Organisation d'images par des cartes de Kohonen sur données de dissimilitude

Tien Ho-Phuoc[1], Anne Guérin-Dugué[1]

[1]GIPSA-lab, Département des Images et des Signaux,

INPG, 46 avenue Félix Viallet

38031 Grenoble, France

{ho ; anne.guerin}@gipsa-lab.inpg.fr

Résumé – Avec l'émergence d'applications en bioinformatique et en « data mining » il y a un intérêt croissant pour développer de nouvelles méthodes efficaces d'analyse de données de dissimilitude. L'algorithme « SOM » appliquée aux données de dissimilitude en est une. Des extensions de « SOM » aux données de dissimilitude (« DSOM ») ont déjà été proposées. Leur analyse révèle une difficulté majeure dans le versant « quantification » de l'algorithme où l'absence de modèles de données sous-jacent à chaque sous-partition pénalise grandement la qualité de l'organisation obtenue. Pour contrer cette difficulté nous proposons une nouvelle spécification de l'algorithme dans la phase de représentation de la partition obtenue. L'originalité réside dans une nouvelle métrique sous des hypothèses euclidiennes, pour la quantification sans avoir à calculer explicitement les prototypes. Grâce à cette nouvelle métrique qui intègre uniquement une référence implicite aux prototypes de quantification, la qualité de l'algorithme pour la préservation en sortie des voisinages est grandement améliorée. Les performances de l'algorithme proposé sont supérieures à celles des algorithmes similaires existants.

Abstract – Adaptation of the Self-Organizing Map to dissimilarity data is of a growing interest. For many applications, vector representation is not available and but only proximity data (distance, dissimilarity, similarity, ranks ...). In this article, we present a new adaptation of the SOM algorithm which is compared with two existing ones. Three metrics for quality estimate (quantization and neighbourhood) are used for comparison. Numerical experiments on artificial and real data show the algorithm quality. The strong point of the proposed algorithm comes from a more accurate prototype estimate which is one of the difficult parts of Dissimilarity SOM algorithms (DSOM).

1. Introduction

L'adaptation des cartes auto-organisatrices de Kohonen [1] aux données de dissimilitude a fait l'objet de nombreux travaux durant cette dernière décennie. Cela s'explique par deux raisons principales. La première vient de la force et de la simplicité des cartes auto-organisatrices (« SOM ») pour l'exploration de données complexes en grande dimension à des fins de visualisation et ou de classification. En deuxième lieu, on observe un nombre croissant de recherche en fouille de données appliquées aux documents, site web…, ou en bio-informatique sur les structures génomiques, où dans ce cas, il y a profusion de données, sous forme de proximité au sens large, en comparant par exemple, 2 observations par la distance de Levenshtein [2]. Egalement dans d'autres domaines d'application, les données ne sont pas toujours disponibles sous forme vectorielle, mais sous forme de proximité (distances, dissimilarités, similarités, rangs, …), comme par exemple en psychologie ou en analyse des signaux et des images.

Dans cet article, nous allons présenter un nouvel algorithme d'extension des cartes de Kohonen aux données de dissimilitudes (« DSOM »). Pour valider cet algorithme, nous présenterons des résultats sur des données réelles d'images en comparant avec deux algorithmes existants.

2. Principe des algorithmes SOM sur dissimilitudes

Parmi les extensions de l'algorithme SOM (« Self-Organizing Map »), nous nous focalisons sur celles développées à partir de l'algorithme initial proposé par Kohonen, dans sa version d'apprentissage en « Batch ». Cette dernière correspond à un algorithme des « K-moyennes » avec prise en compte des relations de voisinage sur la carte de sortie [1]. Considérons une carte avec C unités interconnectées. Chaque unité a une position sur la carte et une position dans l'espace d'entrée par le prototype ω_c qui la représente. Au départ, le prototype ω_c ($c=1..C$) de chaque unité est initialisé aléatoirement par un

vecteur de données dans l'espace de départ. Puis, à chaque itération et jusqu'à convergence, l'algorithme enchaîne une étape d'affectation (partition de l'ensemble des données initiales en C sous-ensembles par assignation au prototype le plus proches dans l'espace d'entrée), puis une étape de représentation (mise à jour des C prototypes comme les centres de gravité de chaque sous-ensemble en pondérant suivant les relations de voisinage sur la carte en sortie). La fonction de voisinage en sortie est mise à jour à chaque itération en réduisant progressivement son rayon T pour tendre à la convergence à un algorithme de type « K-moyennes ».

