**Laboratoire des Sciences de l'Image,
de l'Informatique et de la Télédétection**

UMR ULP/CNRS 7005

École Doctorale Mathématiques, Sciences de l'Information et de l'Ingénieur

Thèse

présentée pour obtenir le grade de

Docteur de l'Université Louis Pasteur – Strasbourg I
Discipline : Informatique

par

Alexandre Blansché

Classification non supervisée avec pondération d'attributs par des méthodes évolutionnaires

Soutenue publiquement le 28 septembre 2006

Membres du jury

Directeur de thèse :	Jerzy J. Korczak, Professeur, Université Louis Pasteur
Co-Directeur de thèse :	Christiane Weber, Directrice de recherche, Université Louis Pasteur
Rapporteur interne :	Pascal Schreck, Professeur, Université Louis Pasteur
Rapporteur externe :	Younès Bennani, Professeur, Université Paris Nord
Rapporteur externe :	Yves Lechevallier, Directeur de recherche, INRIA-Rocquencourt
Examinateur :	Pierre Gançarski, Maître de Conférences, Université Louis Pasteur

An inspiration is what you are to me,
inspiration, look... see

—

Thank You
Led Zeppelin

Remerciements

Je tiens tout d'abord à faire part de toute ma reconnaissance à Pierre Gançarski. Il a encadré mon travail depuis ma maîtrise et m'a appris le métier de chercheur. Durant toutes ces années, il a été présent et a su me guider, me conseiller, tout en me laissant libre de mes choix. Je n'aurais pas pu mener à bien ce travail sans lui.

Je remercie Christiane Weber pour m'avoir encadré et avoir porté un regard extérieur à l'informatique sur mon travail. Elle m'a permis d'ouvrir mon horizon à la géographie, une discipline qui recèle encore bien des mystères pour moi.

Je remercie également Jerzy Korczak pour avoir dirigé ma thèse et pour ses remarques et ses conseils.

Je remercie Younès Bennani, Yves Lechevallier et Pascal Schreck pour l'intérêt qu'ils ont porté à mon travail et pour avoir accepté de rapporter ma thèse.

Je remercie les membres de l'équipe AFD du LSIIT pour leur aide et pour l'ambiance conviviale qu'ils ont su instaurer. Je ferai attention à l'avenir à ne pas « tout casser » lorsque je code. Merci également aux membres du LSIIT avec qui j'ai collaboré ou tout simplement partagé des repas.

Merci (et bravo) à Damien Vouriot qui s'est courageusement plongé dans mon code pour paralléliser l'algorithme MACLAW.

Je remercie également les membres du LIV de m'avoir accueilli « quelques semaines » dans leurs locaux. J'ai ainsi pu profiter de l'agréable ambiance du bureau 417, du thé et des petits gâteaux. Je remercie en particulier Annett, qui m'a beaucoup aidé dans l'application de mon travail en télédétection.

Bien entendu, je remercie mes amis qui m'ont aidé à sortir mon travail et m'ont permis de garder le moral. Un grand merci à ceux qui ont relu des parties de mon mémoire et qui m'ont permis de corriger les (très) nombreuses fautes de frappe, d'orthographe et de grammaire (et tout particulièrement à Marie-Laure qui a eu le courage de lire mon mémoire en entier malgré son « allergie » à l'informatique). Je remercie également de tout mon cœur Machin pour la choucroute.

Je tiens enfin à remercier ma mère qui a sacrifié beaucoup de choses pour que je puisse réussir mes études.

Table des matières

Introduction **1**

État de l'art **5**

1 Classification non supervisée **7**

1.1 Classification supervisée et non supervisée 7

1.2 Différentes approches pour la classification non supervisée 8

1.2.1 Résultat de la classification 8

1.2.1.1 Classification dure, douce et floue 8

1.2.1.2 Classification hiérarchique 10

1.2.2 Méthodes de regroupement 10

1.2.2.1 Méthodes basées sur une distance 11

1.2.2.2 Méthodes basées sur une grille 11

1.2.2.3 Méthodes probabilistes 12

1.2.2.4 Méthodes de formation de concepts 12

1.2.3 Synthèse 12

1.3 Évaluation de la qualité d'une classification non supervisée 12

1.3.1 Mesures basées sur une mesure de distance 13

1.3.2 Mesures de chevauchement dans les classifications floues 15

1.3.3 Mesures probabiliste 16

1.3.4 Méthodes de rééchantillonnage des données 17

1.3.5 Synthèse 17

1.4 Différentes méthodes 17

1.4.1 K-means 17

1.4.2 Fuzzy-C-means 18

1.4.3 SOM 18

1.4.4 EM . . . 19
1.4.5 COBWEB . . . 20
1.4.6 DBSCAN . . . 20
1.4.7 CURE . . . 20
1.4.8 Synthèse . . . 22
1.5 Conclusion . . . 22

2 Algorithmes évolutionnaires 23
2.1 Introduction . . . 23
2.2 Problème d'optimisation . . . 23
2.2.1 Méthodes classiques . . . 24
2.2.2 Méthodes stochastiques . . . 25
2.2.2.1 Recuit simulé . . . 25
2.2.2.2 Algorithmes évolutionnaires . . . 25
2.2.2.3 Optimisation par essaims particulaires . . . 28
2.2.3 Synthèse . . . 28
2.3 Modèles d'évolution artificielle . . . 29
2.3.1 Évolution darwinienne . . . 29
2.3.2 Évolution lamarckienne . . . 29
2.3.3 Évolution baldwinienne . . . 30
2.3.4 Comparaison des trois approches . . . 31
2.4 Coévolution génétique . . . 32
2.4.1 Caractéristiques des algorithmes coévolutionnaires . . . 32
2.4.2 Coévolution compétitive . . . 33
2.4.3 Coévolution coopérative . . . 33
2.4.4 Difficulté liées à la coévolution . . . 34
2.4.4.1 Effet *Red Queen* . . . 34
2.4.4.2 Initialisation des individus représentatifs . . . 35
2.5 Conclusion . . . 35

3 Sélection et pondération d'attributs 37
3.1 Problèmes liés à la dimensionnalité des données . . . 37
3.2 Caractéristiques des méthodes . . . 39
3.2.1 Utilisation des connaissances . . . 40
3.2.2 Espace de recherche . . . 40
3.2.2.1 Sélection d'attributs . . . 40
3.2.2.2 Pondération d'attributs . . . 40
3.2.3 Relation entre l'algorithme de sélection ou de pondération d'attributs et l'algorithme de classification . . . 41

3.2.3.1 Approche filtre . . . 41
3.2.3.2 Approche enveloppe . . . 41
3.2.3.3 Approche intégrée . . . 42
3.2.4 Portée des attributs sélectionnés et des pondérations . . . 42
3.2.5 Type d'évaluation . . . 43
3.2.5.1 Objectifs de l'évaluation . . . 44
3.2.5.2 Paradigmes utilisés . . . 44
3.3 Utilisation des pondérations . . . 45
3.3.1 Méthodes basées sur une distance . . . 45
3.3.2 Méthodes probabilistes . . . 46
3.3.3 COBWEB . . . 46
3.4 Méthodes de sélection d'attributs . . . 46
3.4.1 Classement des attributs . . . 46
3.4.2 Optimisation d'une fonction d'évaluation . . . 47
3.4.2.1 Algorithmes gloutons . . . 47
3.4.2.2 Méthodes stochastiques . . . 48
3.4.3 Classification non supervisée avec sélection locale des attributs . . . 48
3.4.3.1 L'algorithme CLIQUE . . . 48
3.4.3.2 L'algorithme PROCLUS . . . 49
3.4.3.3 L'algorithme FINDIT . . . 49
3.4.3.4 Synthèse . . . 50
3.5 Méthodes de pondération d'attributs . . . 50
3.5.1 Calcul direct . . . 50
3.5.2 Pondération d'attributs par optimisation d'une fonction d'évaluation . . . 51
3.5.3 Méthodes par approche intégrée basées sur K-means . . . 51
3.6 Évaluation de l'importance d'un attribut . . . 54
3.6.1 Critères basés sur l'entropie . . . 54
3.6.2 Critères basés sur la dépendance . . . 54
3.6.3 Synthèse . . . 55
3.7 Évaluation d'un sous-ensemble ou d'une pondération . . . 55
3.7.1 Critères basés sur l'entropie . . . 56
3.7.2 Critères basés sur les résultats de classification . . . 56
3.7.3 Synthèse . . . 56
3.8 Conclusion . . . 57

