ALGORITHMES POUR L'IMPLÉMENTATION D'UN SYSTÈME D'ÉTIQUETAGE AUTOMATIQUE DES IMAGES

Gilles Daigle, Shengrui Wang, Djemel Ziou
Laboratoire de vision et traitement d'image,
Université de Sherbrooke,
Sherbrooke, Québec, Canada, J1K 2R1

Résumé

La révolution informatique a mis à notre disposition un nombre phénoménal de bases de données de toutes sortes. Il est facile de rechercher un texte quelconque à l'aide de mots-clés, mais le domaine émergeant de l'audiovisuel pose un problème de taille. Une solution à ce problème est d'attacher une description textuelle à l'item problématique.

La présente recherche vise à la création d'un système pour l'étiquetage automatique des images. Ce domaine n'est pas entièrement nouveau, mais les systèmes proposés jusqu'ici n'en sont qu'à l'état expérimental. Nous nous sommes donc donnés comme critère de base que le système à développer devait paraître professionnel, être rapide et donner de bons résultats.

Abstract

Content-based image retrieval is emerging as an important research area with potential applications in many domains, notably multimedia databases and digital libraries. The goal of the present research is to create a usable system that performs automatic picture labeling. Recent studies have shown that some improvements can be achieved with a system which selects the best from amongst many models of representation. The focus of our research is thus twofold: creation of models and selection of the best ones for each label.

1 Introduction

La recherche d'images basée sur leur contenu émerge aujourd'hui comme un thème de recherche important ayant de nombreuses applications, notamment dans les bases de données multimédia et les bibliothèques numériques. En effet, avec la vaste quantité d'informations disponibles aujourd'hui, l'utilisation d'index et même d'outils de recherche devient cruciale. L'internet nous en offre un exemple typique. Nous n'avons jamais eu autant d'informations si facilement disponibles, mais trouver l'information *utile* devient un défi. De nouveaux outils de recherche et d'indexage apparaissent à tous les jours. Ces outils manipulent l'information textuelle pour faire leur travail. Cependant, l'internet devient de plus en plus un médium audiovisuel et indexer ce genre d'information est tout un problème.

Il y a beaucoup de recherche dans les systèmes d'extraction de données visuelles, mais beaucoup de ces systèmes exigent que la base de données visuelles et l'outil de recherche se retrouvent tout deux dans le même système informatique. Une meilleure approche serait de pouvoir utiliser les outils textuels que l'on retrouve partout sur l'internet dans le but d'intégrer la recherche d'images avec la recherche de textes. Pour cela, l'information visuelle doit être traduite en un type d'information textuelle utilisable par les outils de recherche.

Présentement, cette traduction se fait par une personne qui soit écrit une description de l'image, soit en retire quelques mots-clés. Si l'on restreint le domaine aux images, une approche pour créer cette information automatiquement est d'étiqueter tous les éléments de l'image. Ceci nous donne une série de mots-clés se rapportant à l'image et même un pourcentage de l'image se rapportant aux mots-clés. Cette méthode a aussi l'avantage d'être moins subjective qu'une description de l'image par une personne.

Il faut souligner ici un point important: il y a un degré de subjectivité dans la création de mots-clés. Loin d'être un inconvénient, nous croyons que ceci est un avantage pour le genre de système que nous proposons. En effet, ceci veut dire que nous pouvons tolérer, sans trop de problème, un certain pourcentage d'erreur dans un système automatique d'étiquetage.

Nous proposons un système d'étiquetage à deux phases. Dans la première phase, l'usager instruit le système à l'aide d'une série d'images typiques (voir fig. 6 à la fin de l'article). C'est la phase d'apprentissage. Dans la deuxième phase, le système peut être utilisé pour faire l'étiquetage d'un nombre illimité d'images. Le résultat est, pour chaque image, un fichier contenant les étiquettes que le système propose

pour l'image. Ces fichiers d'étiquettes peuvent ensuite être utilisés par un autre système.

L'étiquetage automatique de l'image est un nouveau domaine de recherche et un article récent du MIT [7] a montré que des progrès pouvaient être accomplis avec un système qui sélectionne le meilleur modèle de représentation parmi plusieurs. Nous avons choisi, comme base, ce système qui semblait prometteur dans le but d'y apporter des améliorations et d'en faire un système à deux phases. La première partie de la présente recherche fut donc de recréer le système décrit dans l'article. Dès le début de la programmation, il devint évident que la méthode d'agrégation utilisée par le groupe du MIT demandait beaucoup de temps de calcul. Ceci peut limiter l'utilité du système d'étiquetage s'il est utilisé avec une base de données très grande.

