TREIZIÈME COLLOQUE GRETSI - JUAN-LES-PINS DU 16 AU 20 SEPTEMBRE 1991

CLASSIFICATION NON SUPERVISÉE PAR DÉTECTION DES ZONES FRONTIERES APPLICATION EN RECONNAISSANCE DES FORMES ET SEGMENTATION

Changquan GAN et Bernard DUBUISSON

Université de Compiègne, URA CNRS Heudiasyc
BP.649, 60206 Compiègne, France

RÉSUMÉ

Résumé : Cet article présente une méthode de classification non supervisée. L'ensemble des données est représenté par un graphe. Une procédure de relaxation probabiliste est utilisée pour détecter les nœuds frontières. La formation des classes devient facile en identifiant les composantes connexes du graphe après avoir retiré les nœuds frontières.

ABSTRACT

Abstract : We present a clustering method in which data set is represented by a graph and a relaxation process is used for boundary nodes detection. Clusters formation become easy by identifing connected components in the graph after elimination of the boundary nodes.

1. INTRODUCTION

En analyse d'image comme en traitement du signal, on a souvent besoin de constituer des classes à partir des caractères retenus pour représenter les phénomènes. La segmentation en région et la reconnaissance des formes en traitement du signal en sont des exemples. La constitution des classes est un problème difficile lorsqu'on ne dispose pas ou de très peu de connaissance a priori sur la structure sous-jacente des données. Les techniques de classification non supervisée sont des outils importants dans la résolution d'un tel problème [1]. La classification non supervisée dite aussi coalescence tente de résoudre le problème suivant : à un ensemble de N points donné $E = \{ x_1, x_2, \ldots x_N \mid x_i \in \Re^d \}$ trouver une c-partition de telle sorte qu'elle révèle le mieux possible la structure naturelle de E. Les éléments de la partition obtenue sont appelés classes. Dans la pratique chaque point x_i caractérise un individu sur lequel on a relevé d attributs. L'espace $\Re^d$ est dit espace de représentation. Il existe une grande variété de méthodes de coalescence, parmi lesquelles on distingue trois catégories principales, celles qui procèdent par optimisation itérative d'un critère [2] [3], celles qui s'appuient sur l'estimation de la densité de probabilité [4] [5] et celles qui utilisent la théorie des graphes [6] [7] [8]. La méthode que nous proposons dans cet article est en relation avec les deux dernières catégories. La détection de mode/vallée est le chemin suivi par la plupart des méthodes de la deuxième catégorie. Un mode est une région de l'espace de représentation où il y a une forte densité de points, qui se distingue de régions d'alentour nommée vallée. L'idendification de mode passe par la détection de pic ou de partie convexe de la fonction de densité estimée [4] [5]. La phase estimation nécessite un nombre suffisant de points. Elle est normalement importante en calcul. Le choix des paramètres est crucial et souvent difficile. Les méthodes de graphe se basent sur un graphe connexe construit selon une relation de voisinage définie sur l'ensemble E. Chaque point x_i est un nœud dans le graphe. Deux nœuds sont reliés par un arc si et seulement s'ils sont en relation de voisinage. Les critères heuristiques sont employés pour identifier et éliminer les *arcs inconsistents*. Les composantes connexes dans le reste du graphe correspondent à une partition de l'ensemble E. Les structures de graphe usuelles sont MST (minimum spanning tree [6]), RNG (relative neighborhood graph [7]), GG (Gabriel graph [7]) et DT (Delaunay triangulation [8]). Deux facteurs empêchent l'application de ces méthodes aux problèmes de grande dimension (d>3) : manque de critère heuristique et complexité algorithmique de la génération du graphe. Nous proposons une méthode en utilisant une structure de graphe facile à calculer à l'aide de la recherche des K plus proches voisins. Une technique de détection de mode/vallée est adaptée à l'identification des *nœuds inconsistents* que nous appelons les nœuds (ou points) frontières pour la simple raison que les vallées sont les régions frontières des classes. Notre stratégie est donc d'éliminer d'abord les nœuds frontières du graphe, puis de former les noyaux des classes en cherchant les composantes connexes dans le graphe et, enfin, d'affecter les points frontières aux classes établies.

