Un cadre graphique pour la visualisation et la caractérisation de classes en mode non-supervisé

Khalid Benabdeslem*, Haytham Elghazel*
Rakia Jaziri**

*Université Lyon1, LIESP EA 4125
43, Bd du 11 Novembre 1918 Villeurbanne Cedex
{kbenabde, elghazel}@bat710.univ-lyon1.fr
**Université Paris13, LIPN, UMR CNRS 7030
99, Av Jean Baptiste Clément 93430 Villetaneuse
rakia.jaziri@lipn.univ-paris13.fr

Résumé. Dans ce papier, nous proposons un système d'extraction de connaissances à partir de données pour la visualisation et l'interprétation de profils issus d'une classification automatique en mode non supervisé. Une méthode efficace utilisée pour la classification est la carte auto-organisatrice ou Self-Organizing Map (SOM). Dans cette méthode, la détection des regroupements est en général obtenue par d'autres techniques de classification (par partitionnement ou hiérarchisation). Dans le cadre de ce travail, nous explorons une autre voie pour la segmentation de la carte par une approche basée sur la théorie des graphes (la coloration minimale). Enfin, pour caractériser les classes issues de cette segmentation, nous proposons une solution basée sur un test statistique et un arbre couvrant maximum pour la sélection de variables locales à chaque classe permettant ainsi sa caractérisation de manière automatique. Des expérimentations seront présentées sur plusieurs bases de données pour valider l'approche proposée.

1 Introduction

La classification automatique (Clustering) est une étape primordiale dans les problèmes liées à l'apprentissage non supervisé. Elle consiste à détecter la meilleure structure à partir d'une collection de données non étiquetées. Dans ce cadre, son principe revient à organiser les données en groupes (clusters) dits homogènes. Les algorithmes de classification peuvent être regroupés en cinq familles : par partionnement, hiérarchiques, basés sur la densité, basés sur les grilles et ceux basés sur les modèles (Jain et Murty, 1999). Dans ce papier, nous nous basons sur la classification par modèles et plus particulièrement celui des cartes auto-organisatrices de Kohonen (Kohonen, 2001). Connu sous le nom de SOM (Self Organizing Map) ce modèle repose sur un algorithme neuro-inspiré qui, par un processus non-supervisé compétitif, est capable de projeter des données de grandes dimensions dans un espace à faible dimension (en général deux). Ce modèle est populaire et très utilisé pour la visualisation des données dans plusieurs domaines d'application.

Généralement à l'issue d'une classification par SOM, deux questions se posent : Comment optimiser la partition trouvée et comment caractériser les classes obtenues ? Pour répondre à la première question, quelques approches ont été proposées dites à niveaux séquentiels (par partitionnement ou hiérarchiques) (Vesanto et Alhoniemi, 2000) et d'autres dites à niveaux simultanés (Cabanes et Bennani, 2007). Il s'est avéré que l'intégration de la notion de voisinage dans le processus de la segmentation améliore nettement la partition finale (Azzag et Lebbah, 2008). Dans ce cadre nous présentons dans un premier temps une voie alternative basée sur la coloration minimale de graphes pour la segmentation d'une carte topologique. En effet, représentée sous forme d'une grille reliant par des connexions ses différents neurones, une carte SOM peut bénéficier des travaux effectués dans la théorie des graphes pour l'optimisation de la partition finale. Notre premier travail représente donc une extension de la technique de la coloration minimale en se basant sur les relations topologiques offertes par SOM. Les détails peuvent être trouvés dans (Elghazel et al., 2009). Nous avons répondu dans ce dernier papier à deux questions : 1) comment intégrer les informations de voisinage offertes par SOM et 2) quel ordre de voisinage considérer pour délivrer de meilleurs résultats.

Une fois la carte SOM segmentée, reste l'interprétation de ses classes qui est faite en général soit par intervention de l'expert du domaine d'application, soit par des techniques de caractérisation automatique. Dans le deuxième cas, la caractérisation est faite par la sélection de variables locales à chaque classe de la partition. Pour effectuer cette tâche, dite aussi de réduction de dimension, les contributions sont moins conséquentes en apprentissage non supervisé. Dans ce cadre, il existe une panoplie d'approches de caractérisation automatique par pondération de variables. Dans (Frigui et Nasraoui, 2004), les auteurs proposent des algorithmes de pondération locale basée sur une classification floue. Dans (Huang et al., 2005), les auteurs proposent une approche qui minimise la même fonction objective que dans (Frigui et Nasraoui, 2004) mais en proposant une estimation globale des poids des variables. Le mécanisme proposé pour la pondération de variables est ainsi étendu à la classification recouvrante par K-Means flou (Li et Yu, 2006) et à la classification croisée (Parsons et al., 2004). Nous trouvons aussi des méthodes connexionnistes basées sur la pondération globales (Guerif et Bennani, 2007), (Benabdeslem et Lebbah, 2007) et celles basées sur la pondération locales de variables (Grozavu et al., 2009). L'approche de caractérisation que nous proposons dans ce papier, est basée sur une sélection préliminaire faite par un test statistique permettant de détecter des variables dites "Pivot" pour chaque classe de la partition. Ensuite, une approche graphique est proposée et est basée sur l'arbre couvrant maximum (Gondran et Minoux, 1985) pour relier chaque variable "Pivot" aux variables représentant une corrélation multiple et forte est importante pour chacune des classes considérées. Au final, nous aurons un système d'apprentissage connexionniste et graphique permettant d'une part de visualiser la segmentation d'une carte topologique et d'autre part de visualiser les caractéristiques de chaque classe par un graphe.

Le reste de ce papier est organisé comme suit : la section 2 présentera brièvement le modèle des cartes auto-organisatrices. Dans la section 3, nous montrerons comment faire collaborer SOM avec un modèle graphique basé sur la coloration minimale. La section 4 sera consacrée à la définition du test statistique utilisé ainsi qu'à l'arbre couvrant maximum pour la sélection de variables. Enfin, nous terminerons par la présentation des résultats expérimentaux et une conclusion sur notre travail proposé.