Si cet algorithme est appliqué à des données de dissimilitude, la phase la plus problématique à spécifier est celle de représentation, puisqu'il n'y a pas de représentation vectorielle des données. Et plus généralement avec les données de dissimilitude, il n'y a pas de modèle de données pour chaque partition. Pour contourner cette difficulté, les approches proposées dans [3, 4, 5] recherchent l'observation qui minimise la fonction de coût de distorsion sur la carte, sachant que cette observation sera contrainte à appartenir aux observations d'entrée. Ainsi la phase de représentation qui était directe dans la version vectorielle, se transforme dans la version « dissimilitude » (DSOM), par une optimisation coûteuse en temps en calcul où il faut évaluer chaque observation candidate et cela pour tous les prototypes. De plus et non le moindre, cette quantification des prototypes conduit à un risque potentiel (et très souvent observé en pratique) qu'une même observation puisse être prototype pour des unités différentes sur la carte. Ces collisions entraînent une ambiguïté dans la phase d'affectation suivante et donc une forte distorsion. Pour certaines applications, il est possible d'atteindre des prototypes hors de l'ensemble d'entrée, comme par exemple pour l'organisation, la visualisation et classification de chaînes de symbole [6] mais cela aura un coût calculatoire très élevé [7].

Dans l'algorithme que nous proposons, nous avons résolu la difficulté de l'assignation du prototype en maintenant le coût calculatoire. Pour ce faire, nous avons défini une mesure de proximité entre chaque observation et le centre de gravité de la partition, sans que ce centre soit trouvé explicitement. La référence étant implicite, chaque partition aura de fait un prototype différent et il n'y aura plus de collision. Pour cela, il est nécessaire de définir des hypothèses d'existence d'un espace sous-jacent euclidien. Ainsi, si les données de proximité en entrée sont exactement des distances euclidiennes, l'algorithme que nous proposons aura exactement le même comportement que la version SOM vectorielle. Ce qui n'est pas le cas pour les versions DSOM actuelles

3. Description de l'algorithme DSOM proposé

La mesure de proximité que nous proposons pour assigner une observation o_i au prototype le plus proche ω_c* se déduit du théorème de Huyghens de décomposition de l'inertie.

Considérons en toute généralité, un ensemble $X=\{x_i, i=1..N, x_i \in \Re^p\}$, de centre de gravité g et d'inertie $I(X)$. Alors l'inertie des points de X vis-à-vis à d'un point e quelconque se décompose ainsi :

$$I(X,e) = I(X) + d^2(g,e) \quad (1)$$

En conséquence, sous les hypothèses de distances euclidiennes, la distance d'un point e quelconque au centre de gravité g peut se calculer *sans connaître explicitement la position de ce centre de gravité*:

$$d^2(g,e) = \frac{1}{N}\sum_{x_i \in X} d^2(e,x_i) - \frac{1}{N^2}\sum_{x_i \in X}\sum_{x_j \in X, j>i} d^2(x_i,x_j) \quad (2)$$

Cette relation est appliquée au contexte de l'algorithme DSOM pour calculer pour chaque observation o_i, sa proximité à une unité sur la carte représentant une partition ω_c. Comme cette proximité va dépendre du paramètre T de voisinage, elle sera notée $D^T(o_i, \omega_c)$. Les données initiales proviennent d'un ensemble X de N observations $X=\{o_i, i=1..N\}$, dont on connaît pour chaque paire, d(o_i, o_j) la distance ou plus généralement la dissimilitude. Nous faisons l'hypothèse de l'existence d'un sous espace vectoriel euclidien sous-jacent et local à chaque sous-ensemble des partitions obtenues.

