République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mouloud Mammeri de Tizi-Ouzou

Faculté de Génie Électrique Et d'Informatique
Département d'Automatique

THÈSE

pour l'obtention du diplôme de

Doctorat 3ème Cycle En Automatique

présentée et soutenue publiquement le 20 novembre 2023 par

Abderezak SALMI

Segmentation d'images couleur texturées basée sur la sélection d'attributs et la classification spectrale semi-supervisées sous contraintes

Jury

M. Salah HADDAB	Professeur	Université de Tizi-Ouzou	Président
M. Kamal HAMMOUCHE	Professeur	Université de Tizi-Ouzou	Rapporteur
M. Ludovic MACAIRE	Professeur	Université de Lille, France	Co-Rapporteur
M. Khaled HARRAR	Professeur	Université de Boumerdès	Examinateur
M. Mourad LAHDIR	Professeur	Université de Tizi-Ouzou	Examinateur
M. Idir FILALI	MCA	Université de Tizi-Ouzou	Examinateur

Remerciements

Les travaux de recherche présentés dans cette thèse ont été effectués au sein du laboratoire Vision Artificielle et Automatique des Systèmes (LVAAS) de l'Université Mouloud Mammeri de Tizi-Ouzou, Algérie.

Je tiens tout d'abord à remercier mon directeur de thèse Monsieur Kamal HAMMOUCHE, Professeur à l'Université Mouloud Mammeri de Tizi-Ouzou, de m'avoir accueilli au sein de son équipe de recherche. Je le remercie pour sa disponibilité et la confiance dont il a fait preuve, et ce, dès le Master-II. Il m'a toujours accordé du temps et prodigué de précieux conseils afin de m'aider à avancer surtout dans les moments difficiles.

Je tiens également à exprimer ma gratitude à mon co-directeur de thèse Monsieur Ludovic MACAIRE, Professeur à l'Université Lille 1 France, Je le remercie particulièrement pour ses précieuses remarques et conseils constructifs, qui m'ont permis d'améliorer et de faire aboutir ce travail.

Je remercie vivement Monsieur Salah HADDAB, Professeur à l'Université Mouloud Mammeri de Tizi-Ouzou, pour l'honneur qu'il me fait en acceptant de présider le jury de cette thèse.

J'adresse mes plus vifs remerciements à Monsieur Khaled HARRAR, Professeur à l'Université de M'hamed Bougara de Boumerdès, pour avoir accepté d'examiner ce travail. C'est avec un énorme plaisir que je le vois participer au jury.

Mes remerciements vont également à Monsieur Mourad LAHDIR, Professeur à l'Université Mouloud Mammeri de Tizi-Ouzou, pour l'intérêt qu'il manifeste à l'égard de ce travail et pour l'honneur qu'il me fait en acceptant de faire partie du jury.

Que Monsieur Idir FILALI, Maître de conférences à l'Université Mouloud Mammeri de Tizi-Ouzou, veuille trouver ici ma sincère reconnaissance pour avoir accepté d'examiner ce travail et faire partie du jury.

J'adresse mes remerciements les plus sincères à tous les membres du laboratoire LVAAS et spécialement: Zohra, Farid, Kahina, Damia, Merzouk, Dahmane pour leurs encouragements, leurs conseils avisés, et surtout pour tous les bons moments passés au sein du laboratoire.

J'adresse mes plus affectueux remerciements à mes parents, mes sœurs, et mes nièces pour leur tendresse indéfectible et pour m'avoir encouragé pendant toutes ces années.

Abderezak SALMI

Table des matières

Liste des figures

Liste des tableaux

Liste des algorithmes

Notations

I	Image numérique
N	Nombre total de données ou de pixels dans une image
R_i	Région d'indice r
k	Nombre de classes dans un ensemble de données ou de régions dans une image
d	Nombre total d'attributs
f_r	Attribut d'indice r
$\boldsymbol{f}_r$	Vecteur correspondant à l'attribut f_r
F_d	Ensemble total des attributs
x_i	Donnée d'indice i
x_{ir}	Valeur de l'attribut f_r sur la donnée x_i
$\boldsymbol{x}_i$	Vecteur de données correspondant à x_i
$\boldsymbol{x}_i^{(m)}$	Vecteur de la donnée x_i qui est caractérisée par m attributs
$\bar{x}_{\omega^l}$	Centroïde de la classe ω^l
$\bar{\boldsymbol{x}}_{\omega^l}$	Vecteur correspondant au centroïde $\bar{x}_{\omega^l}$
X	Ensemble de données d'apprentissage ou de pixels échantillons
Y	Ensemble de données de test ou de pixels non échantillons
$\boldsymbol{X}$, $\boldsymbol{Y}$	Matrices des données correspondantes respectivement aux ensembles X et Y
$\bar{X}$	Ensemble des centroïdes de classes
n	Nombre de données d'apprentissages ou de pixels échantillons
$\omega(x_i)$	Classe de la donnée x_i
Ω	Ensemble des étiquettes de classes de l'ensemble X
$\boldsymbol{\Omega}$	Vecteur des étiquettes de classes correspondant à Ω
ω^l	Étiquette de la classe l
p	Nombre de prototypes par classe
X^P	Ensemble des données étiquetées ou de pixels prototypes
X^U	Ensemble des données non étiquetées
X^l	Ensemble des p prototypes associés à la classe l
M	Ensemble des contraintes must-link
C	Ensemble des contraintes cannot-link
M^{SS}	Ensemble des nouvelles contraintes must-link

G	Graphe de similarité
v_i	Nœud correspondant à la donnée x_i
e_{ij}	Arête reliant deux nœuds v_i et v_j
V	Ensemble des nœuds dans un graphe
E	Ensemble des arêtes dans un graphe
$\delta(\boldsymbol{x}_i,\boldsymbol{x}_j)$	Distance entre deux vecteurs de données x_i et x_j
w_{ij}	Similarité entre deux vecteurs de données x_i et x_j
d_i	Degré du nœud correspondant à la donnée x_i
$\boldsymbol{\Delta}$	Matrice des distances
$\boldsymbol{W}$	Matrice de similarité
$\boldsymbol{D}$	Matrice des degrés
$\boldsymbol{L}$	Matrice laplacienne non normalisée
$\boldsymbol{L}_{Sym}$	Matrice laplacienne normalisée symétrique
G^M	Graphe des must-link
G^C	Graphe des cannot-link
$\boldsymbol{W}^{M,C}$	Matrice de similarité contrainte
$\boldsymbol{L}_{Sym}^{M,C}$	Matrice laplacienne normalisée symétrique contrainte
λ_i	Valeur propre d'indice i
$\boldsymbol{u}_i$	Vecteur propre d'indice i
$\hat{\boldsymbol{W}}^*$	Matrice de similarité cible
$\tilde{\boldsymbol{W}}$	Noyau gaussien normalisé
$z(y_i)$	Donnée non échantillon projetée
$\tilde{z}(y_i)$	Donnée non échantillon projetée normalisée
$\hat{m}$	Nombre d'attributs sélectionnés
m^*	Nombre optimal d'attributs
$NP(x_i)$	Plus proche prototype de la donnée x_i
$\boldsymbol{\mathcal{M}}$	Matrice de co-occurrence
σ	Paramètre d'échelle de la fonction de similarité gaussienne
K	Paramètre du graphe des K plus proches voisins
s	Paramètre de Haralick
θ	Paramètre d'orientation de Gabor
γ	Paramètre d'échelle de Gabor
$R^{[c]}$	Somme des rangs

Abréviations

ALI	Advanced Land Imager
CCM	Chromatic Co-occurrence Matrix
CFS-SC	Constrained Feature Selection for Spectral Clustering
CHAC	Constrained Hierarchical Agglomerative Clustering
CNN	Convolutional Neural Network
CNML	Correct New Must-Link
CSC	Constrained Spectral Clustering
COF	Co-Occurrence Features
DA	Deep visual model
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DDC	Density Decreased Chain
DHC	Dynamic Hierarchical Classification
DLSRC	Dictionary Learning based Sparse Representation
DSSN	Deep Semantic Segmentation Network
EnsCLS	Ensemble Constrained Laplacian Score
EWT-FCNT	Empirical Wavelet Transform-Fully Convolutional Network for Texture
FCM	Fuzzy c-Means
FCN	Fully Convolutional Network
FCNT	Fully Convolutional Network for Texture
FSEG	Factorization-based texture SEGmentation
GLCM	Gray Level Co-occurrence Matrix
HAC	Hierarchical Agglomerative Clustering
HDC	Hierarchical Divisive Clustering
KNN	K Nearest Neighboor
LBP	Local Binary Pattern
LS	Laplacien Score
MinCut	Minimum Cut
MLLIF	Model-based Learning of Local Image Features
MLP	Multi-Layer Perceptron

mRMR	minimal-Redundancy-Maximal-Relevance
MRF	Markov Random Field
MS	Mean-Shift
MSCNN	Multi-Scale Convolutional Neural Networks
NCut	Normalized Cut
NML	New Must-Link
ORTSEG	Occlusion of Random Textures SEGmentation
PCA-MS	Variational Multi-Phase Segmentation
PDF	Probability Density Function
PSP-Net	Pyramid Scene Parsing Network
R-TFR	Recursive Texture Fragmentation and Reconstruction
RAG	Region Graph Adjacency
RSM	Random Subspace Method
RSSFCA	Robust Self-Sparse Fuzzy Clustering
SFS	Sequential Forward Selection
SC	Spectral Clustering
SCGS	Semi-supervised pairwise Constraint-Guided Sparse
SVM	Support Vector Machine
WSSCGP	Weakly-Supervised Sparse Coding Geometric Priori

Introduction Générale

1 Contexte

Les systèmes d'analyse automatique d'images investissent de plus en plus les différents secteurs d'activités grâce aux outils d'acquisition d'images numériques (appareil photo, caméra, scanner, téléphone portable, etc.), devenus omniprésents et à l'informatique ubiquitaire. L'analyse automatique d'images passe par plusieurs étapes, dont la plus fondamentale est la segmentation d'images. Son rôle est de délimiter dans l'image étudiée un ensemble de zones pertinentes pour l'interprétation ou la modélisation de la scène perçue. Il s'agit donc d'une étape clé, et sa qualité conditionne fortement la réussite globale de l'application envisagée.

Les domaines d'application de la segmentation d'images sont nombreux et variés. Dans le domaine industriel, par exemple, la segmentation permet de détecter les défauts dans les tissus, le bois, les pièces mécaniques, etc. Un autre domaine dans lequel la segmentation est très sollicitée est celui de l'imagerie médicale. Elle permet notamment de découper les images en différentes catégories de tissus, d'organes, de pathologies, ou d'autres structures biologiquement pertinentes, et ce dans le but de diagnostiquer des maladies comme le cancer, ou réaliser des mesures comme le comptage des cellules. La segmentation est également très impliquée dans l'analyse d'images satellitaires afin de détecter les routes, les bâtiments, les cours d'eau, ou les champs. Dans le domaine agricole, la segmentation est exploitée pour discerner des fleurs, des feuilles ainsi que des fruits, ou de localiser les mauvaises herbes. La segmentation est aussi de plus en plus intégrée aux caméras embarquées dans les voitures autonomes ou dans les véhicules sans pilote. Elle permet de détecter tous les objets qu'un véhicule pourrait rencontrer dans son champ de vision (panneaux, piétons, voitures, vélos, animaux, etc). D'autres applications de la segmentation concernent la vidéo-surveillance, la biométrie, l'indexation, et la recherche d'images sur le web ou dans les bases de données.

Concrètement, la segmentation est une procédure qui consiste à partitionner une image numérique en régions homogènes, composée d'un ensemble de pixels connexes ayant des caractéristiques communes en termes d'intensité, de couleur, de texture, de taille, de forme, etc. Chaque région est supposée représenter un objet ou une zone d'intérêt de la scène observée. La couleur et la texture sont les deux sources d'information les plus importantes qui sont liées à notre perception visuelle. Elles doivent être nécessairement prises en compte lors de la segmentation, notamment pour les images couleur texturées. En effet, dans certaines applications, comme par exemple le domaine médical ou celui de la télédétection, les régions qui composent les images à traiter sont non seulement en couleur, mais aussi texturées. Pour ce type d'images, la segmentation doit être capable de discerner des régions ayant chacune une texture couleur spécifique. Cette tâche, relativement simple pour l'être humain, reste difficile et complexe à réaliser par un système d'analyse automatique d'images.

La segmentation d'images couleur texturées constitue le thème majeur de cette thèse. Son processus se déroule principalement en deux étapes. La première est destinée à caractériser chaque pixel par un ensemble d'attributs de texture couleur, et la seconde à partitionner l'image en régions homogènes en tenant compte des attributs extraits. Dans la première étape, plusieurs techniques ont été proposées pour extraire des attributs de texture couleur. Elles diffèrent par la manière de les extraire et la façon de les combiner. L'étape de partitionnement peut être réalisée de différentes façons dépendant de la stratégie utilisée pour détecter et localiser les régions homogènes, ainsi que du contexte d'apprentissage (supervisé, non supervisé, et semi-supervisé) dans lequel opère la segmentation. Parmi la pléthore de méthodes de segmentation d'images couleur texturées, se trouvent les méthodes par apprentissage profond, par coupe de graphe, et celles basées sur la classification.

Dans notre travail, nous nous sommes intéressés aux méthodes de classification, car celles-ci peuvent être exploitées non seulement en segmentation d'images mais également en classification de tout types de données. L'objectif de la classification est de regrouper les données (pixels) en plusieurs classes selon un critère de similarité, c.-à-d. les données dans une même classe doivent être les plus similaires possibles, et les données entre deux classes différentes doivent être les moins similaires possibles. La classification peut être abordée selon les mêmes contextes d'apprentissage que la segmentation. Dans un contexte non supervisé, aucune information a priori sur l'appartenance des données aux classes n'est disponible. Dans ce cas, la classification vise à regrouper l'ensemble de données non étiquetées en classes. Dans un contexte supervisé, on dispose a priori d'un ensemble de données étiquetées, dites d'apprentissage ou prototypes, dont l'appartenance aux classes est connue. Ces données étiquetées sont ensuite utilisées pour prédire la classe d'appartenance d'une donnée non étiquetée lors

d'une phase de décision. Dans un contexte semi-supervisé, on dispose à la fois d'un ensemble de données non étiquetées et d'une petite quantité d'informations, généralement fournie par un expert, sous forme de données étiquetées indiquant la classe à laquelle elles appartiennent, ou sous forme de contraintes mentionnant si deux données appartiennent (must-link) ou non (cannot-link) à la même classe. Cette information a priori sert à guider la classification des données non étiquetées afin d'aboutir à des résultats plus performants.

Parmi les méthodes de classification existantes, nous nous sommes intéressés dans cette thèse à la classification spectrale, qui est une approche non supervisée basée sur l'exploitation du graphe de similarité représentant l'ensemble des données à classer. La classification spectrale permet de détecter des classes de formes complexes et non linéairement séparables. Et surtout, elle peut être adaptée au contexte semi-supervisé permettant une intégration facile des informations a priori telles que les contraintes.

Quel que soit le contexte d'apprentissage dans lequel on se place ou la technique de classification employée, chaque donnée est caractérisée par un ensemble d'attributs. Dans le cas particulier de la segmentation d'images couleur texturées, une donnée correspond à un pixel et les attributs désignent des attributs de texture couleur. Habituellement, l'analyste a tendance à employer un grand nombre d'attributs. Cette multiplicité d'information est censée conduire à une meilleure discrimination entre les différentes classes. Or, dans les faits, un grand nombre d'attributs peut impacter négativement sur le temps de calcul des méthodes de classification. De plus, certains attributs peuvent être redondants et non pertinents, pouvant même dégrader les performances de classification ou de segmentation.

Une des solutions pour surmonter ces problèmes, est de réduire le nombre d'attributs, en sélectionnant les plus pertinents. Dans cette optique, plusieurs méthodes de sélection d'attributs sont proposées dans la littérature. Celles ci se basent sur l'attribution d'un score de pertinence à chaque attribut (feature ranking) ou à plusieurs attributs à la fois (subset selection), indépendamment (filtres) ou non (enveloppes et hybrides) d'un algorithme de classification. De plus, la sélection peut opérer dans un contexte d'apprentissage non supervisé, supervisé ou semi-supervisé, au même titre que la classification ou la segmentation.

La sélection d'attributs constitue un autre thème majeur dans cette thèse. Les méthodes de type filtre sont rapides, simples, et indépendantes de l'algorithme de classification. Nous nous sommes intéressé dans cette thèse à ce type de méthodes, et plus particulièrement à celles qui exploitent les contraintes must-link et cannot-link. Ces dernières tirent profit de la représentation des données par des graphes de similarité, comme dans le cas de la classification spectrale, pour évaluer la pertinence de chaque attribut par un score de contraintes.

2 Objectif et contributions

L'objectif de cette thèse est double. Le premier concerne la sélection d'attributs basée sur un score de contraintes dans les deux contextes d'apprentissage supervisé et semi-supervisé, et le second la segmentation d'images couleur texturées semi-supervisée sous contraintes.

Les méthodes de sélection d'attributs basées sur les scores de contraintes évaluent la pertinence des attributs individuellement ignorant ainsi la corrélation entre eux, et ne sont pas en mesure de déterminer automatiquement le nombre d'attributs à sélectionner. Cela les rend moins efficaces. Pour palier à tous ces inconvénients, nous proposons dans cette thèse une méthode originale de sélection d'attributs de type filtre, basée sur un score de contraintes, capable d'évaluer la pertinence d'un sous-ensemble d'attributs à la fois dans les deux contextes d'apprentissage supervisé et semi-supervisé. Notre score permet également de tenir compte de la corrélation entre les attributs et de déterminer automatiquement le nombre optimal d'attributs.

Pour segmenter efficacement des images couleur texturées, la plupart des méthodes récentes se tournent vers des méthodes supervisées. Cependant, ces méthodes nécessitent des bases de données d'apprentissage ou des images de référence accompagnées de leurs images segmentées (vérité terrain). La construction de ces bases d'apprentissage par un expert humain peut être fastidieuse et coûteuse en temps de calcul. Afin de contourner cet inconvénient, nous proposons dans cette thèse une méthode de segmentation semi-supervisée d'images couleur texturées basée sur la sélection d'attributs et sur la classification spectrale semi-supervisées sous contraintes. Dans cette méthode, seuls quelques pixels prototypes, dont les relations sont représentées par des contraintes, sont fournies par l'utilisateur.

3 Organisation de la thèse

Nous avons scindé le reste de ce manuscrit en cinq chapitres, auxquels s'ajoute une conclusion générale.

Dans le premier chapitre, nous définissons la segmentation d'images, puis nous exposons quelques notions sur la couleur et sur la texture (niveaux de gris et couleur). Par la suite, nous proposons une catégorisation des différentes approches de segmentation d'images couleur texturées. À la fin de ce chapitre, nous exposons quelques méthodes classiques de classification des données, tout en détaillant un peu plus les méthodes non supervisées et semi-supervisées sous contraintes de type must-link et cannot-link.

Le deuxième chapitre est dédié à la classification spectrale semi-supervisée sous contraintes. Tout d'abord, nous montrons comment représenter les données par un graphe car celui-ci permet d'aborder la sélection d'attributs et la classification spectrale sous un même formalisme. Puis, nous décrivons les principales étapes de la classification spectrale et comparons ses performances à celles de plusieurs autres méthodes classiques de classification non supervisée sur des bases de données synthétiques et réelles. Ensuite, nous introduisons la notion de classification semi-supervisée avec contraintes et les différentes manières de les intégrer dans le processus de classification spectrale. En fin, nous validons la classification spectrale sous contraintes en comparant ses performances à celles des autres méthodes classiques de classification sous contraintes.

Dans le troisième chapitre, nous définissons dans un premier temps, la sélection d'attributs et présentons quelques notions qui lui sont liées. Ensuite, nous exposons les principales étapes du processus de sélection d'attributs, ainsi que les différentes approches proposées dans la littérature. Nous terminons ce chapitre par un état de l'art sur les méthodes de sélection d'attributs de type filtre, basées sur les scores de pertinence, dans les trois contextes d'apprentissage (non supervisé, supervisé, et semi-supervisé).

Le quatrième chapitre est consacré à la méthode de sélection d'attributs développée. Nous exposons d'une manière détaillée le score de contraintes proposé pour évaluer la pertinence d'un sous-ensemble d'attributs, ainsi que la procédure de sélection employée pour sélectionner un nombre optimal d'attributs. Les performances de cette méthode sont évaluées et comparées à celles des autres méthodes de sélection d'attributs basées sur les scores de contraintes à travers plusieurs bases de données.

Le cinquième chapitre est réservé à la présentation et l'évaluation de la méthode de segmentation d'images couleur texturées développée. Nous présentons le schéma général de l'approche proposée, puis nous détaillons les différentes étapes qui la composent. Les résultats de segmentation obtenus sur différentes bases d'images (textures couleur, textures en niveaux de gris, satellitaires, médicales, et naturelles) sont présentés, discutés, et confrontés aux résultats fournis par des méthodes de segmentation de l'état de l'art, dans les trois contextes d'apprentissage (supervisé, non supervisé, et semi-supervisé).

Nous terminons ce manuscrit par une conclusion générale, dans laquelle nous dressons un bilan général des contributions apportées dans le cadre de cette thèse, puis nous exposons quelques perspectives possibles pour améliorer les performances des méthodes développées.

Chapitre 1

Segmentation d'images couleur texturées et classification des données

1.1 Introduction

La segmentation d'images est une procédure importante et fondamentale qui intervient dans l'interprétation des images dans beaucoup d'applications de vision par ordinateur. Elle consiste à partitionner une image numérique en plusieurs régions homogènes et connexes. Plusieurs approches de segmentation d'images ont été proposées dans la littérature. Le choix d'une d'entre elles dépend de plusieurs facteurs liés à l'application (type d'image à segmenter, informations recherchées, l'information a priori disponible, etc). La segmentation d'images se base sur les informations d'intensité, de couleur, et/ou de texture, qui caractérisent la plupart des surfaces naturelles. Le processus de segmentation des images couleur texturées comprend généralement deux étapes principales: la caractérisation des pixels de l'image à l'aide d'un ensemble d'attributs de texture couleur, et le partitionnement de l'image en régions homogènes en se basant sur les attributs de texture couleur extraits.

Ce chapitre, consacré à la segmentation d'images couleur texturées et à la classification des données, est organisé de la manière suivante. Tout d'abord, nous définissons de manière générale la segmentation d'images couleur texturées, puis nous exposons quelques notions liées à la couleur et à la texture. Par la suite, nous proposons une catégorisation des différentes approches de segmentation d'images couleur texturées. Nous terminons ce chapitre par un état de l'art sur les méthodes classiques de classification des données dans les trois contextes d'apprentissage (supervisé, non supervisé, et semi-supervisé).

1.2 Segmentation d'images couleur texturées

Segmenter une image revient à trouver ses différentes régions homogènes ou les contours de ces régions. Ces régions correspondent aux objets contenus dans l'image et les contours à leurs frontières [1]. Le résultat de la segmentation se présente soit sous la forme d'une image binaire (1: contour/objet et 0: non contour/fond), soit d'une image étiquetée où chaque étiquette correspond à une région. Formellement, la segmentation est un traitement de bas niveau qui consiste à partitionner une image I en k régions $\{R_1,R_2,\ldots,R_k\}$ selon un prédicat d'uniformité noté $P(R_i)$, associé à un critère d'homogénéité. Chaque région R_i, $i = 1,\ldots,k$, est composée d'un ensemble connexe de pixels ayant des propriétés communes en termes d'intensité, de texture, de couleur, etc, qui les différencient des pixels des régions voisines. Cette définition peut être formulée par les relations suivantes [2]:

- $I = \bigcup_i R_i$,
- $\forall i,\ R_i \neq \varnothing$,
- $\forall i$, R_i est une composante connexe,
- $\forall R_i,\ P(R_i) =$ Vrai,
- $\forall\ i,j\ ;\ i \neq j \Rightarrow\ R_i \cap R_j = \varnothing$,
- $\forall\ i,j,\ P(R_i \cup R_j) =$ Faux.

La segmentation d'images a suscité un grand intérêt dans la communauté scientifique, et un grand nombre de méthodes ont été proposées [3, 4]. Aucune de ces méthodes ne peut se targuer d'être universelle. Le choix de l'une d'entre elles dépend de plusieurs facteurs tels que [2]: *la nature de l'image à segmenter* (texturée, couleur, bruitée), *l'interaction avec l'utilisateur* (supervisée, non supervisée, et semi-supervisée), *les primitives à extraire de l'image* (contours, régions, formes, textures), *le problème à traiter en aval de la segmentation* (reconnaissance des formes, interprétation, suivi d'objets, localisation, diagnostic), et *les contraintes d'exploitation* (complexité algorithmique et temps réel).

La segmentation d'images couleur texturées est une tâche très sollicitée dans diverses applications de vision par ordinateur et dans lesquelles les régions qui composent les images à traiter sont non seulement en couleur, mais aussi texturées. C'est le cas par exemple des images issues du domaine médical [5–7] et de la télé-détection [8, 9]. Pour ce type d'images, la segmentation vise à détecter les régions ayant chacune une texture couleur spécifique. Cette tâche demeure encore difficile car dans une région texturée, les intensités ou les couleurs ne sont pas forcément les mêmes et la frontière entre deux textures couleur ou en niveaux de gris n'est pas toujours facile à discerner.

Le processus de segmentation d'images couleur texturées suit principalement deux étapes. Dans la première étape, chaque pixel de l'image à segmenter est caractérisé par un ensemble d'attributs de texture couleur. Dans la seconde étape, l'image est partitionnée en plusieurs régions en se basant sur les attributs de texture couleur de chaque pixel.

1.3 Caractérisation de la texture couleur

La couleur et la texture sont les deux sources d'information les plus importantes sur lesquelles s'appuie le processus de segmentation d'images couleur texturées [3].

1.3.1 Couleur

La couleur est la perception visuelle de la lumière que renvoie un objet. Elle est utilisée pour désigner un objet en permettant à l'œil de le distinguer des autres objets [10].

Une image couleur numérique est une matrice composée de n pixels dont chaque pixel est représenté par trois composantes de couleur: rouge (R), verte (G), et bleue (B). L'attachement au système de couleur RGB s'explique principalement par la dépendance aux matériels (cartes d'acquisition, cartes vidéo, caméras, écrans, etc.) qui effectuent leurs échanges d'informations en utilisant le triplet (R,G,B). D'autres systèmes de représentation de la couleur ont été conçus en se basant sur le respect de propriétés physiques, physiologiques, et psychologiques de notre perception des couleurs. D'une manière générale, la couleur d'un pixel peut être représentée par trois composantes notées $\mathcal{C}_1$, $\mathcal{C}_2$, et $\mathcal{C}_3$. L'ensemble des n pixels peuvent être représentés comme des points dans un espace tridimensionnel $(\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3)$, appelé *espace de couleur*. Plusieurs espaces de couleur ont été proposés dans la littérature et qui peuvent être regroupés en quatre familles [11, 12]:

Espaces de primaires: Ils supposent que chaque couleur peut être reproduite par le mélange (additif ou soustractif) de trois couleurs primaires. Les espaces de couleur primaires les plus connus sont: RGB et XYZ.

Espaces luminance-chrominance: Ils considèrent la couleur comme composée de trois composantes, l'une liée à la luminance et les deux autres à la chrominance. La luminance est l'attribut d'une sensation visuelle selon laquelle une surface paraît émettre plus ou moins de lumière. Tandis que la chrominance fait référence à l'information de couleur. Parmi les espaces de cette famille, on peut citer: $\mathrm{L}^*a^*b^*$, $\mathrm{L}^*u^*v^*$,$\mathrm{YC}_b\mathrm{C}_r$, et $\mathrm{AC}_1\mathrm{C}_2$ [12, 13].

Espaces perceptuels: Ils sont basés sur le ressenti de la perception humaine. En effet, l'être humain ne perçoit pas la couleur comme une combinaison de couleurs primaires mais plutôt comme des entités liées à la teinte, la saturation, et la luminance. La teinte correspond aux dénominations des couleurs telles que rouge, vert, jaune, etc. Le blanc, le noir, ou les niveaux de gris sont des couleurs qui n'ont pas de teinte. La saturation est une grandeur permettant d'estimer le niveau de coloration d'une teinte indépendamment de la luminosité. Elle représente la pureté de la couleur perçue comme vive, pâle, terne, etc. Parmi les espaces de cette famille, on peut citer: HSV, HSI, et TLS [12].

Espaces d'axes indépendants: La plupart des composantes de couleur des espaces cités précédemment sont plus ou moins corrélées. Les espaces d'axes indépendants cherchent à définir la couleur par trois composantes de couleur indépendantes les unes des autres. Les espaces de couleur X_1,X_2,X_3 et I_1,I_2,I_3 sont les plus connus de cette famille [14].

Il convient de noter qu'il est possible de passer d'un espace de couleur vers un autre par l'intermédiaire de relations mathématiques linéaires ou non linéaires, et que chaque espace de couleur possède ses propres avantages et limites en fonction du domaine d'application. Dans le cadre de la segmentation d'images couleur, les espaces RGB, XYZ, HSV, et $L^*a^*b^*$ décrits dans l'annexe A sont souvent utilisés [3, 10]. Dans ce cas, les attributs de couleur de chaque pixel sont représentés simplement par ses valeurs de composantes de couleur ou déterminés localement en tenant compte de ses pixels voisins situés dans une fenêtre de voisinage centrée sur ce pixel. Parmi ces attributs, on peut citer: les histogrammes de couleur, les couleurs dominantes, les couleurs moyennes, et les moments statistiques [3].

1.3.2 Texture

Au même titre que la couleur, la texture est un facteur fondamental dans la perception de l'environnement et l'identification de ses objets. Elle est liée à l'apparence d'une surface ou d'une région, et se décrit par des termes linguistiques qui nous sont familiers comme fine ou grossière, bosselée ou striée, régulière ou irrégulière, ridée ou granuleuse, isotrope ou anisotrope. C'est une notion intuitivement évidente pour l'homme car elle est proche de ses concepts perceptifs, mais contrairement à la couleur, la texture reste difficile à définir de manière précise. Cette difficulté se traduit dans la littérature par un grand nombre de définitions [15, 16]. La plus couramment citée est celle qui définit la texture comme *la répétition spatiale d'un motif élémentaire de base dans plusieurs directions* [15], sachant qu'un motif élémentaire de base, appelé aussi *primitive* ou *texon*, est composé d'un ensemble connexe de pixels ayant des propriétés similaires entre eux. Une autre définition assez

répondue considère qu'une région de l'image contient une texture constante si *un ensemble de statistiques locales ou autres propriétés de la fonction image sont constantes, faiblement, ou approximativement périodiques* [17]. À partir de ces définitions, il est possible de dégager deux propriétés dans la manière de percevoir la texture. La première est liée à l'échelle d'observation et la seconde concerne la notion de voisinage. Ainsi, on peut distinguer trois grandes familles de textures (Figure 1.1):

- *Textures structurées* (*Macro-textures*): Ces textures sont caractérisées par la répétition spatiale d'un motif élémentaire de base dans différentes directions.
- *Textures aléatoires* (*Micro-textures*): Dans ces textures, on ne distingue aucun motif élémentaire de base apparent. Elles ont par contre un aspect anarchique tout en restant globalement homogènes.
- *Textures directionnelles*: Ces textures ne sont pas totalement aléatoires et ne présentent pas de motif élémentaire de base bien apparent. Néanmoins, elles se caractérisent par une certaine orientation.

Il existe d'autres manières de classer les textures comme, par exemple, les textures *naturelles* et *artificielles*, ou les textures *homogènes*, *faiblement homogènes*, et *non homogènes* (Figure 1.1). Les textures naturelles sont rarement homogènes, et il se peut même qu'une texture unique soit composée d'un mélange d'une même texture de résolutions différentes ou de textures différentes. Ces textures sont généralement qualifiées de *complexes*.

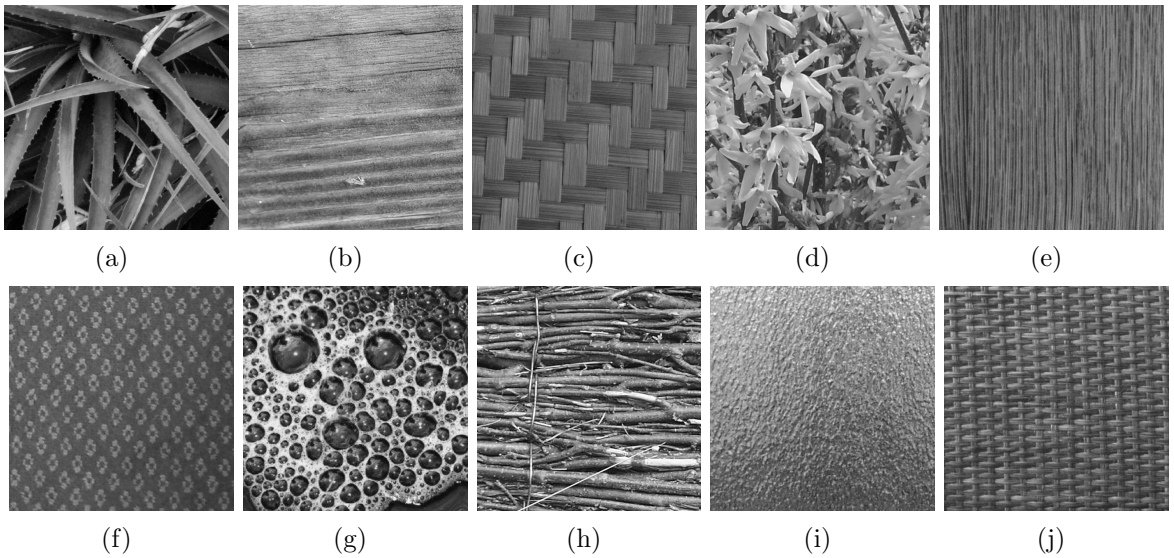


Figure 1.1: Exemples de textures niveaux de gris. [(c),(f),(j)] structurées, [(a),(d),(g),(i)] aléatoires, [(b),(e),(h)] directionnelles, [(a),(b),(d),(e),(h)] naturelles, [(c),(f),(g),(i),(j)] artificielles, [(c),(f),(i),(j)] homogènes, [(b),(e),(h)] faiblement homogènes, [(a),(d),(g)] non homogènes, et [(g)] complexes.

La grande diversité des textures ainsi que la complexité de donner une définition précise de la texture a engendré une multitude de méthodes d'analyse de la texture [18, 19]. Les plus connues sont: les matrices de co-occurrences des niveaux de gris (GLCM) [20], la transformée de Fourier [21], les filtres de Gabor [22], la transformation en ondelettes [23], le modèle de Markov [24], les modèles auto-régressifs [25], les fractales [26], les masques de Laws [27], et les motifs locaux binaires (LBP) [28]. La plupart de ces méthodes ont été utilisées pour segmenter des images texturées en niveaux de gris [22–26, 29].

1.3.3 Texture couleur

Initialement la texture a été définie pour des images en niveaux de gris. Cependant, ces définitions, comme celles présentées dans la section 1.3.2, restent valables pour les textures couleur. Il en est de même pour les différents types de textures (Figure 1.2).

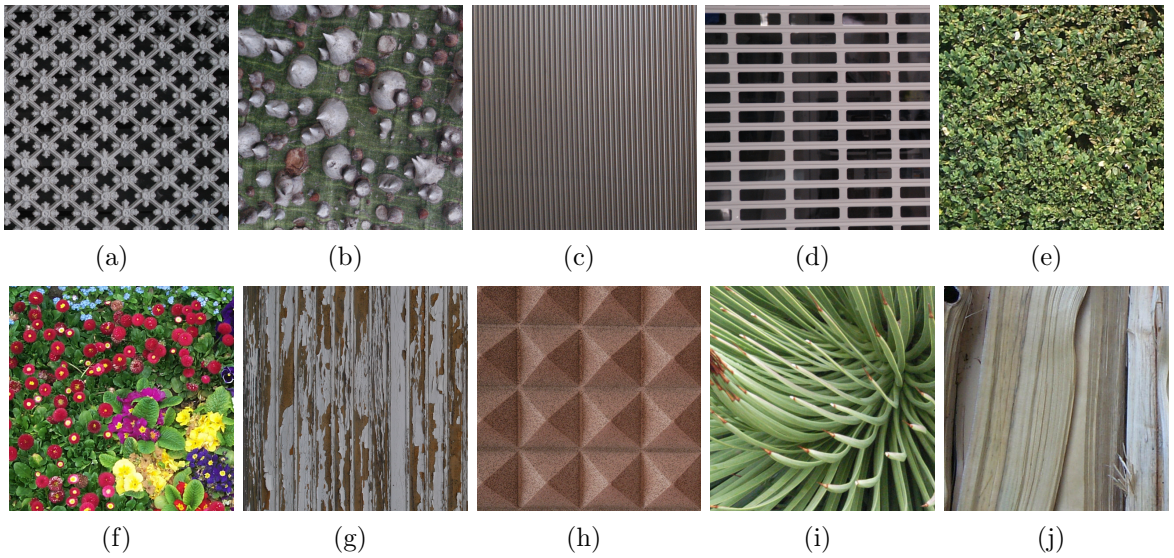


Figure 1.2: Exemples de textures couleur. [(a),(d),(h)] structurées, [(b),(e),(f),(i)] aléatoires, [(c),(g),(j)] directionnelles, [(b),(e),(f),(i),(j)] naturelles, [(a),(c),(d),(g),(h)] artificielles, [(a),(c),(d),(h)] homogènes, [(e),(g),(j)] faiblement homogènes, [(b),(f),(i)] non homogènes, et [(b),(f)] complexes.

Une texture couleur peut être définie comme un ensemble de propriétés locales de voisinage des couleurs d'une région de l'image. Drimbarean et Whelan définissent la texture couleur comme *un motif décrit par la relation entre sa distribution chromatique et spatiale* [30]. Cependant, il est important de différencier les textures en niveaux de gris des textures couleur. En effet, comme l'ont mentionné Drimbarean et Whelan dans [30], deux régions constituées de la même couleur mais de motifs de texture différents (en niveaux de gris), ou du même motif de texture (en niveaux de gris) mais de couleurs différentes sont considérées

comme deux régions de texture couleur différentes. Il est par conséquent plus judicieux de combiner les informations de couleur et de texture afin de segmenter des images couleur texturées. En effet, la couleur va permettre de différencier deux textures identiques, alors que la texture va pouvoir discerner des régions de même couleur mais de structures différentes.

La plupart des méthodes d'analyse de la texture conçues pour les images en niveaux de gris ont été étendues aux images couleur. Elles ont pour but d'extraire un ensemble d'attributs qui caractérisent les textures couleur présentes dans une image couleur. Parmi ces méthodes, on peut citer: les matrices de co-occurrences chromatiques (CCM) [31, 32], les motifs locaux binaires chromatiques [33, 34], les matrices de longueurs de plages [35, 36], la transformée de Fourier chromatique [37], les filtres de Gabor [38–40], la transformée en ondelettes [41, 42], les modèles de Markov [43, 44], et les modèles fractales chromatiques [45, 46]. Certaines de ces techniques ont été exploitées dans la segmentation pour caractériser chaque pixel par des attributs de texture couleur [6, 43, 47, 48].

Nous décrirons dans l'annexe A, deux méthodes d'extraction d'attributs de texture couleur, à savoir, les matrices de co-occurrences chromatiques et les filtres de Gabor. Celles-ci sont fréquemment utilisées dans la segmentation d'images couleur texturées [49]. Elles sont par ailleurs exploitées dans la méthode de segmentation développée au chapitre 5 de cette thèse.

1.4 Approches de segmentation d'images couleur texturées

De nombreux travaux menés dans le cadre de la segmentation d'images couleur texturées ont démontré que l'utilisation conjointe de la couleur et de la texture permet de segmenter des images plus efficacement qu'en utilisant seulement la couleur ou la texture [38, 50, 51]. Celles-ci peuvent être catégorisées de trois façons: selon la manière de caractériser la texture couleur, selon la stratégie utilisée pour identifier et localiser les différentes régions de l'image, et selon le contexte d'apprentissage.

1.4.1 Catégorisation selon la caractérisation des textures couleur

Ilea et Whelan [50] ont regroupé les méthodes de segmentation d'images couleur texturées en trois approches, selon la manière avec laquelle les informations de couleur et de texture sont combinées:

1.4.1.1 Approche séquentielle

Cette approche s'appuie sur l'hypothèse qu'il n'y a pas de règles explicites ou de modèles analytiques décrivant entièrement la dépendance de la couleur et de la texture lors du processus de formation de l'image. Les attributs de texture couleur sont alors extraits en deux étapes successives (Figure 1.3). La première étape consiste à segmenter l'image en tenant compte uniquement des attributs de couleur. L'image segmentée est représentée par des indices scalaires correspondant aux numéros des régions obtenues. Dans la seconde étape, des attributs de texture sont extraits pour chaque pixel de l'image indice par l'une des techniques conçues pour les images en niveaux de gris.

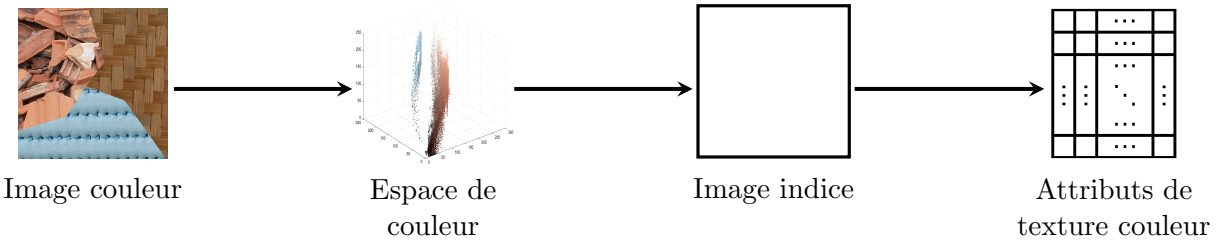


Figure 1.3: Combinaison séquentielle de la couleur et de la texture.

Cette démarche a été suivie par Deng et Manjunath [52]. La méthode de segmentation d'images proposée, nommée JSEG, comprend deux étapes. Dans la première étape, la couleur est quantifiée en un nombre représentatif de classes. Chaque pixel de l'image est alors attribué à une classe permettant ainsi d'obtenir une image indice. Une "J-image" correspondante à des mesures d'homogénéité de l'image indice à différentes échelles est alors générée. Dans la deuxième étape, l'image indice est segmentée en utilisant la méthode de croissance de régions à partir des germes définis sur la J-image. La méthode de segmentation d'images proposée par Mignotte [53] comprend également deux étapes. Dans la première étape, l'algorithme k-means [54] est appliqué sur l'image dans six espaces de couleur (RGB, HSV, YIQ, XYZ, $L^*a^*b^*$, et $L^*u^*v^*$) permettant ainsi d'obtenir plusieurs résultats de segmentation (images indices). Dans la deuxième étape, l'algorithme k-means est appliqué une autre fois en prenant comme attributs les histogrammes locaux calculés pour chaque pixel de l'image à partir des étiquettes des différentes segmentations obtenues lors de la première étape. La méthode de segmentation d'images proposée par Gaetano et al. [9], effectue dans un premier temps une sur-segmentation en utilisant d'abord les attributs de couleur RGB, puis les informations spatiales. Les régions obtenues sont ensuite fusionnées pour former les régions texturées en tenant compte d'un attribut de texture mesurant leurs interactions qui sont modélisées par des chaines de Markov.

1.4.1.2 Approche parallèle

Cette approche suppose aussi que la texture et la couleur sont mutuellement indépendantes. La différence avec l'approche séquentielle est que les attributs de couleur et de texture sont extraits séparément puis combinés dans un seul vecteur d'attributs (Figure 1.4). Dans ce cas, les attributs de couleur sont calculés à partir de la distribution des couleurs dans l'espace de représentation et les attributs de texture sont extraits de l'image en niveaux de gris sans tenir compte de la couleur.

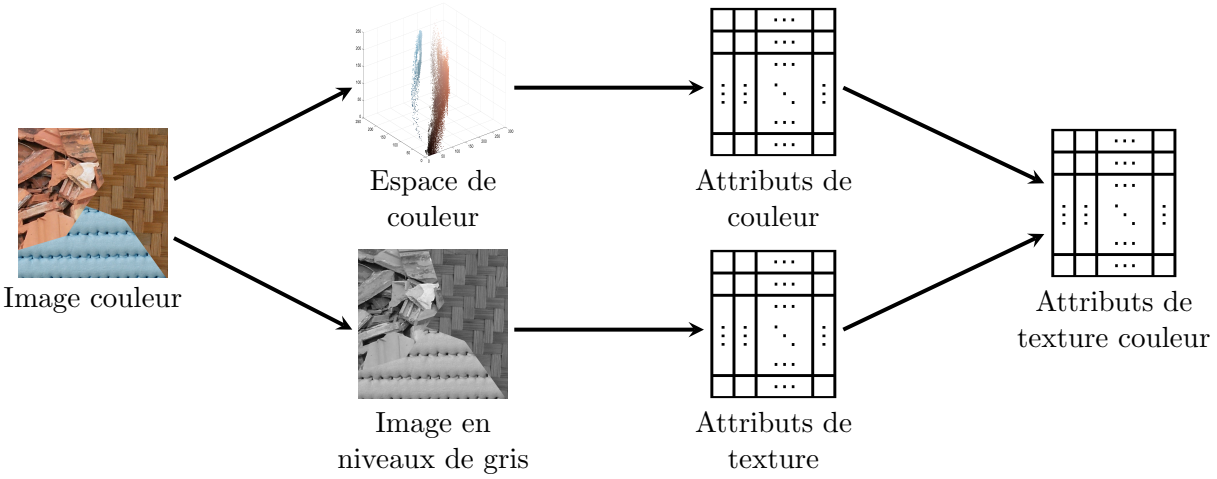


Figure 1.4: Combinaison parallèle de la couleur et de la texture.

À titre d'exemple, Permuter et al. [55] ont proposé une méthode de segmentation d'images dans laquelle les attributs de texture calculés à partir de l'image en niveaux de gris, moyennant les modèles autorégressifs, les ondelettes, et la transformée en cosinus discret, sont concaténés avec des attributs de couleur (moyenne et covariance) extraits à partir des composantes des espaces de couleur RGB et L$^*u^*v^*$. La méthode de segmentation proposée par Kim et Hong [56] combine quant à elle dans un seul vecteur, les attributs de texture dérivés des filtres de Gabor et les attributs de couleur représentés par les composantes de l'espace de couleur RGB. Akbuluta et al. [57] combinent les attributs de couleur L$^*u^*v^*$ avec les attributs de texture extraits à partir des filtres de Hermite appliqués à l'image en niveaux de gris. Subudhi et Mukhopadhyay [58] ont caractérisé les textures couleur par des attributs de texture extraits sur l'image en niveaux de gris via la transformée en ondelettes et par des attributs de couleur représentés par la médiane des trois composantes de l'espace de couleur HSV. Dans les méthodes de segmentation d'images proposées dans [7, 59–61], les textures couleur sont caractérisées en combinant dans un seul vecteur les attributs de couleur relatives aux composantes de l'espace de couleur L$^*a^*b^*$ et les attributs de texture extraits par l'intermédiaire des filtres de Gabor sur l'image en niveaux de gris.

1.4.1.3 Approche conjointe

Cette approche, qualifiée d'intégration implicite par Ilea et Whelan [50], suppose que les informations de couleur et de texture sont mutuellement dépendantes. Deux stratégies peuvent être employées (Figure 1.5). La première prend en considération uniquement des relations intra-composantes. Les attributs de texture sont extraits de chaque composante de couleur sans tenir compte d'aucune interaction spatiale entre les niveaux des autres composantes de couleur. La seconde stratégie tient compte à la fois des relations inter et intra composantes. Les attributs de texture sont calculés à la fois sur chacune des composantes couleur et entre tous les couples de composantes couleur.

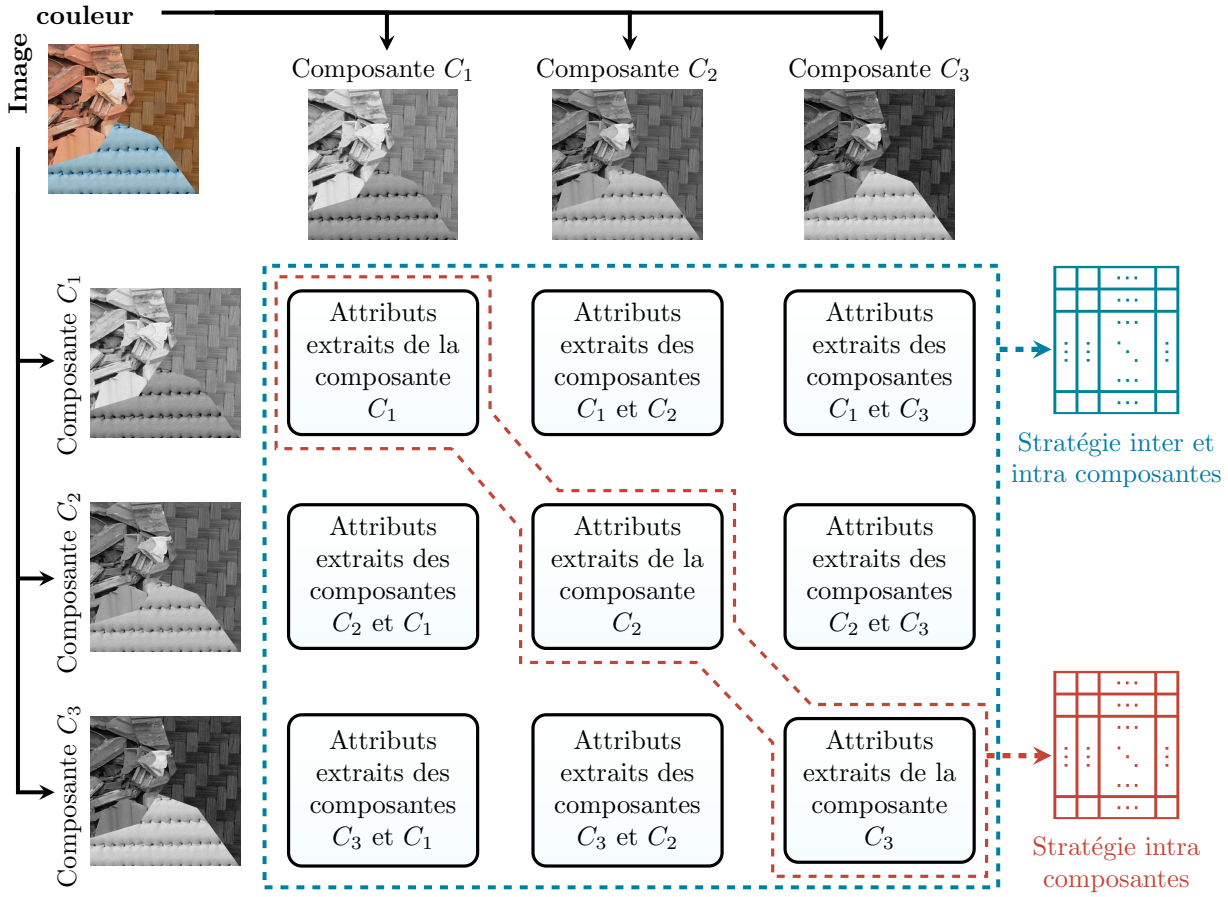


Figure 1.5: Combinaison conjointe de la couleur et de la texture. La première stratégie est représentée par les pointillés rouges et la deuxième stratégie par les pointillés bleus.

À titre d'exemple, Liu et al. ont proposé une méthode de segmentation d'images couleur texturées basée sur les histogrammes locaux [49]. Ces derniers sont utilisés comme attributs de texture et sont calculés pour chaque pixel sur chacune des composantes de l'espace de couleur RGB (intra-composantes). Silva et al. [47] ont développé une méthode de segmentation

d'images dans laquelle chaque pixel de l'image est caractérisé par un ensemble d'attributs de texture calculé en utilisant la transformée en ondelettes sur chacune des composantes de l'espace de couleur YCbCr (intra-composantes). Dans la méthode de segmentation proposée par Maheswari et al. [48], chaque pixel de l'image est caractérisé par un ensemble d'attributs d'Haralick [20]. Ces derniers sont calculés à partir des matrices de co-occurrences couleur inter-composantes de l'espace de couleur HSI. Pour caractériser les pixels de l'image, Benomar et al. [6] ont également utilisé les attributs d'Haralick mais extraits à partir des matrices de co-occurrences couleur inter et intra composantes de trois différents espaces de couleur (RGB, HSV, et YUV).

1.4.2 Catégorisation selon le partitionnent de l'image en régions

La catégorisation établie par Ilea et Whelan [50], adoptée par une grande majorité d'auteurs, ne tient pas compte d'une manière explicite de la façon dont les régions texturées couleur sont formées. Pour compléter cet état de l'art, nous proposons de regrouper les méthodes de segmentation d'images couleur texturées en cinq approches :

1.4.2.1 Approche contour

Les méthodes de segmentation par approche contour sont fondées sur la recherche des frontières (contours) qui séparent les différentes régions homogènes de l'image. Elles reposent sur la détection des discontinuités en termes d'intensité, de couleur, ou de texture dans l'image. Ces méthodes peuvent être classées en *méthodes dérivatives* et en *méthodes déformables*.

Les méthodes dérivatives s'appuient sur des filtres dérivateurs tels que les opérateurs gradient de Sobel, Prewitt, et Roberts, l'opérateur Laplacien et les filtres optimaux comme les filtres de Canny et Deriche [62]. Elles produisent des contours non fermés et bruités sur des images en niveaux de gris ou couleur, mais sont difficilement exploitables dans le cas des images texturées, et de surcroît lorsqu'elles sont en couleur. Tandis que les méthodes déformables (contours actifs) sont basées sur une courbe (fermée ou non) placée près des frontières de la région à détecter. Cette courbe est ensuite déplacée, d'une manière itérative, tout en se déformant afin d'épouser les contours finaux de la région [63]. Contrairement aux méthodes dérivatives, les contours actifs ont été largement exploités dans la segmentation d'images couleur texturées [64–67].

1.4.2.2 Approche région

Les méthodes de segmentation par approche région visent à regrouper les pixels connexes ayant les mêmes caractéristiques (niveaux de gris, couleurs, et/ou textures) en régions homogènes. Contrairement à l'approche contour qui recherche des dissimilarités entre les pixels, l'approche région recherche plutôt les similarités entre les pixels. Dans cette approche, on trouve *les méthodes par croissance de régions* et *les méthodes par division-fusion de régions*.

Les méthodes par division-fusion de régions divisent itérativement l'image en plusieurs régions homogènes. Tout d'abord, elles traitent l'image comme une région, puis cette même région est divisée en zones de tailles plus petites. Ce processus est répété sur chaque nouvelle région jusqu'à ce que aucune région non homogène ne soit obtenue. Dans le cas des images couleur texturées, le critère d'homogénéité est basé sur le calcul des attributs de texture couleur [68, 69]. Ce processus de division peut engendrer un grand nombre de régions (sur-segmentation) dont certaines sont de tailles très petites. On procède alors à une étape de fusion de ces régions. Celle-ci consiste à agréger successivement les petites régions voisines à d'autres régions de sorte que la résultante respecte un critère d'homogénéité en termes de couleur et de texture. Cette démarche a été empruntée dans [9, 53, 68–71] pour segmenter des images couleur texturées. Quant aux méthodes par croissance de régions, elles considèrent au départ un ensemble de petites régions uniformes dans l'image. Ces régions, appelées *germes*, sont composées d'un ou de quelques pixels. Elles croient ensuite au fur et à mesure par l'incorporation des pixels voisins en fonction d'un ou plusieurs critères de similarités [72]. Pour des images couleur texturées, le critère de similarité doit tenir compte à la fois de l'information de texture et celle de la couleur. Cette stratégie a été exploitée dans [52, 72–75] pour segmenter des images couleur texturées.

1.4.2.3 Approche basée sur la coupe de graphe

Les méthodes de segmentation basées sur la coupe de graphe consistent à créer un graphe à partir de l'ensemble des pixels de l'image et exploiter les différentes propriétés de la théorie des graphes pour trouver les différentes régions de l'image. L'idée générale est d'abord de modéliser les pixels (régions) par les nœuds de ce graphe et les caractéristiques (distance, similarité, etc.) reliant ces pixels (régions) par des arêtes. Ensuite, de partitionner le graphe en un ensemble de sous-graphes représentant chacun une région homogène de l'image. Parmi les méthodes clés de coupe de graphe, on trouve la coupe minimale [76], la coupe normalisée [77], et flow maximal/coupe minimale [78]. Ces méthodes ont été largement appliquées à la segmentation d'images couleur texturées [56, 58, 79–82].

1.4.2.4 Approche par classification des pixels

Les méthodes de segmentation par classification des pixels peuvent être aussi classées dans l'approche région. Cependant, ces méthodes ne tiennent pas compte de la disposition spatiale des pixels, mais uniquement des propriétés comme l'intensité, la couleur, la texture, etc. Elles partent de l'hypothèse que les pixels ayant les mêmes propriétés forment une classe dans l'image. Ces méthodes cherchent à identifier les classes présentes dans l'image et à affecter à chaque pixel une étiquette indiquant la classe à laquelle il appartient. Ainsi, les pixels appartenant à une classe peuvent former plusieurs régions non adjacentes dans l'image. La segmentation proprement dite en régions n'est obtenue qu'après analyse de la connexité des pixels appartenant à la même classe. En fonction des informations a priori disponibles, la classification peut être soit: *non supervisée*, *supervisée*, ou *semi-supervisée*.

La classification non supervisée tente de regrouper les pixels de l'image en classes sans aucune connaissance a priori sur l'appartenance de ces pixels aux classes, ni la présence de prototypes de classes. Parmi les méthodes de classification non supervisée appliquées à la segmentation des images couleur texturées, on trouve les algorithmes k-means [83–85], fuzzy c-means [86, 87], mean-shift [57, 88], les cartes topologiques auto-organisatrices de Kohonen [89, 90], et la classification spectrale [91, 92].

La classification supervisée nécessite la présence d'un ensemble de pixels étiquetés (prototypes). Cet ensemble est utilisé pour définir un modèle de classification lors d'une phase d'apprentissage [93]. Segmenter une image consiste alors à affecter ses pixels à l'une des classes définies lors de la phase d'apprentissage. Parmi les méthodes de classification supervisée appliquées à la segmentation d'images couleur texturées, on peut citer le boosting adaptatif [94], les machines à vecteurs de support (SVM) [5, 6, 95], et les réseaux de neurones comme la machine à apprentissage extrême (ELM) [96].

En classification semi-supervisée, seul quelques pixels de l'image sont étiquetés. Ils sont exploités pour regrouper les pixels non étiquetés en classes. Ce type de classification relie les avantages des deux autres classifications, mais reste très peu impliquée dans la segmentation des images couleur texturées. Parmi les travaux trouvés dans la littérature, on peut citer la classification hiérarchique semi-supervisée [97], les SVM transductifs [98] qui est la version semi-supervisée de SVM, et la classification spectrale semi-supervisée [99, 100].

La classification constitue une partie importante de notre travail. Un état de l'art sur les différentes méthodes de classification dans les trois contextes *supervisé*, *non supervisé*, et *semi-supervisé* est dressé dans la section 1.5.

1.4.2.5 Approche par apprentissage profond

L'approche par apprentissage profond est similaire à celle de la classification supervisée avec la seule différence qu'elle ne requiert aucun choix d'une technique spécifique d'extraction d'attributs. Les méthodes par apprentissage profond sont basées sur les réseaux de neurones artificiels et plus particulièrement sur les réseaux de neurones convolutifs (CNN) [101]. Un CNN est composé d'une succession de plusieurs types de couches de neurones (convolution, fonction d'activation, et pooling) jouant le rôle d'extracteur d'attributs et d'un ensemble de couches de neurones entièrement connectées se comportant comme un classifieur. Il reçoit en entrée des images et génère automatiquement des attributs à partir des images de référence. Il est en même temps entraîné pour réaliser une tâche de classification. Lors de la phase de segmentation, le réseau reçoit en entrée l'image originale et produit en sortie une image segmentée. Plusieurs CNNs ont été développés pour la segmentation d'images couleur texturées [102, 103] et parmi lesquels on peut citer: fully convolutional network (FCN) [104, 105], empirical wavelet transform based fully convolutional network for texture (EWT-FCNT) [103], SegNet [106], U-Net [107], pyramid scene parsing network (PSP-Net) [108], deep visual model (DA) [109], deep semantic segmentation network (DSSN) [110], et multi-scale convolutional neural networks (MSCNN) [111].

1.4.3 Catégorisation selon le contexte d'apprentissage

La notion de supervision n'est pas propre à la classification. Les méthodes de segmentation peuvent être aussi regroupées en trois catégories selon le contexte d'apprentissage: *non supervisé*, *supervisé*, et *semi-supervisé*.

1.4.3.1 Segmentation non supervisée

La segmentation d'images non supervisée vise à partitionner les images en régions de manière automatique et sans l'utilisation d'aucune information a priori sur les régions. Les méthodes de segmentation non supervisée ne nécessitent pas de phase d'apprentissage et ne sont pas exposées au problème de généralisation. Outre les méthodes basées sur la classification non supervisée citées précédemment (Section 1.4.2.4), on retrouve les méthodes basées sur la coupe de graphe (texNcut [82]), les méthodes par division-fusion de régions (dynamic hierarchical classification (DHC) [71], recursive texture fragmentation and reconstruction (R-TFR) [9], et eCognition (eCog) [75]), les méthodes par apprentissage des dictionnaires

basées sur la représentation parcimonieuse (dictionary learning based sparse representation (DLSRC) [112]), les méthodes basées sur les modèles variationnels du Mumford-Shah (variational multi-phase segmentation (PCA-MS) [59] et model-based learning of local image features (MLLIF) [113]), la méthode factorization-based texture segmentation (FSEG) [61] laquelle est basée sur la factorisation de la matrice des attributs, ainsi que d'autres méthodes comme: environment for visualizing images feature extraction module (ENVI) [114], occlusion of random textures segmentation (ORTSEG) [7], et celles basées sur les CNNs comme unsupervised fully convolutional network for texture (FCNTunsup) [105].