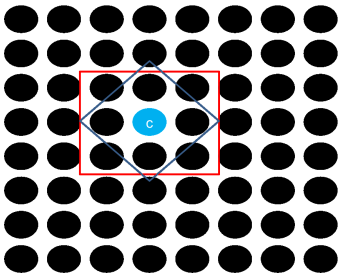

FIG. 1 – *Carte topologique à 2 dimensions avec un voisinage d'ordre 1 autour du neurone c. Voisinage carré (rouge) avec 8 voisins et hexagonal (bleu) avec 4 voisins.*

2 Cartes auto-organisatrices de Kohonen : SOM

Les cartes auto-organisatrices (topologiques ou dites de Kohonen) sont utilisées dans plusieurs domaines d'application pour leur rôle polyvalent (quantification, classification, codage et visualisation) sur les données multidimensionnelles. En effet, outre leur capacité à produire des données similaires au moyen de prototypes comme en quantification vectorielle et/ou en classification, elles autorisent la conservation de la topologie, d'où sa capacité à reproduire des représentations ordonnées, qu'on appelle prototypes, ou vecteurs référents, sur un espace de faible dimension qu'on appelle carte. Cette carte est décrite par un graphe $C(V, W)$ où V est un ensemble de neurones interconnectés. A chacun de ces neurones, est associé un vecteur référent w_c de l'espace des données. W est donc l'ensemble de tous les vecteurs référents : $W = w_1, w_2, ..., w_m$.

L'apprentissage effectué par SOM introduit la conservation de la topologie et impose que deux neurones c, r voisins par rapport à la topologie discrète, soient associés à deux vecteurs w_c et w_r, proches par rapport à la distance choisie entre les données Fig.1.

L'algorithme des SOM est proposé en deux versions : stochastique (en ligne) ou batch (en différé, à la façon des nuées dynamiques). Nous optons dans ce papier pour le deuxième cas pour son déterminisme et sa rapidité bien qu'il faille surveiller l'auto-organisation de la carte. Des comparaisons théoriques entre les deux versions peuvent être trouvées dans (Fort et al., 2001).

L'algorithme considère en entrée un ensemble de n observations $X = z_1, z_2, ..., z_n \in \mathbb{R}^d$ et renvoie en sortie m neurones. A chaque neurone c est associé un ensemble d'observations et un vecteur référent $w_c \in \mathbb{R}^d$.

De manière identique à l'algorithme des nuées dynamiques (Diday, 1971), la version batch des SOM procède en deux phases : affectation et représentation.

Lors de la phase d'affectation, on définit une fonction f de l'espace des données vers la carte C, qui à tout élément z_i associe le neurone dont le vecteur référent est le plus proche de z_i au sens d'une distance généralisée notée d^T qui fait intervenir tous les neurones de la carte :

$$d^T(z_i, w_{f(z_i)}) = \sum_{r \in C} K^T(\delta(r, f(z_i))) \left\| z_i - w_c \right\|^2 \quad (1)$$

Où $K^T(\delta(r, f(z_i)))$ est une fonction de voisinage autour du neurone gagnant issu de $f(z_i)$ et T est le rayon de voisinage qui décroît au cours du temps. $\delta(r, f(z_i))$ est la distance sur la carte entre les deux neurones r et celui issu de $f(z_i)$. En pratique, On utilise souvent $K^T(\delta(c, r)) = e^{-\frac{\delta_{cr}}{2T^2}}$.

La fonction d'affectation, notée f_W est définie comme suit :

$$f_W(z_i) = arg \min_{r \in C} d^T(z_i, w_r) \tag{2}$$

f_W introduit donc une partition $P = \{P_c; c = 1, ..., m\}$ de l'ensemble des observations où chaque partie est définie par $P_c = \{z_i \in X; f(z_i) = c\}$.

Lors de la phase de représentation, l'algorithme met à jour les vecteurs référents de la carte par la formule suivante :

$$w_c^* = \frac{\sum_{r \in C} K^T(\delta_{cr}) Z_r}{\sum_{r \in C} K^T(\delta_{cr}) n_r} \tag{3}$$

Où Z_r représente la somme de toutes les observations qui ont été affectées au neurone r et n_r représente le nombre de ces observations.

Après l'initialisation de la carte par un système de poids initial, l'algorithme procède en réitérant les deux phases d'affectation et représentation jusqu'à stabilisation.

3 Classification des cartes par coloration minimale :McSOM

Nous avons développé récemment une nouvelle voie pour la segmentation des cartes topologiques, habituellement faite par des méthodes de segmentation classiques (par partitions ou hiérarchiques). Il s'agit d'une approche basée sur la théorie des graphes appelée McSOM et qui sera brièvement présentée dans cette section.

Lorsqu'un tableau de dissimilarities $\mathcal{D} = \{\mathcal{D}(w_i, w_j)\}$ est défini sur l'ensemble des neurones $W = \{w_1, w_2, \ldots, w_m\}$ (dans notre cas w_i est un vecteur référent à *d-dimensions* correspondant au neurone i, $w_i = \{w_i^1, w_i^2, \ldots, w_i^d\}$), ces neurones peuvent définir un graphe valué $G(V, E)$ tel que $V = \{v_1, v_2, \ldots, v_m\}$ est l'ensemble des sommets qui sont les neurones à grouper (v_i pour w_i) et $E = V \times V$ est l'ensemble d'arêtes qui correspond, quant à lui, aux paires de sommets (v_i, v_j) pondérés par la dissimilarité $\mathcal{D}(w_i, w_j)$. Une définition simple et claire suppose que "un cluster est un ensemble de neurones similaires, et les éléments de différents clusters sont différents". En conséquence, un cluster devrait satisfaire deux conditions fondamentales : il devrait avoir une homogénéité interne élevée et une hétérogénéité forte entre ses éléments et ceux des autres clusters. Ces deux conditions montrent d'une part, que les arêtes entre deux sommets d'un même cluster devraient avoir de faibles pondérations (indiquant une similarité élevée) et d'autre part, que les sommets de deux clusters différents devraient avoir une forte pondération (indiquant une similarité faible). Le problème de segmentation de la carte avec McSOM est par conséquent ramené à un problème de coloration minimale de graphes.