Méthodes proposées **59**

4 Approches génétiques pour l'amélioration des algorithmes de pondération d'attributs basés sur K-means **61**

4.1 Motivations . . . 61

4.2 Approche évolutionnaire 63
4.2.1 Encodage des solutions et opérations génétiques 63
4.2.1.1 Centres des classes 63
4.2.1.2 Pondérations 64
4.2.2 Algorithme darwinien évolutionnaire 65
4.2.3 Algorithme lamarckien évolutionnaire 65
4.2.4 Algorithme baldwinien évolutionnaire 66
4.3 Approche coévolutionnaire 67
4.3.1 Encodage des solutions et opérations génétiques 67
4.3.2 Algorithme darwinien coévolutionnaire 67
4.3.3 Algorithme lamarckien coévolutionnaire 68
4.3.4 Algorithme baldwinien coévolutionnaire 69
4.4 Évaluation des algorithmes 70
4.4.1 Données 70
4.4.1.1 Données artificielles 70
4.4.1.2 Données de l'UCI 71
4.4.2 Configuration des algorithmes 72
4.4.3 Comparaison selon la fonction d'évaluation 74
4.4.3.1 Comparaison des méthodes de pondération globale 74
4.4.3.2 Comparaison des méthodes de pondération locale 74
4.4.3.3 Conclusion sur l'optimisation de la fonction de coût 75
4.4.4 Comparaison selon des critères externes 75
4.4.4.1 Comparaison des méthodes de pondération globale 77
4.4.4.2 Comparaison des méthodes de pondération locale 77
4.4.4.3 Conclusion sur l'efficacité des méthodes à découvrir les classes réelles des données 79
4.4.5 Stabilité des résultats 79
4.4.5.1 Comparaison des méthodes de pondération globale 79
4.4.5.2 Comparaison des méthodes de pondération locale 79
4.4.5.3 Conclusion sur la stabilité des résultats 80
4.4.6 Comparaison des pondérations 80
4.4.6.1 Comparaison des méthodes de pondération globale 82
4.4.6.2 Comparaison des méthodes de pondération locale 83
4.4.6.3 Conclusion sur les pondérations 84
4.4.7 Temps de calcul 85
4.4.7.1 Comparaison des méthodes de pondération globale 85
4.4.7.2 Comparaison des méthodes de pondération locale 85
4.4.7.3 Conclusion sur le temps de calcul 87

4.5 Conclusion 87

5 MACLAW : un algorithme par approche modulaire pour la classification non supervisée avec pondération d'attributs 91

5.1 Introduction 91
5.2 Architecture générale de l'approche modulaire 92
5.2.1 Notions de base 92
5.2.2 Évaluation de la qualité de la classification 93
5.2.2.1 Degré de partitionnement 94
5.2.2.2 Qualité interne des classes 98
5.2.3 Optimisation 98
5.2.3.1 Encodage des individus : génotype et phénotype 98
5.2.3.2 Fonction d'évaluation 99
5.2.3.3 Évaluation des individus 99
5.2.3.4 Initialisation des individus représentatifs 101
5.2.3.5 Modification des individus représentatifs 102
5.2.4 Descriptif de l'algorithme 102
5.3 Approche modulaire pour pondération d'attribut 103
5.3.1 Extracteurs basés sur des classifieurs 103
5.3.2 Génotype des individus et opérations génétiques 105
5.4 Évaluation de la méthode MACLAW 106
5.4.1 Configuration des algorithmes 106
5.4.2 Comparaison selon la fonction d'évaluation 107
5.4.3 Comparaison selon des critères externes 108
5.4.3.1 Extracteurs basés sur K-means 109
5.4.3.2 Extracteurs basés sur EM 109
5.4.4 Stabilité des résultats 111
5.4.4.1 Extracteurs basés sur K-means 111
5.4.4.2 Extracteurs basés sur EM 111
5.4.5 Comparaison des pondérations 111
5.4.6 Temps de calcul 114
5.5 Conclusion 116

Application 117

6 Utilisation de MACLAW dans le cadre de l'observation de la Terre 119

6.1 Introduction 119
6.2 Expérience préliminaire 121
6.3 Expérimentations sur une image DAIS 123

6.3.1 Description des données . . . 124
6.3.2 Évaluation du résultat de classification de MACLAW . . . 125
6.3.3 Évaluation des pondérations obtenues par MACLAW . . . 126
6.3.3.1 Analyse par un expert . . . 126
6.3.3.2 Comparaison avec d'autres critère d'évaluation des attributs . . . 128
6.3.4 Résultats avec une méthode de classification supervisée . . . 129
6.3.4.1 Résultat de classification supervisée avec les 40 bandes . . . 131
6.3.4.2 Résultats de classification supervisée avec différents sous-ensembles des bandes . . . 132
6.4 Expérimentations sur une image Quickbird . . . 134
6.4.1 Description des données . . . 134
6.4.2 Résultats avec MACLAW . . . 136
6.5 Expérimentations sur une image CASI . . . 137
6.5.1 Description des données . . . 137
6.5.2 Résultats avec MACLAW . . . 139
6.6 Conclusion . . . 139

Conclusion **143**

Bibliographie **147**

Annexes **157**

A Mesures de distance et de similarité **159**
A.1 Distance et similarité . . . 159
A.2 Attributs numériques . . . 160
A.3 Attributs catégoriels . . . 160
A.4 Histogrammes . . . 161

B Critères d'évaluation supervisée pour la sélection ou la pondération d'attributs **163**
B.1 Évaluation de l'importance d'un attribut . . . 163
B.1.1 Critères basés sur la distance . . . 163
B.1.2 Critères basés sur l'entropie . . . 164
B.1.3 Critères basés sur la dépendance . . . 164
B.1.4 Synthèse . . . 165
B.2 Évaluation d'un sous-ensemble ou d'une pondération . . . 165
B.2.1 Critères basés sur la distance . . . 165
B.2.2 Critères basés sur la consistance . . . 166

B.2.3 Synthèse 166

C Évaluation d'une classification non supervisée par des critères externes 167

C.1 Comparaison de résultats de classification 167

C.2 Critères classiques 167

C.3 Indice de Wemmert et Gançarski 169

D Observation de la Terre 171

D.1 Télédétection 171

D.2 Applications de la télédétection 172

D.3 Extraction de l'information dans les images de télédétection 172

Liste des figures

1.1 Données numériques 8
1.2 Classification par partitionnement 9
1.3 Classification hiérarchique 10
1.4 Opérateurs dans COBWEB 21
2.1 Opérateurs génétiques classiques 27
2.2 Sélection des individus dans les modèles d'évolution darwinien, lamarckien et baldwinien 31
3.1 Données avec un attribut non pertinent 38
3.2 Données avec des attributs corrélés 38
3.3 Approche filtre pour la sélection/pondération d'attributs 41
3.4 Approche enveloppe pour la sélection/pondération d'attributs 42
3.5 Approche intégrée pour la sélection/pondération d'attributs 42
3.6 Données pour lesquelles une sélection locale des attributs est préférable 43
4.1 Partie du chromosome correspondant aux pondérations dans l'approche évolutionnaire dans le cas de pondérations globales 64
4.2 Partie du chromosome correspondant aux pondérations dans l'approche évolutionnaire dans le cas de pondérations locales 65
4.3 Partie du chromosome correspondant aux pondérations dans l'approche coévolutionnaire 67
4.4 Ensemble de données DA1 71
4.5 Ensemble de données DA2 72
4.6 Ensemble de données DA3 73
4.7 Évolution de la fonction d'évaluation au cours du temps pour les algorithmes de pondération globale 87
4.8 Évolution de la fonction d'évaluation au cours du temps pour les algorithmes de pondération locale 89
5.1 Classifieur modulaire 93
5.2 Initialisation des individus représentatifs par une méthode de classification 101
5.3 Extracteur basé sur un classifieur 104

5.4 Un chromosome dans une approche évolutionnaire 105
5.5 Un chromosome dans une approche coévolutionnaire 105
6.1 Bandes de l'image DAIS 122
6.2 Évolution de la qualité 122
6.3 Évolution des classes extraites 123
6.4 Extrait de l'image DAIS sur quatre bandes 124
6.5 Photographie aérienne correspondant à l'extrait de l'image DAIS 125
6.6 Résultat de l'algorithme MACLAW sur l'image DAIS 125
6.7 Résultat de la classification supervisée avec les 40 bandes 131
6.8 Résultat de la classification supervisée avec neuf bandes 132
6.9 Image Quickbird 135
6.10 Résultat de l'algorithme MACLAW sur l'image Quickbird 137
6.11 Extrait de l'image CASI sur quatre bandes 138
6.12 Photographie aérienne correspondant à l'extrait de l'image CASI 139
6.13 Résultat de l'algorithme MACLAW sur l'image CASI 140
D.1 Signatures spectrales de trois types de surfaces 172
D.2 Pixel mixte dans une image de télédétection 173

Liste des tableaux

1.1 Méthodes de classification non supervisée 22

3.1 Critères d'évaluation non supervisée de l'importance d'un attribut 55

3.2 Critères d'évaluation non supervisée de la pertinence d'une pondération des attributs 57

4.1 Algorithmes génétiques pour la pondération d'attributs dans K-means 62

4.2 Évaluation des algorithmes de pondération globale selon fonction $cost_{gaw}$ 75

4.3 Évaluation des algorithmes de pondération locale selon la fonction $cost_{law}$ 76