1.4.3.2 Segmentation supervisée

La segmentation d'images supervisée nécessite la présence d'un ensemble de connaissances a priori sur les différentes régions recherchées (base d'apprentissage), qui est utilisé comme vérité terrain pour définir un modèle de segmentation lors d'une phase d'apprentissage. La base d'apprentissage peut contenir soit un ensemble d'images représentant les différentes régions de l'image, ou bien un ensemble de pixels (régions) étiquetés appelés *pixels prototypes*. Les méthodes de cette approche englobent les méthodes basées sur la classification supervisée (Section 1.4.2.4) et celles basées sur les CNNs (Section 1.4.2.5).

1.4.3.3 Segmentation semi-supervisée

En segmentation d'images semi-supervisée, on dispose d'une quantité d'information a priori exprimée sous la forme de pixels prototypes issus des images vérité terrain, d'annotations établies par des experts sur les images à segmenter, ou de contraintes *must-link* et *cannot-link* indiquant si deux pixels appartiennent à la même classe ou non. Les méthodes de segmentation semi-supervisée combinent cette information a priori avec l'information issue des pixels dont on n'a aucune information a priori. Ce type de segmentation est censé relier les avantages de la segmentation supervisée et de la segmentation non supervisée. Dans la littérature, la segmentation semi-supervisée est qualifiée de segmentation sous contraintes (constrained segmentation), faiblement supervisée (weakly supervised), ou encore de segmentation interactive (interactive segmentation). Parmi les méthodes proposées dans la littérature, on retrouve les méthodes basées sur la classification semi-supervisée (Section 1.4.2.4), la méthode FSEG^{+} [60] qui est une version semi-supervisée de FSEG, la méthode weakly-supervised sparse coding geometric prior (WSSCGP) [60], et les méthodes basées sur la coupe de graphe (block-based graph cut (BGC) [58] et weakly-supervised graph cut (WSGC) [115]).

1.5 Classification des données

La classification est une procédure très générale, qui peut s'appliquer non seulement pour segmenter des images, mais aussi pour classifier tout type de données. Les travaux entrepris dans le cadre de cette thèse concernent la sélection d'attributs et la segmentation d'images couleur texturées. Ces deux thématiques sont étroitement liées et partagent les mêmes contextes de classification. C'est la raison pour laquelle nous avons choisi de présenter dans cette partie du manuscrit les différentes méthodes de classification dans un cadre plus général, c.-à-d. celui des données.

Dans ce cas, on dispose d'un ensemble de n données $X = \{x_1,x_2,\ldots,x_n\}$ et qui peut être représenté par une matrice $\boldsymbol{X} = [\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_n] \in \mathbb{R}^{n\times d}$. Chaque donnée $\boldsymbol{x}_i = [x_{i1},x_{i2},\ldots,x_{id}]$ est caractérisée par un ensemble de d attributs noté $F_d = \{f_1,f_2,\ldots,f_d\}$. Dans le cadre de la segmentation d'images couleur texturées, une donnée correspond à un pixel et les d attributs désignent des attributs de texture couleur. D'un point de vue géométrique, chaque donnée peut être représentée par un point dans un espace $\mathbb{R}^d$, appelé *espace des attributs*. L'ensemble de données définit un nuage de points dans l'espace des attributs. La classification a pour but d'assigner chaque donnée x_i à l'une des k classes, notée ω^l, $l = 1,\ldots,k$.

En plus des données, on dispose souvent d'une certaine quantité d'information a priori qui se présente soit sous la forme de données étiquetées indiquant les classes auxquelles elles appartiennent, soit sous la forme de contraintes *must-link* et *cannot-link* spécifiant si deux données doivent ou non être dans la même classe [116]. En fonction de la quantité et/ou du type d'information disponible, on distingue trois contextes de classification: *non supervisé*, *supervisé*, et *semi-supervisé*.

1.5.1 Classification non supervisée

Dans un contexte non supervisé, aucune information a priori sur l'appartenance des données aux classes n'est disponible. Dans ce cas, la classification non supervisée, ou clustering en anglais, vise à regrouper l'ensemble de données non étiquetées, noté, $X^U = X$ en k classes. Le nombre de classes k peut être par contre connu ou inconnu a priori. Les méthodes de classification non supervisée sont généralement basées sur des mesures de similarités ou de distances entre les données (Tableaux 2.1 et 2.2). Elles peuvent être regroupées en cinq catégories: *méthodes hiérarchiques*, *méthodes à centres mobiles*, *méthodes basées sur la densité*, *méthodes probabilistes*, et *méthodes spectrales*.

1.5.1.1 Méthodes hiérarchiques

Les méthodes de classification non supervisée hiérarchiques sont basées sur le calcul de la distance (similarité) pour construire les classes. Ces méthodes sont soit *descendantes* (*divisives*), soit *ascendantes* (*agglomératives*). Les méthodes de classification descendante hiérarchique (HDC) considèrent au début une seule classe contenant toutes les données. Cette classe est ensuite divisée en deux selon un critère donné. Ce processus de division est réitéré aux nouvelles classes jusqu'à ce que chaque donnée forme une classe. Les méthodes de classification ascendante hiérarchique (HAC) considèrent initialement chaque donnée comme une classe. Puis, les classes les plus similaires sont fusionnées de manière itérative jusqu'à obtenir une seule classe qui contient toutes les données. Pour ces deux types de méthodes, les classes sont représentées par un graphe appelé *dendrogramme*. Le résultat de classification est obtenu en coupant le dendrogramme à un niveau donné afin d'obtenir une partition des données.

En pratique, les méthodes ascendantes sont plus utilisées que les méthodes descendantes car elles sont moins complexes et plus rapides. Ces méthodes sont décrites par l'algorithme 1.1. Elles diffèrent selon la métrique d'agrégation D entre les classes de données, comme celles de "single-linkage" (D_{SL}) et de "Ward" (D_{Ward}) [117, 118]:

$$D_{SL}(\omega^1,\omega^2) = \min_{x_i \in \omega^1, x_j \in \omega^2} \delta(\boldsymbol{x}_i,\boldsymbol{x}_j) \tag{1.1}$$

$$D_{Ward}(\omega^1,\omega^2) = \frac{|\omega^1||\omega^2|}{|\omega^1| + |\omega^2|} \delta^2(\bar{\boldsymbol{x}}_{\omega^1},\bar{\boldsymbol{x}}_{\omega^2}) \tag{1.2}$$

$\delta(\boldsymbol{x}_i,\boldsymbol{x}_j)$ étant la distance entre deux données x_i et x_j, et $\delta(\bar{\boldsymbol{x}}_{\omega^1},\bar{\boldsymbol{x}}_{\omega^2})$ la distance entre les deux centres de gravité $\bar{x}_{\omega^1}$ et $\bar{x}_{\omega^2}$ correspondant respectivement aux classes ω^1 et ω^2.

Algorithme 1.1: HAC.

Entrée: Ensemble de données $X = \{x_1,x_2,\ldots,x_n\}$, nombre désiré de classes k.

1. Créer n classes dont chacune contient une donnée x_i.
2. Calculer les similarités entre les classes selon la métrique d'agrégation choisie.
3. Fusionner les deux classes les plus similaires et mettre à jour le dendrogramme.
4. Répéter les étapes 2-3 jusqu'à ce que toutes les données soient regroupés dans une seule et unique classe.
5. Sélectionner un niveau de coupe du dendrogramme pour obtenir les k classes.

Sortie: La classe de chaque donnée x_i de l'ensemble X.

La méthode HAC possède deux principaux inconvénients: un temps de calcul élevé qui limite son utilisation uniquement aux petits ensembles de données et un choix non évident du niveau de coupe du dendrogramme permettant d'obtenir les classes.

1.5.1.2 Méthodes à centres mobiles

Les méthodes de classification non supervisée à centres mobiles sont rapides et facilement applicables sur de grands ensembles de données tels que les images. L'idée principale de ces méthodes est de choisir une partition initiale de l'ensemble de données à classer puis déplacer les éléments d'une classe à une autre pour obtenir une meilleure partition selon un critère défini a priori. Parmi ces méthodes, on trouve: les algorithmes k-means [54] et fuzzy c-means (FCM) [119].

Méthode k-means Cette méthode, proposée par Mac Queen en 1967 [54], est très populaire et demeure la plus utilisée en segmentation d'images à cause de sa simplicité et rapidité [83–85]. Son principe consiste à choisir aléatoirement k données pour représenter les centres initiaux des classes (centroïdes). Chaque donnée x_i de l'ensemble X est ensuite affectée au centroïde le plus proche. Les centroïdes sont ensuite mis à jour en calculant la moyenne des données dans chaque classe. Les procédures d'affectation et de mise à jour des centroïdes sont répétées jusqu'à ce que aucun centroïde ne change de position. Les principales étapes de la méthode k-means sont résumées dans l'algorithme 1.2.

Algorithme 1.2: k-means.

Entrée: Ensemble de données $X = \{x_1, x_2, \ldots, x_n\}$, nombre de classes k.

1. Initialiser aléatoirement k centroïdes $\bar{x}_{\omega^1}, \ldots, \bar{x}_{\omega^k}$.
2. Assigner chaque donnée $x_i \in X$ à la classe du centroïde le plus proche.
3. Mettre à jour les centroïdes de chaque classe en calculant la moyenne des données contenues dans chaque classe.
4. Répéter les étapes 2-3 jusqu'à ce que les centroïdes ne changent pas de position.

Sortie: La classe de chaque donnée x_i de l'ensemble X.

Cependant, la méthode k-means est sensible au choix des centres initiaux de classes. Pour contourner ce problème, une solution consiste à appliquer l'algorithme plusieurs fois sur le même ensemble de données, ensuite choisir la partition qui minimise l'erreur quadratique moyenne entre les données et les centroïdes de leur classe respective.

Méthode FCM Cette méthode, développée par Bezdek et al. en 1984 [119], est la version floue de la méthode k-means. Son idée de base est que chaque donnée x_i de l'ensemble X peut appartenir à toutes les classes ω^l, $l = 1,\ldots,k$, avec des degrés d'appartenances compris entre 0 et 1. FCM est basée sur la minimisation de la fonction objective J_{FCM} définie par:

$$J_{FCM}(\boldsymbol{U},\bar{X}) = \sum_{i=1}^{n}\sum_{l=1}^{k}(u_{i,l})^m \times \delta^2(\boldsymbol{x}_i,\bar{\boldsymbol{x}}_{\omega^l}) \tag{1.3}$$

$\bar{x}_{\omega^l}$ est le centroïde de la classe ω^l, et $\bar{X}$ un ensemble qui regroupe les k centroïdes. $\boldsymbol{U}$ est une matrice dont chaque élément $u_{i,l}$ représente le degré d'appartenance de la donnée x_i à la classe du centroïde $\bar{x}_{\omega^l}$, et $m \geq 2$ le facteur flou.

Comme k-means, la méthode FCM est itérative dans laquelle les degrés d'appartenances $u_{i,l}$ et les centroïdes $\bar{x}_{\omega^l}$ sont mis à jour à chaque itération comme suit:

$$\bar{\boldsymbol{x}}_{\omega^l} = \frac{\sum_{i=1}^{n}(u_{i,l})^m \boldsymbol{x}_i}{\sum_{i=1}^{n}(u_{i,l})^m} \tag{1.4}$$

$$u_{i,l} = \frac{1}{\sum_{j=1}^{k}\left(\frac{\delta^2(\boldsymbol{x}_i,\bar{\boldsymbol{x}}_{\omega^l})}{\delta^2(\boldsymbol{x}_i,\bar{\boldsymbol{x}}_{\omega^j})}\right)^{\frac{2}{m-1}}} \tag{1.5}$$

Les principales étapes de la méthode FCM sont résumées dans l'algorithme 1.3.

Algorithme 1.3: FCM.

Entrée: Ensemble de données $X = \{x_1,x_2,\ldots,x_n\}$, nombre de classes k, facteur flou m, ϵ seuil d'arrêt

1. Initialiser aléatoirement k centroïdes $\bar{x}_{\omega^1},\ldots,\bar{x}_{\omega^k}$.
2. Mettre à jour la matrice $\boldsymbol{U}$ en utilisant l'équation (1.5).
3. Calculer les nouveaux centroïdes en utilisant l'équation (1.4).
4. Calculer la nouvelle fonction objective J_{FCM} en utilisant l'équation (1.3).
5. Répéter les étapes 2-4 jusqu'à la satisfaction du critère d'arrêt $|{J_{FCM}}^{(t)} - {J_{FCM}}^{(t+1)}| < \epsilon$, t étant la t-éme itération.

Sortie: La classe de chaque donnée x_i de l'ensemble X.

1.5.1.3 Méthodes basées sur la densité

Les méthodes de classification non supervisée basées sur la densité supposent que les classes sont des zones denses, séparées par des zones moins denses dans l'espace des attributs.

La densité d'un point de données est fonction du nombre de points présents dans un rayon autour du point sélectionné. Un point de données avec une densité élevée peut être considéré comme un point qui appartient au centre d'une classe. Alors qu'un point de données avec une densité faible peut être considéré comme un point qui appartient à la périphérie d'une classe. Parmi les méthodes de cette catégorie on peut citer: density-based spatial clustering of applications with noise (DBSCAN) [120] et density decreased chain (DDC) [121].

Méthode DBSCAN Cette méthode, introduite par Ester et al. en 1996 [120], consiste à déterminer pour chaque donnée x_i de l'ensemble X si elle appartient ou non à une zone dense. Pour cela, elle nécessite deux paramètres : le rayon de voisinage ϵ autour d'un point de données et le nombre minimum de points minP qui doivent être contenus dans ϵ. Selon les valeurs de minP et de ϵ, un point de données peut être soit: un *point noyau* (s'il contient dans son voisinage de rayon ϵ au moins minP points de données), un *point frontière* (s'il contient moins de minP points dans son voisinage mais qui possède parmi ses voisins au moins un point noyau), ou un *point bruit* (s'il n'est ni point noyau ni point frontière). DBSCAN commence par parcourir les points de l'ensemble X jusqu'à trouver un point noyau x_i qui devient générateur de classe. Les points voisins de x_i et qui n'ont pas d'étiquettes sont affectés à la même classe que x_i. S'il y a des points noyaux parmi les nouveaux points affectés, alors ils vont propager la génération de la classe selon le même principe. Cette opération est répétée pour tous les points restants. Les points non assignés sont alors considérés comme des points bruits. L'algorithme 1.4 résume les principales étapes de la méthode DBSCAN.

Algorithme 1.4: DBSCAN.

Entrée: Ensemble de données $X = \{x_1, x_2, \ldots, x_n\}$, paramètres ϵ et minP.

1. Choisir aléatoirement un point x_i non traité de l'ensemble de données X.
2. Si x_i est un point noyau, créer une nouvelle classe ω^l et assigner les points de son voisinage ϵ à la classe ω^l, sinon et aller à l'étape 5.
3. Pour chaque point de données dans le voisinage ϵ de x_i:
   - S'il est un point noyau, alors ajouter tous ses voisins à la classe ω^l.
4. Répéter les étapes 1-3 jusqu'à ce que tous les points de l'ensemble X soient traités.
5. Les points non assignés à des classes sont des points bruits.

Sortie: La classe de chaque donnée x_i de l'ensemble X.

Méthode DDC Cette méthode récente, proposée par Li et Cai en 2022 [121], est basée sur le concept de chaîne de points de densité décroissante, qui permet de mieux traiter des

ensembles de données ayant des classes de densités et de formes variées. DDC consiste à construire un graphe des K plus proches voisins (KNN) mutuels à partir de l'ensemble de données X et de déterminer localement la densité de chaque point x_i en fonction de ses plus proches voisins mutuels dans un rayon ϵ. Les points dont la densité est la plus élevée parmi leurs plus proches voisins sont alors définis comme les centres de densité. Des chaînes de nœuds reliant les points dans l'ordre décroissant de leurs densités sont construites sur le graphe des KNN mutuels. Le premier maillon de chaque chaîne correspond à un centre de densité. Les points d'une même chaîne ayant des densités proches de celle du centre sont alors considérés comme faisant partie du noyau d'une classe, tandis que les autres points de cette chaîne sont considérés comme des points frontières. Une procédure de chaînage des points noyaux d'une même classe (intra-classes) est alors appliquée afin de détecter les classes représentatives. Les étapes de DDC sont décrites dans l'algorithme 1.5.

Algorithme 1.5: DDC.

Entrée: Ensemble de données $X = \{x_1, x_2, \ldots, x_n\}$, paramètres K et λ, nombre de classes k.

1. Construire le graphe des KNN mutuels de X.
2. Identifier les centres de densité locale du graphe des K plus proches voisins mutuels.
3. Diviser les points de X en points noyaux et points frontières.
4. Regrouper dans une même classe les points noyaux appartenant à la même chaîne de sensité intra-classe.
5. Regrouper les classes qui se chevauchent et affecter les points frontières aux classes.
6. Éliminer les classes peu denses et affecter leurs points aux autres classes.

Sortie: La classe de chaque donnée x_i de l'ensemble X.

1.5.1.4 Méthodes probabilistes

Les méthodes de classification non supervisée de type probabiliste font appel à la notion de densité de probabilité pour décrire la distribution des données. Elles se déclinent en deux variantes: *paramétriques* et *non paramétriques*.

L'hypothèse paramétrique repose sur la fixation préalable d'un modèle aux fonctions de densités de probabilités conditionnelles de chaque classe, généralement gaussienne. La fonction densité de probabilité (PDF) en un point est calculée en combinant k composantes ou fonctions de densité de probabilité conditionnelle, pondérées par leurs probabilités a priori. Les paramètres du modèle correspondant à chaque classe, ainsi que les probabilités a priori des

classes, constituent les paramètres du mélange que nous cherchons à identifier à partir de l'ensemble de données à analyser. L'estimation de ces paramètres est réalisée à l'aide d'une méthode itérative connue sous le nom d'estimation-maximisation (EM) [122].

Sous l'hypothèse non paramétrique, aucune restriction n'est faite quant à la loi de répartition des données et aucune information a priori sur la structure des données à analyser n'est requise. Dans ce cas, la classification s'appuie sur l'estimation des PDF sous-jacente à l'ensemble des données à classer, où chaque mode de cette fonction correspond à une classe ou à des zones denses dans l'espace des attributs. Le problème de classification est donc posé en termes de détection des modes [123]. Dans cette optique, nous trouvons la méthode mean-shift (MS) [123, 124] qui est basée sur la détermination des modes (maxima locaux) de la PDF de l'ensemble de données X. Concrètement, l'algorithme mean-shift consiste à appliquer pour chaque donnée x_i les étapes suivantes:

1. Déterminer ses K plus proches voisins x_j dans l'espace des attributs.
2. Calculer ses centres de gravité:
$$\bar{\boldsymbol{x}}_i = \frac{1}{K} \sum_{j=1}^{K} \boldsymbol{x}_j \tag{1.6}$$
3. Déplacer la donnée x_j vers son centre de gravité.
4. Réitérer le processus de l'étape 1 jusqu'à convergence.

À la fin de cette procédure, chaque donnée aura convergé vers des zones de fortes densités (modes). La classification finale est obtenue en regroupant dans une même classe toutes les données qui convergent vers le même mode.

1.5.1.5 Méthodes spectrales

Les méthodes spectrales sont issues de la théorie spectrale des graphes [125]. Elles considèrent la tâche de classification comme un problème de coupe de graphe qui peut être résolu par le calcul des vecteurs et valeurs propres d'une matrice dite laplacienne. Cette dernière est calculée à partir d'un graphe de similarité qui représente la structure de l'ensemble de données. Dans ce graphe, les nœuds représentent les données et le poids des arêtes entre chaque paire de nœuds représente la similarité entre les deux données. Les méthodes spectrales possèdent plusieurs avantages, elles permettent entre autres de détecter des classes de formes complexes et non linéairement séparables.

Le chapitre suivant est entièrement dédié aux fondements des méthodes de classification spectrale et à leurs mises en œuvre.

1.5.2 Classification supervisée

Dans un contexte supervisé, on dispose a priori d'un ensemble de données étiquetées X^P, dites d'apprentissage ou prototypes, dont l'appartenance aux classes est connue. Ces prototypes sont ensuite utilisés pour prédire la classe d'appartenance d'une donnée non étiquetée. X^P doit donc contenir un nombre suffisant de données représentatives de chaque classe. La classification supervisée se déroule en deux phases: *apprentissage* et *décision*. Lors de la phase d'apprentissage, l'ensemble de données étiquetées X^P est utilisé pour construire un modèle de classification (classifieur). Ces modèles peuvent être de simples centroïdes, des PDF, des surfaces séparatrices entre les classes, des arbres de décision, ou des réseaux de neurones. Lors de la phase de décision, le classifieur permet d'affecter les données non étiquetées X^U à l'une des classes définies lors de la phase d'apprentissage. Un grand nombre de méthodes de classification supervisée sont proposées dans la littérature. Les plus connues sont: KNN [126], SVM [127], MLP [128], et le classifieur naïf bayèsien [129].

Un état de l'art sur les méthodes de classification supervisée est dressé dans [93]. Nous faisons ci-dessous un bref aperçu sur les méthodes SVM et KNN vues leur utilisation par la suite dans cette thèse.

Méthode KNN Cette méthode, introduite par Cover et Hart en 1967 [126], est considérée comme l'une des méthodes de classification supervisée les plus simples et les plus intuitives. Son idée de base est de classer les données non étiquetées X^U en fonction de leurs distances par rapport à l'ensemble des données étiquetées X^P. La phase d'apprentissage est superflue car aucun modèle n'est introduit à partir des prototypes. Tous les calculs sont effectués lors de la phase de décision. Pour chaque donnée à classer x_i, la liste des K plus proches voisins parmi les données étiquetées est établie puis la classe majoritaire de ses K plus proches voisins est assignée à x_i. La méthode KNN est performante lorsque le nombre de données étiquetées est élevé.

Méthode SVM Cette méthode, proposée par Vapnik en 1995 [127], constitue l'une des méthodes de classification supervisée les plus populaires. Elle peut à la fois traiter des données de grande dimension et produire de bons résultats avec de petites bases d'apprentissage. Son idée principale est de trouver dans l'espace des attributs un hyperplan capable de séparer au mieux les données prototypes appartenant à deux classes différentes en maximisant une distance appelée *marge* (Figure 1.6). Cette marge est définie à partir de quelques prototypes appelés *vecteurs supports*. Lors de la phase de décision, une donnée non étiquetée est affectée à l'une des classes selon sa position par rapport à l'hyperplan.

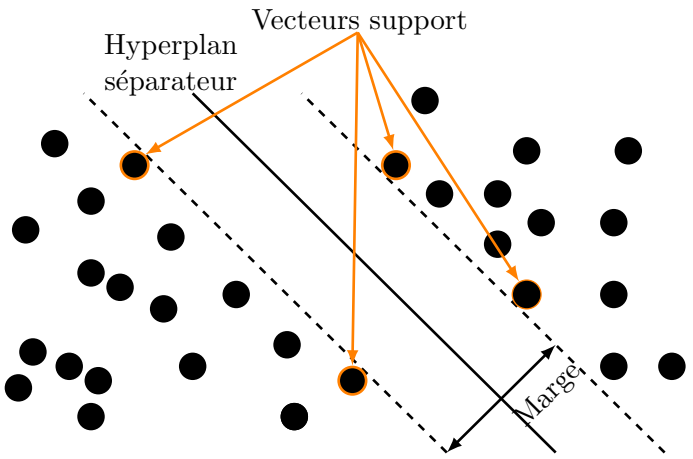


Figure 1.6: Illustration du principe de la méthode SVM.

La méthode SVM est à la base conçue pour effectuer une classification binaire (2 classes). Elle a été ensuite étendue au cas multi-classes en utilisant soit la technique *un-contre-tous* [127], qui consiste à déterminer pour chaque classe un hyperplan qui la sépare de toutes les autres classes, soit la technique *un-contre-un* [130], qui consiste à utiliser un SVM binaire pour chaque paire de classes. De plus, pour traiter des classes de données non linéairement séparables, SVM a été adaptée en projetant les données dans un espace de dimension plus grande, dans lequel il est possible d'effectuer une séparation linéaire des classes.

1.5.3 Classification semi-supervisée

Dans un contexte semi-supervisé, on dispose à la fois d'un ensemble de données non étiquetées X^U et d'une très faible quantité d'information a priori exprimée, soit sous la forme de quelques données étiquetées X^P, soit sous la forme de contraintes must-link et cannot-link.

La classification semi-supervisée se situe entre la classification supervisée et non supervisée [131]. Son objectif est de combiner l'information contenue dans les données non étiquetés avec l'information a priori disponible pour atteindre des performances de classification plus élevées. L'information a priori permet à la fois de valider le résultat de classification et de guider la classification vers des solutions optimales. La classification semi-supervisée peut être abordée sous deux points de vue. Dans le premier, les données non étiquetées sont utilisées pour améliorer une méthode de classification supervisée, et dans le second, l'information a priori est exploitée pour améliorer une méthode de classification non supervisée.

Dans le cadre de cette thèse, nous avons opté pour la seconde approche car souvent, nous disposons d'une grande quantité de données non étiquetées, comme c'est le cas en segmentation d'images, et nous désirons améliorer leur classification par l'intégration d'une petite quantité

d'information a priori. Les méthodes de classification semi-supervisée de cette deuxième approche peuvent être divisées en deux catégories: *méthodes basées sur les données étiquetées* (*seeds*) et *méthodes basées sur les contraintes*.

1.5.3.1 Méthodes basées sur les données étiquetées

Ces méthodes intègrent l'information a priori disponible sous la forme de données étiquetées dans le processus de classification. Dans ce cas, l'ensemble de données X est composé d'un sous-ensemble de données étiquetées X^P de cardinal $|X^P|$ et d'un sous-ensemble de données non étiquetées X^U de cardinal $|X^U|$ ($X = \{X^P \cup X^U\}$ et $|X^P| + |X^U| = n$). L'objectif de ce type de méthodes est d'utiliser une méthode de classification non supervisée sur l'ensemble de données X^U tout en exploitant l'ensemble de données X^P pour améliorer les résultats de classification. La méthode de classification semi-supervisée la plus connue dans cette catégorie est seeded k-means [132]. Cette méthode, introduite par Basu et al., exploite la méthode k-means non supervisée et utilise les données étiquetées pour initialiser les centres de classes. Les étapes d'affectation et de mise à jour des centroïdes sont effectuées comme dans la version classique de k-means. Des variantes de cette méthode telles que farthest seeded k-means [133] et splitting seeded k-means [133] sont couramment citées dans la littérature.

1.5.3.2 Méthodes basées sur les contraintes

Ces méthodes intègrent l'information a priori disponible sous la forme de contraintes must-link et cannot-link dans le processus de classification. Ces contraintes ont été introduites initialement par Wagstaff et Cardie [116]. Une contrainte must-link spécifie que deux données sont similaires et doivent être classées dans la même classe, alors qu'une contrainte cannot-link spécifie que deux données sont différentes et doivent être classées dans deux classes différentes. Les contraintes sont simples à obtenir auprès de l'expert [116]. De plus, elles sont considérées comme étant plus générales car elles peuvent être extraites à partir d'autres types de connaissances, comme par exemple les données étiquetées. Les contraintes disponibles sont rangées en deux sous-ensembles:

- Un sous-ensemble de contraintes must-link M de cardinal $|M|$:
  $M = \{(x_i,x_j) \in X^2$, tel que x_i et x_j doivent être classées ensemble$\}$,

- Un sous-ensemble de contraintes cannot-link C de cardinal $|C|$:
  $C = \{(x_i,x_j) \in X^2$, tel que x_i et x_j ne doivent pas être classées ensemble$\}$.

Plusieurs méthodes intégrant les contraintes dans la classification non supervisée ont été proposées depuis les années 2000. Parmi elles, on peut citer: la classification spectrale sous contraintes [134, 135], COP k-means [136], PC k-means [137], et HAC sous contraintes (CHAC) [138]. Le chapitre 2 sera entièrement consacré à la classification spectrale sous contraintes. Nous décrivons brièvement dans ce qui suit les trois autres méthodes, à savoir COP k-means, PC k-means, et CHAC.

Méthode COP k-means Cette méthode, introduite par Wagstaff et al. en 2001 [136], intègre les contraintes dans la méthode k-means non supervisée lors de l'étape d'affectation des données aux centroïdes les plus proches. Elle affecte la donnée x_i au centroïde le plus proche qui satisfait les contraintes must-link et cannot-link. COP k-means cherche d'abord le centroïde le plus proche. Ensuite, si les contraintes ne sont pas violées, la donnée est assignée à la classe de ce centroïde, sinon elle cherche un autre centroïde qui satisfait les contraintes. Si tous les centroïdes violent les contraintes, l'algorithme s'arrête sur un échec. Les principales étapes de COP k-means sont décrites dans l'algorithme 1.6.

Algorithme 1.6: COP k-means.

Entrée: Ensemble de données $X = \{x_1, x_2, \ldots, x_n\}$, nombre de classes k, ensembles de contraintes must-link M et cannot-link C.

1. Initialiser aléatoirement k centroïdes $\bar{x}_{\omega^1}, \ldots, \bar{x}_{\omega^k}$.
2. Assigner à chaque donnée $x_i \in X$ la classe du centroïde $\bar{x}_{\omega^l}$ le plus proche tout en assurant qu'aucune contrainte de M et C ne soit violée. S'il n'existe pas de classe qui respecte les contraintes alors la sortie est un ensemble vide.
3. Recalculer les centroïdes par la moyenne des données contenues dans chaque classe.
4. Répéter les étapes 2-3 jusqu'à ce que les données ne changent pas de classes.

Sortie: La classe de chaque donnée x_i de l'ensemble X.

Méthode PC k-means Cette méthode, proposée par Basu et al. en 2004 [137], s'articule aussi autour de la méthode k-means. Elle tolère la violation des contraintes à la fin du processus de classification, contrairement à COP k-means où toutes les contraintes doivent être satisfaites. La classification par PC k-means est réalisée par la minimisation de la fonction objective $J_{PCk-\text{means}}$ définie par:

$$J_{PCk-\text{means}} = \sum_{x_i \in X} \delta^2(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_{\omega(x_i)}) + \sum_{(x_i,x_j)\in M} w_{ij} \mathbb{1}[\omega(x_i) \neq \omega(x_j)] + \sum_{(x_i,x_j)\in C} \bar{w}_{ij} \mathbb{1}[\omega(x_i) = \omega(x_j)] \tag{1.7}$$

où $\omega(x_j)$ est la classe estimée de x_i, $\bar{x}_{\omega(x_i)}$ est le centroïde de la classe de x_i. w_{ij} et $\bar{w}_{ij}$ sont des poids associés respectivement à une contrainte must-link et à une contrainte cannot-link, et $\mathbb{1}$[condition] est une fonction indicatrice égale à 1 si la condition est vérifiée, 0 sinon.

Les principales étapes de la méthode PC k-means sont décrites dans l'algorithme 1.7.

Algorithme 1.7: PC k-means.

Entrée: Ensemble de données $X = \{x_1, x_2, \ldots, x_n\}$, nombre de classes k, ensembles de contraintes must-link M et cannot-link C.

1. Initialiser k centroïdes $\bar{x}_{\omega^1}, \ldots, \bar{x}_{\omega^k}$.
2. Assigner à chaque donnée $x_i \in X$ la classe du centroïde qui minimise $J_{PCk\text{–means}}$.
3. Recalculer les centroïdes par la moyenne des données contenues dans chaque classe.
4. Répéter les étapes 2-3 jusqu'à ce que les données ne changent pas de classes.

Sortie: La classe de chaque donnée x_i de l'ensemble X.

Méthode CHAC Cette méthode, développée par Davidson et Ravi en 2005 [138], intègre les contraintes dans la méthode de classification non supervisée hiérarchique ascendante HAC. Elle commence par générer des classes de données à partir des contraintes must-link en utilisant la méthode de transitivité des must-link développée dans [139] et qu'on retrouve dans la section 2.5.1. Elle construit par la suite la hiérarchie des classes tout en empêchant la fusion de deux classes ayant une contrainte cannot-link entres elles. Les principales étapes de CHAC sont décrites dans l'algorithme 1.8.

Algorithme 1.8: CHAC.

Entrée: Ensemble de données $X = \{x_1, x_2, \ldots, x_n\}$, nombre de classes k, ensembles de contraintes must-link M et cannot-link C.

1. Appliquer la transitive des contraintes must-link et obtenir r classes $P_1, P_2, \ldots, P_r$.
2. Vérifier dans $P_1, P_2, \ldots, P_r$ qu'aucune contrainte cannot-link soit violée, si "oui" alors "pas de solution possible".
3. Construire une partition pour la classification composée des r classes $\bigcup_{j=1}^{r} P_j$ et de chacune des données de l'ensemble X restantes non utilisées dans $\bigcup_{j=1}^{r} P_j$.
4. **Tans que** il existe une paire de classes qui ne viole pas les contraintes C **Faire**
   - Fusionner les deux classes les plus similaires et mettre à jour le dendrogramme.
5. Sélectionner un niveau de coupe du dendrogramme pour obtenir les k classes.

Sortie: La classe de chaque donnée x_i de l'ensemble X.

1.6 Conclusion

Dans ce chapitre, nous avons passé en revue les différentes méthodes de segmentation d'images couleur texturées. Ces méthodes suivent généralement le même schéma composé de deux étapes: la caractérisation des pixels de l'image par un ensemble d'attributs de texture couleur et la répartition des pixels de l'image en plusieurs régions en se basant sur leurs attributs. Nous avons classifié ces méthodes de trois manières différentes: selon la stratégie employée pour combiner les attributs de texture couleur (séquentielle, parallèle, et conjointe), selon la technique utilisée pour détecter les régions (contours actifs, croissance de régions, division-fusion de régions, coupe de graphe, classification des pixels, apprentissage profond), et selon le contexte d'apprentissage, défini par l'information a priori apportée par l'utilisateur (non supervisé, supervisé, et semi-supervisé) et dans lequel opère la segmentation. Par la suite, nous nous sommes concentré sur les méthodes de classification automatique car celles-ci peuvent être exploitées non seulement en segmentation d'images, mais aussi pour la classification de tout type de données et pour la sélection d'attributs, et ce dans les trois contextes d'apprentissage. Nous avons décrit quelques méthodes de classification, en mettant l'accent sur les méthodes non supervisées et semi-supervisées qui utilisent des contraintes de type must-link et cannot-link comme information a priori.

Parmi toutes ces méthodes, notre attention s'est portée sur les méthodes de classification spectrale, pour plusieurs raisons que nous expliciterons dans le prochain chapitre.

Chapitre 2

Classification spectrale semi-supervisée sous contraintes

2.1 Introduction

Parmi les méthodes de classification non supervisée évoquées dans le chapitre précédent, la méthode de classification spectrale a attiré particulièrement notre attention, et ce pour plusieurs raisons. Tout d'abord, la classification spectrale est basée sur la théorie des graphes, qui est souvent utilisée pour résoudre les problèmes de classification de données et de sélection d'attributs, constituant ainsi un autre sujet d'intérêt dans cette thèse. Ensuite, elle permet de détecter des classes de formes complexes et non linéairement séparables. Et surtout, elle peut être facilement adaptée au contexte semi-supervisé en intégrant des informations a priori telles que les contraintes de type must-link et cannot-link.

Ce chapitre est dédié à la classification spectrale semi-supervisée sous contraintes. Tout d'abord, nous montrons comment représenter les données par un graphe de similarité, étant donné que ce dernier constitue l'outil de base de la sélection d'attributs et de la classification spectrale. Ensuite, nous décrivons les principales étapes de la classification spectrale et comparons ses performances à celles de plusieurs méthodes classiques de classification non supervisée sur des bases de données synthétiques et réelles. Par la suite, nous introduisons la notion de semi-supervision avec contraintes et les différentes manières de les intégrer dans le processus de classification spectrale non supervisée. Enfin, nous validons la classification spectrale semi-supervisée sous contraintes en comparant ses performances à celles des autres méthodes de classification semi-supervisée sous contraintes.

2.2 Représentation des données par des graphes de similarité

Un graphe est un modèle de représentation des données largement utilisé dans plusieurs domaines [140]. Il permet en outre d'exprimer les relations de voisinage entre les données et de révéler les dépendances entre elles [141]. Cette représentation de données par graphes permet d'aborder la sélection d'attributs et la classification spectrale sous un même formalisme.

2.2.1 Notations et définitions

Un graphe $G = (V,E)$ est défini par un ensemble non vide de nœuds (sommets) $V = \{v_1,v_2,\ldots,v_n\}$ où chaque nœud représente une donnée et d'un ensemble E d'arêtes (liens) e_{ij} qui relient les nœuds v_i et v_j ($E \subset V \times V$). Notons que:

- Le nombre de nœuds $n = |V|$ dans un graphe $G = (V,E)$ est appelé *ordre du graphe* G et le nombre d'arêtes $|E|$ est appelé *taille du graphe* G.
- Un graphe est *orienté* (*digraphe*) si toutes ses arêtes sont orientées, et *non orienté* dans le cas contraire. Une arête e_{ij} est orientée si $e_{ij} \neq e_{ji}$ et non orientée sinon.
- Un graphe est *pondéré* $G = (V,E,\boldsymbol{W})$ si chaque arête e_{ij} est caractérisée par un poids $w_{ij} \in [0,1]$, qui mesure la similarité entre deux données x_i et x_j correspondantes aux nœuds v_i et v_j. $\boldsymbol{W}$ est une matrice de similarité regroupant toutes les similarités entre les paires de nœuds et dont le calcul sera précisé dans la section 2.2.3.2.

Dans ce travail de thèse, nous considérons un graphe de similarité non orienté et pondéré $G = (V,E,\boldsymbol{W})$, composé de $n = |V|$ nœuds et $|E|$ arêtes pondérées par des poids w_{ij} construit à partir d'un ensemble de n données $X = \{x_1,x_2,\ldots,x_n\}$. L'ensemble X peut être représenté par une matrice $\boldsymbol{X} = [\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_n] \in \mathbb{R}^{n\times d}$, où chaque donnée $\boldsymbol{x}_i = [x_{i1},x_{i2},\ldots,x_{id}]$ est caractérisée par un ensemble de d attributs.

2.2.2 Types de graphes de similarité

Les graphes de similarité se distinguent principalement par le nombre d'arrêtes reliant les nœuds entre eux [125]:

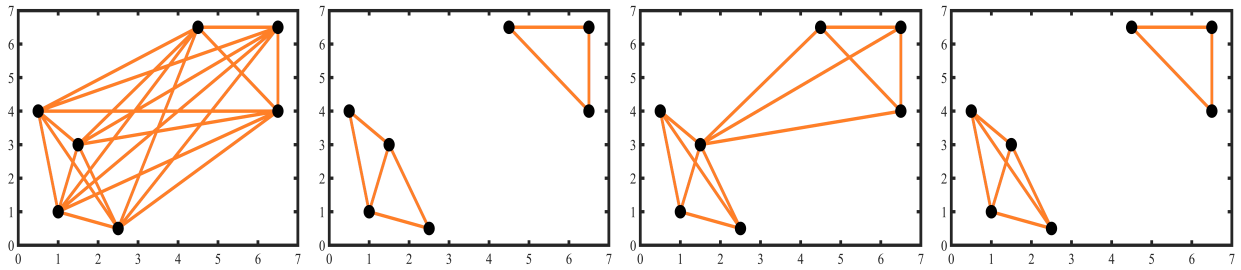
Graphe totalement connecté: Tous les nœuds du graphe sont reliés entre eux par des arêtes pondérées par des valeurs de similarités. Le nombre d'arêtes dans un graphe G de n nœuds

est de $(n(n-1))/2$. La figure 2.1(a) montre un exemple de graphe de similarité totalement connecté.

Graphe de voisinage ϵ: C'est un graphe partiellement connecté ou seuls les nœuds v_i et v_j dont la distance $\delta_{ij} \leq \epsilon$ (ou la similarité $w_{ij} \geq \epsilon$) sont reliés entre eux. ϵ est un seuil fixé par l'utilisateur (Figure 2.1(b)).

Graphe des K plus proches voisins: C'est un graphe partiellement connecté dans lequel un nœud v_i est relié au nœud v_j si v_j fait partie des K plus proches voisins de v_i (K est un nombre entier fixé par l'utilisateur). Cependant, le graphe résultant peut être orienté car un nœud v_i peut appartenir aux K plus proches voisins de v_j, sans que v_j appartienne aux K plus proches voisins de v_i. Pour obtenir un graphe non orienté avec la règle du plus proches voisins, on procède par l'une des deux façons suivantes:

- deux nœuds v_i et v_j sont reliés si $v_i \in KNN(v_j)$ ou si $v_j \in KNN(v_i)$. Ce graphe est appelé *graphe des K plus proches voisins* (Figure 2.1(c)).
- deux nœuds v_i et v_j sont reliés si $v_i \in KNN(v_j)$ et si $v_j \in KNN(v_i)$. Ce graphe est appelé *graphe des K plus proches voisins mutuels* (Figure 2.1(d)).



(a) (b) (c) (d)

Figure 2.1: Graphes de similarité. (a) Graphe totalement connecté, (b) graphe de voisinage ϵ, (c) graphe des K plus proche voisins, et (d) graphe des K plus proches voisins mutuels ($\epsilon = 4$ et $K = 3$).

Les paramètres ϵ et K influent sur les résultats de classification ou de segmentation d'images. En effet, plus les valeurs de ces paramètres sont élevées, plus le nombre d'arêtes dans les graphes est important.

2.2.3 Matrices associées aux graphes

Un graphe peut être représenté par divers matrices telles que *la matrice des distances*, *la matrice de similarité*, *la matrice de degrés*, et *la matrice laplacienne*.

2.2.3.1 Matrice des distances

La distance δ entre deux données x_i et x_j associées à deux nœuds v_i et v_j d'un graphe G, peut être calculée en utilisant une métrique de distance qui satisfait les propriétés suivantes:

- Symétrie $\delta(\boldsymbol{x}_i,\boldsymbol{x}_j) = \delta(\boldsymbol{x}_j,\boldsymbol{x}_i)$,
- Non négativité $\delta(\boldsymbol{x}_i,\boldsymbol{x}_j) \geq 0$,
- Séparation $\delta(\boldsymbol{x}_i,\boldsymbol{x}_j) = 0$ si et seulement si $\boldsymbol{x}_i = \boldsymbol{x}_j$,
- Minimalité $\delta(\boldsymbol{x}_i,\boldsymbol{x}_i) = 0$,
- Inégalité triangulaire $\delta(\boldsymbol{x}_i,\boldsymbol{x}_j) \leq \delta(\boldsymbol{x}_i,\boldsymbol{x}_k) + \delta(\boldsymbol{x}_k,\boldsymbol{x}_j)$.

La distance entre deux données proches dans l'espace des attributs tend vers 0, et la distance entre deux données éloignées dans l'espace des attributs tend vers $+\infty$. Le tableau 2.1 présente les métriques de distance les plus connues.

Tableau 2.1: Métriques de distance.

Métrique de distance	**Formulation**	
Distance de Minkowski	$\delta(\boldsymbol{x}_i,\boldsymbol{x}_j) = \Big(\sum_{r=1}^{d} \lvert x_{ir} - x_{jr}\rvert^2\Big)^{\frac{1}{p}}$	(2.1)
Distance de Manhattan	$\delta(\boldsymbol{x}_i,\boldsymbol{x}_j) = \sum_{r=1}^{d} \lvert x_{ir} - x_{jr}\rvert$	(2.2)
Distance Euclidienne	$\delta(\boldsymbol{x}_i,\boldsymbol{x}_j) = \Big(\sum_{r=1}^{d} \lvert x_{ir} - x_{jr}\rvert^2\Big)^{\frac{1}{2}}$	(2.3)
Distance de Tchebychev	$\delta(\boldsymbol{x}_i,\boldsymbol{x}_j) = \max_{1<r<d}\Big(\lvert x_{ir} - x_{jr}\rvert\Big)$	(2.4)
Distance Cosinus	$\delta(\boldsymbol{x}_i,\boldsymbol{x}_j) = 1 - \frac{\sum_{r=1}^{d} \lvert x_{ir}x_{jr}\rvert}{\sqrt{\sum_{r=1}^{d} \lvert x_{ir}\rvert^2}\sqrt{\sum_{r=1}^{d} \lvert x_{jr}\rvert^2}}$	(2.5)

L'ensemble des distances δ entre les nœuds du graphe peuvent être regroupées dans la matrice des distances, notée $\boldsymbol{\Delta}$, de dimension $(n \times n)$:

$$\boldsymbol{\Delta} = \begin{bmatrix} 0 & \dots & \delta(\boldsymbol{x}_1,\boldsymbol{x}_j) & \dots & \delta(\boldsymbol{x}_1,\boldsymbol{x}_n) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \delta(\boldsymbol{x}_i,\boldsymbol{x}_1) & \dots & 0 & \dots & \delta(\boldsymbol{x}_i,\boldsymbol{x}_n) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \delta(\boldsymbol{x}_n,\boldsymbol{x}_1) & \dots & \delta(\boldsymbol{x}_n,\boldsymbol{x}_i) & \dots & 0 \end{bmatrix}$$

2.2.3.2 Matrice de similarité

La similarité w_{ij} entre deux données x_i et x_j associées à deux nœuds v_i et v_j d'un graphe G, est mesurée à l'aide d'une métrique de similarité qui satisfait les propriétés suivantes:

- Symétrie $w_{ij} = w_{ji}$,
- Normalisation $w_{ij} \in [0,1]$,
- $w_{ij} = 1$ si et seulement si $\boldsymbol{x}_i = \boldsymbol{x}_j$,
- Maximalité $w_{ii} \geq w_{ij}$.

La similarité est liée à la notion de distance. En effet, deux données séparées par une grande distance correspondent à deux données non similaires, alors que deux données proches correspondent à deux données similaires. Le tableau 2.2 regroupe les métriques de similarité les plus connues.

Tableau 2.2: Métriques de similarité.

Métrique de similarité	**Formulation**	
Inverse de la distance Euclidienne	$w(\boldsymbol{x}_i,\boldsymbol{x}_j) = \dfrac{1}{1+\delta^2(\boldsymbol{x}_i,\boldsymbol{x}_j)}$	(2.6)
Similarité Euclidienne ajustée	$w(\boldsymbol{x}_i,\boldsymbol{x}_j) = \dfrac{1}{1+\frac{\delta^2(\boldsymbol{x}_i,\boldsymbol{x}_j)}{\sigma^2}}$	(2.7)
Similarité Gaussienne	$w(\boldsymbol{x}_i,\boldsymbol{x}_j) = \texttt{exp}\left(-\dfrac{\delta^2(\boldsymbol{x}_i,\boldsymbol{x}_j)}{2\sigma^2}\right)$	(2.8)
Similarité Cosinus	$w(\boldsymbol{x}_i,\boldsymbol{x}_j) = \dfrac{\sum_{r=1}^{d} \lvert x_{ir} \times x_{jr}\rvert}{\sqrt{\sum_{r=1}^{d} \lvert x_{ir}\rvert^2} \times \sqrt{\sum_{r=1}^{d} \lvert x_{jr}\rvert^2}}$	(2.9)

La matrice de similarité $\boldsymbol{W}$, de dimension $(n \times n)$, entre les n nœuds du graphe est une matrice symétrique ($\boldsymbol{W} = \boldsymbol{W}^T$) et positive ($\boldsymbol{W} \geq 0$), qui regroupe l'ensemble des similarités w_{ij} entre toutes les paires de l'ensemble de données X:

$$\boldsymbol{W} = \begin{bmatrix} 1 & \dots & w_{1i} & \dots & w_{1n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{i1} & \dots & 1 & \dots & w_{in} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{n1} & \dots & w_{ni} & \dots & 1 \end{bmatrix}$$

2.2.3.3 Matrice des degrés

Le degré d_i d'un nœud v_i est égale à la somme des éléments de la i-ème ligne de la matrice de similarité $\boldsymbol{W}$ tel que:

$$d_i = \sum_{j=1}^{n} w_{ij} \tag{2.10}$$

Dans un graphe partiellement connecté, le degré d_i exprime une mesure de densité au voisinage de la donnée $\boldsymbol{x}_i$ représentée par le nœud v_i dans le graphe G.

Les degrés de tous les nœuds d'un graphe sont regroupés dans une matrice diagonale de dimension $(n \times n)$, dénommée, matrice des degrés, et notée $\boldsymbol{D}$:

$$\boldsymbol{D} = \begin{bmatrix} d_{11} & \dots & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \dots & d_{ii} & \dots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & \dots & d_{nn} \end{bmatrix}$$

2.2.3.4 Matrice laplacienne

La matrice laplacienne peut être assimilée à une matrice de similarité, elle découle d'une combinaison des matrices de similarité $\boldsymbol{W}$ et de degrés $\boldsymbol{D}$. On distingue deux types de matrices laplaciennes: *non normalisées* et *normalisées*.

La matrice laplacienne non normalisée $\boldsymbol{L}$ et la matrice laplacienne normalisée symétrique $\boldsymbol{L}_{Sym}$ sont couramment exploitées dans la littérature liée à la sélection d'attributs et à la classification spectrale des données [125, 142]. Elles sont définies comme suit:

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W} \tag{2.11}$$

et

$$\begin{aligned} \boldsymbol{L}_{Sym} &= \boldsymbol{D}^{-1/2}\boldsymbol{L}\boldsymbol{D}^{-1/2} \\ &= \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2} \end{aligned} \tag{2.12}$$

La normalisation par $\boldsymbol{D}^{-1/2}$ permet de rendre la représentation du graphe moins sensible à la taille et à la densité du graphe. D'autres matrices laplaciennes normalisées et non normalisées sont décrites dans le tableau 2.3.

Tableau 2.3: D'autres matrices laplaciennes normalisées et non normalisées.

Matrice laplacienne	**Formulation**	
Laplacienne non normalisée non signée [143]	$\boldsymbol{L}_{Ns} = \boldsymbol{D} + \boldsymbol{W}$	(2.13)
Laplacienne normalisée asymétrique [125]	$\boldsymbol{L}_{Asym} = \boldsymbol{D}^{-1}\boldsymbol{L} = \boldsymbol{I} - \boldsymbol{D}^{-1}\boldsymbol{W}$	(2.14)
Laplacienne normalisée par division [142]	$\boldsymbol{L}_{Nd} = \boldsymbol{D}^{-1}\boldsymbol{W}$	(2.15)
Laplacienne normalisée par division symétrique [77]	$\boldsymbol{L}_{Ng} = \boldsymbol{D}^{-1}\boldsymbol{W}\boldsymbol{D}^{-1}$	(2.16)
Laplacienne normalisée additive [134]	$\boldsymbol{L}_{Ad} = \dfrac{(\boldsymbol{W} + d_{max}\boldsymbol{I} - \boldsymbol{D})}{d_{max}}$ (avec $d_{max} = \max(\boldsymbol{D})$)	(2.17)

2.2.4 Propriétés des matrices laplaciennes

Les matrices laplaciennes jouissent de certaines propriétés qui sont exploitées à la fois en sélection d'attributs et en classification spectrale, et que nous jugeons utile de rappeler.

Matrice laplacienne non normalisée $\boldsymbol{L}$. La matrice $\boldsymbol{L}$ de l'équation (2.11) satisfait les propriétés suivantes [125]:

Propriété 1. Pour chaque vecteur $\boldsymbol{f} \in \mathbb{R}^n$, nous avons:

$$\boldsymbol{f}^T\boldsymbol{L}\boldsymbol{f} = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(f_i - f_j)^2 \tag{2.18}$$

Cette propriété est déduite à partir des matrices $\boldsymbol{L}$ et $\boldsymbol{D}$, telle que $\forall \boldsymbol{f} \in \mathbb{R}^n$:

$$\begin{aligned}
\boldsymbol{f}^T\boldsymbol{L}\boldsymbol{f} &= \boldsymbol{f}^T(\boldsymbol{D} - \boldsymbol{W})\boldsymbol{f} = \boldsymbol{f}^T\boldsymbol{D}\boldsymbol{f} - \boldsymbol{f}^T\boldsymbol{W}\boldsymbol{f} \\
&= \sum_{i=1}^{n} f_i^2 d_i - \sum_{i=1}^{n}\sum_{j=1}^{n} f_i w_{ij} f_j \\
&= \frac{1}{2}\left(\sum_{i=1}^{n} f_i^2 d_i - 2\sum_{i=1}^{n}\sum_{j=1}^{n} f_i w_{ij} f_j + \sum_{i=1}^{n} f_i^2 d_i\right) \\
&= \frac{1}{2}\left(\sum_{i=1}^{n} f_i^2 d_i - 2\sum_{i=1}^{n}\sum_{j=1}^{n} f_i w_{ij} f_j + \sum_{j=1}^{n} f_j^2 d_j\right) \\
&= \frac{1}{2}\left(\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} f_i^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{n} f_i w_{ij} f_j + \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} f_j^2\right) \\
&= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(f_i - f_j)^2
\end{aligned} \tag{2.19}$$

Propriété 2. La matrice laplacienne $\boldsymbol{L}$ *est une matrice symétrique* ($\boldsymbol{L} = \boldsymbol{L}^T$) *et semi-définie positive* ($\boldsymbol{L} \geq 0$).

À partir de l'équation (2.18), on peut déduire que $\boldsymbol{f}^T\boldsymbol{L}\boldsymbol{f} \geq 0$ (car $w_{ij} \geq 0$ et $(f_i - f_j)^2 \geq 0$), et $\boldsymbol{L}$ est semi-définie positive. Et comme, $\boldsymbol{D}$ et $\boldsymbol{W}$ sont symétriques, alors $\boldsymbol{L}$ l'est aussi.

Propriété 3. La matrice laplacienne $\boldsymbol{L}$ *possède n valeurs propres réelles positives, et* 0 *est la plus petite valeur propre, notées,* $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. *Le vecteur propre associé à la valeur propre* $\lambda_1 = 0$ *est un vecteur constant* $\mathbf{1}$ ($\mathbf{1} = [1,1,\ldots,1]^T$).

Si λ est une valeur propre de $\boldsymbol{L}$ et $\boldsymbol{u}$ son vecteur propre associé, on a alors:

$$\boldsymbol{L}\boldsymbol{u} = \lambda\boldsymbol{u} \tag{2.20}$$

Lorsqu'on multiplie l'équation (2.20) par $\boldsymbol{u}^T$, on obtient $\boldsymbol{u}^T\boldsymbol{L}\boldsymbol{u} = \boldsymbol{u}^T\lambda\boldsymbol{u} = \lambda \parallel \boldsymbol{u} \parallel^2 \geq 0$. Compte tenu de l'équation (2.18), nous avons $\boldsymbol{u}^T\boldsymbol{L}\boldsymbol{u} = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(u_i - u_j)^2 \geq 0$, ce qui induit que les valeurs propres de $\boldsymbol{L}$ sont réelles positives (Propriété 2). Concernant le vecteur propre $\boldsymbol{u} = \mathbf{1}$, on a $\boldsymbol{L}\mathbf{1} = (\boldsymbol{D} - \boldsymbol{W})\mathbf{1} = \boldsymbol{D}\mathbf{1} - \boldsymbol{W}\mathbf{1} = \boldsymbol{D} - \boldsymbol{D} = \mathbf{0}$ (équation (2.20)), ce qui indique que sa valeur propre est $\lambda = 0$ ($\boldsymbol{D}\mathbf{1} = \boldsymbol{D}$ et $\boldsymbol{W}\mathbf{1} = \boldsymbol{D}$).

Matrice laplacienne normalisée symétrique $\boldsymbol{L}_{Sym}$. La matrice $\boldsymbol{L}_{Sym}$ de l'équation (2.12) satisfait les propriétés suivantes [125]:

Propriété 1. Pour chaque vecteur $\boldsymbol{f} \in \mathbb{R}^n$, *nous avons:*

$$\boldsymbol{f}^T\boldsymbol{L}_{Sym}\boldsymbol{f} = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}\left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}}\right)^2 \tag{2.21}$$

Preuve:

$$\begin{aligned}
\boldsymbol{f}^T\boldsymbol{L}_{Sym}\boldsymbol{f} &= \boldsymbol{f}^T(\boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{W}\boldsymbol{D}^{-1/2})\boldsymbol{f} \\
&= \sum_{i=1}^{n} f_i^2 - \sum_{i=1}^{n}\sum_{j=1}^{n} f_i \frac{w_{ij}}{\sqrt{d_i}\sqrt{d_j}} f_j \\
&= \frac{1}{2}\left(\sum_{i=1}^{n} f_i^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{n} f_i \frac{w_{ij}}{\sqrt{d_i}\sqrt{d_j}} f_j + \sum_{i=1}^{n} f_i^2\right) \\
&= \frac{1}{2}\left(\sum_{i=1}^{n} f_i^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{n} f_i \frac{w_{ij}}{\sqrt{d_i}\sqrt{d_j}} f_j + \sum_{j=1}^{n} f_j^2\right) \\
&= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}\left(\frac{f_i^2}{d_i} - 2f_i\frac{1}{\sqrt{d_i}\sqrt{d_j}} f_j + \frac{f_j^2}{d_j}\right) \\
&= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}\left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}}\right)^2
\end{aligned} \tag{2.22}$$

Propriété 2. La matrice laplacienne normalisée $\boldsymbol{L}_{Sym}$ est symétrique et semi-définie positive.

En effet, à partir de l'équation (2.21), on peut voir que $\boldsymbol{f}^T\boldsymbol{L}_{Sym}\boldsymbol{f} \geq 0$ $\Big($car $w_{ij} \geq 0$ et $\Big(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}}\Big)^2 \geq 0\Big)$. Par conséquent, $\boldsymbol{L}_{Sym}$ est semi-définie positive.

Propriété 3. La matrice laplacienne normalisée $\boldsymbol{L}_{Sym}$ possède n valeurs propres réelles positives, et 0 est la plus petite valeur propre, notées, $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. Le vecteur propre associé à la valeur propre $\lambda_1 = 0$ est un vecteur constant $\boldsymbol{D}^{1/2}\mathbf{1}$.

De la même façon que pour la matrice laplacienne non normalisée $\boldsymbol{L}$, on a $\boldsymbol{u}^T\boldsymbol{L}_{Sym}\boldsymbol{u} = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}\Big(\frac{u_i}{\sqrt{d_i}} - \frac{u_j}{\sqrt{d_j}}\Big)^2 \geq 0$. On peut alors écrire $\boldsymbol{u}^T\boldsymbol{L}_{Sym}\boldsymbol{u} = \boldsymbol{u}^T\lambda\boldsymbol{u} = \lambda \parallel \boldsymbol{u} \parallel^2 \geq 0$, et déduire que les valeurs propres de $\boldsymbol{L}_{Sym}$ sont positives. Pour le vecteur propre $\boldsymbol{u} = \boldsymbol{D}^{1/2}\mathbf{1}$, on a $\boldsymbol{L}_{Sym}(\boldsymbol{D}^{1/2}\mathbf{1}) = \boldsymbol{D}^{1/2}\mathbf{1} - \boldsymbol{D}^{-1/2}\boldsymbol{W}\mathbf{1} = \boldsymbol{D}^{1/2}(\mathbf{1} - \mathbf{1}) = \mathbf{0}$, ce qui implique que sa valeur propre est nulle ($\lambda = 0$).

2.2.5 Partitionnement d'un graphe de similarité

Le partitionnement d'un graphe est une opération très utilisée en segmentation d'images [56, 79] et en classification des données [77].

Soit $G = (V,E,\boldsymbol{W})$ un graphe de similarité pondéré et non orienté. Le partitionnement de G consiste à former k sous-groupes disjoints de nœuds $c_1,c_2,\ldots,c_k$. Cette opération peut être obtenue en minimisant la fonction de coupe minimale (MinCut) [76], définie comme suit:

$$\text{MinCut}(c_1,\ldots,c_k) = \sum_{i=1}^{k} \text{Cut}(c_i,\bar{c_i}) \tag{2.23}$$

$\bar{c_i}$ est l'ensemble des nœuds autre que ceux de c_i, et $\text{Cut}(c_i,\bar{c_i})$ la somme des similarités entre les classes c_i et $\bar{c_i}$. Pour deux classes c_1 et c_2, la valeur de $\text{Cut}(c_1,c_2)$ est définie par:

$$\text{Cut}(c_1,c_2) = \sum_{\{v_i \in c_1\},\{v_j \in c_2\}} w_{ij} \tag{2.24}$$

La minimisation de la fonction MinCut favorise la partition des nœuds isolés. Une solution de ce problème est de normaliser la valeur de la coupe de façon à obtenir des partitions avec un nombre assez important de nœuds. Le tableau 2.4 regroupe les fonctions de coupes les plus populaires [125]. Dans ce tableau, $|c_i|$ est le nombre de nœuds du sous-graphe c_i, et $vol(c_i)$ la somme des poids de c_i.

Tableau 2.4: D'autres fonctions de coupes de graphes.

Fonction de coupe	**Formulation**	
Coupe ratio	$\text{RatioCut}(c_1,\ldots,c_k) = \sum_{i=1}^{k} \frac{\text{Cut}(c_i,\bar{c_i})}{\lvert c_i\rvert}$	(2.25)
Coupe normalisée	$\text{NCut}(c_1,\ldots,c_k) = \sum_{i=1}^{k} \frac{\text{Cut}(c_i,\bar{c_i})}{vol(c_i)}$	(2.26)
Coupe min-max	$\text{MinMaxCut}(c_1,\ldots,c_k) = \sum_{i=1}^{k} \frac{\text{Cut}(c_i,\bar{c_i})}{\text{Cut}(c_i,c_i)}$	(2.27)

2.3 Classification spectrale

La classification spectrale est une méthode de classification non supervisée formulée comme un problème de coupe de graphe et qui peut être résolu par l'extraction du spectre (valeurs et vecteurs propres) de la matrice laplacienne. Cette dernière est calculée à partir du graphe de similarité représentant l'ensemble des données à classer. Le processus de classification spectrale se décompose en trois principales étapes [125, 144]:

1. *Construction du graphe de similarité:*
   - Construction d'un graphe de similarité G.
   - Calcul de la matrice de similarité $\boldsymbol{W}$ correspondante au graphe G.
2. *Construction de l'espace spectral:*
   - Calcul de la matrice laplacienne.
   - Extraction du spectre de la matrice laplacienne.
   - Projection des données initiales dans l'espace défini par les vecteurs propres dominants.
3. *Partitionnement des données dans l'espace spectral:*
   - Classification des données projetées en utilisant une méthode classique de classification.

