Apprentissage supervisé pour la classification des images basé sur la structure P-tree

Rim Faiz*, Najeh Naffakhi**, Khaled Mellouli*

*LARODEC, IHEC, 2016 Carthage-Présidence, Tunisie
Rim.Faiz@ihec.rnu.tn
Khaled.Mellouli@ihec.rnu.tn

**LARODEC, ISG Tunis, 2000 Le Bardo, Tunisie
Najeh.Naffakhi@isg.rnu.tn

Résumé. Un problème important de la production automatique de règles de classification concerne la durée de génération de ces règles ; en effet, les algorithmes mis en œuvre produisent souvent des règles pendant un certain temps assez long. Nous proposons une nouvelle méthode de classification à partir d'une base de données images. Cette méthode se situe à la jonction de deux techniques : l'algèbre de P-tree et l'arbre de décision en vue d'accélérer le processus de classification et de recherche dans de grandes bases d'images. La modélisation que nous proposons se base, d'une part, sur les descripteurs visuels tels que la couleur, la forme et la texture dans le but d'indexer les images et, d'autre part, sur la génération automatique des règles de classification à l'aide d'un nouvel algorithme C4.5(P-tree).
Pour valider notre méthode, nous avons développé un système baptisé C.I.A.D.P-tree qui a été implémenté et confronté à une application réelle dans le domaine du traitement d'images. Les résultats expérimentaux montrent que cette méthode réduit efficacement le temps de classification.

1 Introduction

La caractérisation de classes et la classification sont d'importants domaines de recherche en fouille de données et apprentissage. Leur objectif est l'extraction non triviale des informations utiles et potentielles non connues d'avance à partir de données structurées. Mais, cet objectif est difficile à atteindre lorsqu'on est devant une base de données multimédia.

Les applications actuelles traitent plutôt des bases de données structurées bien que l'extraction des connaissances issues de données multimédia (des images, des signaux, des séquences vidéo,…) devient de plus en plus un axe de recherche actif, notamment pour le traitement d'informations recueillies lors des recherches sur le web. Nous nous intéressons à l'un des aspects de ce problème complexe, celui de l'extraction des connaissances à partir d'une base de données images.

Les documents images ont comme caractéristique principale le manque d'un langage permettant d'en exprimer la sémantique. Il en résulte une très grande difficulté pour proposer des systèmes de classification et de recherche à partir d'une base d'images (Chahir et al. 1999).

Notre travail s'inscrit précisément dans ce domaine de recherche qui vise à générer des règles de production et réduire considérablement le temps de classification. L'exemple que nous avons utilisé est une base d'images de fraise. Ces images sont représentées par un ensemble de couples (attribut, valeur).

Dans cette étude, nous proposons une nouvelle méthode de classification en se basant sur l'algorithme C4.5 développé par Quinlan (Quinlan 1993). Pour la classification d'images, le temps est un facteur important. Cependant, ces données non structurées sont volumineuses (taille de données) et leurs classifications par les méthodes existantes se fait en un temps assez élevé. Dans ce travail, nous avons développé une nouvelle méthode de classification basée sur l'arbre de décision en utilisant une structure de donnée appelée P-tree (arbre de Peano Count). Cette méthode, d'une part, évite le balayage direct de la base de données qui est une opération assez coûteuse en mémoire et en temps d'exécution et, d'autre part, elle offre un gain de comparaison des attributs (items) car la comparaison s'effectue par bloc.

Dans la section suivante, nous présentons quelques travaux existants liés aux méthodes de classification et d'apprentissage. La section 3 introduit la structure P-tree. Nous détaillons, dans la section 4, notre méthode de classification des images basée sur l'algorithme C4.5 et utilisant la structure P-tree. La section 5 est consacrée à l'implémentation du système *C.I.A.D.P-tree* qui valide notre méthode. Nous terminons par une présentation de quelques perspectives de ce travail.

2 Classification et apprentissage automatique

Dans le cadre de cette étude, nous nous intéressons à un type particulier d'apprentissage qui est l'apprentissage inductif. Le processus d'apprentissage inductif peut être considéré comme une recherche de descriptions générales plausibles qui expliquent les données d'entrée et qui sont utiles pour en prédire de nouvelles (Borgi et al. 2001). Il existe de nombreux travaux sur la théorie de l'apprentissage en général et sur l'apprentissage inductif en particulier (Dietterich et al. 1983) (Michalski et al. 1983) (Carbonell et al. 1986) (Kodratoff et al. 1991). Deux approches distinctes de l'apprentissage existent : la première qualifiée d'apprentissage non supervisé vise à regrouper en classes des objets en se basant sur des ressemblances ou similarités entre eux, la deuxième approche est l'apprentissage supervisé qui se base quand a lui sur un ensemble d'apprentissage constitué d'objets dont la classe est connue *a priori*. C'est à ce dernier type d'apprentissage que nous nous intéressons (cf. FIG. 1).