Le premier défi fut donc d'améliorer la méthode d'agrégation. La méthode utilisée par le groupe du MIT est une modification de la méthode de Jarvis-Patrick [5]. Nous proposons ici une nouvelle méthode: l'arbre des distances. Cette méthode est beaucoup plus rapide que la méthode modifiée de Jarvis-Patrick et semble donner des résultats acceptables.

2 Principes de l'étiquetage

Poser une étiquette sur une image (ou une imagette) est essentiellement un problème de classification. Généralement, un tel système fonctionne comme il est indiqué à la figure 1. Une étude récente [6] de différents systèmes de classification qui fonctionnent selon ce principe donne un pourcentage de classification correcte qui se situe en général dans les 80%. Cependant, il existe un problème important dans ce genre d'approche. Certaines caractéristiques peuvent être excellentes pour certains types d'images, mais donneront des résultats douteux pour un autre type d'images. Nous ne pouvons pas, tout simplement, additionner toutes les caractéristiques car ceci nous conduit à une dégradation des résultats, ce que l'on appelle communément la «malédiction de la dimension» (curse of dimensionality).

1. Extraction de certaines caractéristiques de l'objet.
2. Construction d'un vecteur de caractéristiques.
3. Apprentissage: création de **classes** de vecteur.
4. Recherche: comparaison d'un nouveau vecteur avec ceux des classes existantes.

Figure 1: Étapes usuelles pour la classification d'un objet.

Au coeur du système décrit ici, est un mécanisme qui fait automatiquement un choix, pour chaque étiquette, parmi plusieurs sous-ensembles de caractéristiques de l'image. Ces caractéristiques peuvent correspondre à la distribution des couleurs, au contraste, à la rugosité de l'image, à la directionnalité, à la périodicité, etc. Dans ce document, nous appelons un sous-ensemble spécifique de ces caractéristiques un **modèle**.

2.1 Aperçu du fonctionnement

Pour générer des étiquettes à partir d'une image, le système doit suivre les étapes suivantes (voir fig. 2):

Premièrement, l'image doit être divisée en parties appelées imagettes. L'imagette est l'objet de base de notre système et les imagettes seront étiquetées manuellement par l'usager durant la période *d'apprentissage* ou automatiquement par le système durant la période *d'auto-étiquetage*.

Dans l'étape deux, l'usager ajoute les étiquettes sur les imagettes qui doivent servir d'étalonnage au processus d'étiquetage (étape 3c). L'algorithme d'étiquetage exige qu'il y ait, dans un agrégat, au moins deux imagettes qui ont reçu une étiquette identique de l'usager pour que la génération d'étiquette se fasse dans cet agrégat.

1. Diviser une image en petites imagettes.
2. Ajouter aux imagettes les étiquettes spécifiées par l'usager.
3. Pour chaque modèle dans le système:
   (a) Calculer une représentation du modèle pour chaque imagette.
   (b) Créer un arbre hiérarchique d'agrégats avec les vecteurs du modèle.
   (c) Dégager individuellement les agrégats de l'arbre, proposer des étiquettes où il en faut.
4. Comparer toutes les propositions des différents modèles (étape 3 a-c) pour trouver les *meilleures*.
5. Imprimer (ou afficher) les résultats.

Figure 2: Étapes pour le système d'étiquetage d'images.

L'étape trois se divise en trois sous-étapes qui se répètent pour chaque modèle (e.g. *chaque sous-ensemble de caractéristiques*) alors que l'étape quatre fera l'intégration de l'information ainsi obtenue. Si l'usager choisit de n'exécuter qu'un seul modèle, pour en vérifier l'efficacité par exemple, l'étape quatre sera sautée.

L'Étape 3a consiste en la construction des vecteurs de caractéristiques tel que mentionné dans la figure 1.

L'étape 3b consiste en la création d'un arbre hiérarchique d'agrégats. Les vecteurs de l'image en traitement sont combinés à tous les vecteurs stockés dans le système (vecteurs d'apprentissages).

L'Étape 3c, consiste à extraire de l'arbre hiérarchique d'agrégats les plus gros agrégats qui ne contiennent qu'un seul type d'étiquette. Toutes les imagettes appartenant à cet agrégat recevront ensuite cette étiquette.