2. DESCRIPTION DE LA MÉTHODE

A. GRAPHE

Désignons par ρ une distance de l'espace de représentation $\Re^d$. A un entier K donné ($1 \le K \le N = \mathrm{Card}(E)$), nous définissons une relation symétrique R_v sur E par la notion de K Plus Proches Voisins (KPPV) au sens de la distance ρ. Soit x_i et x_j deux points différents de E. Si x_i est parmi les KPPV de x_j et réciproquement si x_j est parmi les KPPV de x_i, alors $x_i R_v x_j$. Cette relation de voisinage peut être représentée par un graphe non orienté G = (E, A) où les nœuds sont les éléments de E et où les arcs sont définis par $A = \{ (x_i, x_j) \in E \times E \mid x_i R_v x_j \}$. Nous avons $N < M < KN$ avec M = Card(A). Il est important de remarquer que G n'est pas forcément connexe si les classes potentielles sont bien séparées et si K est faible. En pratique nous exprimons G sous forme d'une liste d'adjacence. Elle est facile à obtenir à partir de la liste des KPPV. La complexité algorithmique est au plus O(KN).

B. INDICE DE LA VARIATION DE DENSITÉ

Nous proposons un indice local pour mesurer la variation de densité des points dans l'espace de représentation. Il ne sera évalué que sur les points de E. Concrètement, pour un x_i, on

cherche ses KPPV : x_{i_1}, x_{i_2}, ... x_{iK} dans E. A chaque j ($1 \le j \le K$), on examine x_i afin de savoir s'il est aussi parmi les KPPV de x_{ij}, si oui on note par $r_{i,j}$ son rang dans les K voisins, si non on prend $r_{i,j}$ égal à K+1. Posons :

$$s(x_i) = \sum_{j=1}^{K} r_{i,j}, \quad s_{Min} = \min_{i=1,N}\{s(x_i)\} \quad \text{et} \quad s_{Max} = \max_{i=1,N}\{s(x_i)\}$$

L'indice de la variation de densité est défini par :

$$I(x_i) = \frac{s(x_i) - s_{Min}}{s_{Max} - s_{Min}}$$

La valeur de I est toujours comprise entre 0 et 1. En général, si x_i est à l'intérieur d'une région modale, $I(x_i)$ sera petit. Si, au contraire, il appartient aux régions frontières, $I(x_i)$ sera grand. Une forte valeur de K rend l'indice I plutôt sensible à la variation globale qu'à la variation locale de densité, et une faible valeur de K produit un effet inverse. En pratique, l'évaluation de $s(x_i)$ pourra être intégrée dans la construction de la liste d'adjacence du graphe G. Il en résulte que la complexité de calcul des $I(x_i)$ ne dépasse pas O(KN).

C. DETECTION DES NŒUDS FRONTIERES PAR RELAXATION

Nous devons étiqueter chaque point (ou nœud) de l'ensemble E par l'une des deux étiquettes F (frontière) ou $\overline{F}$ (non frontière). L'indice I que nous venons de définir est une mesure imprécise et sensible aux irrégularités locales de la distribution des points. L'étiquetage obtenu par un simple seuillage de I n'est pas souvent satisfaisant. Un point frontière (ou de mode) peut être incorrectement étiqueté par $\overline{F}$ (ou par F) si I est localement inférieur (ou supérieur) au seuil. Ces erreurs pourront être considérablement réduites voire même éliminées en utilisant une procédure de relaxation. La relaxation est un schéma d'adaptation de probabilité d'étiquetage, dans lequel la décision d'étiquetage prise sur un point n'est plus indépendante de celles prises sur les points voisins et doit être compatible avec elles. Nous avons adapté une technique de relaxation présentée dans [5] à la résolution de notre problème.