Durant l'étape d'affectation de l'algorithme, toutes les observations de l'ensemble X sont partitionnées en C sous-ensembles X_c, ($c=1..C$). Pour chaque observation, le prototype le plus proche est assigné ω_c*, tel que :

$$c^* = f(i) = Arg\left[\underset{c}{Min}\left(D^T\left(o_i,\omega_c\right)\right)\right] \quad (3)$$

La phase de représentation, qui a pour but de mettre à jour le prototype comme centre de gravité de chaque sous-ensemble pondéré par les voisinages en sortie, effectuera ici simplement la mise à jour de la distance $D^T(o_i, \omega_c)$ faisant implicitement référence au centre de gravité :

$$D^T(o_i,\omega_c) = I(X_c,\omega_c) - I(X_c) \quad (4)$$

$$D^T(o_i,\omega_c) = \sum_{o_j \in X} m_{j/c} d^2(o_i,o_j) - \sum_{o_i \in X}\sum_{o_j \in X, j>i} m_{i/c}.m_{j/c} d^2(o_i,o_j)$$

$$m_{j/c} = \frac{h^T\left(\delta(c,f(j))\right)}{\sum_{l=1}^{N} h^T\left(\delta(c,f(l))\right)}, \sum_{j=1}^{N} m_{j/c} = 1 \quad (5)$$

La fonction de voisinage en sortie est notée $h^T(.)$, avec T le rayon de voisinage décroissant au fil des itérations. Les distances sur la carte entre 2 unités quelconques k et l sont notées $\delta(k,l)$. Durant la phase d'affectation suivante, une nouvelle partition est réalisée, et ainsi de suite.

A la fin de l'algorithme, si pour des raisons de visualisation, il est nécessaire d'obtenir une instance de chaque prototype ω_c, cela sera alors possible en prenant par exemple l'observation qui minimise $I(X_c)$.

4. Expérimentations

Pour des raisons de place, nous présentons uniquement les résultats expérimentaux obtenus sur une base d'images segmentées puis binarisées, pour la visualisation et la classification de silhouettes de parties de poulet (« Chicken Silhouette »). Cette base disponible sur le web [8] est composée de 446 observations réparties en 5 classes.

TAB. 1 : Répartition des observations en classe

Classe	Nom	Effectif
1	« Wing »	117
2	« Back »	76
3	« Drumstick »	96
4	« Thigh and Back »	61
5	« Breast »	96
Total		446

Les observations sont décrites via les distances de Levenshtein entre les séquences d'orientation des segments qui définissent le contour fermé des images. Nous avons comparé les résultats obtenus avec trois implémentations d'algorithmes DSOM, celui proposé par Kohonen [3] (DSOM(K)), par A. ElGolli [4] (DSOM(EG)) et le notre (DSOM). Pour les trois implémentations, une carte de 7 × 7 unités avec un voisinage hexagonal a été choisie. A l'issue de l'initialisation, les trois algorithmes débutent avec les mêmes partitions aléatoires. Trois métriques sont utilisées pour la comparaison : la qualité de la projection par l'erreur de quantification globale (E_q), et la qualité de la préservation du voisinage ($M_1(k)$, $M_2(k)$). Ces deux derniers indices normalisés entre 0 et 1, sont proposés par Venna et Kaska [9] pour évaluer la « fidélité des voisinages ». Pour un nombre k donné de premiers voisins, $M_1(k)$ est d'autant plus petit les voisinages en sortie comportent des observations qui ne sont pas dans les k premiers voisins en entrée. La « continuité » $M_2(k)$ est la propriété réciproque de $M_1(k)$). $M_1(k)$ est ainsi défini, (et $M_2(k)$ de façon très similaire, en remplaçant $U_k(o_i)$ par $V_k(o_i)$ et $r(o_i,o_j)$ par $\hat{r}(o_i,o_j)$) :

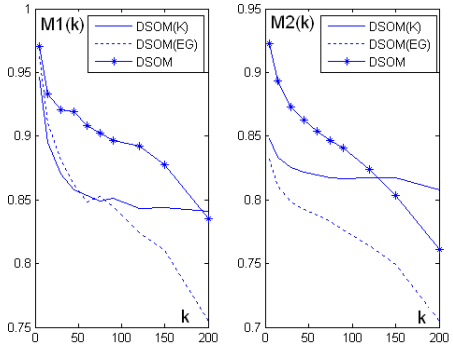
$$M_1(k) = 1 - \frac{2}{Nk(2n-3k-1)} \sum_{i=1}^{N} \sum_{o_j \in U_k(o_i)} \left(r(o_i,o_j) - k \right), \quad (6)$$

