Classification non supervisée hiérarchique incrémentale basée sur le calcul de dissimilarités

Eugen Barbu, Pierre Héroux, Eric Trupin

Classification non supervisée hiérarchique incrémentale basée sur le calcul de dissimilarités

Eugen Barbu – Pierre Héroux – Eric Trupin

Laboratoire Perception Systèmes Information
CNRS FRE 2645 – Université de Rouen
UFR des Sciences et Techniques
Place E. Blondel
76 821 Mont-Saint-Aignan cedex - France

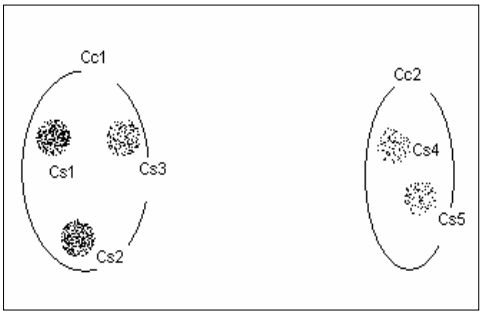
RÉSUMÉ. Cet article présente un algorithme incrémental de classification non supervisée. Cet algorithme se base uniquement sur la connaissance des dissimilarités entre les objets pris deux à deux. Une structure hiérarchique est construite puis mise à jour pour découvrir la structure cachée des données. Cette approche s'inspire des systèmes de classification conceptuelle non supervisée comme COBWEB ou HIERARCH... La recherche de plus proches voisins est optimisée en utilisant une structure de type M-tree.

MOTS-CLES : Classification non supervisée incrémentale, Dissimilarités.

1 Introduction

Etant donné un ensemble d'objets décrits par un nombre fixe d'attributs, l'objectif d'une tâche de classification non supervisée [KAU 90], [GOR 99] consiste à proposer une partition des objets en *k* sous-ensembles où le paramètre *k* est le nombre de regroupements attendus par l'utilisateur. Une variation de cette tâche est de ne pas utiliser le nombre attendu de regroupements comme une donnée du problème. Dans ce cas, l'algorithme construit plusieurs partitions candidates et choisit la meilleure. La meilleure partition est celle qui optimise un critère de qualité des partitions [MIL 85], [ROU 87] (statistiques sur les distances entre regroupements, indice de Calinsky-Harabasz, C-index, Silhouette Index, indice de Duda-Hart...). Plusieurs stratégies permettant la recherche des regroupements dans l'espace de toutes les partitions. On distingue les méthodes procédant par partitionnement, les méthodes hiérachiques (ascendantes ou procédant par divisions successives), les méthodes basées sur les densités et les méthodes de quantification. Dans certaines applications, les données à regrouper ne sont pas représentées par des vecteurs d'attributs. Elles peuvent par exemple être représentées par des graphes, des listes d'attributs de longueur variable, des mots ou sac de mots, des images... La seule information accessible est alors une mesure de dissimilarité entre objets. Plusieurs algorithmes de classification non supervisée [KAU 90]peuvent être appliqués à partir de la matrice contenant les mesures de dissimilarité entre les objets pris deux à deux. Les données peuvent être assignées à des regroupements, ces derniers pouvant alors être associés jusqu'à obtenir une hiérarchie de partitions (Fig. 1). La présentation hiérarchique des données se révèle souvent utile car la relation de catégorie à sous-catégorie existe dans bon nombre d'applications. Par exemple, dans une application d'analyse d'image de document, chaque caractère représente un regroupement de pixels, mais ces caractères peuvent eux-mêmes être regroupés en mots, lignes et paragraphes. Par ailleurs, certains regroupements de pixels correspondent non pas des données textuelles mais à des parties graphiques (schéma, images). Cette idée illustre les faiblesses des méthodes ne