Notons (x_i, λ) l'étiquetage de x_i avec λ. $\lambda \in \{F, \overline{F}\}$. D'abord nous prenons $I(x_i)$ comme estimation initiale de probabilité pour (x_i, F), soit :

$$P_i^{(0)}(F) = I(x_i) \quad \text{et donc} \quad P_i^{(0)}(\overline{F}) = 1 - P_i^{(0)}(F) \quad \text{pour } (x_i, \overline{F})$$

Ces probabilités seront ajustées itérativement par la formule ci-dessous :

$$P_i^{(n+1)}(\lambda) = \frac{P_i^{(n)}(\lambda)[1+Q_i^{(n)}(\lambda)]}{P_i^{(n)}(F)[1+Q_i^{(n)}(F)] + P_i^{(n)}(\overline{F})[1+Q_i^{(n)}(\overline{F})]}$$

$Q_i^{(n)}(\lambda)$ représente l'ajustement de la probabilité $P_i^{(n)}(\lambda)$ de l'étiquetage (x_i, λ) à la n ième itération. Il totalise les contributions venant des points voisins et sera spécifié par la suite. La nouvelle estimation est donc basée sur l'estimation précédente et sur la contribution des voisins. On définit une mesure de compatibilité entre (x_i, λ) et (x_j, λ') par l'expression suivante :

$$C_{ij}(\lambda, \lambda') = \frac{[P_i^{(n)}(\lambda) - P_{Moy}^{(n)}(\lambda)]\cdot[P_j^{(n)}(\lambda') - P_{Moy}^{(n)}(\lambda')]}{[P_{Max}^{(n)}(\lambda) - P_{Moy}^{(n)}(\lambda)]\cdot[P_{Max}^{(n)}(\lambda') - P_{Moy}^{(n)}(\lambda')]}$$

où $i \neq j$, $P_{Moy}^{(n)}(\lambda) = \frac{1}{N}\sum_{h=1}^{N} P_h^{(n)}(\lambda)$ et $P_{Max}^{(n)}(\lambda) = \max_{h=1,N}\{P_h^{(n)}(\lambda)\}$.

Si l'étiquetage (x_i, λ) est compatible avec l'étiquetage (x_j, λ'), $C_{ij}(\lambda, \lambda') > 0$. Dans le cas contraire $C_{ij}(\lambda, \lambda') < 0$. Quand (x_i, λ) et (x_j, λ') sont indépendants $C_{ij}(\lambda, \lambda') = 0$.

Le terme $Q_i^{(n)}(\lambda)$ prend la forme suivante :

$$Q_i^{(n)}(\lambda) = \frac{1}{K}\sum_{x_j \in V_i(K)} [\, C_{ij}(\lambda, F)P_j^{(n)}(F) + C_{ij}(\lambda, \overline{F})P_j^{(n)}(\overline{F}) \,]$$

où $V_i(K)$ est l'ensemble des KPPV de x_i. Pour que la nouvelle estimation $P_i^{(n+1)}(\lambda)$ reste toujours positive, $|Q_i^{(n)}(\lambda)|$ doit être inférieur ou égal à 1. Il est donc préférable que la mesure de compatibilité soit comprise entre -1 et 1, or $|C_{ij}(\lambda, \lambda')| > 1$ n'est pas exclu avec la définition ci-dessus. Nous devons tronquer la valeur de $C_{ij}(\lambda, \lambda')$ en cas de besoin. D'après la définition de $I(x_i)$ et de $P_i^{(0)}(\lambda)$ nous avons $P_{Max}^{(0)}(F) = P_{Max}^{(0)}(\overline{F}) = 1$. La formule itérative nous indique que, si une probabilité atteint 0 ou 1, cette valeur persistera. Donc $P_{Max}^{(n)}(F) = P_{Max}^{(n)}(\overline{F}) = 1$ quelque soit n.

Le processus de relaxation converge quand les valeurs de probabilité se stabilisent. Un test sur la moyenne des changements de valeur de probabilité décide l'arrêt de l'itération. D'où la nécessité de se donner un seuil. Après l'arrêt du processus, la plupart des probabilités finales ont atteint leur valeur limite 1 ou 0 qui indique un étiquetage sans ambiguïté de "frontière" ou de "non frontière". Un point x_i dont la probabilité finale de l'étiquette F est supérieure à 0.5 sera désigné comme point frontière, et il sera marqué dans la tête de liste d'adjacence. La convergence de la relaxation a été prouvée expérimentalement. Le seuil de critère d'arrêt dépend peu des données et, une fois choisi, peut être considéré comme fixe. Dans les exemples réalisés, il a été fixé à 0,01, soit 1% de l'intervalle de probabilité [0, 1].