Concrètement, la classification spectrale consiste à augmenter la dimension des données en calculant les vecteurs propres de la matrice laplacienne afin de favoriser la séparabilité des classes. Les données sont ensuite projetées dans un nouvel espace, appelé *espace spectral*, de dimension réduite, engendré par une partie des vecteurs propres. Dans cet espace, les classes sont supposées être bien séparées et facilement détectables par une simple méthode de classification non supervisée comme k-means.

Afin de bien comprendre le principe de la classification spectrale, un exemple illustratif de ses principales étapes est présenté dans la figure 2.2. Dans cet exemple, nous avons généré un ensemble de données composé de trois classes. La matrice de similarité présente une structure bloc-diagonale car les données sont ordonnées successivement selon la classe. Nous pouvons constater que dans l'espace de projection, les données deviennent facilement séparables en trois classes.

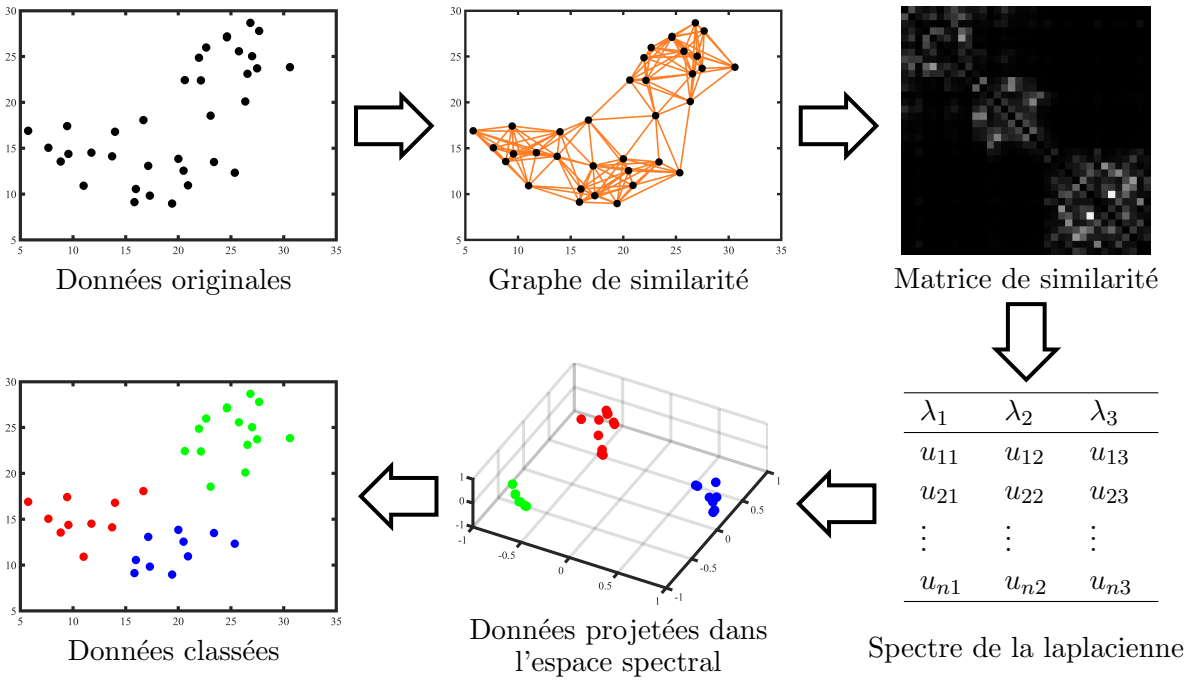


Figure 2.2: Illustration de la classification spectrale.

Les méthodes de la classification spectrale peuvent être classées en deux catégories, selon le nombre de vecteurs propres utilisés pour le partitionnement des données:

- La première catégorie est basée sur le partitionnement bipartite récursif qui utilise le vecteur propre associé à la seconde valeur propre dominante de la matrice laplacienne [79]. Parmi les méthodes de cette catégorie, on peut citer: la méthode de Shi et Malik [79] et la méthode de Perona et Freeman [145]. Elles sont néanmoins connues pour être très lentes.

- La deuxième catégorie est basée sur la projection des données originales dans un espace défini par un ensemble de vecteurs propres dominants extraits de la matrice laplacienne [79]. Une méthode classique de classification est ensuite appliquée sur les données projetées afin d'obtenir la partition finale. Parmi les méthodes de cette catégorie, on peut citer, la méthode de Von Luxburg [125], la méthode de Ng et al. [77], et la méthode de Rohe et al. [146].

Dans cette thèse, nous avons opté pour la méthode de classification spectrale normalisée (SC_{njw}) proposée par Ng et al. [77], et qui découle de la minimisation de la coupe normalisée (NCut) décrite dans l'équation (2.26) avec $\boldsymbol{L}_{Sym}$ comme matrice laplacienne (équation (2.12)). Cet algorithme de classification spectrale reste très populaire dans la littérature [147, 148]. Ses principales étapes sont résumées dans l'algorithme 2.1.

Algorithme 2.1: SC_{njw}.

Entrée: Ensemble des données $X = \{x_1, x_2, \ldots, x_n\}$, nombre de classes k.

1. Construire le graphe de similarité G.
2. Calculer la matrice de similarité $\boldsymbol{W} \in \mathbb{R}^{n \times n}$ (équation (2.8)).
3. Calculer la matrice laplacienne normalisée $\boldsymbol{L}_{Sym}$ (équation (2.12)) associée à $\boldsymbol{W}$.
4. Calculer le spectre de $\boldsymbol{L}_{Sym}$ (valeurs λ_j et vecteurs $\boldsymbol{u}_j$ propres, $j = 1, \ldots, n$).
5. Définir la matrice $\boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k] \in \mathbb{R}^{n \times k}$ à partir des k plus grands vecteurs propres correspondants aux k plus petites valeurs propres de $\boldsymbol{L}_{Sym}$.
6. Construire la matrice $\boldsymbol{T} \in \mathbb{R}^{n \times k}$ en normalisant les lignes de $\boldsymbol{U}$ comme suit:
$$t_{ij} = \frac{u_{ij}}{\sqrt{\sum_{j=1}^{k} u_{ij}^2}}$$
7. Définir le nouvel ensemble de données Z dans l'espace spectral où chaque nouvelle donnée $\boldsymbol{z}_i$ est définie par la i-ème ligne de $\boldsymbol{T}$.
8. Classer l'ensemble de données Z en k classes en utilisant la méthode des k-means.

Sortie: La classe de chaque donnée x_i de l'ensemble X.

2.4 Résultats de classification spectrale

Pour prouver l'efficacité de la méthode de classification spectrale, nous l'avons confrontée à plusieurs autres méthodes classiques de classification non supervisée.

La comparaison est conduite sur douze bases de données: six d'entre elles contiennent des données synthétiques, disponibles dans Tomas Barton Repository [149], et les six autres des données réelles, téléchargeables sur le site de l'UCI Machine Learning Repository [150]. Le tableau 2.5 présente les différentes caractéristiques de chacune des douze bases de données. Notons que, toutes ces bases sont caractérisées par des attributs numériques, normalisés dans le cas des données synthétiques. En revanche, nous avons normalisé chaque attribut des bases de données réelles entre 0 et 1 pour que l'échelle des valeurs soit la même.

Tableau 2.5: Description des bases de données utilisées dans l'expérimentation et les paramètres des méthodes de classification MS, DBSCAN, DDC, et SC_{njw}. n nombre de données, d nombre d'attributs, et k nombre de classes.

Bases de données		**Caractéristiques**			MS	DBSCAN		DDC		SC_{njw}
		n	d	k	h	ϵ	minP	K	λ	σ
Synthétiques	`Flame`	240	2	2	3.41	1.56	14	10	0.6	0.3
	`Spiral`	312	2	3	7.78	1.11	1	5	0.3	1
	`Jain`	373	2	2	9.63	2.02	14	10	0.3	1
	`DS4C2SC8`	485	2	8	0.05	0.02	18	10	0.6	0.01
	`DS850`	850	2	5	0.88	0.40	8	10	0.3	0.26
	`Chain Link`	1000	3	2	0.96	0.11	1	10	0.3	0.31
Réelles	`Iris`	150	4	3	0.30	0.13	9	10	0.6	0.1
	`Wisc`	699	9	2	1.19	0.48	15	20	0.3	0.13
	`Vowels`	1456	12	2	0.68	0.33	3	10	0.7	0.1
	`Vote`	435	16	2	2.02	1.08	23	15	0.4	0.09
	`German`	1000	24	2	1.88	1.83	1	20	0.3	0.23
	`Urban`	168	147	9	2.48	1.75	7	5	0.6	1

Les méthodes de classification non supervisée choisies pour la comparaison sont: k-means [54], fuzzy c-means (FCM) [151], mean-shift (MS) [123], les méthodes hiérarchiques ascendantes single-linkage (HAC_{SL}) [118] et Ward (HAC_{Ward}) [117], les méthodes basées sur la densité DBSCAN [120] et DDC [121], ainsi que la méthode de classification spectrale normalisée de Ng et al. (SC_{njw}) [77]. Les paramètres de ces méthodes sont: la largeur de bande (h) pour MS, le rayon maximum (ϵ) et le nombre minimum de points (minP) pour DBSCAN, le nombre de voisins (K) dans le graphe des KNN mutuels et le paramètre (λ) pour DDC, et le paramètre d'échelle (σ) pour SC_{njw}. Pour les autres méthodes, aucun paramètre n'est nécessaire. Ces paramètres sont fixés de manière empirique à des valeurs ayant donné les meilleurs résultats de classification. Ces valeurs sont indiquées dans le tableau 2.5. Pour évaluer les performances des méthodes de classification, le taux de classification (accuracy) est utilisé comme critère de comparaison.

Le tableau 2.6 présente les taux de classification obtenus par les huit méthodes de classification non supervisée (k-means, FCM, MS, HAC_{SL}, HAC_{Ward}, DBSCAN, DDC, et SC_{njw}) sur les douze bases de données. Les meilleurs taux de classification sont indiqués en gras. Globalement, nous constatons que la méthode de classification spectrale SC_{njw} surclasse toutes les autres méthodes dans la majorité des cas. Elle est suivie par la méthode DDC, qui fournit des résultats de classification satisfaisants sur les données synthétiques et moins bons sur les données réelles.

Tableau 2.6: Taux de classification obtenus par les différentes méthodes de classification.

	Bases de données	k-means	FCM	MS	HAC_{SL}	HAC_{Ward}	DBSCAN	DDC	SC_{njw}
Synthétiques	`Flame`	83.75	85.00	98.33	64.58	72.08	98.75	99.17	**100**
	`Spiral`	33.97	33.97	45.19	**100**	37.50	**100**	**100**	**100**
	`Jain`	78.28	77.48	68.10	80.97	86.06	**100**	**100**	**100**
	`DS4C2SC8`	84.54	90.10	67.63	19.38	91.13	66.39	**96.70**	96.50
	`DS850`	95.88	95.88	**99.88**	56.00	99.18	98.94	**99.88**	**99.88**
	`Chain Link`	65.30	64.80	81.10	**100**	76.50	**100**	**100**	**100**
Réelles	`Iris`	88.67	89.33	68.67	66.00	88.67	86.00	96.00	**96.67**
	`Wisc`	95.99	95.42	**97.28**	65.67	96.42	96.85	96.42	97.14
	`Vowels`	50.62	50.34	61.88	96.63	54.53	98.21	84.75	**98.28**
	`Vote`	86.67	86.21	61.61	61.61	86.21	80.92	86.90	**89.43**
	`German`	64.10	55.10	**70.10**	**70.10**	61.60	70.09	68.30	70.00
	`Urban`	68.45	48.21	26.19	20.24	65.48	47.02	60.12	**70.83**

Pour analyser visuellement ces résultats, nous affichons dans la figure 2.3 les données classées par les méthodes k-means, MS, HAC_{Ward}, DDC, et SC_{njw} sur les bases de données `Jain`, `Spiral`, et `Chain Link`. Ces trois bases de données ont des classes de formes arbitraires et non linéairement séparables. Seules les méthodes SC_{njw} et DDC ont correctement identifié toutes les classes de ces trois bases. Ces résultats confirment la supériorité des méthodes SC_{njw} et DDC par rapport aux méthodes k-means, MS, et HAC_{Ward} lorsque les classes ont des formes non globulaires et non linéairement séparables.

La figure 2.4 affiche les résultats de classification obtenus par les méthodes k-means, MS, HAC_{Ward}, DDC, et SC_{njw} sur les bases de données `Flame`, `DS850`, et `DS4C2SC8`, qui ont la particularité d'avoir des classes chevauchantes. Cette figure montre que les méthodes SC_{njw} et DDC identifient correctement les deux classes de la base `Flame`, ainsi que la majorité des classes des deux autres bases (`DS850` et `DS4C2SC8`). HAC_{Ward} donne de bons résultats pour la base `DS850`, mais pas pour les bases `Flame` et `DS4C2SC8`. k-means n'a pas pu aboutir à de bons résultats sur les trois bases `Flame`, `DS850`, et `DS4C2SC8`. MS donne de bons résultats sur les bases `Flame` et `DS850`, mais de mauvais résultats sur la base `DS4C2SC8` dont le degré de chevauchement entre les classes est plus élevé. Ces résultats montrent clairement que les méthodes SC_{njw} et DDC sont plus performantes que les méthodes k-means, MS, et HAC_{Ward} lorsque les classes se chevauchent.

Finalement, ces résultats corroborent ceux de la littérature [125], faisant état de la capacité de la classification spectrale à détecter des classes de formes non globulaires et non linéairement séparables, et de sa supériorité sur les méthodes de classification classiques.

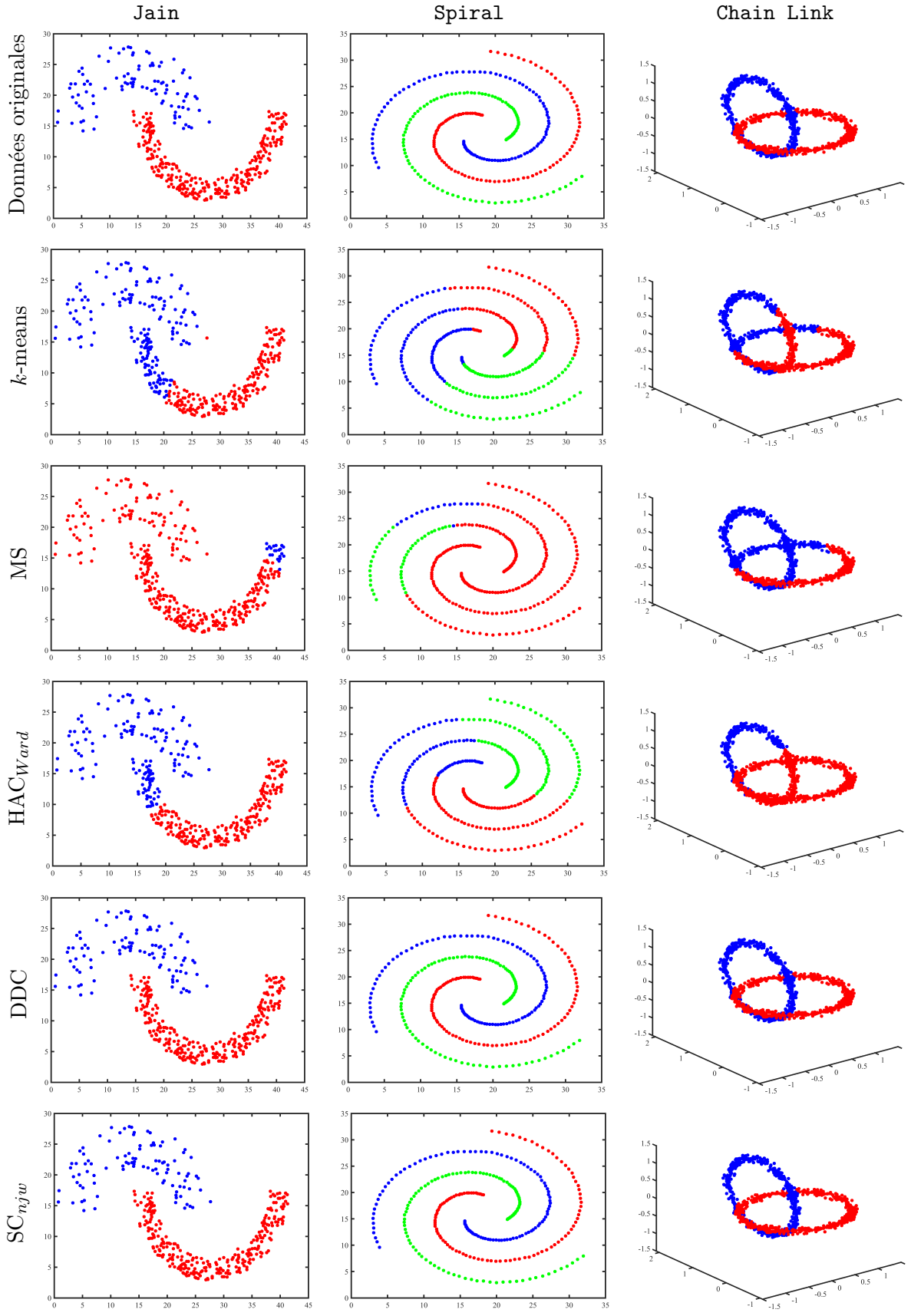


Figure 2.3: Résultats de classification obtenus par les méthodes k-means, MS, HAC$_{Ward}$, DDC, et SC$_{njw}$ sur les bases de données `Jain`, `Spiral`, et `Chain Link`.

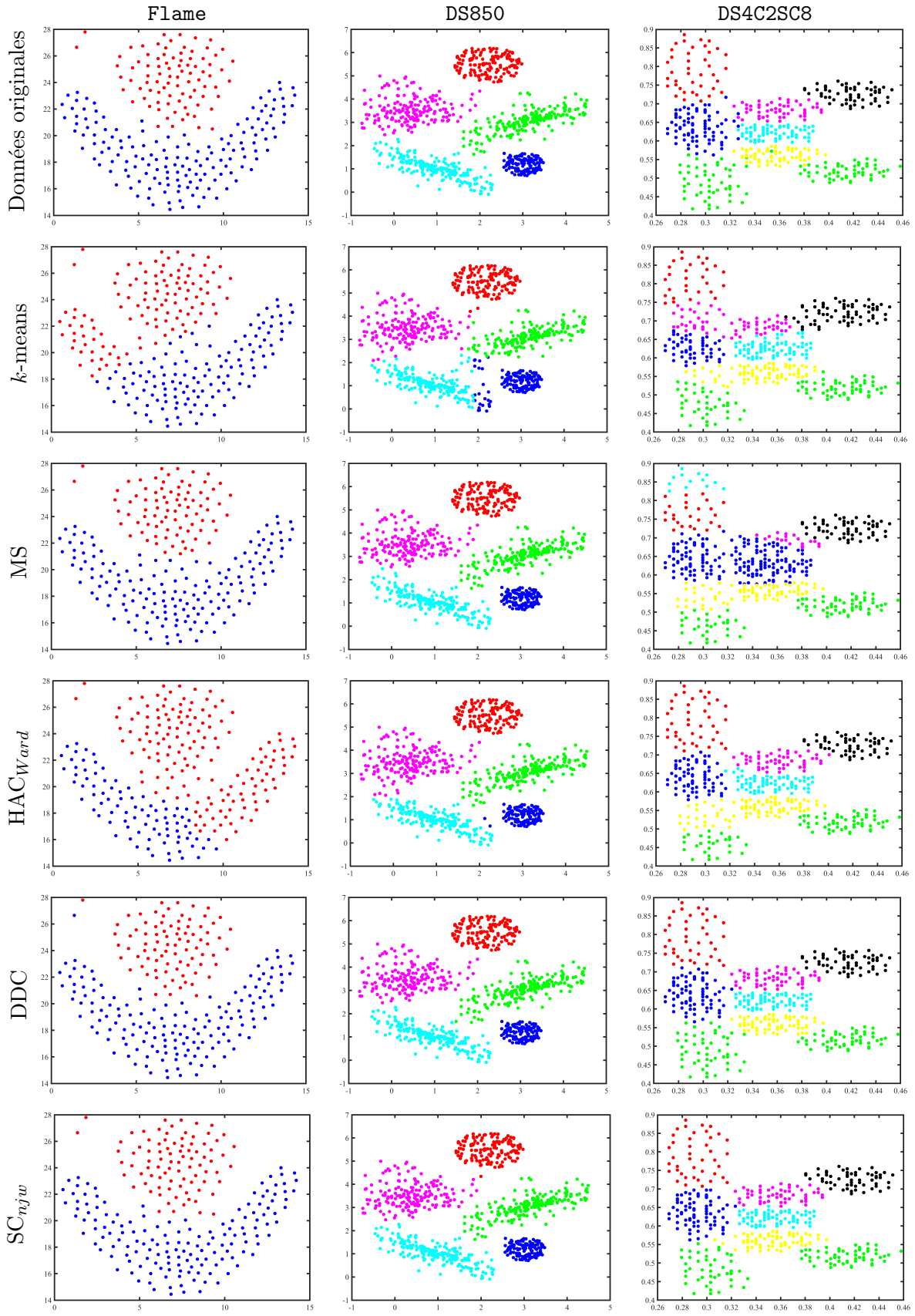


Figure 2.4: Résultats de classification obtenus par les méthodes k-means, MS, HAC$_{Ward}$, DDC, et SC$_{njw}$ sur les bases de données `Flame`, `DS850`, et `DS4C2SC8`.

2.5 Classification spectrale sous contraintes

Une des solutions pour améliorer davantage les résultats de la classification spectrale est de s'orienter vers le contexte semi-supervisé avec contraintes [136, 137]. Les contraintes définissent une relation de similarité entre les données. Par conséquent, elles peuvent être facilement intégrées dans le processus de classification spectrale, qui est lui-même basé sur la notion de similarité. L'objectif de la classification spectrale sous contraintes est de trouver une partition des données non étiquetées qui satisfait les contraintes.

2.5.1 Génération des contraintes

Les ensembles de contraintes peuvent être fournis directement par l'utilisateur ou bien générés à partir de quelques données étiquetées [152]. Dans cette thèse, nous supposons qu'on dispose uniquement de quelques données étiquetées (prototypes) qui caractérisent les k classes ω^l, $l = 1,\ldots,k$ de l'ensemble de données $X = \{x_1,x_2,\ldots,x_n\}$. Soit X^l, l'ensemble des p prototypes associé à la classe ω^l ($X^l \subset X$), et soit X^P l'ensemble de tous les prototypes disponibles $\left(X^P = \bigcup_{l=1}^{k} X^l\right)$. À partir de l'ensemble des prototypes X^P, nous générons les ensembles de contraintes must-link et cannot-link de la manière suivante:

- Une contrainte must-link est intégrée dans l'ensemble des contraintes must-link (M) si et seulement si l'étiquette de classe de la donnée x_i est la même que celle de la donnée x_j. L'ensemble M est défini par:

$$M = \left\{(x_i,x_j) \in X^2 \mid \exists\, l = 1,\ldots,k \text{ tel que } x_i \in X^l \text{ et } x_j \in X^l\right\} \tag{2.28}$$

- Une contrainte cannot-link est intégrée dans l'ensemble des contraintes cannot-link (C) si et seulement si l'étiquette de classe de la donnée x_i est différente de celle de la donnée x_j. L'ensemble C est défini par:

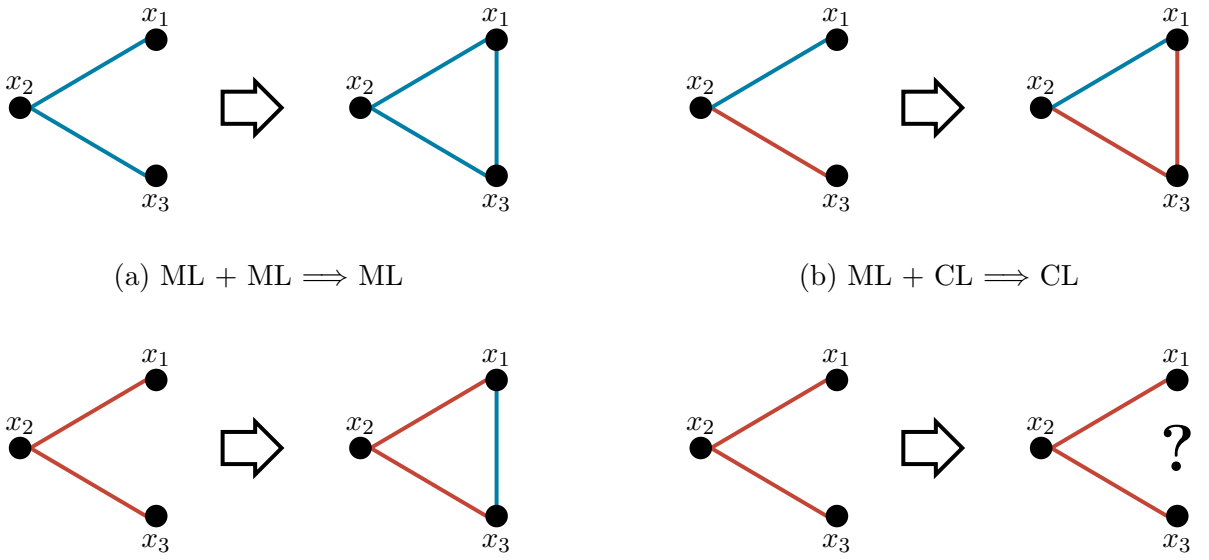
$$C = \left\{(x_i,x_j) \in X^2 \mid \exists\, (l,m);\ l \neq m;\ \text{tel que } x_i \in X^l \text{ et } x_j \in X^m\right\} \tag{2.29}$$

Il est possible d'augmenter automatiquement le nombre de contraintes générées en appliquant des règles de propagation obtenues par les différentes combinaisons possibles des contraintes must-link (ML) et des contraintes cannot-link (CL) (Figure 2.5):

- *Combinaison ML + ML.* Si les deux paires de données (x_1, x_2) et (x_2, x_3) sont liées par des contraintes must-link alors les données x_1 et x_3 sont également liées par une contrainte must-link (Figure 2.5(a)).

– *Combinaison ML + CL.* Si les données x_1 et x_2 sont liées par une contrainte must-link et que les données x_2 et x_3 sont liées par une contrainte cannot-link, alors les données x_1 et x_3 sont liées par une contrainte cannot-link (Figure 2.5(b)).

– *Combinaison CL + CL.* Dans le cas de deux classes, si deux paires de données (x_1, x_2) et (x_2, x_3) sont liées par des contraintes cannot-link alors les données x_1 et x_3 sont liées par une contrainte must-link (Figure 2.5(c)). Par contre, dans le cas multi-classes, il est impossible de prédire la contrainte qui va lier les données x_1 et x_3 (Figure 2.5(d)).



(a) ML + ML $\Longrightarrow$ ML

(b) ML + CL $\Longrightarrow$ CL

(c) CL + CL $\Longrightarrow$ ML (2 classes)

(d) CL + CL $\Longrightarrow$? (multi-classes)

Figure 2.5: Règles de propagation des contraintes [153].

L'avantage de la génération des contraintes à partir des étiquettes de classes est qu'elles sont cohérentes. Cependant, certaines contraintes peuvent diminuer la performance de classification, même si elles proviennent des étiquettes de classes [154]. Certains auteurs proposent alors d'effectuer un choix au préalable sur les contraintes [154, 155].

2.5.2 Représentation des contraintes par des graphes de similarité

Les ensembles de contraintes must-link et cannot-link peuvent être aussi représentés par des graphes de similarité. Il existe plusieurs façons de le faire, qui diffèrent selon le contexte d'utilisation (supervisé ou semi-supervisé) et l'application envisagée (sélection d'attributs ou classification spectrale). Les différentes représentations des données contraintes par des graphes de similarité dans le cadre de la sélection d'attributs supervisée et semi-supervisée sont décrites dans le prochain chapitre.

Dans le cadre de la classification spectrale sous contraintes, Kamvar et al. [134] ont proposé de représenter les données contraintes et les données non étiquetées par un seul graphe de similarité $G^{M,C}$. Dans ce graphe, les valeurs de similarité $w_{i,j}$ entre les paires de données $\{x_i,x_j\} \in M$ sont mises à 1 et les valeurs des similarité $w_{i,j}$ entre les paires de données $\{x_i,x_j\} \in C$ sont mises à 0. La similarité entre les données non étiquetées est évaluée par la fonction gaussienne basée sur la distance euclidienne indiquée dans l'équation (2.8).

La matrice de similarité $\boldsymbol{W}^{M,C}$ correspondante au graphe $G^{M,C}$ est alors définie comme suit:

$$w_{ij}^{M,C} = \begin{cases} 1 & \text{si} \quad (x_i,x_j) \in M \\ 0 & \text{si} \quad (x_i,x_j) \in C \\ \texttt{exp}\left(-\frac{\delta^2(\boldsymbol{x}_i,\boldsymbol{x}_j)}{2\sigma^2}\right) & \text{sinon} \end{cases} \qquad i,j = 1,2,\ldots,n \tag{2.30}$$

2.5.3 Prise en compte des contraintes en classification spectrale

Les contraintes peuvent être intégrées durant l'une des trois principales étapes du processus de la classification spectrale (Section 2.3): *construction du graphe de similarité*, *construction de l'espace spectral*, et *partitionnement des données dans l'espace spectral*.

2.5.3.1 Intégration lors de la construction du graphe de similarité

Les méthodes de cette catégorie intègrent les contraintes lors de la construction du graphe de similarité en modifiant les similarités entre les données contraintes dans la matrice de similarité initiale (Figure 2.6).

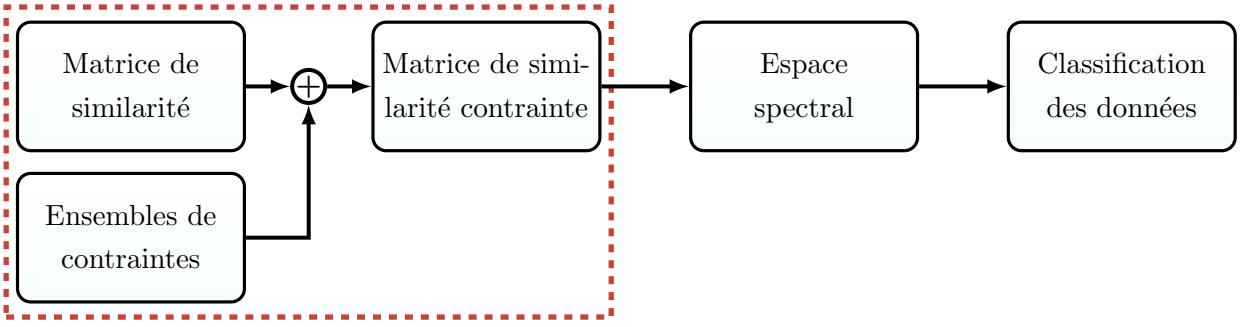


Figure 2.6: Intégration des contraintes lors de la construction du graphe de similarité.

Dans [134], Kamvar et al. ont proposé une méthode de classification spectrale sous contraintes qui force les valeurs de similarités à 1 entre les paires de données liées en must-link et à 0 entre les paires de données liées en cannot-link. La matrice laplacienne normalisée additive

(équation (2.17)) est calculée avec la matrice de similarité $\boldsymbol{W}^{M,C}$ (équation (2.30)). Dans [156], Xu et al. ont proposé une méthode de classification spectrale sous contraintes qui intègre les contraintes must-link et cannot-link dans la matrice de similarité de la même manière que dans [134]. Mais au lieu d'utiliser la matrice laplacienne normalisée additive, les auteurs ont utilisé la matrice laplacienne normalisée asymétrique (équation (2.14)). La méthode de classification spectrale sous contraintes proposée par Lu et Carreira-Perpinan [135], propage l'information contenue dans les contraintes must-link et cannot-link dans la matrice de similarité en utilisant un processus gaussien.

2.5.3.2 Intégration lors de la construction de l'espace spectral

Les méthodes de cette catégorie tentent de corriger l'espace spectral de sorte que les couples de données liées par des contraintes must-link soient proches et les couples de données liées par des contraintes cannot-link soient éloignés (Figure 2.7).

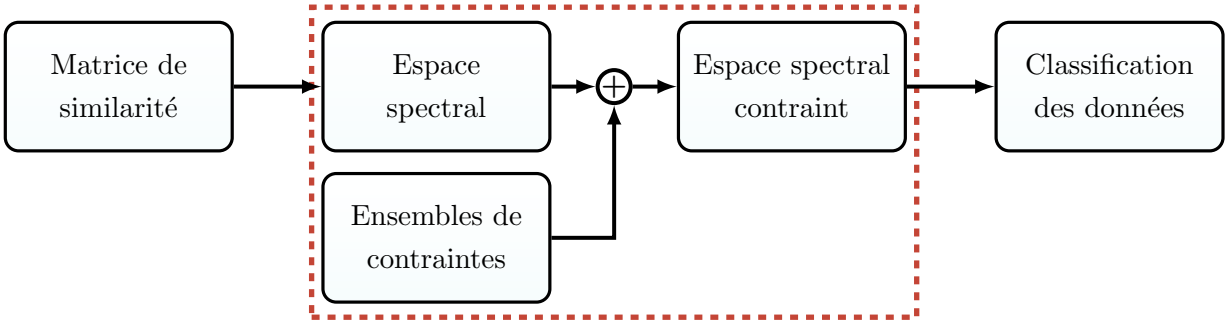


Figure 2.7: Intégration des contraintes lors de la construction de l'espace spectral.

Dans [157], Xu et al. ont introduit les contraintes must-link et cannot-link lors de la construction de l'espace spectral. Li et al. [158], ont proposé une méthode qui intègre des contraintes souples (pouvant ne pas être satisfaites) dans l'espace spectral. Wang et Davidson [152] ont proposé une méthode flexible de classification spectrale qui intègre les contraintes must-link et cannot-link durant la construction de l'espace spectral. Cette méthode permet de régler l'impact des contraintes sur la structure originale des données en utilisant un seuil qui ajuste la quantité minimale de contraintes à respecter. Wacquet et al. [144] ont proposé une méthode de classification spectrale sous contraintes qui introduit des pondérations entre la contribution des données non supervisées et les taux de respect des contraintes. La méthode de classification spectrale proposée par Chatel et al. [159] intègre les contraintes dans l'espace spectral de telle sorte que la majorité des contraintes soient satisfaites, puis les propagent dans l'espace spectral en utilisant une fonction gaussienne.

2.5.3.3 Intégration lors du partitionnement des nouvelles données

Les contraintes peuvent être aussi intégrées lors du partitionnement des données projetées (Figure 2.8). On peut ainsi utiliser une méthode de classification sous contraintes au lieu d'une méthode de classification non supervisée [160]. Parmi ces méthodes, on peut citer: COP k-means [136], HAC sous contraintes [138], PC k-means [137], et MPC k-means [161].

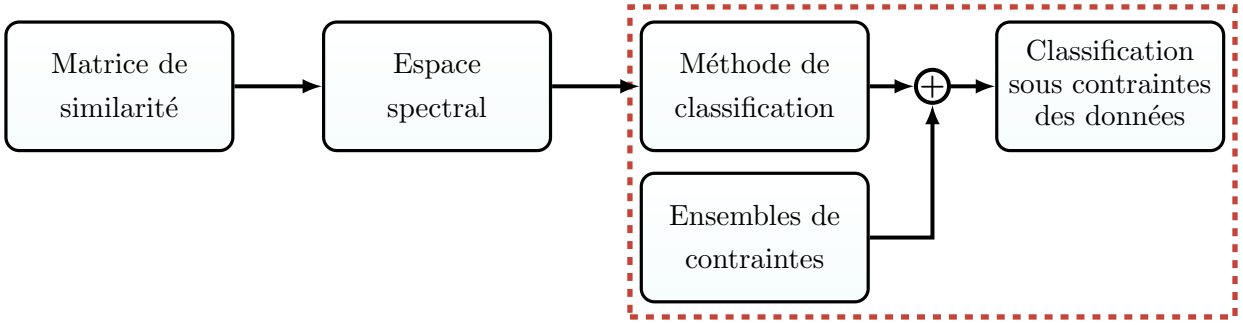


Figure 2.8: Intégration des contraintes lors du partitionnement des nouvelles données.

Dans le cadre de cette thèse, nous considérons la méthode de classification spectrale de Ng et al. (Algorithme 2.1), pour laquelle les contraintes sont intégrées dans la matrice de similarité, comme suggéré par Kamvar et al. [134] (équation (2.30)). L'algorithme 2.2 décrit les principales étapes de la méthode de classification spectrale sous contraintes (CSC$_{njw}$).

Algorithme 2.2: CSC$_{njw}$.

Entrée: Ensemble de données $X = \{x_1, x_2, \ldots, x_n\}$, nombre de classes k, ensembles des contraintes must-link M et cannot-link C.

1. Construire le graphe de similarité G.
2. Calculer la matrice de similarité $\boldsymbol{W}^{M,C}$ (équation (2.30)).
3. Calculer la matrice laplacienne normalisée $\boldsymbol{L}_{Sym}^{M,C}$ associée à $\boldsymbol{W}^{M,C}$.
4. Calculer le spectre de $\boldsymbol{L}_{Sym}^{M,C}$ (valeurs λ_j et vecteurs $\boldsymbol{u}_j$ propres, $j = 1, \ldots, n$).
5. Définir la matrice $\boldsymbol{U}^{M,C} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k] \in \mathbb{R}^{n \times k}$ à partir des k plus grands vecteurs propres correspondants aux k plus petites valeurs propres de $\boldsymbol{L}_{Sym}^{M,C}$.
6. Construire la matrice $\boldsymbol{T}^{M,C} \in \mathbb{R}^{n \times k}$ en normalisant les lignes de $\boldsymbol{U}^{M,C}$ comme suit:
$$t_{ij} = \frac{u_{ij}}{\sqrt{\sum_{j=1}^{k} u_{ij}^2}}$$
7. Définir le nouvel ensemble de données $Z^{M,C}$ dans l'espace spectral où chaque nouvelle donnée $\boldsymbol{z}_i$ est définie par la i-ème ligne de $\boldsymbol{T}^{M,C}$.
8. Classer l'ensemble de données $Z^{M,C}$ en k classes en utilisant la méthode k-means.

Sortie: La classe de chaque donnée x_i de l'ensemble X.

2.6 Résultats de classification spectrale sous contraintes

Pour évaluer la méthode de classification spectrale sous contraintes (CSC$_{njw}$), décrite dans l'algorithme 2.2, nous l'avons comparée à d'autres méthodes de classification semi-supervisée sous contraintes à savoir COP k-means [136], PC k-means [137], et CHAC$_{SL}$ [138]. De plus, pour montrer l'apport des contraintes, nous avons comparé les méthodes de classification semi-supervisée sous contraintes suscitées avec leurs versions non supervisées correspondantes, à savoir, SC$_{njw}$ [77], k-means [54], et HAC$_{SL}$ [118]. Cette comparaison a été effectuée sur les six bases de données réelles présentées dans la section 2.4 (`Iris`, `Wisc`, `Vowels`, `Vote`, `German`, et `Urban`). La qualité de la classification a été mesurée par le taux de classification (accuracy).

Les méthodes de classification semi-supervisée sont exécutées plusieurs fois en générant à chaque fois différents ensembles de contraintes. Les taux de classification obtenus sont alors moyennés. La génération des contraintes s'effectue comme suit: pour chaque classe, nous sélectionnons aléatoirement k sous-ensembles de prototypes X^l ($|X^l| = p$) à partir de l'ensemble de données X. Ensuite, nous déduisons les ensembles de contraintes must-link M et cannot-link C en utilisant respectivement les équations (2.28) et (2.29). Pour cette expérimentation, le nombre de prototypes par classe p varie de 2 à 10 (p doit être ≥ 2, pour générer au moins une contrainte must-link par classe).

Le tableau 2.7 comprend à la fois les taux moyens de classification obtenus par les méthodes de classification non supervisée (k-means, HAC$_{SL}$, et SC$_{njw}$) et les taux moyens de classification obtenus par les méthodes de classification semi-supervisée (COP k-means, PC k-means, CHAC$_{SL}$, et CSC$_{njw}$) sur 20 exécutions avec différents ensembles de contraintes générés avec un nombre de prototypes par classe p égal à 5.

Tableau 2.7: Taux de classification obtenus par les différentes méthodes de classification non supervisée et semi-supervisée sous contraintes ($p = 5$).

Bases de données	**Non supervisée**			**Semi-supervisée avec contraintes**			
	k-means	HAC$_{SL}$	SC$_{njw}$	COP k-means	PC k-means	CHAC$_{SL}$	CSC$_{njw}$
`Iris`	88.67	66.00	96.67	90.73	96.87	**97.40**	97.00
`Wisc`	95.99	65.67	97.14	96.25	96.02	94.29	**97.32**
`Vowels`	50.62	96.63	98.28	50.57	53.04	98.29	**99.07**
`Vote`	86.67	61.61	89.43	87.70	86.76	88.94	**90.11**
`German`	64.10	70.10	70.00	64.89	64.70	64.93	**70.23**
`Urban`	68.45	20.24	70.83	77.08	**84.46**	80.24	84.35

À partir de ce tableau, nous constatons que pour la plupart des bases de données, les performances de toutes les méthodes de classification non supervisée se sont améliorées grâce à l'intégration des contraintes must-link et cannot-link. Cette amélioration peut être considérable comme dans le cas de la base de données `Urban`. Nous remarquons aussi que la méthode CSC_{njw} reste plus performante que les méthodes semi-supervisées (COP k-means, PC k-means, et CHAC_{SL}) et ce pour la majorité des bases de données.

Pour étudier l'influence du nombre de contraintes, nous proposons de comparer les taux moyens de classification obtenus par les méthodes de classification semi-supervisée COP k-means, PC k-means, CHAC_{SL}, et CSC_{njw} pour différents nombres de prototypes p à partir desquels les contraintes sont générées. Pour cela, nous avons varié le nombre p de 2 à 10 et réalisé à chaque fois 20 exécutions.

La figure 2.9 affiche les variations des taux moyens de classification en fonction du nombre de prototypes par classe p obtenus par chaque méthode de classification semi-supervisée. Elle montre que les taux moyens de classification augmentent au fur et à mesure que le nombre de prototypes augmente. En regardant de près les différentes courbes, nous constatons que les courbes de CSC_{njw} se démarquent de celles des autres méthodes, à savoir, COP k-means, PC k-means, et CHAC_{SL}, confirmant ainsi la supériorité de CSC_{njw} sur les autres méthodes quelque soit le nombre de contraintes.

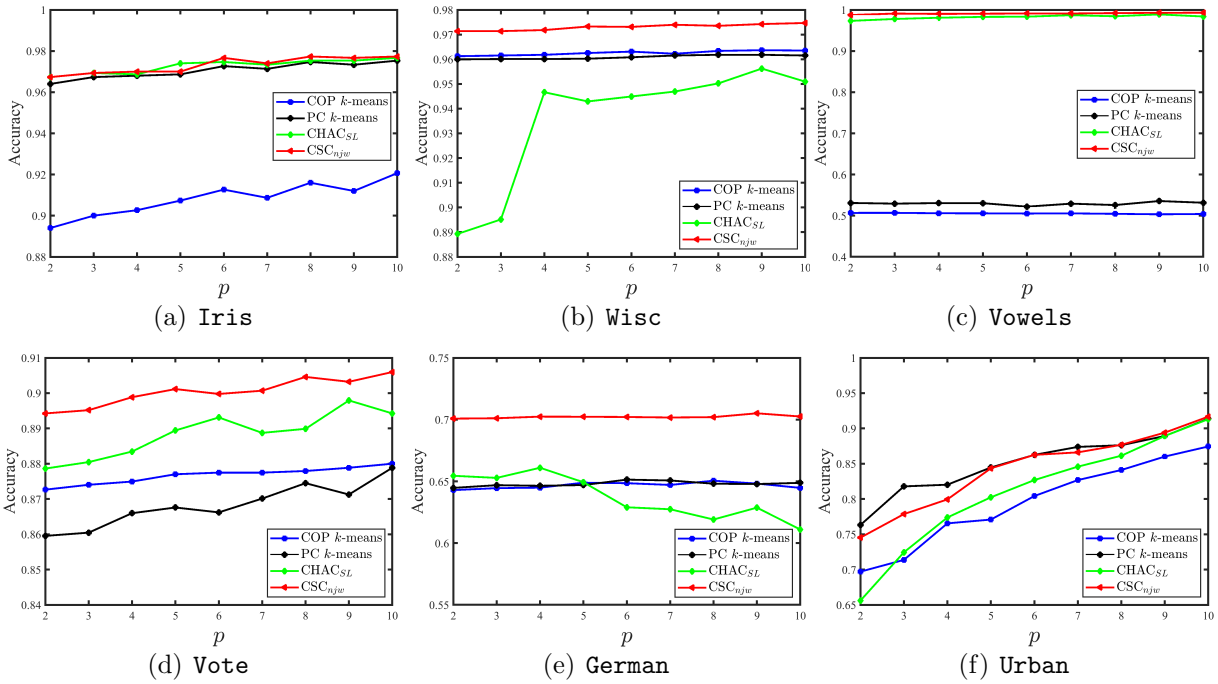


Figure 2.9: Taux moyens de classification obtenus par les méthodes de classification semi-supervisée COP k-means, PC k-means, CHAC_{SL}, et CSC_{njw} en fonction du nombre p de prototypes sur des bases de données réelles.

2.7 Conclusion

Nous avons présenté dans ce chapitre une méthode de classification spectrale sous contraintes et démontré l'intérêt d'intégrer des contraintes must-link et cannot-link dans le processus de classification spectrale. Pour cela, nous avons montré comment représenter les données par un graphe de similarité et les différentes matrices associées, telles que les matrices de similarité et les matrices laplaciennes. Ce type de représentation nous a permis de décrire le processus de classification spectrale par un problème de coupe de graphe qui est résolu par la projection des données originales dans un espace de dimension réduite engendré par une partie des vecteurs propres calculés à partir de la matrice laplacienne. Dans cet espace, les classes sont censées être facilement détectables par une simple méthode de classification non supervisée, telle que k-means. Une série de tests menés sur des bases de données synthétiques et réelles a confirmé la supériorité de la classification spectrale sur les méthodes classiques comme k-means, FCM, MS, HAC, DBSCAN, et DDC.

Par la suite, nous avons montré comment générer les ensembles de contraintes et les différentes manières de les intégrer dans le processus de classification spectrale non supervisée. Nous avons retenu la méthode la plus simple qui consiste à modifier la matrice de similarité en forçant à 1 les similarités des données de type must-link et à 0 les similarités des données de type cannot-link, puis appliquer la méthode de classification spectrale classique en tenant compte de cette matrice de similarité modifiée. Des résultats expérimentaux attestent, d'une part, de l'intérêt d'utiliser les contraintes en classification et d'autre part, de l'efficacité de la méthode de classification spectrale sous contraintes par rapport aux méthodes de classification semi-supervisée avec contraintes.

La classification spectrale sous contraintes possède tout de même certains inconvénients comme une applicabilité limitée aux petits ensembles de données. Ce problème sera traité dans le chapitre 5 dans le cadre de la segmentation d'images. En plus de cet inconvénient, il est important de souligner que les résultats obtenus par la classification spectrale sous contraintes dépendent des similarités entre les données, qui sont conditionnées par les distances δ calculées en utilisant l'ensemble des d attributs. Utiliser l'ensemble des attributs peut ne pas toujours améliorer les résultats de classification, mais parfois les dégrader. La raison est que dans l'ensemble des attributs, certains d'entre eux sont redondants, non pertinents, et inutiles pour la classification spectrale. L'une des solutions à ce problème est de réduire la dimension des données originales en sélectionnant les attributs les plus discriminants pour la classification spectrale. C'est à cette tâche que nous consacrons les deux prochains chapitres.

Chapitre 3

Sélection d'attributs sous contraintes

3.1 Introduction

Dans le domaine de la classification des données et de la segmentation d'images, il est courant d'avoir un grand nombre d'attributs pour caractériser chaque donnée ou pixel. Bien que l'utilisation d'un grand nombre d'attributs puisse théoriquement améliorer les performances de la classification ou de la segmentation d'images. Néanmoins, en pratique, certains de ces attributs peuvent être non pertinents, redondants, et peu informatifs. Ils peuvent conduire à de mauvais résultats de classification ou de segmentation d'images, augmenter les temps de calcul, et rendre la visualisation et l'interprétation des données plus difficiles. Par conséquent, il est impératif de les identifier puis de les supprimer de l'ensemble total des attributs avant d'effectuer une classification des données ou une segmentation d'images. Cette tâche peut être réalisée par des techniques de réduction de dimension, qui se déclinent en deux approches: *extraction d'attributs* et *sélection d'attributs*. On s'intéressera dans ce chapitre à la sélection d'attributs et plus précisément aux méthodes de sélection d'attributs basées sur les graphes, dans les contextes d'apprentissage non supervisé, supervisé, et semi-supervisé avec et sans contraintes de type must-link et cannot-link.

Nous entamons ce chapitre par la définition de la réduction de dimension, puis nous présentons les différentes notions liées à la sélection d'attributs, tout en donnant un aperçu général des principales étapes du processus de sélection d'attributs. Par la suite, nous présentons les différentes catégorisations des méthodes de sélection d'attributs proposées dans la littérature. Enfin, nous terminerons ce chapitre par un état de l'art sur les méthodes de sélection d'attributs de type filtre basées sur les scores de pertinence dans les trois contextes d'apprentissage: non supervisé, supervisé, et semi-supervisé.

3.2 Réduction de la dimension des données

La réduction de dimension est une étape de pré-traitement souvent utile pour la classification. Elle offre deux avantages majeurs. Le premier est d'éviter le phénomène de la malédiction de la dimension. Le second est d'améliorer la performance de l'algorithme d'apprentissage et/ou de réduire son temps de calcul. On décèle dans la littérature deux manières de réduire la dimension des données, par *extraction d'attributs* ou par *sélection d'attributs*.

L'extraction d'attributs consiste à combiner l'ensemble original des attributs d'une manière linéaire ou non pour en générer un nouvel ensemble d'attributs de taille réduite et qui conserve au mieux l'information originale. De nombreuses méthodes de réduction de la dimension par extraction d'attributs ont été proposées dans la littérature [162]. Parmi elles, on peut citer: principal component analysis (PCA) [163], multi dimensional scaling (MDS) [164], fisher linear discriminant (FLD) [165], et locally preserving projection (LPP) [166].

La sélection d'attributs, quant à elle, consiste à sélectionner à partir de l'ensemble original des attributs, les attributs les plus pertinents pour la tâche de classification sans changer leurs significations physiques (la sémantique des attributs originaux est préservée). Notre intérêt dans cette thèse s'est porté sur la sélection d'attributs, pour laquelle de nombreuses méthodes ont été proposées dans la littérature [162, 167].

3.3 Sélection d'attributs

La sélection d'attributs (aussi appelée *sélection de variables* ou *de caractéristiques*) est définie comme un problème d'optimisation [168]. Elle peut être formulée comme suit:

Soit $X = \{x_1,x_2,\ldots,x_n\}$ un ensemble de n données où chaque donnée est caractérisée par un ensemble de d attributs $F_d = \{f_1,f_2,\ldots,f_r,\ldots,f_d\}$. L'ensemble X peut être représenté par une matrice $\boldsymbol{X} = [\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_n] \in \mathbb{R}^{n\times d}$, où $\boldsymbol{x}_i = [x_{i1},x_{i2},\ldots,x_{ir},\ldots,x_{id}] \in \mathbb{R}^d$ désigne la i-ème donnée, et x_{ir} $(r = 1,\ldots,d)$ la r-ème valeur d'attribut de la i-ème donnée. L'ensemble des d attributs peut être également représenté par une matrice $\boldsymbol{X}^T = [\boldsymbol{f}_1,\boldsymbol{f}_2,\ldots,\boldsymbol{f}_r,\ldots,\boldsymbol{f}_d]$, où $\boldsymbol{f}_r = [x_{1r},x_{2r},\ldots,x_{nr}] \in \mathbb{R}^n$ désigne le r-ème vecteur d'attributs de l'ensemble de données. L'objectif de la sélection d'attributs est de chercher le sous-ensemble d'attributs F^*_m $(F^*_m \subseteq F_d)$ de taille m $(m \leqslant d)$ qui optimise une fonction d'évaluation J (maximisation ou minimisation selon la nature de la fonction J) tel que:

$$J(F^*_m) = \max_{F_m \subseteq F_d} J(F_m) \tag{3.1}$$

Dans le cas où le nombre d'attributs qui caractérisent les données est supérieur au nombre de données ($d > n$), le phénomène dit *malédiction de la dimension* est évoqué. De plus, lorsque le nombre d'attributs est grand, il existe une forte probabilité que certains attributs soient non pertinents, redondants, ou carrément inutiles.

3.3.1 Malédiction de la dimension

La malédiction de la dimension (curse of dimensionality), originellement identifiée par Richard Bellman [169], désigne les différents phénomènes qui se produisent lorsque les données sont étudiées dans des espaces de grande dimension et qui ne se produisent pas lorsque les données sont analysées dans des espaces de faible dimension. Par exemple, lorsque la dimension augmente, le volume de l'espace augmente aussi et, par conséquent, les données se retrouvent isolées les unes des autres, ce qui aura un impact sur leur analyse.

Pour bien comprendre ce phénomène, considérons l'exemple de classification de la figure 3.1. L'ensemble de données dans la figure 3.1(a) contient 300 données caractérisées par $d = 2$ attributs et réparties en 3 classes. La matrice de distance euclidienne calculée sur cet ensemble de données est bloc diagonal (Figure 3.1(b)). L'application de la méthode k-means permet d'identifier les trois classes (Figure 3.1(c)). Cependant, lorsque le nombre d'attributs est augmenté à $d = 500$, la matrice de distance euclidienne n'est plus bloc diagonal (Figure 3.1(d)) et la méthode k-means ne peut plus identifier les trois classes (Figure 3.1(e)).

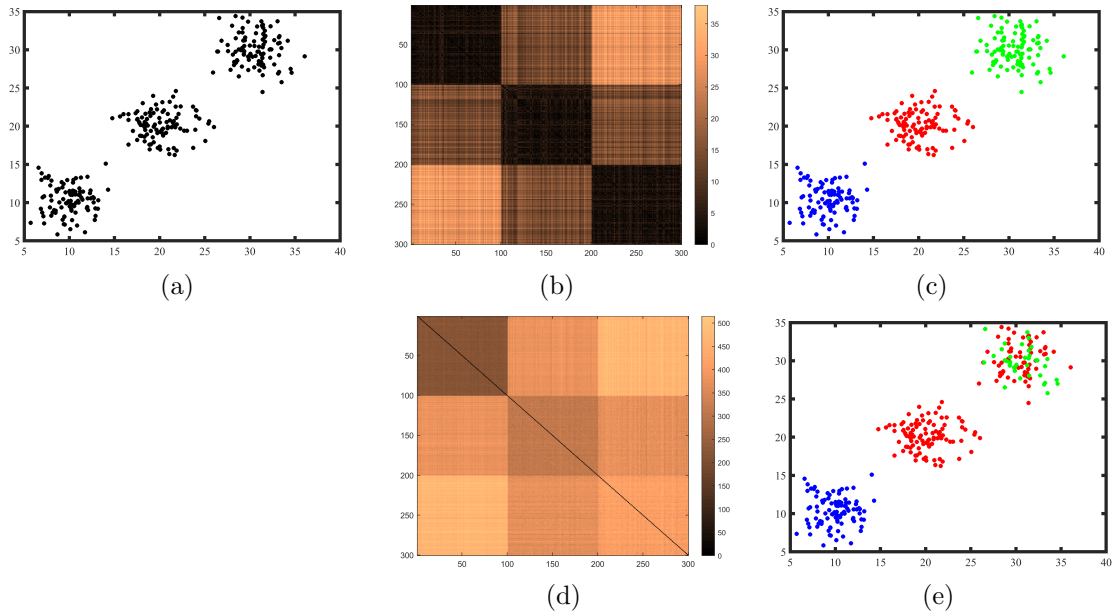


Figure 3.1: Illustration de la malédiction de la dimension. (a) Données caractérisées par $d = 2$. (b) Matrice de distance calculée avec $d = 2$. (c) Résultat de k-means pour $d = 2$. (d) Matrice de distance calculée avec $d = 500$. (e) Résultat de k-means pour $d = 500$.

3.3.2 Pertinence et redondance des attributs

La sélection d'attributs est fortement liée aux notions de pertinence et de redondance. Dans le cas de la classification, la pertinence d'un attribut est liée au pouvoir discriminant. On dit qu'un attribut est pertinent si son absence détériore le résultat de la classification [170, 171]. On distingue trois types de pertinences:

- Un attribut f_i est *fortement pertinent* si son absence implique une détérioration significative du résultat de classification. f_i est alors considéré comme étant indispensable et devrait figurer dans chaque sous-ensemble optimal d'attributs.
- Un attribut f_i est *faiblement pertinent* si f_i n'est pas fortement pertinent, mais il existe un sous-ensemble F_m tel que la performance de $F_m \cup \{f_i\}$ est meilleure que celle de F_m.
- Un attribut f_i est *non pertinent* s'il n'est ni fortement pertinent ni faiblement pertinent. Dans ce cas, sa présence est considérée inutile dans un sous-ensemble optimal d'attributs.

La redondance est exprimée en termes de corrélation entre deux ou plusieurs attributs [168]. Deux attributs sont redondants s'ils sont fortement corrélés.

En fonction des notions de pertinence et de redondance, un ensemble d'attributs peut être divisé en quatre catégories, comme le montre la figure 3.2 [172]: non pertinents (partie I), faiblement pertinents et redondants (partie II), faiblement pertinents et non redondants (partie III), et fortement pertinents (partie IV). Un sous-ensemble optimal d'attributs regroupe les parties III et IV.

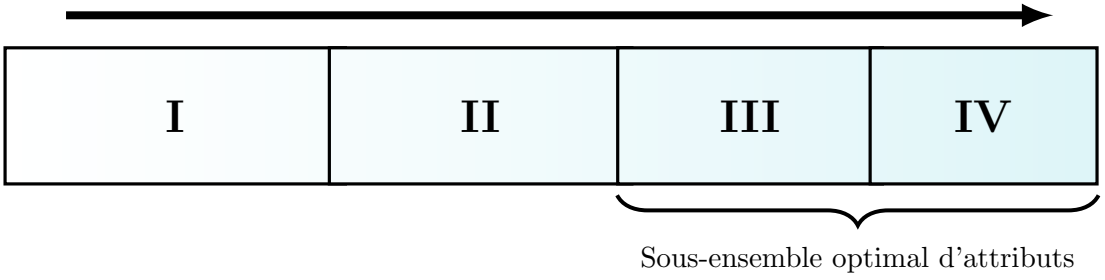


Figure 3.2: Catégorisation d'un ensemble d'attributs [172].

3.3.3 Processus de sélection d'attributs

Selon Dash et Liu [173], le processus de sélection d'attributs se déroule en quatre principales étapes, comme le montre la figure 3.3 : *procédure de génération*, *fonction d'évaluation*, *critère d'arrêt*, et *procédure de validation*.

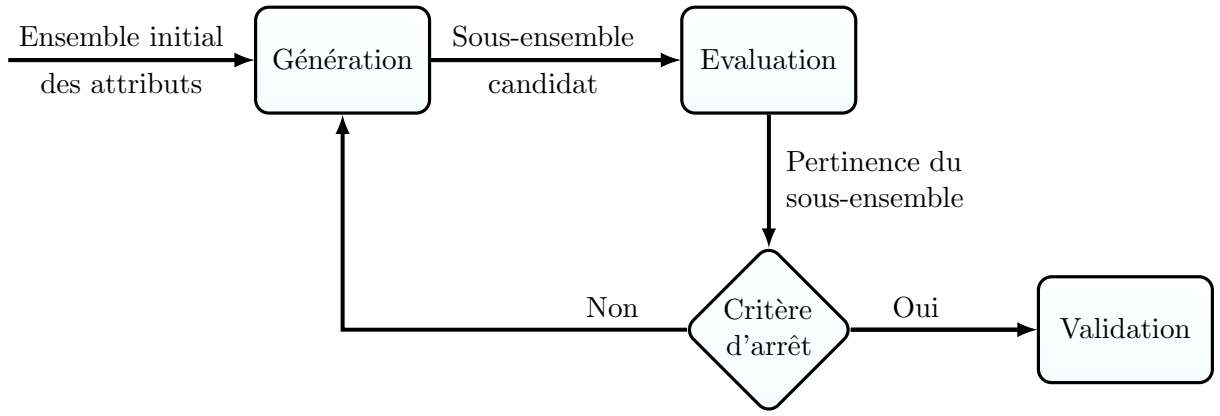


Figure 3.3: Illustration des différentes étapes de la sélection d'attributs [173].

Le processus de sélection d'attributs commence par la génération des sous-ensembles d'attributs candidats. Chaque sous-ensemble est évalué avec une mesure d'évaluation, puis comparé au sous-ensemble d'attributs précédent. Les procédures de génération et d'évaluation sont répétées jusqu'à la satisfaction du critère d'arrêt. Le sous-ensemble d'attributs sélectionnés est finalement validé par des tests.

3.3.3.1 Procédure de génération

Dans cette étape, la génération d'un sous-ensemble d'attributs candidats peut commencer soit avec un ensemble vide d'attributs, avec tous les attributs, ou avec un sous-ensemble d'attributs. De plus, elle peut être: *complète*, *séquentielle*, ou *aléatoire* [174].

Une procédure de *génération complète* évalue tous les sous-ensembles d'attributs candidats possibles, puis sélectionne le sous-ensemble optimal selon la fonction d'évaluation. Cette procédure exhaustive permet de déterminer le sous-ensemble optimal, mais elle est coûteuse en temps de calcul, notamment lorsque le nombre de sous-ensembles candidats est grand (pour un ensemble de d attributs, il existe 2^d sous-ensembles candidats possibles). Parmi les méthodes de sélection d'attributs qui utilisent ce type de génération, on peut citer: FOCUS [175] et Automatic Branch Bound [176].