Cependant, une modélisation brutale de la carte topologique par un graphe complet (non orienté et pondéré) dont les sommets sont les neurones et les arêtes sont les liens pondérés par les dissimilarités, ne convient pas au problème de la classification non supervisée. En effet, la

w_i	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9
w_1	0								
w_2	0.18	0							
w_3	0.42	0.29	0						
w_4	0.21	0.19	0.48	0					
w_5	0.47	0.34	0.54	0.27	0				
w_6	0.49	0.30	0.24	0.44	0.37	0			
w_7	0.75	0.68	0.90	0.53	0.36	0.72	0		
w_8	0.98	0.96	0.99	0.92	0.69	0.94	0.39	0	
w_9	0.80	0.65	0.73	0.62	0.35	0.48	0.43	0.54	0

TAB. 1 – *Matrice de dissimilarités entre les neurones.*

coloration minimale du graphe retournerait la classification "triviale" où chaque cluster (couleur) contient un seul neurone (singleton). Notre algorithme d'optimisation de la carte McSOM passe donc par la construction d'un graphe "seuil-supérieur" défini comme le graphe partiel du graphe de départ. La construction de ce graphe est contrainte par les valeurs de dissimilarités entre les neurones et les relations topologiques (voisinage) entre eux. Ainsi, un graphe seuil-supérieur est défini par $G_{>\theta,\alpha} = (V, E_{>\theta,\alpha})$, où :

- $V = \{v_1, v_2, \ldots, v_m\}$ est l'ensemble de sommets correspondant aux neurones, v_i pour w_i.
- $E_{>\theta,\alpha}$ est l'ensemble d'arêtes, où pour chaque deux sommets v_i et v_j de V, une arête $(v_i, v_j) \in E_{>\theta,\alpha}$ *si* $\mathcal{D}(w_i, w_j) > \theta$ *ou* $\mathcal{SN}(w_i, w_j) > \alpha$. $\mathcal{D}$ est la matrice de dissimilarité et $\mathcal{SN}$ est *la matrice d'ordre de voisinage dans la carte*. Par exemple, deux neurones considérés dans le rectangle rouge dans la figure 2 ont un *ordre de voisinage* égal à 1.
- θ est le seuil de *dissimilarité*.
- α est le seuil de *voisinage*.

Les neurones à grouper sont maintenant représentés par un graphe non complet $G_{>\theta,\alpha} = (V, E_{>\theta,\alpha})$, McSOM consiste par la suite à appliquer un nouvel algorithme de coloration minimale sur ce graphe seuil . Cet algorithme est une amélioration d'un algorithme populaire pour la coloration minimale appelé LF (pour Largest First) (Welsh et Powell, 1967). L'idée principale de McSOM est d'appliquer l'algorithme LF sur le graphe $G_{>\theta,\alpha}$ pour la coloration des neurones de la carte en lui apportant certaines modifications liées aux choix de couleurs pour certains sommets lors du processus de coloration. Ce choix de couleurs est contraint par l'ordre de voisinage entre les neurones dans la carte. Pour plus de détails concernant l'algorithme voir (Elghazel et al., 2009).

Exemple pratique : Soit l'ensemble de neurones $W = \{w_1, w_2, \ldots, w_9\}$ de la carte de la figure 2 relatif à la matrice des dissimilarités $\mathcal{D}$ de la table 1. La Figure 3 montre le graphe seuil-supérieur $G_{>0.48,2}$ ($\theta = 0.48$ et $\alpha = 2$). L'approche McSOM appliquée sur ce graphe génère la coloration dans la figure 4. Ainsi, McSOM induit la partition en trois classes $C_1 = \{w_1, w_2, w_3, w_4\}$, $C_2 = \{w_5, w_6, w_9\}$ and $C_3 = \{w_7, w_8\}$. (chaque couleur correspond à une classe) figure 5.

L'approche McSOM a été validée sur plusieurs bases de données issues de la littérature et a présenté des résultats très encourageants comparée à d'autres méthodes de segmentation.

w_1	w_4	w_7
w_2	w_5	w_8
w_3	w_6	w_9

FIG. 2 – *Carte des neurones relatifs à la matrice de dissimilarités de la table. 1.*

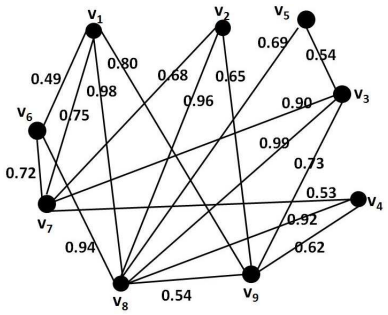

FIG. 3 – *Le graphe seuil-supérieur* $G_{>0.48,2}$ *(*$\theta = 0.48$ *et* $\alpha = 2$*).*

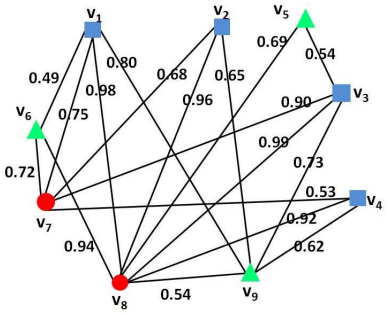

FIG. 4 – *Coloration minimale du graphe* $G_{>0.48,2}$ *par McSOM. Trois couleurs ont été identifiées.*

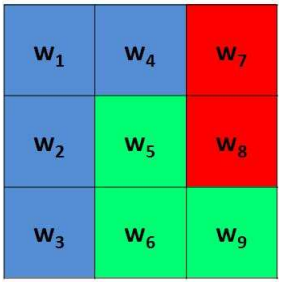

FIG. 5 – *Segmentation de la carte de la figure 2.*

4 Caractérisation de McSOM : G-Select

Nous avons vu dans la section précédente comment segmenter la carte topologique par une méthode basée sur la coloration minimale de graphes (McSOM). A l'issue de cette segmentation, nous obtenons une partition avec un nombre réduit de classes. Nous allons maintenant décrire une approche graphique, que nous appelons G-Select, pour caractériser chacune de ces classes.

4.1 Caractérisation uni-dimensionnelle

Dans cette partie, nous décrivons le test statistique utilisé pour détecter, dans un premier temps, une variable "Pivot" pour chaque classe. Une variable "Pivot" est la variable la plus représentative de la classe, autrement dit, c'est la variable qui contribue le plus pour le regroupement des observations de la classe associée.

Le test statistique pour une variable donnée dans une classe donnée définit une valeur test VT simple mais d'un intérêt pratique indéniable. La formulation que nous utilisons dans cette approche est tirée de (Lebart et al., 2001) qui met en avant VT pour interpréter les classes issues de la classification automatique.