4.4 Évaluation des algorithmes de pondération globale par critères externes 77

4.5 Évaluation des algorithmes de pondération locale par critères externes 78

4.6 Stabilité des algorithmes de pondération globale 80

4.7 Stabilité des algorithmes de pondération locale 81

4.8 Degré d'utilisation des attributs par les algorithmes de pondération globale pour l'ensemble de données DA1 82

4.9 Degré d'utilisation des attributs par les algorithmes de pondération globale pour l'ensemble de données DA2 82

4.10 Degré d'utilisation des attributs par les algorithmes de pondération globale pour l'ensemble de données DA3 83

4.11 Degré d'utilisation des attributs par les algorithmes de pondération globale pour l'ensemble de données IRIS 83

4.12 Degré d'utilisation des attributs par les algorithmes de pondération locale pour l'ensemble de données DA1 83

4.13 Degré d'utilisation des attributs par les algorithmes de pondération locale pour l'ensemble de données DA2 84

4.14 Degré d'utilisation des attributs par les algorithmes de pondération locale pour l'ensemble de données DA3 84

4.15 Degré d'utilisation des attributs par les algorithmes de pondération locale pour l'ensemble de données IRIS 84

4.16 Temps de calcul des algorithmes de pondération globale 86

4.17 Temps de calcul des algorithmes de pondération locale 88

5.1 Évaluation du degré de partitionnement dans une classification dure 95

5.2 Évaluation du degré de partitionnement dans une classification floue 97

5.3 Évaluation d'un individu dans une approche coévolutionnaire 100
5.4 Évaluation des combinaisons 102
5.5 Configurations de MACLAW testées 107
5.6 Évaluation de degré de partitionnement (extracteurs basés sur K-means) 108
5.7 Évaluation de MACLAW par critères externes (extracteurs basés sur K-means) . . 110
5.8 Évaluation de MACLAW par critères externes (extracteurs basés sur EM) 111
5.9 Stabilité de MACLAW (extracteurs basés sur K-means) 112
5.10 Stabilité de MACLAW (extracteurs basés sur EM) 113
5.11 Degré d'utilisation des attributs par MACLAW pour l'ensemble de données DA1 (extracteurs basés sur K-means) 113
5.12 Degré d'utilisation des attributs par MACLAW pour l'ensemble de données DA2 (extracteurs basés sur K-means) 113
5.13 Degré d'utilisation des attributs par MACLAW pour l'ensemble de données DA3 (extracteurs basés sur K-means) 114
5.14 Degré d'utilisation des attributs par MACLAW pour l'ensemble de données IRIS (extracteurs basés sur K-means) 114
5.15 Temps de calcul de MACLAW (extracteurs basés sur K-means) 115
5.16 Temps de calcul de MACLAW (extracteurs basés sur EM) 116
6.1 Matrices de confusion entre les résultats de classification initial et final de MACLAW et la vérité-terrain 126
6.2 Évaluation des résultats de K-means et de MACLAW sur l'image DAIS par critères externes 126
6.3 Pondérations obtenues par MACLAW sur l'image DAIS 127
6.4 Classement des bandes selon l'indice DM 128
6.5 Classement des bandes selon l'indice IS 129
6.6 Classement des bandes selon les indices I, J et E 130
6.7 Matrice de confusion entre le résultat de classification supervisée avec les 40 bandes et les exemples d'apprentissage 131
6.8 Évaluation des résultats de l'algorithme supervisé avec 40 bandes sur l'image DAIS par critères externes 131
6.9 Comparaison entre les résultats de classification supervisée avec neuf bandes et le résultat de la classification supervisée avec 40 bandes 133
6.10 Matrice de confusion entre les résultats de classification supervisée avec neuf bandes et les exemples d'apprentissage 134
6.11 Évaluation des résultats de l'algorithme supervisé avec neuf bandes sur l'image DAIS par critères externes 134
6.12 Pondérations obtenues par MACLAW sur l'image Quickbird 137
B.1 Critères d'évaluation supervisée de l'importance d'un attribut 165
B.2 Critères d'évaluation supervisée de la pertinence d'une pondération des attributs . 166

Liste des algorithmes

2.1 Algorithme génétique 27
3.1 Algorithme de pondération d'attributs basé sur K-means 52
4.1 Algorithme darwinien évolutionnaire 66
4.2 Algorithme lamarckien évolutionnaire 66
4.3 Algorithme darwinien coévolutionnaire 68
4.4 Algorithme lamarckien coévolutionnaire 69

Introduction

Problématique

La classification est une étape importante pour l'analyse de données. Elle consiste à regrouper les objets d'un ensemble de données en classes homogènes. Il existe deux types d'approches : la classification supervisée et la classification non supervisée. Ces deux approches se différencient par leurs méthodes et par leur but. La *classification supervisée* (ang. *classification*) est basée sur un ensemble d'objets L (appelé ensemble d'apprentissage) de classes connues, le but étant de découvrir la structure des classes à partir de l'ensemble L afin de pouvoir généraliser cette structure sur un ensemble de données plus large. La *classification non supervisée* (ang. *clustering*) consiste à diviser un ensemble de données D en sous-ensembles, appelés classes (ang. *clusters*), tels que les objets d'une classe sont similaires et que les objets de classes différentes sont différents, afin d'en comprendre la structure sous-jacente.

Dans ce travail de thèse, nous nous intéresserons à la classification non supervisée. Les algorithmes de classification non supervisée sont souvent utilisés pour étudier des données pour lesquelles peu d'informations sont disponibles (trop peu d'exemples pour un apprentissage supervisé). Ainsi plusieurs recherches ont été menées sur la classification non supervisée au sein de l'équipe Apprentissage et Fouille de Données (AFD) du Laboratoire des Sciences de l'Image, de l'Informatique et de la Télédétection (LSIIT, ULP/CNRS UMR 7005), concernant les méthodes hiérarchiques [Ketterlin, 1995], les méthodes neuronales [Hammadi-Mesmoudi, 1995 ; Novak, 2000] ou la combinaison de plusieurs méthodes de classification [Wemmert, 2000].

Dans la perspective d'obtenir une classification plus précise, on cherche souvent à décrire les données de la manière la plus détaillée possible. Les données sont alors représentées par de nombreux attributs. Les attributs peuvent cependant présenter les caractéristiques pénalisantes suivantes :

- manque de pertinence : un attribut non pertinent n'apporte aucune information permettant de discriminer les classes entre elles ;
- bruit : un attribut bruité porte des informations incorrectes ;
- corrélations : des attributs corrélés portent la même information ; cette information redondante aura alors plus de poids qu'une information portée par un attribut indépendant ;
- ordre de grandeur : une même mesure peut s'exprimer selon différentes unités ; la choix de l'unité peut avoir une influence majeure sur le résultat de la classification, alors que l'information portée est intrinsèquement la même ;
- coût : la saisie de données sur de nombreux attributs peut avoir un coût important (en termes financiers ou en termes de temps), il est donc nécessaire de déterminer les attributs indispensables pour classifier les données.

Ainsi, l'augmentation de la dimensionnalité peut parfois nuire à la qualité de la classification. En effet, les méthodes de classification classiques ne sont pas adaptées à des données de grande dimensionnalité et présentant les caractéristiques citées.

La recherche de nouveaux algorithmes de classification plus efficaces pour ce type de données conduit souvent à des solutions *ad hoc*. Une autre approche, communément utilisée, consiste à adapter les données aux algorithmes de classification. Différentes méthodes existent [Motoda et Liu, 2003] et peuvent être catégorisées comme suit :

- l'extraction d'attributs : consiste à transformer l'ensemble d'attributs de départ en un nouvel ensemble d'attributs, généralement plus petit, contenant la même information ;
- la construction d'attributs : consiste à créer de nouveaux attributs basés sur les relations entre les attributs existants ;
- la sélection et la pondération d'attributs : consistent à chercher des poids binaires (sélection d'attributs) ou réels (pondération d'attributs) afin de faire varier l'influence relative des attributs lors de la classification.

Nous avons choisi de nous intéresser à la sélection/pondération d'attributs car elle permet de traiter aisément des attributs de types hétérogènes (attributs numériques, catégoriels, histogrammes, intervalles, ...). En effet, les méthodes d'extraction d'attributs (telles que l'Analyse en Composantes Principales) sont généralement des méthodes statistiques applicables uniquement sur des données numériques. Les méthodes de construction d'attributs, quant à elles, nécessitent des traitements spécifiques entre chaque type d'attributs pour en extraire des relations pertinentes et demandent donc un travail laborieux.

Dans le cadre de la classification supervisée, de nombreux travaux sur la sélection/pondération d'attributs ont déjà été menés [John *et al.*, 1994 ; Wettschereck *et al.*, 1997 ; Blum et Langley, 1997 ; Aha, 1998 ; Motoda et Liu, 2003]. Ces travaux ont montré que :

- des poids réels (pondération d'attributs) permettent d'obtenir de meilleurs résultats de classification que des poids binaires (sélection d'attributs) ;
- une pondération d'attributs par approche filtre (poids indépendants de la méthode de classification), bien que plus rapide, produit de moins bons résultats que les approches enveloppe ou intégrée (poids liés à la méthode de classification) ;
- une pondération locale des attributs (ensemble de pondérations spécifiques à chaque classe) est plus efficace qu'une pondération globale.

Nous pensons que ces hypothèses, démontrées dans le cas de la classification supervisée, se vérifient également en classification non supervisée. Dans ce cadre, le domaine de recherche est plus récent et n'a donné lieu, à notre connaissance, qu'à peu de publications. Il n'est pas possible, à l'heure actuelle, de comparer expérimentalement les différentes approches car les méthodes existantes sont trop peu nombreuses. Les premières méthodes non supervisées développées sont des méthodes de sélection d'attributs [Dy et Brodley, 2000 ; Dash et Liu, 2000 ; Søndberg-Madsen *et al.*, 2003 ; Morita *et al.*, 2003]. Les travaux les plus récents concernent des méthodes basées sur K-means [Chan *et al.*, 2004 ; Frigui et Nasraoui, 2004 ; Huang *et al.*, 2005] et permettent d'obtenir des pondérations globales ou locales pour des classifications dures ou floues.