Une procédure de *génération séquentielle* ne parcourt pas tous les sous-ensembles d'attributs candidats, elle rajoute ou supprime un ou plusieurs attributs au fil des itérations. Cette génération ne garantit pas de trouver le sous-ensemble optimal d'attributs, mais elle est plus rapide que la génération complète. La génération séquentielle peut être soit: *forward* (elle commence avec un ensemble vide d'attributs, puis les attributs pertinents et non redondants sont ajoutés de manière successive à chaque itération), *backward* (elle commence avec l'ensemble de tous les attributs, puis les attributs non pertinents et redondants sont supprimés

à chaque itération), ou *stepwise* (elle ajoute et supprime des attributs à chaque itération). Parmi les méthodes de sélection d'attributs qui utilisent cette génération, on peut citer: sequential forward selection (SFS) [177]. Celle-ci sera exploitée dans le prochain chapitre.

Une procédure de *génération aléatoire* parcourt au hasard l'ensemble des sous-ensembles d'attributs candidats possibles. Elle s'assure qu'à chaque itération, le sous-ensemble généré est meilleur que celui de l'itération précédente. Tout comme la génération séquentielle, elle ne garantit pas de trouver le sous-ensemble optimal d'attributs. Parmi les méthodes de sélection d'attributs qui utilisent cette procédure de génération, on peut citer: les algorithmes évolutionnaires tels que les algorithmes génétiques et les essaims de particules [178].

3.3.3.2 Fonction d'évaluation

Dans cette étape, la pertinence des attributs ou des sous-ensembles d'attributs générés dans l'étape précédente est mesurée par une fonction d'évaluation. Les fonctions d'évaluation proposées dans la littérature peuvent être classées en cinq types [173, 179]:

Mesures du taux de classification: Elles évaluent la pertinence des attributs par l'intermédiaire des taux de classification. Ces mesures permettent de sélectionner le sous-ensemble d'attributs le plus discriminant. Cependant, elles sont coûteuses en temps de calcul car elles nécessitent une classification pour chaque sous-ensemble d'attributs candidats.

Mesures de distance: Elles évaluent la capacité d'un attribut ou d'un sous-ensemble d'attributs à séparer les classes. Les attributs qui maximisent la dispersion inter-classes et minimisent la dispersion intra-classe des données dans l'espace des attributs sont choisis.

Mesures d'information: Elles déterminent le gain d'information apporté par un attribut en calculant la différence entre la probabilité a priori et la probabilité a posteriori. Parmi ces mesures on peut citer: l'information mutuelle et la mesure d'entropie [180].

Mesures de dépendance: Elles calculent le degré de corrélation entre les attributs. Ces mesures permettent de caractériser la corrélation entre un attribut et les classes (pertinence) ou la corrélation entre les attributs eux-mêmes (redondance).

Mesures de consistance: Elles évaluent la capacité d'un attribut à discriminer les classes. Un attribut est dit consistant s'il contient suffisamment d'informations pour séparer les différentes classes. Dans le cas contraire, l'attribut est dit non consistant. La consistance est évaluée en fonction d'une mesure de séparabilité entre toutes les paires de classes [179].

3.3.3.3 Critère d'arrêt

Le critère d'arrêt permet de stopper le processus de sélection d'attributs. Il dépend de la procédure de génération et/ou de la fonction d'évaluation. Le choix d'un critère d'arrêt est très important car les résultats de la méthode de sélection d'attributs en dépendent. Malheureusement, il est très difficile de faire ce choix. Dans la pratique, les critères d'arrêt les plus utilisés sont: le nombre d'attributs sélectionnés, le temps de calcul, le nombre d'itérations, et le taux d'erreur de classification [173, 179].

3.3.3.4 Procédure de validation

La procédure de validation permet de tester la pertinence des attributs sélectionnés sur des bases de données artificielles et/ou réelles. Généralement, l'ensemble de données est divisé en: *un sous-ensemble d'apprentissage* dont on connait les étiquettes de classes des données qui le constitue et qui sert à sélectionner les attributs les plus pertinents et *un sous-ensemble de test* dont on ne connait pas les étiquettes de classes et qui sert à évaluer les attributs sélectionnés. En fonction de la répartition des données entre ces deux sous-ensembles, la procédure de validation peut s'effectuer avec cinq méthodes différentes [179, 181]:

Méthode de resubstitution: Le même ensemble de données utilisé comme ensemble d'apprentissage est également utilisé comme ensemble de test.

Méthode holdout: L'ensemble de données est partitionné en deux sous-ensembles: apprentissage et test. Le sous-ensemble d'apprentissage contient au moins 50% de l'ensemble des données, et le reste des données forme le sous-ensemble de test.

Méthode p-validation croisée: L'ensemble de données est divisé aléatoirement en p sous-ensembles de taille à peu près égale. Ensuite,un sous-ensemble est utilisé pour la validation et les $p-1$ sous-ensembles restants pour l'apprentissage. Ce processus est répété pour chacun des p sous-ensembles de validation. Le résultat final est la moyenne des p performances.

Méthode leave-one-out: Cette méthode est un cas particulier de la méthode p-validation croisée avec $p = n$, n est le nombre de données. Dans ce cas, l'ensemble de test comprend une seule donnée. Cette méthode est recommandée lorsque l'ensemble de données est petit.

Méthode bootstrap: Le sous-ensemble d'apprentissage est généré en tirant, avec remise, un certain nombre de données à partir de l'ensemble total des données. Dans ce cas, une donnée peut ne pas apparaître ou apparaître plusieurs fois dans le sous-ensemble d'apprentissage. Le sous-ensemble de test est constitué soit de toutes les données, soit de celles non sélectionnées.

3.4 Approches de la sélection d'attributs

Différentes méthodes de sélection d'attributs ont été proposées dans la littérature [182–184]. Ces méthodes peuvent être catégorisées de trois façons différentes: selon le nombre d'attributs évalués, selon la fonction d'évaluation, et selon le contexte d'apprentissage.

3.4.1 Catégorisation selon le nombre d'attributs évalués

La sélection d'attributs évalue la pertinence d'un ou plusieurs attributs à la fois en leur attribuant un score par l'intermédiaire des mesures d'évaluation citées précédemment. On distingue alors deux catégories de méthodes [185]:

Méthodes de classement d'attributs (*feature ranking*): Ces méthodes attribuent un score de pertinence à chaque attribut indépendamment des autres, puis elles les rangent dans l'ordre décroissant (croissant) en fonction de leurs scores. Les m premiers attributs sont finalement sélectionnés, le plus souvent de manière empirique par l'utilisateur. Parmi les méthodes de cette catégorie, on peut cité: ReliefF [186], mRMR [187], et le score Laplacien (LS) [188].

Méthodes de recherche de sous-ensembles d'attributs (*subset selection*): Ces méthodes évaluent des sous-ensembles d'attributs candidats dans leurs totalité, puis sélectionnent le sous-ensemble le plus pertinent selon un critère de sélection. Parmi les méthodes de cette catégorie, on peut citer: la sélection d'attributs basée sur la corrélation [189].

3.4.2 Catégorisation selon la fonction d'évaluation

Selon la dépendance ou non à l'algorithme de classification, les méthodes de sélection d'attributs peuvent être classées en trois catégories [167, 174, 182, 190]:

Méthodes filtres (*filter*): Ces méthodes effectuent la sélection d'attributs indépendamment de l'algorithme de classification. Elles attribuent un score de pertinence à chaque attribut (feature ranking) ou à un sous-ensemble d'attributs (subset selection) en utilisant uniquement des mesures de distance, d'information, de dépendance, ou de consistance.

Méthodes enveloppes (*wrapper*): Ces méthodes effectuent la sélection d'attributs en utilisant un algorithme d'apprentissage comme fonction d'évaluation. L'algorithme d'apprentissage est utilisé pour attribuer un score de pertinence, en l'occurrence le taux de classification, à chaque attribut ou sous-ensemble d'attributs candidats.

Méthodes hybrides (*embedded*): Ces méthodes effectuent la sélection d'attributs en combinant les approches filtres et enveloppes afin de bénéficier de leurs avantages respectifs. Dans une méthode hybride, une fonction d'évaluation de type filtre est d'abord utilisée pour présélectionner les sous-ensembles d'attributs les plus pertinents. Ensuite, les taux d'erreur de classification obtenus sur les sous-ensembles pré-sélectionnés lors de l'étape précédente sont comparés pour déterminer le sous-ensemble optimal d'attributs.

3.4.3 Catégorisation selon le contexte d'apprentissage

Au même titre que la classification, les méthodes de sélection d'attributs peuvent être classées en trois catégories selon la disponibilité ou non d'informations a priori sur les données [167, 190]:

Méthodes non supervisées: Dans un contexte non supervisé, les méthodes de sélection d'attributs ont pour objectif de trouver un sous-ensemble d'attributs pertinents sans l'utilisation d'aucune information a priori sur l'appartenance des données aux classes. Elles consistent à sélectionner les attributs de l'ensemble de données non étiquetées, en exploitant leurs similarités, les variances de leurs attributs (score de la variance [191]), ou en préservant leur graphe de similarité (score Laplacien [188]).

Méthodes supervisées: Dans un contexte supervisé, les méthodes de sélection d'attributs utilisent uniquement les données étiquetées ou les données contraintes pour choisir le sous-ensemble d'attributs le plus pertinent. Dans le cas des données étiquetées, le principe consiste à mesurer la corrélation entre les données projetées dans l'espace des attributs et les étiquettes de classes. Le score de Fisher [192] en est un exemple le plus connu. Dans le cas des données contraintes, il s'agit de sélectionner les attributs qui respectent les contraintes must-link et cannot-link. Deux scores de cette approche sont proposés par Zhang et al. [193] et décrits dans la section 3.6.2.

Méthodes semi-supervisées: Dans un contexte semi-supervisé, les méthodes de sélection d'attributs évaluent la pertinence d'un attribut en tenant compte à la fois les données non étiquetées X^U et les données étiquetées X^P ou des lonnées contraintes must-link M et cannot-link C. Les données non étiquetées permettent de préserver la structure originale de l'ensemble de données et les données étiquetées ou contraintes permettent de maximiser la dispersion interclasses [167]. Un état de l'art sur les scores de contraintes semi-supervisés est dressé en section 3.6.3.

3.5 Exemples de méthodes de sélection supervisée

Nous nous sommes intéressé dans notre travail aux méthodes de type filtre par classement des attributs à cause de leurs avantages (simplicité, rapidité, indépendance par rapport aux algorithmes d'apprentissage). Nous décrirons dans cette section, deux méthodes bien connues de sélection d'attributs supervisée utilisant des données étiquetées comme information a priori, puis dans la section suivante, nous dresserons un état de l'art sur les méthodes de sélection d'attributs sous contraintes.

3.5.1 Méthode ReliefF

La méthode ReliefF [186] est une version améliorée de la méthode de sélection d'attributs Relief [194]. Elle est de type filtre et s'applique dans un contexte supervisé. La méthode ReliefF mesure la capacité de chaque attribut à regrouper les données appartenant à la même classe et à discriminer celles appartenant à des classes différentes. Son idée de base est d'estimer un poids w_r pour chaque attribut f_r à partir des données de même et différentes classes. Pour cela, il faut d'abord choisir aléatoirement une donnée x_i de l'ensemble X, puis chercher l'ensemble des K plus proches voisins de x_i ($KNN(\boldsymbol{x}_i)$) dont les étiquettes sont les mêmes que celle de x_i, noté $Hits_i$, et les différents ensembles des K plus proches voisins de x_i dont les étiquettes sont différentes de celle de x_i, notés $Miss_i(\omega^l)$, avec $l = 1,\ldots,k$. Ensuite, à chaque itération t ($t = 1,\ldots,T$), la valeur du poids w_r est mise à jour par l'équation suivante:

$$w_r = w_r - \sum_{j=1}^{K} \frac{diff(\boldsymbol{f}_r,\boldsymbol{x}_i,Hits_j)}{\tau \cdot K} + \sum_{\omega^l \neq \omega(x_i)} \frac{\frac{P(\omega^l)}{1-P(\omega(x_i))} \sum_{j=1}^{K} diff(\boldsymbol{f}_r,\boldsymbol{x}_i,Miss_j(\omega^l))}{\tau \cdot K} \tag{3.2}$$

où $P(\omega^l)$ est la probabilité antérieure de la classe ω^l. $diff(\boldsymbol{f}_r,\boldsymbol{x}_i,Hits_j)$ et $diff(\boldsymbol{f}_r,\boldsymbol{x}_i,Miss_j(\omega^l))$ sont des mesures de différence.

Pour deux données x_i et x_j, la mesure de différence entre les valeurs de l'attribut f_r est définie par:

$$diff(\boldsymbol{f}_r,\boldsymbol{x}_i,\boldsymbol{x}_j) = \frac{|x_{ir} - x_{jr}|}{\max(\boldsymbol{f}_r) - \min(\boldsymbol{f}_r)} \tag{3.3}$$

Le facteur τ dans l'équation (3.2) permet de normaliser les poids $\boldsymbol{w}$ dans l'intervalle $[-1,+1]$.

Les principales étapes de la méthode ReliefF sont décrites dans l'algorithme 3.1.

Algorithme 3.1: ReliefF.

Entrée: Ensemble de données $X = \{x_1,x_2,\ldots,x_n\}$, ensemble des attributs $F_d = \{f_1,\ldots,f_r,\ldots,f_d\}$, ensemble des étiquettes de classes des données $\Omega = \{\omega(x_1),\omega(x_2),\ldots,\omega(x_n)\}$, nombre d'itérations T, seuil τ.

1. Initialiser tous les poids w_r ($r = 1$ à d) à 0.
2. Pour $t = 1$ à τ (τ est limité à n)
   (a) Choisir aléatoirement une donnée x_i.
   (b) Trouver l'ensemble $Hits_i$ des $KNN(x_i)$ appartenant à la même classe que x_i.
   (c) Pour chaque classe $\omega^l \neq \omega(x_i)$, trouver l'ensemble $Miss_i(\omega^l)$ des $KNN(x_i)$ appartenant à des classes différentes de celle de x_i.
   (d) Mettre à jour les poids w_r ($r = 1$ à d) en utilisant l'équation (3.2).
3. Choisir les m attributs ayant un fort poids $F_m = \{f_r | w_r > \tau\}$.

Sortie: Sous-ensemble F_m des m attributs sélectionnés.

3.5.2 Méthode mRMR

La méthode mRMR (minimal redundancy-maximal relevance) est une méthode de sélection d'attributs de type filtre, proposée par Peng et al. [187] et appliquée dans un contexte supervisé. Elle utilise comme fonction d'évaluation la mesure d'information. L'idée principale de mRMR est de sélectionner les attributs qui simultanément minimisent la redondance et maximisent la pertinence, définies pour un attribut f_r par les équations ci-dessous:

$$\text{Redondance } (f_r) = \frac{1}{d^2} \sum_{f_r,f_q \in F_d} IM(f_r,f_q) \tag{3.4}$$

$$\text{Pertinence } (f_r) = \frac{1}{d} \sum_{f_r,f_q \in F_d} IM(f_r,X) \tag{3.5}$$

$IM(f_r,f_q)$ représente l'information mutuelle entre le r-ème attribut f_r et le q-ème attribut f_q, et $IM(f_r,X)$ l'information mutuelle entre le r-ème attribut et l'ensemble de données X appartenant à la classe ω^l. L'information mutuelle entre deux variables a et b est définie en terme de leurs fonctions de densités de probabilités $p(a)$, $p(b)$, et $p(a,b)$ comme suit [187]:

$$IM(a,b) = \sum_a \sum_b p(a,b) \log \frac{p(a,b)}{p(a)p(b)} \tag{3.6}$$

Le score mRMR d'un attribut f_r est la combinaison des termes des équations (3.4) et (3.5) tel que:

$$\text{Score}_{mRMR}(f_r) = \text{Redondance } (f_r) - \text{Pertinence } (f_r) \tag{3.7}$$

La sélection d'attributs basée sur la méthode mRMR consiste à ordonner les attributs f_r dans l'ordre décroissant de leurs scores Score_{mRMR}, pour ensuite sélectionner les m premiers attributs, c.-à-d. ceux dont le Score_{mRMR} est supérieur à un seuil.

3.6 Méthodes de sélection sous contraintes

Nous nous focalisons dans cette section sur les méthodes de sélection d'attributs de type filtre qui exploitent des contraintes (must-link et cannot-link) et leurs représentations par des graphes de similarité. Ces méthodes sont basées sur l'analyse de la structure du graphe de similarité pour évaluer la pertinence de chaque attribut tout en lui attribuant un score de contraintes. En fonction de l'utilisation ou non des contraintes, ces méthodes de sélection sont réparties en trois catégories [167]: *non supervisées*, *supervisées*, et *semi-supervisées*. Dans ce qui suit, nous passons en revue ces différentes méthodes.

3.6.1 Sélection non supervisée

Les méthodes de sélection d'attributs non supervisée utilisent uniquement les données non étiquetées (non contraintes) pour attribuer un score de pertinence à chaque attribut.

3.6.1.1 Score de la variance

Le score de la variance (Var) est simple et rapide à estimer. Il évalue la pertinence d'un attribut en calculant sa variance. Le score Var_r d'un attribut f_r est défini comme suit:

$$\text{Var}_r = \frac{1}{n}\sum_{i=1}^{n}(x_{ir} - \mu_r)^2 \tag{3.8}$$

où μ_r est la moyenne de l'ensemble de données sur l'attribut f_r telle que:

$$\mu_r = \frac{\sum_{i=1}^{n} x_{ir}}{n} \tag{3.9}$$

En théorie spectrale des graphes, la variance pondérée d'un attribut f_r est définie par [188]:

$$\text{Var}_r = \sum_{i=1}^{n}(x_{ir} - \bar{f}_r)^2 d_{ii} \tag{3.10}$$

où $\bar{f}_r$ est la moyenne pondérée de l'ensemble de données sur l'attribut f_r. Elle est définie par:

$$\bar{f}_r = \frac{\sum_{i=1}^{n} x_{ir} d_{ii}}{\sum_{i=1}^{n} d_{ii}} = \frac{\boldsymbol{f}_r^T \boldsymbol{D} \mathbf{1}}{\mathbf{1}^T \boldsymbol{D} \mathbf{1}} \tag{3.11}$$

où $\boldsymbol{D}$ est la matrice des degrés et $\mathbf{1} = [1, \ldots, 1]^T$ un vecteur dont les éléments sont des 1. En notant $\tilde{x}_{ir} = x_{ir} - \bar{f}_r$, l'équation (3.10) peut être réécrite comme suit:

$$\text{Var}_r = \sum_{i=1}^{n} \tilde{x}_{ir}^2 d_{ii} = \tilde{\boldsymbol{f}}_r^T \boldsymbol{D} \tilde{\boldsymbol{f}}_r \tag{3.12}$$

Les attributs ayant les scores Var_r les plus élevés sont considérés comme les plus pertinents.

3.6.1.2 Score Laplacien

He et al. [188] ont proposé une méthode de sélection d'attributs, notée LS, qui évalue la pertinence d'un attribut en se basant sur sa capacité à préserver la structure géométrique de l'ensemble original des données, c.-à-d. les données qui sont proches dans l'espace original des attributs, doivent aussi être proches dans l'espace de l'attribut à examiner. En théorie des graphes, ce principe est formalisé en représentant l'ensemble de données X par un graphe $G = (V,E,\boldsymbol{W})$ dont la matrice de similarité $\boldsymbol{W}$ est calculée avec l'équation (2.8). Le score Laplacien LS_r d'un attribut f_r est défini par:

$$LS_r = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} (x_{ir} - x_{jr})^2 w_{ij}}{\text{Var}(\boldsymbol{f}_r)} \tag{3.13}$$

En suivant la même démonstration que celle de l'équation (2.19) du chapitre précédent, le numérateur de l'équation (3.13) peut être réécrit comme suit:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} (x_{ir} - x_{jr})^2 w_{ij} = \boldsymbol{f}_r^T \boldsymbol{L} \boldsymbol{f}_r \tag{3.14}$$

En remplaçant le numérateur et le dénominateur de l'équation (3.13) par les équations (3.14) et (3.12) respectives, LS_r s'écrit sous la forme vectorielle suivante:

$$LS_r = \frac{\boldsymbol{f}_r^T \boldsymbol{L} \boldsymbol{f}_r}{\tilde{\boldsymbol{f}}_r^T \boldsymbol{D} \tilde{\boldsymbol{f}}_r} = \frac{\tilde{\boldsymbol{f}}_r^T \boldsymbol{L} \tilde{\boldsymbol{f}}_r}{\tilde{\boldsymbol{f}}_r^T \boldsymbol{D} \tilde{\boldsymbol{f}}_r} \tag{3.15}$$

avec

$$\tilde{\boldsymbol{f}}_r = \boldsymbol{f}_r - \frac{\boldsymbol{f}_r^T \boldsymbol{D} \mathbf{1}}{\mathbf{1}^T \boldsymbol{D} \mathbf{1}} \mathbf{1} \tag{3.16}$$

En substituant l'équation (3.16) dans l'équation (3.15), on peut facilement démontrer que $\boldsymbol{f}_r^T \boldsymbol{L} \boldsymbol{f}_r = \tilde{\boldsymbol{f}}_r^T \boldsymbol{L} \tilde{\boldsymbol{f}}_r$. En effet:

$$\begin{aligned} \tilde{\boldsymbol{f}}_r^T \boldsymbol{L} \tilde{\boldsymbol{f}}_r &= \left(\boldsymbol{f}_r - \frac{\boldsymbol{f}_r^T \boldsymbol{D1}}{\boldsymbol{1}^T \boldsymbol{D1}} \boldsymbol{1}\right)^T \boldsymbol{L} \left(\boldsymbol{f}_r - \frac{\boldsymbol{f}_r^T \boldsymbol{D1}}{\boldsymbol{1}^T \boldsymbol{D1}} \boldsymbol{1}\right) \\ &= (\boldsymbol{f}_r^T \boldsymbol{L} \boldsymbol{f}_r) - \left(\left(\frac{\boldsymbol{f}_r^T \boldsymbol{D1}}{\boldsymbol{1}^T \boldsymbol{D1}} \boldsymbol{1}\right)^T \boldsymbol{L} \boldsymbol{f}_r\right) - \left(\boldsymbol{f}_r^T \boldsymbol{L} \left(\frac{\boldsymbol{f}_r^T \boldsymbol{D1}}{\boldsymbol{1}^T \boldsymbol{D1}} \boldsymbol{1}\right)\right) + \left(\left(\frac{\boldsymbol{f}_r^T \boldsymbol{D1}}{\boldsymbol{1}^T \boldsymbol{D1}} \boldsymbol{1}\right)^T \boldsymbol{L} \left(\frac{\boldsymbol{f}_r^T \boldsymbol{D1}}{\boldsymbol{1}^T \boldsymbol{D1}} \boldsymbol{1}\right)\right) \end{aligned}$$

En posant $z = \frac{\boldsymbol{f}_r^T \boldsymbol{D1}}{\boldsymbol{1}^T \boldsymbol{D1}}$, qui est une valeur réelle, on obtient:

$$\tilde{\boldsymbol{f}}_r^T \boldsymbol{L} \tilde{\boldsymbol{f}}_r = \boldsymbol{f}_r^T \boldsymbol{L} \boldsymbol{f}_r - z \boldsymbol{1}^T \boldsymbol{L} \boldsymbol{f}_r - \boldsymbol{f}_r^T \boldsymbol{L} z \boldsymbol{1} + z \boldsymbol{1}^T \boldsymbol{L} z \boldsymbol{1} \tag{3.17}$$

Comme $\boldsymbol{1}^T \boldsymbol{L} = \boldsymbol{L} \boldsymbol{1} = \boldsymbol{0}$ (avec $\boldsymbol{0} = [0, \ldots, 0]^T$), l'équation ci-dessus devient $\tilde{\boldsymbol{f}}_r^T \boldsymbol{L} \tilde{\boldsymbol{f}}_r = \boldsymbol{f}_r^T \boldsymbol{L} \boldsymbol{f}_r$.

Les attributs ayant les scores Laplaciens LS_r les plus bas sont les plus pertinents.

3.6.2 Sélection supervisée sous contraintes

Les méthodes de sélection d'attributs supervisée sous contraintes exploitent uniquement les contraintes must-link et cannot-link pour sélectionner les attributs.

3.6.2.1 Scores de contraintes supervisés C^1 et C^2

Zhang et al. [193] ont proposé deux scores de contraintes en se basant uniquement sur les ensembles de contraintes must-link M et cannot-link C. Ces deux scores, notés respectivement C^1 et C^2, évaluent la capacité des attributs à garantir le respect des contraintes M et C. Les scores C_r^1 et C_r^2 d'un attribut f_r sont définis par:

$$C_r^1 = \frac{\sum_{(x_i, x_j) \in M} (x_{ir} - x_{jr})^2}{\sum_{(x_i, x_j) \in C} (x_{ir} - x_{jr})^2} \tag{3.18}$$

$$C_r^2 = \sum_{(x_i, x_j) \in M} (x_{ir} - x_{jr})^2 - \lambda \sum_{(x_i, x_j) \in C} (x_{ir} - x_{jr})^2 \tag{3.19}$$

où $\lambda \in [0,1]$ est un paramètre de régularisation.

Concrètement, ces deux scores permettent de sélectionner les attributs qui minimisent les distances entre les données en must-link et qui maximisent les distances entre les données en cannot-link. Ils peuvent être aussi définis en utilisant la théorie des graphes. Dans ce cadre la, les ensembles de contraintes M et C sont représentés par deux graphes de similarité:

- *Graphe des must-link* G^M: deux nœuds v_i et v_j sont liés dans G^M si $(x_i,x_j) \in M$.
- *Graphe des cannot-link* G^C: deux nœuds v_i et v_j sont liés dans G^C si $(x_i,x_j) \in C$.

Les matrices de similarité $\boldsymbol{W}^M \in \mathbb{R}^{n\times n}$ et $\boldsymbol{W}^C \in \mathbb{R}^{n\times n}$, liées respectivement aux graphes G^M et G^C, sont définies comme suit:

$$w_{ij}^M = \begin{cases} 1 & \text{si } (x_i,x_j) \in M \\ 0 & \text{sinon} \end{cases} \tag{3.20}$$

$$w_{ij}^C = \begin{cases} 1 & \text{si } (x_i,x_j) \in C \\ 0 & \text{sinon} \end{cases} \tag{3.21}$$

En intégrant les matrices $\boldsymbol{W}^M$ et $\boldsymbol{W}^C$ dans les équations (3.18) et (3.19), on obtient:

$$C_r^1 = \frac{\sum_{i=1}^n \sum_{j=1}^n (x_{ir} - x_{jr})^2 w_{ij}^M}{\sum_{i=1}^n \sum_{j=1}^n (x_{ir} - x_{jr})^2 w_{ij}^C} \tag{3.22}$$

$$C_r^2 = \sum_{i=1}^n \sum_{j=1}^n (x_{ir} - x_{jr})^2 w_{ij}^M - \lambda \sum_{i=1}^n \sum_{j=1}^n (x_{ir} - x_{jr})^2 w_{ij}^C \tag{3.23}$$

En suivant la même démarche que celle de l'équation (2.19), les scores C_r^1 et C_r^2 peuvent être réécrits sous la forme vectorielle comme suit:

$$C_r^1 = \frac{\boldsymbol{f}_r^T \boldsymbol{L}^M \boldsymbol{f}_r}{\boldsymbol{f}_r^T \boldsymbol{L}^C \boldsymbol{f}_r} \tag{3.24}$$

$$C_r^2 = \boldsymbol{f}_r^T \boldsymbol{L}^M \boldsymbol{f}_r - \lambda \boldsymbol{f}_r^T \boldsymbol{L}^C \boldsymbol{f}_r \tag{3.25}$$

$\boldsymbol{L}^M = \boldsymbol{D}^M - \boldsymbol{W}^M$ et $\boldsymbol{L}^C = \boldsymbol{D}^C - \boldsymbol{W}^C$ sont respectivement les matrices laplaciennes non normalisées des graphes G^M et G^C. $\boldsymbol{D}^M$ et $\boldsymbol{D}^C$ sont respectivement la matrice des degrés du graphe des must-link définie par: $d_{ii}^M = \sum_{j=1}^n w_{ij}^M$ et la matrice des degrés du graphe des cannot-link définie par: $d_{ii}^C = \sum_{j=1}^n w_{ij}^C$.

Les attributs ayant les scores de contraintes C^1 ou C^2 les plus bas sont les plus pertinents.

Notons que les scores C^1 et C^2 sont très dépendants des ensembles de contraintes must-link et cannot-link. Par conséquent, lorsque l'ensemble de contraintes est modifié, les scores d'attributs changent aussi. De plus, la sélection d'attributs est effectuée avec de très petits ensembles de données contraintes et ignore les informations qui peuvent être apportées par l'ensemble de données non contraintes.

3.6.3 Sélection semi-supervisée sous contraintes

Les méthodes de sélection d'attributs semi-supervisée avec contraintes utilisent à la fois les données contraintes et les données non contraintes pour sélectionner les attributs. L'utilisation des données non contraintes a pour but de représenter au mieux la structure originale des données et rendre le processus de sélection d'attributs moins sensible aux contraintes.

3.6.3.1 Score de contraintes semi-supervisé C^3

Zhao et al. [195] ont proposé un score de contraintes semi-supervisé, nommé "locality sensitive discriminant analysis score". Ce score, noté C^3, évalue la pertinence d'un attribut par sa capacité à préserver la structure originale des données et à garantir le respect des contraintes must-link et cannot-link disponibles.

D'un point de vue théorie des graphes, un nouveau graphe de similarité G^{KNN1} qui représente à la fois les données non contraintes et les données contraintes par des must-link est construit. Dans le graphe G^{KNN1}, deux nœuds v_i et v_j sont liés si $(x_i,x_j) \in M$ ou si x_i et x_j ne sont pas des données contraintes, mais qui sont voisines dans l'espace original des attributs. La matrice de similarité $\boldsymbol{W}^{KNN1} \in \mathbb{R}^{n\times n}$ correspondante au graphe G^{KNN1} est définie par:

$$w_{ij}^{KNN1} = \begin{cases} \gamma & \text{si } (x_i,x_j) \in M \text{ ou } (x_j,x_i) \in M \\ 1 & \text{si } x_i \text{ ou } x_j \text{ sont des données non contraintes et } x_i \in KNN(x_j) \text{ ou } x_j \in KNN(x_i) \\ 0 & \text{sinon} \end{cases} \tag{3.26}$$

où γ est une constante et KNN l'ensemble des K plus proches voisins de x_i ou de x_j. Dans nos expérimentations, γ est fixé à 100 et K est fixé à 5 comme dans [195].

Le score C_r^3 combine la matrice de similarité $\boldsymbol{W}^c$ (équation (3.21)), construite à partir du graphe des cannot-link G^C, et la matrice de similarité $\boldsymbol{W}^{KNN1}$ (équation (3.26)), construite à partir du graphe G^{KNN1}. Le score C_r^3 d'un attribut f_r est donné par:

$$C_r^3 = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(x_{ir}-x_{jr})^2 w_{ij}^{KNN1}}{\sum_{i=1}^{n}\sum_{j=1}^{n}(x_{ir}-x_{jr})^2 w_{ij}^{C}} \tag{3.27}$$

Le score C_r^3 de l'équation (3.27) peut être aussi réécrit sous la forme vectorielle comme suit:

$$C_r^3 = \frac{\boldsymbol{f}_r^T \boldsymbol{L}^{KNN1} \boldsymbol{f}_r}{\boldsymbol{f}_r^T \boldsymbol{L}^{C} \boldsymbol{f}_r} \tag{3.28}$$

$\boldsymbol{L}^{KNN1} = \boldsymbol{D}^{KNN1} - \boldsymbol{W}^{KNN1}$ est la matrice laplacienne non normalisée de $\boldsymbol{W}^{KNN1}$ et $\boldsymbol{D}^{KNN1}$ sa matrice diagonale des degrés.

Ce score garantit que les paires de données reliées par des contraintes cannot-link soient bien séparées. Il tient compte aussi des données non contraintes, mais favorise les paires de contraintes must-link en leurs assignant des similarités élevées ($\gamma = 100$) dans la matrice $\boldsymbol{W}^{KNN1}$, alors que les similarités entre les paires de données non contraintes sont mises à 1.

3.6.3.2 Score de contraintes semi-supervisé C^4

Kalakech et al. [196] ont proposé un score de contraintes semi-supervisé, que nous avons noté C^4, capable de préserver la structure originale des données et de rendre le processus de sélection d'attributs moins sensible aux ensembles de contraintes. Le score C_r^4 de l'attribut f_r est le résultat du produit entre le score Laplacien non supervisé LS_r (équation (3.15)) et le score de contraintes supervisé C_r^1 (équation (3.24)) tel que:

$$C_r^4 = LS_r \cdot C_r^1 \tag{3.29}$$

Le score Laplacien LS favorise les attributs qui préservent la structure originale des données tandis que le score C^1 cherche les attributs qui garantissent le respect des contraintes.

Le score C_r^4 d'un attribut f_r peut être réécrit sous la forme vectorielle comme suit:

$$C_r^4 = \frac{\tilde{\boldsymbol{f}}_r^T \boldsymbol{L} \tilde{\boldsymbol{f}}_r}{\tilde{\boldsymbol{f}}_r^T \boldsymbol{D} \tilde{\boldsymbol{f}}_r} \cdot \frac{\boldsymbol{f}_r^T \boldsymbol{L}^M \boldsymbol{f}_r}{\boldsymbol{f}_r^T \boldsymbol{L}^C \boldsymbol{f}_r} \tag{3.30}$$

3.6.3.3 Score de contraintes semi-supervisé C^5

Benabdeslem et Hindawi [197] ont proposé un score de contraintes semi-supervisé, nommé "constrained laplacian score", et que nous avons noté C^5. Pour cela, ils proposent deux graphes de similarité: le graphe des cannot-link G^C et un nouveau graphe de similarité G^{KNN2} dans lequel deux nœuds v_i et v_j sont liés si $(x_i,x_j) \in M$, ou si x_i et x_j sont voisins dans l'espace original des attributs. La matrice de similarité $\boldsymbol{W}^{KNN2} \in \mathbb{R}^{n \times n}$ correspondante au graphe G^{KNN2} est définie comme suit:

$$w_{ij}^{KNN2} = \begin{cases} w_{ij} & \text{si } ((x_i,x_j) \in M) \text{ ou si } (x_i \in KNN(x_j) \text{ ou } x_j \in KNN(x_i)) \\ 0 & \text{sinon} \end{cases} \tag{3.31}$$

Le score de contraintes C_r^5 cherche les attributs f_r qui respectent la structure des graphes G^{KNN2} et G^C. Il est défini par:

$$C_r^5 = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(x_{ir}-x_{jr})^2 w_{ij}^{KNN2}}{\sum_{i=1}^{n}\sum_{j=1|\exists l,(x_l,x_j)\in C}^{n}(x_{ir}-\alpha_{jr}^{i})^2 d_{ii}^{KNN2}} \tag{3.32}$$

où

$$\alpha_{jr}^{i} = \begin{cases} x_{jr} & \text{si } (x_i,x_j) \in C \\ \bar{f}_r & \text{si } \left(i=j\right) \text{ et } \left(x_i \in X^U\right) \\ x_{ir} & \text{sinon} \end{cases} \tag{3.33}$$

Le score C^5 est une version améliorée du score Laplacien LS et du score C^1. En absence de données étiquetées ($X^P = \emptyset$ et $X = X^U$), $\alpha_{jr}^{i} = \bar{f}_r$ et le score C^5 est équivalent au score LS_r. En présence de données étiquetées ($X^U = \emptyset$ et $X = X^P$), $\alpha_{jr}^{i} = x_{jr}$, et le score C^5 devient identique au score de contraintes C^1.

Le score C_r^5 d'un attribut f_r peut être réécrit sous la forme vectorielle comme suit:

$$C_r^5 = \frac{\boldsymbol{f}_r^T \boldsymbol{L}^{KNN2} \boldsymbol{f}_r}{\boldsymbol{f}_r^T \boldsymbol{L}^{C} \boldsymbol{D}^{KNN2} \boldsymbol{f}_r} \tag{3.34}$$

$\boldsymbol{L}^{KNN2} = \boldsymbol{D}^{KNN2} - \boldsymbol{W}^{KNN2}$ étant la matrice laplacienne non normalisée de $\boldsymbol{W}^{KNN2}$ et $\boldsymbol{D}^{KNN2}$ sa matrice diagonale des degrés.

3.6.3.4 Score de contraintes semi-supervisé C^6

Benabdeslem et Hindawi [198] ont proposé un autre score semi-supervisé, nommé "constrained selection based feature selection", et qui est une version améliorée du score C^5. La motivation derrière ce score, noté ici C^6, est que certaines contraintes (dites incohérentes et non informatives) peuvent avoir un effet négatif sur les performances de la classification même si elles sont générées à partir des étiquettes de classes [154].

Le score C_r^6 s'appuie sur l'utilisation des ensembles de contraintes cohérentes M' et C' sélectionnés à partir des ensembles de contraintes M et C. La cohérence peut être interprétée géométriquement à partir des projections des vecteurs associés aux paires de contraintes [154]. Une contrainte must-link ou cannot-link est sélectionnée respectivement dans M' ou dans C' si elle est totalement cohérente avec les autres contraintes, c.-à-d. la projection du vecteur correspondant à la contrainte must-link ou cannot-link sur les vecteurs de toutes les contraintes dans M ou C possède un chevauchement nul (Figure 3.4).

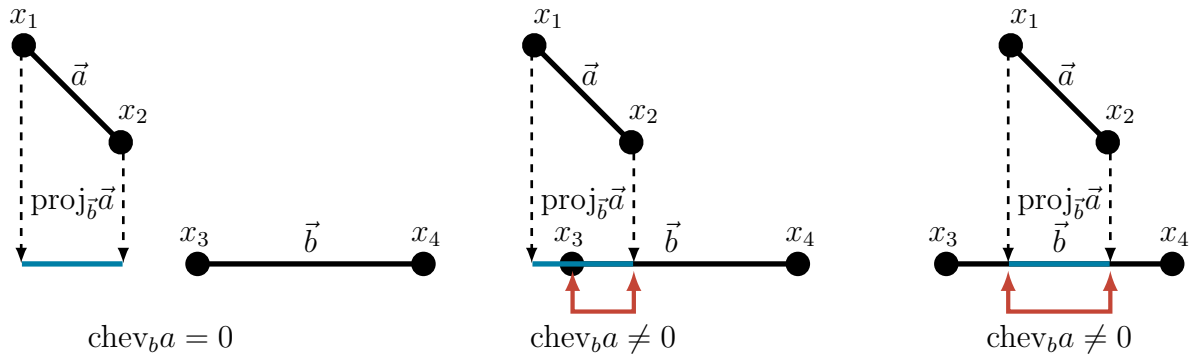


Figure 3.4: Illustration du chevauchement entre deux contraintes a et b ($\text{proj}_{\vec{b}}\vec{a}$ est projection du vecteur $\vec{a}$ sur $\vec{b}$, et $\text{chev}_b a$ le chevauchement de $\text{proj}_{\vec{b}}\vec{a}$ avec $\vec{b}$).

Le score de contraintes C_r^6 d'un attribut f_r est exprimé comme suit:

$$C_r^6 = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(x_{ir}-x_{jr})^2(w_{ij}^{KNN}+\mathcal{N}_{ij}^{KNN})}{\sum_{i=1}^{n}\sum_{j=1}^{n}(x_{ir}-\alpha_{jr}^{i})^2 d_{ii}^{KNN}} \tag{3.35}$$

où α_{jr}^{i} est défini dans l'équation (3.33) et $\boldsymbol{W}^{KNN}$ la matrice de similarité du graphe G^{KNN} définie par:

$$w_{ij}^{KNN} = \begin{cases} w_{ij} & \text{si } x_i \in KNN(x_j) \text{ ou } x_j \in KNN(x_i) \\ 0 & \text{sinon} \end{cases} \tag{3.36}$$

et $\boldsymbol{\mathcal{N}}^{KNN}$ une autre matrice de similarité définie par:

$$\mathcal{N}_{ij}^{KNN} = \begin{cases} -w_{ij} & \text{si } (x_i \in KNN(x_j) \text{ ou } x_j \in KNN(x_i)) \text{ et } (x_i,x_j) \in M' \\ w_{ij}^2 & \text{si } (x_i \in KNN(x_j) \text{ ou } x_j \in KNN(x_i)) \text{ et } (x_i,x_j) \in C' \text{ ou} \\ & \text{si } (x_i \notin KNN(x_j) \text{ et } x_j \notin KNN(x_i)) \text{ et } (x_i,x_j) \in M' \\ 0 & \text{sinon} \end{cases} \tag{3.37}$$

Le score C^6 s'appuie sur deux graphes de similarité G^{KNN3} et G^C qui sont pondérés respectivement par les matrices $(\boldsymbol{W}^{KNN}+\boldsymbol{\mathcal{N}}^{KNN})$ et $\boldsymbol{W}^C$ (équation (3.21)). La matrice $\boldsymbol{W}^{KNN3}$ résultante de la combinaison de $\boldsymbol{W}^{KNN}$ et $\boldsymbol{\mathcal{N}}^{KNN}$ est définie par:

$$w_{ij}^{KNN3} = \begin{cases} w_{ij}^2 + w_{ij} & \text{si } \big((x_i,x_j) \in C'\big) \text{ et } \big(x_i \in KNN(x_j) \text{ ou } x_j \in KNN(x_i)\big) \\ w_{ij}^2 & \text{si } \big((x_i,x_j) \in M'\big) \text{ et } \big(x_i \notin KNN(x_j) \text{ et } x_j \notin KNN(x_i)\big) \\ w_{ij} & \text{si } \big(x_i \in X^U \text{ ou } x_j \in X^U\big) \text{ et } \big(x_i \in KNN(x_j) \text{ ou } x_j \in KNN(x_i)\big) \\ 0 & \text{sinon} \end{cases} \tag{3.38}$$

Le score C_r^6 peut être alors réécrit sous la forme suivante:

$$C_r^6 = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(x_{ir}-x_{jr})^2 w_{ij}^{KNN3}}{\sum_{i=1}^{n}\sum_{j=1}^{n}(x_{ir}-\alpha_{jr}^{i})^2 d_{ii}^{KNN}} \tag{3.39}$$

ou encore sous la forme vectorielle suivante:

$$C_r^6 = \frac{\boldsymbol{f}_r^T \boldsymbol{L}^{KNN3} \boldsymbol{f}_r}{\boldsymbol{f}_r^T \boldsymbol{L}^C \boldsymbol{D}^{KNN} \boldsymbol{f}_r} \tag{3.40}$$

$\boldsymbol{L}^{KNN3} = \boldsymbol{D}^{KNN3} - \boldsymbol{W}^{KNN3}$ étant la matrice laplacienne non normalisée de $\boldsymbol{W}^{KNN3}$. $\boldsymbol{D}^{KNN}$ et $\boldsymbol{D}^{KNN3}$ sont respectivement les matrices des degrés issues de $\boldsymbol{W}^{KNN}$ et de $\boldsymbol{W}^{KNN3}$.

À partir de l'équation (3.38), nous pouvons faire quelques remarques sur le score C^6:

- Lorsque deux données sont voisines et liées par une contrainte must-link, la similarité entre elles est nulle. Cela signifie que le score C^6 ne tient pas compte de ces données dans la construction du graphe, alors que ces données sont considérées comme très fiables.
- Lorsque deux données sont liées par une contrainte must-link et ne sont pas voisines, la similarité est augmentée de w_{ij}^2, alors qu'en principe ces données ne sont pas voisines $w_{ij} \approx 0$ et par conséquent $w_{ij}^2 \approx 0$.
- Lorsque deux données sont voisines et liées par une contrainte cannot-link, la similarité est augmentée de w_{ij}^2, alors qu'en principe elle doit être diminuée.
- Lorsqu'un poids supplémentaire w_{ij}^2 est ajouté à la valeur de similarité peut être supérieur à 1 ce qui est contradictoire à la propriété de normalisation $w_{i,j} \in [0,1]$.

3.6.3.5 Score de contraintes semi-supervisé C^7

Yang et al. [199, 200] ont récemment proposé un nouveau score de contraintes semi-supervisé, nommé "constraint compensated laplacian score". Ce score exploite à la fois l'information apportée par des données non étiquetées X^U, des données étiquetées X^P, et des contraintes must-link M et cannot-link C, qui peuvent être déduites ou non à partir des données étiquetées X^P. Pour un attribut f_r, ce score, noté C_r^7, s'écrit:

$$C_r^7 = \frac{\sum_{i=1}^n \sum_{j=1}^n (x_{ir} - x_{jr})^2 (w_{ij}^{KNN} + \bar{\mathcal{N}}_{ij})}{S_r + S_r^b - S_r^w} \tag{3.41}$$

w_{ij}^{KNN} est donné par l'équation (3.36) et $\bar{\mathcal{N}}_{ij}$ est définie comme suit:

$$\bar{\mathcal{N}}_{ij} = \begin{cases} 1 - w_{ij} & \text{si } \big((x_i,x_j) \in M\big) \text{ et } \big(x_i \in KNN(x_j) \text{ ou } x_j \in KNN(x_i)\big) \\ \lambda & \text{si } \big((x_i,x_j) \in M\big) \text{ et } \big(x_i \notin KNN(x_j) \text{ et } x_j \notin KNN(x_i)\big) \\ -\gamma w_{ij} & \text{si } \big((x_i,x_j) \in C\big) \text{ et } \big(x_i \in KNN(x_j) \text{ ou } x_j \in KNN(x_i)\big) \\ 0 & \text{sinon} \end{cases} \tag{3.42}$$

γ et λ sont deux paramètres fixés empiriquement à 0.9 pour γ et 0.5 pour λ [199, 200].

S_r est la variance du r-ème attribut calculée sur tout l'ensemble de données $X = X^U \cup X^P$. S_r^b et S_r^w sont respectivement, la variance intra-classe et la variance inter-classes du r-ème attribut, calculées sur l'ensemble de données étiquetées X^P.

$$S_r = \sum_{i=1}^{n}(x_{ir} - \bar{f}_r)^2 d_{ii} \tag{3.43}$$

$$S_r^b = \sum_{l=1}^{k} |X^l|(\bar{f}_r^{(l)} - \bar{f}_r^P)^2 \tag{3.44}$$

$$S_r^w = \sum_{l=1}^{k} |X^l|(\sigma_r^{(l)})^2 \tag{3.45}$$

Notons que $|X^l| = p$ est le nombre de prototypes de la l-ème classe, $\bar{f}_r^P = \sum_{i=1|x_i \in X^P}^{n} \frac{x_{ir}}{k.p}$ est la moyenne du r-ème attribut calculée sur l'ensemble de données étiquetées X^P, $\bar{f}_r^{(l)} = \sum_{i=1|x_i \in X^l}^{n} \frac{x_{ir}}{p}$ et $\sigma_r^{(l)}$ sont respectivement la moyenne et la variance du r-ème attribut calculées sur les données étiquetées de la l-ème classe.

En combinant les matrices $\boldsymbol{W}^{KNN}$ et $\bar{\boldsymbol{\mathcal{N}}}$, on obtient la matrice $\boldsymbol{W}^{KNN4}$ suivante:

$$w_{ij}^{KNN4} = \begin{cases} 1 & \text{si } \big((x_i,x_j) \in M\big) \text{ et } \big(x_i \in KNN(x_j) \text{ ou } x_j \in KNN(x_i)\big) \\ \lambda & \text{si } \big((x_i,x_j) \in M\big) \text{ et } \big(x_i \notin KNN(x_j) \text{ et } x_j \notin KNN(x_i)\big) \\ (1-\gamma)w_{ij} & \text{si } \big((x_i,x_j) \in C\big) \text{ et } \big(x_i \in KNN(x_j) \text{ ou } x_j \in KNN(x_i)\big) \\ w_{ij} & \text{si } \big(x_i \in X^U \text{ ou } x_j \in X^U\big) \text{ et } \big(x_i \in KNN(x_j) \text{ ou } x_j \in KNN(x_i)\big) \\ 0 & \text{sinon} \end{cases} \tag{3.46}$$

En termes vectoriels, le score C_r^7 de l'équation (3.27) prend la forme suivante [200]:

$$C_r^7 = \frac{2(\boldsymbol{f}_r)^T \boldsymbol{L}^{KNN4} \boldsymbol{f}_r}{(\tilde{\boldsymbol{f}}_r)^T \boldsymbol{D}^{KNN4} \tilde{\boldsymbol{f}}_r + 2(\tilde{\boldsymbol{f}}_r^P)^T \boldsymbol{L}^P \tilde{\boldsymbol{f}}_r^P - (\tilde{\boldsymbol{f}}_r^P)^T \boldsymbol{D}^P \tilde{\boldsymbol{f}}_r^P} \tag{3.47}$$

où $\tilde{\boldsymbol{f}}_r^P = \boldsymbol{f}_r - \bar{f}_r^P \mathbf{1} = \boldsymbol{f}_r - \frac{\boldsymbol{f}_r^T \boldsymbol{D^P} \mathbf{1}}{\mathbf{1}^T \boldsymbol{D^P} \mathbf{1}} \mathbf{1}$. $\boldsymbol{L}^P = \boldsymbol{D}^P - \boldsymbol{W}^P$ et $\boldsymbol{L}^{KNN4} = \boldsymbol{D}^{KNN4} - \boldsymbol{W}^{KNN4}$ sont respectivement les matrices laplaciennes des matrices $\boldsymbol{W}^P$ et $\boldsymbol{W}^{KNN4}$. $\boldsymbol{D}^P$ et $\boldsymbol{D}^{KNN4}$ sont les matrices des degrés déduites respectivement de $\boldsymbol{W}^P$ et $\boldsymbol{W}^{KNN4}$. La matrice $\boldsymbol{W}^P$ est définie par:

$$w_{ij}^P = \begin{cases} 1/|X^l| & \text{si } x_i \in X^l \text{ and } x_j \in X^l \\ 0 & \text{sinon} \end{cases} \tag{3.48}$$

La sélection semi-supervisée à base des scores de contraintes cités précédemment est de type "feature ranking". Elle consiste à déterminer les scores de contraintes C_r^3, C_r^4, C_r^5, C_r^6, et C_r^7 de chaque attribut f_r, puis trier les attributs dans l'ordre décroissant. Les m premiers attributs sont alors sélectionnés.

De plus, il est utile de rappeler que tous ces scores de contraintes sont basés sur les matrices laplaciennes et sur les matrices des degrés déduites à partir des matrices de similarité calculées dans l'espace original des attributs. Ils peuvent, par conséquent, être atteints par la malédiction de la dimension.

3.6.3.6 Méthode de sélection semi-supervisée EnsCLS

Benabdeslem et al. [201] ont proposé une méthode de sélection d'attributs semi-supervisée, appelée "ensemble constrained laplacian score" (EnsCLS). Cette méthode se base principalement sur le score de contraintes semi-supervisé C^5 (équation (3.34)), dont l'inconvénient majeur est sa sensibilité au choix des contraintes. Pour résoudre ce problème, EnsCLS varie les sources de contraintes en combinant à la fois la méthode de ré-échantillonnage des données (Bagging) [202] et la méthode des sous-espaces aléatoires, nommée "random subspace method" (RSM) [203]. La méthode Bagging permet d'estimer la pertinence des attributs par rapport à de petits changements dans les contraintes tandis que la méthode RSM permet de réduire l'effet de la malédiction de la dimension en traitant de petits sous-ensembles d'attributs. Les principales étapes de cette méthode sont décrites dans l'algorithme 3.2.

Algorithme 3.2: EnsCLS.

Entrée: Ensemble de données étiquetés X^P, ensemble de données non étiquetés X^U, ensemble de d attributs $F_d = \{f_1, \ldots, f_r, \ldots, f_d\}$.

1. Initialiser les scores $I(f_r)$ et les occurrences $O(f_r)$ à 0 pour chaque attribut f_r.
2. Fixer la taille ℓ du comité à $\ell = 10 \times \text{ceil}\big(\log(0.01)/\log(1 - 1/\sqrt{d})\big)$.
3. Pour i=1: ℓ
   (a) Construire l'ensemble d'attributs $\mathcal{A}_i$ à partir de F_d en utilisant la méthode RSM.
   (b) Générer l'ensemble $X^{P,B}$ de données étiquetées ré-échantillonnées à partir de X^P en utilisant la méthode Bagging puis projeter le dans $\mathcal{A}_i$ pour obtenir $X_i^{P,B}$.
   (c) Construire les ensembles de contraintes à partir de l'ensemble $X_i^{P,B}$.
   (d) Construire l'ensemble X_i^U en projetant X^U sur l'espace des attributs $\mathcal{A}_i$.
   (e) Pour chaque attribut $f_r \in \mathcal{A}_i$
      i. Calculer le score semi-supervisé C_r^5.
      ii. Mettre $I(f_r) = I(f_r) + C_r^5$ et $O(f_r) = O(f_r) + 1$.
4. Calculer $I(f_r) = \frac{I(f_r)}{O(f_r)}$ pour chaque attribut $f_r \in F_d$ $(r = 1, \ldots, d)$.
5. Classer les attributs dans F_d selon l'ordre croissant de leurs scores I et sélectionner les m premiers attributs.

Sortie: Sous-ensemble F_m des m attributs sélectionnés.

3.6.3.7 Méthode de sélection semi-supervisée SCGS

Liu et Zhang [204] ont proposé une méthode de sélection d'attributs semi-supervisée de type filtre, nommée "semi-supervised constrained guided sparse" (SCGS). Pour sélectionner les attributs les plus pertinents, cette méthode utilise à la fois des données non étiquetées X^U, des données étiquetées X^P, et des contraintes de type must-link M et cannot-link C. La méthode SCGS consiste à déterminer le vecteur poids $\boldsymbol{w}$ qui minimise la fonction objective suivante:

$$\min_{\boldsymbol{w}} \frac{1}{2}||\boldsymbol{A}(\boldsymbol{\Omega} - \boldsymbol{X}\boldsymbol{w})||_2^2 + \lambda_1 \boldsymbol{w}^T \boldsymbol{X}^T (\boldsymbol{L}^M - \alpha \boldsymbol{L}^C) \boldsymbol{X}\boldsymbol{w} + \lambda_2 ||\boldsymbol{w}||_1 + \lambda_3 \boldsymbol{w}^T \boldsymbol{X}^T \boldsymbol{L} \boldsymbol{X} \boldsymbol{w} \quad (3.49)$$

où λ_1,λ_2,λ_3, et α sont des paramètres de régularisation. $\boldsymbol{L}^M$, $\boldsymbol{L}^C$, et $\boldsymbol{L}$ sont les matrices laplaciennes non normalisées définies respectivement sur l'ensemble des contraintes must-link M, cannot-link C, et l'ensemble des données $X = X^U \cup X^P$. $\boldsymbol{\Omega}$ étant le vecteur d'étiquettes. $\boldsymbol{A}$ est une matrice indicatrice des étiquettes de classe des données disponibles. Elle est diagonale de taille $n \times n$ dont la valeur $a_{ii} = 1$ si l'étiquette de classe de la donnée x_i est connue, $a_{ii} = 0$ sinon.

Dans l'équation (3.49), le premier terme représente la perte empirique sur les données étiquetées, le second est un terme de régularisation basé sur la structure locale des données représentée par les contraintes must-link et cannot-link, et le dernier terme représente l'estimation non supervisée de la structure géométrique des données originales.

Les principales étapes de la méthode SCGS sont décrites dans l'algorithme 3.3.

Algorithme 3.3: SCGS.

Entrée: Ensemble de données $X = \{x_1,x_2,\ldots,x_n\}$, ensemble des étiquettes de classes des données $\Omega = \{\omega(x_1),\omega(x_2),\ldots,\omega(x_n)\}$, ensemble des contraintes must-link M et cannot-link C.

1. Choisir les paramètres λ_1,λ_2,λ_3, et α.
2. Calculer les matrices de similarité $\boldsymbol{W}$, $\boldsymbol{W}^M$, et $\boldsymbol{W}^C$ respectivement avec les équations (2.8), (3.20), et (3.21).
3. Calculer les matrices laplaciennes non normalisées $\boldsymbol{L}$, $\boldsymbol{L}^M$, et $\boldsymbol{L}^C$.
4. Déterminer la solution optimale $\boldsymbol{w}$ de l'équation (3.49).
5. Sélectionner les m attributs dont les poids w_r, $r = 1,\ldots,d$, sont non nuls.

Sortie: Sous-ensemble F_m des m attributs sélectionnés.

3.7 Conclusion

Nous nous sommes intéressés dans ce chapitre à la sélection d'attributs. Cette discipline de l'analyse de données et de l'apprentissage machine apparaît comme un domaine de recherche très actif, qui consiste à sélectionner, à partir de l'ensemble original des attributs, les attributs les plus pertinents pour la tâche d'apprentissage visée. Nous avons clarifié dans un premier temps les notions de pertinence et de redondance, et justifié la nécessité de procéder à la sélection d'attributs à travers le problème de la malédiction de la dimension. Ensuite, nous avons décrit les différentes étapes qui constituent le processus de sélection d'attributs. Par la suite, nous avons synthétisé les différentes méthodes de la sélection d'attributs en plusieurs catégories de différentes manières, selon le contexte d'apprentissage (non supervisé, supervisé, et semi-supervisé), selon la dépendance ou non à un algorithme de classification (filtre, enveloppe, et hybride), et selon ce qu'on évalue un attribut (feature ranking) ou plusieurs attributs à la fois (subset selection).

Au cours de cette étude, nous avons constaté le grand nombre de méthodes de sélection d'attributs disponibles. Par ailleurs, nous avons pu constater l'avantage des méthodes de type filtre en termes de rapidité, de simplicité, et de l'utilisation des contraintes must-link et cannot-link comme information a priori. C'est la raison pour laquelle nous nous sommes focalisés dans la deuxième partie de ce chapitre sur les méthodes de sélection d'attributs basées sur les scores de contraintes, dans lesquelles les données ainsi que les contraintes sont représentées par des graphes de similarité. Ces méthodes de type filtre sont indépendantes de l'algorithme de classification et sont basées sur un score de pertinence qui dépend du contexte d'apprentissage dans lequel la sélection d'attributs est effectuée. De plus, ce type de méthodes intègre facilement les informations a priori exprimées sous la forme de contraintes must-link et cannot-link. Nous avons ainsi dressé un état de l'art assez exhaustif sur les scores de contraintes dans les différents contextes d'apprentissage (non supervisé, supervisé, et semi-supervisé). Cependant, ces scores évaluent la pertinence des attributs individuellement, ignorant ainsi la corrélation entre eux. De plus, ils évaluent la pertinence des attributs dans l'espace défini par l'ensemble original des attributs. Par conséquent, ces méthodes peuvent être affectées par la malédiction de la dimension.

Pour remédier à ces inconvénients, nous proposons dans le prochain chapitre une nouvelle méthode de sélection d'attributs de type filtre basée sur un score original de contraintes capable d'évaluer la pertinence d'un sous-ensemble d'attributs à la fois dans les deux contextes supervisé et semi-supervisé.

Chapitre 4

Méthode de sélection d'attributs proposée

4.1 Introduction

Les méthodes de sélection d'attributs basées sur les scores de contraintes présentées dans le précédent chapitre évaluent la pertinence des attributs individuellement et ignorent la corrélation entre les attributs. De plus, elles sont basées sur les matrices laplaciennes et les matrices des degrés qui sont à leur tour déduites à partir des matrices de similarité calculées dans l'espace original des attributs. Par conséquent, ces méthodes sont susceptibles d'être affectées par le phénomène de la malédiction de la dimension. Dans le but de limiter ces inconvénients, nous avons proposé dans [205] une nouvelle méthode de sélection d'attributs de type filtre qui est basée sur un score de contraintes capable d'évaluer la pertinence d'un sous-ensemble d'attributs dans l'espace des attributs sélectionnés à la fois dans les contextes d'apprentissage supervisé et semi-supervisé.

Cette méthode de sélection d'attributs est décrite en détail dans la première partie de ce chapitre. La seconde partie de ce chapitre est dédiée à une étude expérimentale approfondie de la méthode de sélection d'attributs proposée. Une comparaison avec plusieurs méthodes de sélection d'attributs basées sur les scores de contraintes de l'état de l'art à travers plusieurs bases de données bien connues dans le domaine de la sélection d'attributs est réalisée.

4.2 Méthode de sélection d'attributs proposée

La méthode de sélection d'attributs que nous avons proposée dans [205] est de type filtre. Elle est basée sur un nouveau score de contraintes capable d'évaluer la pertinence d'un sous-ensemble d'attributs (subset selection) dans les deux contextes d'apprentissage supervisé et semi-supervisé. Ce score peut être avantageusement combiné avec une procédure de sélection d'attributs dans le but de sélectionner un nombre optimal d'attributs.

4.2.1 Score de contraintes proposé

Contrairement aux scores de contraintes existants dans la littérature, et qui sont basés sur les matrices laplaciennes, le score de contraintes que nous avons proposé, noté $\varepsilon^*(F_m)$, est basé uniquement sur les matrices de similarité pour évaluer la pertinence d'un sous-ensemble de m attributs, noté $F_m = \{f_1, f_2, \ldots, f_m\}$ (avec $m = 1,2,\ldots,d$). Il peut être adapté aux deux contextes d'apprentissage: supervisé ($* = S$) ou semi-supervisé ($* = SS$). Dans le contexte supervisé, $\varepsilon^S(F_m)$ utilise uniquement les données contraintes pour sélectionner les attributs, tandis que dans le contexte semi-supervisé, $\varepsilon^{SS}(F_m)$ utilise à la fois les données contraintes et non contraintes (non étiquetées) pour sélectionner les attributs. Le score $\varepsilon^*(F_m)$ est défini comme étant l'erreur quadratique entre les matrices de similarité $\hat{\boldsymbol{W}}^*$ et $\boldsymbol{W}(F_m)$, tel que:

$$\varepsilon^*(F_m) = \|\boldsymbol{W}(F_m) - \hat{\boldsymbol{W}}^*\|_2 \qquad \text{avec} \quad * = \text{S ou SS} \tag{4.1}$$

où $\|\cdot\|_2$ est la norme Euclidienne. $\hat{\boldsymbol{W}}^* \in \mathbb{R}^{n\times n}$ est une matrice de similarité cible dont les cellules correspondantes aux paires de contraintes must-link sont mises à 1 et les cellules correspondantes aux paires de contraintes cannot-link sont mises à 0. $\boldsymbol{W}(F_m) \in \mathbb{R}^{n\times n}$ est la matrice de similarité gaussienne calculée sur l'ensemble de données X avec le sous-ensemble d'attributs F_m. Elle est définie comme suit:

$$w_{ij}(F_m) = \texttt{exp}\left(-\frac{\delta^2\left(\boldsymbol{x}_i^{(m)}, \boldsymbol{x}_j^{(m)}\right)}{2\sigma^2}\right) \qquad i,j = 1,2,\ldots,n \tag{4.2}$$

où $\boldsymbol{x}_i^{(m)}$ est le vecteur de la i-ème donnée caractérisée par le sous-ensemble d'attributs F_m.