La quatrième étape consiste en une sélection des agrégats *gagnants* parmi tout ceux qui ont été générés dans l'étape trois. Cette sélection n'est pas globale, mais se fait pour chaque étiquette. L'étape consiste en la sélection d'un minimum d'agrégats qui regroupent toutes les imagettes.

Enfin, dans la dernière étape, les résultats pour chaque image sont sauvegardés dans des fichiers se rapportant à l'image et selon le cas, les étiquettes affichées à l'écran seront mises à jour.

Dans les sections qui suivent, certaines de ces étapes seront reprises avec plus de détails.

2.2 Regroupement d'imagettes (agrégation)

L'étiquetage est une opération subjective qui ne correspond pas nécessairement à une simple agrégation. Par exemple, le ciel peut être bleu ou même orange, le lac peut être bleu ou blanc s'il y a de l'écume, la texture des arbres varie d'un endroit à l'autre, etc... Nous pouvons voir ici qu'une étiquette spécifique se retrouvera dans plusieurs agrégats séparés dépendant des variations dans le type d'objets en question.

D'un autre coté il est possible, car aucun algorithme d'agrégation n'est parfait, qu'un beau gros agrégat *ciel* se retrouve avec un étiquette *lac* à l'intérieur. Il serait dommage de rejeter l'agrégat au complet. Une meilleure solution serait de séparer l'agrégat en trois parties: deux agrégats ciel et un petit agrégat lac.

L'utilisation d'un arbre hiérarchique d'agrégats résout ces problèmes. Premièrement, pour créer l'arbre il n'est pas nécessaire de spécifier le nombre d'agrégats voulus, l'arbre contient un continuum d'agrégats. Deuxièmement, une fois l'arbre complété, nous pouvons utiliser une autre technique (voir 2.3) pour sélectioner les agrégats qui ont les caractéristiques et la grosseur qui conviennent.

2.2.1 Création de l'arbre

La création d'un arbre hiérarchique d'agrégats demande beaucoup de calcul et c'est l'un des domaines où notre approche diffère substantiellement de la méthode d'agrégation préconisée par l'équipe du MIT [7]. Ce problème de temps de calcul se fait surtout sentir lors de la phase d'apprentissage qui nécessite une interaction directe entre l'utilisateur et le système.

L'équipe du MIT utilise une adaptation de la méthode des voisins-communs telle que décrite dans [4]. Cette méthode, créée par Jarvis et Patrick, fut l'une des premières à être publiée (en 1973) et est donc utilisée couramment. Avec cette méthode, deux points sont regroupés ensemble s'ils ont un certain nombre de voisins en commun. Il est à noter que c'est une méthode **non** hiérarchique. Il est possible de la transformer en méthode hiérarchique en augmentant progressivement le nombre de voisins et en regroupant les agrégats ainsi formés, mais ceci multiplie le nombre de calculs à faire.

L'application de cette méthode d'agrégation au système d'étiquetage d'images soulève deux problèmes principaux:

La distance entre tous les points doit être calculée.

De nouveaux points ne peuvent être ajoutés sans refaire tous les calculs.

L'équipe du MIT ont ignoré ces problèmes en spécifiant que la création de l'arbre doit être faite séparément dans une phase de pré-traitement.

Pour résoudre ces problèmes, nous avons créé une nouvelle méthode d'agrégation hiérarchique que nous appelons **l'arbre des distances**.

L'arbre des distances résout les deux problèmes principaux de la méthode d'agrégation des voisins-communs en faisant les comparaisons sur un arbre binaire et non entre tous les points et en permettant d'ajouter chaque point un à un. D'un autre coté, l'ordre de présentation des points semble important et l'agrégation ne semble pas aussi bon. Nous avons donc sacrifié quelque peu la précision pour un gain appréciable de vitesse.

Vu l'importance de ce sujet dans notre recherche, la section 3 se consacre entièrement à l'étude de l'arbre des distances.

2.3 Extraction des agrégats et étiquetage

Une fois en possession de l'arbre hiérarchique d'agrégats nous pouvons utiliser l'information que nous donne les étiquettes connues pour extraire de l'arbre les agrégats désirés. L'algorithme d'extraction des agrégats est récursif et fait un parcours linéaire de l'arbre de gauche à droite. L'algorithme reçoit comme argument la racine de l'arbre et le nom de l'étiquette à extraire. Le résultat est une liste d'agrégats.