Beaucoup de ces algorithmes de classification avec sélection/pondération d'attributs sont basés sur l'optimisation d'une fonction d'évaluation mais, jusqu'à présent, les méthodes évolutionnaires ont été très peu utilisées, alors qu'elles sont pourtant connues pour leur efficacité à résoudre les problèmes d'optimisation dans des espaces de recherche de grande dimension [Goldberg, 1989]. Plusieurs travaux de recherche sur les algorithmes évolutionnaires ont d'ailleurs été menés au sein du LSIIT dans le cadre des prédictions boursières [Lipinski, 2004] ou de la classification supervisée [Quirin, 2005].

Approches proposées

Plusieurs méthodes d'optimisation basées sur l'algorithme K-means ont été mises au point dans des travaux récents [Chan *et al.*, 2004 ; Frigui et Nasraoui, 2004 ; Huang *et al.*, 2005] : ces méthodes

combinent classification (dure ou floue) et pondération (globale ou locale) des attributs. L'algorithme d'optimisation est inspiré de celui de K-means et consiste en trois optimisations partielles (classes des objets, centres des classes et pondération des attributs) répétées à chaque itération dans le but de minimiser une fonction de coût (qui dépend de l'algorithme utilisé). Cependant, il est connu que ce type d'approche pour l'optimisation est très sensible aux conditions initiales et a de forts risques d'être bloqué dans un minimum local. Or, les algorithmes évolutionnaires sont des méthodes d'optimisation robustes et efficaces.

Nous proposons donc d'utiliser des algorithmes évolutionnaires pour optimiser la fonction de coût présentée dans ces travaux. Nous avons ainsi étudié et défini une famille de méthodes de classification non supervisée avec pondération d'attributs basées sur K-means par une approche évolutionnaire. Différentes stratégies de recherche ont été étudiées : évolution classique et coévolution coopérative, modèles d'évolution darwinien, lamarckien ou baldwinien.

Néanmoins, deux limites théoriques, induites de la définition même du critère d'évaluation, ont été mises en évidence dans ces méthodes. D'une part, le critère d'évaluation défini implique que les classes cherchées sont convexes et définies par un prototype, ce qui ne correspond pas toujours à la structure des classes réelles dans les données. De plus, il est indispensable de disposer d'une mesure de distance, ce qui n'est pas le cas pour tous les types de données. D'autre part, la méthode est très sensible aux corrélations entre les attributs, ce qui a d'ailleurs pu être mis en évidence d'un point de vue expérimental.

Pour nous affranchir de ces limites, nous avons étudié et proposé un nouveau schéma théorique permettant de découvrir des classes non nécessairement convexes et ne nécessitant pas de mesure de distance. Ce schéma, appelé *approche modulaire pour la classification non supervisée* consiste à décomposer le problème de classification en K classes en K sous-problèmes d'extraction d'une classe. Chacun de ces K sous-problèmes se ramène à la recherche du « meilleur » extracteur en fonction d'un résultat global obtenu par l'ensemble des K extracteurs. Une telle solution globale est évaluée selon un critère nouvellement défini, tenant compte de la complémentarité des classes (le résultat de la classification n'étant pas nécessairement une partition) et de la qualité interne des classes. La recherche du meilleur ensemble d'extracteurs est réalisée par un algorithme de coévolution coopérative.

Dans ce cadre, nous avons étudié et défini une méthode de classification non supervisée avec pondération locale d'attributs par approche enveloppe appelée MACLAW. Dans cette méthode, l'extraction d'une classe est réalisée en deux étapes :

- une classification de l'ensemble des données est réalisée en utilisant une pondération globale des attributs. La méthode utilisée n'est pas imposée et peut être de n'importe quel type (méthode probabiliste ou basée sur une distance, par exemple) ;
- l'une des classes obtenues est sélectionnée comme classe extraite, selon un critère donné.

Il est à noter que l'algorithme MACLAW cherche des pondérations locales, mais que chacun des extracteurs n'utilise que des pondérations globales des données, ce qui permet d'utiliser toutes les méthodes classiques de classification non supervisée pour définir les extracteurs.

Enfin, les extracteurs pouvant être définis à partir d'algorithmes de classification différents, cette méthode peut être qualifiée de multi-stratégique [Kittler, 1998].

Application

Cette thèse s'inscrit dans le cadre du projet FoDoMuSt[1]. Il s'agit d'un projet ACI de fouille de données multi-stratégie pour extraire et qualifier la végétation urbaine à partir de bases de données d'images. Des méthodes de classification y sont utilisées pour créer des cartes d'occupation du sol, voire d'utilisation du sol (chaque classe correspondant à un type d'occupation ou d'utilisation du sol).

La classification peut être réalisée directement au niveau des pixels (l'algorithme de classification doit alors découvrir la classe de chacun des pixels). Chaque pixel de l'image est caractérisé

[1] http://lsiit.u-strasbg.fr/afd/fodomust/

par une information radiométrique sur différentes bandes spectrales. Or, les images hyperspectrales comportent de nombreuses bandes spectrales contiguës (plusieurs dizaines de bandes) et induisent de nouveaux problèmes :

- une augmentation forte de la dimensionnalité des données, c'est-à-dire du nombre de bandes spectrales ;
- des corrélations fortes entre les bandes liées à la contiguïté de celles-ci.

L'analyse peut également être réalisée au niveau des objets présents dans l'image (bâtiments, routes, zones de végétation, etc.) : une image est découpée en segments (ou régions) par un processus appelé *segmentation*. Ces segments sont alors classifiés. Idéalement, chaque segment doit représenter un objet de la scène et doit pouvoir être affecté à une classe thématique. Pour cela, les segments peuvent être caractérisés par des attributs variés (informations spectrales, de texture ou de forme), ce qui nous ramène au problème de leur pertinence pour la classification.

Dans ce contexte, nous nous proposons de vérifier la validité de l'algorithme MACLAW dont l'objectif est de traiter ces types de problèmes.

Plan du mémoire

Dans ce mémoire, nous commencerons par présenter un état de l'art des trois principaux domaines étudiés. Ainsi, les principes de base de la classification non supervisée ainsi que les principales notations seront introduits dans le chapitre 1, l'intérêt des algorithmes génétiques par rapport aux autres méthodes d'optimisation sera exposé dans le chapitre 2 et la problématique de la pondération d'attributs sera plus profondément étudiée dans le chapitre 3.

Nous présenterons ensuite les deux approches qui ont été étudiées et développées au cours de cette thèse. Les méthodes basées sur K-means seront présentées dans le chapitre 4. L'algorithme MACLAW, basé sur l'approche modulaire, sera présenté dans le chapitre 5. Dans ces deux chapitres, nous présenterons à la fois le cadre théorique de ces méthodes ainsi qu'une validation expérimentale sur différents ensembles de données (données artificielles, ensembles de données de l'UCI). Des comparaisons ont été effectuées à la fois avec des méthodes de classification classiques et des méthodes récentes de classification avec pondération des attributs.

L'apport de l'algorithme MACLAW dans le cadre de l'observation de la Terre et les résultats obtenus sur des images de télédétection, seront présentés dans le chapitre 6.

Finalement, nous conclurons sur notre travail et présenterons des perspectives de recherche, concernant principalement la classification modulaire.

État de l'art

Chapitre 1

Classification non supervisée

1.1 Classification supervisée et non supervisée

La classification est une étape importante dans l'analyse de données et consiste à regrouper les données en classes homogènes.

Le processus de classification peut se découper en deux phases. La première est une phase d'apprentissage, pendant laquelle l'algorithme cherche des règles de classification (au sens large), qui permettront de prédire la classe d'appartenance d'un objet en fonction de sa description. La seconde phase consiste à appliquer les règles de classification découvertes à un ensemble d'objets afin d'identifier la classe d'appartenance de chacun des objets.

La phase d'apprentissage d'un algorithme de *classification supervisée* est basée sur un ensemble d'objets L (appelé ensemble d'apprentissage) dont la classe de chacun est connue. Le but de la classification supervisée est de découvrir la structure des classes sur cet ensemble de données et de généraliser cette structure à un ensemble de données plus large.

La *classification non supervisée* (ang. *clustering*) consiste à diviser un ensemble de données D en sous-ensembles, appelés classes (ang. *clusters*), de sorte que les classes soient le plus homogène possible suivant un critère défini. Les critères les plus couramment utilisés sont la similarité entre les objets, la densité des classes ou des mesures probabilistes. Les objets sont regroupés selon le critère qu'utilise la méthode de classification employée. Les deux phases du processus de classification sont généralement appliquées au même ensemble de données.

Nous ne nous intéresserons ici qu'à la classification non supervisée. Le but de ce chapitre est d'introduire les notations et les concepts de base sur lesquels s'appuiera la suite de ce mémoire, et de mettre en évidence la diversité qu'il existe parmi les différentes méthodes de classification non supervisée. Des états de l'art plus complets ont été publiés, comme par exemple [Jain *et al.*, 1999 ; Grabmeier et Rudolph, 2002 ; Xu et Wunsch, 2005]. Dans ce chapitre, nous décrirons différentes approches possibles pour la classification non supervisée (section 1.2). Les principaux critères d'évaluation du résultat d'un algorithme de classification seront présentés dans la section 1.3. Finalement, nous détaillerons quelques méthodes de classification (section 1.4) avant de conclure sur la classification non supervisée.