La sélection d'attributs consiste alors à déterminer le sous-ensemble d'attributs F_m qui minimise le score ε^*. Le score $\varepsilon^*(F_m)$ de l'équation (4.1) peut être réécrit sous la forme suivante:

$$\varepsilon^*(F_m) = \sum_{i=1}^{n}\sum_{j=1}^{n}\left(w_{ij}(F_m) - \hat{w}_{ij}^*\right)^2 \tag{4.3}$$

4.2.1.1 Score de contraintes supervisé ε^S

Dans le contexte supervisé, la matrice de similarité cible $\hat{\boldsymbol{W}}^S$ peut être définie comme suit:

$$\hat{w}_{ij}^S = \begin{cases} 1 & \text{si } (x_i,x_j) \in M \\ 0 & \text{si } (x_i,x_j) \in C \\ w_{ij}(F_m) & \text{sinon} \end{cases} \tag{4.4}$$

Les cellules de la matrice $\hat{\boldsymbol{W}}^S$ correspondantes aux paires de données non contraintes sont fixées à $w_{ij}(F_m)$ afin qu'elles ne soient pas prises en compte par $\varepsilon^S(F_m)$, $(w_{ij}(F_m) - \hat{w}_{ij}^S = 0)$. En effet, à partir des équations (4.3) et (4.4), il est facile de voir que le score de contraintes supervisé $\varepsilon^S(F_m)$ prend en compte que les données contraintes et peut être reformulé comme suit:

$$\varepsilon^S(F_m) = \sum_{\substack{(x_i,x_j)\in M \\ (x_i,x_j)\in C}} \left(w_{ij}(F_m) - \hat{w}_{ij}^S\right)^2 \tag{4.5}$$

L'idée sous-jacente à $\varepsilon^S(F_m)$ est simple et évidente: un sous-ensemble d'attributs F_m est considéré comme pertinent s'il produit une similarité $w_{ij}(F_m)$ proche de 1 entre deux données liées par une contrainte must-link d'une part, et une similarité $w_{ij}(F_m)$ proche de 0 entre deux données liées par une contrainte cannot-link, d'autre part. Par conséquent, le score de contraintes supervisé $\varepsilon^S(F_m)$ favorise le sous-ensemble d'attributs qui garantit le respect des contraintes must-link et cannot-link.

4.2.1.2 Score de contraintes semi-supervisé ε^{SS}

Dans le contexte semi-supervisé, de nouvelles paires de contraintes sont déduites à partir de quelques données étiquetées (prototypes) et des données non étiquetées afin de définir une matrice de similarité cible $\hat{\boldsymbol{W}}^{SS}$. $\hat{\boldsymbol{W}}^{SS}$ est une matrice binaire définie comme suit:

$$\hat{w}_{ij}^{SS} = \begin{cases} 1 & \text{si } (x_i,x_j) \in M^{SS} \\ 0 & \text{sinon} \end{cases} \tag{4.6}$$

Les cellules (i,j) de la matrice $\hat{\boldsymbol{W}}^{SS}$ sont mises à 1 (0 sinon) si leurs données correspondantes (x_i,x_j) appartiennent à M^{SS}. M^{SS} étant un nouvel ensemble de contraintes must-link déduit des sous-ensembles de prototypes X^l, $l = 1,\ldots,k$ et des données non étiquetées tel que:

$$M^{SS} = \left\{(x_i,x_j) \in X^2 | \exists l = 1,\ldots,k \quad \text{tel que} \quad NP(x_i) \in X^l \quad \text{et} \quad NP(x_j) \in X^l\right\} \tag{4.7}$$

où $NP(x_i)$ est le plus proche prototype, défini comme étant le prototype dont la distance avec $x_i \in X$ dans l'espace original des attributs est la plus petite. $NP(x_i)$ est défini par:

$$NP(x_i) = \underset{y \in \bigcup_{l=1,\dots,k} X^l}{\arg\min} \left(\delta^2(\boldsymbol{x}_i,\boldsymbol{y})\right) \tag{4.8}$$

Ainsi, une paire de données (x_i,x_j) appartient à M^{SS} si le plus proche prototype de x_i ($NP(x_i)$) et le plus proche prototype de x_j ($NP(x_j)$) appartiennent tous les deux au même sous-ensemble de prototypes X^l. Si $NP(x_i)$ est x_i, alors x_i appartient au sous-ensemble de prototypes X^l. Il est facile de voir que l'ensemble de contraintes must-link M est inclus dans M^{SS} ($M \subset M^{SS}$). La figure 4.1 présente une illustration de M^{SS} (pour la clarté de la figure, uniquement les contraintes must-link sont représentées). La figure 4.1(d) affiche les liens correspondants aux matrices $\boldsymbol{W}^{KNN1}$ (équation (3.26)), $\boldsymbol{W}^{KNN2}$ (équation (3.31)), $\boldsymbol{W}^{KNN3}$ (équation (3.38)), et $\boldsymbol{W}^{KNN4}$ (équation (3.46)) qui sont respectivement utilisées par les scores semi-supervisés C^3, C^4, C^5, C^6, et C^7. Ces liens sont déduits du graphe des KNN. Pour les comparer avec M^{SS}, nous fixons K à 1. Les liens $1NN = \{(x_i,x_j) \in X^2 \ | x_i = 1NN(x_j) \text{ or } x_j = 1NN(x_i)\}$ sont affichés sur la figure 4.1(d). La figure 4.1(c) montre clairement que les liens de M^{SS} respectent mieux la structure géométrique des classes contrairement à *1NN* qui regroupe plusieurs paires de données de classes différentes.

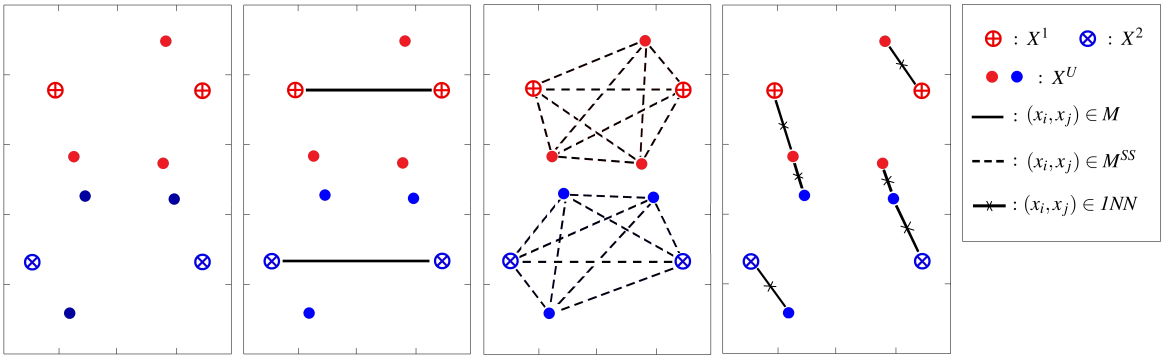


Figure 4.1: Contraintes must-link dans le contexte semi-supervisé. (a) Ensemble de données avec deux sous-ensembles de prototypes X^1 et X^2. (b) M. (c) M^{SS}. (d) *1NN*.

Le score de contraintes semi-supervisé $\varepsilon^{SS}(F_m) = \sum_{i=1}^{n}\sum_{j=1}^{n}\left(w_{ij}(F_m) - \hat{w}_{ij}^{SS}\right)^2$ évalue la capacité d'un sous-ensemble d'attributs à respecter les contraintes M, C, et M^{SS}.

Il est à noter que les scores de contraintes supervisés C^1 et C^2 ne calculent pas de similarité entre les données contraintes d'aucun espace d'attributs, tandis que les scores semi-supervisés C^3, C^4, C^5, C^6, et C^7 calculent les similarités uniquement dans l'espace original des attributs. En revanche, ε^S et ε^{SS} calculent la matrice de similarité $\boldsymbol{W}(F_m)$ dans le sous-ensemble

d'attributs sélectionnés F_m. Cependant, il convient de signaler que ε^{SS} s'applique uniquement dans le cas où les contraintes sont déduites à partir des étiquettes de classes et par conséquent toutes les classes doivent être représentées.

4.2.2 Procédure de sélection d'attributs

Les scores de contraintes existants dans la littérature (LS_r et C^c ($c = 1,\ldots,7$)) évaluent la pertinence des attributs individuellement en attribuant un score à chaque attribut. Les attributs sont alors ordonnés selon leurs scores de pertinence pour pouvoir sélectionner les plus pertinents en choisissant les m premiers attributs d'une manière empirique. Comme ε^* peut évaluer la pertinence d'un sous-ensemble d'attributs à la fois, il est donc avantageux de l'exploiter dans une méthode de sélection qui combine les attributs entre eux. Pour cela, nous avons choisi d'utiliser la méthode SFS [177] en raison de sa simplicité.

L'idée principale de cette technique est de commencer avec un ensemble vide d'attributs $F_0 = \{\varnothing\}$. L'attribut f_r qui minimise $\varepsilon^*(F_1)$ (avec $F_1 = \{f_r\}$) est sélectionné. Ensuite, les $d-1$ attributs restants sont combinés avec F_1 et le couple qui minimise $\varepsilon^*(F_2)$ est sélectionné. Une fois que m des d attributs ont été sélectionnés, le $(m+1)$-ème attribut combiné avec F_m qui minimise $\varepsilon^*(F_{m+1})$ est sélectionné. Cette procédure est répétée jusqu'à ce que les d attributs soient ordonnés, ou bien un critère d'arrêt prédéfini est satisfait. Le sous-ensemble $F_{\hat{m}}$ qui correspond au minimum de $\varepsilon^*(F_m)$ est choisi comme sous-ensemble optimal d'attributs. Le pseudo-code de cette méthode est résumé dans l'algorithme 4.1.

Algorithme 4.1: SFS.

Entrée: Ensemble des d attributs $F_d = \{f_1,\ldots,f_r,\ldots,f_d\}$.

1. Créer un ensemble vide d'attributs $F_0 = \{\varnothing\}$.
2. Pour $m = 1$ à d
   (a) Sélectionner l'attribut le plus pertinent $f_r^+ = \underset{f_r \in F_d \setminus F_{m-1}}{\arg\min} \big(\varepsilon^*(F_{m-1} \cup \{f_r\})\big)$.
   (b) Mettre à jour $F_m = F_{m-1} \cup \{f_r^+\}$.
3. Sélectionner le nombre optimal d'attributs $\hat{m}$ tel que $\hat{m} = \underset{m=1,2,\ldots,d}{\arg\min} \big(\varepsilon^*(F_m)\big)$.

Sortie: Sous-ensemble $F_{\hat{m}}$ des $\hat{m}$ attributs pertinents.

Il est à noter que même si cette procédure de sélection, combinée avec notre score, évalue la pertinence d'un sous-ensemble d'attributs et considère la corrélation entre les attributs, elle est du type "subset selection" et appartient à la catégorie des méthodes filtres. Elle conserve les avantages de cette dernière (indépendance vis-à-vis du classifieur et simplicité).

4.3 Évaluation de la méthode de sélection proposée

Dans cette section, nous évaluons et comparons notre méthode de sélection d'attributs avec plusieurs méthodes de l'état de l'art. Dans cette perspective, nous proposons que la sélection d'attributs et la classification soient réalisées dans le même contexte d'apprentissage. À cet effet, nous étudions le score ε^S dans le contexte d'apprentissage supervisé, et le score ε^{SS} dans le contexte d'apprentissage semi-supervisé. Les expérimentations sont réalisées sur douze bases de données réelles largement utilisées dans le domaine de la sélection d'attributs [150, 206]. Le tableau 4.1 résume les différentes caractéristiques de chacune de ces bases.

Tableau 4.1: Description des bases de données utilisées dans l'expérimentation. d nombre d'attributs, n nombre total de données, n_{App} nombre de données d'apprentissage, n_{Test} nombre de données de test, et k nombre de classes.

Bases de données	d	n	n_{App}	n_{Test}	k
`WBCD`	09	683	410	273	2
`Image Segmentation`	19	2310	1155	1155	7
`WDBC`	30	569	376	193	2
`Ionosphere`	34	351	176	175	2
`Dermatology`	34	366	183	183	6
`Libras Movement`	90	360	180	180	15
`Multiple Feature`	649	2000	1000	1000	10
`CNAE-9`	856	1080	540	540	9
`Yale`	1024	165	90	75	15
`ORL`	1024	400	200	200	40
`Pie10P`	2420	210	110	100	10
`ALLAML`	7129	72	37	35	2

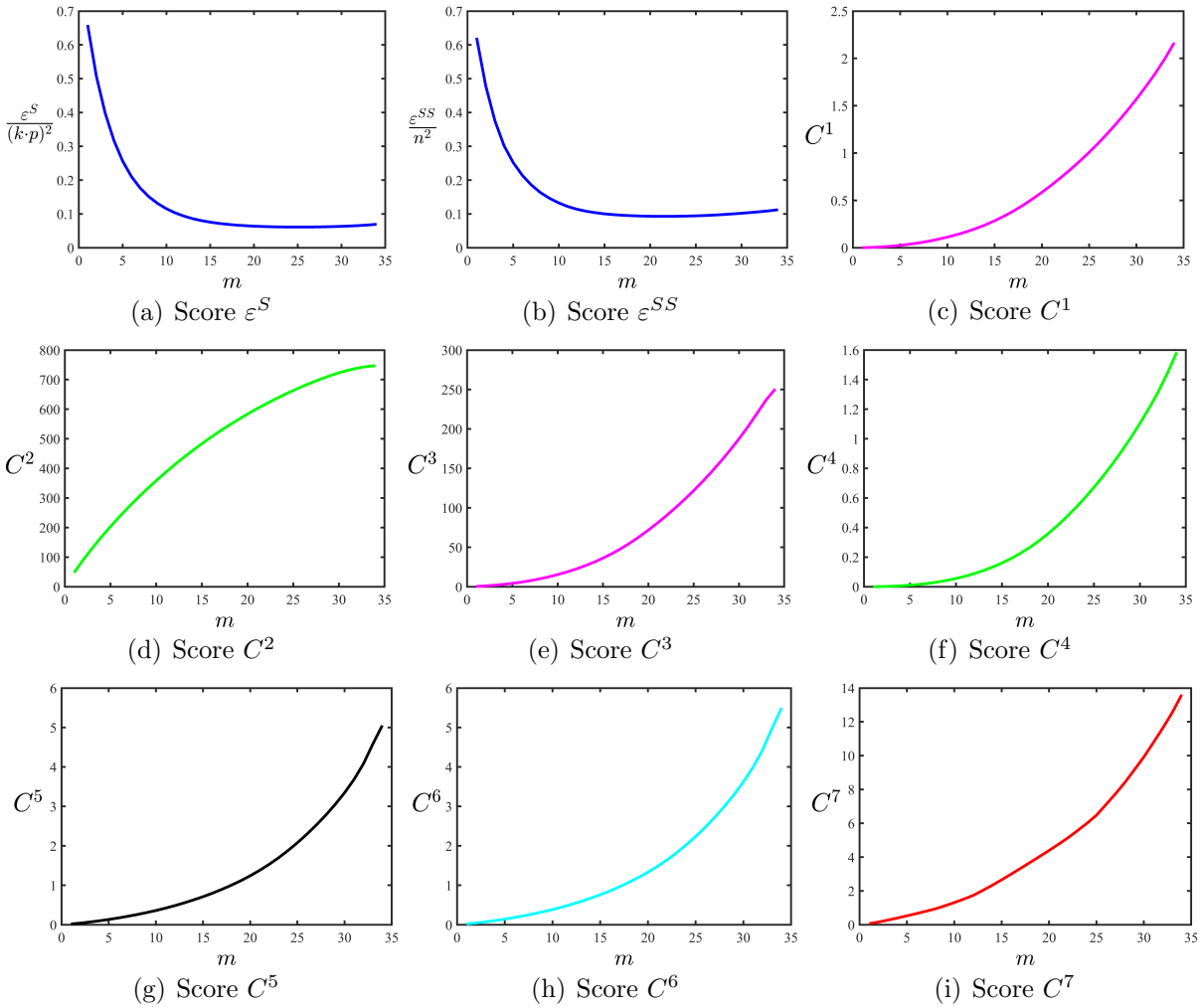
Toutes ces bases de données sont caractérisées par des attributs numériques. Les bases `Image Segmentation`, `Libras Movement`, `Multiple Feature`, `CNAE-9`, `Yale`, et `ORL` ont des classes équiprobables. Dans nos expériences, nous avons normalisé les différents attributs entre 0 et 1 pour que l'échelle des valeurs soit la même. La valeur du paramètre σ utilisé dans le calcul des matrices de similarité est fixée à 1. La stratégie "holdout" est utilisée comme procédure de validation. La procédure de sélection d'attributs est appliquée sur l'ensemble des données d'apprentissage X et répétée 100 fois. Pour chaque exécution, nous générons automatiquement un ensemble de contraintes comme suit: pour chaque classe, nous sélectionnons aléatoirement k sous-ensembles de prototypes X^l ($|X^l| = p$) à partir de l'ensemble X. Puis, nous déduisons les ensembles de contraintes M, C, et M^{SS} en utilisant respectivement les équations (2.28), (2.29), et (4.7). Dans nos expériences, p varie de 2 à 4. Les $k \cdot p$ prototypes générés sont exploités à la fois dans la sélection d'attributs et dans la

classification des données tests. La performance de la sélection d'attributs est mesurée par le taux de classification (accuracy) des données tests.

Avant de présenter les résultats d'évaluation de nos scores ε^S et ε^{SS} dans leurs contextes d'apprentissage respectifs, nous allons d'abord analyser l'évolution des scores ε^S, ε^{SS}, et C^c ($c = 1, \ldots, 7$) en fonction du nombre d'attributs sélectionnés.

4.3.1 Scores de contraintes en fonction du nombre d'attributs

La figure 4.2 affiche les variations des scores de contraintes ε^S, ε^{SS}, et C^c ($c = 1, \ldots, 7$) en fonction du nombre d'attributs sélectionnés m sur la base de données `Dermatology`. Le nombre de prototypes par classe est fixé à $p = 3$. Pour l'ensemble des scores de contraintes C^c ($c = 1, \ldots, 7$) de l'état de l'art, leur valeur pour un sous-ensemble de m attributs est estimée par la somme cumulée des scores des m attributs.



(a) Score ε^S (b) Score ε^{SS} (c) Score C^1

(d) Score C^2 (e) Score C^3 (f) Score C^4

(g) Score C^5 (h) Score C^6 (i) Score C^7

Figure 4.2: Performances des scores de contraintes ε^S, ε^{SS}, et C^c ($c = 1, \ldots, 7$) en fonction du nombre m d'attributs sélectionnés obtenues sur la base de données `Dermatology`.

À partir de cette figure, nous remarquons que les courbes de nos scores ε^S et ε^{SS} sont quasi-convexes, tandis que les courbes des scores C^1, C^3, C^4, C^5, C^6, et C^7 augmentent de façon monotone, et la courbe du score C^2 diminue de façon monotone en fonction de différents nombres d'attributs sélectionnés m. Des résultats similaires, qui ne sont pas présentés dans le but de réduire le volume de la thèse, sont observés sur d'autres bases de données et pour différentes valeurs de p. Nous pouvons donc conclure que contrairement aux autres scores, seuls ε^S et ε^{SS} présentent des minimums qui peuvent être exploités pour déterminer automatiquement le nombre optimal d'attributs à sélectionner.

4.3.2 Résultats de la sélection d'attributs supervisée

Dans un contexte supervisé, nous disposons uniquement de k sous-ensembles de prototypes X^l (avec $|X^l| = p$) comme information a priori. Cette information peut être exploitée à la fois dans la sélection d'attributs et dans l'évaluation des attributs sélectionnés. Pour juger l'efficacité de notre score ε^S, nous comparons ses performances avec celles des scores C^1 et C^2, ainsi qu'avec celles des méthodes de sélection supervisée ReliefF [186] et mRMR [187].

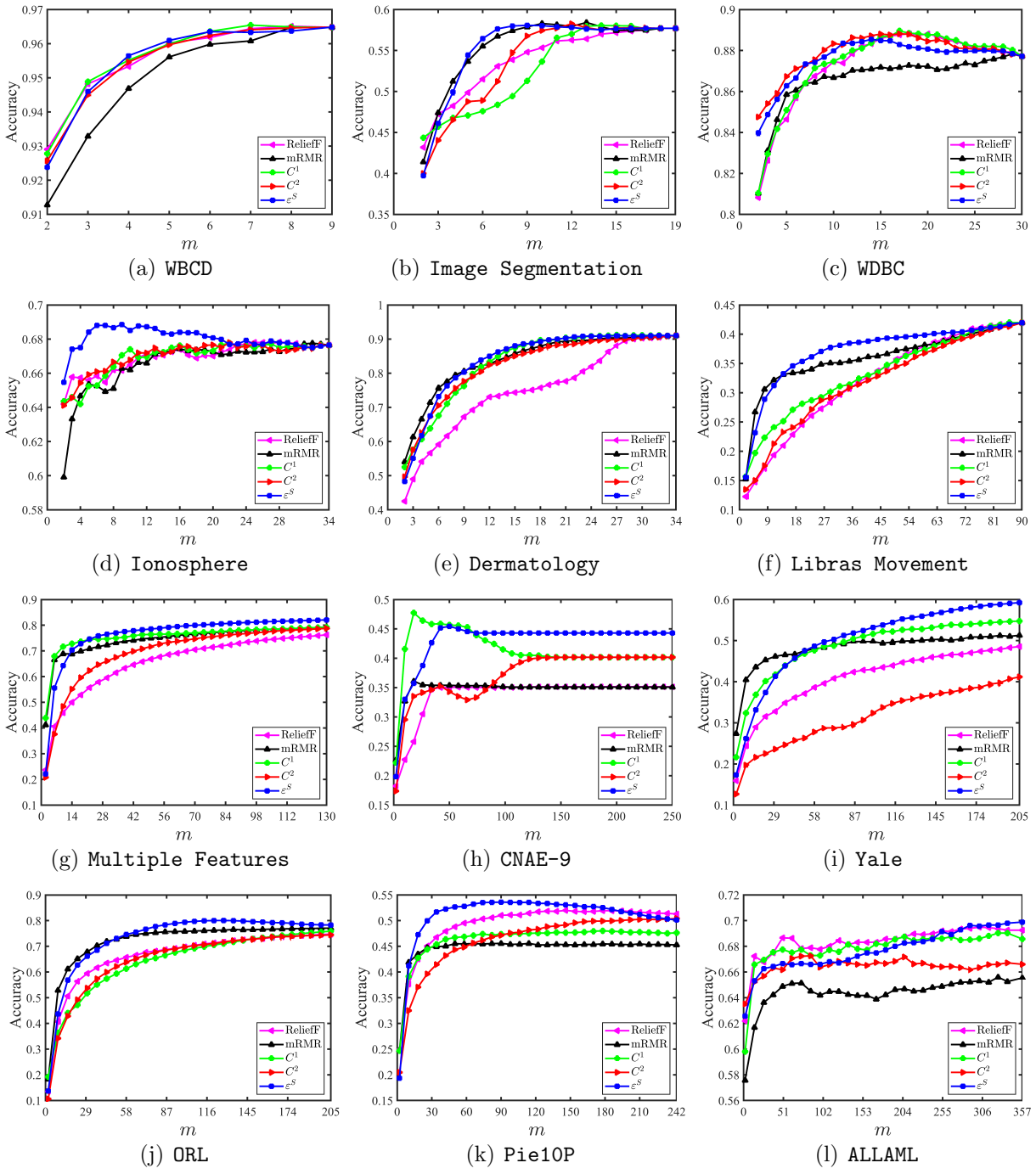
Pour évaluer la pertinence des attributs sélectionnés, chaque donnée test est projetée dans l'espace des attributs sélectionnés puis affectée à l'une des k classes en utilisant la méthode KNN. Cette méthode est couramment utilisée pour la validation des méthodes de sélection d'attributs. Pour être indépendant des règles de décision sur le vote majoritaire, en cas d'égalité de votes, nous fixons K à 1 (méthode 1NN). En général, pour classer les données tests, toutes les données d'apprentissage sont utilisées comme prototypes par la méthode 1NN, alors que seuls quelques prototypes sont utilisés par les scores de contraintes supervisés [193, 196, 197]. Dans notre cas, nous évaluons les attributs sélectionnés en utilisant uniquement des $k \cdot p$ prototypes utilisés pour la sélection d'attributs. De cette manière, nous nous plaçons dans des conditions similaires à celles des applications réelles.

Étant donné que le taux de classification dépend essentiellement du nombre d'attributs sélectionnés m et du nombre de prototypes p, nous examinons d'abord le taux de classification en fonction du nombre d'attributs m pour un nombre p fixe. Ensuite, pour un nombre d'attributs m fixe, nous comparons les taux de classification pour différentes valeurs de p.

4.3.2.1 Taux de classification en fonction du nombre d'attributs

La figure 4.3 affiche les variations des taux moyens de classification en fonction du nombre d'attributs sélectionnés par ε^S, C^1, C^2, ReliefF, et mRMR sur les douze bases de données

tests. Le nombre de prototypes p est fixé à 3 et les taux moyens de classification sont moyennés sur 100 exécutions.



(a) WBCD (b) Image Segmentation (c) WDBC

(d) Ionosphere (e) Dermatology (f) Libras Movement

(g) Multiple Features (h) CNAE-9 (i) Yale

(j) ORL (k) Pie10P (l) ALLAML

Figure 4.3: Taux moyens de classification en fonction du nombre m d'attributs sélectionnés par les méthodes ε^S, C^1, C^2, ReliefF, et mRMR sur les douze bases de données.

À partir de cette figure, nous constatons que, pour la plupart des bases de données, les taux moyens de classification obtenus avec ε^S sont supérieurs à ceux obtenus avec C^1, C^2, ReliefF, et mRMR. En examinant de prés les différentes courbes, nous pouvons remarquer

que, pour les bases de données `Ionosphere`, `Libras Movement`, et `Pie10P`, les courbes de ε^S se démarquent clairement de celles de C^1, C^2, ReliefF, et mRMR. Tandis que, pour les autres bases de données, les courbes de ε^S, C^1, C^2, ReliefF, et mRMR se chevauchent, rendant difficile leur distinction et leur comparaison. Pour cela, nous proposons d'examiner, dans la section 4.3.2.3 leurs taux moyens de classification pour un nombre fixe d'attributs sélectionnés.

4.3.2.2 Nombre optimal d'attributs sélectionnés par ε^S

Déterminer le nombre optimal d'attributs dans le cadre de la sélection d'attributs de type filtre par classement d'attributs (feature ranking) demeure à ce jour un challenge. En effet, les méthodes C^1, C^2, ReliefF, et mRMR ne sont pas en mesure de déterminer le nombre optimal d'attributs. Par contre, notre score ε^S peut convenir à le faire automatiquement grâce à l'algorithme 4.1. Comme mentionné précédemment, la courbe de ε^S en fonction du nombre d'attributs présente un minimum qui peut être considéré comme le nombre optimal d'attributs. Pour étayer cette assertion, nous présentons simultanément dans le tableau 4.2 le taux moyen de classification $Acc_{(\hat{m})}$ obtenu avec le sous-ensemble $F(\hat{m})$ d'attributs sélectionnés par ε^S et le taux moyen maximal de classification $Acc_{(m^*)}$.

Tableau 4.2: Taux moyens de classification et nombre optimal d'attributs obtenus avec ε^S.

Bases de données	$p=2$		$p=3$		$p=4$	
	$Acc_{(m^*)}$	$Acc_{(\hat{m})}$	$Acc_{(m^*)}$	$Acc_{(\hat{m})}$	$Acc_{(m^*)}$	$Acc_{(\hat{m})}$
`WBCD`	$96.21_{(9)}$	$95.79_{(6)}$	$96.48_{(9)}$	$96.35_{(6)}$	$96.05_{(9)}$	$95.91_{(6)}$
`Image Segmentation`	$55.88_{(18)}$	$55.88_{(18)}$	$58.07_{(9)}$	$57.70_{(18)}$	$60.03_{(7)}$	$59.29_{(18)}$
`WDBC`	$86.55_{(21)}$	$86.41_{(24)}$	$88.54_{(14)}$	$87.97_{(25)}$	$89.17_{(11)}$	$88.38_{(25)}$
`Ionosphere`	$67.07_{(3)}$	$65.61_{(13)}$	$68.85_{(9)}$	$68.73_{(12)}$	$70.47_{(9)}$	$70.45_{(11)}$
`Dermatology`	$88.65_{(32)}$	$88.03_{(26)}$	$91.07_{(32)}$	$90.85_{(24)}$	$91.88_{(26)}$	$91.72_{(24)}$
`Libras Movement`	$38.34_{(90)}$	$38.34_{(90)}$	$42.03_{(89)}$	$41.95_{(90)}$	$43.53_{(90)}$	$43.53_{(90)}$
`Multiple Feature`	$77.86_{(629)}$	$67.92_{(25)}$	$82.80_{(534)}$	$75.27_{(24)}$	$86.06_{(217)}$	$79.28_{(25)}$
`CNAE-9`	$37.46_{(37)}$	$36.64_{(92)}$	$45.68_{(49)}$	$44.29_{(86)}$	$50.24_{(60)}$	$48.81_{(93)}$
`Yale`	$49.80_{(191)}$	$42.37_{(66)}$	$60.47_{(274)}$	$49.05_{(61)}$	$63.60_{(202)}$	$51.88_{(58)}$
`ORL`	$71.04_{(121)}$	$71.04_{(121)}$	$80.06_{(125)}$	$79.91_{(115)}$	$86.11_{(136)}$	$85.6_{(110)}$
`Pie10P`	$48.32_{(121)}$	$47.70_{(75)}$	$53.66_{(91)}$	$53.26_{(67)}$	$53.74_{(120)}$	$53.18_{(64)}$
`ALLAML`	$67.94_{(308)}$	$65.66_{(47)}$	$70.11_{(351)}$	$66.29_{(29)}$	$76.71_{(344)}$	$72.89_{(25)}$

À partir de ce tableau, nous constatons que, pour la plupart des bases de données, les taux moyens de classification $Acc_{(\hat{m})}$ et $Acc_{(m^*)}$ sont proches, à l'exception des bases `Multiple Feature` et `Yale`. De plus, nous remarquons que le nombre d'attributs sélectionnés $\hat{m}$ reste souvent inférieur au nombre optimal d'attributs m^*.

4.3.2.3 Taux de classification en fonction du nombre de prototypes

Pour étudier l'influence du nombre de prototypes, nous proposons de comparer les taux de classification obtenus par les méthodes ε^S, C^1, C^2, ReliefF, et mRMR pour différents nombres de prototypes p par classe. Pour cela, nous utilisons le nombre optimal d'attributs $\hat{m}$ obtenu dans la section précédente. Le nombre de prototypes p varie entre 2 et 4. Pour un nombre $\hat{m}$ fixe d'attributs et pour chacune des 100 exécutions, nous proposons d'ordonner les 5 méthodes de sélection d'attributs supervisée ε^S, C^1, C^2, ReliefF, et mRMR dans un ordre décroissant de leurs taux de classification.

Soit $\text{rang}_q^{[c]}$ le rang des méthodes de sélection $c = \varepsilon^S$, C^1, C^2, ReliefF, ou mRMR à l'exécution q. Ce rang peut prendre les valeurs 1, 2, 3, 4, ou 5. Pour chaque exécution q, la méthode ayant le meilleur taux de classification est classée 1, tandis que la méthode ayant le plus petit taux de classification est classée 5. Les méthodes ayant le même taux de classification obtiennent le même rang. La somme des rangs $R^{[c]}$ pour chaque méthode est définie par:

$$R^{[c]} = \sum_{q=1}^{100} \text{rang}_q^{[c]} \tag{4.9}$$

La méthode ayant la somme des rangs la plus petite est considérée comme la plus efficace.

Le tableau 4.3 montre les sommes des rangs obtenues par les méthodes de sélection d'attributs supervisée sur les douze bases de données et pour différents nombres de prototypes p. La somme des rangs ayant la plus petite valeur est indiquée en gras. Ces résultats indiquent que notre méthode ε^S possède la somme des rangs la plus petite (23 fois sur 36), suivie de la méthode C^1 (15 fois sur 36).

L'apport de ε^S par rapport aux autres méthodes de sélection d'attributs supervisée peut s'expliquer par les deux points suivants:

- ε^S tient compte de la corrélation entre les attributs (ε^S estime la pertinence d'un sous-ensemble d'attributs), alors que les méthodes C^1, C^2, et ReliefF ignorent la corrélation entre les attributs.
- ε^S calcule les similarités entre les données dans l'espace des attributs sélectionnés (équation (4.5)), alors que les méthodes C^1, C^2, ReliefF, et mRMR ne calculent aucune similarité entre les données.

Il est à noter qu'avec les $\hat{m}$ attributs, la méthode mRMR n'est pas très efficace malgré le fait qu'elle minimise la redondance entre les attributs.

Tableau 4.3: Sommes des rangs obtenues par les différentes méthodes de sélection d'attributs supervisées sur les douze bases de données pour différents nombres p de prototypes.

Bases de données	p	$\hat{m}$	ReliefF	mRMR	C^1	C^2	ε^S
	2	6	221	272	**207**	219	216
WBCD	3	6	203	299	**188**	222	200
	4	6	**188**	298	196	198	206
	2	18	**107**	112	**107**	**107**	**107**
Image Segmentation	3	18	**107**	124	**107**	**107**	**107**
	4	18	**108**	144	**108**	**108**	**108**
	2	24	232	351	**219**	239	257
WDBC	3	25	258	322	**217**	249	257
	4	25	253	301	**220**	234	280
	2	13	290	**242**	276	291	278
Ionosphere	3	12	279	307	275	296	**259**
	4	11	312	297	**270**	284	272
	2	26	415	302	221	290	**209**
Dermatology	3	24	473	265	209	283	**205**
	4	24	443	255	216	304	**202**
	2	90	**100**	**100**	**100**	**100**	**100**
Libras Movement	3	90	**100**	**100**	**100**	**100**	**100**
	4	90	**100**	**100**	**100**	**100**	**100**
	2	25	489	262	259	333	**155**
Multiple Feature	3	24	485	264	195	394	**161**
	4	25	473	267	174	420	**160**
	2	92	283	394	203	242	**156**
CNAE-9	3	86	422	308	201	405	**146**
	4	93	415	297	**161**	443	176
	2	66	362	**190**	208	493	216
Yale	3	61	378	204	209	499	**184**
	4	58	377	196	202	499	**187**
	2	121	460	231	328	359	**104**
ORL	3	115	375	199	418	384	**104**
	4	110	386	178	339	449	**110**
	2	75	337	359	259	335	**163**
Pie10P	3	67	260	364	325	352	**161**
	4	64	229	343	340	357	**181**
	2	47	229	319	**225**	267	254
ALLAML	3	29	240	323	**231**	281	260
	4	25	**220**	357	244	289	254

4.3.3 Résultats de la sélection d'attributs semi-supervisée

Dans un contexte semi-supervisé, nous disposons à la fois des k sous-ensembles de prototypes X^l et des données non étiquetées qui appartiennent à l'ensemble des données d'apprentissage X. Ces ensembles sont utilisés à la fois dans la sélection des attributs et dans l'évaluation des attributs sélectionnés. Pour juger de l'efficacité de notre score ε^{SS}, nous comparons ses performances avec celles des scores semi-supervisés C^c ($c = 3, \ldots, 7$) et celles des méthodes de sélection d'attributs semi-supervisée EnsCLS [201] et SCGS [204]. Pour cela, nous suivons le même schéma expérimental que celui de la sélection d'attributs supervisée. Tout d'abord, nous évaluons le taux de classification en fonction du nombre d'attributs sélectionnés m pour un nombre de prototypes p fixe. Ensuite, pour un nombre optimal d'attributs $\hat{m}$, nous comparons les taux de classification obtenus par ε^{SS}, d'abord avec ceux obtenus par C^c ($c = 3, \ldots, 7$), puis avec ceux obtenus par les méthodes EnsCLS et SCGS.

Avant d'effectuer toutes ces comparaisons, il est nécessaire de mesurer la qualité de l'ensemble de contraintes M^{SS} sur lequel se base notre score de contraintes semi-supervisé ε^{SS}.

4.3.3.1 Évaluation de la qualité de l'ensemble M^{SS}

Le score de contraintes ε^{SS} propage les contraintes must-link M sur l'ensemble des données non étiquetées pour créer un nouvel ensemble de contraintes must-link M^{SS} (équation (4.7)). Pour évaluer la pertinence de l'ensemble de contraintes M^{SS}, nous proposons de le comparer à l'ensemble initial de contraintes must-link M en calculant les deux mesures suivantes:

Taux des nouvelles contraintes must-link (NML): Ce taux évalue le rapport entre le nombre de contraintes must-link dans M^{SS} et le nombre de contraintes must-link dans M. NML permet de quantifier la quantité de contraintes must-link ajoutées. Il est défini par:

$$\text{NML} = \frac{|M^{SS}|}{|M|} \tag{4.10}$$

Taux des nouvelles contraintes must-link correctes (CNML): Ce taux correspond au nombre de contraintes must-link correctement prédites sur le nombre total de contraintes must-link possibles dans M^{SS}. CNML permet de quantifier la qualité des contraintes must-link ajoutées. Il est défini par:

$$\text{CNML} = \frac{|\ \{(x_i,x_j) \in M^{SS} \setminus M \ \ |\ \ \omega(x_i) = \omega(x_j)\}\ |}{|M^{SS} \setminus M|} \tag{4.11}$$

$\omega(x_i)$ représente la vraie classe de la donnée x_i.

Pour ces deux mesures, nous avons NML ≥ 1 et $0 \leq$ CNML ≤ 1. Des valeurs élevées de NML et CNML indiquent que les contraintes must-link de l'ensemble M^{SS} sont correctes.

Le tableau 4.4 contient les taux moyens de NML et CNML obtenus sur les douze bases de données avec 100 ensembles de prototypes générés aléatoirement (p varie de 2 à 4). Pour une bonne interprétation de NML, nous affichons également la taille de l'ensemble initial M des contraintes must-link ($|M| = k \cdot p \cdot (p-1)$). Nous constatons que la valeur de NML varie fortement d'une base de données à l'autre, elle diminue lorsque p augmente et elle prend des valeurs élevées lorsque la taille de l'ensemble de contraintes must-link $|M|$ est petite. La valeur de CNML est proche ou dépasse 0.7 pour la plupart des bases de données. Nous pouvons donc en déduire que la majorité des nouvelles contraintes must-link sont correctes.

4.3.3.2 Taux de classification en fonction du nombre d'attributs

Afin d'effectuer la sélection d'attributs et l'évaluation des attributs sélectionnés dans le même contexte d'apprentissage semi-supervisé, nous avons suivi la même stratégie que celle adoptée par Kalakech et al. [207]. Pour cela, nous utilisons d'abord la méthode de classification spectrale sous contraintes (Algorithme 2.2) pour classer les données d'apprentissage non étiquetées. Ensuite, ces données étiquetées sont définies comme prototypes de classes pour classer les données tests. Enfin, chaque donnée test est projetée dans l'espace des attributs sélectionnés puis affecté à l'une des k classes en utilisant la méthode 1NN. Cette méthode 1NN utilise à la fois les $(k \cdot p)$ prototypes initialement disponibles et les nouveaux prototypes estimés par la méthode de classification spectrale sous contraintes.

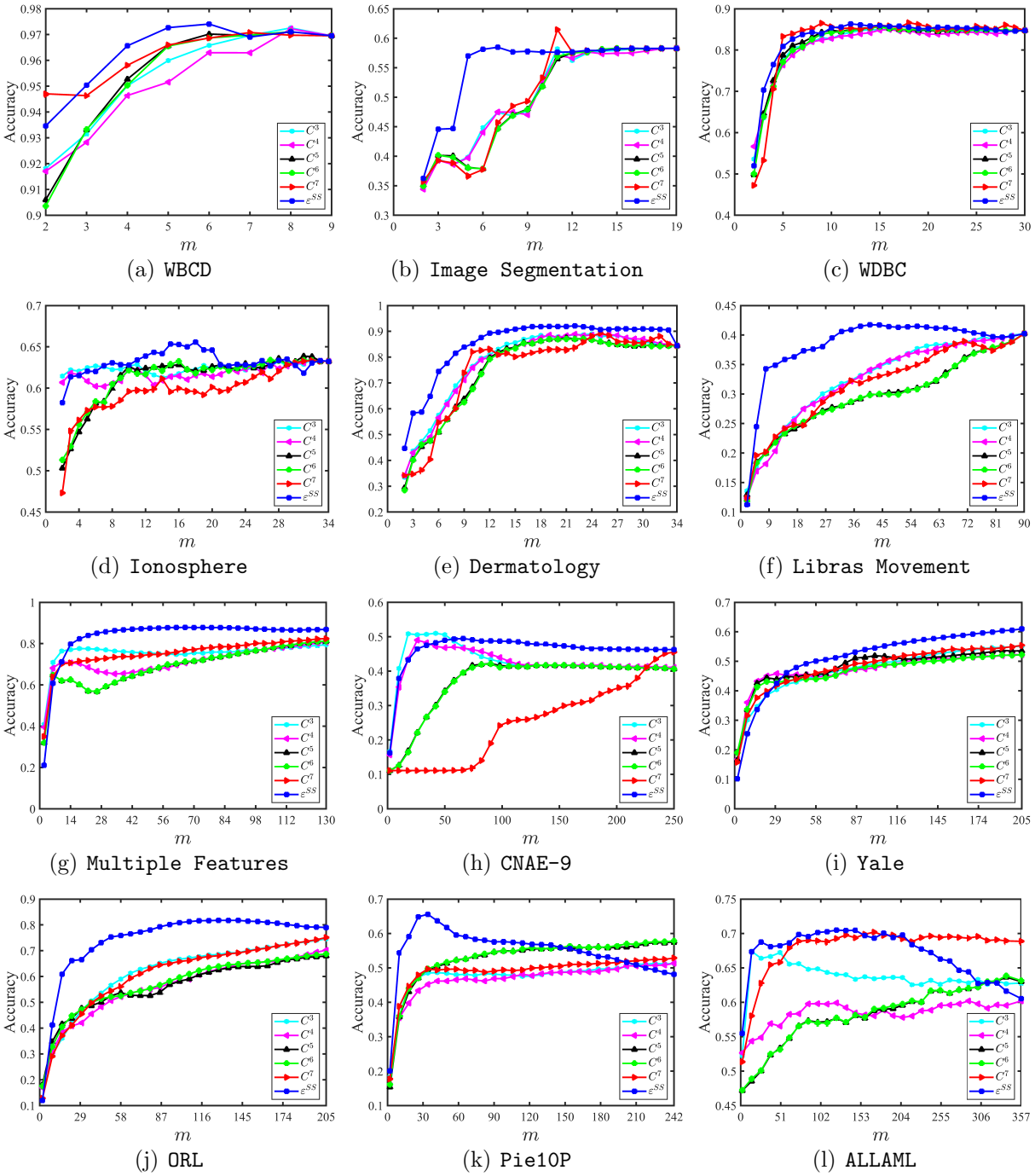
La figure 4.4 affiche les taux moyens de classification en fonction du nombre d'attributs sélectionnés par les scores semi-supervisés ε^{SS}, C^c $(c = 3, \ldots, 7)$ sur les douze bases de données. Le nombre p de prototypes est fixé à 3 et les taux de classification sont moyennés sur 100 exécutions. Il ressort de cette figure que les taux moyens de classification fournis par ε^{SS} sont supérieurs à ceux obtenus par C^c $(c = 3, \ldots, 7)$ pour toutes les bases de données. En examinant les différentes courbes, nous constatons que celles de ε^{SS} sont plus élevées et se démarquent nettement de celles de C^c $(c = 3, \ldots, 7)$, qui se chevauchent entre elles.

4.3.3.3 Nombre optimal d'attributs sélectionnés par ε^{SS}

Comme pour le score supervisé ε^{S}, le score semi-supervisé ε^{SS} peut être également exploité pour déterminer automatiquement le nombre optimal d'attributs $\hat{m}$, car la courbe du score ε^{SS} en fonction du nombre d'attributs présente un minimum qui peut être considéré comme

Tableau 4.4: Taux moyens de NML et CNML obtenus sur les douze bases de données.

Bases de données	p	$\|M\|$	NML	CNML
	2	04	24037.0	0.8596
WBCD	3	12	7956.2	0.8647
	4	24	3915.4	0.8799
	2	14	15234.0	0.4690
Image Segmentation	3	42	4922.4	0.4966
	4	84	2408.2	0.5126
	2	04	19638.0	0.7962
WDBC	3	12	6392.0	0.8094
	4	24	3140.9	0.8337
	2	04	5450.1	0.5826
Ionosphere	3	12	1752.6	0.5962
	4	24	879.9	0.6008
	2	12	534.2	0.8050
Dermatology	3	36	174.4	0.8504
	4	72	86.5	0.8630
	2	30	81.19	0.4305
Libras Movement	3	90	25.0	0.5077
	4	180	12.0	0.5684
	2	20	5198.6	0.6269
Multiple Features	3	60	1698.3	0.6939
	4	120	841.3	0.7401
	2	80	11.1	0.6324
ORL	3	240	3.4	0.7694
	4	480	1.7	0.8780
	2	18	6394.5	0.1654
CNAE-9	3	54	1720.6	0.1899
	4	108	659.5	0.2304
	2	30	18.7	0.3282
Yale	3	90	5.6	0.4444
	4	180	2.6	0.6021
	2	20	62.9	0.3924
Pie10P	3	60	19.6	0.5600
	4	120	9.6	0.6755
	2	4	188.0	0.6572
ALLAML	3	12	61.3	0.6954
	4	24	30.0	0.7387



(a) WBCD (b) Image Segmentation (c) WDBC

(d) Ionosphere (e) Dermatology (f) Libras Movement

(g) Multiple Features (h) CNAE-9 (i) Yale

(j) ORL (k) Pie10P (l) ALLAML

Figure 4.4: Taux moyens de classification en fonction du nombre m d'attributs sélectionnés par les scores semi-supervisés ε^{SS}, C^3, C^4, C^5, C^6, et C^7 sur les douze bases de données.

le nombre optimal d'attributs. Le tableau 4.5 présente simultanément le taux moyen de classification $Acc_{(\hat{m})}$ obtenu avec le sous-ensemble d'attributs $F(\hat{m})$ calculé par le score ε^{SS}, et le taux moyen maximal de classification $Acc_{(m^*)}$. On observe que, pour la plupart des bases de données, les taux moyens de classification $Acc_{(\hat{m})}$ et $Acc_{(m^*)}$ sont proches, et que les nombres d'attributs sélectionnés $\hat{m}$ sont inférieurs aux nombres optimaux d'attributs m^*.

Tableau 4.5: Taux moyens de classification et nombre optimal d'attributs obtenus avec ε^{SS}.

Bases de données	$p=2$		$p=3$		$p=4$	
	$Acc_{(m^*)}$	$Acc_{(\hat{m})}$	$Acc_{(m^*)}$	$Acc_{(\hat{m})}$	$Acc_{(m^*)}$	$Acc_{(\hat{m})}$
WBCD	$96.20_{(9)}$	$95.29_{(7)}$	$97.41_{(6)}$	$96.90_{(7)}$	$96.56_{(6)}$	$96.26_{(7)}$
Image Segmentation	$56.45_{(19)}$	$56.45_{(18)}$	$58.47_{(7)}$	$58.28_{(18)}$	$60.67_{(15)}$	$60.63_{(18)}$
WDBC	$83.57_{(9)}$	$83.30_{(24)}$	$86.33_{(12)}$	$85.24_{(25)}$	$86.47_{(13)}$	$85.74_{(25)}$
Ionosphere	$64.05_{(16)}$	$60.50_{(6)}$	$65.57_{(18)}$	$62.73_{(7)}$	$69.57_{(13)}$	$65.21_{(6)}$
Dermatology	$90.02_{(18)}$	$89.39_{(22)}$	$92.14_{(22)}$	$91.86_{(21)}$	$92.60_{(20)}$	$92.52_{(21)}$
Libras Movement	$38.82_{(51)}$	$36.76_{(81)}$	$41.73_{(41)}$	$39.73_{(85)}$	$43.89_{(55)}$	$42.56_{(88)}$
Multiple Feature	$82{,}85_{(73)}$	$78{,}71_{(22)}$	$87.85_{(63)}$	$84.24_{(23)}$	$89{,}67_{(62)}$	$86{,}48_{(24)}$
CNAE-9	$42.47_{(65)}$	$41.23_{(235)}$	$49.43_{(65)}$	$46.26_{(239)}$	$53.54_{(70)}$	$47.88_{(234)}$
Yale	$52.96_{(398)}$	$41.79_{(48)}$	$63.64_{(307)}$	$49.19_{(51)}$	$71.20_{(338)}$	$53.31_{(53)}$
ORL	$73.42_{(118)}$	$72.90_{(106)}$	$81.79_{(136)}$	$81.28_{(108)}$	$86.68_{(130)}$	$85.98_{(109)}$
Pie10P	$55.06_{(29)}$	$50.12_{(59)}$	$65.58_{(34)}$	$59.30_{(60)}$	$70.04_{(27)}$	$64.15_{(59)}$
ALLAML	$69.49_{(156)}$	$66.09_{(21)}$	$71.20_{(132)}$	$68.71_{(20)}$	$76.31_{(191)}$	$72.89_{(20)}$

4.3.3.4 Taux de classification en fonction du nombre de prototypes

Nous avons aussi comparé les taux de classification obtenus par les scores semi-supervisés ε^{SS} et C^c $(c = 3,\ldots,7)$ en fonction du nombre p de prototypes. Pour chaque base de données, nous avons calculé la somme des rangs $R^{[c]}$ obtenue par ε^{SS} et C^c $(c = 3,\ldots,7)$ sur 100 exécutions, en considérant le nombre optimal d'attributs $\hat{m}$ obtenu par ε^{SS} (Tableau 4.5).

Le tableau 4.6 présente les sommes des rangs obtenues par les scores de contraintes semi-supervisés avec 100 ensembles de prototypes différents (p varie de 2 à 4). Nous constatons que ε^{SS} obtient la plus petite somme des rangs (20 fois sur 36), suivi de C^3 (10 fois sur 36), et de C^7 (8 fois sur 36). Ces résultats indiquent que les attributs sélectionnés par ε^{SS} sont plus discriminants que ceux sélectionnés par C^c $(c = 3,\ldots,7)$, quelle que soit la valeur de p. Cela peut s'expliquer par la capacité de ε^{SS} à tenir compte de la corrélation entre les attributs et à calculer la matrice de similarité dans l'espace fourni par les attributs sélectionnés. En revanche, les autres scores semi-supervisés calculent les similarités dans l'espace fourni par l'ensemble original des attributs, qui peut être de grande dimension.

Pour mettre d'avantage en évidence l'efficacité de notre score semi-supervisé ε^{SS}, nous l'avons comparé à deux autres méthodes de sélection semi-supervisée, à savoir EnsCLS [201] et SCGS [204]. Le tableau 4.7 présente les sommes des rangs obtenues par ε^{SS}, EnsCLS, et SCGS avec 100 ensembles de prototypes différents pour chacune des douze bases de données. La somme des rangs de chaque méthode est calculée en considérant le nombre optimal d'attributs $\hat{m}$ présenté dans le tableau 4.5. Les résultats montrent que ε^{SS} obtient le meilleur classement (22 fois sur 36), suivi de SCGS (9 fois sur 36), et de EnsCLS (5 fois sur 36).

Tableau 4.6: Sommes des rangs obtenues par les scores de contraintes semi-supervisés sur les douze bases de données pour différents nombres de prototypes p.

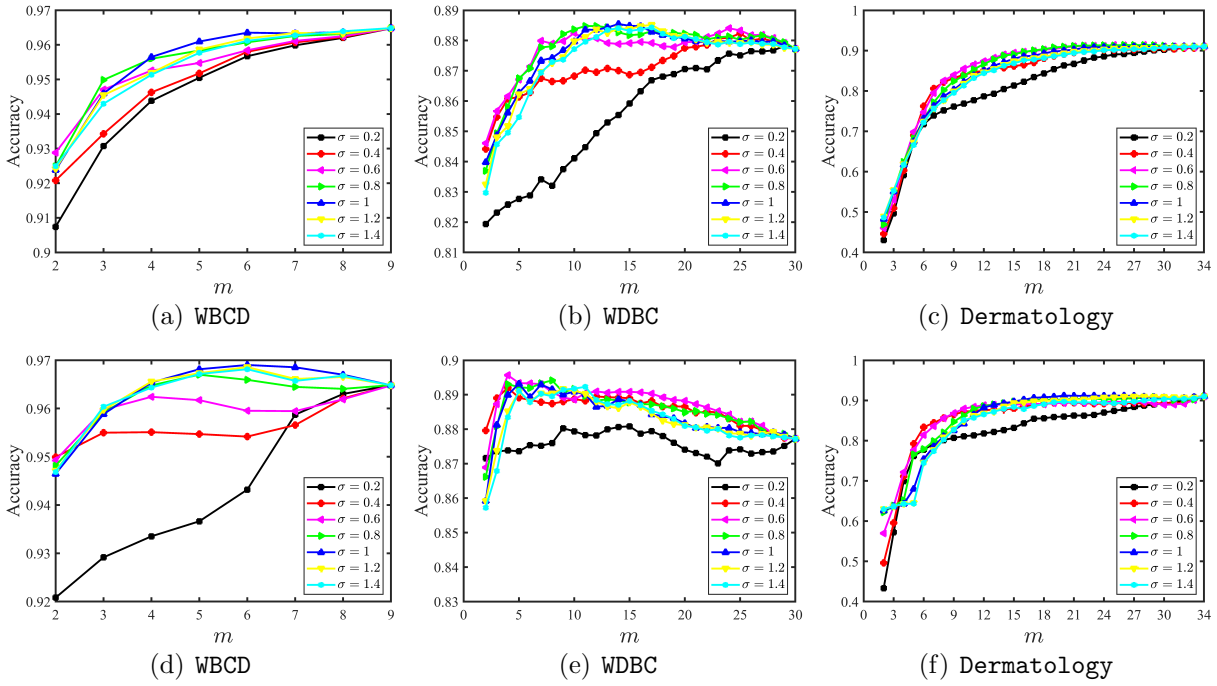
Bases de données	p	$\hat{m}$	C^3	C^4	C^5	C^6	C^7	ε^{SS}
	2	7	236	308	211	216	**205**	234
WBCD	3	7	232	286	212	210	**185**	196
	4	7	281	274	206	203	**199**	221
	2	18	**131**	250	**131**	**131**	**131**	144
Image Segmentation	3	18	**120**	225	**120**	**120**	**120**	124
	4	18	**116**	185	**116**	**116**	**116**	**116**
	2	24	308	363	**266**	274	324	331
WDBC	3	25	359	378	297	297	**234**	305
	4	25	343	359	308	299	**213**	340
	2	6	**296**	301	345	355	377	302
Ionosphere	3	7	**269**	298	347	346	393	322
	4	6	**258**	295	356	295	461	307
	2	22	267	322	330	338	477	**225**
Dermatology	3	21	313	315	305	318	491	**204**
	4	21	314	346	295	291	492	**196**
	2	81	335	297	307	301	411	**260**
Libras Movement	3	85	**269**	281	337	340	345	304
	4	88	**245**	267	293	309	380	248
	2	22	179	366	458	458	404	**133**
Multiple Feature	3	23	222	371	492	491	306	**118**
	4	24	225	372	501	502	285	**119**
	2	235	**243**	308	**243**	**243**	416	328
CNAE-9	3	239	280	381	280	280	305	**248**
	4	234	299	368	299	299	289	**205**
	2	48	379	**269**	307	327	359	275
Yale	3	51	388	336	295	352	352	**208**
	4	53	396	330	345	380	337	**182**
	2	106	267	476	495	465	252	**100**
ORL	3	108	238	507	521	448	260	**100**
	4	109	231	504	544	430	266	**100**
	2	59	377	405	336	334	321	**208**
Pie10P	3	60	404	441	321	310	368	**169**
	4	59	464	400	274	277	448	**169**
	2	21	**216**	365	396	408	306	235
ALLAML	3	20	231	361	396	392	329	**219**
	4	20	222	391	423	417	267	**197**

Tableau 4.7: Sommes des rangs obtenues par les méthodes de sélection d'attributs semi-supervisées ε^{SS}, EnsCLS, et SCGS sur les douze bases de données pour différents nombres de prototypes p.

Bases de données	p	$\hat{m}$	SCGS	EnsCLS	ε^{SS}
	2	7	201	234	**148**
WBCD	3	7	210	245	**127**
	4	7	204	244	**133**
	2	18	202	226	**164**
Image Segmentation	3	18	245	196	**150**
	4	18	264	183	**145**
	2	24	**150**	247	190
WDBC	3	25	172	231	**170**
	4	25	**147**	227	191
	2	6	247	**160**	192
Ionosphere	3	7	252	**156**	192
	4	6	268	**147**	178
	2	22	219	238	**136**
Dermatology	3	21	209	263	**120**
	4	21	210	268	**119**
	2	81	**141**	227	214
Libras Movement	3	85	**128**	229	222
	4	88	**162**	214	189
	2	22	201	299	**100**
Multiple Feature	3	23	203	297	**100**
	4	24	202	298	**100**
	2	235	**112**	300	187
CNAE-9	3	239	**111**	300	189
	4	234	**109**	300	191
	2	48	226	266	**102**
Yale	3	51	226	269	**100**
	4	53	227	260	**100**
	2	106	229	263	**100**
ORL	3	108	209	289	**100**
	4	109	200	300	**100**
	2	59	235	**126**	233
Pie10P	3	60	187	**168**	235
	4	59	**153**	214	224
	2	21	188	228	**173**
ALLAML	3	20	182	234	**167**
	4	20	179	252	**157**

4.3.4 Influence du paramètre σ

Les scores proposés ε^S et ε^{SS} dépendent du paramètre σ utilisé dans le calcul des matrices de similarité (équations (4.1) et (4.2)). Dans les expériences précédentes, nous avons fixé σ à 1. Pour étudier son influence, nous avons effectué des tests en variant σ de 0.2 à 1.4. La figure 4.5 affiche les taux moyens de classification obtenus par ε^S et ε^{SS} avec différentes valeurs de σ sur trois bases de données (`WBCD`, `WDBC`, et `Dermatology`). Les résultats sont moyennés sur 100 exécutions avec différents ensembles de prototypes (p étant fixé à 3). Les attributs sélectionnés sont évalués sur l'ensemble de test et chaque donnée de test est classée en utilisant la méthode 1NN avec uniquement les $k \cdot p$ prototypes utilisés lors de la sélection d'attributs. Cette figure montre que σ influe significativement sur les résultats de classification. Cependant, une valeur de σ comprise entre 0.6 et 1.2 convient bien à la plupart des bases de données testées.



(a) `WBCD` (b) `WDBC` (c) `Dermatology`

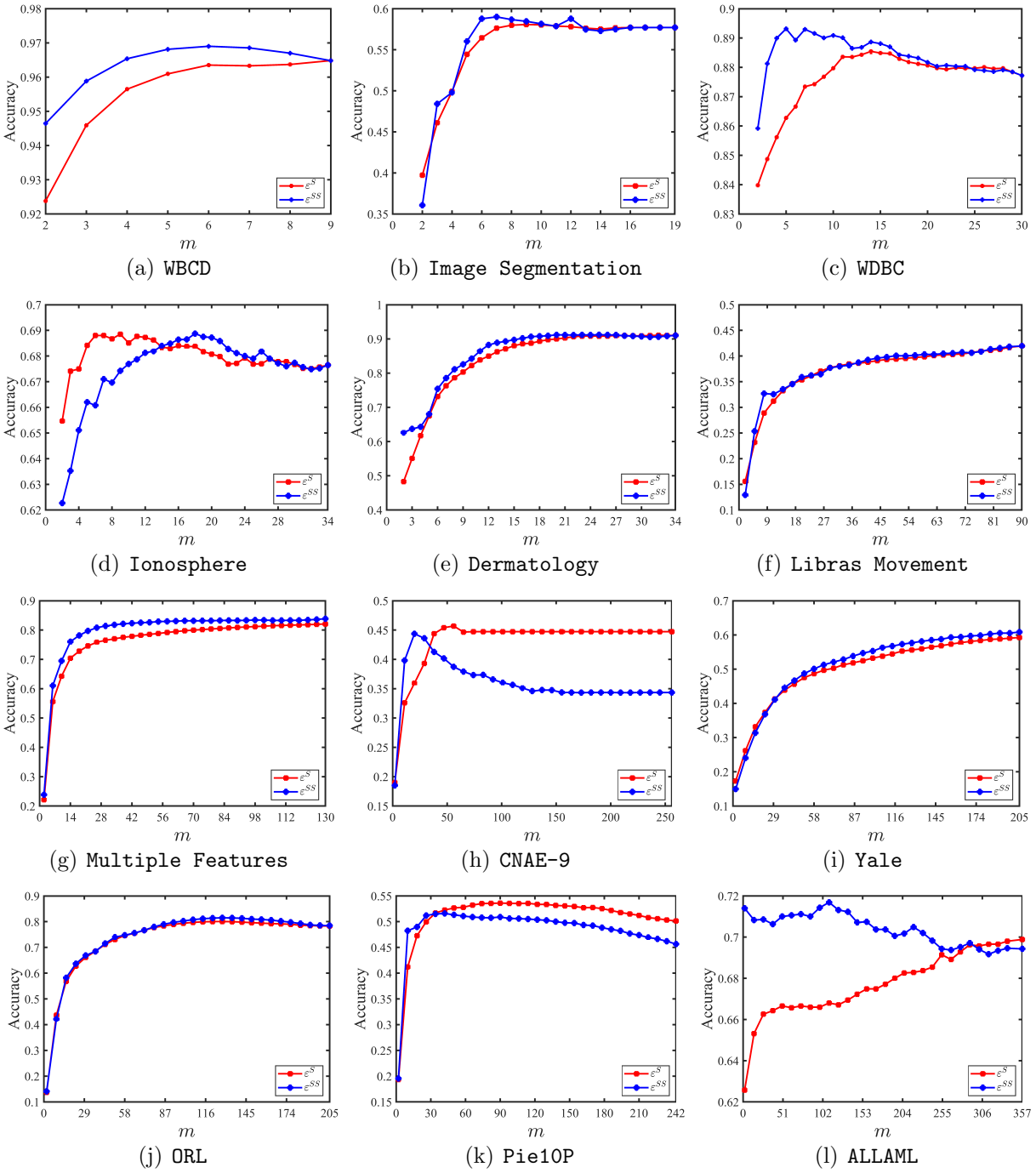
(d) `WBCD` (e) `WDBC` (f) `Dermatology`

Figure 4.5: Influence du paramètre σ sur les taux de classification obtenus en utilisant les attributs sélectionnés par les scores de contraintes ε^S (ligne du haut) et ε^{SS} (ligne du bas).

