similaridade entre a partição a priori e a partição fornecida pelo algoritmo de agrupamento. Além disso, o índice CR não é sensível ao número de classes nas partições ou à distribuição dos itens dentro dos grupos. O índice CR pode assumir valores no intervalo [-1,1], onde o valor 1 indica perfeita combinação entre as partições, valores perto de 0 ou negativos indicam pouca semelhança. O índice F-*measure* entre a partição a priori e a partição obtida pelo algoritmo de agrupamento assume valores no intervalo [0,1], em que 1 indica perfeita combinação entre as partições. O índice OERC mede a habilidade de um algoritmo de agrupamento encontrar classes a priori presentes na base de dados.

Alguns parâmetros são considerados para a realização dos experimentos. São eles: topologia do mapa, o valor de T_{min}, T_{max} e o número de iterações N_{iter}. A topologia do mapa define o número máximo de grupos que serão formados. Todos os parâmetros foram definidos empiricamente. A tabela 1 mostra os valores dos parâmetros utilizados em todos os experimentos descritos a seguir. Cada experimento foi repetido 100 vezes com a mesma configuração, dentre as 100 repetições foi selecionada a que obteve menor critério de adequação.

5.1. Base de dados Wine

A base de dados wine consiste em resultados da análise química de vinhos produzidos numa mesma região na Itália, mas derivados de três cultivos diferentes, formando três

Tabela 1. Parâmetros utilizados nos experimentos

Parâmetros	Valor
Topologia	2x5
T_{min}	0,2
T_{max}	2,0
N_{iter}	300

classes de vinhos. As classes 1, 2 e 3 possuem, respectivamente 59, 71 e 48 instâncias. Cada indivíduo é descrito por 13 atributos com valores reais representando os valores de 13 componentes encontrados em cada um dos três tipos de vinho.

As tabelas 2 e 3 mostram a matriz de confusão obtidas pelos algoritmos batch SOM e batch SOM adaptativo para múltiplas tabelas de dissimilaridade.

Tabela 2. Wine: Matriz de confusão obtida pelo algoritmo batch SOM

Classe/Grupo	0,0	0,1	0,2	0,3	0,4	1,0	1,1	1,2	1,3	1,4
1	0	0	0	14	6	0	8	5	10	16
2	17	8	16	0	0	16	3	10	0	1
3	0	14	3	0	0	14	5	12	0	0

Tabela 3. Wine: Matriz de confusão obtida pelo algoritmo batch SOM adaptativo

Classe/Grupo	0,0	0,1	0,2	0,3	0,4	1,0	1,1	1,2	1,3
1	0	0	0	13	31	0	0	0	15
2	20	21	0	0	8	18	4	0	0
3	0	0	7	0	0	0	22	19	0

Os índices CR, F-*measure* e OERC obtiveram, respectivamente, os valores 0,45, 0,55, 0,067 para o modelo SOM adaptativo, e os valores, respectivamente 0,29, 0,40, 0,27 para o modelo SOM batch não-adaptativo. O algoritmo SOM adaptativo para múltiplas tabelas de dissimilaridade obteve melhores índices do que o modelo batch não-adaptativo.

A tabela 4 mostra a matriz de pesos final obtida pelo algoritmo batch SOM adaptativo para múltiplas tabelas de dissimilaridade. Cada peso representa a relevância de uma determinada matriz de dissimilaridade para a formação de cada grupo. Valores acima de 1 indicam uma forte influência da matriz durante a afetação de objetos em um determinado grupo. Por exemplo, na tabela 4 o grupo 0,0 tem uma influência maior pela matriz 1 (3,19).

5.2. Base de dados Iris

Esta base de dados consiste em três tipos (classes) de planta íris: iris setosa, iris versicolor e iris virginica. Cada uma das classes possui 50 exemplos. Cada instância é descrita por quatro atributos.

As tabelas 5 e 6 mostram a matriz de confusão obtidas pelos algoritmos batch SOM e batch SOM adaptativo para múltiplas tabelas de dissimilaridade.