Nous disposons d'un échantillon de taille n. Soit une classe K, produite par McSOM, d'effectif n_K . Bien évidemment, $n_K < n$.

Parmi les variables disponibles, soit j une variable quantitative. Sa moyenne calculée dans l'échantillon global est μ_j, sa variance empirique est égale à σ_j^2 ; la moyenne calculée dans le sous échantillon est μ_{jK}.

La valeur test sur j est définie comme suit :

$$VT_K(j) = \frac{\mu_{jK} - \mu_j}{\sqrt{\frac{n-n_K}{n-1} \times \frac{\sigma_j^2}{n_K}}} \tag{4}$$

La valeur test $VT_K(j)$ peut se lire comme la statistique d'un test de comparaison de moyennes où, sous l' hypothèse nulle de tirage au hasard de n_K parmi n, elle suivrait de manière asymptotique la loi normale centrée réduite. Pour les niveaux de risque usuels (5%), on peut considérer que la différence est significative lorsque la valeur absolue de la $VT_K(j)$ est supérieure à 2. Ce n'est pas évident car dans le cas de la classification, si la variable j a

participé à la constitution des classes, on constate généralement que sa valeur test est exagérée. C'est évident car elle a servi à créer des classes les plus éloignés possibles dans l'espace de représentation, il est normal dans ce cas qu'elle concourt à distinguer ces classes. Dans ce type de processus, on distingue en général les variables actives, qui ont servi au calcul, et les variables illustratives ou supplémentaires, qui sont uniquement mises en avant pour l' interprétation.

En se basant sur une partition issue d'une classification automatique, nous ne disposons pas dans notre analyse, de variables illustratives. Toutes nos variables sont actives. Pour cette raison, nous ne pouvons pas comparer entre les valeurs test de ces variables avec un hypothétique seuil, très difficile à définir dans la pratique. Il convient avant tout de s'en servir comme un critère permettant de hiérarchiser les variables, afin de distinguer la variable "Pivot" qui joue un rôle notoire dans l'interprétation des classes. Par conséquent, la variable "Pivot" p d'une classe K est définie dans la formule suivante :

$$p_K = arg \max_{1 \leq j \leq d} |VT_K(j)| \tag{5}$$

4.2 Arbre couvrant maximum pour la sélection de variables

Dans cette section, nous allons montrer comment détecter automatiquement les variables formant une corrélation multiple forte et qui sont liées à la variable pivot décrite ci-dessus. Nous proposons donc d'utiliser la technique de l'arbre couvrant maximum qui permet de déterminer cette corrélation. Cette technique nécessite une matrice dite de poids entre les variables. Nous calculons donc la matrice de corrélations pour chaque classe K, tel que la correlation entre deux variables j et j' est définie comme suit :

$$corr(j, j') = \frac{1}{\sigma_j \sigma_{j'}} \times \frac{1}{n_K} \sum_{k=1} (z_j^k - \mu_j)(z_{j'}^k - \mu_{j'}) \tag{6}$$

Pour chaque classe K, nous définissons un graphe valué $G_K(D, CORR)$ tel que D est l'ensemble des variables décrivant les données d'apprentissage et CORR est l'ensemble des arêtes étiquetées selon la formule (6).

un arbre couvrant maximum, noté G', est un sous graphe de G, qui doit être connexe, sans cycle et dont la somme des longueurs des arêtes est maximale. Cette méthode graphique permet ainsi de détecter les variables directement liées à la variable pivot sans être obligé de fixer un seuil (Fig. 6).

Dans la figure 6, à gauche, est présenté le graphe complet G reliant toutes les variables d'une classe donnée telle que une arête reliant deux variables différentes est valuée par la corrélation entre elles. Dans la partie droite, l'arbre couvrant maximum G' dans lequel on distingue la variable pivot (rond bleu) , les variables fortement liées à la variable pivot (rond noir) et les arêtes contribuant à la construction de l'arbre (trais épais).

Pour construire l'arbre couvrant maximum, nous utilisons l'algorithme optimisé de Prim :

Initialisation : Soient V et E deux ensembles vides. On affecte à V un élément quelconque de D :

- Trouver l'arête (j, j') de $\overline{V} \times V$ de longueur maximum ($\overline{V} = (D - V)$).
- Mettre alors j dans V et l'arête (j, j') dans E.

Après $d - 1$ itérations, le graphe (D, E) est un arbre maximum.

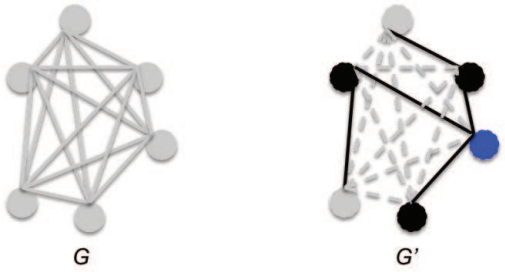

FIG. 6 – *G : Graphe original ; G′ :Arbre couvrant maximum*

En fait, Prim propose une procédure de marquage de mise à jour réduisant la complexité de cet algorithme. Le marquage est basé sur la propriété suivante :

Soit j_k le sommet sélectionné dans $\overline{V}$ à l'étape k et affecté à V. On note alors $V^* = V \cup \{j_k\}$ et $\overline{V}^*$=$\overline{V} - \{j_k\}$. A l'étape $k+1$ on cherche donc :

$$\max_{(j,j')\in \overline{V^*}V^*} corr(j,j') = \max_{j\in\overline{V^*}} \max_{j'\in V^*} corr(j,j') \tag{7}$$

La procédure de marquage consiste, à l'étape k, après avoir sélectionné j_k, à conserver, pour chaque $j \in \overline{V^*}$ les valeurs $\max_{j'\in V^*} corr(j,j')$.