Quelques explications sont de mise. La variable S contient la solution trouvée à date. Une solution peut être imprimée (avec imprime(S)) ou être remplacée par une solution plus générale. Les valeurs de retour de l'algorithme sont: bon, mauvais, neutre selon que les branches contiennent de bonne, mauvaise ou pas d'étiquette. Une bonne étiquette est évidemment une étiquette qui est conforme à l'étiquette à extraire.

```
Extrait_racine( racine, étiquette )
  bonne_étiquette = étiquette
  si Extrait_branche( racine ) == bon alors imprime(S)
fin
Extrait_branche( branche )
  si branche == feuille alors
    si feuille.étiquette == bonne_étiquette
      retourne bon
    si feuille.étiquette == pas_étiquette
      retourne neutre
    sinon retourne mauvais
  gauche = Extrait_branche( branche.gauche )
  droite = Extrait_branche( branche.droite )
  si gauche == bon alors
    si droite == bon { S = branche; retourne bon }
    si droite == neutre { S = gauche; retourne bon }
    sinon { imprime( S ) ; retourne mauvais }
  si gauche == neutre retourne droite
  si gauche == mauvais alors
    si droite == bon alors S = droite
    retourne mauvais
fin
```

Figure 3: Algorithme récursif d'extraction des agrégats.

2.4 Algorithme de sélection globale

Chaque série d'agrégats (une pour chaque modèle) créée à la section précédente est en soi une solution au problème d'étiquetage. Une étiquette peut se retrouver dans des agrégats passablement différents dépendant des variations dans l'objet étiqueté. Il est donc possible qu'un seul modèle ne fasse pas l'affaire pour une étiquette particulière. Pour cette raison, la sélection ne se fait pas directement au niveau modèle, mais plutôt au niveau des agrégats générés par les modèles. En d'autres mots, nous utilisons le meilleur de chaque modèle.

Nous utilisons présentement une approche exhaustive pour la sélection des agrégats. Cette approche exhaustive est combinée avec des heuristiques pour réduire le champs de recherche. L'algorithme présenté à la figure 4 va nous donner en sortie la liste d'agrégats qui satisfait à nos critères.

3 Arbre des distances

Comme il est souligné dans l'introduction, nous soumettons dans ce projet une nouvelle méthode d'agrégation. C'est une méthode agglomérative géométrique, mais jusqu'ici nous n'avons rien trouvé d'identique dans la littérature. L'agrégation est utilisé dans plusieurs domaines (astronomie, biologie, chimie, démographie, économie, ...) et

1. Regrouper tous les agrégats de tous les modèles qui ont la même étiquette.
2. Faire un tri des agrégats en ordre décroissant du nombre d'éléments qui ont une étiquette.
3. Enlever les agrégats redondants et les singletons (heuristiques 1 et 2).
4. Faire une liste des imagettes qui ont une étiquette.
5. Comparer, dans l'ordre, toutes les combinaisons d'agrégats avec la liste précédente. Arrêter l'étape quand le nombre d'éléments qui reste est trop petit pour remplir la liste (heuristique 3). Arrêter l'algorithme quand le nombre d'agrégats dans la solution devient trop grand (heuristique 4). Si toutes les imagettes de la liste de l'étape 4 se retrouvent dans la combinaison nous avons la solution et l'algorithme s'arrête.

Figure 4: Algorithme de sélection globale.

la littérature sur le sujet se trouve donc éparpillée un peu partout. Beaucoup de livres en parlent, mais il y a très peu de livres qui se concentrent uniquement sur ce sujet. Nous avons soumis notre algorithme à plusieurs tests pour savoir s'il était capable de faire une bonne agrégation et s'il n'y avait pas de problèmes importants.

3.1 Algorithme

L'arbre des distances pourrait être classé comme une méthode d'agrégation hiérarchique par agglomération. La méthode s'inspire de l'idée des centres de concentration que l'on retrouve dans les réseaux de neurones. Étant donné une liste de points dans un espace vectoriel, l'algorithme (fig. 5) va créer un arbre binaire en fonction de la distance entre les points.

Chaque noeud de l'arbre contient les informations suivantes: un point dans l'espace vectoriel, un rayon, un poids, et possiblement une étiquette. Les noeuds terminaux (les feuilles) représentent les points d'entrée (les imagettes). Une feuille a un poids de 1, un rayon de 0, et peut avoir une étiquette. Les noeuds non terminaux représentent des sphères regroupant les feuilles. Le poids de ces noeuds est égal au nombre de feuilles qu'il englobe.

La construction de l'arbre se fait une feuille à la fois. L'algorithme a comme données d'entrée la racine d'un arbre et une feuille. La feuille est ajoutée à l'arbre.