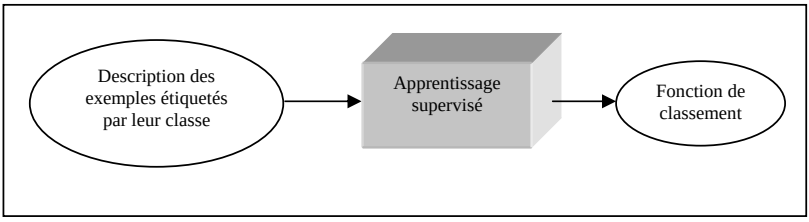


FIG. 1 – *L'apprentissage supervisé*

Dans notre méthode, la prémisse est une conjonction de descripteurs logiques du type *Attribut = Valeur*, et la conclusion est de la forme «appartenance à une classe parmi les classes de $\boldsymbol{C} = \{y_1, y_2, \ldots, y_C\}$».

L'objectif de l'apprentissage inductif par génération de règles est d'identifier un nombre minimal de règles qui généralisent les exemples de la base d'apprentissage. Le problème de recherche d'un sous ensemble minimal de combinaisons est un problème NP-Complet et les méthodes existantes sont toujours heuristiques (Bonnet 1984). Ces heuristiques diffèrent par leur façon de construire les combinaisons minimales. La plupart des méthodes de génération de règles construisent les prémisses en ajoutant un par un des descripteurs correspondant à des optima locaux, ces dernières sont des méthodes mono-attributs ou monothétiques.

Les arbres de décision correspondent à une approche de classification mono-attributs. Trois systèmes ont plus particulièrement marqué les travaux sur les arbres de décision : ID3 et C4.5 (Quinlan 1993), (Quinlan 1983) dans la communauté de l'IA, et CART (Breiman et al. 1984) dont l'origine est statistique. Plus récemment, des travaux ont étudiés les arbres de décision flous (Marsala 1998), (Ramdani 1994), qui permettent de traiter des connaissances imprécises. D'autres ont étudiés les arbres de décisions basées sur la croyance (Mellouli et al. 2001), qui traitent des incertitudes.

Les principales caractéristiques de la méthode développée dans le cadre de notre étude sont les suivantes :

- La méthode d'apprentissage est mono-*attribut* ou mono*thétique*, cela permet de prendre en compte d'un fort pouvoir explicatif sur l'ensemble des exemples. En effet, cette méthode consiste à décomposer la base d'apprentissage en sous-bases, chacune induite par une modalité de la liste des modalités de l'attribut traité.
- La construction des prémisses des règles se fait étape par étape. A chacune d'elles, on ajoute une condition sur un meilleur attribut, et le choix du meilleur attribut se fait selon le gain d'information. L'élu attribut est celui possédant le gain le plus important.
- Les règles générées sont des règles de classification : leurs conclusions sont de la forme appartenance à une classe. Un chemin dans un arbre de décision est équivalent à une règle de classification. La prémisse de cette règle est une conjonction des tests réalisés sur le parcours en question, et la partie conclusion contient la classe libellant la feuille du chemin, atteinte en fin de parcours.
- Afin de réduire la masse d'information pour ne pas surcharger l'espace mémoire et pour éviter des temps et complexité de traitements trop élevés, nous utilisons la structure P-tree pour comprimer les données originales et par conséquence faciliter et accélérer l'exploitation efficace de ces données.

Notre méthode se situe à la jonction des méthodes de classification *mono-attribut* et de la structure P-tree. Elle permet de combiner les avantages des classifieurs basés sur les règles de classification et la structure P-tree. La structure P-tree permet d'accélérer le processus de classification. En effet, les données d'origine sont converties sous format binaire ce qui permet de réduire le temps de classification. Nous proposons dans la section suivante quelques détails supplémentaires de l'algèbre de P-tree.