proposant pas une hiérarchie des données mais un partitionnement à un seul niveau. Cependant, le problème majeur des algorithmes de classification hiérarchique est la complexité spatiale et temporelle (quadratique).
Dans les applications recevant en entrée un flux de données, comme c'est le cas pour les systèmes d'analyse de documents, log analysis ou les market-basket analysis systems, le caractère incrémental des algorithmes est primordial. Les algorithmes hiérarchiques de classification non supervisée ont été étudiés dans le domaine de l'apprentissage automatique et plus précisément dans le cadre de la formation de concepts dont les systèmes COBWEB, Classit et HIERARCH sont des mises en oeuvre. Ces systèmes utilisent en entrée des descriptions à base de vecteurs d'attributs, ce qui n'est pas sans répercussion sur la fonction objectif utilisée (par exemple le critère d'utilité d'une partition) pour construire la hiérarchie.
Dans la suite de cet article, nous décrivons une méthode de mise à jour incrémentale d'une hiérarchie (taxonomie) d'objets basée sur une mesure de dissimilarité entre objets. Lorsque la seule information disponible concernant objets à regrouper est leur dissimilarité réciproque, il n'est pas possible de déterminer le centre d'un regroupement. La mesure de qualité du partitionnement doit en conséquence se dispenser du calcul du centre du regroupement.



Comme beaucoup d'autres, notre méthode utilise une recherche de plus proche voisin. Nous l'optimisons en utilisant une méthode d'indexation basée sur la structure M-tree [CIA 97].
La section 2 décrit les opérations appliquées à la hiérarchie à l'arrivée de nouvelles données. L'algorithme proposé est donné en section 3. Cette section présente également la fonction

Figure 1 Catégories et sous-catégories de données bidimensionnelles

objectif utilisée et les transformations appliquées aux données d'entrée pour permettre l'utilisation de la méthode d'indexation M-tree. Un exemple d'application de notre algorithme est donné en section 4. La section 5 conclue cet article en pointant les faiblesses de notre approche et en proposant un certain nombre de perspectives.

2 Gestion de la hiérarchie

Nous distinguons les regroupements dits de base (C_{s1} à C_{s5} de la figure 1) des regroupements composites (C_{c1} et C_{c2}). Les regroupements composites contiennent au minimum deux regroupements (simples ou composites). Lorsqu'un nouvel élément est présenté deux évènements peuvent survenir : l'élément est assigné à un regroupement existant ou un nouveau regroupement est créé. Dans chaque cas de figure la hiérarchie est mise à jour. La figure 2 présente les modifications pouvant affecter la hiérarchie de partitions lors de l'ajout d'un nouvel élément. En plus de l'assignation de l'élément à un regroupement existant et de la création d'un nouveau regroupement, nous envisageons deux autres scénarios à l'arrivée d'un élément : deux regroupements peuvent être fusionnés ou, au

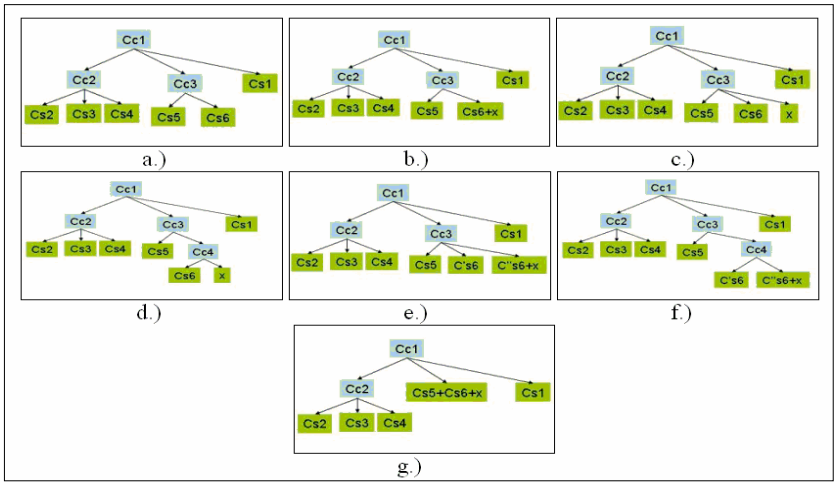


Figure 2 Mise à jour de la hiérarchie à l'ajout d'une nouvelle donnée (objet x)

contraire, un regroupement existant peut être divisé.