Notations

- on note D l'ensemble de données composé des objets à classifier, avec $N = card\,(D)$;
- on note $F = \{F_1, \ldots, F_n\}$ l'ensemble des attributs qui caractérisent les objets de D ;
- on note $o \in D$ un objet de D avec $o = (o_1, \ldots, o_n)$, où o_j est la valeur de l'objet o pour l'attribut F_j.

Exemple :

Les données représentées sur la figure 1.1, sont composées de 800 objets. Chaque objet est représenté par deux attributs numériques. On distingue nettement quatre classes naturelles de 200 objets chacune. Cet ensemble de données simple servira à illustrer certaines méthodes de classification dans la suite de ce chapitre.

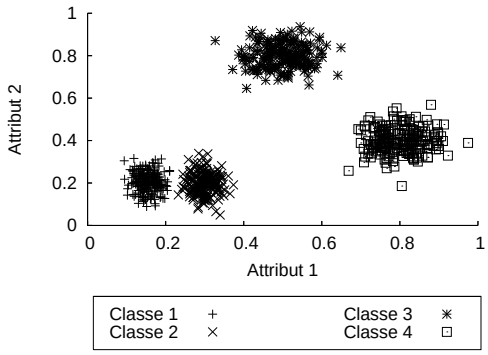


Fig. 1.1 : *Données numériques*

1.2 Différentes approches pour la classification non supervisée

La classification non supervisée est un domaine toujours très actif qui a engendré un nombre considérable de publications. De très nombreuses méthodes ont été définies, et il serait difficile, et hors de propos, d'en faire une liste exhaustive ici. On peut cependant distinguer différentes approches couramment utilisées.

Ces approches se distinguent tout d'abord par le type des résultats obtenus. Les classes peuvent être des ensembles durs ou flous. Il peut y avoir des objets non classés ou encore des superpositions entre les classes. Le résultat d'un algorithme de classification peut également prendre la forme d'une hiérarchie de classes. Les différents types de résultats seront détaillés dans la section 1.2.1.

Les algorithmes de classification non supervisée se distinguent également par la méthode de regroupement utilisée pour déterminer les classes. Les méthodes les plus souvent employées se basent sur une mesure de distance. D'autres approches sont basées sur une discrétisation des données, sur des modèles probabilistes ou sur la formation de concepts. Ces différentes méthodes seront présentées dans la section 1.2.2.

Une synthèse des différentes approches sera faite dans la section 1.2.3.

1.2.1 Résultat de la classification

Il existe différentes façons de représenter le résultat d'une classification, selon qu'il y ait des chevauchements entre les classes ou non (on distingue les classifications dures, douces et floues), et selon qu'il y ait des objets non classés (on parle alors de classifications partielles) ou non. Ces différents types de résultats sont présentés dans la section 1.2.1.1. Par ailleurs, le résultat d'une classification peut également être représenté sous la forme d'une structure plate ou d'une hiérarchie de classes (section 1.2.1.2).

1.2.1.1 Classification dure, douce et floue

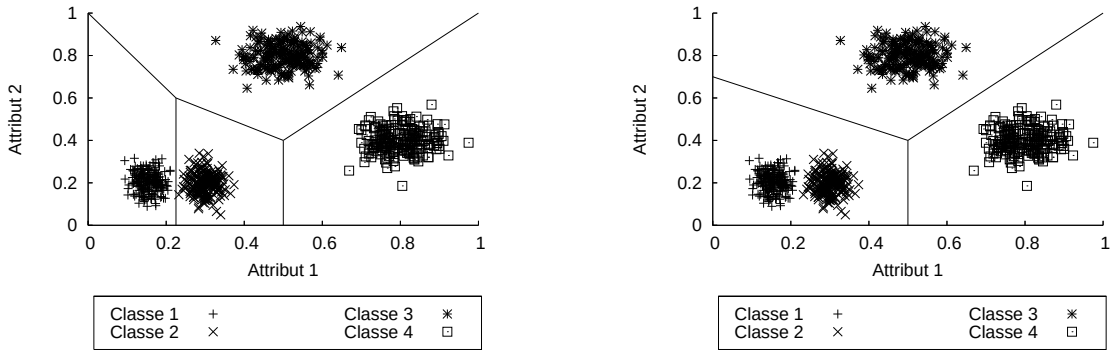
La plus simple façon de représenter les résultats d'un algorithme de classification non supervisée est une *classification dure* (ang. *hard clustering*). Dans une classification dure, chaque objet

appartient à une et une seule classe. L'ensemble de données D est donc divisé en un ensemble de classes $C = \{C_1, \ldots, C_K\}$ formant une partition de D, c'est-à-dire que $\bigcup_{k=1}^{K} C_k = D$ et $C_k \cap C_{k'} = \varnothing$ pour $k \neq k'$. On notera $C(o) = k$ le fait que $o \in C_k$.

Les méthodes de classification non supervisée produisant ce type de résultat sont appelées *méthodes de partitionnement.*

Exemple :

Si l'on considère l'ensemble de données présenté sur la figure 1.1, un algorithme de partitionnement en quatre classes produira les classes présentées sur la figure 1.2(a). Un algorithme de partitionnement en trois classes regroupera les classes 1 et 2 et produira les classes présentées sur la figure 1.2(b).



(a) *Classification en 4 classes* **(b)** *Classification en 3 classes*

Fig. 1.2 : *Classification par partitionnement*

Ce type de résultat à l'avantage d'être facilement interprétable par un utilisateur, mais il est parfois nécessaire de donner plus de flexibilité à la définition des classes. En effet, il se peut que des objets se distinguent trop des autres. Intégrer ce type d'objets dans les classes risque d'en altérer la qualité, et il est alors préférable de rejeter l'objet et de ne pas le classer. Ainsi, dans une *classification dure partielle* (ang. *partial hard clustering*), chaque objet appartient à au plus une classe. Une telle classification en K classes $C = \{C_1, \ldots, C_K\}$ vérifie néanmoins $C_k \cap C_{k'} = \varnothing$. On appelle objet atypique (ang. *outlier*) un objet o tel que $o \notin C_k, \forall C_k \in C$.

Dans une *classification douce* (ang. *soft clustering*), un objet peut appartenir à une ou plusieurs classes. L'ensemble de données D est donc divisé en un ensemble de classes $C = \{C_1, \ldots, C_K\}$ de telle sorte que $\bigcup_{k=1}^{K} C_k = D$.

On peut alors définir une *classification douce partielle* (ang. *partial soft clustering*) dans laquelle un objet peut appartenir à une ou plusieurs classes ou à aucune d'entre elle. L'ensemble de données D est donc divisé en un ensemble de classes $C = \{C_1, \ldots, C_K\}$ sans autre contrainte que celle du critère de regroupement utilisé par la méthode de classification.

Dans une *classification floue* (ang. *fuzzy clustering*), chaque objet o appartient à chacune des classes avec un certain degré d'appartenance $\mu_k(o)$ (chaque classe est définie par une fonction d'appartenance μ_k). On note l'ensemble des fonctions d'appartenance $\mu = \{\mu_1, \ldots, \mu_K\}$. Ces méthodes ont l'avantage de tenir d'avantage compte des objets stéréotypiques d'une classe par rapport aux objets atypiques ou à cheval entre deux classes. De nombreuses méthodes imposent $\sum_{k=1}^{K} \mu_k(o) = 1$.

Une classification floue reste cependant difficilement interprétable par un utilisateur et est souvent transformée en classification dure en affectant chaque objet o à la classe C_k telle que $\mu_k(o) = \max_{\mu_h \in \mu} \mu_h(o)$.

Dans [Cleuziou *et al.*, 2004] une méthode de transformation d'une classification floue en classification douce, appelé PoBOC, est proposée. Pour chaque objet o, on calcule le degré d'appartenance $\mu_k(o)$ de o dans chacune des classes $C_1, \ldots, C_K$ par un algorithme de classification floue. Les classes sont alors ordonnées par ordre décroissant du degré d'appartenance de o : on note $\widetilde{C}_k(o)$ la classe qui correspond au k-ième degré d'appartenance le plus élevé pour l'objet o et $\widetilde{\mu}_k(o)$ le degré d'appartenance de o à la classe $\widetilde{C}_k(o)$. L'objet o est ensuite affecté à la classe $\widetilde{C}_k(o)$ si l'un des conditions suivantes est respectée :

- $k = 1$;
- $1 < k < K$, $\widetilde{\mu}_k(o) \geq \frac{\widetilde{\mu}_{k-1}(o) + \widetilde{\mu}_{k+1}(o)}{2}$ et $\forall h < k, o \in \widetilde{C}_h(o)$;
- $k = K$ et $\widetilde{\mu}_K(o) \geq \frac{\widetilde{\mu}_{K-1}(o)}{2}$.

1.2.1.2 Classification hiérarchique

La plupart des méthodes représentent les résultats de classification sous la forme d'une structure plate.