3 Présentation de la structure P-tree

L'extraction des règles de classification à partir des données non structurées telles que les données images est une tâche importante et complexe (*classer des images de l'environnement en deux groupes: environnement pollué et environnement dépollué, en médecine regrouper des images des poumons en deux classes: poumon contaminé par le concert ou non, etc.*). Cependant, dans la plupart des cas, les tailles des données des images sont trop grandes pour être extraites dans un temps raisonnable en utilisant des méthodes standards.

Une nouvelle organisation spatiale de données appelée Band Sequential Organisation (BSQ)[1] et une nouvelle structure de données appelée Peano Count Tree (P-tree) sont proposées par (Perrizo et al. 2001) (Ding Qiang et al. 2002). L'idée principale de P-tree est de diviser récursivement les données en quadrants et enregistrer le nombre de bit à 1 (1-bit) de chaque quadrant dans un arbre. En utilisant la structure P-tree, on peut accélérer le processus de calcul 1-bit. Cela facilite de manière efficace la représentation et l'exploitation de données de type image (Ding Q. et al. 2002).

L'algèbre de P-tree inclut trois opérations de base : Complement, AND et OR. Dans notre méthode, nous utilisons uniquement l'opérateur AND pour faire l'intersection des arbres et par conséquent le calcul des probabilités afin de déterminer l'attribut qui présente un gain d'information le plus important.

Le tableau 1 illustre les règles à respecter pour déterminer l'intersection entre deux P-trees ou sous-P-trees (opérande1 et opérande 2) avec racines X_1 et X_2 respectivement.

Opérande 1	Opérande 2	Résultat
1	X_2	Sous arbre avec la racine X_2
0	X_2	0
X_1	1	Sous arbre avec la racine X_1
X_1	0	0
m	m	**Si** les quatre quadrants ont comme conséquence 0 **alors** 0 ; **Sinon** m

TAB 1 – *Les règles de P-tree ANDing.*

Un PM-tree (Peano Mask tree) est une variante de P-tree, particulièrement utile pour l'optimisation de l'opération ANDing entre deux P-trees. PM-tree consiste à utiliser une logique de 3-valeurs, dans laquelle 1 est employé pour représenter un quadrant pure-1, 0 est employé pour représenter un quadrant pure-0 et m utilisé pour représenter un quadrant mixte. La figure 2 illustre l'opération AND appliquée sur deux arbres P-tree-1 et P-tree-2 de type PM-tree (cf. FIG. 2).

[1] BSQ (Bande Séquentielle) est une forme d'organisation de données proposée par Ding Q. (Ding et al. 2001). Cette forme permet une meilleure compression des données.

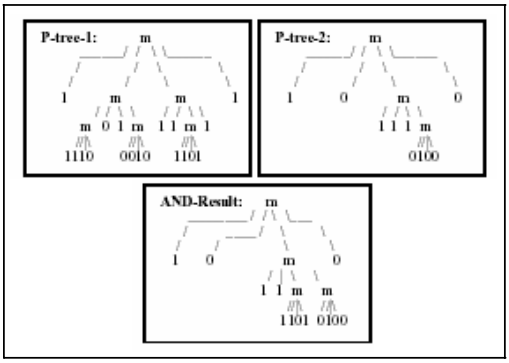


FIG. 2 - *L'intersection de deux P-trees.*

Nous allons voir dans la suite de ce travail la proposition que nous formulons pour classer des images à l'aide de la technique de l'arbre de décision combinée avec la structure P-tree.

4 Classification des images par arbre de décision en utilisant la structure P-tree

Dans notre méthode, chaque image est représentée par un vecteur binaire dérivé à partir des descripteurs visuels. Cette représentation correspond donc à la conversion de la base de données en base binaire, cette dernière sera représentée par un ensemble des P-trees représentant chacun un vecteur binaire. Nous montrons par la suite que la technique P-tree combinée avec l'algorithme C4.5 réduisent considérablement le temps de classification.

4.1 De la base de données à la base binaire

Dans notre étude, la base de données est convertie en une base binaire. Dans cette base de données chaque attribut est associé à un ensemble de valeurs. Par exemple, l'attribut *classe* possède deux items (X et Y) transformés en bits binaires (1 ou 0 pour oui ou non). Donc, les attributs sont caractérisés par des valeurs binaires 0 et 1, représentant la présence d'un item ou son absence (cf. FIG. 3). La liste des items est représentée par l'ensemble $A = \{a_1, a_2, \ldots, a_n\}$ tel que n est le nombre des items dans une base de données.