D. CONSTITUTION DES CLASSES

Il s'agit donc d'identifier les composantes connexes dans le graphe G sans tenir compte des nœuds marqués (nœuds frontières). Nous adoptons un algorithme bien connu dit de "recherche selon la profonteur d'abord" [9]. Sa complexité est O(Max{N, M}) où N est le nombre des nœuds et M le nombre des arcs. D'après le sous paragraphe A, O(Max{N, M}) = O(M) avec M<KN. Les nœuds appartenant à une composante connexe forment le noyau d'une classe. S'il n'y a qu'une seule composante connexe, nous devons éliminer les nœuds frontières de l'ensemble E, adapter la liste des KPPV, recalculer l'indice I, recommencer la relaxation et répéter ces opérations jusqu'à ce que le nombre de composantes connexes soit supérieur à 1 ou Card(E) inférieur à K. Ce dernier cas signifie qu'il n'y a pas de structure de classe présente dans E. Pour éviter les calculs redondants dans l'adaptation de la liste des KPPV (éviter de recommencer totalement la recherche des KPPV dans E), nous pouvons adopter une stratégie de recherche des KPPV présentée dans [10].

Après la constitution des classes par les éléments noyaux, nous affectons les nœuds frontières aux classes existantes selon la règle du plus proche voisin. Etant donné que nous avons la liste des KPPV, cette opération sera très rapide. La complexité est certainement inférieure à O(N).

3. ALGORITHME

Etape 1. Lire un ensemble de données E. Entrer le paramètre K.

Etape 2. Dresser le tableau de la liste des KPPV dans lequel on fait correspondre à chaque point de E ses K plus proches voisins déterminés selon la stratégie proposée dans [10]. Construire la liste d'adjacence du graphe G.

Etape 3. Calculer l'indice I pour chaque point.

Etape 4. Détecter les nœuds frontières de E par la relaxation. Marquer sur la liste d'adjacence de G.Eliminer ces nœuds de l'ensemble E et les sauvegarder dans une pile.

Etape 5. Idendifier les composantes connexes du graphe G sans tenir compte des nœuds marqués.

Etape 6. Soit c le nombre des composantes connexes obtenues.
Si c=1, si Card(E) < K, aller à l'étape 8.
sinon, adapter la liste des KPPV, aller à l'étape 3
Sinon, passer à l'étape suivante.

Etape 7. Constituer un noyau de classe pour chaque composante

connexe obtenue par les nœuds qui lui appartiennent.
Affecter les nœuds frontières stockés dans la pile aux classes existantes selon la règle du plus proche voisin.

Etape 8. Sortir le résultat.

La complexité algorithmique de certaines parties de cet algorithme est déjà connue. Il reste celle de la détermination des KPPV et celle de la relaxation que nous ne pouvons évaluer que expérimentalement.

4. ETUDE DE LA PERFORMANCE

L'algorithme précédent a été implanté en un programme C nommé CLUST. Comme le nombre de voisins K est pratiquement le seul paramètre à choisir dans l'utilisation, ce programme a été conçu de manière à ce qu'on puisse multiplier les essais sur K tout en évitant les calculs redondants pour la recherche des KPPV. Nous avons réalisé de nombreux tests pour évaluer la performance de CLUST. Voici quelques résultats obtenus :

1) Données simulées gaussiennes (voir la figure (a)) :

L'ensemble consiste en 450 points issus de 3 classes gaussiennes de dimension 2 dont les paramètres sont les suivants :

Probabilités a priori : $P_1 = \frac{200}{450}$, $P_2 = \frac{100}{450}$, $P_1 = \frac{150}{450}$.

Vecteurs moyennes : $\mathbf{m}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\mathbf{m}_2 = \begin{bmatrix} 3.6 \\ 0 \end{bmatrix}$, $\mathbf{m}_3 = \begin{bmatrix} 1.8 \\ 3.6 \end{bmatrix}$.

Matrices de covariance : $\Sigma_1 = \Sigma_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $\Sigma_3 = \begin{bmatrix} 1 & 0 \\ 0 & 1.2 \end{bmatrix}$.