Il est cependant naturel de représenter la connaissance sous forme de hiérarchie de classes ou de concepts. Par exemple, une classe de végétation découverte dans une image de télédétection peut-être divisée en différentes sous classes selon le type de végétation. Dans une *classification hiérarchique*, une classe peut être divisée en plusieurs sous-classes, l'ensemble des classes formant alors une hiérarchie (représentée par un arbre). Un objet appartient généralement à une et une seule classe-feuille mais aussi à la classe mère de celle-ci, etc. Les algorithmes de classification hiérarchique permettent de construire ce type de résultats. On distingue deux types d'approches : les méthodes agglomératives et les méthodes divisives. Les méthodes agglomératives partent d'un grand nombre de classes (éventuellement une classe par objet) et fusionnent les classes similaires entre elles. Les méthodes divisives partent de l'ensemble de données et le divisent en classes qui sont alors ensuite divisées récursivement.

Exemple :

Si l'on considère l'ensemble de données présenté sur la figure 1.1, un algorithme de classification hiérarchique produira la hiérarchie de classes présentées sur la figure 1.3. La racine de l'arbre correspond à l'ensemble de données à classifier.

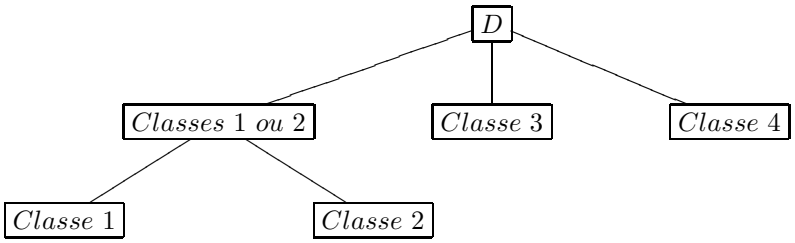


Fig. 1.3 : *Classification hiérarchique*

1.2.2 Méthodes de regroupement

Après avoir présenté les différents types de résultats d'un algorithme de classification, nous allons présenter les principales familles de méthodes utilisées pour regrouper les données dans des classes [Jain *et al.*, 1999 ; Grabmeier et Rudolph, 2002 ; Xu et Wunsch, 2005] :

- les méthodes basées sur une distance (section 1.2.2.1) ;
- les méthodes basées sur une grille (section 1.2.2.2) ;
- les méthodes probabilistes (section 1.2.2.3) ;
- les méthodes de formation de concepts (section 1.2.2.4).

1.2.2.1 Méthodes basées sur une distance

Pour définir les classes, de nombreuses méthodes se basent sur une mesure de distance ou de dissimilarité entre les objets (ces mesures sont présentées dans l'annexe A). Ces méthodes se basent sur l'idée que des objets proches appartiennent à une même classe. Trois types d'approches utilisent la notion de distance pour construire les classes : les méthodes basées sur un prototype, les méthodes neuronales et les méthodes basées sur la densité.

Les méthodes basées sur des prototypes (ang. *prototype-based methods*) sont des méthodes qui définissent les classes en fonction d'un objet représentatif de la classe. Cet objet représentatif est appelé *centroïde* s'il s'agit du centre de gravité des objets appartenant à la classe ou *médoïde* s'il s'agit d'un objet particulier de la classe. Les algorithmes K-means et Fuzzy-C-means sont les algorithmes les plus connus de cette famille de méthodes. Ces méthodes ne permettent généralement que de trouver des classes convexes dans l'espace de données considéré. Il existe cependant des méthodes, comme l'algorithme CURE, utilisant plusieurs individus représentatifs, permettant de découvrir des classes de formes plus complexes.

Le Perceptron [Rosenblatt, 1958] est le premier algorithme inspiré du fonctionnement des neurones biologiques. Il s'agit d'un algorithme de classification supervisée. Un *réseau de neurones* (artificiels) est un graphe orienté où les noeuds sont les *neurones* et les arrêtes sont les *axones*. Les réseaux de neurones sont généralement décomposés en couches. La *couche d'entrée* (neurones qui ont un degré entrant nul) est composée d'un neurone par attribut, la *couche de sortie* (neurones qui ont un degré sortant nul) est composée d'un neurone par classe. Entre ces deux couches, il peut y avoir une ou plusieurs couches de neurones dites *couches cachées*. Les arrêtes du graphe sont pondérées par des poids positifs (lien excitateur) ou négatif (lien inhibiteur). Quand un objet est présenté à un réseau de neurones, chaque attribut va activer plus ou moins un neurone de la couche d'entrée. Selon le degré d'activation des neurones d'une couche et des liens entre neurones, les neurones de la couche suivante sont plus ou moins activés. L'objet est associé à une classe en fonction du neurone de la couche de sortie le plus activé.

Dans les méthodes neuronales de classification non supervisée, comme SOM, le réseau est divisé en deux couches. Les neurones de la couche de sortie correspondent chacun à un prototype d'une classe (et possèdent donc des coordonnées dans l'espace des données). De plus, ces neurones sont liés entre eux : la modification d'un neurone pendant la phase d'apprentissage va provoquer une modification de ses voisins. Le nombre de neurones de la couche de sortie peut varier ainsi que les arrêtes qui les relient. Le degré d'activation d'un neurone de la couche de sortie dépend de la distance ou de la similarité par rapport à l'objet présenté au réseau (et non plus des poids des arrêtes). Généralement, plusieurs neurones sont utilisés pour définir une classe.

Les méthodes basées sur la densité consistent à définir la classe d'un objet en fonction de la classe des objets dans son voisinage le plus proche. Le voisinage d'un objet $o \in D$ peut être défini par $V(o) = \{o' \mid d(o, o') < \varepsilon\}$. Si $card(V(o))$ est suffisement élevé, o est considéré comme étant dans un région dense de l'espace. Un algorithme de type *Seed fill*, qui consiste à propager la classe dans le voisinage tant que celui-ci est dense, peut alors être utilisé. Ainsi, chaque zone dense dans l'espace des données sera vue comme une classe par l'algorithme. Contrairement aux méthodes basées sur des prototypes, ces méthodes permettent de découvrir des classes concaves dans l'espace de données. L'algorithme DBSCAN est un algorithme basé sur la densité.

1.2.2.2 Méthodes basées sur une grille

Les méthodes de classification basées sur une grille (ang. *grid-based clustering*) consistent à discrétiser l'espace des données (représentées par des attributs numériques), c'est-à-dire à découper

chaque attribut en intervalles de valeurs (pouvant être fixes ou dynamiques). Une *unité* se définit comme un point de cet espace discret ou comme un hyperrectangle construit par l'intersection d'un intervalle de chacune des dimensions de l'espace de données considéré. Les classes sont construites en utilisant un algorithme de type *Seed fill* en propageant les classes tant que le nombre d'objets par unités est supérieur à un seuil. Parmi ces méthodes, on retrouve l'algorithme CLIQUE.

1.2.2.3 Méthodes probabilistes

Un problème de classification peut se traduire par un problème d'estimation des paramètres d'une loi de probabilité, définie par un modèle de mélange de lois (le plus souvent un mélange gaussien). Ces méthodes font l'hypothèse qu'à chaque classe C_k est associée une loi de probabilité $P(o, \theta_k)$ de paramètres θ_k qui permet de déterminer la probabilité d'appartenance d'un objet o à la classe C_k. On note π_k la proportion de la k-ième loi dans le mélange. Les paramètres du mélange sont alors notés $\Phi = (\pi_1, \ldots, \pi_K, \theta_1, \ldots, \theta_K)$ et la fonction de densité est donnée par $P(o, \Phi) = \sum_{k=1}^{K} \pi_k P(o, \theta_k)$.

Les méthodes de classification basées sur ce principe, comme par exemple EM, cherchent à approximer les paramètres Φ du modèle. Pour chaque objet, une probabilité d'appartenance à chacune des classes, pouvant être assimilée à un degré d'appartenance, est calculée.

1.2.2.4 Méthodes de formation de concepts

Les méthodes de *formation de concept* ont pour objectif de construire, à partir d'un ensemble de données, une hiérarchie de concepts. Un concept donne une description intentionnelle d'une classe, en décrivant les propriétés communes des objets de la classe. Une hiérarchie est donc un « modèle » des données présentées. L'algorithme COBWEB est le plus couramment utilisé pour construire de telles hiérarchies.

La notion de concept peut-être étendue à celle de *concept probabiliste*. Un concept est alors un prototype d'une classe qui présente l'ensemble des attributs initiaux et sur lequel des critères probabilistes peuvent être calculés. Cela permet de caractériser la similarité intra-classe qui assure que les objets sont semblables et la dissimilarité inter-classes qui assure que deux concepts différents représentent des réalités différentes. C'est le cas par exemple de l'algorithme CLASSIT.

1.2.3 Synthèse

Nous avons vu que la classification non supervisée renvoie à un grand nombre d'approches très différentes les unes des autres, dans la façon de représenter les résultats et dans la définition même de ce qu'est une classe.

Le résultat d'une classification peut prendre la forme d'une structure plate ou d'une hiérarchie de classe. Un objet peut appartenir à plusieurs classes à la fois (éventuellement avec un certain degré d'appartenance) ou à aucune d'entre elles, selon le type de résultat proposé par la méthode de classification.

Une classe peut être définie par des prototypes, c'est à dire des objets stéréotypiques de la classes, par une relation de voisinage, par une loi de probabilité ou par une description des objets qui la composent.