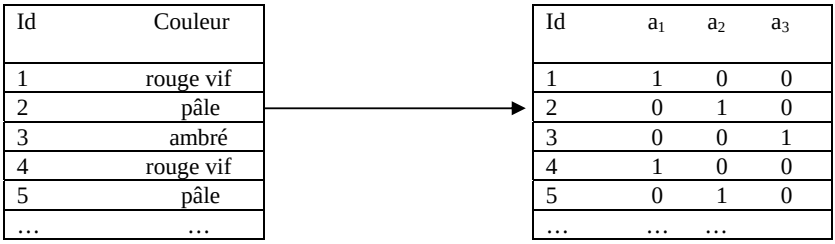
Id	Classe
1	x
2	y
3	x
4	y
…	…

→

Id	a_1
1	1
2	0
3	1
4	0
…	…

FIG. 3 - *Conversion des données pour les attributs binaire.*

Pour un attribut avec un domaine non binaire (multi-valeurs), nous associons à chacune de ses valeurs un item. Par exemple, pour une meilleure représentation de l'attribut *couleur*, nous lui affectons les valeurs suivantes {rouge vif, pâle, ambré}. Le résultat de la conversion des items est défini comme suit : a_1 = rouge vif, a_2 = pâle, a_3 = ambré (cf. FIG. 4).



Id	Couleur
1	rouge vif
2	pâle
3	ambré
4	rouge vif
5	pâle
…	…

Id	a_1	a_2	a_3
1	1	0	0
2	0	1	0
3	0	0	1
4	1	0	0
5	0	1	0
…	…	…	

FIG. 4 - *Conversion des données pour les attributs non binaire.*

4.2 Construction de P-tree

Pour simplifier le travail, nous supposons que le fan-out (les nœuds sortants) est quatre. Pour chaque vecteur de bits, nous associons un P-tree. Chaque P-tree présente un nombre de bits divisible par 4, donc, il est constitué par quatre sous quadrants dont le quadrant d'origine représente la totalité des bits formant un attribut (ou un item) de la table binaire. Il y a toujours n P-trees de base pour une table binaire formée de n items. L'idée de construction d'un P-tree consiste à diviser périodiquement le vecteur binaire en des sous quadrants et d'enregistrer le nombre de bits à 1 pour chaque sous quadrant, en formant de ce fait, un arbre de calcul des quadrants.

4.3 Stockage de P-tree

D'une manière générale, la génération de P-tree à partir des trames de données se fait d'une manière ascendante et le stockage de chaque arbre se fait dans un vecteur des entiers. La génération de P-tree dépend du nombre de fan-out dans les nœuds internes de l'arbre P-tree et dans le nœud racine. Pour représenter P-tree avec différents fan-out, nous utilisons la notation suivante P-tree (r-i) ; où r = fan-out du nœud racine ; i = fan-out de tous les nœuds internes du niveau 1. Dans notre cas, nous utilisons la représentation P-tree (4-4-n) c à d nous divisons le nombre de tuples composant la base de données par 4 blocs (les blocs doivent avoir au minimum quatre tuples). Si le nombre de tuples est inférieur à 16, nous complétons par des 0 pour obtenir le format P-tree sur 16 tuples. D'une manière générale, si le nombre de tuples est compris entre 2^{n+1} et 2^n alors le P-tree de base est stocké avec un nombre de tuples égale à 2^{n+1}.

Dans la procédure de stockage de P-tree, nous stockons les nœuds racines de P-tree dans un vecteur et les nœuds binaires (formant les quadrants) dans un autre vecteur afin de faciliter l'accès aux données des sous quadrants de chaque bande (colonne de la table binaire), en vue de les comparer avec les sous quadrants d'autres bandes pour déterminer la probabilité de chaque item.

Soit Nt = nombres de tuples ; n est initialisé à 3 et I désigne le nombre total des items, nous considérons l'algorithme suivant :

Procédure Stockage_de_P-tree

```
Pour (bj = 1 ; j <I ; j++)
      racine[j] := rootcount (1 ; bandj) // vecteur qui stocke les racines des P-trees.
      Si (2^n ≤ N_t et N_t < 2^(n+1)) alors
         Pour (i = N_t; i≤2^(n+1); i++)
            bandj[i+1]:= 0;
         FinPour
      FinSi
      Pour (i =1; i≤2^n /4; 2^n /4) // calcul des racines et sous-quadrants.
            rootbandj[i] := rootcount (1 ; bandj); // vecteur racines internes
      FinPour
      Pour (i =1; i≤2^n ; 2^n /4)
         Si (rootbandj[i] <>2^n ou rootbandj[i] <>0) alors
            Bitsbandj[i] := subquadrant_i;//vecteur des bits
         FinSi
      FinPour
FinPour
```

4.4 Génération des P-trees de base

Après avoir classé les données, la question qui se pose est la suivante : comment produire les P-trees ?