3 Algorithme

Add(Object obj)

```
PartitionHierarchy p; // initial partition
int hierarchyLevel;
Cluster cs1 = findNearestSimpleCluster(p, obj, &hierarchyLevel);
Cluster cs2 = findSecondNearestSimpleClusterAtHierarchyLevel(p, obj, hierarchyLevel);
if(isAddableTo(cs1))
{
        cs1'=cs1+obj;
  if(isSplitable(cs1'))
  {
            (cs1a,cs1b)=split(cs1');

        if(isBetterToCreateNewCategory(p,cs1a,cs1b))
            {
                cc=
                createNew
                ComplexCluster(cs1a,cs1b);
                add(p,cc,cs1.father);
                delete(p,cs1);
                //Fig. 2.f
            }
            else
            {
                add(p,cs1a,cs1.father);
                add(p,cs1b,cs1.father);
                delete(p,cs1);
                //Fig. 2.e
            }
  }
  else
  {
            if(isBetterToUnify(p,cs1',cs2))
            {
                cs = unify(p,cs1',cs2);
                delete(p,cs1);
                //Fig. 2.g
            }
            else
            {
                add(p,cs1',cs1.father);
                delete(p,cs1);
                //Fig. 2.b
            }
  }
}
else
{
        Cluster cs3 = new Cluster(obj);
        if(isBetterToCreateNewCategory(p,cs1,cs3))
        {
            cc = createNewComplexCluster(cs1,cs3);
            add(p,cc,cs1.father);
            delete(p,cs1);
            //Fig. 2.d
        }
        else
        {
            add(p,cs3,cs1.father);
            //Fig. 2.c
        }
}
```

La méthode M-tree est utilisée afin d' optimiser la recherche de plus proche voisin. Cette méthode, basée sur une structure dynamique (incrémentale), partitionne et organise l' espace de recherche. La mesure de dissimilarité entre éléments doit être une métrique. En effet, l' inégalité triangulaire y est employée pour réduire le nombre d' étapes lors de la résolution d' une requête donnée. Nous appliquons une transformations aux mesures de dissimilarité entre objets de respectant pas cette contrainte. Si *d(a,b)* est une mesure de dissimilarité alors la quantité $D(a,b)=\frac{d(a,b)}{1+d(a,b)}+1$ respecte la même relation d' ordre que *d*, ainsi que l' inégalité triangulaire.

$$D(a,b)+D(a,c)\geq D(b,c)\Leftrightarrow \frac{d(a,b)}{1+d(a,b)}+1+\frac{d(a,c)}{1+d(a,c)}+1\geq \frac{d(b,c)}{1+d(b,b)}+1\Leftrightarrow \frac{d(a,b)}{1+d(a,b)}+\frac{d(a,c)}{1+d(a,c)}+1\geq \frac{d(b,c)}{1+d(b,c)}$$

mais $\frac{d(a,b)}{1+d(a,b)}+\frac{d(a,c)}{1+d(a,c)}+1\geq 1>\frac{d(b,c)}{1+d(b,c)}$ .Aussi $D(a,b)\geq D(a,c)\Leftrightarrow d(a,b)\geq d(a,c)$ utilisant le propriétés de la fonction $\frac{x}{1+x}$. La propriété *d(a,a)=0* n' est plus respectée par *D*, mais cela n' empêche pas l' utilisation de la méthode M-tree. La transformation *D* permet donc d' obtenir une nouvelle base de dissimilarité dans laquelle est effectuée la recherche de plus proche voisin. Notre implémentation des fonctions permettant de décider de l' action à effectuer (création d' un nouveau regroupement, fusion de deux regroupements, division d' un regroupement) est actuellement basée sur la maximisation de la valeur moyenne du Silhouette-index [ROU 87]. Pour chaque objet u d'un regroupement A, nous définissons les valeurs suivantes : $a(u)=\frac{1}{|A|-1}\sum_{v\in A, v\neq u} d(u,v)$, $d(u,C)=\frac{1}{|C|}\sum_{v\in C} d(u,v)$, $b(u)=\min_{C\neq A} d(u,C)$ Le silhouette-index est

alors défini par : $s(u)=\frac{b(u)-a(u)}{\max\{a(u),b(u)\}}$. Si s*(u)* est proche de *1.0*, alors le rattachement de *u* au regroupement *A* est justifié. Si *s(u)* est proche de *0*, alors *u* se situe entre deux regroupement. Enfin, si s(u) est proche de –1.0, le rattachement de u au regroupement A n'est pas justifié, il devrait être rapproché d'un autre regroupement..