4.3.5 Comparaison des scores ε^S et ε^{SS}

Pour départager nos deux scores de contraintes ε^S et ε^{SS}, nous comparons leurs taux de classification obtenus sur les données de test en utilisant la méthode 1NN avec uniquement les $k \cdot p$ prototypes utilisés lors de la sélection d'attributs. La figure 4.6 illustre les taux de classification obtenus par les scores ε^S et ε^{SS} sur les douze bases de données. Les résultats

sont moyennés sur 100 exécutions avec différents ensembles de prototypes (p étant fixé à 3). Nous constatons que les taux de classification obtenus avec le score ε^{SS} sont meilleurs que ceux fournis par le score ε^{S} pour toutes les bases de données, à l'exception des bases `Ionosphere`, `CNAE-9`, et `Pie10P`. Ceci peut s'expliquer par la prise en compte d'un plus grand nombre de contraintes par ε^{SS} (grâce à M^{SS}).



(a) `WBCD` (b) `Image Segmentation` (c) `WDBC`

(d) `Ionosphere` (e) `Dermatology` (f) `Libras Movement`

(g) `Multiple Features` (h) `CNAE-9` (i) `Yale`

(j) `ORL` (k) `Pie10P` (l) `ALLAML`

Figure 4.6: Comparaison des taux moyens de classification en fonction du nombre m d'attributs sélectionnés par les scores proposés ε^{S} et ε^{SS} sur les douze bases de données.

4.3.6 Temps de calcul

Pour quantifier les temps de calcul requis par notre méthode de sélection d'attributs, nous comparons d'abord les temps de calcul de nos deux scores ε^S et ε^{SS} (avec tous les attributs et sans sélection séquentielle d'attributs) avec les temps consommés par C^1 et C^7. Ensuite, nous comparons le temps de calcul requis par la méthode de sélection d'attributs basée sur nos scores avec ceux requis par mRMR et EnsCLS pour un même nombre d'attributs sélectionnés ($m = 50$) et un même nombre de prototypes ($p = 3$).

Le tableau 4.8 présente les temps de calcul obtenus sur les bases de données `ORL`, `Multiple Features`, et `ALLAML`. Tous les tests ont été réalisés sur un ordinateur équipé d'un processeur Intel Core i7 à 3.60 GHz et d'une mémoire vive de 8 Go. D'après ce tableau, le temps de calcul de ε^S est inférieur à celui de C^1 lorsque le nombre de classes est petit ($k = 10$ pour `Multiple Features` et $k = 2$ pour `ALLAML`), alors qu'il est plus élevé que celui de C^1 lorsque le nombre de classes est élevé ($k = 40$ pour `ORL`). La complexité de calcul de ε^S dépend du nombre $k \cdot p$ de prototypes. De plus, malgré la sélection séquentielle, ε^S apparait plus rapide que la méthode mRMR dans le cas de la base `ALLAML`. En revanche, le temps de calcul de ε^{SS} reste supérieur à ceux de C^7 et EnsCLS. Le temps de calcul élevé de ε^{SS} est principalement dû au calcul des matrices de similarité, dont la complexité est proportionnelle à la taille de l'ensemble d'apprentissage.

Tableau 4.8: Temps de calcul (en secondes).

Bases de données	**Scores de contraintes**				**Méthodes de sélection d'attributs**			
	Supervisé		Semi-supervisé		Supervisé		Semi-supervisé	
	ε^S	C^1	ε^{SS}	C^7	ε^S	mRMR	ε^{SS}	EnsCLS
`Multiple Features`	0.015	0.400	6.788	2.286	2.142	0.329	418.99	5.488
`ORL`	0.133	0.016	0.367	0.057	7.925	0.614	16.361	2.752
`ALLAML`	0.013	0.033	0.217	0.091	1.045	3.417	8.163	0.102

4.3.7 Évaluation avec d'autres méthodes de classification

Dans les expériences précédentes, la pertinence des attributs sélectionnés est évaluée par la méthode 1NN sur l'ensemble de données test. Dans cette partie, nous évaluons le taux de classification obtenu avec d'autres méthodes de classification, à savoir SVM [127] et la classification spectrale (SC) (Algorithme 2.1). Il convient de noter que la méthode de classification spectrale, tout comme nos scores de contraintes, est également basée sur le

calcul des matrices de similarité. Pour compléter la comparaison, nous calculons les taux de classification obtenus par chaque méthode de classification avec les attributs sélectionnés par ε^S et C^1 d'une part, et ε^{SS} et C^7 d'autre part. Les tableaux 4.9 et 4.10 montrent que les taux de classification obtenus par ε^S et ε^{SS} sont toujours supérieurs à ceux obtenus par C^1 et C^7, quelle que soit la méthode de classification utilisée. Ces résultats illustrent la capacité de généralisation de nos scores de contraintes pour la sélection d'attributs.

Tableau 4.9: Taux moyens de classification obtenus par les méthodes 1NN, SVM, et SC en utilisant les attributs sélectionnés avec les scores supervisés ε^S et C^1.

Bases de données	1NN		SVM		SC	
	ε^S	C^1	ε^S	C^1	ε^S	C^1
WBCD	**96.35**	96.32	**96.82**	96.78	**96.74**	96.13
Image Segmentation	**57.70**	**57.70**	59.06	**59.07**	**56.92**	56.88
WDBC	87.97	**88.20**	91.21	**91.36**	**86.80**	86.56
Ionosphere	**68.73**	66.95	**66.92**	66.58	65.59	**65.78**
Dermatology	**90.85**	**90.85**	**90.16**	88.98	**90.58**	87.63
Libras Movement	**41.95**	**41.95**	**34.14**	34.12	**39.93**	**39.93**
Multiple Feature	**75.27**	74.36	**75.92**	73.39	**77.73**	72.27
CNAE-9	**44.29**	42.81	**47.78**	45.14	54.66	**57.91**
Yale	**49.05**	48.05	**48.75**	43.72	**49.52**	46.91
ORL	**79.91**	69.88	**70.13**	60.39	**78.11**	69.58
Pie10P	**53.26**	47.13	**54.16**	46.57	**56.28**	48.03
ALLAML	66.29	**67.51**	66.37	**67.37**	**73.51**	72.23

Tableau 4.10: Taux moyens de classification obtenus par les méthodes 1NN, SVM, et SC en utilisant les attributs sélectionnés avec les scores semi-supervisés ε^{SS} et C^7.

Bases de données	1NN		SVM		SC	
	ε^{SS}	C^7	ε^{SS}	C^7	ε^{SS}	C^7
WBCD	96.85	**96.92**	97.21	**97.42**	**96.97**	96.64
Image Segmentation	**57.70**	**57.70**	59.06	**59.07**	**56.89**	56.85
WDBC	87.92	**88.25**	91.42	**91.48**	**86.01**	85.91
Ionosphere	**67.10**	62.63	**68.08**	61.09	**63.60**	59.67
Dermatology	**91.16**	88.23	**90.20**	86.21	**91.20**	86.15
Libras Movement	**41.84**	40.92	**33.62**	32.99	**40.33**	39.02
Multiple Feature	**80.01**	73.97	**81.65**	73.51	**83.22**	71.27
CNAE-9	35.41	**41.74**	31.12	**36.26**	**52.84**	40.05
Yale	**48.72**	47.15	**48.53**	43.05	**50.20**	46.45
ORL	**80.72**	68.41	**71.27**	59.24	**78.57**	68.71
Pie10P	**51.05**	47.19	**52.75**	46.51	**54.98**	48.93
ALLAML	**70.69**	65.29	**72.37**	65.77	**75.26**	60.49

4.4 Conclusion

Dans ce chapitre, nous avons présenté une nouvelle méthode de sélection d'attributs basée sur un score de contraintes qui peut s'appliquer à la fois dans les contextes supervisé et semi-supervisé. Le score de contraintes supervisé ε^S évalue la capacité des attributs à respecter les contraintes, tandis que le score de contraintes semi-supervisé ε^{SS} évalue la capacité des attributs à respecter à la fois les contraintes initialement disponibles et celles déduites à partir des données étiquetées et non étiquetées. Les scores de contraintes proposés sont basés sur la théorie des graphes et offrent l'avantage d'évaluer la pertinence d'un sous-ensemble d'attributs à la fois. Ils permettent également de tenir compte de la corrélation entre les attributs. Un autre avantage de nos scores est leurs capacités à déterminer automatiquement le nombre optimal d'attributs.

Étant donné que les scores de contraintes proposés évaluent la similarité entre les données dans l'espace défini par le sous-ensemble d'attributs sélectionnés, les attributs choisis peuvent être avantageusement utilisés pour la classification. Les expériences menées sur douze bases de données bien connues dans le domaine de la sélection d'attributs montrent que les scores de contraintes proposés surpassent les scores de contraintes supervisés et semi-supervisés de l'état de l'art, ainsi que les méthodes de référence de sélection d'attributs.

Dans le prochain chapitre, nous exploiterons notre score de contraintes semi-supervisé dans le contexte de la segmentation d'images couleur texturées semi-supervisée.

Chapitre 5

Méthode de segmentation d'images couleur texturées proposée

5.1 Introduction

Nous décrivons dans ce chapitre la méthode de segmentation d'images couleur texturées semi-supervisée par classification des pixels que nous avons mise en œuvre [208]. Cette méthode est basée sur l'algorithme de classification spectrale classique, auquel de nombreuses étapes ont été ajoutées afin d'obtenir de meilleures performances de segmentation d'images. Il s'agit en particulier de la procédure de sélection d'attributs développée dans le chapitre précédent. L'information a priori disponible sous forme de quelques pixels prototypes est convertie en contraintes de type must-link et cannot-link. Ces contraintes sont ensuite intégrées dans deux phases de notre approche de segmentation d'images, à savoir la sélection des attributs de texture couleur et la classification des pixels.

Pour cela, nous avons organisé ce chapitre en deux parties. La première partie décrit la méthodologie de segmentation d'images couleur texturées développée. La deuxième partie est consacrée, dans un premier temps, à l'étude des performances du score de contraintes semi-supervisé dans la sélection des attributs de texture couleur. Dans un deuxième temps, une étude expérimentale est effectuée dans le but d'évaluer et de valider la méthode de segmentation d'images couleur texturées proposée. Une comparaison avec plusieurs méthodes de segmentation d'images (supervisées, non supervisées, et semi-supervisées) issues de la littérature sur cinq différentes bases d'images (textures couleur, textures niveaux de gris, satellitaires, médicales, et naturelles) est réalisée.

5.2 Méthode de segmentation d'images proposée

La méthode de segmentation d'images, nommée "constrained feature selection-spectral clustering" (CFS-SC) [208], est essentiellement basée sur la sélection d'attributs et sur la classification spectrale semi-supervisées sous contraintes. Elle utilise l'information a priori dans le but de guider le processus de segmentation et aboutir à de meilleures performances. Cette information a priori est exprimée par quelques pixels prototypes représentant chaque texture couleur. À partir de ces pixels prototypes, il est facile de générer des contraintes M et C, puis de s'en servir dans la sélection d'attributs et dans la classification spectrale. La figure 5.1 donne une vue globale de la méthode de segmentation d'images proposée.

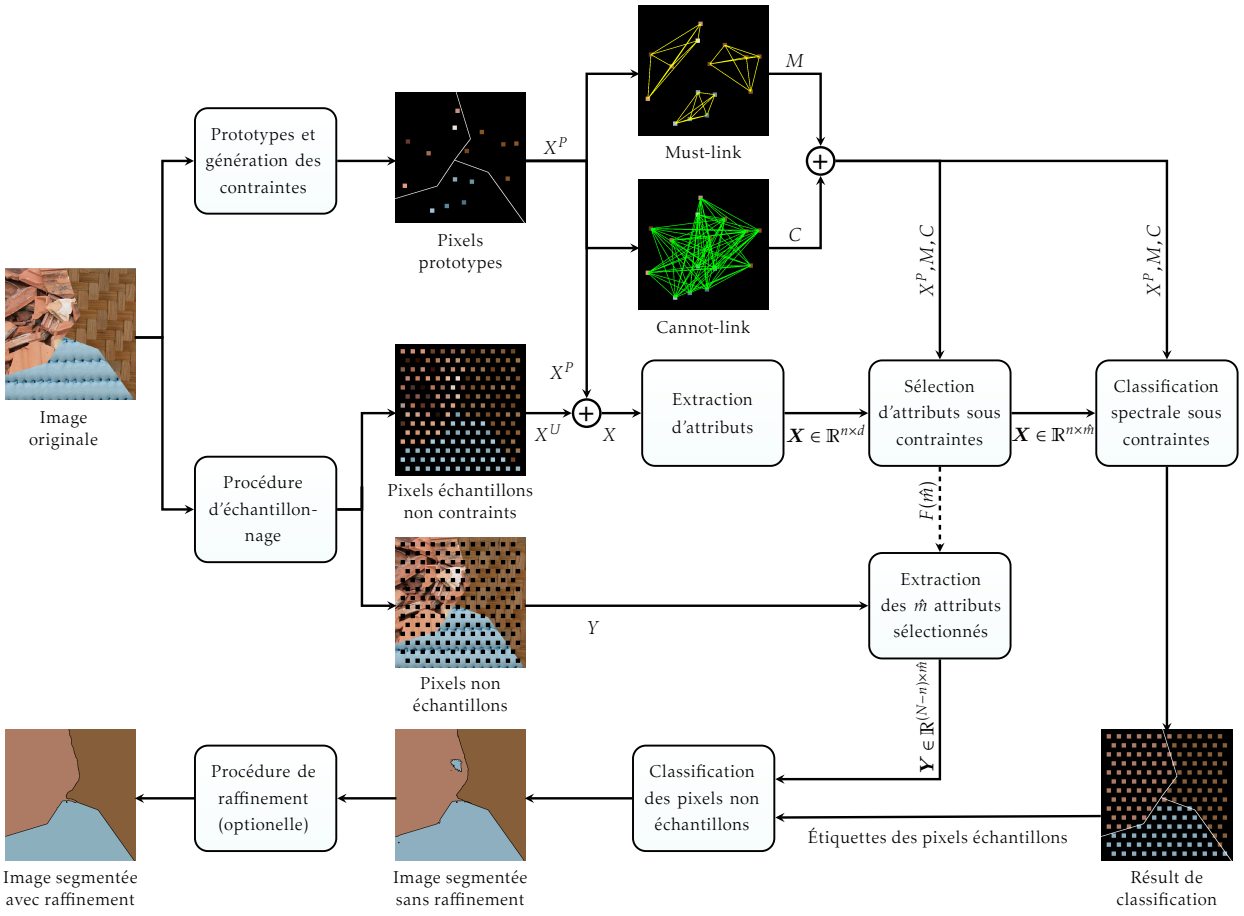


Figure 5.1: Schéma illustrant la méthode de segmentation d'images CFS-SC.

Dans CFS-SC, chaque pixel de l'image est caractérisé par un ensemble d'attributs de texture couleur. La méthode de classification spectrale qui analyse une matrice de similarité peut être avantageusement utilisée pour effectuer une segmentation semi-supervisée sous contraintes. Étant donné que ces dernières représentent les similarités entre pixels, elles peuvent être facilement intégrées dans une matrice de similarité. Parmi les attributs extraits, certains d'entre eux peuvent être non pertinents et peu informatifs. Par conséquent, le calcul des

similarités qui est basé sur le calcul de la distance euclidienne dans l'espace de tous les attributs est aussi corrompu et les performances de la classification spectrale peuvent être dégradées. Pour contourner ce problème, une sélection d'attributs est effectuée avant le processus de classification par notre méthode de sélection basée sur le score de contraintes semi-supervisé qui est lui-même basé sur le concept des matrices de similarité. Dans ce qui suit, nous détaillons chacune des étapes de la méthode de segmentation CFS-SC.

5.2.1 Génération des pixels prototypes et des contraintes

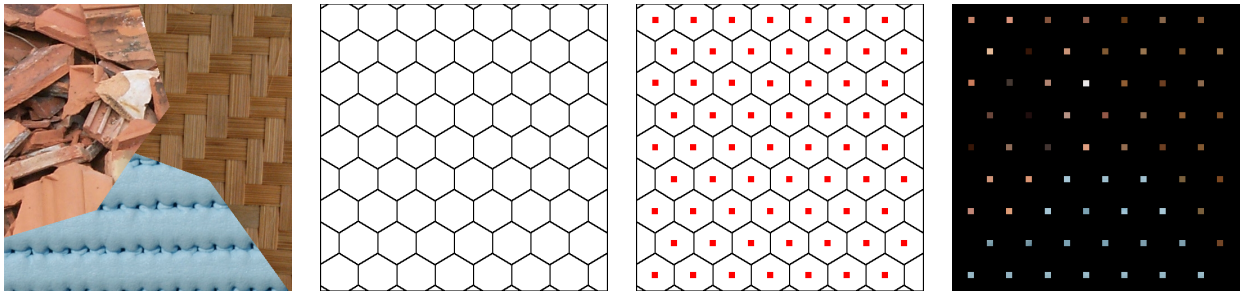
Vu que la méthode CFS-SC est effectuée dans un contexte d'apprentissage semi-supervisé, l'information a priori disponible est exprimée sous forme de quelques pixels prototypes caractérisant les k régions de l'image. Ces pixels prototypes peuvent être choisis de manière interactive par l'utilisateur sur l'image à segmenter ou générés aléatoirement à partir de l'image de référence. Dans cette thèse, nous avons opté pour la deuxième solution afin d'analyser les résultats de manière objective indépendamment du choix de l'utilisateur.

On note par X^l l'ensemble des p pixels prototypes associés à la région ω^l, $l = 1, \cdots, k$ et par X^P l'ensemble de tous les pixels prototypes tel que $X^P = \bigcup_{l=1}^{k} X^l$. À partir de cet ensemble de prototypes, nous construisons deux ensembles de contraintes must-link et cannot-link en utilisant respectivement les équations (2.28) et (2.29). $(k \cdot p \cdot (p-1))$ contraintes must-link et $(k \cdot (k-1) \cdot p^2)$ contraintes cannot-link sont ainsi générées pour chaque texture couleur.

5.2.2 Échantillonnage

Appliquer la classification spectrale dans le cadre de la segmentation d'images n'est pas une tâche facile, notamment lorsque le nombre N de pixels dans l'image est très élevé. En effet, les coûts de calcul élevés de la matrice de similarité $O(N^2)$ et de la décomposition spectrale de la matrice laplacienne $O(N^3)$ rendent la classification spectrale difficilement adaptable pour la segmentation. Pour pallier ce problème, plusieurs solutions sont proposées dans la littérature. Elles peuvent être classées en deux catégories: la première utilise des approximations matricielles comme celle de Nyström [91, 209] ou des matrices de similarité creuses [210, 211]. La seconde consiste à remplacer l'ensemble des pixels de l'image par un autre plus petit (points représentatifs) en utilisant soit la sélection des pixels de l'image [212, 213] ou la décomposition de l'image en superpixels [214, 215]. Généralement, les méthodes de la deuxième classe sont plus pratiques que celles de la première classe car elles réduisent les coûts de calcul de toutes les étapes de la méthode de classification spectrale.

Dans notre travail, nous avons opté pour une méthode simple et rapide qui appartient à la deuxième catégorie. Celle-ci permet de réduire considérablement le nombre de pixels à classer. Elle consiste à appliquer d'abord une grille régulière de structure hexagonale sur l'image à segmenter, puis à sélectionner les pixels correspondants aux coordonnées spatiales sur l'image des centres d'hexagones. La figure 5.2 illustre le principe de cette méthode.



(a) (b) (c) (d)

Figure 5.2: Illustration du principe de la méthode de sélection de pixels. (a) Image originale, (b) grille hexagonale, (c) centres d'hexagones marqués en rouge, et (d) pixels échantillons.

Les N pixels de l'image couleur I sont décomposés en deux sous-ensembles:

- *Sous-ensemble des pixels échantillons* $X = \{x_1, x_2, \ldots, x_n\}$ qui regroupe à la fois le sous-ensemble X^P composé des $(k \cdot p)$ pixels prototypes et le sous-ensemble X^U composé des $(n - (k \cdot p))$ pixels échantillons non étiquetés sélectionnés en appliquant une grille de structure hexagonale sur l'image I, où $n \ll N$.
- *Sous-ensemble des pixels non échantillons* (*out-of-sample pixels*) $Y = \{y_1, y_2, \ldots, y_{(N-n)}\}$ qui regroupe les $(N - n)$ pixels restants.

5.2.3 Extraction des attributs

Afin d'assurer une bonne segmentation des images couleur texturées, il est plus intéressant de tirer le maximum d'informations de l'image à segmenter. La multiplicité des informations permet une bonne caractérisation des pixels et, par conséquent, une séparabilité facile des régions de l'image. Pour ce faire, chaque pixel échantillon est caractérisé par un large ensemble d'attributs de texture couleur. De cette façon, la méthode de segmentation d'images développée ne sera pas dépendante d'un seul type d'attributs.

Les attributs de texture couleur choisis pour la caractérisation des pixels dans notre méthode de segmentation CFS-SC sont les suivants :

Attributs de Haralick: Ils décrivent les corrélations entre les pixels voisins et sont calculés à partir des matrices de co-occurrences chromatiques (Annexe A). Dans une image couleur I, chaque pixel x est représenté par trois composantes de couleur R, G, et B. Les six matrices de co-occurrences chromatiques ($\mathcal{M}^{R,R}[x](i,j)$, $\mathcal{M}^{G,G}[x](i,j)$, $\mathcal{M}^{B,B}[x](i,j)$, $\mathcal{M}^{R,G}[x](i,j)$, $\mathcal{M}^{R,B}[x](i,j)$, et $\mathcal{M}^{G,B}[x](i,j)$) sont calculées localement pour chaque pixel x en considérant toutes les co-occurrences dans une fenêtre de voisinage de taille $(2s+1) \times (2s+1)$ centrée sur le pixel x (Figure 5.3). Pour chacune des matrices de co-occurrences, nous calculons quatorze attributs de Haralick.

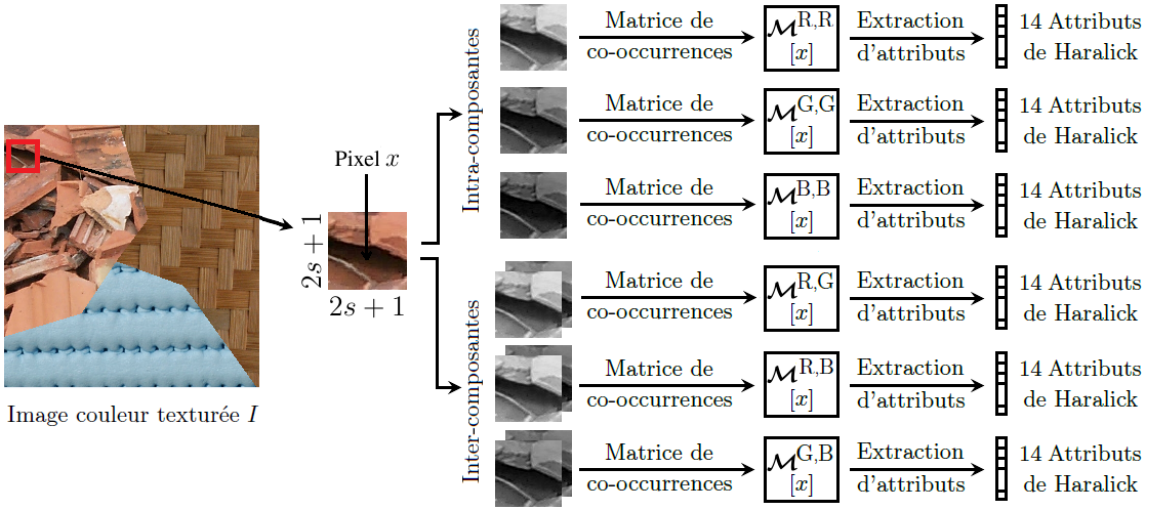


Figure 5.3: Illustration du calcul des attributs de Haralick.

Attributs basés sur les filtres de Gabor: Ils sont générés à partir de la convolution des filtres de Gabor sur chacune des composantes de l'image couleur I [22, 216]. Tout comme dans [22], pour extraire les attributs de texture, nous considérons sept échelles ($\gamma^\star = 2\sqrt{2}$, $4\sqrt{2}$, $8\sqrt{2}$, $16\sqrt{2}$, $32\sqrt{2}$, $64\sqrt{2}$, et $128\sqrt{2}$) et quatre orientations ($\theta^\star = 0°$, $45°$, $90°$, et $135°$) pour chacune des trois composantes de couleur R, G, et B (avec * = R,G, ou B). Il en découle vingt-huit ($7 \times 4 = 28$) attributs de texture pour chacune des composantes couleur.

En plus des attributs de texture couleur, nous considérons des attributs de couleur représentés par les valeurs des composantes issues de quatre espaces de couleur, à savoir RGB, HSV, XYZ, et L$^*a^*b^*$ (Annexe A). Ce choix est motivé par le fait que dans la segmentation d'images, il n'existe pas un espace de couleur adapté à toutes les applications et à toutes les images. Bien au contraire, le choix d'un espace est d'abord lié au type d'images à traiter [10, 217].

Au final, chaque pixel échantillon est caractérisé par un ensemble d'attributs $F_m = \{f_1, \ldots, f_d\}$ (avec $d = 180$ dont 84 attributs de texture couleur de Haralick, 84 attributs de texture couleur issus des filtres de Gabor, et 12 attributs de couleur). Chacun des d attributs est normalisé entre 0 et 1 afin que l'échelle des valeurs des attributs soit la même.

5.2.4 Sélection d'attributs sous contraintes

Parmi l'ensemble des attributs que nous avons extraits, certains d'entre eux ne sont pas pertinents et n'apportent aucune information supplémentaire pour la méthode de classification. L'utilisation de ces attributs pourrait même dégrader les performances de la méthode de segmentation d'images. Pour éviter ce problème, nous appliquons une procédure de sélection d'attributs afin d'identifier puis de supprimer les attributs non pertinents. Cette procédure permet de sélectionner les attributs les plus pertinents. Un meilleur choix de l'ensemble des attributs permet non seulement d'améliorer la performance de la méthode de classification, mais aussi de réduire le temps de calcul et la taille de la mémoire utilisée pour le stockage.

Dans notre approche de segmentation d'images CFS-SC, nous proposons d'appliquer notre méthode de sélection d'attributs basée sur le score de contraintes semi-supervisé ε^{SS} (définie dans la section 4.2 [205]). Ce choix est motivé à la fois par le contexte semi-supervisé de la méthode de segmentation d'images proposée et par plusieurs avantages conclus à partir du chapitre précédent, que nous rappelons ci-dessous:

- La supériorité des performances de la méthode ε^{SS} par rapport aux méthodes de l'état de l'art en matière de sélection d'attributs.
- La capacité de notre score de contraintes ε^{SS} à évaluer la pertinence d'un sous-ensemble d'attributs à la fois contrairement aux méthodes de l'état de l'art sur la sélection d'attributs basées sur les scores de contraintes qui ignorent les corrélations entre les attributs.
- La capacité de notre méthode de sélection d'attributs ε^{SS} à identifier automatiquement le nombre optimal d'attributs.
- La supériorité de ε^{SS} par rapport à ε^{S} dans un même contexte d'apprentissage.

5.2.5 Classification des pixels échantillons

Une fois que le sous-ensemble d'attributs pertinents $F_{\hat{m}}$ a été sélectionné, l'ensemble des n pixels échantillons X sont regroupés en k classes via la méthode de classification spectrale sous contraintes décrite dans l'algorithme 2.2. Dans cette méthode, nous proposons de remplacer la méthode de classification non supervisée k-means utilisée pour classer les données projetées dans l'espace défini par les k vecteurs propres dominants par une méthode de classification supervisée, en l'occurrence la méthode 1NN. La méthode 1NN permet d'exploiter les k sous-ensembles de pixels prototypes disponibles X^l, $l = 1, \ldots, k$.

De plus, la matrice de similarité $\boldsymbol{W}^{M,C}$ définie dans l'équation (2.30) est calculée dans l'espace des $\hat{m}$ attributs sélectionnés comme suit:

$$w_{ij}^{M,C}(F_{\hat{m}}) = \begin{cases} 1 & \text{si } \left(x_i,x_j\right) \in M \\ 0 & \text{si } \left(x_i,x_j\right) \in C \\ \texttt{exp}\left(-\frac{\delta^2(\boldsymbol{x}_i^{(\hat{m})},\boldsymbol{x}_j^{(\hat{m})})}{2\sigma^2}\right) & \text{sinon.} \end{cases} \qquad i,j = 1,2,\ldots,n \tag{5.1}$$

Le paramètre σ est fixé à 1 pour toutes les expériences de ce chapitre.

5.2.6 Classification des pixels non échantillons

La dernière étape de la méthode de segmentation d'images CFS-SC consiste à affecter chaque pixel non échantillon (out-of-sample) y_i, $y_i \subset Y$ où $i = 1,\ldots,N-n$, aux classes des pixels échantillons qui ont été déterminées précédemment par la méthode de classification spectrale sous contraintes (Section 5.2.5). Chaque pixel non échantillon y_i est caractérisé par les $\hat{m}$ attributs pertinents $F_{\hat{m}}$.

Généralement, cette étape est réalisée en attribuant chaque pixel non échantillon y_i à la classe majoritaire de ses plus proches pixels échantillons $x_i \in X$ dans l'espace des $\hat{m}$ attributs sélectionnés [218]. Dans notre cas, nous proposons de classer les pixels non échantillons dans le même espace d'attributs dans lequel les n pixels échantillons ont été classés. Pour cela, nous projetons les pixels non échantillons y_i dans un espace spectral de k-dimensions qui est engendré par les k vecteurs propres dominants $\boldsymbol{u}_l$ correspondant aux k plus petites valeurs propres λ_l de la matrice laplacienne $\boldsymbol{L}_{Sym}^{M,C}$, calculée sur l'ensemble des pixels échantillons X [219].

La projection d'un pixel non échantillon y_i dans l'espace engendré par le l-ème vecteur propre dominant est définie par:

$$z_l(y_i) = \frac{1}{(1-\lambda_l)} \sum_{j=1}^{n} u_{lj} \tilde{w}^{(\hat{m})}(\boldsymbol{y}_i,\boldsymbol{x}_j) \qquad l = 1,\ldots,k \tag{5.2}$$

où $\tilde{w}^{(\hat{m})}(\boldsymbol{y}_i,\boldsymbol{x}_j)$ est la similarité normalisée équivalente entre le pixel non échantillon y_i et le pixel échantillon x_j telle que:

$$\tilde{w}^{(\hat{m})}(\boldsymbol{y}_i,\boldsymbol{x}_j) = \frac{1}{n} \frac{w^{(\hat{m})}(\boldsymbol{y}_i,\boldsymbol{x}_j)}{\sqrt{E\left[w^{(\hat{m})}(\boldsymbol{y}_i,\boldsymbol{X})\right] E\left[w^{(\hat{m})}(\boldsymbol{x}_j,\boldsymbol{X'})\right]}} \tag{5.3}$$

$E[\cdot]$ représente l'opérateur moyen tel que $E\left[w^{(\hat{m})}(\boldsymbol{y}_i,\boldsymbol{X})\right] = \frac{1}{n}\sum_j^n w^{(\hat{m})}(\boldsymbol{y}_i,\boldsymbol{x}_j)$.

La similarité $w^{(\hat{m})}(\boldsymbol{y}_i,\boldsymbol{x}_j)$ entre le pixel non échantillon y_i et le pixel échantillon x_j est définie comme suit:

$$w^{(\hat{m})}(\boldsymbol{y}_i,\boldsymbol{x}_j) = \texttt{exp}\left(-\frac{\delta^2\left(\boldsymbol{y}_i^{(\hat{m})},\boldsymbol{x}_j^{(\hat{m})}\right)}{2\sigma^2}\right) \tag{5.4}$$

Le pixel non échantillon y_i projeté dans le sous-espace de dimension k est normalisé à partir de la formule ci-dessous:

$$\tilde{z}_l(y_i) = \frac{z_l(y_i)}{\sqrt{\sum_{j=1}^{k} z_j(y_i)^2}} \qquad l = 1,\ldots,k. \tag{5.5}$$

Finalement, chaque pixel non échantillon y_i est assigné à la classe du pixel échantillon le plus proche de $\tilde{z}(y_i)$ dans le sous-espace de dimension k.

5.2.7 Procédure de raffinement

Il est possible d'améliorer la qualité de la segmentation en appliquant un post-traitement à l'image segmentée obtenue précédemment. En effet, certaines régions de l'image segmentée peuvent s'avérer trop petites pour constituer des régions d'intérêt, ce qui augmente le nombre total de régions détectées. Afin d'améliorer la qualité des images segmentées, nous proposons de supprimer les petites régions insignifiantes en les fusionnant avec les plus grandes régions adjacentes, comme le font plusieurs méthodes de segmentation proposées dans la littérature [60, 61, 103, 105, 112].

La procédure de fusion des petites régions appliquée dans la méthode de segmentation d'images couleur texturées CFS-SC proposée consiste à maintenir toutes les régions qui couvrent au minimum $T\%$ du nombre total N de pixels de l'image. Une région dont la surface est inférieure à ce seuil est fusionnée avec la région voisine la plus grande et ré-étiquetée avec son indice de classe. La stratégie itérative de fusion des petites régions se poursuit tant qu'il existe une région dont la surface est inférieure au seuil $T\%$. À l'issue des itérations, toutes les petites régions sont fusionnées. La fusion de régions s'opère à l'aide d'un graphe d'adjacence des régions (RAG) [220]. Dans nos expériences, la valeur du seuil T est fixée à 0.5.

Au final, les principales étapes de la méthode de segmentation CFS-SC proposée sont résumées dans l'algorithme 5.1.

Algorithme 5.1: CFS-SC.

Entrée: Image I, nombre de classes k, nombre de pixels échantillons n, nombre de pixels prototypes par classe p, paramètre de Haralick s, paramètres de Gabor (θ,γ).

1. Générer l'ensemble des pixels prototypes X^P puis déduire les ensembles de contraintes M et C.
2. Générer les sous-ensembles de pixels échantillons X et non échantillons Y.
3. Calculer d attributs de texture couleur pour chaque pixel échantillon de X.
4. Sélectionner les $\hat{m}$ attributs les plus pertinents $F_{\hat{m}}$ sur X en utilisant notre méthode de sélection d'attributs sous contraintes (Algorithme 4.1).
5. Calculer la matrice de similarité $\boldsymbol{W}^{M,C}(F_{\hat{m}}) \in \mathbb{R}^{n\times n}$ de X (équation (5.1)).
6. Calculer la matrice laplacienne normalisée $\boldsymbol{L}_{Sym}^{M,C}$ associée à $\boldsymbol{W}^{M,C}(F_{\hat{m}})$.
7. Extraire les k vecteurs propres $\boldsymbol{U}^{M,C} = [\boldsymbol{u}_1,\ldots,\boldsymbol{u}_k] \in \mathbb{R}^{n\times k}$ de $\boldsymbol{L}_{sym}^{M,C}$ correspondants aux k plus petites valeurs propres $\boldsymbol{\lambda}^{M,C} = [\lambda_1,\ldots,\lambda_k] \in \mathbb{R}^k$ de $\boldsymbol{L}_{sym}^{M,C}$.
8. Construire la matrice $\boldsymbol{T}^{M,C} \in \mathbb{R}^{n\times k}$ à partir de $\boldsymbol{U}^{M,C}$ en normalisant les lignes de $\boldsymbol{U}^{M,C}$ comme suit:
$$t_{ij} = \frac{u_{ij}}{\sqrt{\sum_{j=1}^{k} u_{ij}^2}}.$$
9. Définir chaque ligne de $\boldsymbol{T}^{M,C}$ comme un point, puis classer ces points en k classes en utilisant la méthode du 1NN avec les p pixels prototypes de chaque classe.
10. Pour chaque pixel non échantillon $y_i \in Y$:
    - Calculer les $\hat{m}$ attributs les plus pertinents.
    - Calculer le noyau normalisé $\tilde{w}$ défini par l'équation (5.3).
    - Calculer le point projeté $\boldsymbol{z}(y_i)$ défini par l'équation (5.2).
    - Calculer le point projeté normalisé $\tilde{\boldsymbol{z}}(y_i)$ en utilisant l'équation (5.5).
    - Assigner y_i à la classe du plus proche pixel échantillon, parmi $\boldsymbol{T}^{M,C}$, de $\tilde{\boldsymbol{z}}(y_i)$.
11. Procédure de raffinement (optionnelle).

Sortie: Image segmentée.

5.3 Évaluation de la méthode CFS-SC

Dans cette section, nous évaluons les performances de la méthode de segmentation d'images CFS-SC proposée. Nous étudions d'abord les performances du score de contraintes semi-supervisé ε^{SS} dans la sélection des attributs de texture couleur. Ensuite, nous présentons les résultats de segmentation d'images obtenus par la méthode CFS-SC et les comparons avec ceux de plusieurs méthodes de l'état de l'art sur cinq bases d'images.

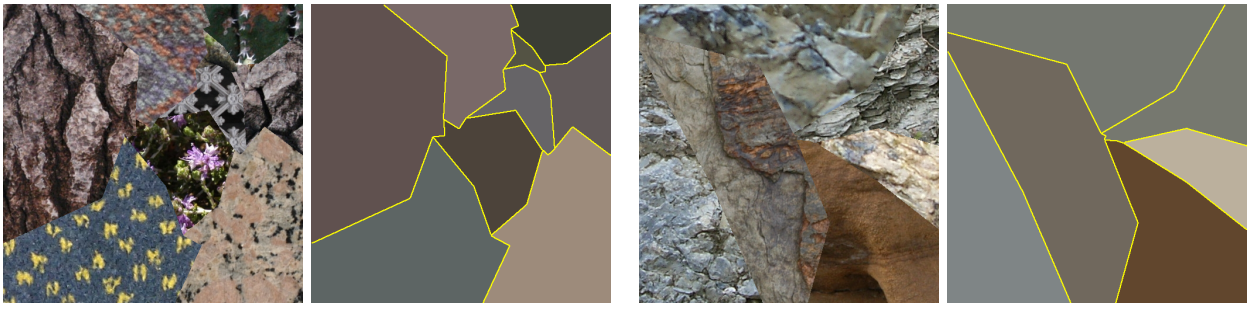
5.3.1 Évaluation du score de contraintes semi-supervisé ε^{SS}

Pour évaluer les performances du score de contraintes semi-supervisé ε^{SS} dans la sélection des attributs de texture couleur, nous adoptons la même démarche que celle utilisée dans le chapitre précédent (Section 4.3). Les résultats obtenus avec le score de contraintes semi-supervisé ε^{SS} sont comparés à ceux obtenus avec les scores de contraintes semi-supervisés C^3, C^4, C^5, C^6, et C^7.

Les procédures de sélection d'attributs sont réalisées sur l'ensemble des pixels échantillons X et répétées 100 fois. Pour chaque exécution, nous générons aléatoirement $(k \cdot p)$ pixels prototypes à partir de chaque image de référence (dans nos expériences p varie de 2 à 5), puis nous dérivons les ensembles de contraintes M, C, et M^{SS}. Pour le calcul des attributs de Haralick, la taille de la fenêtre de voisinage s est fixée à 11. Pour l'évaluation des attributs sélectionnés, les mêmes pixels prototypes utilisés pour la sélection des attributs sont utilisés par la classification spectrale sous contraintes.

La procédure d'échantillonnage décrite dans la section 5.2.2, permet d'extraire $n - (k \cdot p)$ pixels échantillons non étiquetés de chaque image. Dans nos expériences, nous choisissons de tester plusieurs valeurs pour le nombre n de pixels échantillons ($n = 200$, 300, et 400) afin d'étudier son influence sur la qualité des attributs sélectionnés. La performance des scores de contraintes est mesurée par le taux de classification obtenu avec la méthode de classification spectrale sous contraintes (décrite dans la section 5.2.5) sur l'ensemble des pixels échantillons non étiquetés X^U.

Pour mener ces premières expériences, nous avons sélectionné deux images de la base Prague [221] (Figure 5.4). Ces deux images sont utilisées en vue d'analyser les paramètres de la méthode de segmentation CFS-SC.

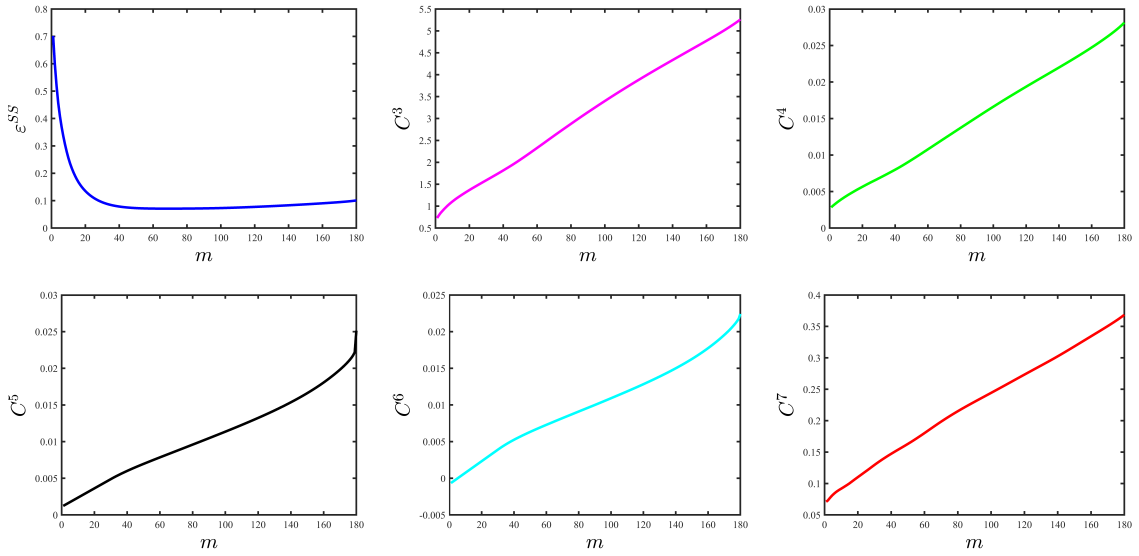


(a) Image 7_1_1 (b) Image 17_1_1

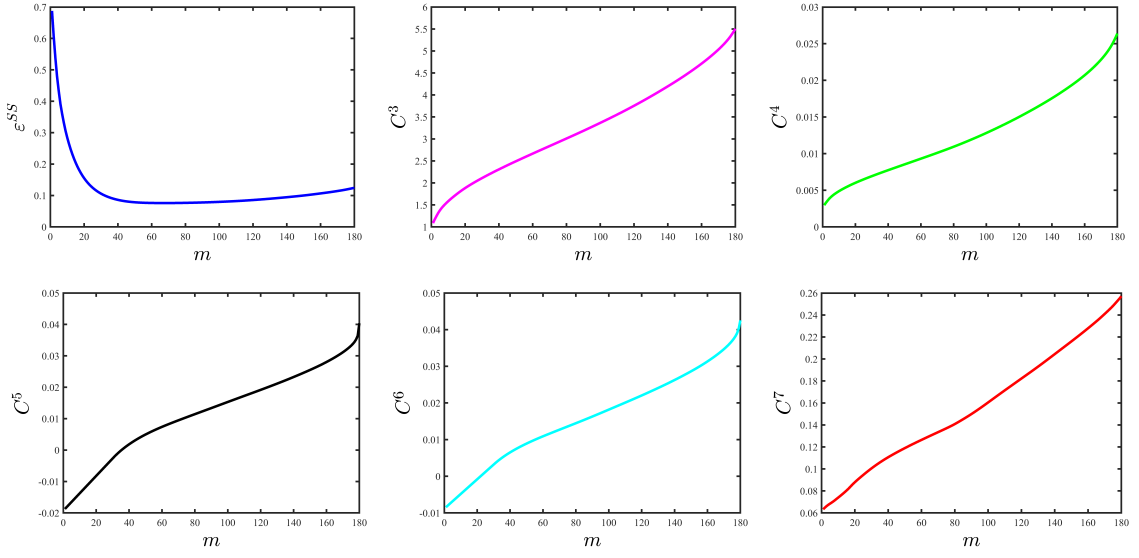
Figure 5.4: Images couleur texturées de test et leur images de références correspondantes issues de la base d'images Prague.

5.3.1.1 Scores semi-supervisés en fonction du nombre d'attributs

La figure 5.5 présente la variation des scores de contraintes semi-supervisés ε^{SS}, C^3, C^4, C^5, C^6, et C^7 en fonction du nombre m d'attributs sélectionnés pour des valeurs fixes de p et n ($p = 5$ et $n = 300$). À partir de cette figure, nous confirmons les résultats obtenus dans le chapitre 4 à savoir que les courbes de ε^{SS} sont quasi-convexes, alors que celles de C^3, C^4, C^5, C^6, et C^7 augmentent de manière monotone en fonction du nombre d'attributs sélectionnés. Des résultats similaires, non présentés dans ce manuscrit par souci d'encombrement, sont observés pour d'autres valeurs de p et de n. Ainsi, nous pouvons affirmer que le score ε^{SS} présente une valeur minimale qui peut être considérée comme le nombre optimal d'attributs.



(a) Image 7_1_1

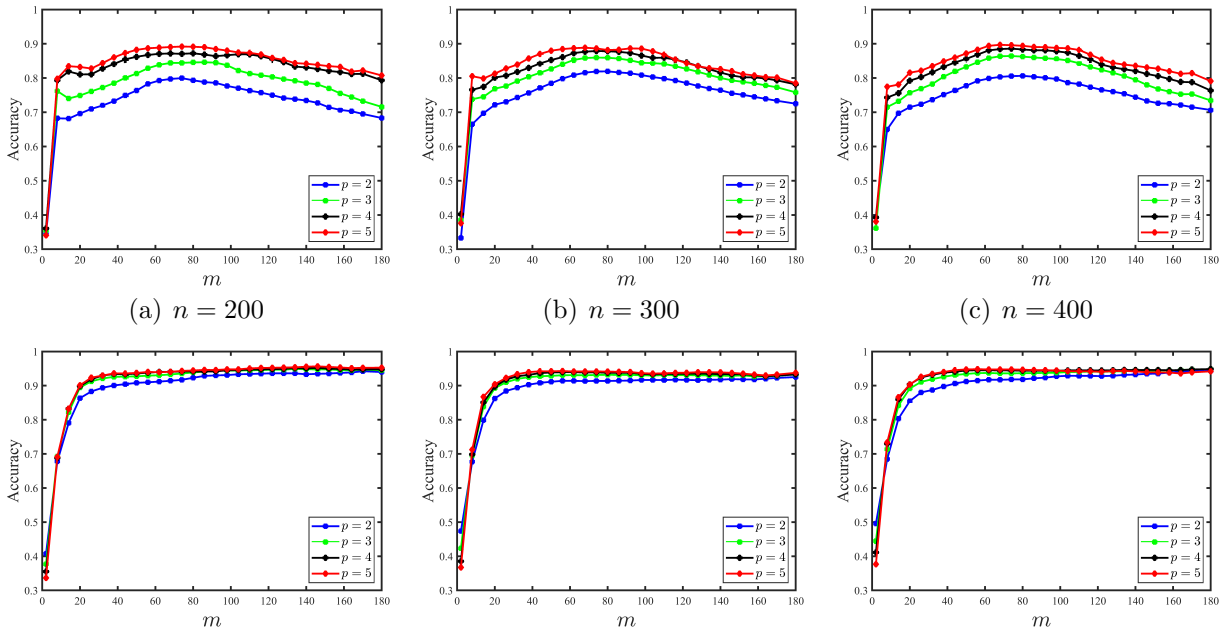


(b) Image 17_1_1

Figure 5.5: Performances des scores de contraintes semi-supervisés ε^{SS}, C^3, C^4, C^5, C^6, et C^7 en fonction du nombre m d'attributs sélectionnés obtenues sur les deux images tests.

5.3.1.2 Influence du nombre de prototypes

Le nombre de prototypes par classe p est étroitement lié au nombre de contraintes et influe sur le résultat de la classification. La figure 5.6 illustre les taux moyens de classification obtenus par ε^{SS} sur 100 exécutions en fonction du nombre d'attributs sélectionnés pour différentes valeurs de p et n (p varie de 2 à 5 et n de 200 à 400). Comme attendu, nous constatons que, pour un nombre n donné, les courbes de variation des taux moyens de classification obtenus par ε^{SS} s'améliorent en augmentant le nombre p.



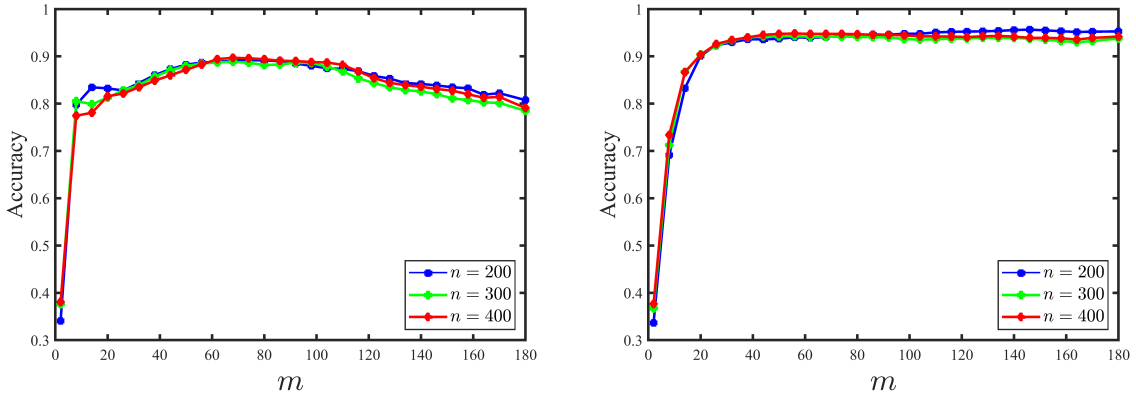
(a) $n = 200$ (b) $n = 300$ (c) $n = 400$

(d) $n = 200$ (e) $n = 300$ (f) $n = 400$

Figure 5.6: Taux moyens de classification en fonction du nombre m d'attributs sélectionnés par ε^{SS} pour différentes valeurs de n et p. La 1-ère ligne montre les résultats obtenus sur l'image 7_1_1 et la 2-ème ligne montre les résultats obtenus sur l'image 17_1_1.

5.3.1.3 Influence du nombre de pixels échantillons

Le nombre de pixels échantillons n est lié à l'échantillonnage de l'image. Il doit être suffisamment grand d'une part pour atteindre une meilleure définition spatiale des régions de l'image, et suffisamment petit d'autre part pour réduire la complexité de calcul de la classification spectrale sous contraintes. Pour satisfaire ce compromis, nous avons fait varier n de 200 à 400, ce qui correspond en moyenne à 0.10% du nombre N de pixels de l'image. La figure 5.7 affiche les variations des taux moyens de classification obtenus avec les attributs sélectionnés par le score semi-supervisé ε^{SS} (avec $p = 5$). Elle montre que les courbes obtenues avec $n = 200$, 300, et 400 se chevauchent. Cela signifie que les taux moyens de classification ne changent pas de manière significative lorsque n varie de 200 à 400.

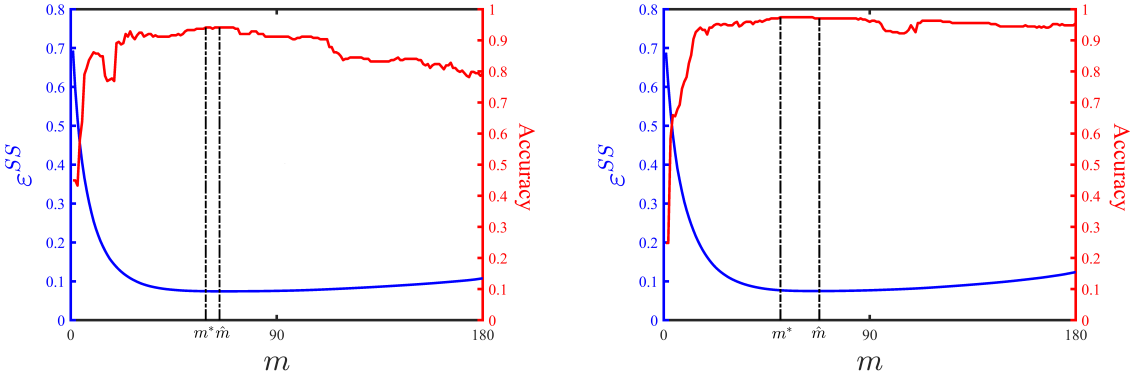


(a) Image 7_1_1 (b) Image 17_1_1

Figure 5.7: Taux moyens de classification en fonction du nombre m d'attributs sélectionnés avec le score semi-supervisé ε^{SS} sur les deux images de test pour différents nombres n.

5.3.1.4 Nombre optimal d'attributs sélectionné par ε^{SS}

Comme indiqué dans la section 5.3.1.1, la courbe du score de contraintes ε^{SS} en fonction du nombre d'attributs sélectionnés présente un minimum qui peut être considéré comme le nombre optimal d'attributs. Pour étayer cette assertion, nous affichons simultanément sur la figure 5.8 les courbes du score ε^{SS} et du taux de classification en fonction du nombre d'attributs sélectionnés (pour $n = 300$ et $p = 5$). Nous constatons que le taux de classification $Acc_{(\hat{m})}$ obtenu avec le sous-ensemble d'attributs sélectionnés $F_{\hat{m}}$ qui correspond au minimum du score ε^{SS} coïncide ou proche de la valeur maximale du taux de classification $Acc_{(m^*)}$.



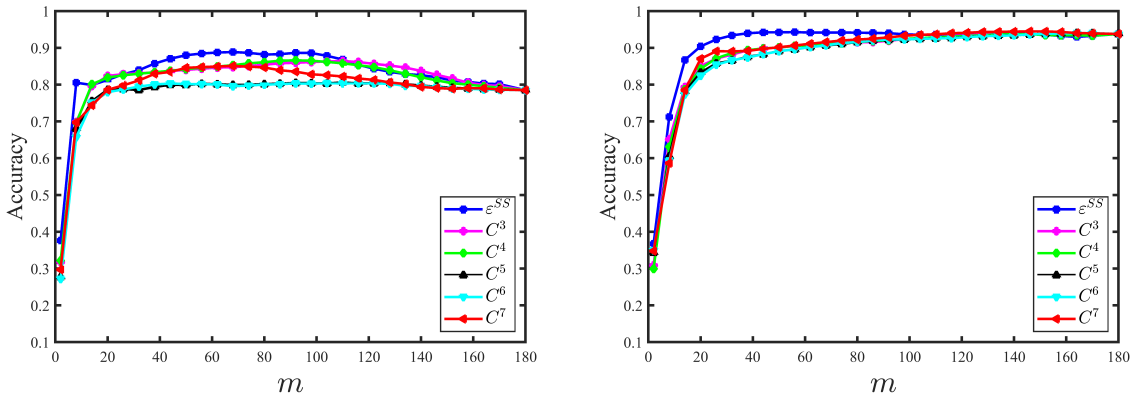
(a) Image 7_1_1 (b) Image 17_1_1

Figure 5.8: Variation du taux de classification et du score semi-supervisé ε^{SS} en fonction du nombre d'attributs sélectionnés pour les deux images de test.

5.3.1.5 Comparaison avec les scores de contraintes semi-supervisés

Nous allons à présent comparer les taux de classification obtenus par notre score semi-supervisé ε^{SS} avec ceux obtenus par les scores C^3, C^4, C^5, C^6, et C^7.

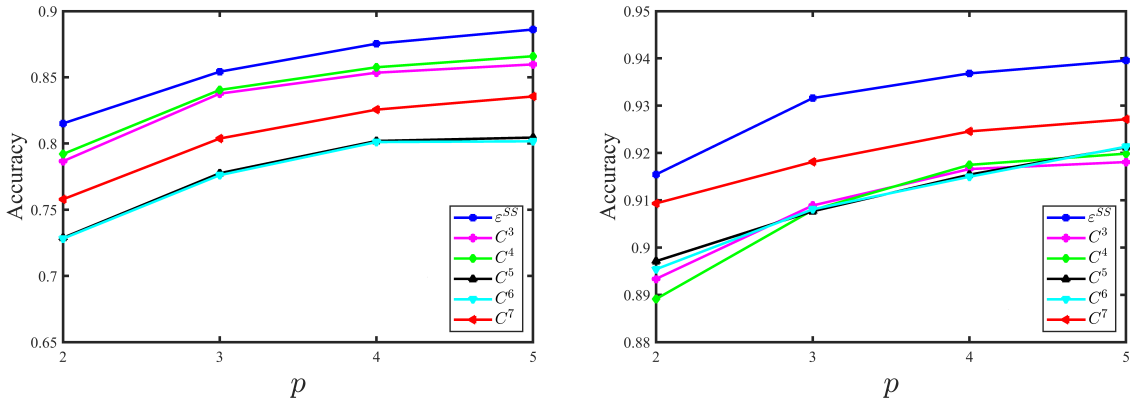
Taux de classification en fonction du nombre d'attributs sélectionnés. La figure 5.9 montre les variations des taux moyens de classification de l'ensemble X^U obtenus par les scores de contraintes semi-supervisés ε^{SS}, C^3, C^4, C^5, C^6, et C^7 avec un nombre n de pixels échantillons fixé à 300 (milieu de la plage $[200-400]$). Les résultats sont moyennés sur 100 exécutions avec différents ensembles de pixels prototypes (p étant fixé à 5). Cette figure indique que les taux moyens de classification obtenus par le score de contraintes ε^{SS} sont meilleurs que ceux obtenus avec les scores de contraintes C^3, C^4, C^5, C^6, et C^7.



(a) Image 7_1_1 (b) Image 17_1_1

Figure 5.9: Taux moyens de classification en fonction du nombre m d'attributs sélectionnés par les scores semi-supervisés ε^{SS}, C^3, C^4, C^5, C^6, et C^7 sur les deux images de test.

Taux de classification en fonction du nombre de prototypes. La figure 5.10 affiche les taux moyens de classification obtenus par les scores de contraintes semi-supervisés en fonction du nombre de prototypes p avec un nombre d'attributs sélectionnés fixé à la moitié du nombre total d'attributs, c.-à-d. $m = 90$. Les résultats sont moyennés sur 100 exécutions avec différents ensembles de prototypes et le nombre n de pixels échantillons est fixé à 300. Nous remarquons que pour toutes les valeurs de p, les taux moyens de classification obtenus avec le score ε^{SS} sont supérieurs à ceux obtenus avec les scores C^3, C^4, C^5, C^6, et C^7.



(a) Image 7_1_1 (b) Image 17_1_1

Figure 5.10: Taux moyens de classification obtenus avec les scores ε^{SS}, C^3, C^4, C^5, C^6, et C^7 en fonction du nombre p de pixels prototypes sur les deux images de test.

5.3.2 Résultats de segmentation d'images

Nous présentons dans cette section les résultats obtenus par notre méthode de segmentation semi-supervisée CFS-SC (Algorithme 5.1) sur cinq différentes bases d'images.

5.3.2.1 Base d'images Prague

La base d'images Prague [222] contient 80 mosaïques de textures naturelles, de résolution 512×512 pixels, générées synthétiquement à partir de 114 images couleur texturées. Le nombre de textures qui composent les mosaïques varie de 3 à 12.

Pour juger de l'efficacité de la méthode CFS-SC, nous comparons ses résultats à ceux des méthodes existantes appliquées sur la base d'images Prague. Parmi celles-ci, nous trouvons *six méthodes non supervisées* (texNcut, PCA-MS [59], MLLIF [113], FSEG [61], FCNTunsup [105], et DLSRC [112]), *huit méthodes supervisées* (MRF [223], co-occurrence features (COF) [222], Con-Col [222], FCNTsup [105], EWT-FCNT [103], U-Net [107], DA [109], et PSP-Net [108]), et *quatre méthodes semi-supervisées* (WSSCGP [60], FSEG$^+$ [60], notre méthode avec (CFS-SC) et sans sélection d'attributs (CSC)). Il est important de noter que la majorité de ces méthodes impliquent une étape de raffinement pour améliorer les performances de segmentation, à l'exception de texNcut, MRF, COF, et Con-Col. Pour CSC et CFS-SC, nous avons fixé n et p respectivement à 300 et 5. Pour FSEG$^+$ et WSSCGP le nombre p de pixels prototypes est fixé à 81 (régions carrées de 9×9).

L'évaluation des performances de segmentation est basée sur les vingt-et-un critères fournies dans le site Web [222], à savoir correct segmentation (CS), over-segmentation (OS), under-segmentation (US), missed error (ME), noise error (NE), omission error (O), commission error (C), class accuracy (CA), recall (CO), precision (CC), type I error (I.), type II error (II.), mean class accuracy estimate (EA), mapping score (MS), root mean square proportion estimation error (RM), comparison index (CI), global consistency error (GCE), local consistency error (LCE), Mirkin metric (dM), Van Dongen metric (dD), et variation of information (dVI). Ces critères sont définis dans l'annexe B.

Le tableau 5.1 présente les résultats obtenus par les méthodes non supervisées et semi-supervisées sur la base d'images Prague large (80 images). Nous constatons que toutes les méthodes semi-supervisées, à l'exception de FSEG$^+$, surclassent les méthodes non supervisées. Notre méthode avec (CFS-SC) et sans raffinement (CFS-SC-nr) est plus performante que les autres méthodes non supervisées et semi-supervisées sur tous les critères, à l'exception de GCE et LCE. CFS-SC exhibe un taux de classification (CO = 95.98%) largement

supérieur à celui des autres méthodes. En comparant les résultats de CFS-SC et CSC, nous pouvons constater l'avantage de la sélection d'attributs ε^{SS} sur la segmentation. En effet, en utilisant l'ensemble des attributs, le taux de classification CO chute à 91.68%.