Nous avons n tuples et nous voulons choisir le fan-out de P-tree en tant que 4.

- balayer les données une fois pour chaque 4 tuples. Ajouter un nœud additionnel dans P-tree. Après chaque 4 nœuds produits, ajouter un nœud interne comme nœud parent et ainsi de suite.
- de cette façon, P-tree est produit du bas vers le haut et de gauche vers la droite tout en balayant les données de la base de données convertie (cf. FIG. 5).

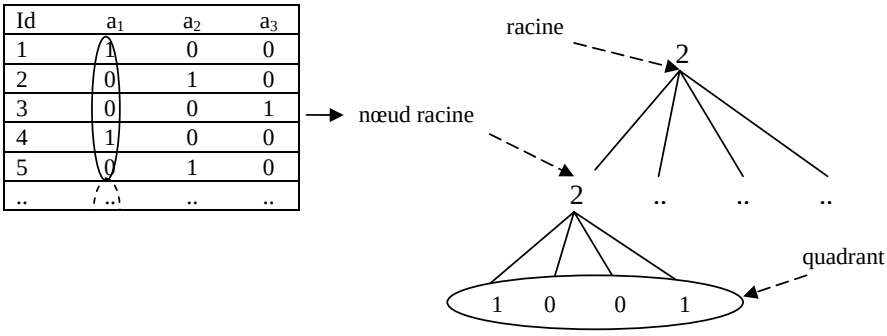


FIG. 5 - *Génération de P-tree de base pour l'Item a_1.*

4.5 L'algorithme C4.5(P-tree)

Une des étapes difficiles de cet algorithme est de choisir le prochain attribut. Si on choisit un attribut qui résulte en une partition dont tous les exemples se retrouvent dans le même sous-ensemble, alors on augmente la taille de l'arbre sans augmenter l'habileté à classer les exemples. En classant les éléments, on veut rejoindre une feuille contenant un ou des exemples en posant le moins de questions; on veut donc que l'arbre soit le plus petit possible. Puisque c'est difficile de trouver l'arbre du chemin le plus court parmi tous les arbres possibles, on construit l'arbre de haut en bas en utilisant une heuristique permettant de choisir l'attribut le plus déterminant (le plus informatif). Il existe plusieurs techniques tels que Index Gini, X^2, théorie de l'information. C'est ce que nous allons utiliser dans la suite de ce travail.

Algorithme C4.5(P-tree)

Si tous les éléments du nœud sont dans la même classe
Alors quitter.
Sinon /*Faire les étapes suivantes*/
1. Choisir un attribut et l'assigner au nœud /* Gain d'information déterminer à partir des P-trees*/
2. Partitionner les exemples dans des sous-ensembles selon les valeurs de cet attribut
3. Créer des nouveaux nœuds pour chaque sous-ensemble non vide de la partition
4. La liste de nouveaux noeuds devient les enfants du nœud.
5. Appliquer la procédure récursivement sur les nouveaux nœuds.

FinSi

Au début, l'arbre de décision est un noeud simple représentant l'ensemble entier d'apprentissage. Si tous les échantillons sont dans la même classe, ce noeud devient une feuille et est marqué avec cette étiquette de classe. Une mesure basée par entropie " gain de l'information ", est employée comme une heuristique pour sélectionner l'attribut qui sépare mieux les échantillons dans les différentes classes (l'attribut de décision). Une branche est créée pour chaque valeur de l'attribut d'essai et des échantillons sont divisés en conséquence. L'algorithme avance périodiquement pour former l'arbre de décision pour le sous échantillon réglé à chaque nouveau nœud. L'algorithme s'arrête quand tous les échantillons pour un noeud donné appartiennent à la même classe ou quand il n'y a aucun attribut restant. L'attribut choisi à chaque niveau d'arbre de décision est celui avec le gain le plus élevé de l'information.