1.3 Évaluation de la qualité d'une classification non supervisée

Avant de décrire les principales méthodes, il est nécessaire de décrire les différents critères d'évaluation d'une classification non supervisée. En effet, quelque soit le type de résultat que l'on

souhaite construire, il existe de nombreuses classifications possibles pour un ensemble de données. Par exemple, dans le cas d'une classification dure d'un ensemble D de N objets en K classes, il existe K^N classifications possibles. Or, sélectionner la « meilleure » classification parmi ces K^N possibilités n'a de sens que si l'on dispose d'un critère d'évaluation de celles-ci. Malheureusement, contrairement à la classification supervisée, il n'existe pas de critère naturel. En effet, dans le cas des algorithmes de classification supervisée, il est assez aisé de vérifier la validité des règles de classification obtenues, étant donné que la classe de chaque objet de l'ensemble d'apprentissage est connue. En revanche, un algorithme non supervisé n'a pas cette information à disposition. Des critères statistiques, évaluant la cohérence des classes ont dû être définis. Ces critères évaluent la pertinence de la classe par rapport au paradigme utilisé, par exemple si les objets d'une classe sont similaires entre eux et différents des objets des autres classes.

De fait, l'évaluation d'une classification réalisée par un algorithme non supervisé a engendré un grand nombre de travaux [Nicoloyannis *et al.*, 1997 ; Bezdek et Pal, 1998 ; Günter et Burke, 2001 ; Halkidi *et al.*, 2001b ; Halkidi *et al.*, 2001a ; Bolshakova et Azuaje, 2003]. Nous allons présenter ici une partie des critères qui ont été proposés dans la littérature. Ces critères d'évaluation se basent sur différentes notions. On distinguera principalement les critères basés sur une mesure de distance (section 1.3.1), les critères spécifiques aux classifications floues (section 1.3.2), les critères spécifiques aux méthodes de classification probabiliste (section 1.3.3) et les critères basés sur une méthode de rééchantillonnage des données (section 1.3.4). Une synthèse sur les critères d'évaluation sera faite dans la section 1.3.5.

1.3.1 Mesures basées sur une mesure de distance

De nombreux critères de qualité sont basés sur une mesure de distance entre les objets. Ils utilisent des notions de *compacité des classes* (critères intra-classes) ou de *séparabilité des classes* (critères inter-classes), ou les deux à la fois. La compacité est définie en fonction de la distance entre les objets d'une même classe ou en fonction de la distance entre les objets d'une classe et le centre de la classe. La séparabilité des classes est définie en fonction de la distance entre objets de classes différentes ou en fonction de la distance entre les centres des classes.

L'erreur au carré (ang. *square error*) est une des mesures de compacité les plus couramment utilisés dans le cas des classifications dures [MacQueen, 1965]. Une faible valeur indique une forte compacité des classes.

Définition 1.1 (Erreur au carré)
L'erreur au carré d'une classification non supervisée est définie par :

$$cost_{km}(c, C) = \sum_{1 \leq k \leq K} \sum_{o \in C_k} d(o, c_k)^2$$

avec $c = \{c_1, \ldots, c_n\}$ *les centres des classes et* $C = \{C_1, \ldots, C_n\}$ *les classes*

Cette définition peut être étendue à une classification floue en pondérant les distances par le degré d'appartenance aux classes. Une faible valeur indique toujours une forte compacité des classes.

Définition 1.2 (Erreur au carré (floue))
L'erreur au carré d'une classification non supervisée floue est définie par :

$$cost_{fcm}(c, \mu) = \sum_{1 \leq k \leq K} \sum_{o \in D} (\mu_k(o))^f \, d(o, c_k)^2$$

avec $C = \{C_1, \ldots, C_K\}$ *les classes,* $c = \{c_1, \ldots, c_K\}$ *leurs centres et* $\mu = \{\mu_1, \ldots, \mu_K\}$*, où* $\mu_k(o)$ *le degré d'appartenance de l'objet* o *à la* k*-ième classe et* f *un réel tel que* $f > 1$*.*

Le critère décrit dans [Dunn, 1974b] tient compte à la fois de la compacité et de la séparabilité des classes dans le cas d'une classification dure. Ce critère est cependant rarement utilisé car il est coûteux en temps de calcul. Une forte valeur indique une forte compacité des classes et une forte séparation des classes.

DÉFINITION 1.3 (INDICE DE DUNN)
*L'*indice de Dunn *d'une classification en K classes est défini par :*

$$D = \frac{\min\limits_{k,k' \in [1;K]} \left\{ \min\limits_{o \in C_k, o' \in C_{k'}} d(o,o') \right\}}{\max\limits_{k'' \in [1;K]} \left\{ \max\limits_{o,o' \in C_{k''}} d(o,o') \right\}}$$

Le critère décrit dans [Davies et Bouldin, 1979] tient compte à la fois de la compacité et de la séparabilité des classes dans le cas d'une classification dure. Une faible valeur indique une forte compacité des classes et une forte séparation des classes.

DÉFINITION 1.4 (INDICE DE DAVIES ET BOULDIN)
*L'*indice de Davies et Bouldin *est défini par :*

$$DB = \frac{1}{K} \sum_{k=1}^{K} \max_{k' \in [1;K] \setminus \{k\}} \left\{ \frac{S(C_k) + S(C_{k'})}{d(c_k, c_{k'})} \right\}$$

avec $S(C_k) = \frac{1}{|C_k|} \sum\limits_{o \in C_k} d(o, c_k)$

Le critère défini dans [Hubert *et al.*, 1985] mesure la séparabilité des classes dans le cas d'une classification dure. Une valeur élevée sur cet indice indique une forte séparation des classes.

DÉFINITION 1.5 (STATISTIQUE DE HUBERT MODIFIÉE)
La statistique de Hubert modifiée *est définie par :*

$$\Gamma = \sum_{o,o' \in D} d(o,o') S(o,o')$$

avec $S(o,o') = 1$ *si* o *et* o' *sont dans la même classe et* $S(o,o') = 0$ *sinon.*

Il est possible de normaliser cet indice pour qu'il prenne ses valeurs entre -1 et 1 :

$$\overline{\Gamma} = \frac{1}{M} \times \frac{1}{\sigma_d \sigma_S} \times \sum_{o,o' \in D} (d(o,o') - \mu_d)(S(o,o') - \mu_S)$$

où $M = \frac{N(N-1)}{2}$*, et* μ_d *et* μ_S *(respectivement* σ_d *et* σ_S*) représentent la moyenne (respectivement la variance) de* d *et de* S*.*

Un critère très connu pour l'évaluation de classifications floues, qui tient compte à la fois de la compacité et de la séparabilité des classes, a été présenté dans [Xie et Beni, 1991]. Une faible valeur indique une forte séparation des classes.

DÉFINITION 1.6 (INDICE DE XIE ET BENI)
*L'*indice de Xie et Beni *d'une classification floue en K classes est défini par :*

$$XB = \frac{\sum\limits_{k=1}^{K} \sum\limits_{o \in D} \mu_k(o)^2 \times d(o, c_k)^2}{n \times \min\limits_{o,o' \in D} d(o,o')^2}$$

avec $\sum_{k=1}^{K} \mu_k(o) = 1, \forall o \in D$.

L'indice de Xie et Beni peut également être défini en fonction du critère de qualité de Fuzzy-C-means par :

$$XB = \frac{cost_{fcm}}{n \times d_{min}^2}$$

avec $d_{min} = \min_{o,o' \in D} d(o, o')$.

L'indice de Xie et Beni peut enfin être étendu à une classification dure par :

$$XB = \frac{cost_{km}}{n \times d_{min}^2}$$

Le critère de compacité de Wemmert et Gançarski [Wemmert, 2000 ; Wemmert *et al.*, 2000] s'appuie sur le rapport entre deux distances pour chaque objet o : la distance entre o et le centre c_k de sa classe d'appartenance et la distance entre o et le centre différent de c_k le plus proche de o. Ce critère tient compte à la fois de la compacité et de la séparabilité des classes.

Définition 1.7 (Indice de compacité de Wemmert et Gançarski pour une classe)
*L'*indice de compacité de Wemmert et Gançarski pour une classe C_k *est défini par :*

$$WG_q(C_k) = \begin{cases} 0 & si \ \frac{1}{card(C_k)} \sum_{o \in C_k} \frac{d(o, c_k)}{d(o, c_{k'})} > 1 \\ 1 - \frac{1}{card(C_k)} \sum_{o \in C_k} \frac{d(o, c_k)}{d(o, c_{k'})} & sinon \end{cases}$$

où $k' = \underset{h \neq k}{\text{Argmin}}\,(d(o, c_h))$.

L'indice de compacité de Wemmert et Gançarski pour une classe prend ses valeurs sur $[0; 1]$.

Définition 1.8 (Indice de compacité de Wemmert et Gançarski)
*L'*indice de compacité de Wemmert et Gançarski *d'une classification en* K *classes est défini par :*

$$WG_q = \frac{1}{N} \sum_{k=1}^{K} card(C_k) \, WG_q(C_k)$$

L'indice de compacité de Wemmert et Gançarski prend ses valeurs sur $[0; 1]$.