Le programme a été testé pour K = 6, 8, 10, ... , 80. La figure (b) montre le nombre de classes trouvé par CLUST en fonction du paramètre K. Nous pouvons observer une large plage pour K (de 28 à 80) où le nombre de classes obtenu est stable et égal à 3 (le nombre exact des classes présentes). Pour chaque valeur de K, nous avons relevé le taux d'erreur de classification par comptage du nombre de points mals classés (voir (c)). Comparant (b) et (c) nous constatons que, dans la plage de stabilité du nombre de classes, les erreurs sont aussi peu fluctuantes et très faibles. En effet la majorité des points ont eu leur affectation stable dans cette plage. Nous allons retrouver cette propriété dans les exemples suivants. Donc, la stabilité du nombre de classes et la stabilité d'affectation des points peuvent être considérées comme un critère de validité pour le résultat obtenu. Afin d'apprécier la qualité de classification, ces 450 points sont aussi classés par le discriminateur de Bayes avec les paramètres de classes données ci-dessus. Le taux d'erreur trouvé est de 5.1%. C'est le résultat optimal qu'on peut obtenir puisqu'il y a des chevauchements entre ces trois classes. Pour CLUST nous trouvons dans la figure (c) un taux d'erreur de 5% à 6% sur la plage stable et une moyenne de 5.2%. Ce qui est presque le résultat optimal. En réalité, il y a peu de points qui sont affectés différemment par notre programme et par le discriminateur de Bayes. Ces points sont très proches des surfaces de décision de la règle Bayes.

Pour donner une idée sur le rôle de relaxation, nous présentons un exemple avec K = 40. La procédure de relaxation a été appelée deux fois (la classification n'a pas eu lieu après le premier appel car une seule composante connexe a été identifiée). Les points frontières détectés après le premier appel sont dessinés dans la figure (d) par le symbole "+". Les points non frontières sont marqués par "·". La figure (e) montre le résultat du deuxième appel de la relaxation. Comme nous pouvons le constater, les noyaux des classes sont bien saisis et bien isolés. Ce qui explique le succès de classification qui présente un taux d'erreur minimum de 5% (voir figure (c)). La convergence du processus de relaxation est rapide, dans tous les tests réalisés le nombre d'itération n'a jamais dépassé 12.

Après avoir trouvé correctement 3 classes, nous avons appliqué le programme séparément sur chacune de ces 3 classes pour tester le comportement du programme envers un ensemble de données qui ne présente pas de structure de classes. La figure (f) montre le résultat de la deuxième classe. Nous pouvons observer qu'il n'y a plus de plage de stablilité pour le nombre de classes obtenu sauf si ce nombre égal à 1. Le même type de courbe a été trouvé pour les deux autres classes.

2) Données simulées dimension 2 (voir la figure (g)):

L'ensemble est constitué de deux classes, l'une "parabolique" (194 points) et l'autre gaussienne (93 points). Le but de ce test est de prouver la capacité de CLUST pour les structures de formes variées. Les valeurs de K sont des nombres pairs compris entre 4 et 60. Dans les figures (h) et (i) nous constatons encore une fois une nette plage de stabilité (de 8 à 60) pour le nombre de classes égal à 2. Du fait que les deux classes originales sont bien séparées, la plage commence très tôt (K = 8) et le taux d'erreur est extrêmement faible (0.7%). Deux points seulement sont mals classés. Ils se situent entre les classes.

3) Données réelles (IRIS) :