L'algorithme a besoin de deux fonctions:

Distance(a, b) Calculer la distance entre deux points et selon la formule usuelle:

$$d = \sqrt{\sum_i (\ _i - \ _i)^2}$$

Nouveau_point(a, b) Calculer un nouveau point qui est sur la droite qui passe par les deux points donnés. La position de ce nouveau point sur la droite dépend d'un poids qui est attaché à chaque point. Par exemple, si le poids P_a est égal à 1 et le poids $P_b = 2$, le nouveau point sera au $2/3$ sur la droite entre et .

$$\frac{P_a + bP_b}{+ P}$$

```
Arbre_distance( branche, feuille )
  si branche.poid == 0 retourne feuille
  si branche.poid == 1 alors
    retourne Nouveau_point( branche, feuille )
  si Distance( branche, feuille ) > branche.rayon alors
    retourne Nouveau_point( branche, feuille )
  si Distance( branche.gauche, feuille ) <
    Distance( branche.droite, feuille )
    alors branche.gauche =
      Arbre_distance( branche.gauche, feuille )
    sinon branche.droite =
      Arbre_distance( branche.droite, feuille )
  branche.point =
    Nouveau_point( branche.gauche, branche.droite )
  branche.rayon =
    max( Distance( branche, branche.gauche ),
      Distance( branche, branche.droite ) )
  retourne branche
fin
```

Figure 5: Algorithme récursif de l'arbre des distances.

Comme l'arbre des distances est un arbre binaire, le nombre de noeuds est égal au nombre de feuilles moins un. Le vecteur de caractéristiques est le seul élément possiblement encombrant de ces noeuds et feuilles. L'espace utilisé par l'arbre est donc modeste surtout si comparé aux méthodes où il faut calculer une matrice de distance entre tous les points. L'arbre peut donc facilement être stocké sur disque pour être réutilisé lors de l'ajout de nouveaux points.

3.2 Résultats

Nous avons utilisé des agrégats artificiels pour vérifier la précision de l'agrégation et la vitesse d'exécution pour la méthode de Jarvis-Patrick et la méthode de l'arbre des distances. Chaque configuration contient exactement 64 points pour que toutes les configurations puissent être comparées entre elles.

Les tests préliminaires, faits au début du projet avec quelques agrégats différents, semblaient indiquer que les deux méthodes donnaient le même genre de précision. Les derniers tests indiquent que l'arbre des distances fonctionne bien quand il y a peu d'agrégats ou que les agrégats sont séparés par de grandes distances relativement à la grosseur des agrégats. La méthode de Jarvis-Partick donne le même genre d'erreur dans les cas simples, mais fonctionne mieux quand il y a beaucoup d'agrégats.

La complexité de la méthode de Jarvis-Patrick se situe entre $(N$ et $(N^3$, dépendant de l'implémentation. La complexité de la méthode de l'arbre des distances est $(N \log\ (N))$. Les valeurs du tableau 1 ont été calculées à partir d'un Pentium de 150 MHz qui utilise Linux comme système d'exploitation.

Points	Jarvis-Patrick	Arbre des distances
64	0.27	0.04
128	1.74	0.08
256	15.95	0.14
512	134.83	0.25

Tableau 1: Temps de l'UCT en seconde pour chaque méthode.

L'arbre des distances est certainement plus rapide que la méthode de Jarvis-Patrick, mais son avantage principal provient du fait que de nouveaux points peuvent être ajoutés à un arbre existant. L'ajout de 64 nouveaux points à un arbre qui en contient déjà 512 ne prend que 0.06 secondes. Pour ce genre d'opération, la méthode de Jarvis-Patrick prend environ 150 secondes car il faut refaire tous les calculs.

4 Résultats du système d'étiquetage

Dans les résultats qui suivent nous avons pris soin de séparer les données d'entraînement et les données test. Les résultats ne sont donc que pour les données test. Les tests ont été faits à petite échelle pour montrer les détails du fonctionnement de l'étiquetage, et à grande échelle pour calculer la précision de l'étiquetage.

4.1 Petite échelle

Les résultats qui suivent furent calculés à partir de 4 images dont chacune est subdivisée en 64 imagettes. Nous avons divisé les résultats en différentes catégories pour analyser la contribution des différents éléments du système. Il est à noter que certaines imagettes sont difficiles à étiqueter.

Par exemple, certaines sont mi-ciel, mi-arbre ou mi-eau, mi-plage. Néanmoins, les résultats semblent prometteurs.