Ces valeurs sont calculées par la formule de mise à jour suivante :

$$\max_{j'\in V^*} corr(j,j') = max(\max_{j'\in V} corr(j,j') corr(j_k,j)) \tag{8}$$

Ainsi, à l'étape k on mémorise pour tout $j \in \overline{V^*}$ le plus éloigné de j :

$$Pred(j) = arg \max_{j'\in V*} corr(j,j') \tag{9}$$

$L(j)$ est la longueur entre j et $Pred(j)$ donc :

$$L(j) = \max_{j'\in V^*} corr(j,j') \tag{10}$$

La construction de l'arbre de longueur maximum va se faire à partir de deux tableaux $Pred$ et L de dimension d et de la matrice de corrélation CORR de dimension $d \times d$. A chaque itération on cherche le sommet Max , tel que $L(Max)$ est maximum (Max correspond au sommet j_k sélectionné dans $\overline{V}$). Puis on pose $L(Max) = -1$ (on note ainsi que le sommet Max appartient à $\overline{V^*}$). Puis pour tous les sommets j tels que $L(j) = -1$ (les sommets appartiennent à $\overline{V^*}$) si $D(j, Max) > L(j)$ alors $L(j) = D(j, Max)$, i.e. est mis à jour et $Pred(j) = Max$. L'algorithme Prim est présenté dans Algorithm.1

Après les $d-1$ itérations, les $d-1$ arêtes sont $(j, Pred(j))$. Cet algorithme est de complexité $O(d^2)$.

Algorithm 1 . Prim

$Pred(d) := 0$
$Pour\ j := 1\ à\ d-1$ Faire
$L(j) := CORR(j,d)$
$Pred := d$
$Pour\ k := 1\ à\ d-1\ Faire$
$Lmax := -1$
$Pour\ j := 1\ à\ d-1\ Faire$
$Si\ L(j) > Lmax\ Alors\ Lmax := L(j)\ ;\ Max = j$
$L(Max) := -1$
$Pour\ j := 1\ à\ d-1\ tel\ que\ L(j) \neq$-1 $Faire$
$Si\ D(j,Max) > Alors\ L(j) := D(j,Max)\ ;\ Pred(j) := Max$

Bases	n	d	#classes
Wave	5000	40	3
Madelon	2000	500	2
Wdbc	569	30	2
Spamb	4601	57	2

TAB. 2 – *Caractéristiques des bases d'apprentissage*

5 Expérimentations

5.1 Bases d'apprentissage

Dans cette section, nous allons discuter notre approche sur plusieurs bases de données issues de la base UCI (Blake et Merz, 1998). Les bases sont volontairement choisies pour pouvoir comparer nos résultats avec d'autres approches issues de l'état de l'art utilisant ces mêmes bases (table. 2).

Nous détaillons ces bases d'apprentissage comme suit :

– *Wave* : connue sous le nom des "Vagues de Breiman bruitées". La base est composée de 5000 exemples divisés en 3 classes. La base originale comportait 21 variables, mais 19 variables additionnelles distribuées selon une loi normale ont été rajoutées sous forme de bruit. Chaque observation a été générée comme une combinaison de 2 sur 3 vagues.

– *Wdbc* (Wisconsin Diagnostic Breast Cancer) : ce jeu de données contient 569 individus qui sont décrits par 30 variables. 357 individus sont atteints d'un cancer bénin et les 212 d'un cancer malin. Les variables décrivent les caractéristiques des noyaux de cellules présentes dans l'image numériques.

– *Madelon* : Ces données posent un problème à 2 classes proposé à l'origine pendant la compétition sur la sélection de variables organisée lors de la conférence NIPS'2003, (Guyon et al., 2006). Les exemples sont situés sur les sommets d'un hypercube en dimension 5, mais 15 variables redondantes et 480 dimensions bruitées ont été rajoutés.

– *Spamb* : est un jeu de données composé de 4601 observations décrites par 57 variables, chacune décrivant un mail et sa catégorie : spam ou non-spam. Les attributs descriptifs de ces

mails sont les fréquences d'apparition de certains mots ou caractères ainsi que des informations sur la quantité de caractères mis en capitale.

5.2 Critères d'évaluation

Les expérimentations que nous présentons dans ce travail se font à deux niveaux. Nous allons, dans un premier temps, montrer l'évaluation de notre approche sur la qualité de la segmentation faite par McSOM. Pour cette évaluation, nous allons utiliser deux indices de qualité externes (*Pureté* et Indice de *Rand*). Ensuite, nous utiliserons ces deux indices pour évaluer l'approche G-Select sur la caractérisation des classes issues de McSOM i.e. sur la sélection de variables à chaque classe de la partition finale.

Étant donné un ensemble d'entrées $\Omega = \{z_1, z_2, ..., z_n\}$ regroupées dans une partition $P_k = \{C_1, C_2, ..., C_K\}$ (Nous rappelons qu'ici l'ensemble d'entrées est celui des référents issus de SOM : W, donc $n = m\}$. $\forall C_i, C_j \in P_K, C_j \cap C_j = \oslash$ pour $i \neq j$ et $\sum \eta_i = m$ tel que η_i est la cardinalité de la classe C_i.

– *Pureté* : c'est une mesure d'évaluation simple qui calcule un pourcentage global d'adéquation entre la segmentation proposée et une segmentation souhaitée. La meilleure partition est celle qui maximise la valeur :

$$Pureté(P_K) = \frac{1}{n} \sum_{i=1}^{K} \underset{j \in \{1,2,...,K'\}}{max} |C_i \cap L_j| \tag{11}$$

L_j est une des classes de la partition souhaitée. K' est le nombre total des classes.

– *Rand* : cet indice calcule le pourcentage du nombre de couples d'observations ayant la même classe et se retrouvant dans le même sous ensemble après segmentation de la carte. La meilleure partition est celle qui maximise la valeur :

$$Rand(P_K) = \frac{a+b}{a+b+c+d} \tag{12}$$

a est le nombre de paires d'observations dont les deux éléments sont dans la même classe souhaitée et la même classe proposée. b est le nombre de pairs d'observations dont les deux éléments sont dans la même classe souhaitée mais pas la même classe proposée, alors que c est le nombre d'observations dont les deux éléments sont dans la même classe proposée mais pas la même classe souhaitée. Enfin, d est le nombre de pairs d'observations dont les deux éléments ne sont ni dans la même classe proposée, ni dans la même classe souhaitée.