IRIS est un ensemble de données en dimension 4. Il est composé de 150 observations sur 3 espèces des fleurs Iris, 50 pour Iris Setosa, 50 pour Iris Versicolor et 50 pour Iris Virginica. Chaque observation comprend 2 mesures de sépale et 2 mesures de pétale. Iris Setosa est connu être linéairement séparable de Iris Versicolor et de Iris Virginica. Tandis que ces deux dernières classes présentent un chevauchement important entre elles. Le programme CLUST a été exécuté avec K variant de 1 à 40. La figure (j) nous indique l'existence de deux classes (plage de stabilité de 10 à 40). En effet, la classification réalisée dans cette plage reste toujours la même et correspond à la séparation de Iris Setosa des deux autres classes. Nous avons continué l' examen pour chacune des deux classes obtenues avec CLUST. Iris Setosa a été jugé comme une seule classe, par contre pour l'union de Iris Versicolor et Iris Virginica le programme a manifesté une préférence pour une séparation en 2 classes (plage de stabilité de 8 à 12 sur la figure (k)). Nous avons donc choisi une classification pour K = 10 (milieu de la plage). La poursuite d'examen des deux classes obtenues n'a conduit à aucune nouvelle classification. Le programme a donc encore une fois trouvé la bonne réponse. Nous avons également rélevé les erreurs totales de classification en fonction de K. La courbe est présentée dans (l). Nous avons un taux d'erreur de 16.7% pour la solution choisie K = 10, soit 83.3% observations correctement classés sur 150. Ce résultat est moins bon mais tout de même comparable à celui donné par l'algorithme des c-moyennes [2] en spécifiant 3 comme nombre de classe. Ce dernier est égal à 12%. Notons que pour K = 14, nous avons une classification en 3 classes bien meilleure et le taux d'erreur est seulement de 10%, mais malheureusement c'est n'est pas une solution stable pour CLUST.

Les expériences ci-dessus ont donc montré que la performance de notre programme CLUST est très satisfaisante en général.

5. CONCLUSION

Nous avons proposé une méthode de classification non supervisée basée sur un graphe. Une technique de relaxation a été adaptée à la détection des points frontières pour isoler les noyaux de classes potentielles. La constitution de classes devient facile en identifiant les composantes connexes dans le graphe. Cette méthode a les avantages suivants : (1) Elle peut travailler sans aucune connaissance a priori sur les données. Le nombre de

classes n'est pas nécessaire. (2) Il y n'a pratiquement qu'un seul paramètre à choisir. Donc elle est facile à utiliser en pratique. (3) Elle n'a recours à aucun critère heuristique. (4) La bonne stabilité et la bonne qualité de classification sur la plage de paramètre K où le nombre des classes obtenus est constant. L'application de cette méthode en reconnaissance des formes est immédiate. Elle peut être aussi adaptée pour résoudre efficacement certains problèmes en traitement d'images, par exemple une segmentation du type [11].

Références

[1] A.K.Jain and R.C.Dubes, Algorithms for clustering data, Prentice Hall Advanced Reference Series, 1988.

[2] J.B.McQueen, "Some methods of classification and analysis of multivariate observations", Proceeding of 5th Berkeley symposium on mathematical statistics and probability, pp.281-297, 1967.

[3] E.Diday et collaborateurs, Optimisation en classification automatique, INRIA, 1979.

[4] C.P.Vasseur and J.G.Postaire, "A convexity testing method for cluster analysis", IEEE Trans. Syst., Man., Cybern.,Vol.SMC-10, No.3, 1980.

[5] A.Touzani and J.G.Postaire, "Mode detection by relaxation", IEEE,Trans. Pattern Anal. Machine Intell, Vol.PAMI-10, No.6, pp.970-978, 1988.

[6] C.T.Zahn, "Graph-theoretical methodes for detecting and describing Gestalt clusters", IEEE Trans. Comput., Vol.C-20, No.1, pp.68-86, 1971.

[7] R.Urquhart, "Graph theoritical clustering based on limited neighborhood sets", Pattern Recognition Vol.15, No.3, pp.173-187, 1982.

[8] N.Ahuja, "Dot pattern processing using voronoi neighborhoods", IEEE, Trans. Pattern Anal. Machine Intell,Vol.PAMI-4,No.3,pp.336-343,1982.

[9] S.Baase, Computer algorithms : Introduction to design and analysis, Addison-Wesley Pubishing Company, 1978.

[10]A.J.Broder,"Strategies for efficient incremental nearest neighbor search", Pattern Recognition, Vol.23, No.1/2, pp.171-178, 1990.

[11]N.Ahuja and M.Tuceryan, "Extraction of early perceptual structure in dot patterns : integrating region, boundary, and component gestalt", Computer vision, Graphics and Image processing, 48, pp.304-356, 1989.