Tableau 5.1: Performances obtenues par les méthodes non supervisées et semi-supervisées sur la base Prague large (80 images). Les flèches ↑ | ↓ indiquent les directions requises des critères et 'nr' indique une segmentation sans raffinement.

Méthode	**Non supervisée**						**Semi-supervisée**					
	FCNT unsup	DLS RC	tex Ncut	PCA MS	MLL IF	FSE G	FSE G^+	WSS CGP	CSC nr	CSC	CFS SC-nr	CFS SC
↑ CS	79.34	77.46	72.54	72.27	77.73	69.18	75.97	84.18	79.73	82.97	93.82	**94.74**
↓ OS	13.67	28.40	10.92	18.33	15.92	14.69	3.38	10.95	0.14	0.14	**0.00**	**0.00**
↓ US	6.25	**0.00**	9.61	9.41	6.31	13.64	5.53	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
↓ ME	3.80	7.13	10.25	4.19	3.93	5.13	11.82	6.90	14.64	11.92	1.98	**1.41**
↓ NE	3.80	7.39	9.83	3.92	3.92	4.63	11.49	7.04	15.48	12.34	2.59	**1.85**
↓ O	6.47	8.58	7.33	7.25	7.68	9.25	9.12	6.84	8.59	7.32	4.35	**3.70**
↓ C	22.88	29.48	8.17	6.44	24.24	12.55	9.34	7.09	7.33	6.02	3.87	**3.39**
↑ CA	84.17	83.41	80.58	81.13	82.80	78.22	80.26	87.05	84.33	86.01	91.89	**92.62**
↑ CO	87.97	87.36	86.89	85.96	86.89	84.44	88.09	91.58	90.50	91.68	95.51	**95.98**
↑ CC	94.15	95.16	88.28	91.24	93.65	87.38	88.19	94.85	91.92	92.92	95.94	**96.31**
↓ I.	12.03	12.64	13.11	14.04	13.11	15.56	11.91	8.42	9.50	8.32	4.49	**4.02**
↓ II.	1.42	1.19	2.36	1.59	1.50	2.53	2.47	1.28	1.79	1.64	0.75	**0.71**
↑ EA	88.97	89.70	86.39	87.08	88.03	84.24	87.40	92.53	90.69	91.76	95.61	**96.04**
↑ MS	85.23	84.74	80.33	81.84	83.93	78.81	82.46	88.63	85.75	87.51	93.27	**93.97**
↓ RM	3.12	2.42	3.69	5.11	3.27	4.74	2.99	1.89	1.99	1.91	0.81	**0.77**
↑ CI	89.91	90.44	86.97	87.81	89.03	85.03	87.76	92.86	90.94	92.02	95.67	**96.09**
↓ GCE	**6.46**	9.56	11.92	8.35	7.40	9.35	15.08	10.21	14.62	12.86	8.00	7.26
↓ LCE	**4.75**	7.17	6.85	5.61	5.62	6.08	11.71	7.71	11.52	9.79	6.28	5.59
↓ dD	7.79	9.08	9.18	9.06	8.57	10.01	10.34	7.21	9.28	8.12	4.48	**4.02**
↓ dM	4.88	5.40	6.03	5.89	5.30	7.01	6.59	4.40	5.57	4.95	2.66	**2.41**
↓ dVI	14.75	15.18	14.19	14.54	14.88	14.33	14.32	14.52	14.43	14.32	14.18	**14.13**

Le tableau 5.2 présente les résultats obtenus par les méthodes supervisées et semi-supervisées sur la base d'images Prague normale (20 images). Nous constatons que, même si les méthodes supervisées utilisent une information a priori plus importante que celle utilisée par CFS-SC, les résultats de CFS-SC sont meilleurs que ceux obtenus par les méthodes supervisées classiques (MRF, COF, et Con-Col). Par contre, les méthodes basées sur l'apprentissage profond (FCNTsup, DA, U-Net, PSP-Net, et EWT-FCNT) affichent des résultats légèrement meilleurs que ceux obtenus par CFS-SC. Néanmoins, CFS-SC reste compétitive avec ces méthodes, à l'exception de EWT-FCNT, qui fournit des résultats exceptionnels. Notons que, CFS-SC sans raffinement est plus performante que FCNTsup sans raffinement.

Tableau 5.2: Performances obtenues par les méthodes supervisées et semi-supervisées sur la base Prague normale (20 images). Les flèches ↑ | ↓ indiquent les directions requises des critères et 'nr' indique une segmentation sans raffinement.

Méthode	Supervisée									Semi-supervisée	
	MRF	COF	Con Col	FCNT sup-nr	FCNT sup	EWT FCNT	U-Net	DA	PSP Net	CFS SC-nr	CFS SC
↑ CS	46.11	52.48	84.57	87.52	96.01	**98.45**	96.71	94.18	96.45	94.03	94.97
↓ OS	0.81	**0.00**	**0.00**	**0.00**	1.56	**0.00**	1.71	**0.00**	0.17	**0.00**	**0.00**
↓ US	4.18	1.94	1.70	**0.00**	1.20	**0.00**	**0.00**	1.18	0.41	**0.00**	**0.00**
↓ ME	44.82	41.55	9.50	6.70	0.78	**0.37**	0.68	3.42	1.23	1.63	1.09
↓ NE	45.29	40.97	10.22	6.90	0.89	**0.46**	0.48	3.24	1.12	2.32	1.58
↓ O	14.52	20.74	7.00	7.46	2.72	0.93	**0.72**	3.13	2.75	4.42	3.83
↓ C	16.77	22.10	5.34	6.16	2.29	1.04	**0.70**	1.32	2.39	4.45	3.80
↑ CA	65.42	67.01	86.21	87.08	93.95	**97.67**	95.86	94.53	93.89	91.85	92.61
↑ CO	76.19	77.86	92.02	92.61	96.73	**98.78**	96.91	96.23	96.06	95.49	95.97
↑ CC	80.30	78.34	92.68	93.26	97.02	**98.81**	97.38	97.01	96.41	95.91	96.29
↓ I.	23.81	22.14	7.98	7.39	3.27	**1.22**	3.09	3.77	3.94	4.51	4.03
↓ II.	4.82	4.40	1.70	1.49	0.68	**0.25**	0.41	0.58	0.69	0.79	0.76
↑ EA	75.40	76.21	91.72	92.68	96.68	**98.77**	97.01	96.24	96.08	95.59	96.04
↑ MS	64.29	66.79	88.03	88.92	95.10	**98.17**	95.37	94.35	94.08	93.24	93.96
↓ RM	6.43	4.47	2.08	1.38	0.86	**0.24**	0.61	1.07	0.70	0.79	0.70
↑ CI	76.69	77.05	92.02	92.81	96.77	**98.78**	97.08	96.41	96.15	95.65	96.08
↓ GCE	25.79	23.94	11.76	12.54	5.55	2.33	**2.13**	3.50	4.67	8.07	7.30
↓ LCE	20.68	19.69	8.61	9.94	3.75	1.68	**1.46**	2.47	3.52	6.40	5.75
↓ dD	20.35	17.86	7.50	-	3.06	**1.21**	1.45	2.41	2.59	4.50	4.02
↓ dM	13.25	10.62	4.69	-	1.96	**0.74**	0.77	1.35	1.56	2.76	2.48
↓ dVI	14.51	14.22	13.99	-	13.80	**13.68**	**13.68**	13.71	13.77	14.01	13.96

La figure 5.11 affiche quelques images segmentées de la base Prague. Pour des raisons de clarté, les frontières des textures sont marquées en jaune. À partir de cette figure, nous pouvons constater que les méthodes CFS-SC, FCNTsup, et EWT-FCNT fournissent des résultats satisfaisants qui sont proches des segmentations de référence.

5.3.2.2 Base d'images ALI

La base d'images ALI [222] contient 10 mosaïques synthétiques composées d'images satellitaires multispectrales acquises par le capteur "advanced land imager" (ALI). Chaque image a une résolution de 512×512 pixels et est accompagnée de sa segmentation de référence. Comme pour la base d'images Prague, le nombre de textures présentes dans les images de la base ALI varie de 3 à 12.

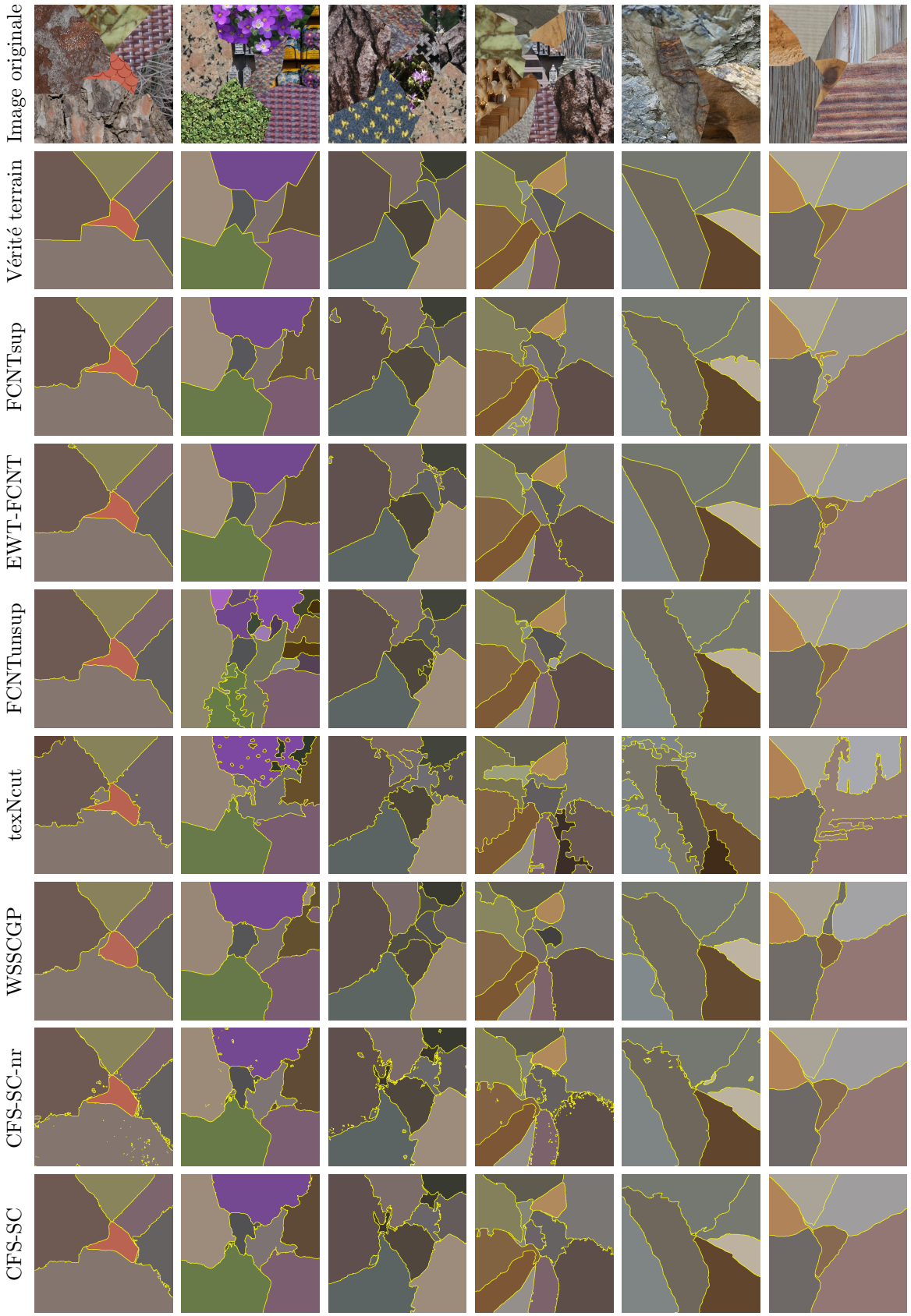


Figure 5.11: Résultats de segmentation de quelques images de la base Prague.

L'évaluation sur cette base d'images compare *quatre méthodes non supervisées* (eCog [75], DHC [71], R-TFR [9], et ENVI [114]), *quatre méthodes supervisées* (DSSN [110], MSCNN [111], KNN [222], et 1NN [222]), et *deux méthodes semi-supervisées* (WSSCGP [60] et notre méthode avec (CFS-SC) et sans raffinement (CFS-SC-nr)). Les paramètres s, n, et p de la méthode CFS-SC sont fixés respectivement à 5, 300, et 5. Les résultats sont évalués en utilisant les mêmes critères que ceux utilisés pour la base Prague.

Le tableau 5.3 présente les résultats obtenus par l'ensemble des méthodes de segmentation (non supervisées, supervisées, et semi-supervisées) sur les images de la base ALI. Nous constatons que les méthodes CFS-SC et CFS-SC-nr surclassent toutes les autres méthodes sur la plupart des critères, à l'exception de DSSN et MSCNN qui affichent de meilleures performances. Cependant, il est important de noter que ces méthodes de segmentation sont basées sur l'apprentissage profond et nécessitent une large base d'images d'apprentissage.

Tableau 5.3: Performances obtenues par les méthodes non supervisées, supervisées, et semi-supervisées sur la base ALI (10 images). Les flèches ↑ | ↓ indiquent les directions requises des critères et 'nr' indique une segmentation sans raffinement.

Méthode	**Non supervisée**				**Supervisée**				**Semi-supervisée**		
	eCog	DHC	R-TFR	ENVI	DSSN	MS CNN	KNN	1NN	WSS CGP	CFS SC-nr	CFS SC
↑ CS	91.91	84.60	78.45	73.49	**96.62**	95.19	92.45	51.29	89.01	91.35	94.51
↓ OS	10.54	6.78	10.39	24.01	4.02	24.70	**0.00**	15.63	**0.00**	**0.00**	**0.00**
↓ US	1.11	8.73	14.33	16.74	**0.00**	**0.00**	**0.00**	10.76	**0.00**	**0.00**	**0.00**
↓ ME	1.20	4.70	2.90	5.46	**0.00**	0.19	5.36	25.82	7.38	5.12	3.07
↓ NE	0.98	5.33	1.28	4.71	**0.00**	0.07	5.32	27.45	7.52	5.72	3.58
↓ O	**0.06**	1.69	1.22	2.33	0.27	0.63	1.98	20.10	4.31	3.82	2.23
↓ C	0.52	0.70	3.11	80.45	0.91	7.80	1.85	42.53	3.90	2.77	**0.49**
↑ CA	94.13	89.69	84.74	81.53	**97.88**	96.40	93.53	69.44	90.53	91.9	94.51
↑ CO	95.42	92.80	89.72	86.48	**98.19**	96.85	96.29	77.19	94.57	95.3	96.86
↑ CC	98.16	92.78	88.98	88.25	**99.69**	99.53	97.01	84.33	95.51	96.06	97.44
↓ I.	4.58	7.20	10.28	13.52	**1.81**	3.15	3.71	22.81	5.43	4.70	3.14
↓ II.	0.24	0.90	1.25	1.67	**0.06**	0.13	0.69	3.29	0.85	0.54	0.35
↑ EA	96.13	92.29	88.37	85.52	**98.66**	97.82	96.34	78.86	94.72	95.43	96.92
↑ MS	94.40	89.59	85.45	81.87	**98.03**	96.61	94.43	69.44	91.85	92.95	95.29
↓ RM	1.62	1.94	3.46	2.79	**0.75**	0.83	1.39	3.53	1.08	0.98	0.87
↑ CI	96.44	92.53	88.84	86.38	**98.80**	98.00	96.49	79.75	94.88	95.55	97.03
↓ GCE	2.67	4.38	4.34	5.76	**0.63**	0.88	5.74	18.64	9.07	7.80	5.17
↓ LCE	1.16	2.67	2.55	1.98	**0.43**	0.60	4.04	14.17	7.03	5.82	3.21
↓ dD	2.91	4.61	6.20	7.58	**1.07**	1.80	3.71	16.08	5.40	4.68	3.12
↓ dM	1.24	2.17	3.06	4.13	**0.50**	0.85	2.43	9.52	2.76	2.10	1.38
↓ dVI	14.75	14.51	**14.43**	14.79	14.57	14.81	14.68	15.77	14.85	14.78	14.61

La figure 5.12 affiche quelques images segmentées de la base ALI. Elle montre clairement que les méthodes de segmentation CFS-SC et DSSN fournissent les résultats de segmentation les plus satisfaisants.

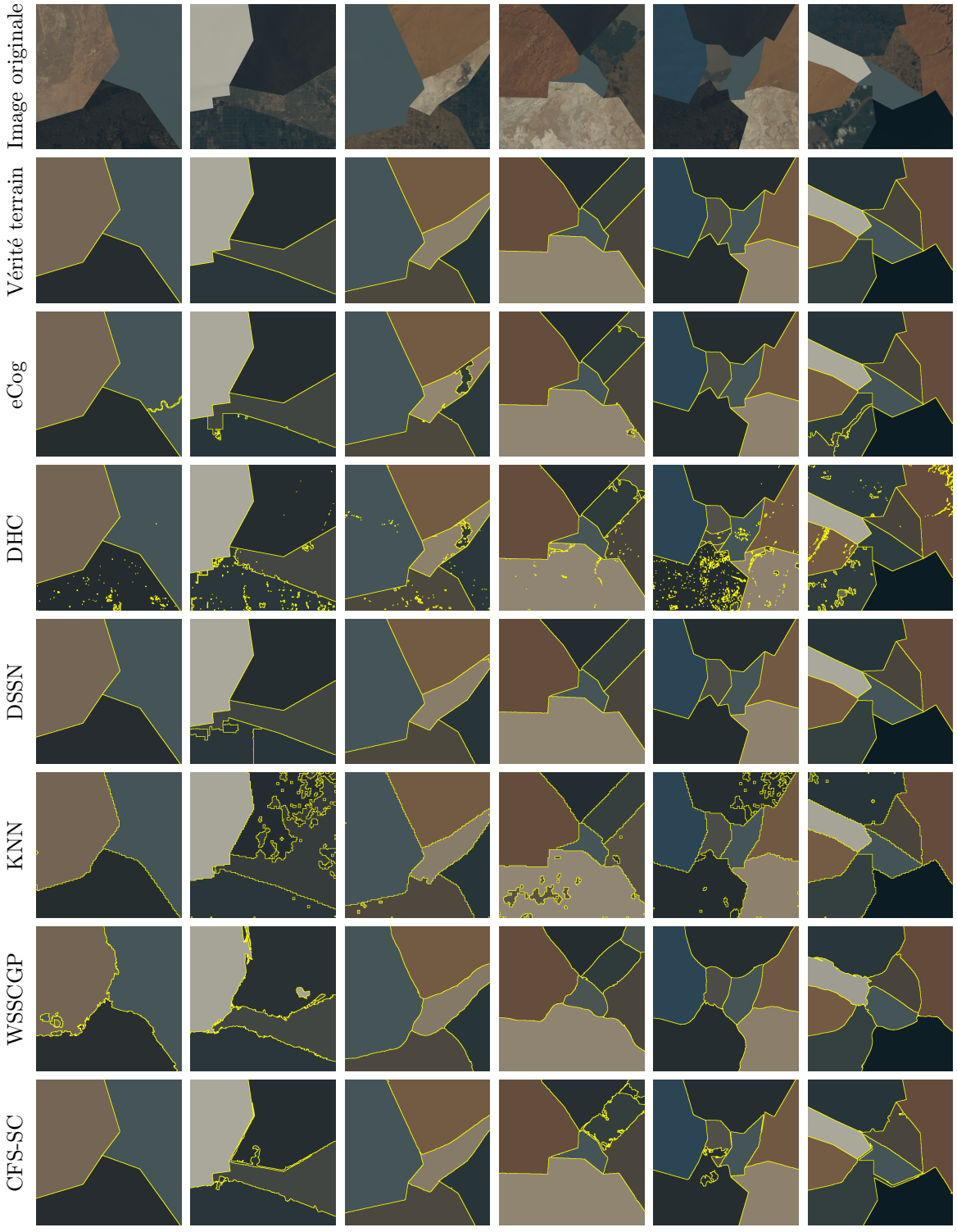


Figure 5.12: Résultats de segmentation de quelques images de la base ALI.

5.3.2.3 Base d'images Histologie

Nous confrontons à présent les résultats de segmentation obtenus sur la base d'images Histologie par nos méthodes CSC, CFS-SC-nr, et CFS-SC avec ceux de plusieurs autres méthodes de l'état de l'art. Cette base contient 36 images couleur de résolution 128 × 128 pixels, générées à partir de 20 types de tissus organiques [7, 224]. Chaque image est représentée par deux types de tissus organiques. Les segmentations de référence ont été déterminées par des experts pathologistes. La grande variété de tissus et la complexité de leurs apparences en font un problème difficile.

Les méthodes impliquées dans cette étude sont FSEG+, WSSCGP, DLSRC, FSEG, et ORTSEG. Les paramètres s, n, et p de la méthode CFS-SC sont fixés respectivement à 5, 300, et 5. L'évaluation des résultats de segmentation est basée sur les mêmes critères utilisés dans l'évaluation des résultats obtenus sur les bases d'images Prague et ALI. Les mesures de performance obtenues sur la base Histologie sont regroupées dans le tableau 5.4.

Tableau 5.4: Performances obtenues sur la base Histologie. Les flèches ↑ | ↓ indiquent les directions requises des critères et 'nr' indique une segmentation sans raffinement.

Méthode	**Non-supervisée**			**Semi-supervisée**				
	DLS RC	ORT SEG	FSEG	FSEG$^+$	WSS CGP	CSC	CFS SC-nr	CFS SC
↑ CS	82.98	72.53	38.98	82.06	86.90	82.54	96.93	**97.07**
↓ OS	3.34	1.71	35.95	1.78	3.29	**0.00**	**0.00**	**0.00**
↓ US	2.82	2.78	**0.00**	10.25	1.94	**0.00**	**0.00**	**0.00**
↓ ME	9.69	20.09	19.96	7.67	5.95	11.33	**0.00**	**0.00**
↓ NE	11.49	20.31	21.05	8.25	7.02	11.99	**0.00**	**0.00**
↓ O	7.64	10.17	24.77	13.71	6.70	8.69	3.26	**3.10**
↓ C	6.96	10.57	24.23	7.42	6.21	9.62	3.47	**3.31**
↑ CA	86.80	81.81	67.17	84.90	87.69	84.71	94.12	**94.37**
↑ CO	91.44	88.84	72.38	90.70	92.07	91.20	96.93	**97.07**
↑ CC	94.96	91.27	92.27	92.02	95.28	91.99	97.00	**97.14**
↓ I.	8.56	11.16	27.62	9.30	7.93	8.80	3.07	**2.93**
↓ II.	5.08	8.98	6.99	11.40	5.07	8.57	3.45	**3.27**
↑ EA	92.60	88.71	78.78	90.66	93.15	91.21	96.94	**97.08**
↑ MS	87.17	83.43	67.63	86.05	88.11	86.8	95.40	**95.61**
↓ RM	6.55	9.93	17.29	7.42	5.82	5.22	1.06	**1.13**
↑ CI	92.89	89.37	80.49	91.00	93.40	91.4	96.95	**97.09**
↓ GCE	10.42	12.36	14.10	9.64	9.73	13.65	5.77	**5.52**
↓ LCE	6.77	7.23	11.24	6.12	6.48	9.68	4.82	**4.51**
↓ dD	7.54	9.73	18.55	7.65	6.90	8.66	3.07	**2.93**
↓ dM	12.88	17.19	26.18	14.00	11.99	14.96	5.90	**5.65**
↓ dVI	5.88	5.64	7.23	5.61	5.85	5.68	5.43	**5.42**

D'après ce tableau, nous constatons que nos méthodes de segmentation avec (CFS-SC) et sans raffinement (CFS-SC-nr) fournissent des résultats nettement meilleurs que les autres méthodes en ce qui concerne les vingt-et-un critères évalués. Une comparaison entre notre méthode avec (CFS-SC) et sans sélection d'attributs (CSC) nous permet de constater encore une fois l'apport significatif de la méthode de sélection d'attributs ε^{SS} dans la segmentation des images Histologie. Par exemple, le taux de classification (CO) atteint par CFS-SC (97.07%) est nettement supérieur à celui de CSC (91.20%).

La figure 5.13 affiche quelques images segmentées de la base Histologie, et sur laquelle nous pouvons clairement observer que les frontières entre les deux tissus (indiquées en jaune) sont mieux détectées par notre méthode CFS-SC et sont proches des frontières de référence.

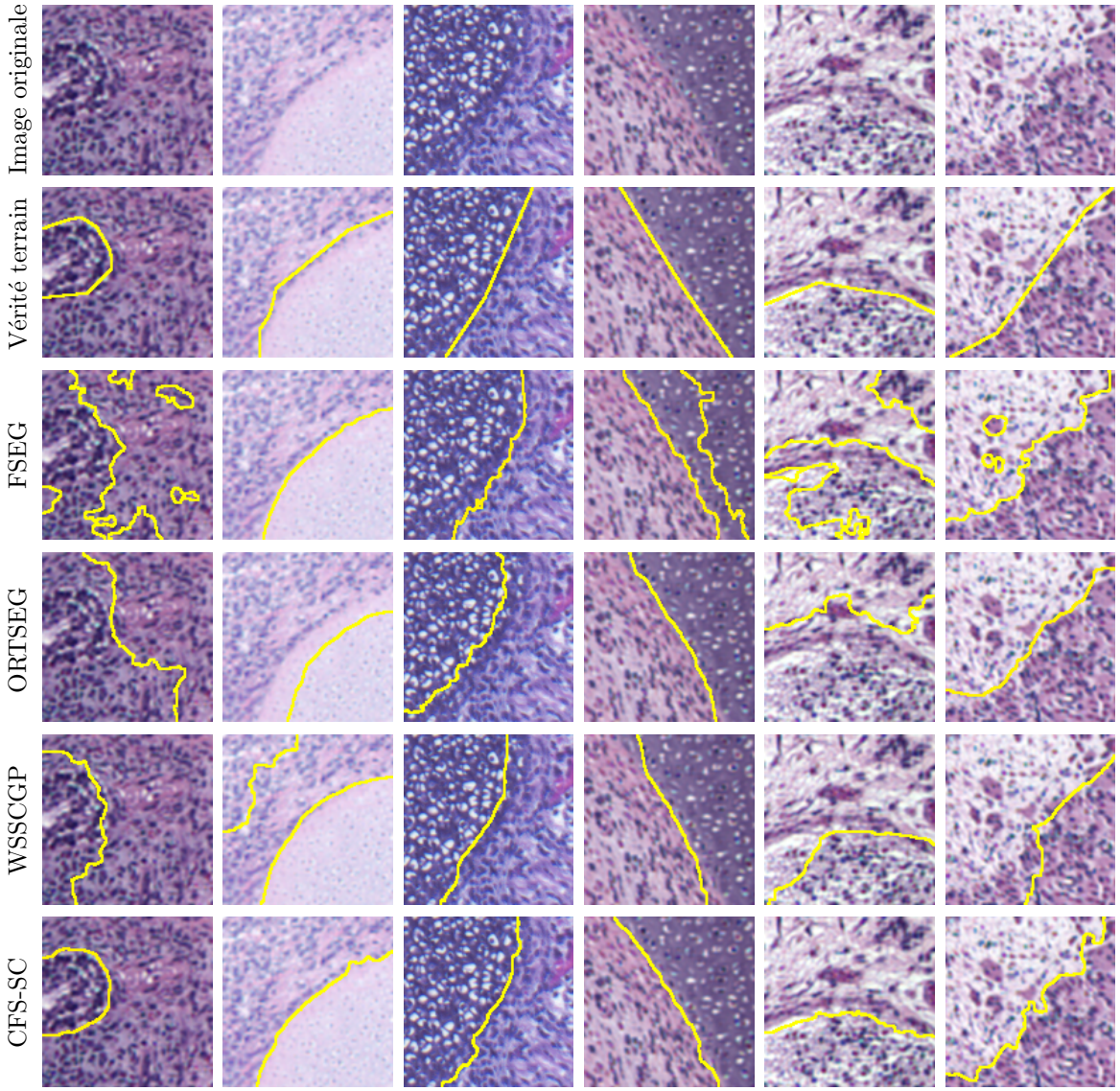


Figure 5.13: Résultats de segmentation de quelques images de la base Histologie.

5.3.2.4 Base d'images Outex

Notre méthode de segmentation CFS-SC est conçue pour segmenter des images couleur texturées, mais elle convient également à la segmentation des images texturées en niveaux de gris. Pour vérifier cette assertion, nous proposons de l'appliquer sur des images de la base Outex (suite de tests US_00000). Cette base contient 100 images en niveaux de gris texturées de résolution 512×512 pixels, dont chaque image est composée de cinq textures [225].

Nous avons exécuté la méthode CFS-SC en utilisant les mêmes paramètres que ceux utilisés sur les bases d'images ALI et Histologie. Il convient de noter que seuls les attributs de texture (Haralick et Gabor) extraits des images en niveaux de gris ont été utilisés. Nous avons comparé les résultats de notre méthode avec ceux de plusieurs autres méthodes de l'état de l'art, notamment ORTSEG, PCA-MS, FSEG, EWT-FCNT, U-Net, DA, PSP-Net, et WSSCGP. Cette comparaison est basée sur les mêmes critères utilisés dans [103]: recall (CO), normalized variation of information (NVOI), swapped directional hamming distance (SDHD), Van Dongen metric (VD), swapped segmentation covering (SSC), bipartite graph matching (BGM), et bidirectional consistency error (BCE) [226]. Tous ces critères fournissent des valeurs comprises entre 0% et 100% et ils sont décrits dans l'annexe B.

Le tableau 5.5 présente les valeurs des critères obtenues par l'ensemble des méthodes mentionnées appliquées à la base d'images Outex. Nous constatons que, contrairement aux images couleur, nos méthodes de segmentation CFS-SC et CFS-SC-nr surclassent largement toutes les méthodes non supervisées (ORTSEG, FSEG, et PCA-MS), la méthode semi-supervisée WSSCGP, et même les méthodes basées sur l'apprentissage profond (U-Net, DA, et PSP-Net), à l'exception de la méthode EWT-FCNT qui affiche de meilleurs résultats, tout comme pour les images couleur.

Tableau 5.5: Performances obtenues sur la base Outex. Les flèches ↑ | ↓ indiquent les directions requises des critères et 'nr' indique une segmentation sans raffinement.

Méthode	**Non supervisée**			**Supervisée**				**Semi-supervisée**		
	ORT SEG	FSEG	PCA MS	EWT FCNT	U-Net	DA	PSP Net	WSS CGP	CFS SC-nr	CFS SC
↑NVOI	77.99	73.65	83.86	**96.52**	72.83	70.70	86.00	85.41	87.88	88.70
↑ SSC	70.88	62.23	84.32	**98.27**	64.25	60.65	84.22	89.11	92.42	93.01
↑ SDHD	80.64	65.08	89.67	**99.13**	71.40	67.77	88.00	94.71	96.02	96.34
↑ BGM	76.42	64.97	89.24	**99.13**	70.85	67.27	87.83	93.59	96.02	96.34
↑ VD	83.87	80.60	91.39	**99.13**	80.47	78.75	91.64	94.15	96.02	96.34
↑ BCE	69.50	61.08	82.45	**98.02**	61.89	58.58	82.63	86.64	91.19	91.77
↑ CO	76.42	64.97	89.24	**98.36**	70.28	67.28	86.72	93.59	96.02	96.34

Une inspection visuelle de quelques images segmentées (Figure 5.14) confirme l'efficacité de notre méthode CFS-SC par rapport aux autres méthodes de segmentation.

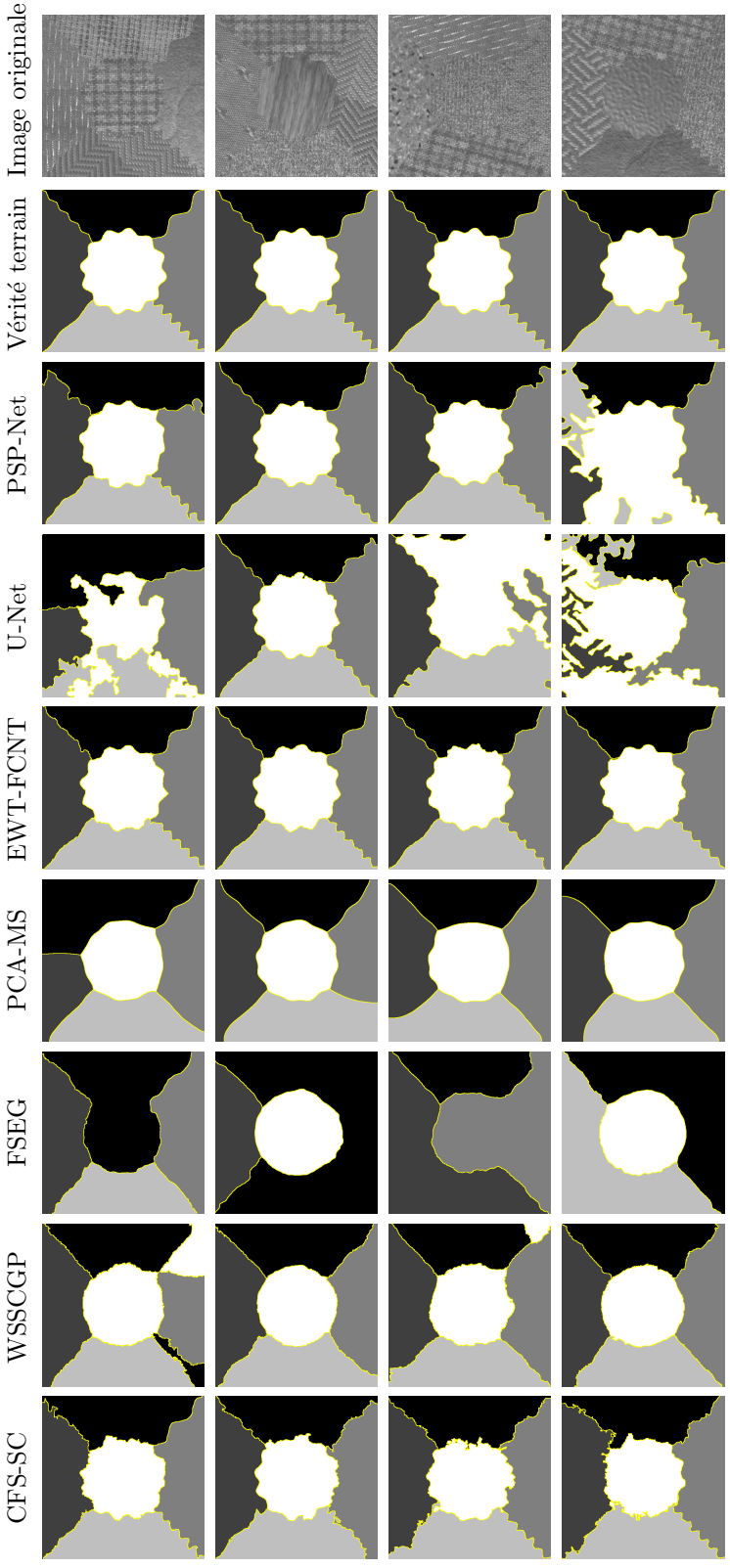


Figure 5.14: Résultats de segmentation de quelques images de la base Outex.

5.3.2.5 Base d'images Berkeley

Pour valider davantage l'efficacité de notre méthode de segmentation CFS-SC, nous l'avons appliquée à quelques images naturelles extraites de la base d'images Berkeley [227]. Cette base couvre une grande variété de scènes naturelles et est largement utilisée dans le cadre de la segmentation d'images couleur. Elle contient initialement 300 images naturelles couleur (BSDB300), puis elle a été élargie à 500 images (BSDB500). Les images de cette base ont une résolution de 481×321 ou de 321×481 pixels. Pour nos tests, nous avons sélectionné 16 images qui présentent des scènes avec des régions texturées en couleur (Figure 5.15).

Les paramètres de notre méthode CFS-SC, à savoir s, n, et p sont fixés respectivement à 5, 300, et 5. Les résultats obtenus par la méthode CFS-SC sont comparés à ceux obtenus par une méthode récente appelée robust self-sparse fuzzy clustering (RSSFCA) [228]. L'évaluation des performances de la segmentation est réalisée en utilisant trois mesures: F-mesure, indice de rand probabiliste (PRI), et le rappel (CO), qui sont décrites dans l'annexe B. Plus les valeurs de ces mesures sont élevées plus les segmentations sont meilleures.

La figure 5.15 présente les 16 images segmentées de la base d'images Berkeley. Elle montre clairement que les régions qui composent chaque image sont mieux séparées par notre méthode CFS-SC. Les images segmentées sont très proches des segmentations de référence. Cette observation est confortée par les valeurs des critères de comparaison obtenues par la méthode CFS-SC, qui sont nettement supérieures à celles obtenues par la méthode RSSFCA.

5.3.3 Temps de calcul

Dans cette partie, nous évaluons les temps de calcul des différentes étapes qui composent notre méthode de segmentation CFS-SC.

Le tableau 5.6 présente les temps de calcul consommés par les méthodes de segmentation FSEG, WSSCGP, et CFS-SC sur trois bases d'images choisies en fonction de leur taille (128×128 ou 512×512) et de leur type (couleur ou niveau de gris). Le temps de calcul de ces méthodes peut être divisé en deux parties: le temps d'extraction des attributs (FE) et le temps cumulé par les étapes de sélection des attributs et de classification des pixels (OS). À partir de ce tableau, nous constatons que le temps de calcul global de la méthode CFS-SC est le plus élevé. Par exemple, il est d'environ de ≈ 518 secondes pour une image de résolution 512×512 pixels, et la majeure partie de ce temps est consommée par l'étape

Figure 5.15: Résultats de segmentation de quelques images de la base Berkeley. Les valeurs des mesures de performance CO, F-measure, et PRI sont indiquées entre les parenthèses.

d'extraction des attributs (486 secondes), en particulier par le calcul des attributs de Haralick (403 secondes). Les 34 secondes restantes ont été consommées par les trois étapes de base de la méthode CFS-SC (sélection d'attributs sous contraintes (20.7 secondes), classification spectrale sous contraintes (0.033 seconde), et classification des pixels non échantillons (11.01

secondes)). Ainsi, à l'exception de l'étape d'extraction des attributs, notre méthode CFS-SC est relativement rapide. Cependant, il est toujours possible de réduire le temps d'extraction des attributs en choisissant ceux dont le temps de calcul est relativement court, ou en s'équipant des ordinateurs récents munis de cartes GPU.

Tableau 5.6: Temps de calcul (en seconde) des méthodes FSEG, WSSCGP, et CFS-SC. (FE: étape d'extraction d'attributs et OS: autres étapes.)

Méthode	**Prague** (512×512) **couleur**			**Histologie** (128×128) **couleur**			**Outex** (512×512) **niveaux de gris**		
	FE	OS	Total	FE	OS	Total	FE	OS	Total
FSEG	0.41	0.67	1.08	0.02	0.04	0.06	0.14	0.41	0.55
WSSCGP	0.90	31.92	32.82	0.06	0.61	1.21	0.72	26.63	27.35
CFS-SC	486	31.74	517.74	26.89	18.21	45.10	67.87	10.69	78.56

5.4 Conclusion

Dans ce chapitre, nous avons décrit en détail notre méthode de segmentation d'images couleur texturées CFS-SC, qui combine à la fois la sélection d'attributs et la classification spectrale semi-supervisées sous contraintes. La sélection des attributs de texture couleur est réalisée par notre score de contraintes semi-supervisé présenté dans le chapitre précédent. Des résultats expérimentaux ont confirmé la supériorité de notre score de contraintes semi-supervisé sur les autres scores de contraintes dans les deux contextes d'apprentissage semi-supervisé et dans le cadre de la sélection des attributs de texture couleur.

Pour valider notre méthode de segmentation CFS-SC, nous l'avons testée sur cinq bases d'images différentes. Les résultats obtenus ont démontré l'efficacité de notre méthode CFS-SC à segmenter des images texturées, qu'elles soient en couleur ou en niveaux de gris, synthétiques ou réelles, et qu'elles proviennent de domaines tels que les images naturelles, médicales ou satellitaires. Malgré l'utilisation d'une technique d'échantillonnage simple et des attributs classiques de texture couleur, notre méthode a produit des résultats remarquables. Ces résultats ont été confirmés par la comparaison avec plusieurs méthodes de segmentation (non supervisées, semi-supervisées, et supervisées).

Conclusion et Perspectives

Le travail abordé dans cette thèse concerne la segmentation d'images couleur texturées basée sur la sélection d'attributs et la classification spectrale semi-supervisées sous contraintes. Nos contributions dans ce travail sont réparties à travers les cinq chapitres de ce manuscrit dont nous faisons ici une synthèse.

1 Synthèse des chapitres

Dans le chapitre 1, nous avons établi un état de l'art sur les méthodes de segmentation d'images couleur texturées en les catégorisant de trois manières différentes: selon la stratégie de caractérisation de la texture couleur (séquentielle, parallèle, ou conjointe), selon la technique employée pour détecter les régions (contours actifs, croissance de régions, division-fusion de régions, coupe de graphe, classification, ou apprentissage profond), et selon le contexte d'apprentissage défini par la connaissance a priori disponible (non supervisé, supervisé, et semi-supervisé). Parmi les méthodes utilisées en segmentation d'images, notre attention s'est portée sur la classification spectrale, car cette dernière permet de détecter des classes de formes complexes et non linéairement séparables. De plus, elle s'appuie sur le même formalisme que la sélection d'attributs qui est la théorie des graphes et peut être facilement adaptée au contexte semi-supervisé sous contraintes de type must-link et cannot-link.

Pour justifier le choix de la méthode de classification spectrale, nous avons confronté ses résultats, dans le chapitre 2, à quelques méthodes classiques telles que la méthode k-means et sa version floue FCM, la méthode mean-shift, la méthode HAC, et les méthodes basées sur la densité DBSCAN et DDC. Les résultats obtenus sur des bases de données synthétiques et réelles confirment la supériorité de la classification spectrale sur ces méthodes classiques.

Pour conforter ce choix, nous avons comparé la méthode de classification spectrale semi-supervisée sous contraintes à d'autres méthodes semi-supervisées classiques telles que COP k-means, PC k-means, et HAC sous contraintes. Les résultats obtenus attestent, d'une part, de l'intérêt à utiliser les contraintes en classification; et d'autre part, l'efficacité de la méthode de classification spectrale sous contraintes par rapport aux méthodes de classification semi-supervisée sous contraintes. Il ressort de cette étude que les méthodes de classification spectrale avec et sans contraintes dépendent des similarités entre les données, qui sont conditionnées par les distances calculées en utilisant l'ensemble d'attributs. Or, utiliser un grand ensemble d'attributs peut dégrader les résultats de classification car certains d'entre eux sont redondants, non pertinents, ou inutiles. Pour faire face à ce problème, nous avons proposé de sélectionner, à partir de l'ensemble des attributs originaux, les attributs les plus pertinents pour la classification.

C'est dans cette optique que nous nous sommes intéressés au chapitre 3 à la sélection d'attributs. Nous avons classé les méthodes de sélection d'attributs en fonction du contexte d'apprentissage (non supervisé, supervisé, et semi-supervisé), selon la dépendance ou non à un algorithme de classification (filtre, enveloppe, et hybride), et selon le nombre d'attributs évalués (un: feature ranking versus plusieurs attributs à la fois: subset selection). Cette étude nous a conduit à nous focaliser sur les méthodes de sélection d'attributs de type filtre basée sur les scores de contraintes, où un état de l'art assez exhaustif sur les scores de contraintes dans les différents contextes (non supervisé, supervisé, et semi-supervisé) est dressé. Ces scores semblent les mieux adaptés à être utilisés en pratique et notamment en segmentation d'images, car ils sont indépendants du classifieur et rapides à calculer. En outre, ils évaluent la pertinence des attributs individuellement en ignorant ainsi la corrélation entre eux, et ils ne sont pas en mesure de déterminer automatiquement le nombre d'attributs à sélectionner. De plus, ils se basent sur des matrices de similarité qui sont mesurées dans l'espace défini par l'ensemble original des attributs. Par conséquent, ces méthodes peuvent être affectées par la malédiction de la dimension.

Pour remédier à tous ces inconvénients, nous avons proposé dans le chapitre 4, notre première principale contribution, à savoir une méthode originale de sélection d'attributs de type filtre qui est basée sur un score de contraintes, capable d'évaluer la pertinence d'un sous-ensemble d'attributs à la fois dans les deux contextes d'apprentissage supervisé et semi-supervisé. Notre score de contraintes permet également de tenir compte de la corrélation entre les attributs et de déterminer automatiquement le nombre optimal d'attributs. Des résultats expérimentaux ont montré que le score de contraintes proposé est meilleur que des scores

de contraintes supervisés et semi-supervisés de l'état de l'art, ainsi que des méthodes de référence de sélection d'attributs.

Notre deuxième principale contribution dans cette thèse est une méthode de segmentation d'images couleur texturées, dénommée CFS-SC. Cette méthode, décrite en détail dans le chapitre 5, combine à la fois la sélection d'attributs de texture couleur par notre score de contraintes semi-supervisé et la classification spectrale semi-supervisée sous contraintes. Des tests menés sur des images couleur texturées ont confirmé une fois de plus la supériorité de notre score sur les autres scores de contraintes semi-supervisés de l'état de l'art. Pour juger de l'efficacité de la méthode CFS-SC, nous avons réalisé une étude comparative avec plusieurs méthodes de segmentation d'images non supervisées, semi-supervisées, et supervisées. Les résultats obtenus sur la base d'images couleur texturées Prague, la base d'images en niveaux de gris texturées Outex, la base d'images médicale Histologie, la base d'images satellitaire ALI, et les images couleur naturelles de la base Berkeley montrent que notre méthode surclasse les méthodes classiques et reste très compétitive avec les méthodes récentes par apprentissage profond, malgré l'utilisation de quelques prototypes seulement.

2 Perspectives

Ce travail ouvre plusieurs perspectives relatives à la méthode de sélection d'attributs et à la méthode de segmentation d'images couleur texturées proposées.

2.1 Méthode de sélection d'attributs

La méthode de sélection d'attributs élaborée s'appuie sur le score de contraintes que nous avons également proposé, qui combiné avec la procédure SFS, permet de choisir un nombre optimal d'attributs. SFS est une méthode de sélection d'attributs sous-optimale. Une alternative serait de la substituer par une méthode optimale basée sur les algorithmes d'optimisation méta-heuristiques, comme les algorithmes génétiques ou les essaims de particules. D'un autre coté, le score de contraintes proposé permet d'évaluer la pertinence d'un sous-ensemble d'attributs dans les deux contextes d'apprentissage supervisé et semi-supervisé, où l'information a priori est représentée initialement par des prototypes de chaque classe à partir desquels les ensembles de contraintes sont générés. Une perspective serait d'impliquer non pas des prototypes mais directement des contraintes comme information a priori, et de procéder au choix des contraintes les plus cohérentes. Une autre perspective à laquelle nous attachons un intérêt particulier est d'étendre notre méthode de sélection d'attributs

au contexte d'apprentissage non supervisé. L'idée serait d'exploiter la matrice de similarité afin d'extraire des prototypes représentatifs des classes sans aucune information a priori.

2.2 Méthode de segmentation d'images

Dans notre méthode de segmentation d'images, les pixels sont caractérisés par des attributs de texture couleur extraits à partir des matrices de co-occurrences chromatiques et des filtres de Gabor. Une possibilité d'améliorer les résultats serait de faire appel à d'autres types d'attributs de texture couleur comme les LBP, les ondelettes, les fractales, ou les CNN pré-entrainés. Aussi, nous préconisons d'employer d'autres méthodes d'échantillonnage, plus élaborées comme les superpixels, pour améliorer encore davantage les performances de la segmentation. Nous prévoyons également d'appliquer notre méthode de segmentation sur d'autres images, notamment médicales. L'extension au contexte non supervisé constitue pour nous un nouveau challenge que nous tentons de relever dans un futur proche.

Annexe A

Attributs de texture couleur

Cette annexe présente un aperçu sur les attributs utilisés dans notre méthode de segmentation (Chapitre 5). Il s'agit des attributs de couleur issus des espaces de couleur RGB, XYZ, HSV, et L$^*a^*b^*$, ainsi que des attributs de texture couleur extraits des matrices de co-occurrence chromatiques et des filtres de Gabor.

A.1 Espaces de couleur

Pour chaque pixel de l'image, nous avons extrait les attributs de couleur correspondant aux valeurs des composantes dans les espaces de couleur RGB, XYZ, HSV, et L$^*a^*b^*$.

Espace de couleur RVB. Cet espace est l'un des espaces de couleur couramment utilisés dans les systèmes d'acquisition (moniteurs, caméras, scanners, etc). Il est basé sur un mélange additif des trois couleurs primaires: R (rouge), G (vert), et B (bleu). La représentation de la couleur dans cet espace forme un cube, où le noir correspond à l'absence de couleur et le blanc est obtenu en combinant les trois couleurs primaires.

Espace de couleur XYZ. Cet espace est utilisé comme un espace intermédiaire entre l'espace RGB et d'autres espaces de couleur. Les composantes X, Y, et Z sont virtuelles et n'ont pas de signification physique directe. La conversion de l'espace RGB vers l'espace XYZ est effectuée à l'aide de la matrice de passage suivante:

$$\begin{bmatrix} \mathrm{X} \\ \mathrm{Y} \\ \mathrm{Z} \end{bmatrix} = \begin{bmatrix} 0.4124 & 0.3575 & 0.1804 \\ 0.2126 & 0.7151 & 0.0721 \\ 0.0193 & 0.1191 & 0.9502 \end{bmatrix} \begin{bmatrix} \mathrm{R} \\ \mathrm{G} \\ \mathrm{B} \end{bmatrix} \tag{A.1}$$

L'illuminant D_{65} correspondant à la moyenne des lumières durant la journée est utilisé.

Espace de couleur HSV. Cet espace offre une perception visuelle de la couleur proche de celle de la vision humaine. Les composantes de cet espace sont: la *teinte* (hue) qui correspond à la couleur, la *saturation* qui caractérise la pureté de la couleur par rapport à un illuminant, et la *valeur* (value) qui mesure la brillance de la couleur. Le passage de l'espace RGB vers l'espace HSV est effectué en utilisant les équations définies ci-dessous.

$$\mathrm{h} = \begin{cases} \theta & \text{si} \quad \mathrm{b} \leq \mathrm{g} \\ 2\pi - \theta & \text{si} \quad \mathrm{b} > \mathrm{g} \end{cases} \tag{A.2}$$

$$\mathrm{s} = \begin{cases} 0 & \text{si max (r, g, b)} = 0 \\ 1 - \dfrac{\min\ (\mathrm{r},\ \mathrm{g},\ \mathrm{b})}{\max\ (\mathrm{r},\ \mathrm{g},\ \mathrm{b})} & \text{sinon} \end{cases} \tag{A.3}$$

$$\mathrm{v} = \max\ (\mathrm{r},\ \mathrm{g},\ \mathrm{b}) \tag{A.4}$$

avec r, g, b représentent les coordonnées du vecteur de l'espace de couleur RGB. h, s, et v représentent les coordonnées du vecteur de l'espace de couleur HSV, et

$$\theta = \begin{cases} \text{non défini} & \text{si r} = \mathrm{g} = \mathrm{b} \\ \cos^{-1}\left(\dfrac{(\mathrm{r}-\mathrm{g})+(\mathrm{r}-\mathrm{b})}{2\sqrt{(\mathrm{r}-\mathrm{g})^2+(\mathrm{r}-\mathrm{b})(\mathrm{g}-\mathrm{b})}}\right) & \text{sinon.} \end{cases}$$

Espace de couleur $\mathbf{L^*a^*b^*}$. Cet espace est dérivé de l'espace XYZ et d'un blanc de référence. La composante L^* mesure l'opposition noir-blanc (luminance) par une valeur comprise entre 0 (noir) et 100 (blanc). La composante a^* mesure l'opposition vert-rouge par une valeur comprise dans l'intervalle $[-100 + 100]$. La composante b^* mesure l'opposition bleu-jaune par une valeur comprise dans l'intervalle $[-100 + 100]$. Le passage de l'espace XYZ vers l'espace $\mathrm{L}^*a^*b^*$ est effectué à partir des équations suivantes:

$$\mathrm{L}^* = 116 f\left(\frac{\mathrm{X}}{\mathrm{Y}_{ref}}\right) - 16 \tag{A.5}$$

$$a^* = 500\left[f\left(\frac{\mathrm{X}}{\mathrm{X}_{ref}}\right) - f\left(\frac{\mathrm{Y}}{\mathrm{Y}_{ref}}\right)\right] \tag{A.6}$$

$$b^* = 200\left[f\left(\frac{\mathrm{Y}}{\mathrm{Y}_{ref}}\right) - f\left(\frac{\mathrm{Z}}{\mathrm{Z}_{ref}}\right)\right] \tag{A.7}$$

avec

$$f(t) = \begin{cases} t^{(\frac{1}{3})} & \text{si} \quad t > \delta^3 \\ \frac{t}{3\delta^2} + \frac{4}{29} & \text{si} \quad t \leq \delta^3 \end{cases} \qquad \text{avec} \qquad \delta = \frac{6}{29}$$

Les valeurs X_{ref},Y_{ref}, et Z_{ref} sont les composantes X, Y, et Z associées à un blanc de référence (par défaut l'illuminant D_{65} est utilisé).

A.2 Matrices de co-occurrences chromatiques

Haralick et al. [20] ont suggéré de caractériser la texture contenue dans une image en calculant un certain nombre d'attributs statistiques à partir des matrices de co-occurrences des niveaux de gris (GLCM). Ces GLCM décrivent les dépendances spatiales entre les niveaux de gris. Le concept des GLCM a ensuite été étendu aux images couleur en définissant les matrices de co-occurrences chromatiques (CCM) [31, 32].

Soit I une image couleur codée dans l'espace ($\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$) et soit $\mathcal{C}_k$, $\mathcal{C}_{k'}$ deux des trois composantes ($\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$). La matrice de co-occurrences couleur $\boldsymbol{\mathcal{M}}^{\mathcal{C}_k,\mathcal{C}_{k'}}$ mesure l'interaction spatiale entre les composantes $\mathcal{C}_k$ et $\mathcal{C}_{k'}$ de deux pixels séparés d'une distance spatiale δ suivant une orientation θ. Le contenu des cellules de $\mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(i,j)$ est égal au nombre de fois qu'un pixel x, dont le niveau de la composante $\mathcal{C}_k(x)$ est égal à i, possède un pixel voisin x' se trouvant à une distance δ de x et selon une direction θ et dont le niveau de la composante $\mathcal{C}_{k'}(x')$ est égal à j. Les matrices de co-occurrences sont de taille $(Nc \times Nc)$ où Nc est le niveau de quantification des composantes de couleur. Nc est souvent fixé à 8, 16, et 32 afin de réduire le temps de calcul des attributs à partir de ces matrices. Pour rendre ces attributs invariants à la rotation, les matrices de co-occurrences sont calculées avec plusieurs orientations puis moyennées [15]. Pour une image couleur I, il est alors possible de déterminer six matrices de co-occurrences chromatiques: 3 matrices intra-composantes ($\boldsymbol{\mathcal{M}}^{\mathcal{C}_1,\mathcal{C}_1}$, $\boldsymbol{\mathcal{M}}^{\mathcal{C}_2,\mathcal{C}_2}$, et $\boldsymbol{\mathcal{M}}^{\mathcal{C}_3,\mathcal{C}_3}$) et 3 matrices inter-composantes ($\boldsymbol{\mathcal{M}}^{\mathcal{C}_1,\mathcal{C}_2}$, $\boldsymbol{\mathcal{M}}^{\mathcal{C}_1,\mathcal{C}_3}$, et $\boldsymbol{\mathcal{M}}^{\mathcal{C}_2,\mathcal{C}_3}$).

Les matrices de co-occurrences chromatiques sont riches en informations mais gourmandes en mémoire de stockage, surtout dans le cas de la segmentation d'images, où six matrices de co-occurrences chromatiques sont déterminées pour chaque pixel de l'image. Celles-ci sont calculées pour chaque pixel x de l'image à partir des pixels situés dans une fenêtre de voisinage de taille $(2s+1)\times(2s+1)$ centrée sur le pixel x. Pour contourner ce problème, des attributs statistiques comme ceux de Haralick [20], au nombre de quatorze, permettent de résumer les informations portées par les matrices de co-occurrences chromatiques (Tableau A.1). Certaines notations utilisées dans ce tableau sont définies ci-dessous:

$$\mathcal{M}_x^{\mathcal{C}_k,\mathcal{C}_{k'}}(i) = \sum_{j=1}^{N_c} \mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(i,j) \quad \text{et} \quad \mathcal{M}_y^{\mathcal{C}_k,\mathcal{C}_{k'}}(j) = \sum_{i=1}^{N_c} \mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(i,j)$$

$$\mathcal{M}_{x+y}^{\mathcal{C}_k,\mathcal{C}_{k'}}(l) = \sum_{i=1}^{N_c}\sum_{j=1}^{N_c} \mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(i,j),\ l = i+j,\ l = 1,\ldots,2N_c$$

$$\mu_{x-y} = \sum_{l=1}^{N_c} l \cdot \mathcal{M}_{x-y}^{\mathcal{C}_k,\mathcal{C}_{k'}}(l) \text{ avec } \mathcal{M}_{x-y}^{\mathcal{C}_k,\mathcal{C}_{k'}}(l) = \sum_{i=1}^{N_c}\sum_{j=1}^{N_c} \mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(i,j),\ l = |i-j|,\ \ l = 1,\ldots,N_c$$

$$\text{HXY1} = -\sum_{i=1}^{N_c}\sum_{j=1}^{N_c} \mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(i,j)\log\{\mathcal{M}_x^{\mathcal{C}_k,\mathcal{C}_{k'}}(i) \times \mathcal{M}_y^{\mathcal{C}_k,\mathcal{C}_{k'}}(j)\}$$

$$\text{HXY2} = -\sum_{i=1}^{N_c}\sum_{j=1}^{N_c} \mathcal{M}_x^{\mathcal{C}_k,\mathcal{C}_{k'}}(i) \times \mathcal{M}_y^{\mathcal{C}_k,\mathcal{C}_{k'}}(j) \log\{\mathcal{M}_x^{\mathcal{C}_k,\mathcal{C}_{k'}}(i) \times \mathcal{M}_y^{\mathcal{C}_k,\mathcal{C}_{k'}}(j)\}$$

$$Q(j) = \sum_{i=1}^{N_c} \frac{\mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(i,l) \times \mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(j,l)}{\mathcal{M}_x^{\mathcal{C}_k,\mathcal{C}_{k'}}(i) \times \mathcal{M}_y^{\mathcal{C}_k,\mathcal{C}_{k'}}(j)}$$

Mesures	Mesures
Énergie $f_1 = \sum_{i=1}^{N_c}\sum_{j=1}^{N_c}\{\mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(i,j)\}^2$	**Contraste** $f_2 = \sum_{n=1}^{N_c} n^2 \Big\{ \sum_{i=1}^{N_c}\sum_{j=1}^{N_c} \mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(i,j) \Big\}$ où $n = \lvert i-j \rvert$
Corrélation $f_3 = \frac{\sum_{i=1}^{N_c}\sum_{j=1}^{N_c}(i-\mu_x)(j-\mu_y)\mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(i,j)}{\sigma_x\sigma_y}$ μ_x, μ_y, σ_x et σ_y sont respectivement les moyennes et les écarts types de $\mathcal{M}_x^{\mathcal{C}_k,\mathcal{C}_{k'}}$ et $\mathcal{M}_y^{\mathcal{C}_k,\mathcal{C}_{k'}}$.	**Variance** $f_4 = \frac{1}{N_c^2}\sum_{i=1}^{N_c}\sum_{j=1}^{N_c}(i-\mu)^2\mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(i,j)$ où μ est la moyenne de la matrice $\mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}$.
Moment différentiel inverse $f_5 = \sum_{i=1}^{N_c}\sum_{j=1}^{N_c}\frac{1}{1+(i-j)^2}\mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(i,j)$	**Moyenne des sommes** $f_6 = \sum_{l=1}^{2N_c} l \cdot \mathcal{M}_{x+y}^{\mathcal{C}_k,\mathcal{C}_{k'}}(l)$
Variance des sommes $f_7 = \sum_{l=1}^{2N_c}(l-f_6)^2\mathcal{M}_{x+y}^{\mathcal{C}_k,\mathcal{C}_{k'}}(l)$	**Entropie des sommes** $f_8 = -\sum_{l=1}^{2N_c}\mathcal{M}_{x+y}^{\mathcal{C}_k,\mathcal{C}_{k'}}(l)\log\{\mathcal{M}_{x+y}^{\mathcal{C}_k,\mathcal{C}_{k'}}(l)\}$
Entropie $f_9 = -\sum_{i=1}^{N_c}\sum_{j=1}^{N_c}\mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(i,j)\log\{\mathcal{M}^{\mathcal{C}_k,\mathcal{C}_{k'}}(i,j)\}$	**Variance des différences** $f_{10} = \sum_{l=1}^{N_c}(l-\mu_{x-y})^2\mathcal{M}_{x-y}^{\mathcal{C}_k,\mathcal{C}_{k'}}(l)$
Entropie des différences $f_{11} = -\sum_{l=1}^{N_c}\mathcal{M}_{x-y}^{\mathcal{C}_k,\mathcal{C}_{k'}}(l)\log\{\mathcal{M}_{x-y}^{\mathcal{C}_k,\mathcal{C}_{k'}}(l)\}$	**Information sur la corrélation I** $f_{12} = \frac{f_9 - \text{HXY1}}{\max\{\text{HX},\text{HY}\}}$ HX et HY sont les entropies de $\mathcal{M}_x^{\mathcal{C}_k,\mathcal{C}_{k'}}$ et $\mathcal{M}_y^{\mathcal{C}_k,\mathcal{C}_{k'}}$
Information sur la corrélation II $f_{13} = (1-\exp[-2(\text{HXY2}-f_9)])^{\frac{1}{2}}$	**Coefficient de corrélation maximal** $f_{14} = (\text{plus grande valeur propre de } Q)^{\frac{1}{2}}$

Tableau A.1: Attributs de texture d'Haralick

A.3 Filtres de Gabor

Les filtres de Gabor sont des filtres passe-bande orientés qui présentent d'excellentes propriétés de localisation fréquentielle et spatiale. Un filtre de Gabor peut être défini comme une sinusoïde modulée par une fonction gaussienne. Il est défini comme suit:

$$g(x,y;\gamma,\theta) = \exp^{-\frac{x'^2}{\sigma_x{}^2}+\frac{y'^2}{\sigma_y{}^2}} \exp^{j2\pi\gamma x'} \tag{A.8}$$

avec $x' = x\cos\theta + y\sin\theta$ et $y' = -x\sin\theta + y\sin\theta$. θ représente l'orientation de la fonction gaussienne, γ la fréquence centrale du filtre σ_x, et σ_y sont les écarts types associés à la fonction gaussienne respectivement le long des axes x et y.