Succès par modèle

Nous voulons analyser ici la contribution de chaque modèle et le résultat de la combinaison des modèles. Le tableau 2 nous montre, pour chaque modèle, le nombre d'éléments du vecteur de caractéristiques, le pourcentage d'imagettes non-étiquetées et le pourcentage d'erreurs.

Modèle	grandeur	non-étiq.	erreurs
Hasard	1	89.0	41.8
Uniformité	1	72.4	24.0
Couleur moyenne	6	70.4	2.7
Histogramme de teinte	360	38.2	2.6
Histogramme couleur	768	32.0	0.0
Tout les modèles	-	21.8	5.1

Tableau 2: Succès par modèle.

Le premier modèle, appelé *hasard*, ne calcule pas de caractéristiques, mais génère des points au hasard dans l'espace vectoriel du modèle. Le modèle *hasard* nous montre ce qui arrive si le modèle n'est pas bon. Le système ne peut créer de regroupement et la plupart des points demeurent sans étiquettes. Nous pouvons voir que le modèle *uniformité* n'est pas très bon. Ce modèle n'est composé que d'une des caractéristiques de la matrice de co-occurence. Ce modèle n'est pas assez riche. La couleur moyenne, qui utilise 6 nombres, est un peu mieux. Il semble que plus le vecteur est grand, plus les résultats sont bons.

Quand tous les modèles sont combinés nous pouvons voir que le pourcentage d'erreurs a quelque peu augmenté, mais que le pourcentage des imagettes non-étiquettées a diminué de beaucoup. C'est un bon résultat si nous considérons qu'un léger pourcentage d'erreurs est acceptable. De toute façon, ces résultats ne sont que des indications et il faut étudier les tests à grande échelle pour connaitre la précision du système.

Succès par étiquette

Le tableau précédent, en donnant des résultats globaux, laisse de coté un point important. Le but du système est de sélectionner les meilleurs agrégats de chaque modèle, et ce *pour chaque étiquette*. Le tableau 3 nous donne, pour chaque modèle, le pourcentage de succès pour chaque étiquette.

Chaque nombre représente le succès du modèle vis-à-vis une étiquette spécifique. Par exemple, l'histogramme couleur reconnait correctement 87.5% des imagettes ciel. Nous pouvons donc voir ici les forces et les faiblesses de chaque modèle. Nul modèle n'a reconnu la forêt car ses caractéristiques ressemblaient trop à celles des plantes. Pour augmenter la performance du système, il faudrait donc créer

Étiq.	mod	mod	mod	mod
ciel	87.5	31.2	71.9	40.6
montagne	68.8	12.5	31.2	18.8
gazon long	66.7	33.3	50.0	25.0
arbres	42.9	28.6	0.0	0.0
eau	35.3	29.4	82.4	23.5
foret	0.0	0.0	0.0	0.0
terre	88.2	29.4	94.1	5.9
gazon	93.8	37.5	81.2	50.0
plantes	0.0	50.0	0.0	0.0

Tableau 3: Succès par étiquette.

un nouveau modèle qui prendrait en considération les différences entre la forêt et les feuilles.

4.2 Grande échelle

L'article du MIT que nous utilisons comme comparaison [7] cite un taux de réussite de 90% dans un test d'une centaine d'images. Cependant, 80% des imagettes dans ce test ne furent pas étiquetées. Nous avons récupéré les images utilisées dans ces tests[1] pour y comparer notre propre système. Ce sont des photos *de voyage* qui sont plus ou moins nettes et qui sont très variées. Ces photos contiennent des scènes d'intérieur et d'extérieur, de ville et de campagne, de personnes et de foules, etc.

Nous avons fait nos tests en deux étapes. Dans la première étape qui est la phase d'apprentissage, nous avons visionné chaque image pour y ajouter quelques étiquettes. Nous avons ainsi ajouté environ 250 étiquettes à la base de données. Dans la deuxième étape nous avons fait un traitement par lots pour étiqueter toutes les images puis nous les avons visionnés de nouveau pour noter les erreurs sur papier. Il est certain qu'en mode entièrement interactif, en «jouant» avec l'ajout des étiquettes, nous aurions obtenu de meilleurs résultats.

Dans le tableau 4 nous avons reporté à la première ligne les résultats du test à petite échelle. La deuxième ligne nous donnent les résultats du test avec les images du MIT. Enfin, la troisième ligne affiche les résultats publiés par l'équipe du MIT. La deuxième colonne du tableau donne le pourcentage des imagettes qui ne furent pas étiquetées. La troisième colonne indique le pourcentage d'erreurs pour les imagettes étiquetées par le système.