avec $C_k(o_i)$ l'ensemble des k premiers voisins en entrée de o_i, $\hat{C}_k(o_i)$ l'ensemble des k premiers voisins en sortie de o_i, $U_k(o_i)$ l'ensemble des observations telles que $o_i \in \hat{C}_k(o_i) \wedge o_i \notin C_k(o_i)$, $V_k(o_i)$ l'ensemble des observations telles que $o_i \notin \hat{C}_k(o_i) \wedge o_i \in C_k(o_i)$, $r(o_i,o_j)$ le rang de o_j dans le voisinage de o_i en entrée et $\hat{r}(o_i,o_j)$ le rang de o_j dans le voisinage de o_i en sortie.

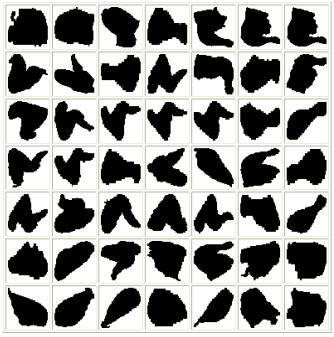
TAB. 2 : Comparaison des critères quantitatifs

Chicken	DSOM(K)	DSOM(EG)	DSOM
E_q	11.718	12.082	11.797
$\overline{M}_1$	0.892	0.904	0.936
$\overline{M}_2$	0.832	0.808	0.888

Les quantités $\overline{M}_1$ et $\overline{M}_2$ intègrent les critères topologiques $M_1(k)$ et $M_2(k)$ jusque dans un voisinage local incluant 10% des observations.



(a)



(b)

FIG. 1 : (a) Evolution comparée des critères de préservation du voisinage, (b) Illustration de la carte de sortie obtenue avec l'algorithme DSOM.

Les 3 algorithmes sont quasi équivalents en terme d'erreur de quantification (Tab. 2). Par contre la qualité de la préservation du voisinage locale est meilleure pour l'algorithme DSOM proposé (Fig. 1a). Cela provient de l'absence de collision si plusieurs unités partagent un même prototype. L'algorithme tire parti de la référence implicite aux prototypes durant la phase d'affectation. La figure 1.b illustre l'organisation des prototypes obtenue à la convergence, pour l'algorithme DSOM.

Les tableaux suivants montrent pour les algorithmes DSOM(K) (Tab. 3) et DSOM (Tab. 4), la répartition des 446 observations sur les 49 unités de la carte. Les cases grisées correspondent aux unités cohérentes sur les classes des observations. Le numéro de la classe est le premier

chiffre et le nombre d'observations pour cette classe est indiqué ensuite entre parenthèses. La classe indiquée en caractère gras est la classe du prototype (évaluée à la convergence pour DSOM). On observe une meilleure cohérence pour l'algorithme DSOM (nombre supérieur d'unités cohérentes) et pour les autres unités moins de dispersion des classes.

TAB. 3 : Organisation des observations sur la carte finale de DSOM(K) (cf texte)

5(1) **4**(30)	5(6) **4**(9)	**5**(5) 4(8)	1 (3) **5**(2) 2(2) 4(1)	4(1) **3**(5)	1(2) **3**(5)	1(1) 2(2) **3**(7) 5(3)
1(1) 4(1) **5**(11)	4(1) **5**(3)	**1**(25)	1(2) 2(3) **5**(3)	1(2) **3**(5)	2(1) 4(1) 1(2) **3**(4)	2(1) 1(3) **3**(5)
1(3) **5**(12)	4(2) **5**(10)	2(1) **5**(9)	4(1) **1**(16)	**3**(6)	2(1) **3**(4)	1(2) 5(2) **3**(4)
4(1) **5**(1)	2(1) **5**(3)	1(2) 2(3) **5**(4)	**1**(11)	**3**(5)	**3**(2) 1(9)	1(2) **3**(5)
1(1) **5**(7)	1(1) **5**(4)	1(1) **5**(2)	1(1) **3**(4)	**3**(5)	2(1) **3**(4) 1(8)	**3**(5) 1(8)
5(1) **4**(2)	1(1) **5**(1)	1(1) 5(3) **3**(4)	1(1) **3**(5)	**3**(1)	1(3) **2**(15)	**2**(15)
1(1) 3(1) 2(2) 4(2) **5**(2)	1(1) **5**(1)	1(1) **3**(1) 4(1)	2(1) **3**(5)	1(1) **3**(4)	1(1) **2**(9)	2(18)