5.3 Segmentation et visualisation par G-Select

Tout d'abord, nous donnons quelques détails sur les différents paramètres pour toutes les bases utilisées. Pour la construction des cartes, nous avons utilisé l'heuristique de Kohonen qui détermine automatiquement les dimensions des cartes. Cette heuristique se base sur l'analyse en composante principale. Son application a donc donné pour la base "Wave", Dimension= 26×14, pour Wdbc, Dimension=30×4, pour Madelon, Dimension=17×13 et pour Spamebase, Dimension=38×9. En ce qui concerne l'initialisation des poids, elle a été faite par l'enveloppe convexe des données. Par ailleurs , pour chaque ordre de voisinage α (entre 1 est la dimension de la carte), McSOM est exécutée plusieurs fois et chaque exécution accrois le

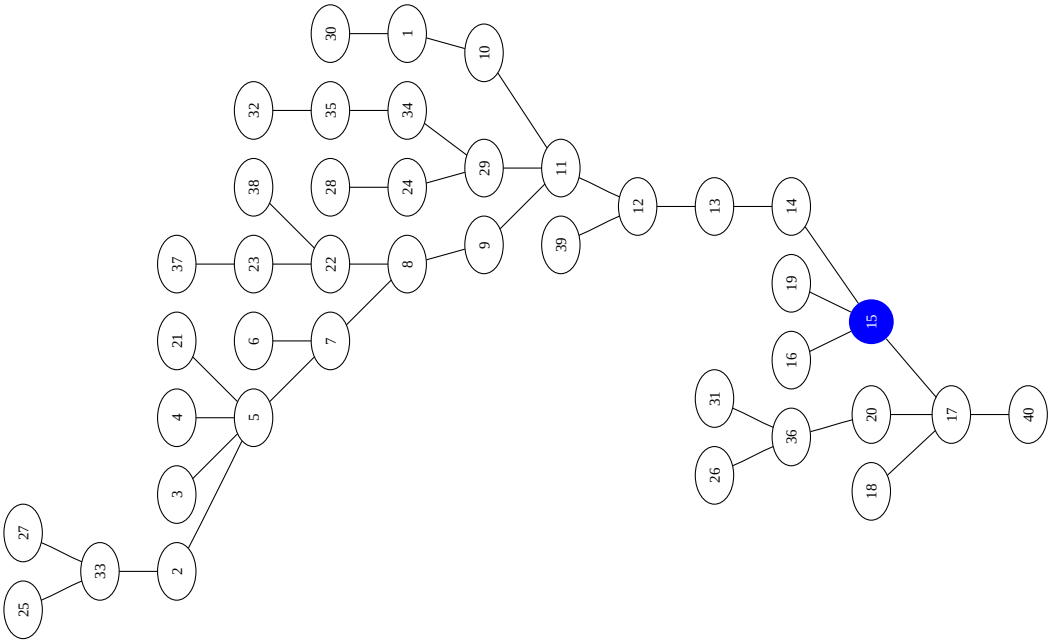

FIG. 7 – *Variables sélectionnées dans Classe1 pour la base Wave*

seuil de similarité θ. Pour chaque seuil de voisinage et de dissimilarité, l'algorithme fournit la partition optimale qui maximise l'indice de Dunn généralisé, (Kalyani et Sushmita, 2003).

Nous allons maintenant évaluer les résultats de notre approche sur toutes les bases d'apprentissage en utilisant les indices de qualité : $Rand$ et $Pureté$.

Nous donnons une attention particulière à la base *Wave*, car nous connaissons a priori les variables pertinentes et les variables bruitées. Ceci, pour voir si G-Select arrive à caractériser les classes de la partition par les bonnes variables. Nous signalons qu'à l'issue de McSOM, nous avons obtenu une segmentation en 3 classes de manière automatique avec une pureté de 0.55 et un Rand égal à 0.6706 . Ce qui correspond au nombre de classes réel défini dans la base initiale. Nous appliquons donc, G-Select sur chaque classe et nous obtenons les variables caractéristiques avec une visualisation de l'arbre couvrant maximum qui a permis de les déterminer.

Nous pouvons facilement remarquer dans les trois figures 7, 8 et 9 les variables qui caractérisent les trois classes de Brieman (*Classe1, Classe2, Classe3*), respectivement. Pour chacune de ces classes, G-Select détermine la variable pivot représentée par un rond bleu. Ainsi, cette variable est directement liée avec un ensemble de variables qui sont sélectionnées pour caractériser la classe. Les variables [14, 15, 16, 17, 19] pour *Classe1*, [2, 6, 7, 8, 30] pour *Classe2* et [1, 9, 10, 11, 12] pour *Classe3*. Nous remarquons donc que G-Select permet la sélection de 14 variables pertinentes pour la caractérisation plus une seule variable bruit. Néanmoins, la méthode affiche des taux de *Pureté* et *Rand* intéressants : 0.5680 et 0.6713, respectivement. Ces taux sont meilleurs que ceux obtenus sans sélection et sont également meilleurs que ceux obtenus par deux méthodes de caractérisation par pondération de variables dans (Grozavu et al.,