4 Exemple didactique

Nous employons la F-mesure globale afin d' évaluer la qualité du partitionnement effectué par notre algorithme de classification non supervisée. Soient *D* l' ensemble des objets et *C = {C1, . . . , Ck}* une partition de *D*. Soit par ailleurs, *C' = {C'1 , . . . , C'l }* la partition de référence. Le rappel, la précision et la F-mesure du regroupement *j* par rapport à la classe *i* sont respectivement définis par $rec(i,j)=\frac{|C_j\cap C'_i|}{C'_i}$, $prec(i,j)=\frac{|C_j\cap C'_i|}{C_j}$ et $F_{ij}=\frac{2prec(i,j)rec(i,j)}{prec(i,j)+rec(i,j)}$. La F-mesure globale d' une partition définie par rapport à une partition de référence est définie par $F=\sum_{i=1}^{l}\frac{|C'_i|}{|D|}*\max\{F_{ij}\}_{j=1...k}$. Cette mesure vaut *1.0* lorsqu' il y a correspondance parfaite entre *C* et *C'*.

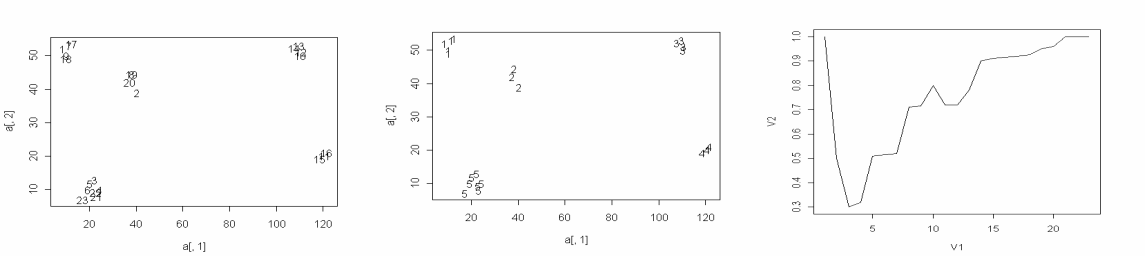


Figure 3. a) Ordre d' introduction des données b) Vérité terrain c) Evolution de la F-Mesure lors de l' introduction des données

5 Conclusion et perspectives

Dans cet article, nous avons proposé une construction incrémentale d' une classification hiérarchique n' utilisant que des dissimilarités entre les éléments. Nous envisageons de tester l' approche décrite dans cet article sur des données synthétiques et réelles. D' autres fonctions objectif devront également être implémentées et testées. L' étude de la distance entre la sortie de l' algorithme de classification non supervisée et la partition de référence devra être approfondie.

6 Bibliographie

[CIA 97] CIACCIA C., PATELLA M., ZEZULA P., " M-tree an Efficient Access Method for Similarity Search in Metric Spaces ", In Proc. of the 23th Conference on Very Large Databases, p. 426-435, 1997.

[FIS 87] FISCHER D., " Knowledge acquisition via Incremental Conceptual Clustering ", ML, n° 2, 1987

[GOR 99] GORDON A. D., *Classification 2nd Edition*, Chapman & Hall, 1999.

[KAU 90] KAUFMAN L., ROUSSEEUW P. J., *Finding Groups in Data*, John Willey & Sons, 1990.

[MIL 85] MILLIGAN G. W., COOPER M. C., " An Examination of Procedures for Determining the Number of Clusters in a Data Set ", *Psychometrika*, vol. 58, n° 2, 1985, p. 159-179.

[NEV 95] NEVINS A. J., " A Branch and Bound Incremental Conceptual Clusterer ", ML , n° 18, 1995.

[ROU 87] ROUSSEEUW P. J., " Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis ", *J. Comput. Applied. Math.*, n° 20, 1987, p. 53-65.