Plusieurs filtres de Gabor (banc de filtres) sont utilisés pour caractériser les textures présentes dans les images en variant les paramètres θ et γ de la fonction de Gabor. La figure A.1 illustre un banc de 28 filtres obtenus en utilisant sept valeurs d'échelles (γ = 10, 20, 30, 40, 50, 60, et 70) et quatre orientations (θ = 0, 45, 90, et 135).

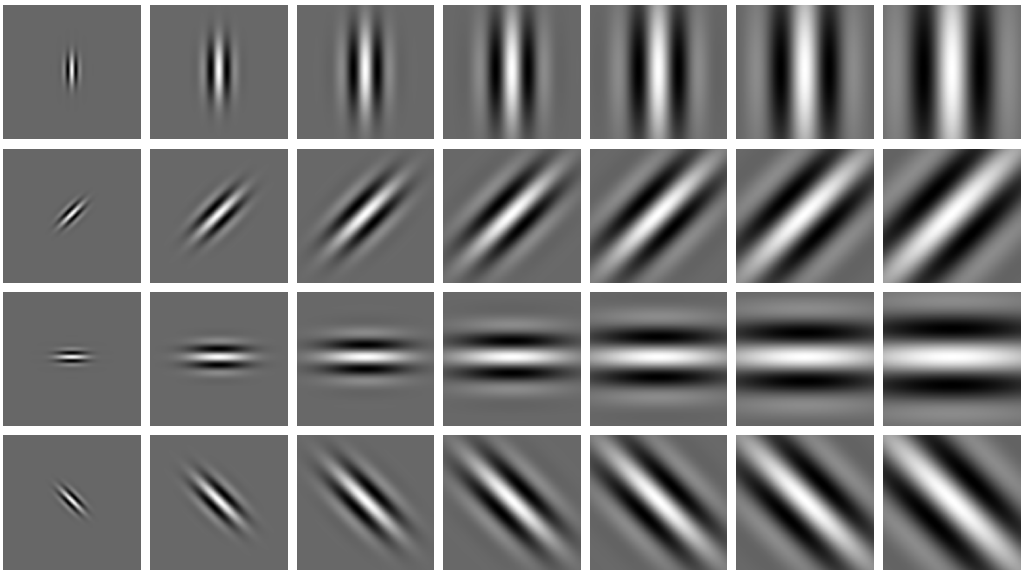


Figure A.1: Banc de filtres de Gabor avec sept échelles et quatre orientations (28 filtres).

Pour caractériser la texture couleur, les filtres de Gabor sont appliqués soit à chacune des composantes de l'image couleur (intra-composantes) [39] ou bien sur toutes les paires possibles de composantes de l'image couleur (inter et intra-composantes) [38] par l'intermédiaire de l'opération de convolution. Des attributs de texture sont alors déterminés pour chaque pixel de l'image en calculant les valeurs locales de l'énergie, de l'entropie et/ou de la variance sur chaque image filtrée.

Annexe B

Critères d'évaluation

Cette annexe présente les différents critères utilisés pour l'évaluation de la qualité des résultats de segmentation d'images.

Soient $X = \{x_1,x_2,\ldots,x_N\}$ l'ensemble des pixels de l'image (N étant le nombre total des pixels de l'image), S la segmentation à évaluer, et S' la segmentation de référence. On note par R_i, $i = 1,\ldots,k$, l'ensemble des régions de S et par R'_j, $j = 1,\ldots,k'$, l'ensemble des régions de S'. $N_i = |R_i|$ est le cardinal de la région R_i et $N'_j = |R'_j|$ est le cardinal de la région R'_j (on a $\sum_{i=1}^{k} N_i = \sum_{j=1}^{k'} N'_j = N$). La matrice de confusion est un outil qui permet de mesurer la concordance entre les pixels de S et de S'. Elle est de taille $k \times k'$. Chaque élément de cette matrice est le nombre de pixels en commun entre les régions $R_i \in S$ et $R'_j \in S'$, noté N_{ij}, et défini par: $N_{ij} = |R_i \cap R'_j|$.

B.1 Critères basés sur les régions

Hoover [229] a proposé cinq critères pour comparer les régions des segmentations S et S'. Ces critères sont basés sur un seuil de recouvrement minimum t défini par l'utilisateur dans l'intervalle $[0.5 \;\; 1]$ (dans nos évaluations t est fixé à 0.75).

$\uparrow$ **CS : Correct Detection**. Les régions R_i dans S et R'_j dans S' sont classées en détection correcte si et seulement si:

$$\begin{cases} |R_i \cap R'_j| \geq t \times |R_i| \\ |R_i \cap R'_j| \geq t \times |R'_j| \end{cases} \tag{B.1}$$

$\downarrow$ **OS : Over-Segmentation**. La sur-segmentation est détectée si la région R'_j dans S' est divisée en plusieurs régions $R_{i1},\ldots,R_{il}$ (avec $l \in [2 \;\; k]$) dans S comme suit:

$$\begin{cases} \forall s \in [1 \;\; l], \quad |R_{is} \cap R'_j| \geq t \times |R_{is}| \\ \sum_{s=1}^{l} |R_{is} \cap R'_j| \geq t \times |R'_j| \end{cases} \tag{B.2}$$

↓ **US : Under-Segmentation**. La sous-segmentation est détectée si la région R_i dans S est divisée en plusieurs régions $R'_{j1}, \ldots, R'_{jl}$ (avec $l \in [2 \;\; k']$) dans S' comme suit:

$$\begin{cases} \sum_{s=1}^{l} |R_i \cap R'_{js}| \geq t \times |R_i| \\ \forall s \in [1 \;\; l], \quad |R_i \cap R'_{js}| \geq t \times |R'_{js}| \end{cases} \tag{B.3}$$

↓ **ME : Missed Error**. La région R'_j dans S' est non détectée dans S si et seulement si :

$$\begin{cases} R'_j \notin & \text{Correcte segmentation} \\ R'_j \notin & \text{Sur-segmentation} \\ R'_j \notin & \text{Sous-segmentation} \end{cases} \tag{B.4}$$

↓ **NE : Noise Error**. La région R_i dans S est une région bruit (R_i ne possède pas de correspondante dans S') si et seulement si:

$$\begin{cases} R_i \notin & \text{Correcte segmentation} \\ R_i \notin & \text{Sur-segmentation} \\ R_i \notin & \text{Sous-segmentation} \end{cases} \tag{B.5}$$

B.2 Critères basés sur les pixels

Étant donné $N_{i,\bullet} = \sum_{j=1}^{k'} N_{i,j}$ et $N_{\bullet,i} = \sum_{j=1}^{k} N_{j,i}$ avec k et k' sont respectivement le nombre de régions (classes) de S et S', et $N_{i,j}$ le nombre de pixels attribués à la i-ème classe mais qui appartiennent à la j-ème classe. Soit $K = \max\{k,k'\}$. La matrice de confusion étendue à $K \times K$ est obtenue en remplissant les entrées $\{N_{i,j}\}$ manquantes par des zéros. Dans le cas supervisé $\hat{\imath}$ est i alors que dans le cas non supervisé $\hat{\imath}$ est la projection de la i-ème classe de S' dans S en utilisant la méthode de Munkres [230].

↓ **O : Omission error**. L'erreur d'omission représente la proportion globale des pixels perdus par une classe lors de la classification. Elle est définie par:

$$O = \text{median} \left\{ \frac{O_i}{N_{\bullet,i}} \right\}_{i=1}^{k'} = \text{median} \left\{ \frac{(N_{\bullet,i} - N_{\hat{\imath},i})}{N_{\bullet,i}} \right\}_{i=1}^{k'} \tag{B.6}$$

O_i est l'erreur d'omission de la i-ème classe.

↓ **C : Commission error**. L'erreur de commission représente la proportion globale des pixels ajoutés par une classe lors de la classification. Elle est définie par:

$$C = \text{median} \left\{ \frac{C_i}{N_{\hat{i},\bullet}} \right\}_{i=1}^{k} = \text{median} \left\{ \frac{(N_{\hat{i},\bullet} - N_{\hat{i},i})}{N_{\hat{i},\bullet}} \right\}_{i=1}^{k} \tag{B.7}$$

C_i est l'erreur de commission de la i-ème classe.

↑ **CA : Class Accuracy**. La mesure de précision de classe évalue le taux de confusion entre les classes après classification, telle que:

$$CA = \frac{1}{N} \sum_{i=1}^{K} \frac{N_{\hat{i},i} N_{\bullet,i}}{N_{\bullet,i} + N_{\hat{i},\bullet} - N_{\hat{i},i}} \tag{B.8}$$

↑ **CO : Recall - Accuracy**. Le rappel (taux de classification) représente le taux des pixels bien classés. Il est défini comme suit:

$$CO = \frac{1}{N} \sum_{i=1}^{K} N_{\bullet,i} CO_i = \frac{1}{N} \sum_{i=1}^{K} N_{\hat{i},i} \tag{B.9}$$

CO_i est le rappel de la i-ème classe.

↑ **CC : Precision**:

$$CC = \frac{1}{N} \sum_{i=1}^{K} N_{\bullet,i} CC_i = \frac{1}{N} \sum_{i=1}^{K} \frac{N_{\hat{i},i} N_{\bullet,i}}{N_{\hat{i},\bullet}} \tag{B.10}$$

CC_i est la précision de la i-ème classe.

↓ **I. : Type I error**:

$$I = \frac{1}{N} \sum_{i=1}^{K} \left(N_{\bullet,i} - N_{\hat{i},i} \right) = 1 - CO \tag{B.11}$$

↓ **II. : Type II error**:

$$II = \frac{1}{N} \sum_{i=1}^{K} \frac{N_{\hat{i},\bullet} N_{\bullet,i} - N_{\hat{i},i} N_{\bullet,i}}{N - N_{\bullet,i}} \tag{B.12}$$

↑ **EA : Mean Class Accuracy Estimate**:

$$EA = \frac{1}{N} \sum_{i=1}^{K} \frac{2 N_{\hat{i},i} N_{\bullet,i}}{N_{\bullet,i} + N_{\hat{i},\bullet}} \tag{B.13}$$

↑ **MS : Mapping Score**:

$$MS = \frac{1}{N} \sum_{i=1}^{K} \left(1.5 N_{\hat{i},i} - 0.5 N_{\hat{i},\bullet} \right) \tag{B.14}$$

↓ **RM : Root Mean Square Proportion Estimation Error** :

$$RM = \sqrt{\frac{1}{K}\sum_{i=1}^{K}\left(\frac{N_{\hat{\imath},\bullet} - N_{\bullet,i}}{N}\right)^2} \tag{B.15}$$

↑ **CI : Comparison Index** :

$$CI = \frac{1}{N}\sum_{i=1}^{K} N_{\hat{\imath},i}\sqrt{\frac{N_{\bullet,i}}{N_{\hat{\imath},\bullet}}} = \frac{1}{N}\sum_{i=1}^{K} N_{\bullet,i}\sqrt{CC_i CO_i} \tag{B.16}$$

↑ **F-measure**. Elle est basée sur la précision et le rappel et elle est définie comme suit:

$$\text{F-measure} = \frac{1}{N}\sum_{i=1}^{K} N_{\bullet,i}\frac{CC_i\ CO_i}{(1-\gamma)\ CC_i + \gamma\ CO_i} \tag{B.17}$$

γ est un paramètre qui appartient à l'intervalle [0 1].

↑ **PRI : Probabilistic Rand Index**. Il compte le nombre de paires de pixels qui sont soit dans les mêmes régions dans S et S' (n_{11}), soit dans différentes régions dans S et S' (n_{00}), soit dans les mêmes régions dans S mais dans différentes régions dans S' (n_{10}), et soit dans les mêmes régions dans S' mais dans différentes régions dans S (n_{01}). Il est défini par:

$$\text{PRI}(S,S') = \frac{n_{11}+n_{00}}{n_{11}+n_{10}+n_{01}+n_{00}} = \frac{2(n_{11}+n_{00})}{N(N-1)} \tag{B.18}$$

B.3 Critères basés sur les erreurs de cohérences

Martin et al. [227] ont défini quatre critères basés sur les erreurs de cohérences pour comparer les segmentations S et S'. Ces critères sont calculés sur chaque pixel de l'image à partir des erreurs, dites de raffinement. Pour un pixel x qui appartient à la région R_i dans S et à R'_j dans S', ces erreurs sont: i) l'erreur de raffinement local de S par rapport à S' définie par $LRE(S,S',x) = \frac{|R_i \setminus R'_j|}{|R_i|}$ et ii) l'erreur de raffinement local de S' par rapport à S définie par $LRE(S',S,x) = \frac{|R'_j \setminus R_i|}{|R'_j|}$. Ces quatre critères sont les suivants:

↓ **LCE : Local Consistency Error** :

$$LCE(S,S') = \frac{1}{N}\sum_{x}^{N} \min\left\{LRE(S,S',x), LRE(S',S,x)\right\} \tag{B.19}$$

↓ **GCE : Global Consistency Error** :

$$GCE(S,S') = \frac{1}{N}\min\left\{\sum_{x}^{N} LRE(S,S',x), \sum_{x}^{N} LRE(S',S,x)\right\} \tag{B.20}$$

↓ BCE : Bidirectional Consistency Error :

$$BCE(S,S') = \frac{1}{N}\sum_{x}^{N}\max\left\{LRE(S,S',x), LRE(S',S,x)\right\} \tag{B.21}$$

↓ GBCE : Global Bidirectional Consistency Error :

$$GBCE(S,S') = \frac{1}{N}\max\left\{\sum_{x}^{N} LRE(S,S',x), \sum_{x}^{N} LRE(S',S,x)\right\} \tag{B.22}$$

B.4 Critères de comparaison de classification

Ces critères se basent sur le calcul d'une métrique entres les deux segmentations S et S'.

↓ dM : Mirkin Metric. Cette métrique est liée au nombre pixels en désaccord dans les segmentations S et S'. Elle est définie comme suit:

$$dM(S,S') = \frac{1}{N^2}\left(\sum_{i}^{k} N_i^2 + \sum_{j}^{k'} N'^2_j - 2\sum_{i}^{k}\sum_{j}^{k'} N_{ij}^2\right) \tag{B.23}$$

↑ SDHD : Swapped Directional Hamming Distance. Cette métrique évalue la surface totale de chevauchement entre les régions dans S et S' par l'intermédiaire d'une distance de Hamming. La distance directionnelle de Hamming [231] d'une partition S vers une partition S' est définie par:

$$DHD(S,S') = \sum_{j}^{k'}\left(\sum_{i}^{k} N_{ij} - \max_i N_{ij}\right) \tag{B.24}$$

SDHD est alors définie comme suit:

$$\mathrm{SDHD}(S,S') = \mathrm{DHD}(S',S) = N^2 - \sum_{i=1}^{k}\max_j |R'_j \cap R_i| \tag{B.25}$$

↓ dD et ↑ VD : Van Dongen Metric [232]. Cette métrique est également basée sur la distance de Hamming (DHD) entre les segmentations S et S'. Elle est définie par:

$$dD(S,S') = \frac{DHD(S,S') + DHD(S',S)}{2N} \tag{B.26}$$

$$dD(S,S') = 1 - \frac{1}{2N}\left(\sum_{i}^{k}\max_j N_{ij} - \sum_{j}^{k'}\max_i N_{ij}\right) \tag{B.27}$$

$$VD(S,S') = 1 - dD \tag{B.28}$$

↑ **BGM : Bipartite Graph Matching**. Cette métrique mesure la correspondance maximale entre les régions dans S et S' en utilisant un graphe G bipartite pondéré par $w_{ij} = |R_i \cup R'_j|$. La correspondance maximale dans G est obtenue en supprimant les arrêtes par lesquelles les nœuds R_i et R'_j possèdent au plus une seule arrête incidente et que $\sum w_{ij}$ est maximale. BGM est définie comme suit:

$$\mathrm{BGM}(S,S') = 1 - \frac{\sum w_{ij}}{N} \tag{B.29}$$

↑ **SC : Segmentation Covering**.

$$SC(S,S') = \frac{1}{N} \sum_{i}^{k} N_i \max_j \frac{N_{ij}}{N_i + N'_j - N_{ij}} \tag{B.30}$$

↑ **SSC : Swapped Segmentation Covering**.

$$SSC(S,S') = SC(S',S) = \frac{1}{N} \sum_{j}^{k'} N'_j \max_i \frac{N_{ij}}{N_i + N'_j - N_{ij}} \tag{B.31}$$

↓ **dVI : Variation of Information.** Cette métrique utilise à la fois l'entropie et l'information mutuelle des segmentations S et S'. Elle est définie comme suit:

$$dVI(S,S') = H(S) + H(S') - 2MI(S,S') \tag{B.32}$$

ou H et MI représentent respectivement l'entropie et l'information mutuelle entre S et S'. L'entropie d'une segmentation S est définie par:

$$H(S) = -\sum_{i=1}^{k} \frac{N_i}{N} \log \frac{N_i}{N} \tag{B.33}$$

et l'information mutuelle MI entre S et S' est définie par:

$$MI(S,S') = -\sum_{i=1}^{k} \sum_{j=1}^{k'} \frac{N_{ij}}{N} \log \frac{N_{ij}}{N} \frac{N_i}{N} \frac{N'_j}{N} \tag{B.34}$$

↑ **NVOI : Normalized Variation Of Information**. Cette métrique est obtenue en normalisant la variation de l'information de l'équation (B.32) [233]. NVOI est définie par:

$$\mathrm{NVOI}(S,S') = 1 - \frac{1}{\log(k \times k')} \mathrm{dVI}(S,S') \tag{B.35}$$

Bibliographie

[1] S. Jain and V. Laxmi, "Color image segmentation techniques: A survey," in *Proceedings of the International Conference on Microelectronics, Computing & Communication Systems*. Springer, 2018, pp. 189–197.

[2] P. Bolon, J.-M. Chassery, J.-P. Cocquerez, D. Demigny, C. Graffigne, A. Montanvert, S. Philipp, R. Zéboudj, J. Zerubia, and H. Maître, *Analyse d'images: filtrage et segmentation*. Masson, 1995.

[3] F. Garcia-Lamont, J. Cervantes, A. López, and L. Rodriguez, "Segmentation of images by color features: A survey," *Neurocomputing*, vol. 292, pp. 1–27, 2018.

[4] S. Dahiya, S. Puri, and S. Singh, "Image segmentation techniques: A survey," *International Journal of Engineering and Applied Physics*, vol. 1, no. 2, p. 127–135, 2021.

[5] R. Rashmi, K. Prasad, C. B. K. Udupa, and V. Shwetha, "A comparative evaluation of texture features for semantic segmentation of breast histopathological images," *IEEE Access*, vol. 8, pp. 64 331–64 346, 2020.

[6] M. L. Benomar, M. Benazzouz, and M. El Habib Daho, "Colour texture features based approach for white blood cells segmentation," in *2019 International Conference on Networking and Advanced Systems (ICNAS)*, 2019, pp. 1–6.

[7] M. T. McCann, D. G. Mixon, M. C. Fickus, C. A. Castro, J. A. Ozolek, and J. Kovacević, "Images as occlusions of textures: A framework for segmentation," *IEEE Transactions on Image Processing*, vol. 23, no. 5, pp. 2033–2046, 2014.

[8] G. Liu, P. Li, and Y. Zhang, "A color texture image segmentation method based on fuzzy c-means clustering and region-level markov random field model," *Mathematical Problems In Engineering*, 2015.

[9] R. Gaetano, G. Scarpa, and G. Poggi, "Recursive texture fragmentation and reconstruction segmentation algorithm applied to vhr images," in *2009 IEEE International Geoscience and Remote Sensing Symposium*, vol. 4, 2009, pp. 101–104.

[10] L. Busin, N. Vandenbroucke, and L. Macaire, "Color spaces and image segmentation," in *Advances in Imaging and Electron Physics*. Elsevier, 2009, pp. 65–168.

[11] *International Commission on Illumination.* Available from : https://cie.co.at/, (Consulté le 30 mai 2021).

[12] N. Vandenbroucke, "Segmentation d'images couleur par classification de pixels dans les espaces d'attributs colorimétriques adaptés: application à l'analyse d'image," Ph.D. dissertation, 2000.

[13] O. D. Faugeras, "Digital color image processing and psychophysics within the framework of a human visual model," Ph.D. dissertation, 1976.

[14] Y.-I. Ohta, T. Kanade, and T. Sakai, "Color information for region segmentation," *Computer Graphics and Image Processing*, vol. 13, no. 3, pp. 222–241, 1980.

[15] R. M. Haralick, "Statistical and structural approaches to texture," *Proceedings of the IEEE*, vol. 67, no. 5, pp. 786–804, 1979.

[16] M. Tuceryan and A. K. Jain, *Texture analysis*. Editions World Scientific Publishing Co, 1998, pp. 235–276.

[17] J. Sklansky, "Image segmentation and feature extraction," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 8, no. 4, pp. 237–247, 1978.

[18] A. Humeau-Heurtier, "Texture feature extraction methods: A survey," *IEEE Access*, vol. 7, pp. 8975–9000, 2019.

[19] E. Cernadas, M. Fernandez-Delgado, E. González-Rufino, and P. Carrión, "Influence of normalization and color space to color texture classification," *Pattern Recognition*, vol. 61, pp. 120–138, 2017.

[20] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 3, no. 6, pp. 610–621, 1973.

[21] R. Azencott, J.-P. Wang, and L. Younes, "Texture classification using windowed fourier filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, pp. 148–153, 1997.

[22] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using gabor filters," *Pattern Recognition*, vol. 24, no. 12, pp. 1167–1186, 1991.

[23] M. Unser, "Texture classification and segmentation using wavelet frames," *IEEE Transactions on Image Processing*, vol. 4, no. 11, pp. 1549–1560, 1995.

[24] B. Manjunath and R. Chellappa, "Unsupervised texture segmentation using markov random field models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 5, pp. 478–482, 1991.

[25] J. Mao and A. K. Jain, "Texture classification and segmentation using multiresolution simultaneous autoregressive models," *Pattern Recognition*, vol. 25, no. 2, pp. 173–188, 1992.

[26] J. M. Keller, S. Chen, and R. M. Crownover, "Texture description and segmentation through fractal geometry," *Computer Vision, Graphics, and Image Processing*, vol. 45, no. 2, pp. 150–166, 1989.

[27] K. I. Laws, "Rapid Texture Identification," in *Image Processing for Missile Guidance*, vol. 0238, International Society for Optics and Photonics. SPIE, 1980, pp. 376 – 381.

[28] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[29] B. B. Chaudhuri and N. Sarkar, "Texture segmentation using fractal dimension," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 1, pp. 72–77, 1995.

[30] A. Drimbarean and P. F. Whelan, "Experiments in colour texture analysis," *Pattern Recognition Letters*, vol. 22, no. 10, pp. 1161–1167, 2001.

[31] C. Palm, "Color texture classification by integrative co-occurrence matrices," *Pattern Recognition*, vol. 37, no. 5, pp. 965–976, 2004.

[32] V. Arvis, C. Debain, M. Berducat, and A. Benassi, "Generalization of the cooccurrence matrix for colour images: application to colour texture classification," *Image Analysis & Stereology*, vol. 23, no. 1, pp. 63–72, 2004.

[33] T. Mäenpää and M. Pietikäinen, "Texture analysis with local binary patterns," in *Handbook of Pattern Recognition and Computer Vision*, 2005, pp. 197–216.

[34] F. Bianconi, R. Bello-Cerezo, and P. Napoletano, "Improved opponent color local binary patterns: an effective local image descriptor for color texture classification," *Journal of Electronic Imaging*, vol. 27, no. 1, p. 011002, 2017.

[35] C. Vertan, M. Ciuc, V. Buzuloiu, and C. Fernandez-Maloigne, "Compact color-texture run-length description for ornamental stones recognition and indexing," *Machine Vision Applications in Industrial Inspection X*, 2002.

[36] J. Da Rugna and H. Konik, "Automatic blur detection for meta-data extraction in content-based retrieval context," in *Internet imaging V*, vol. 5304. SPIE, 2003, pp. 285–294.

[37] J. Nicolás, M. J. Yzuel, and J. Campos, "Colour information as a third dimension in fourier transform and correlation," in *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, vol. 2, 2000, pp. 515–518.

[38] A. Jain and G. Healey, "A multiscale representation including opponent color features for texture recognition," *IEEE Transactions on Image Processing*, vol. 7, no. 1, pp. 124–128, 1998.

[39] G. Paschos, "Perceptually uniform color spaces for color texture analysis: an empirical evaluation," *IEEE Transactions on Image Processing*, vol. 10, no. 6, pp. 932–937, 2001.

[40] F. Bianconi, A. Fernández, F. Smeraldi, and G. Pascoletti, "Colour and texture descriptors for visual recognition: a historical overview," *Journal of Imaging*, vol. 7, no. 11, p. 245, 2021.

[41] P. Hiremath, S. Shivashankar, and J. Pujari, "Wavelet based features for color texture classification with application to cbir," *International Journal of Computer Science and Network Security*, vol. 6, no. 9A, pp. 124–133, 2006.

[42] S. Arivazhagan, L. Ganesan, and V. Angayarkanni, "Color texture classification using wavelet transform," in *Sixth International Conference on Computational Intelligence and Multimedia Applications (ICCIMA'05)*, 2005, pp. 315–320.

[43] Z. Kato and T.-C. Pong, "A markov random field image segmentation model for color textured images," *Image and Vision Computing*, vol. 24, no. 10, pp. 1103–1114, 2006.

[44] S. Panda and P. K. Nanda, "Color and texture segmentation using an unified mrf model," *Journal of Computer and Communications*, vol. 10, no. 6, pp. 139–164, 2022.

[45] A. R. Backes, D. Casanova, and O. M. Bruno, "Color texture analysis based on fractal descriptors," *Pattern Recognition*, vol. 45, no. 5, pp. 1984–1992, 2012.

[46] D. Casanova, J. B. Florindo, M. Falvo, and O. M. Bruno, "Texture analysis using fractal descriptors estimated by the mutual interference of color channels," *Information Sciences*, pp. 58–72, 2016.

[47] R. D. d. Silva, R. Minetto, W. R. Schwartz, and H. Pedrini, "Satellite image segmentation using wavelet transforms based on color and texture features," in *Advances in Visual Computing*. Springer, 2008, pp. 113–122.

[48] G. U. Maheswari, K. Ramar, D. Manimegalai, and V. Gomathi, "An adaptive region based color texture segmentation using fuzzified distance metric," *Applied Soft Computing*, vol. 11, no. 2, pp. 2916–2924, 2011.

[49] Y. Liu, G. Liu, C. Liu, and C. Sun, "A novel color-texture descriptor based on local histograms for image segmentation," *IEEE Access*, vol. 7, pp. 160 683–160 695, 2019.

[50] D. E. Ilea and P. F. Whelan, "Image segmentation based on the integration of colour–texture descriptors—a review," *Pattern Recognition*, vol. 44, no. 10, pp. 2479–2501, 2011.

[51] A. Humeau-Heurtier, "Color texture analysis: A survey," *IEEE Access*, vol. 10, pp. 107 993–108 003, 2022.

[52] Y. Deng and B. S. Manjunath, "Unsupervised segmentation of color-texture regions in images and video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 8, pp. 800–810, 2001.

[53] M. Mignotte, "Segmentation by fusion of histogram-based k-means clusters in different color spaces," *IEEE Transactions on Image Processing*, vol. 17, no. 5, pp. 780–787, 2008.

[54] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, no. 14, 1967, pp. 281–297.

[55] H. Permuter, J. Francos, and I. Jermyn, "A study of gaussian mixture models of color and texture features for image classification and segmentation," *Pattern Recognition*, vol. 39, no. 4, pp. 695–706, 2006.

[56] J.-S. Kim and K.-S. Hong, "Color–texture segmentation using unsupervised graph cuts," *Pattern Recognition*, vol. 42, no. 5, pp. 735–750, 2009.

[57] Y. Akbulut, Y. Guo, A. Şengür, and M. Aslan, "An effective color texture image segmentation algorithm based on hermite transform," *Applied Soft Computing*, vol. 67, pp. 494–504, 2018.

[58] P. Subudhi and S. Mukhopadhyay, "An efficient graph reduction framework for interactive texture segmentation," *Signal Processing: Image Communication*, vol. 74, pp. 42–53, 2019.

[59] N. Mevenkamp and B. Berkels, "Variational multi-phase segmentation using high-dimensional local features," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016, pp. 1–9.

[60] Y. Quan, H. Teng, T. Liu, and Y. Huang, "Weakly-supervised sparse coding with geometric prior for interactive texture segmentation," *IEEE Signal Processing Letters*, vol. 27, pp. 116–120, 2020.

[61] J. Yuan, D. Wang, and A. M. Cheriyadat, "Factorization-based texture segmentation," *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 3488–3497, 2015.

[62] M. Kumar, R. Saxena *et al.*, "Algorithm and technique on various edge detection: A survey," *Signal & Image Processing*, vol. 4, no. 3, p. 65, 2013.

[63] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.

[64] G. Wang, Z. Pan, Q. Dong, X. Zhao, Z. Zhang, and J. Duan, "Unsupervised texture segmentation using active contour model and oscillating information," *Journal of Applied Mathematics*, vol. 2014, no. none, pp. 1 – 11, 2014.

[65] L. Mabood, H. Ali, N. Badshah, K. Chen, and G. A. Khan, "Active contours textural and inhomogeneous object extraction," *Pattern Recognition*, vol. 55, pp. 87–99, 2016.

[66] G. Wang, J. Lu, Z. Pan, and Q. Miao, "Color texture segmentation based on active contour model with multichannel nonlocal and tikhonov regularization," *Multimedia Tools and Applications*, vol. 76, no. 22, pp. 24 515–24 526, 2017.

[67] Y. Dong, H. Zhang, Z. Liu, C. Yang, G.-S. Xie, L. Zheng, and L. Wang, "Neutrosophic set transformation matrix factorization based active contours for color texture segmentation," *IEEE Access*, vol. 7, pp. 93 887–93 897, 2019.

[68] K.-M. Chen and S.-Y. Chen, "Color texture segmentation using feature distributions," *Pattern Recognition Letters*, vol. 23, no. 7, pp. 755–771, 2002.

[69] X. Hu, C. V. Tao, and B. Prenzel, "Automatic segmentation of high-resolution satellite imagery by integrating texture, intensity, and color features," *Photogrammetric Engineering & Remote Sensing*, vol. 71, no. 12, pp. 1399–1406, 2005.

[70] P. Nammalwar, O. Ghita, and P. F. Whelan, "Integration of feature distributions for colour texture segmentation," in *Proceedings of the 17th International Conference on Pattern Recognition.*, vol. 1, 2004, pp. 716–719.

[71] G. Scarpa, G. Masi, R. Gaetano, L. Verdoliva, and G. Poggi, "Dynamic hierarchical segmentation of remote sensing images," in *International Conference on Image Analysis and Processing*. Springer, 2013, pp. 371–380.

[72] J. Chen, T. N. Pappas, A. Mojsilovic, and B. E. Rogowitz, "Adaptive perceptual color-texture image segmentation," *IEEE Transactions on Image Processing*, vol. 14, no. 10, pp. 1524–1536, 2005.

[73] L. G. Ugarriza, E. Saber, S. R. Vantaram, V. Amuso, M. Shaw, and R. Bhaskar, "Automatic image segmentation by dynamic region growth and multiresolution merging," *IEEE Transactions on Image Processing*, vol. 18, no. 10, pp. 2275–2288, 2009.

[74] I. Fondón, C. Serrano, and B. A. Pinero, "Color-texture image segmentation based on multistep region growing," *Optical Engineering*, vol. 45, no. 5, p. 057002, 2006.

[75] M. Baatz, "Multi resolution segmentation: an optimum approach for high quality multi scale image segmentation," in *Beutrage zum AGIT-Symposium*, 2000, pp. 12–23.

[76] Z. Wu and R. Leahy, "An optimal graph theoretic approach to data clustering: theory and its application to image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1101–1113, 1993.

[77] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, ser. NIPS'01, 2001, p. 849–856.

[78] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1124–1137, 2004.

[79] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[80] S. Han, W. Xu, W. Tao, and Y. Chen, "Color-texture cosegmentation based on nonlinear compact multi-scale structure tensor and tv-flow," *Signal Processing*, vol. 131, pp. 456–471, 2017.

[81] S. Han, W. Tao, D. Wang, X.-C. Tai, and X. Wu, "Image segmentation based on grabcut framework integrating multiscale nonlinear structure tensor," *IEEE Transactions on Image Processing*, vol. 18, no. 10, pp. 2289–2302, 2009.

[82] M. Haindl and S. Mikeš, "A competition in unsupervised color image segmentation," *Pattern Recognition*, vol. 57, pp. 136–151, 2016.

[83] C. Dharmagunawardhana, S. Mahmoodi, M. Bennett, and M. Niranjan, "Gaussian markov random field based improved texture descriptor for image segmentation," *Image and Vision Computing*, vol. 32, no. 11, pp. 884–895, 2014.

[84] A. Heshmati, M. Gholami, and A. Rashno, "Scheme for unsupervised colour–texture image segmentation using neutrosophic set and non-subsampled contourlet transform," *IET Image Processing*, vol. 10, no. 6, pp. 464–473, 2016.

[85] D. E. Ilea and P. F. Whelan, "Ctex—an adaptive unsupervised segmentation algorithm based on color-texture coherence," *IEEE Transactions on Image Processing*, vol. 17, no. 10, pp. 1926–1939, 2008.

[86] M. Palanivel and M. Duraisamy, "Color textured image segmentation using icicm-interval type-2 fuzzy c-means clustering hybrid approach," *Engineering Journal*, vol. 16, no. 5, pp. 115–126, 2012.

[87] S. Xu, L. Hu, C. Li, X. Yang, and X. P. Liu, "An unsupervised color-texture segmentation using two-stage fuzzy c-means algorithm," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 28, no. 02, p. 1455002, 2014.

[88] Y. Wang, J. Yang, and N. Peng, "Unsupervised color–texture segmentation based on soft criterion with adaptive mean-shift clustering," *Pattern Recognition Letters*, vol. 27, no. 5, pp. 386–392, 2006.

[89] M. Sujaritha and S. Annadurai, "Color texture segmentation using quaternion-wavelet filters and som," in *2010 Second International conference on Computing, Communication and Networking Technologies*, 2010, pp. 1–9.

[90] K. Salhi, E. M. Jaara, and M. T. Alaoui, "A neural approach for color-textured images segmentation," *International Journal of Computer and Information Engineering*, vol. 10, no. 10, pp. 1847–1850, 2016.

[91] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, "Spectral grouping using the nystrom method," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 2, pp. 214–225, 2004.

[92] G. Akbarizadeh and M. Rahmani, "Efficient combination of texture and color features in a new spectral clustering method for polsar image segmentation," *National Academy Science Letters*, vol. 40, pp. 117–120, 2017.

[93] S. B. Kotsiantis, I. Zaharakis, P. Pintelas *et al.*, "Supervised machine learning: A review of classification techniques," *Emerging artificial intelligence applications in computer engineering*, vol. 160, no. 1, pp. 3–24, 2007.

[94] S. H. Lee, H. I. Koo, and N. I. Cho, "Image segmentation algorithms based on the machine learning of features," *Pattern Recognition Letters*, vol. 31, no. 14, pp. 2325–2336, 2010.

[95] K. Zou, L. Ge, C. Zhang, T. Yuan, and W. Li, "Broccoli seedling segmentation based on support vector machine combined with color texture features," *IEEE Access*, vol. 7, pp. 168 565–168 574, 2019.

[96] Z. Haliche, K. Hammouche, O. Losson, and L. Macaire, "Fuzzy color aura matrices for texture image segmentation," *Journal of Imaging*, vol. 8, no. 9, 2022.

[97] L. Sankar and C. Chandrasekar, "Semi supervised image segmentation using optimal hierarchical clustering by selecting interested region as prior information," *Journal of Global Research in Computer Science*, vol. 2, no. 11, pp. 1–5, 2011.

[98] Y. Artan, "Interactive image segmentation using machine learning techniques," in *2011 Canadian Conference on Computer and Robot Vision*, 2011, pp. 264–269.

[99] W. Lei, "Semi-supervised spectral clustering combined with bayesian decision," in *2014 Seventh International Symposium on Computational Intelligence and Design*, vol. 1, 2014, pp. 437–440.

[100] I. Ahn and C. Kim, "Face and hair region labeling using semi-supervised spectral clustering-based multiple segmentations," *IEEE Transactions on Multimedia*, vol. 18, no. 7, pp. 1414–1421, 2016.

[101] Y. LeCun and Y. Bengio, *Convolutional Networks for Images, Speech, and Time Series*. Cambridge, MA, USA: MIT Press, 1998, p. 255–258.

[102] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 7, pp. 3523–3542, 2022.

[103] Y. Huang, F. Zhou, and J. Gilles, "Empirical curvelet based fully convolutional network for supervised texture image segmentation," *Neurocomputing*, vol. 349, pp. 31–43, 2019.

[104] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[105] V. Andrearczyk and P. F. Whelan, "Texture segmentation with fully convolutional networks," *arXiv preprint arXiv:1703.05230*, 2017.

[106] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[107] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.

[108] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6230–6239.

[109] W. Wang and J. Shen, "Deep visual attention prediction," *IEEE Transactions on Image Processing*, vol. 27, no. 5, pp. 2368–2378, 2018.

[110] E. Basaeed, H. Bhaskar, and M. Al-Mualla, "Supervised remote sensing image segmentation using boosted convolutional neural networks," *Knowledge-Based Systems*, vol. 99, pp. 19–27, 2016.

[111] E. Basaeed, H. Bhaskar, P. Hill, M. Al-Mualla, and D. Bull, "A supervised hierarchical segmentation of remote-sensing images using a committee of multi-scale convolutional neural networks," *International Journal of Remote Sensing*, vol. 37, no. 7, pp. 1671–1691, 2016.

[112] S. Yang, Y. Lv, Y. Ren, L. Yang, and L. Jiao, "Unsupervised images segmentation via incremental dictionary learning based sparse representation," *Information Sciences*, vol. 269, pp. 48–59, 2014.

[113] M. Kiechle, M. Storath, A. Weinmann, and M. Kleinsteuber, "Model-based learning of local image features for unsupervised texture segmentation," *IEEE Transactions on Image Processing*, vol. 27, no. 4, pp. 1994–2007, 2018.

[114] X. Jin, "Segmentation-based image processing system," 2012, uS Patent 8,260,048.

[115] A. V. Bhavsar, "An efficient weakly supervised approach for texture segmentation via graph cuts," *Journal Of Intelligent Systems*, vol. 22, no. 3, pp. 253–267, 2013.

[116] K. Wagstaff and C. Cardie, "Clustering with instance-level constraints," in *Proceedings of the Seventeenth International Conference on Machine Learning*, ser. ICML '00. Morgan Kaufmann Publishers Inc., 2000, p. 1103–1110.

[117] J. H. Ward Jr, "Hierarchical grouping to optimize an objective function," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, 1963.

[118] W. H. Day and H. Edelsbrunner, "Efficient algorithms for agglomerative hierarchical clustering methods," *Journal of classification*, vol. 1, no. 1, pp. 7–24, 1984.

[119] J. C. Bezdek, R. Ehrlich, and W. Full, "Fcm: The fuzzy c-means clustering algorithm," *Computers & Geosciences*, vol. 10, no. 2, pp. 191–203, 1984.

[120] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, vol. 96. AAAI Press, 1996, p. 226–231.

[121] R. Li and Z. Cai, "A clustering algorithm based on density decreased chain for data with arbitrary shapes and densities," *Applied Intelligence*, vol. 53, pp. 2098–2109, 2022.

[122] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, no. 1, pp. 1–22, 1977.

[123] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[124] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Transactions on Information Theory*, vol. 21, no. 1, pp. 32–40, 1975.

[125] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[126] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[127] V. Vapnik, *The Nature of Statistical Learning Theory*, 01 2000, vol. 8.

[128] R. Rojas, *Neural Networks: A Systematic Introduction.* Berlin, Heidelberg: Springer-Verlag, 1996.

[129] D. D. Lewis, "Naive (bayes) at forty: The independence assumption in information retrieval," in *European Conference on Machine Learning*, 1998, pp. 4–15.

[130] S. Knerr, L. Personnaz, and G. Dreyfus, "Single-layer learning revisited: a stepwise procedure for building and training a neural network," in *Neurocomputing*, 1990, pp. 41–50.

[131] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning.* The MIT Press, 2006.

[132] S. Basu, A. Banerjee, and R. Mooney, "Semi-supervised clustering by seeding," in *Proceedings of 19th International Conference on Machine Learning*, 2002, pp. 19–26.

[133] C. Wang, W. Chen, P. Yin, and J. Wang, "Semi-supervised clustering using incomplete prior knowledge," in *International Conference on Computational Science*, 2007, pp. 192–195.

[134] K. Kamvar, S. Sepandar, K. Klein, D. Dan, M. Manning, and C. Christopher, "Spectral learning," in *International Joint Conference of Artificial Intelligence*, 2003.

[135] Z. Lu and M. A. Carreira-Perpinan, "Constrained spectral clustering through affinity propagation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[136] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl, "Constrained k-means clustering with background knowledge," in *Proceedings of the Eighteenth International Conference on Machine Learning*, ser. ICML '01, 2001, p. 577–584.

[137] S. Basu, A. Banerjee, and R. J. Mooney, "Active semi-supervision for pairwise constrained clustering," in *Proceedings of the 2004 SIAM International Conference on Data Mining*, 2004, pp. 333–344.

[138] I. Davidson and S. Ravi, "Agglomerative hierarchical clustering with constraints: Theoretical and empirical results," in *Knowledge Discovery in Databases: PKDD 2005*, 2005, pp. 59–70.

[139] I. Davidson and S. S. Ravi, "Clustering with constraints: Feasibility issues and the k-means algorithm," in *Proceedings of the 2005 SIAM International Conference on Data Mining*, 2005, pp. 138–149.

[140] A. Majeed and I. Rauf, "Graph theory: A comprehensive survey about graph theory applications in computer science and social networks," *Inventions*, vol. 5, 2020.

[141] A. Bretto, A. Faisant, and F. Hennecart, *Éléments de théorie des graphes.* Springer, 2012.

[142] M. Meila and J. Shi, "Learning segmentation by random walks," in *Advances in Neural Information Processing Systems*, vol. 13, 2000, pp. 873–879.

[143] D. Cvetković, P. Rowlinson, and S. K. Simić, "Signless laplacians of finite graphs," *Linear Algebra and its Applications*, vol. 423, no. 1, pp. 155–171, 2007.

[144] G. Wacquet, P.-A. Hébert, E. C. Poisson, and D. Hamad, "Semi-supervised k-way spectral clustering using pairwise constraints." in *International Conference on Neural Computation Theory and Applications*, vol. 1, 2011, pp. 72–81.

[145] P. Perona and W. Freeman, "A factorization approach to grouping," in *European Conference on Computer Vision*, 1998, pp. 655–670.

[146] K. Rohe, S. Chatterjee, and B. Yu, "Spectral clustering and the high-dimensional stochastic blockmodel," *The Annals of Statistics*, vol. 39, no. 4, pp. 1878 – 1915, 2011.

[147] L. Zelnik-Manor and P. Perona, "Self-tuning spectral clustering," in *Advances in Neural Information Processing Systems*, vol. 17, 2005, pp. 1601–1608.

[148] C. Yuan, X. Qin, Z. Qin, and R. Wang, "Image segmentation based on modified superpixel segmentation and spectral clustering," *The Journal of Engineering*, vol. 2018, no. 16, pp. 1704–1711, 2018.

[149] T. B. Repository, "Tomas barton repository," 2013. [Online]. Available: https://github.com/deric/clustering-benchmark/tree/master/src/main/resources/datasets/artificial

[150] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[151] I. Gath and A. B. Geva, "Unsupervised optimal fuzzy clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 773–780, 1989.

[152] X. Wang and I. Davidson, "Flexible constrained spectral clustering," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2010, p. 563–572.

[153] N. Voiron, A. Benoit, A. Filip, P. Lambert, and B. Ionescu, "Semi-supervised spectral clustering with automatic propagation of pairwise constraints," in *13th International Workshop on Content-Based Multimedia Indexing*, 2015, pp. 1–6.

[154] I. Davidson, K. L. Wagstaff, and S. Basu, "Measuring constraint-set utility for partitional clustering algorithms," in *Knowledge Discovery in Databases*, 2006, pp. 115–126.

[155] A. A. Abin, "Querying informative constraints for data clustering: An embedding approach," *Applied Soft Computing*, vol. 80, pp. 31–41, 2019.

[156] Q. Xu, M. Desjardins, and K. Wagstaff, "Constrained spectral clustering under a local proximity structure assumption," in *FLAIRS Conference*. Citeseer, 2005.

[157] L. Xu, W. Li, and D. Schuurmans, "Fast normalized cut with linear constraints," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 2866–2873.

[158] Z. Li, J. Liu, and X. Tang, "Constrained clustering via spectral regularization," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 421–428.

[159] D. Chatel, P. Denis, and M. Tommasi, "Fast gaussian pairwise constrained spectral clustering," in *Machine Learning and Knowledge Discovery in Databases*, 2014, pp. 242–257.

[160] S. Basu, I. Davidson, and K. Wagstaff, *Constrained Clustering: Advances in Algorithms, Theory, and Applications*, 1st ed. Chapman & Hall/CRC, 2008.

[161] M. Bilenko, S. Basu, and R. J. Mooney, "Integrating constraints and metric learning in semi-supervised clustering," in *Proceedings of the Twenty-First International Conference on Machine Learning*, 2004, p. 11.

[162] R. Zebari, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed, "A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction," *Journal of Applied Science and Technology Trends*, vol. 1, no. 2, pp. 56–70, 2020.

[163] I. Jolliffe, *Principal Component Analysis*. Springer, 2002.

[164] S. J. Messick and R. P. Abelson, "The additive constant problem in multidimensional scaling," *Psychometrika*, vol. 21, no. 1, pp. 1–15, 1956.

[165] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[166] X. He and P. Niyogi, "Locality preserving projections," in *Advances in Neural Information Processing Systems*, vol. 16, 2003, pp. 153–160.

[167] R. Sheikhpour, M. A. Sarram, S. Gharaghani, and M. A. Z. Chahooki, "A survey on semi-supervised feature selection methods," *Pattern Recognition*, vol. 64, pp. 141–158, 2017.

[168] H. Liu and H. Motoda, *Computational methods of feature selection.* Chapman & Hall/CRC, 2007.

[169] R. Bellman, *Adaptive Control Processes – A Guided Tour.* Princeton University Press, 1961, vol. 2045.

[170] G. H. John, "Enhancements to the data mining process," Ph.D. dissertation, 1997.

[171] G. H. John, R. Kohavi, and K. Pfleger, "Irrelevant features and the subset selection problem," in *Machine Learning Proceedings*, 1994, pp. 121–129.

[172] L. Yu and H. Liu, "Efficient feature selection via analysis of relevance and redundancy," *Journal of Machine Learning Research*, vol. 5, p. 1205–1224, 2004.

[173] M. Dash and H. Liu, "Feature selection for classification," *Intelligent Data Analysis*, vol. 1, no. 1, pp. 131–156, 1997.

[174] H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 4, pp. 491–502, 2005.

[175] H. Almuallim and T. G. Dietterich, "Learning with many irrelevant features," in *Proceedings of the Ninth National Conference on Artificial Intelligence*, 1991, p. 547–552.

[176] H. Liul, H. Motoda, and M. Dash, "A monotonic measure for optimal feature selection," in *European Conference on Machine Learning*, 1998, pp. 101–106.

[177] W. Siedlecki and J. Sklansky, "On automatic feature selection," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 02, no. 02, pp. 197–220, 1988.

[178] T. Dokeroglu, A. Deniz, and H. E. Kiziloz, "A comprehensive survey on recent metaheuristics for feature selection," *Neurocomputing*, vol. 494, pp. 269–296, 2022.

[179] A. Porebski, "Sélection d'attributs de texture couleur pour la classification d'images. application à l'identification de défauts sur les décors verriers imprimés par sérigraphie," Ph.D. dissertation, Université Lille 1, 2009.

[180] T. M. Cover, *Elements of information theory.* John Wiley & Sons, 1999.

[181] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: a review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4–37, 2000.

[182] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16–28, 2014.

[183] J. Miao and L. Niu, "A survey on feature selection," *Procedia Computer Science*, vol. 91, pp. 919–926, 2016.

[184] P. Dhal and C. Azad, "A comprehensive survey on feature selection in the various fields of machine learning," *Applied Intelligence*, vol. 52, no. 4, pp. 4543–4581, 2022.

[185] R. A. Ghazy, E.-S. M. El-Rabaie, M. I. Dessouky, N. A. El-Fishawy, and F. E. A. El-Samie, "Feature selection ranking and subset-based techniques with different classifiers for intrusion detection," *Wireless Personal Communications*, vol. 111, no. 1, pp. 375–393, 2020.

[186] M. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of relieff and rrelieff," *Machine Learning*, vol. 53, no. 1-2, pp. 23–69, 2003.

[187] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[188] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection," in *Advances in Neural Information Processing Systems*, vol. 18, 2005.

[189] M. A. Hall and L. A. Smith, "Feature selection for machine learning: comparing a correlation-based filter approach to the wrapper." in *Proceedings of the Twelfth International Florida Artificial Intelligence Research Society Conference*, 1999, pp. 235–239.

[190] S. Hijazi, "Semi-supervised margin-based feature selection for classification," Ph.D. dissertation, Université du Littoral Côte d'Opale, 2019.

[191] L. Liu, J. Kang, J. Yu, and Z. Wang, "A comparative study on unsupervised feature selection methods for text clustering," in *2005 International Conference on Natural Language Processing and Knowledge Engineering*, 2005, pp. 597–601.

[192] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., 1995.

[193] D. Zhang, S. Chen, and Z.-H. Zhou, "Constraint score: A new filter method for feature selection with pairwise constraints," *Pattern Recognition*, vol. 41, no. 5, pp. 1440–1451, 2008.

[194] K. Kira and L. A. Rendell, "A practical approach to feature selection," in *Machine Learning Proceedings*, 1992, pp. 249–256.

[195] J. Zhao, K. Lu, and X. He, "Locality sensitive semi-supervised feature selection," *Neurocomputing*, vol. 71, no. 10, pp. 1842–1849, 2008.

[196] M. Kalakech, P. Biela, L. Macaire, and D. Hamad, "Constraint scores for semi-supervised feature selection: A comparative study," *Pattern Recognition Letters*, vol. 32, no. 5, pp. 656–665, 2011.

[197] K. Benabdeslem and M. Hindawi, "Constrained laplacian score for semi-supervised feature selection," in *Machine Learning and Knowledge Discovery in Databases*. Springer, 2011, pp. 204–218.

[198] K. Benabdeslem and M. Hindawi, "Efficient semi-supervised feature selection: Constraint, relevance, and redundancy," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 5, pp. 1131–1143, 2014.

[199] X.-K. Yang, L. He, D. Qu, W.-Q. Zhang, and M. T. Johnson, "Semi-supervised feature selection for audio classification based on constraint compensated laplacian score," *EURASIP Journal on Audio, Speech, and Music Processing*, no. 1, p. 9, 2016.

[200] X.-K. Yang, L. He, D. Qu, and W.-Q. Zhang, "Semi-supervised minimum redundancy maximum relevance feature selection for audio classification," *Multimedia Tools and Applications*, vol. 77, no. 1, pp. 713–739, 2018.

[201] K. Benabdeslem, H. Elghazel, and M. Hindawi, "Ensemble constrained laplacian score for efficient and robust semi-supervised feature selection," *Knowledge and Information Systems*, vol. 49, no. 3, pp. 1161–1185, 2016.

[202] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*, ser. ICML'96, 1996, p. 148–156.

[203] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 832–844, 1998.

[204] M. Liu and D. Zhang, "Pairwise constraint-guided sparse learning for feature selection," *IEEE Transactions on Cybernetics*, vol. 46, no. 1, pp. 298–310, 2016.

[205] A. Salmi, K. Hammouche, and L. Macaire, "Similarity-based constraint score for feature selection," *Knowledge-Based Systems*, vol. 209, p. 106429, 2020.

[206] Z. Zhao, F. Morstatter, S. Sharma, S. Alelyani, A. Anand, and H. Liu, "Advancing feature selection research," *Feature Selection Repository Arizona State University*, pp. 1–28, 2010.

[207] M. Kalakech, P. Biela, D. Hamad, and L. Macaire, "Constraint score evaluation for spectral feature selection," *Neural Processing Letters*, vol. 38, no. 2, pp. 155–175, 2013.

[208] A. Salmi, K. Hammouche, and L. Macaire, "Constrained feature selection for semi-supervised color-texture image segmentation using spectral clustering," *Journal of Electronic Imaging*, vol. 30, no. 1, pp. 1 – 28, 2021.

[209] L. Zhongmin, L. Bohao, L. Zhanming, and H. Wenjin, "Error based nyström spectral clustering image segmentation," in *Intelligent Computing Theories and Application*, 2016, pp. 546–556.

[210] W.-Y. Chen, Y. Song, H. Bai, C.-J. Lin, and E. Y. Chang, "Parallel spectral clustering in distributed systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 568–586, 2011.

[211] D. Cai and X. Chen, "Large scale spectral clustering via landmark-based sparse representation," *IEEE Transactions on Cybernetics*, vol. 45, no. 8, pp. 1669–1680, 2015.

[212] D. Yan, L. Huang, and M. I. Jordan, "Fast approximate spectral clustering," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '09, 2009, p. 907–916.

[213] J. Sourati, D. H. Brooks, J. G. Dy, and D. Erdogmus, "Constrained spectral clustering for image segmentation," in *2012 IEEE International Workshop on Machine Learning for Signal Processing*, 2012, pp. 1–6.

[214] L. Cong, S. Ding, L. Wang, A. Zhang, and W. Jia, "Image segmentation algorithm based on superpixel clustering," *IET Image Processing*, vol. 12, no. 11, pp. 2030–2035, 2018.

[215] S. Ji, H. Zhu, P. Wang, and X. Ling, "Image clustering algorithm using superpixel segmentation and non-symmetric gaussian–cauchy mixture model," *IET Image Processing*, vol. 14, no. 16, pp. 4132–4143, 2020.

[216] L. Liu, J. Chen, P. Fieguth, G. Zhao, R. Chellappa, and M. Pietikäinen, "From bow to cnn: Two decades of texture representation for texture classification," *International Journal of Computer Vision*, vol. 127, no. 1, pp. 74–109, 2019.

[217] N. M. Kwok, Q. P. Ha, and G. Fang, "Effect of color space on color image segmentation," in *2009 2nd International Congress on Image and Signal Processing*, 2009, pp. 1–5.

[218] H. Zou, W. Zhou, L. Zhang, C. Wu, R. Liu, and L. Jiao, "A new constrained spectral clustering for sar image segmentation," in *2009 2nd Asian-Pacific Conference on Synthetic Aperture Radar*, 2009, pp. 680–683.

[219] Y. Bengio, J.-f. Paiement, P. Vincent, O. Delalleau, N. Roux, and M. Ouimet, "Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering," in *Advances in Neural Information Processing Systems*, vol. 16. MIT Press, 2003.

[220] A. Trémeau and P. Colantoni, "Regions adjacency graph applied to color image segmentation," *IEEE Transactions on Image Processing*, vol. 9, no. 4, pp. 735–744, 2000.

[221] M. Haindl and S. Mikes, "Texture segmentation benchmark," in *Proceedings of the 19th International Conference on Pattern Recognition*, 2008, pp. 1–4.

[222] *Prague texture segmentation data generator and benchmark.* Available from: http://mosaic.utia.cas.cz, (accessed May 23, 2020).

[223] Z. Kato, T.-C. Pong, and J. Chung-Mong Lee, "Color image segmentation and parameter estimation in a markovian framework," *Pattern Recognition Letters*, vol. 22, no. 3, pp. 309–321, 2001.

[224] J. A. Ozolek and C. A. Castro, "Teratomas derived from embryonic stem cells as models for embryonic development, disease, and tumorigenesis," in *Embryonic Stem Cells-Basic Biology to Bioengineering*. IntechOpen, 2011, ch. 13.

[225] T. Ojala, T. Maenpaa, M. Pietikainen, J. Viertola, J. Kyllonen, and S. Huovinen, "Outex - new framework for empirical evaluation of texture analysis algorithms," in *2002 International Conference on Pattern Recognition*, vol. 1, 2002, pp. 701–706.

[226] J. Pont-Tuset and F. Marques, "Measures and meta-measures for the supervised evaluation of image segmentation," in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2131–2138.

[227] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proceedings Eighth IEEE International Conference on Computer Vision*, vol. 2, 2001, pp. 416–423.

[228] X. Jia, T. Lei, X. Du, S. Liu, H. Meng, and A. K. Nandi, "Robust self-sparse fuzzy clustering for image segmentation," *IEEE Access*, vol. 8, pp. 146 182–146 195, 2020.

[229] A. Hoover, G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bunke, D. B. Goldgof, K. Bowyer, D. W. Eggert, A. Fitzgibbon, and R. B. Fisher, "An experimental comparison of range image segmentation algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 7, pp. 673–689, 1996.

[230] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[231] R. W. Hamming, "Error detecting and error correcting codes," *The Bell System Technical Journal*, vol. 29, no. 2, pp. 147–160, 1950.

[232] S. Van Dongen, *Performance criteria for graph clustering and Markov cluster experiments*. Centre for Mathematics and Computer Science, 2000.

[233] A. Strehl, J. Ghosh, and R. Mooney, "Impact of similarity measures on web-page clustering," in *Workshop on artificial intelligence for web search (AAAI 2000)*, vol. 58, 2000, p. 64.

Segmentation d'images couleur texturées basée sur la sélection d'attributs et la classification spectrale semi-supervisées sous contraintes

Résumé

Les méthodes de segmentation par apprentissage automatique impliquent des techniques de caractérisation et de sélection d'attributs, ainsi qu'une technique de classification. Dans un contexte semi-supervisé, les méthodes de sélection d'attributs et de classification exploitent une petite quantité d'informations a priori donnée sous la forme de pixels prototypes ou de contraintes par paires, indiquant si deux pixels appartiennent à la même classe (must-link) ou non (cannot-link). L'apprentissage semi-supervisé sous contraintes suscite actuellement un vif intérêt car il permet de guider le processus de segmentation et d'améliorer ses performances sans nécessiter une grande base d'apprentissage comme dans le cas de l'apprentissage supervisé ou profond.

C'est dans ce contexte semi-supervisé que nous proposons, dans cette thèse, une méthode de segmentation d'images couleur texturées qui combine à la fois la sélection d'attributs et la classification semi-supervisées sous contraintes. Elle consiste à caractériser chaque pixel de l'image par un ensemble d'attributs de texture couleur. Une nouvelle méthode de sélection d'attributs de type filtre, basée sur un score de contraintes, est développée afin de choisir les attributs les plus pertinents. Ce score offre l'avantage d'évaluer la pertinence d'un sous-ensemble d'attributs à la fois et de déterminer automatiquement le nombre optimal d'attributs. Ces attributs sont finalement utilisés pour regrouper l'ensemble des pixels en classes via la méthode de classification spectrale sous contraintes. Des tests expérimentaux nous ont permis, d'une part, de valider le score de contraintes proposé en le comparant à d'autres scores de contraintes sur plusieurs bases de données. D'autre part, les résultats de segmentation obtenus sur des bases d'images texturées couleur ou en niveaux de gris, naturelles ou artificielles, médicales ou satellitaires attestent que la méthode de segmentation proposée est plus performante que les méthodes classiques, qu'elles soient supervisées, semi-supervisées, ou non supervisées, et reste compétitive avec les méthodes par apprentissage profond, malgré le peu d'information de supervision.

Mots clés — segmentation d'images couleur texturées; contraintes must-link et cannot-link; sélection d'attributs; score de contraintes; classification spectrale sous contraintes.

Color-texture image segmentation based on semi-supervised constrained feature selection and spectral clustering

Abstract

Machine learning segmentation methods involve feature extraction and feature selection procedures, as well as a classification technique. In a semi-supervised context, feature selection and classification methods exploit a small amount of a priori information given in the form of pixel prototypes or pairwise constraints indicating whether two pixels belong to the same class (must link) or not (cannot link). Constrained semi-supervised learning is currently attracting a lot of interest because it allows to guide the segmentation process and improve its performance without requiring a large learning datasets like supervised or deep learning.

In this semi-supervised context, we propose in this thesis a color-texture image segmentation method that combines both constrained feature selection and constrained clustering. It consists characterizing each pixel of the image with a set of color-texture features. A new filter feature selection method based on a constraint score is developed in order to select the most relevant features. The proposed constraint score is able to estimate the relevance of a subset of features at one time and to determinate automatically the optimal number of relevant features. Finally, the selected subset of features is used to classify all pixels in the image into classes through the constrained spectral clustering method. Experimental tests have allowed us, on one hand, to validate the proposed constraint score by comparing it to other constraint scores on several real benchmark datasets. On the other hand, the segmentation results achieved on benchmark datasets of color and grayscale textures, natural and artificial, medical and satellite, demonstrate that the proposed color-texture image segmentation method outperforms classical methods, whether they are supervised, semi-supervised, or unsupervised, and remains competitive with deep learning-based methods despite the limited amount of supervision information.

Keywords — color texture image segmentation; must-link and cannot-link constraints; feature selection; constraint score; constraint spectral clustering.