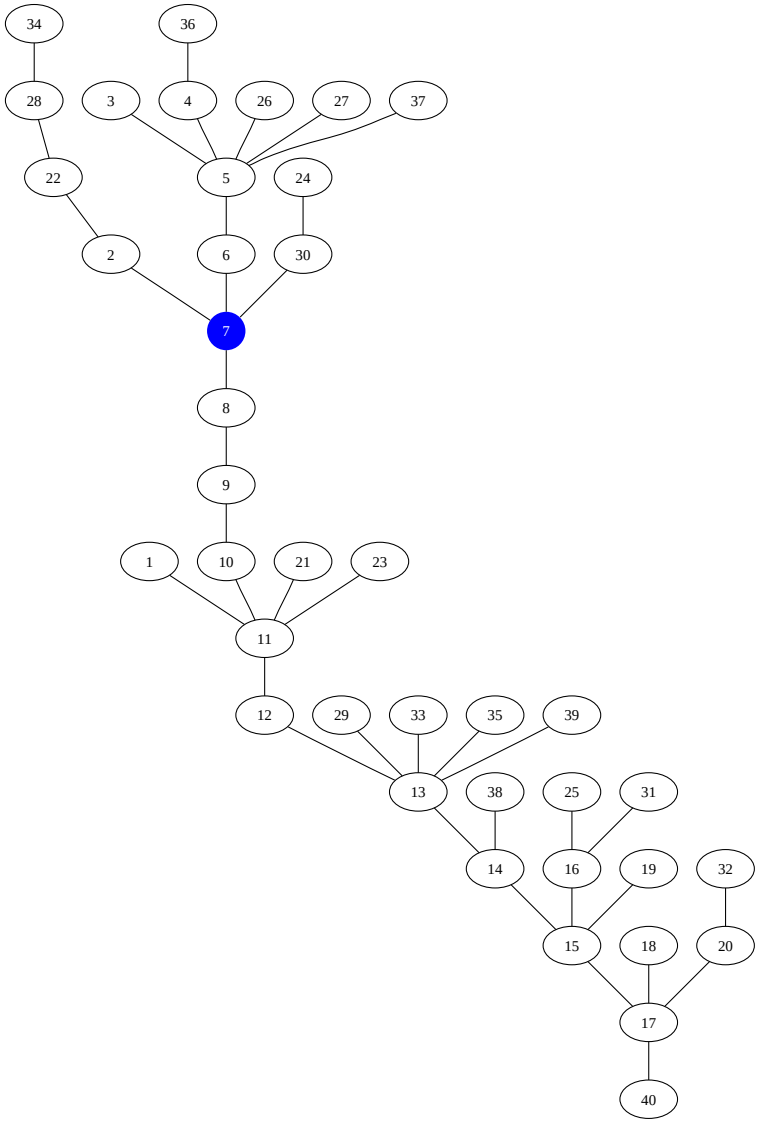

FIG. 8 – *Variables sélectionnées dans Classe2 pour la base Wave*

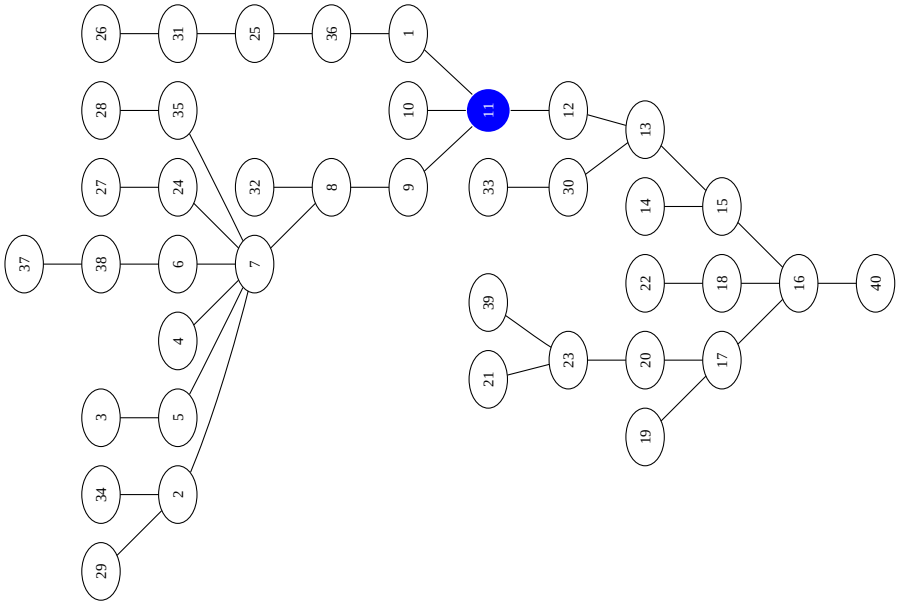

FIG. 9 – *Variables sélectionnées dans Classe3 pour la base Wave*

2009) $lwd - SOM$ et $lwo - SOM$ avec un nombre de variables réduit cf Table.3.

Nous avons ré-appliqué McSOM sur la nouvelle base de *Wave* avec les variables sélectionnées par G_Select pour visualiser la nouvelle segmentation (Figure. 10). Nous pouvons facilement remarquer une nette amélioration par rapport à la segmentation obtenue avant la sélection de variables. La structure des vagues de Brieman est mieux représentée avec une bonne auto-organisation de la carte.

Nous présentons dans la table.4 une comparaison de la qualité de la segmentation avant et après la sélection de variables sur les autres bases d'apprentissage.

Nous présentons dans la table.5 les résultats de notre approche en comparaison avec lwd-SOM et lwo_SOM sur les mêmes bases d'apprentissage.

	lwd-SOM	lwo-SOM	Classification croisée	G-Select
Classes :[Variables]	cl_1 :[6-15] cl_2 :[4-10] cl_3 :[7-19]	cl_1 :[3-8,11-16] cl_2 :[8-11,14-19] cl_3 :[3-20]	cl_1 :[3-12] cl_2 :[7-15] cl_3 :[10-18] cl_4 :[5-17]	cl_1 :[14-17,19] cl_2 :[2,6-8,30] cl_3 :[1,9-12]
Pureté	0.5374	0.5416	NC	0.5680
Rand	0.6068	0.6164	NC	0.6713

TAB. 3 – *Comparaisons entre plusieurs méthodes de caractérisation sur la base Wave. NC : Non communiqué*

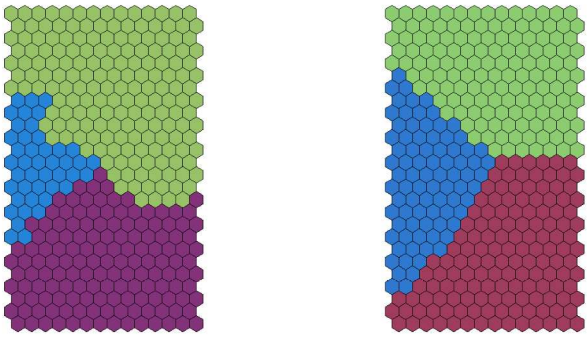

FIG. 10 – *Segmentation sur la base Wave obtenue par McSOM avant (à gauche) et après (à droite) sélection de variables faite par G-Select*

Bases	Avant	Sélection	Après	Sélection
	Rand	Pureté	Rand	Pureté
Wdbc	0.6769	0.7979	0.8843	0.9385
Madelon	0.5085	0.5825	0.5094	0.5945
Spamb	0.5195	0.6059	0.5197	0.6124