4.5.1 Calcul des probabilités et valeur de l'entropie à l'aide de la structure P-tree

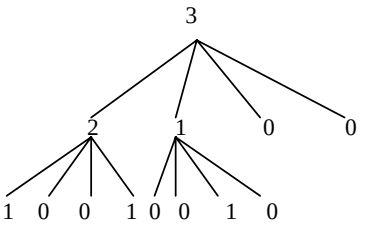
Dans cet exemple, nous allons essayer de calculer le gain d'information pour l'attribut couleur. Il est à noter que cet attribut contient trois items (*rouge vif, pâle, ambré*). Ce travail sera répété pour les autres attributs qui sont : la *texture* et la *forme*. Une fois la phase de calcul de la valeur de gain d'information pour chaque attribut est achevée, l'attribut qui présente la plus grande valeur sera choisi comme meilleur attribut de classification (racine de

l'arbre) et nous allons partitionner tous les tuples en sous-ensembles (nœuds) selon la valeur du meilleur attribut retenu.

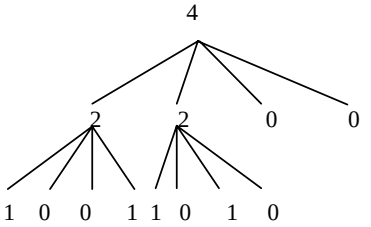
Id	rouge vif = a_1	pâle = a_2	ambré = a_3	Classe X = a_4	Classe Y = a_5
1	1	0	0	1	0
2	0	1	0	0	1
3	0	0	1	0	1
4	1	0	0	1	0
5	0	1	0	1	0
6	0	0	1	0	1
7	1	0	0	1	0
8	0	1	0	0	1
..	..	..	..	..	..

TAB 2 - *Tableau binaire de l'attribut couleur.*

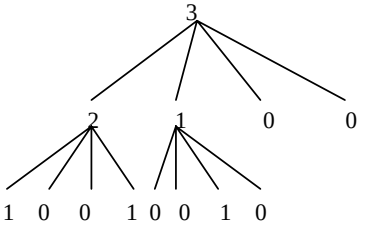
A partir de cette table binaire (cf. TAB. 2), nous allons générer des arbres P-trees associés chacun à un item bien défini (cf. FIG. 6). Par exemple, dans l'arbre P-tree1, la valeur 2 du nœud racine est obtenue en comptant le nombre de bits à 1 dans le premier sous quadrant. De même pour la valeur 1 du deuxième nœud racine qui correspond au deuxième sous quadrant. La valeur 3 (2+1) de la racine est obtenue par addition des valeurs de deux nœuds racines.



P-tree1 de base pour l'item rouge vif



P-tree 4 de base pour l'item classe X



Résultat de P-tree1 ANDing P-tree4

FIG. 6 - *Résultats des intersections des P-trees (ANDing).*

Le calcul des probabilités sera déduit automatiquement à partir des P-trees. Par exemple, pour déterminer la valeur de probabilité d'apparition de la couleur rouge vif (3/8), on divise la valeur de la racine de l'arbre P-tree1 (égale à 3) par le nombre total des enregistrements de

la table binaire (égale à 8). Les valeurs des probabilités conditionnelles sont obtenues par application de l'opérateur AND sur deux P-trees. Par exemple, pour calculer la valeur de P(X|rouge vif)= 3/4 , on divise la valeur de la racine de l'arbre P-tree1 ANDing P-tree4 (égale à 3) par la valeur de la racine de l'arbre P-tree4 (égale à 4). Pour calculer l'entropie de l'attribut couleur, nous appliquons la formule suivante :

Soit une conclusion C pouvant valoir c_1, c_2, …, c_n. Dans notre exemple n = 2. Soit un attribut A pouvant valoir a_1, a_2, …, a_m. La probabilité que C vaut c_i sachant que A vaut a_j est $p(c_i|a_j)$. La quantité d'information de a_j sur c_i est notée : $-\log_2(c_i|a_j)$. L'entropie de a_j par rapport à la conclusion C est notée $E(a_j) = -\sum p(c_i|a_j) \log_2(c_i|a_j)$. Le gain d'information est calculé comme suit : Gain (A) = E(C) – E (A) (cf. TAB. 3).