TAB. 4 : Organisation des observations sur la carte finale de DSOM (cf texte)

4(16) 5(3)	**4**(16)	1(1) **4**(8)	**4**(3) 1(5)	**1**(16)	**1**(13)	**1**(14)
5(15)	4(1) **5**(8)	5(1) **4**(4)	**5**(1)	**1**(7)	**1**(1)	**1**(3)
5(2)	**5**(11)	2(1) **5**(9)	5(1) **2**(3)	**2**(1) 4(1)	**1**(1)	**1**(18)
5(8)	**5**(13)	2(2) 5(2) **4**(4)	5(1) 4(2) **2**(5)	∅	**1**(9)	**1**(10)
3(4) **5**(11)	**5**(4)	**5**(1)	4(1) **5**(1) 2(2)	**2**(4)	**1**(4)	2(1) **1**(7)
5(1) **3**(4)	**3**(4)	**5**(3)	**3**(8)	**1**(5)	**2**(4)	**2**(11)
3(20)	**3**(14)	**3**(23)	**3**(11)	**3**(8)	1(2) **2**(10)	**2**(32)

Des expérimentations supplémentaires sur d'autres bases de données réelles (bases de signaux, d'images de scènes naturelles, et de mots) ont confirmé le bon comportement de l'algorithme proposé dont les performances sont supérieures aux algorithmes comparés.

5. Conclusions

Les résultats expérimentaux montrent l'efficacité de l'algorithme proposé. Les critères de « fidélité » et de « continuité » du voisinage montrent une très bonne qualité de conservation du voisinage qui est un des objectifs de l'algorithme SOM. Cette efficacité est due à la nouvelle formulation de la distance aux prototypes sans avoir à les calculer explicitement dans l'espace d'entrée. Même si les hypothèses euclidiennes ne sont pas exactement vérifiées localement, les distorsions qui en découlent en conséquence, ont un effet bien moindre que les distorsions provoquées par le phénomène de collisions quand plusieurs unités partagent une même observation pour les représenter. Les expérimentations sur d'autres bases de données réelles ont confirmé ce bon comportement. Elles devront se poursuivrent pour complètement valider ce nouvel algorithme.

Remerciements. Ce travail est financé en partie par le Fonds National pour la Science, via le programme "ACI Masse de Données" et le projet "DataHighDim". Tien Ho-Phuoc a une bourse de thèse du MESR.

Références

[1] Kohonen T., *Self-Organizing Maps*. Springer Verlag New York (1997)

[2] Levenshtein V.I., *Binary codes capable of correcting deletions, insertions and reversals*, Soviet Physics Dokl., vol. 10 (8), (1966), 707-710

[3] Kohonen T., Somervuo P.J., *How to make large self-organizing maps for non vectorial data*, Neural networks, 21(8), (2002), 2002.

[4] El Golli A., Conan-Guez B., Rossi F., *A self organizing map for dissimilarity data*, IFCS-04, International Federation of Classification Societies, Chicago, (2004), 61-68.

[5] Ambroise C. and Govaert G., *Analyzing dissimilarity matrices via Kohonen maps*, IFCS-96, Int. Federation of Classification Societies, (2), Kobe (Japan), (1996), 96-99.

[6] Kohonen T., Somervuo P.J., *Self-organizing maps for symbol strings*, Neurocomputing, (21), (1998), 19-30.

[7] Martínez C. D., Juan A., Casacuberta F., *Improving classification using median string and NN rules*, IX Spanish Symp. on Pattern Recognition and Image Analysis, (2), (2001), 391-395

[8] http://algoval.essex.ac.uk:8080/data/sequence/chicken/chicken.tgz

[9] Venna J., Kaski S., *Neighborhood preservation in nonlinear projection methods: An experimental study*, ICANN 2001, Berlin, (2001), 485-491.