TAB. 4 – *Résultats de la segmentation avant et après sélection de variables*

Bases	#cl réel	lwd-SOM	lwo-SOM	Classification croisée	G-Select
Wdbc	2	$cl_1 - cl_9$: [4,24] <0.6274>	$cl_1 - cl_9$: [4,24] <0.8682>	$cl_1 - cl_5$: [4,24] < NC >	$cl1$:[7,8,23,28] ; cl_2 :[3,7,8,28] <0.9385>
Madelon	2	cl_1 :1 ; cl_2 : [91, 281, 403-424] <0.5242>	cl_1 :1 ; cl_2 :[242, 417-452] <0.5347>	cl_1 : [445-450] ; cl_2 : [450-460] < NC >	$cl_1 - cl_4$: [*] <0.5945>
Spamb	2	cl_1 :56 ; cl_2 :57 <0.6103>	cl_1 :56 ; cl_2 :57 <0.6413>	cl_1 :56 ; cl_2 :57 < NC >	cl_1 :[26,32,34,55] cl_2 :[27,29,31,32, 34,40,43] <0.6124>

TAB. 5 – *Sélection de variables pertinentes par classe pour les bases : Madelon, Wdbc, Spamb. [Variables] ; <Pureté> ; [*] : [49 65 106 129 242 339 344 356 443 454 476 494] ; NC : Non communiqué*

Nous pouvons constater à partir des tables 3, 4, 5 et la figure 10 que l'approche que nous proposons fournit :

- Une segmentation avec un nombre de classes égal ou proche à celui défini dans la segmentation souhaitée, surtout pour un nombre de classe réduit. En effet, McSOM a tendance à minimiser le nombre de classes dans la segmentation.
- Une meilleure segmentation de la carte avec McSOM comparée à celle obtenue à plusieurs autres méthodes de segmentation avec une meilleure qualité sur des indices de qualité internes et externes. Des résultats plus détaillés peuvent être trouvés dans (Elghazel et al., 2009).
- Une caractérisation automatique des classes issues de la partition, sélectionnant le maximum de variables pertinentes.
- Une élimination de bruit considérable.
- Une amélioration de la segmentation de la carte après la sélection de variables par G-Select.
- De meilleurs taux de *Pureté* et *Rand* après sélection, comparés à ceux d'autres méthodes de caractérisation automatique.

6 Conclusion

Nous avons proposé dans ce papier un "framework" basé sur des approches graphiques pour visualiser d'une part, la segmentation des cartes topologiques et d'autre part, les caractéristiques des classes obtenues dans la partition finale. L'approche est basée sur un test statistique pour détecter la variable la plus importante dans chacune des classes. A partir de cette variable, dite "Pivot", un arbre couvrant maximum est construit pour représenter la corrélation la plus forte entre les variables de la base d'apprentissage. Finalement, l'ensemble constituée de la variable Pivot avec les variables qui lui sont directement liées, est sélectionné pour caractériser la classe associée. Grâce à cette approche, nous avons amélioré la qualité de la segmentation au niveau de la visualisation et de la caractérisation par sélection de variables locales. Plusieurs perspectives peuvent être envisagées à l'issu de ce travail, par exemple, la construction de graphes à partir d'un ensemble de variables "pivot" et l'optimisation de l'approche pour le passage à l'échelle.

Références

Azzag, H. et M. Lebbah (2008). Clustering of self organizing map. In *European Symposium on Artificial Neural Networks*, pp. 209–214.

Benabdeslem, K. et M. Lebbah (2007). Feature selection for self organizing map. In *IMAC/IEEE International conference on information technology interface*, pp. 45–50.

Blake, C. et C. Merz (1998). *UCI repository of machine learning databases*.

Cabanes, G. et Y. Bennani (2007). A simultaneous two-level clustering algorithm for automatic model selection. In *International Conference on Machine Learning and Applications*, pp. 316–321.

Diday, E. (1971). La méthode des nuées dynamiques. *Revue statistique appliquée 2*, 19–34.

Elghazel, H., K. Benabdeslem, et H. Kheddouci (2009). Mcsom: Minimal coloring of self organizing map. In *Advanced Data Mining and Applications*, pp. 128–139.

Fort, J., P. Letrémy, et M. Cottrel (2001). Stochastic on-line algorithm versus batch algorithm for quantization for quantification and sel organizing map. In *Second NNSP Conference*.

Frigui, H. et O. Nasraoui (2004). Unsupervised learning of prototypes and attribute weights. *Pattern recognition 37(3)*, 567–581.

Gondran, M. et M. Minoux (1985). *Graphes et algorithmes*. Eyrolles.

Grozavu, N., Y. Bennani, et M. Lebbah (2009). From variable weighting to cluster characterization in topographic unsupervised learning. In *IEEE International joint conference on neural network*.

Guerif, S. et Y. Bennani (2007). Dimensionality reduction trough unsupervised feature selection. In *International conference on engineering applications of Neural Networks*, pp. 98–106.

Guyon, I., S. Gunn, et M. Nikravesh (2006). *FE, Found. and Appl.*

Huang, J., M. Ng, H. Rong, et Z. Li (2005). Automated variable weighting. *IEEE Trans Pattern Anal. Mach. Intell 27*, 657–668.

Jain, A. et M. Murty (1999). Data clustering: Areview. *ACM Computing Surveys 31*, 264–323.

Kalyani, M. et M. Sushmita (2003). Clustering and its validation in a symbolic framework. *Pattern Recognition Letters 24(14)*, 2367–2376.

Kohonen, T. (2001). *Self organizing Map*. Berlin: Springer Verlag.

Lebart, L., A. Morineau, et M. Piron (2001). *Statistique exploratoire multidimensionnelle*. Dunod.

Li, C. et J. Yu (2006). A novel fuzzy c-means clustering algorithm. *Lecture Notes in Computer Science 4062*, 510–515.

Parsons, L., E. Haque, et H. Liu (2004). Subspace clustering for high dimensional data: a review. *SIGKDD 6(1)*, 90–105.

Vesanto, J. et E. Alhoniemi (2000). Clustering of the self organizing map. *IEEE Transactions on Neural Networks 11(3)*, 586–600.

Welsh, D. et M. Powell (1967). An upper bound for chromatic number of a graph and its application to timetabling problems. *Computer journal 10(1)*, 85–87.

Summary

In this paper, we propose a knowledge discovery system for visualizing and categorizing obtained clusters from unsupervised learning. Self-Organizing Map (SOM) is one popular method for clustering and visualizing high dimensional data. In general, this method is succeeded by another clustering methods (partitional or hierarchical) for optimizing the final partition. In this work, we explore a recent developed method based on minimal coloring of graphs . Then, for automatically categorizing the classes we propose a new approach combining a statistical test with a maximum spanning tree for local features selection in each class. Experimental results will be given over several data bases for validating our approach.