À première vue nos résultats semblent comparables à ceux de l'équipe du MIT. Nous avons étiqueté un peu plus d'images, mais nous avons un peu plus d'erreurs. Il est cependant difficile de comparer ces deux résultats car trop de variables restent incontrolées. Nous avons certainement les mêmes images, mais nous n'avons pas le même étique-

[1] Nous remercions l'équipe du MIT pour nous avoir donné accès à ces images.

Modèle	non-étiquetées	erreurs
Petite échelle	21.8	5.1
Grande échelle	74.6	12.5
Résultat du MIT	80	10

Tableau 4: Test à grande échelle.

tage. De plus, comme l'étiquetage est en partie subjectif, les résultats ne seront pas les mêmes pour deux individus.

Deux points importants soulignent l'aspect subjectif des résultats: le grand nombre d'imagettes qui restent non-étiquetées (75% à 80%) et la distribution des erreurs d'étiquetages. Le tableau 5 qui nous montre le pourcentage d'images par rapport aux erreurs. La deuxième colonne indique le pourcentage des images qui ont la quantité d'erreurs indiquée à la colonne un. Nous voyons que la majorité des images ont zéro ou une erreur. Donc, quand les erreurs apparaissent, elles sont assez nombreuses pour être visibles à l'utilisateur qui peut faire les corrections qui s'imposent en ajoutant une ou deux étiquettes.

Nombre d'erreurs	% des images
0	38
1	30
2-5	20
5	12

Tableau 5: Distribution des erreurs.

Conclusion

Ce projet avait pour but la création d'un système pour l'étiquetage automatique des images. Autant que possible, le système devait paraître professionnel, être rapide et donner de bons résultats.

Pour paraître professionnel, le système devait se conformer aux normes établies dans le domaine informatique. L'utilisation d'une interface graphique est maintenant devenue un prérequis que nous avons satisfait en utilisant Amulet, une interface graphique développée par l'université Carnegie Mellon. Notre système est de conception modulaire, ce qui en facilite la programmation et l'entretien. Enfin, en plus du mode interactif, le système peut fonctionner en traitement par lots.

L'objectif de vitesse fut atteint grâce à un nouvel algorithme d'agrégation qui fonctionne de façon incrémentale. Ce nouvel algorithme, que nous avons nommé arbre des distances, est une contribution originale de ce travail. Grâce à cet algorithme, ajouter une nouvelle image au système d'étiquetage est un processus qui prend moins de 10 secondes. Avec l'algorithme de Jarvis-Patrick, le même travail prendrait près d'une demi-heure.

Avec l'utilisation de l'excellente idée d'une équipe du MIT, à savoir la combinaison de modèles multiples, nous nous sommes assurés de résultats qui sont supérieurs à l'utilisation de modèles individuels. Le système d'étiquetage ne fonctionne pas sans erreur, mais ce système n'est qu'une étape dans un processus de classification de l'image. Un système de classification pourrait rejeter l'étiquette qui n'apparaît qu'une fois dans une image ou qui semble suspecte dans le contexte des autres étiquettes de l'image.

Afin d'améliorer l'exactitude du système, plusieurs nouveaux projets pourraient être entrepris. Une segmentation de l'image au lieu d'un découpage en quadrillé augmenterait de beaucoup l'exactitude du système. Le système fonctionnerait mieux avec une série de modèles qui se complémentent l'un l'autre. L'algorithme d'agrégation qui fonctionne maintenant de façon binaire devrait être refait en arbre n-aire. Finalement, l'implémentation d'un système de classification qui utilise les résultats de l'étiquetage ferait un bon *démonstrateur* pour le système d'étiquetage.

Références

[1] Gilles Daigle, Shengrui Wang, Djemel Ziou. *Recherche d'images basée sur le contenu.* Rapport technique No. 212, Université de Sherbrooke, 1998.

[2] Gilles Daigle, Shengrui Wang, Djemel Ziou, Béchir El Ayeb. *A System for Picture Labeling.* IEEE International Conference on Systems, Man, and Cybernetics, 1998, pp. 4453-4458.

[3] Robert M. Haralick and Linda G. Shapiro. *Computer and Robot Vision.* Addison-Wesley, 1992.

[4] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data.* Prentice Hall Inc, Englewood Cliffs, NJ, 1988.