Tabela 4. Wine: Matriz de pesos obtida pelo algoritmo batch SOM adaptativo

Matriz/Grupo	0,0	0,1	0,2	0,3	0,4	1,0	1,1	1,2	1,3
1	**3,19**	**1,2**	**1,27**	0,6	0,71	0,47	**1,19**	0,7	**1,5**
2	**1,68**	0,27	**1,14**	**4,0**	0,73	0,51	0,27	0,41	**6,17**
3	0,42	0,43	**10,78**	0,73	0,41	0,43	**1,25**	0,76	**1,02**
4	0,76	0,74	**4,35**	0,21	0,26	0,5	**1,01**	0,58	0,87
5	0,11	0,95	0,65	**1,06**	0,47	**1,27**	0,5	0,5	0,99
6	**1,04**	**2,23**	0,28	**1,97**	**1,35**	0,57	**1,03**	**1,74**	0,56
7	**1,8**	**2,93**	**2,24**	**1,71**	**1,8**	**6,17**	**3,91**	**5,0**	0,79
8	0,95	0,64	0,36	0,69	**1,53**	0,61	0,33	0,31	0,47
9	0,46	0,17	0,24	0,63	**1,73**	0,8	**1,83**	**1,54**	**1,13**
10	**2,52**	**8,87**	0,22	**1,06**	**2,6**	**4,97**	0,6	0,37	0,87
11	**1,44**	0,23	0,97	**1,35**	**1,6**	0,27	0,61	**1,17**	0,86
12	0,75	**2,0**	**1,93**	**2,65**	**1,64**	**1,56**	**2,1**	**7,42**	**1,46**
13	**2,35**	**3,24**	0,96	0,43	**1,64**	**4,1**	**2,6**	**1,12**	0,47

Tabela 5. Iris: Matriz de confusão obtida pelo algoritmo batch SOM

Classe/Grupo	0,0	0,1	0,2	0,3	0,4	1,0	1,1	1,2	1,3	1,4
1	0	0	0	11	17	0	0	0	0	22
2	18	13	16	0	0	0	1	2	0	0
3	3	0	1	0	0	8	15	11	12	0

Tabela 6. Iris: Matriz de confusão obtida pelo algoritmo SOM adaptativo

Classe/Grupo	0,0	0,1	0,2	0,3	0,4	1,0	1,1	1,2	1,3	1,4
1	0	0	0	0	0	0	0	0	13	37
2	10	6	16	0	17	1	0	0	0	0
3	0	7	0	3	0	15	16	9	0	0

Os índices CR, F-*measure* e OERC obtiveram, respectivamente, os valores 0,53, 0,61 e 0,04 para o modelo SOM adaptativo para múltiplas tabelas de dissimilaridade, e os valores, respectivamente 0,43, 0,52, 0,04 para o modelo SOM batch não-adaptativo. Os dois algoritmos obtiveram desempenhos semelhantes para esta base de dados.

A tabela 7 mostra a matriz de pesos final obtida pelo algoritmo batch SOM adaptativo para múltiplas tabelas de dissimilaridade.

5.3. Base de dados Thyroid

Esta base possui dados sobre os estados da glândula tireóide: normal, hipertireoidismo e hipotireoidismo. As três classes (1, 2 e 3) possuem, respectivamente, 150, 35 e 30 exemplos. Cada instância é descrita por 5 atributos com valores reais.

As tabelas 8 e 9 mostram a matriz de confusão obtidas pelos algoritmos batch SOM e batch SOM adaptativo para múltiplas tabelas de dissimilaridade.

Os índices CR, F-*measure* e OERC obtiveram, respectivamente, os valores 0,63, 0,66 e 0,03 para o modelo SOM adaptativo, e os valores, respectivamente 0,36, 0,50 e 0,13

Tabela 7. Iris: Matriz de pesos obtida pelo algoritmo batch SOM adaptativo

Matriz/Grupo	0,0	0,1	0,2	0,3	0,4	1,0	1,1	1,2	1,3	1,4
1	0,25	**2,13**	0,36	0,55	0,12	**2,11**	0,6	0,16	0,42	0,25
2	0,28	0,10	0,15	**1,11**	0,55	0,33	0,88	0,69	0,11	0,06
3	**1,37**	**6,05**	**2,51**	**1,59**	**4,49**	**4,4**	**1,41**	**1,93**	**5,04**	**5,12**
4	**10,14**	0,75	**7,31**	**1,03**	**3,22**	0,32	**1,32**	**4,46**	**4,07**	**12,03**