Probabilités des classes	***Entropie de l'attribut classe C (X et Y)***
P(X) = 4/8 (P-tree4) P(Y) = 4/8 (P-tree5)	E (C) = $-(4/8\log_2(4/8) + 4/8 \log_2(4/8))$ = 0,301
Probabilités d'apparition	***Probabilités conditionnelles***
P(rouge vif) = 3/8 (P-tree1); P(pâle) = 3/8 (P-tree2); P(ambré) =2/8 (P-tree3)	P(X|rouge vif) = 3/4 (P-tree1 ANDing P-tree4); P(X|pâle) =1/4 (P-tree2 ANDing P-tree4); P(X|ambré) = 0 (P-tree3 ANDing P-tree4); P(Y|rouge vif) = 0 (P-tree1 ANDing P-tree5); P(Y|pâle) = 2/4 (P-tree2 ANDing P-tree5); P(Y|ambré) = 2/4 (P-tree3 ANDing P-tree5)
Entropie de l'attribut couleur	***Gain d'information de l'attribut couleur***
E(Couleur) = - **[**3/8 E(rouge vif) + 3/8 E(pâle) + 2/8 E(ambré)**]** = - **[**3/8 **(**3/4 $\log_2(3/4)$ + 0**)** + 3/8 **(**1/4 $\log_2(1/4)$ + 2/4 $\log_2(2/4)$ + 2/8 **(**1/4 $\log_2(1/4)$ + 0**)]**= 0,185	Gain(Couleur) = E(C) – E(Couleur) = 0,301 – 0,185 = 0,116

TAB 3 – *Calcul de gain d'information de l'attribut couleur à l'aide des P-trees.*

5 Implémentation et validation

La base de données d'évaluation des fruits de fraise contient des exemples réels représentant des images de fraise avec quatre attributs. Trois attributs représentent la partie prémisse de la règle de classification et un attribut pour la partie conclusion. Les valeurs des attributs sont définies comme suit :

Valeurs des attributs de décision :
Classe : fraise à petits fruits, fraise à gros fruits.
Valeurs des attributs de prémisse :
Couleur : pâle, rouge vif, ambré.
Texture : lisse, caillé.
Forme : allongé, oblongue (au forme du cœur), conique.

La comparaison des résultats de classification obtenus avec l'algorithme C4.5(P-tree) d'une part et l'algorithme C4.5 d'autre part montre une amélioration des performances en utilisant la structure P-tree. Une étude comparative entre les deux algorithmes est présentée par la figure suivante (cf. FIG. 7).

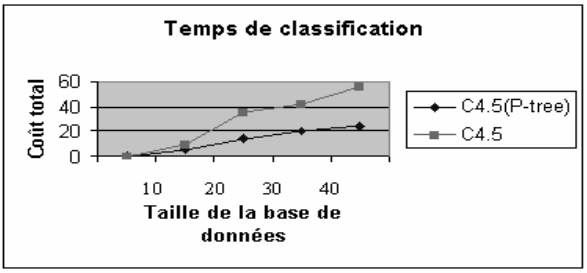


FIG. 7 - *Comparaison entre l'algorithme C4.5(P-tree) et l'algorithmeC4.5 en temps de classification (en seconde).*

Une nette amélioration des performances est à noter. Cette amélioration atteint 30% du temps de classification (0,30). En effet, la structure de données utilisée pour représenter la base de données est un ensemble de P-trees représentant un vecteur binaire de la table binaire et tous ces P-trees sont stockés dans des fichiers binaires. Cette structure de données permet de charger complètement la base des images (base volumineuse) dans un fichier binaire. L'objectif de cette opération est d'éviter le balayage direct de la base d'images.

A travers l'application que nous avons développée, nous avons montré que l'algorithme C4.5 basé sur la structure de donnée P-tree offre un gain de temps considérable par rapport à l'algorithme C4.5 pour la classification de la même base de données. Enfin, la technique P-tree combinée avec l'algorithme C4.5 peut offrir un temps minimum de classification comparée avec l'algorithme C4.5.

6 Conclusion et perspectives

Nous avons présenté une méthode binaire de classification. L'algorithme proposé C4.5(P-tree) est implémenté dans un système baptisé C.I.A.D.P-tree. Le calcul des probabilités pour choisir le meilleur attribut a été basé sur des règles de comparaison des P-trees ($P\text{-tree}_i$ ANDing $P\text{-tree}_j$).

Notre système présente plusieurs avantages car, d'une part, il évite le balayage direct de la base de données qui est une opération assez coûteuse en mémoire et en temps d'exécution et, d'autres part, il offre un gain de comparaison des attributs. En effet, la comparaison s'effectue par bloc des tuples dans la base binaire.