[5] R. A. Jarvis, Edward A. Patrick. *Clustering Using a Similar Measure Based on Shared Near Neighbors. IEEE Trans. Comput.*, 1973, C-22, 1025-1034.

[6] Dinesh Nair, Amar Mitiche, J. K. Aggarwal. *On comparing the performance of object recognition systems.* IEEE International Conference on Image Processing, Vol 2, 1995.

[7] Rosalind W. Picard and T. P. Minka. *Vision Texture for Annotation.* MIT Media Laboratory, Technical Report No. 302, 1995.

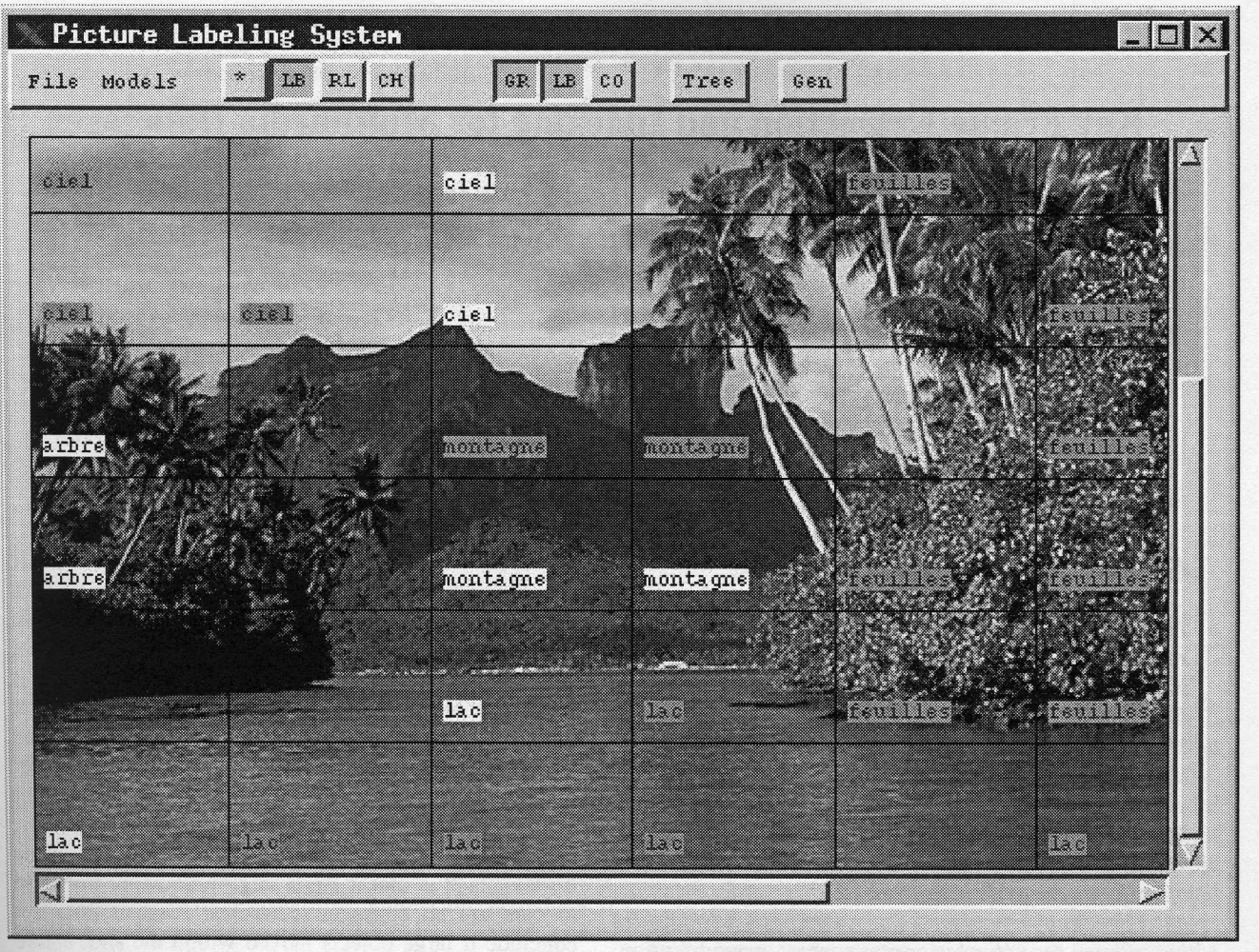

Figure 6: Programme d'étiquetage.