Tabela 8. Thyroid: Matriz de confusão obtida pelo algoritmo batch SOM

Classe/Grupo	0,0	0,1	0,2	0,3	0,4	1,0	1,1	1,2	1,3	1,4
1	0	8	41	20	0	0	1	49	30	1
2	0	3	0	4	5	0	0	1	9	13
3	12	3	1	0	0	5	5	2	2	0

Tabela 9. Thyroid: Matriz de confusão obtida pelo algoritmo batch SOM adaptativo

Classe/Grupo	0,0	0,1	0,2	0,3	0,4	1,1	1,2	1,3	1,4
1	75	74	1	0	0	0	0	0	0
2	0	2	0	0	9	24	0	0	0
3	2	2	4	10	0	0	4	5	3

para o modelo SOM batch não-adaptativo. O algoritmo SOM adaptativo para múltiplas tabelas de dissimilaridade obteve melhores índices do que o modelo batch não-adaptativo.

A tabela 10 mostra a matriz de pesos final obtida pelo algoritmo batch SOM adaptativo para múltiplas tabelas de dissimilaridade.

5.4. Base de dados Ecoli

A base de dados Ecoli contém dados de proteínas classificadas em 8 classes: (1) cp (citoplasma), com 143 exemplos; (2) im (membrana interna sem sequência de sinal), com 77 exemplos; (3) pp (perisplasm), com 52 exemplos; (4) imU (inner membrane, uncleavable signal sequence), com 35 exemplos; (5) om (outer membrane), com 20 exemplos; (6) omL (outer membrane lipoprotein), com 5 exemplos; (7) imL (inner membrane lipoprotein) com 2 exemplos e (8) imS (inner membrane, cleavable signal sequence), com 2 exemplos. Cada indivíduo é descrito por 6 atributos.

As tabelas 11 e 12 mostram a matriz de confusão obtidas pelos algoritmos batch SOM e batch SOM adaptativo para múltiplas tabelas de dissimilaridade, respectivamente.

Os índices CR, F-*measure* e OERC obtiveram, respectivamente, os valores 0,43, 0,52 e 0,24 para o modelo SOM adaptativo, e os valores, respectivamente 0,37, 0,52 e 0,24 para o modelo SOM batch não-adaptativo. O algoritmo SOM adaptativo para múltiplas tabelas de dissimilaridade obteve índices semelhantes ao modelo batch não-adaptativo.

A tabela 13 mostra a matriz de pesos final obtida pelo algoritmo batch SOM adaptativo para múltiplas tabelas de dissimilaridade.

Tabela 10. Thyroid: Matriz de pesos obtida pelo algoritmo batch SOM adaptativo

Grupo/Matriz	0,0	0,1	0,2	0,3	0,4	1,0	1,1	1,2	1,3	1,4
1	0,16	0,16	0,12	0,67	0,03	0,25	0,04	0,79	**5,64**	0,66
2	0,55	0,32	**2,44**	**3,31**	0,94	0,28	0,33	**8,56**	**1,97**	**2,08**
3	**1,88**	0,59	**1,62**	**3,31**	0,04	0,19	0,034	**2,79**	**1,02**	0,33
4	**5,27**	**19,76**	0,57	0,09	**17,26**	**22,55**	**20,44**	0,02	0,89	**2,71**
5	**1,13**	**1,63**	**3,74**	**1,37**	**41,08**	**3,24**	**103,65**	**3,12**	0,09	0,8

Tabela 11. Ecoli: Matriz de confusão obtida pelo algoritmo batch SOM

Classe/Grupo	0,0	0,1	0,2	0,3	0,4	1,0	1,1	1,2	1,3	1,4
1	47	18	19	0	0	25	27	6	1	0
2	0	1	0	2	25	0	0	0	0	7
3	0	0	0	0	1	0	0	0	1	0
4	0	0	0	0	1	0	0	1	0	0
5	5	0	0	32	24	1	0	0	1	14
6	1	1	0	0	1	0	0	13	35	1
7	0	0	0	0	0	0	0	4	1	0
8	0	0	0	0	0	0	0	8	11	1