Les perspectives que nous proposons concernent également l'amélioration de notre système de classification. Il s'agit de développer une interface qui permettrait à l'utilisateur de déterminer directement les vecteurs d'attributs associés aux images au lieu de les saisir. L'algorithme proposé pourrait par ailleurs être comparé à d'autres algorithmes de classification et appliqué sur d'autres bases de données.

Références

Bonnet A. (1984), L'Intelligence Artificielle : promesses et réalités, Inter éd., Paris, 1984.

Borgi A. et Akdag H. (2001), Apprentissage supervise et raisonnement approximatif, l'hypothèse des imperfections, Revue d'IA, Vol. 15, N°. 1, pp 55-85, 2001.

Breiman L., Friedman J.H., Olshen A. et Stone C.J. (1984), Classification of regression trees, Champan and Hall, 1984.
Carbonell J.G., Michalski R.S. et Mitchell T.M. (1986), An Overview of Machine Learning, Machine Learning: an Artificial Intelligence Approach, R.S. Michalski, J.G. Carbonell, T.M. Mitchell editors, Morgan Kaufmann, 1986.
Chahir Y. et Chen L. (1999), Spatialized Visual Features Based Image Retrieval, Inter. Jour. of Computers and Their Applications, Vol. 6, N°. 4, pp. 190-199, December 1999.
Dietterich T.G. et Michalski R. S. (1983), A Comparative Review of Selected Methods for Learning from Examples, Machine Learning, An Artificial Intelligence Approach, Vol. 1, pp. 41-81, Morgan Kaufmann, 1983.
Ding Q., Khan M. et Perrizo W. (2001), The P-tree Algebra, Proc. of ACM SIGKDD, 2001.
Ding Q., Ding Qiang et Perrizo W. (2002), Association Rules Mining on Remotely Sensed Images using P-trees, In Proceedings of the PAKDD, Taipei, Taiwan, pp. 66-79, 2002.
Ding Qiang, Ding Q. et Perrizo W. (2002), Decision Tree Classification of Spatial Data Streams Using Peano Count Tree, Technical Report NDSU-CSOR-TR-01-1, 2002.
Kodratoff Y. et Diday E. (1991), Préambule: Approche symbolique et Approche Numérique, Induction Symbolique et numérique à partir de Données, Ed. Cépaduès, 1991.
Marsala C. (1998), Apprentissage inductif en présence de données imprécises: construction et utilisation d'arbres de décision flous, Thèse de doctorat de l'univ. Paris 6, Janv. 1998.
Mellouli K., Elouedi Z. et Smets P. (2001), Belief decision trees: Theoretical foundations, Inter. Jour. of Approximate Reasoning, pp. 91-124, Compiègne, France: AS-MDA 2001.
Michalski R.S. et Ryszard S. (1983), A theory and methodology of inductive learning, An A.I. Approach, Morgan Kaufmann, 1983.
Perrizo W., Ding Q., Ding Qiang et Roy A. (2001), Deriving High Confidence Rules from spatial data using Peano Count Trees. In advances in Web-Age Information Management: Second International Conference WAIM 2001, Wang X.S., Yu G. and Lu H. (Eds), Springer-Verlag, LNCS 2118, pp.91-102, 2001.
Quinlan J.R. (1983), Learning efficient classification procedures and their application to chess and games: An Artificial Intelligence Approach, Vol.1, pp 463-482, Morgan Kaufman Publishers, 1983.
Quinlan J.R. (1993), C4.5 : Programs for Machine Learning, Morgan Kaufmann, 1993.
Ramdani M. (1994), Système d'Induction Formelle à base de connaissances Imprécises, Thèse de doctorat de l'université Paris 6, 1994.

Summary

We propose a new method of classification starting from images which is at the junction of two techniques: P-tree algebra and the decision tree. Such an approach is necessary to accelerate the process of classification and research in great bases of images. Our modelling is based, on the one hand, on the visual descriptors such as the colour, the form and texture for the indexing of the images and, on the other hand, on the automatic generation of the rules of classification. The approach of training proposed is supervised, it is based on a whole of training made up of objects whose classes are known a priori. A system baptized C.I.A.D.P-tree was implemented and confronted with a real application in the field of the image processing. The results obtained make it possible to validate our method and authorize us to carry out experimental tests on various data bases. It would be interesting to then use other descriptors to build the rules of classification.