Tabela 12. Ecoli: Matriz de confusão obtida pelo algoritmo batch SOM adaptativo

Classe/Grupo	0,0	0,1	0,2	0,3	0,4	1,0	1,2	1,3	1,4
1	0	0	3	36	57	0	0	3	44
2	10	6	1	0	0	17	0	1	0
3	0	1	0	0	0	0	0	1	0
4	1	0	1	0	0	0	0	0	0
5	23	17	0	8	0	28	0	1	0
6	0	1	5	2	2	0	13	29	0
7	0	0	5	0	0	0	0	0	0
8	0	0	1	0	0	0	11	8	0

6. Conclusão

Este trabalho apresentou um modelo de agrupamento baseado no algoritmo SOM com treinamento batch. Este modelo realiza o agrupamento de objetos levando em consideração seus dados relacionais descritos por uma tabela de dissimilaridades. Como saída, o algoritmo produz uma partição dos dados de entrada, os protótipos de cada grupo e pesos adaptativos que indicam a relevância de cada tabela de dissimilaridade na formação dos agrupamentos, otimizando um critério de adequação que mede o ajuste entre os grupos e seus protótipos. Tais pesos mudam a cada iteração e são diferentes para cada grupo. Experimentos utilizando bases de dados reais foram realizados para mostrar a utilidade do modelo proposto comparando-o com o modelo apresentado por [Golli et al. 2004].

Tabela 13. Ecoli: Matriz de pesos obtida pelo algoritmo batch SOM adaptativo

Grupo/Matriz	0,0	0,1	0,2	0,3	0,4	1,0	1,2	1,3	1,4
1	0,36	0,49	0,96	0,86	**1,26**	0,5	**2,44**	**2,55**	0,48
2	0,5	0,75	0,88	**1,46**	0,9	0,65	0,83	0,53	0,72
3	0,53	0,54	0,78	**1,12**	0,63	0,64	0,25	0,38	0,46
4	**2,55**	**2,27**	**2,09**	0,78	**1,2**	**2,15**	**1,93**	**1,53**	**2,28**
5	**3,96**	**2,16**	0,71	0,89	**1,15**	**2,2**	0,98	**1,25**	**2,69**

Referências

Badran, F., Yacoub, M., and Thiria, S. (2005). Self-organizing maps and unsupervised classification. In *Neural networks: methodology and applications*, pages 379–442. Springer-Verlag.

Bock, H.-H. and Diday, E. (2000). *Analysis of Symbolic Data*. Springer, Heidelberg.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. (1984). *Classification and Regression Trees*. Chapman and Hall/CRC.

Conan-Guez, B., Rossi, F., and Golli, A. E. (2006). Fast algorithm and implementation of dissimilarity self-organizing maps. *Neural Networks*, 19:855–863.

Frank, A. and Asuncion, A. (2010). Uci machine learning repository.

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.

Gopalakrishnan, K., Khaitan, S., and Manik, A. (2008). Enhanced clustering analysis and visualization using kohonen?s self-organizing feature map networks. *International Journal of Computational Intelligence*, 4:64–71.

Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of Classification*, 2:193–218.

Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data clustering: a review. *ACM Comput. Surv.*, 31:264–323.

Kohonen, T. (1990). The self-organizing maps. *Proceedings of the IEEE*, 78:1464–1480.

Lechevallier, Y., de Carvalho, F. A. T., Despeyroux, T., and de Melo, F. M. (2010). Clustering of multiple dissimilarity data tables for documents categorization. *19th International Conference on Computational Statistics - COMPSTAT 2010*, pages 1263–1270.

Murtagh, F. and Hernández-Pajares, M. (1995). The kohonen self-organizing map method: An assessment. *Journal of Classification*, 12:165–190.

van Rijsbergen, C. J. (1979). *Information Retrieval*. London: Butterworths.