1.3.2 Mesures de chevauchement dans les classifications floues

Des critères spécifiques aux classifications floues ont été définis. Ils mesurent le degré de chevauchement entre les classes. Moins une classification est «floue» (moins il y a de doute sur l'appartenance des objets aux classes), meilleure sera son évaluation selon ces critères. Ces mesures utilisent uniquement le degré d'appartenance des objets aux classes pour définir la qualité de la classification.

Un critère très connu pour l'évaluation de classifications floues a été présenté dans [Bezdek, 1974]. Cet indice permet de déterminer le degré de chevauchement entre les classes.

DÉFINITION 1.9 (INDICE DE BEZDEK)
*L'*indice de Bezdek *d'une classification floue en K classes est défini par :*

$$F = \frac{1}{N}\sum_{k=1}^{K}\sum_{o\in D}\mu_k(o)^2, \text{ avec } \sum_{k=1}^{K}\mu_k(o) = 1, \forall o \in D$$

L'indice de Bezdek prends ses valeurs sur $[0;1]$, une forte valeur indiquant une forte séparation des classes.

Un autre critère est l'entropie de partitionnement (ang. *partition entropy coefficient*) [Halkidi *et al.*, 2001b].

DÉFINITION 1.10 (ENTROPIE DE PARTITIONNEMENT)
*L'*entropie de partitionnement *d'une classification floue en K classes est définie par :*

$$PE = -\frac{1}{N}\sum_{k=1}^{K}\sum_{o\in D}\mu_k(o)\times\log(\mu_k(o)), \text{ avec } \sum_{k=1}^{K}\mu_k(o) = 1, \forall o \in D$$

Plus l'entropie de partitionnement est proche de 0, plus les classes sont séparées.

1.3.3 Mesures probabiliste

Les méthodes de classification probabilistes cherchent les paramètres $\Phi = (\pi_1, \ldots, \pi_K, \theta_1, \ldots, \theta_K)$ d'une loi de probabilité (loi de mélange de K lois), où la fonction de densité est notée $P(o, \Phi) = \sum_{k=1}^{K}\pi_k P(o, \theta_k)$. Si le modèle correspond à l'échantillon de données, la majorité des objets de l'échantillon devront avoir une probabilité élevée dans le modèle. Ceci est souvent évalué par la log-vraisemblance (Définition 1.11).

DÉFINITION 1.11 (LOG-VRAISEMBLANCE)
La log-vraisemblance *d'un modèle de mélange de paramètres Φ (avec une fonction de densité $P(o, \Phi)$) est définie par :*

$$L = \sum_{o\in D}\log(P(o, \Phi))$$

Une autre mesure probabiliste, utilisée dans le cadre des méthodes hiérarchiques de formation de concepts, est la prédictivité (Définition 1.12).

DÉFINITION 1.12 (PRÉDICTIVITÉ)
La prédictivité *PU d'un concept c composé de K sous-concepts $c_1, \ldots, c_K$ est définie par :*

$$\begin{aligned}
PU(c) &= \frac{1}{K}\left(\sum_{k=1}^{K}P(c_k)\times(\Pi(c_k) - \Pi(c))\right)\\
\Pi(c) &= \frac{1}{N}\left(\sum_{j=1}^{N}\Pi_j(c)\right)\\
\Pi_j(c) &= \sum_{i=1}^{m_j}P\left(o_j = F_j^i \mid o \in c\right), \textit{ si } F_j \textit{ est un attribut catégoriel}\\
\Pi_j(c) &= \frac{1}{2\sqrt{\pi}\sigma_{c,j}}, \textit{ si } F_j \textit{ est un attribut numérique}
\end{aligned}$$

où m_j est le nombre de modalités du j-ième attribut et F_j^i la i-ième modalité du j-ième attribut, $P\left(o_j = F_j^i \mid o \in c\right)$ est la probabilité que le j-ième attribut d'un objet o prenne la i-ième modalité et $\sigma_{k,j}$ est l'écart type du j-ième attribut pour le concept c.

1.3.4 Méthodes de rééchantillonnage des données

Il existe des méthodes plus complexes consistant à tester la stabilité d'un algorithme sur des sous-ensembles des données à classifier [Bel Mufti et Bertrand, 1997 ; Tibshirani *et al.*, 2000 ; Levine et Domany, 2001]. L'algorithme est alors appliqué plusieurs fois, avec les mêmes paramètres, sur différents sous-ensembles de D obtenus aléatoirement. Si la répartition des objets dans les classes obtenues en appliquant l'algorithme sur les sous-ensembles est identique à celle obtenue en appliquant l'algorithme sur l'ensemble D, l'algorithme peut être considéré comme stable et les paramètres corrects. De telles méthodes sont principalement utilisées pour chercher les paramètres optimaux des algorithmes, en particulier le nombre de classes.

1.3.5 Synthèse

La définition d'un critère d'évaluation de la qualité d'une classification non supervisée dépend de la définition que l'on se fait de la structure d'une classe. En effet, certain critères sont spécifiques à une définition précise des classes, comme par exemple l'erreur au carré (Définition 1.1) qui implique que les classes soient sphériques dans l'espace des données. De nombreuses méthodes de classification sont basées sur l'optimisation de l'un ou l'autre de ces critères d'évaluation, ce qui les limite une certaine définition des classes.

On remarque également que de nombreux critères ne sont pas normalisés, l'ordre de grandeur pouvant varier considérablement d'une jeu de données à l'autre, ce qui rend plus difficile leur interprétation.

1.4 Différentes méthodes

Après avoir présenté les différentes familles de méthodes de classification non supervisée et les principaux critères d'évaluation de la qualité d'une classification, nous allons détailler ici les méthodes les plus couramment utilisées. Un récapitulatif des caractéristiques de ces méthodes sera fait dans la section 1.4.8. Nous ne présenterons pas les algorithmes en détails, mais juste les principes élémentaires sur lesquels ils se basent, toujours afin de mettre en évidence la diversité qu'il existe parmis ces méthodes.

1.4.1 K-means

La méthode K-means [MacQueen, 1965] cherche un partitionnement défini en fonction des centres des classes en minimisant l'erreur au carré (Définition 1.1).

Chaque itération de l'algorithme comporte deux étapes. Chacune de ses étapes consiste à fixer l'un des paramètres de la fonction d'évaluation et à trouver la valeur optimale pour l'autre :

- redéfinition des centres : le centre c_k est défini comme l'isobarycentre des objets de la k-ième classe ;
- affectation des objets aux classes : un objet o est affecté à la classe dont le centre est le plus proche.

Les centres sont initialisés aléatoirement, par exemple en prenant K objets de D.

Une variante de cet algorithme (K-means adaptatif) consiste à présenter plusieurs fois les données une à une, dans un ordre aléatoire. À chaque objet o présenté, le centre le plus proche est déplacé vers l'objet o de la manière suivante :

$$c'_{k,i} = c_{k,i} + \eta\,(o_i - c_{k,i})$$

où $\eta \in [0;1]$ est le pas d'apprentissage (ang. *learning ratio*) qui peut être constant ou varier au cours du processus d'apprentissage.

Cet algorithme peut être implanté comme un réseau de neurones. On considère alors des attributs numériques, normalisés dans $[0;1]$. Le réseau de neurones est constitué de n neurones sur la couche d'entrée (un par attribut) et de K neurones sur la couche de sortie (un par classe). Chaque neurone d'entrée est connecté à chacun des neurones de la couche de sortie, ce qui fait $n \times K$ connexions. La connexion entre le j-ième neurone d'entrée et le k-ième neurone de sortie est pondérée par un poids $w_{k,j}$.

L'activation d'un neurone de sortie est la somme des activations des neurones d'entrée, pondérées par les poids des connections. Ainsi on calcule l'activation du k-ième neurone de sortie en fonction d'un objet o par $A_k(o) = \sum_{j=1}^{n} w_{k,j} \times o_i$. Le neurone le plus activé N_v est celui correspondant à la classe de l'objet.

Les poids des connexions sont choisis aléatoirement au début de l'apprentissage. L'ensemble des objets est alors présenté plusieurs fois au réseau de neurones. À chaque objet présenté, les poids des connexions du neurone de sortie le plus activé par un objet o sont modifiés de la manière suivante :

$$w'_{v,j} = \frac{w''_{v,j}}{\|w''_v\|}, \text{ avec } w''_{v,j} = w_{v,j} + \eta\,(o_j - w_{v,j})$$

où $\eta \in [0;1]$ est le pas d'apprentissage (ang. *learning ratio*) qui peut être constant ou varier au cours du processus d'apprentissage selon une fonction décroissante.

Cet algorithme est équivalent à K-means adaptatif. Chaque neurone de la couche de sortie représente un centre et les poids $w_{k,j}$ représentent les valeurs sur les attributs de c_k. Le degré d'activation $A_k(o)$ du k-ième neurone de sortie par un objet o est une mesure de similarité entre l'objet o et le centre de la k-ième classe (par un produit scalaire).

1.4.2 Fuzzy-C-means

Une version floue de l'algorithme K-means, appelée Fuzzy-C-means, a été définie [Dunn, 1974a ; Bezdek, 1981]. La fonction à minimiser est alors la version floue de l'erreur au carré (Définition 1.2).

Comme pour K-means, chaque itération de l'algorithme comporte deux étapes :

- redéfinition des centres : le centre c_k est défini comme le barycentre de tous les objets de D pondérés par leur degré d'appartenance à la k-ième classe ;
- calcul du degré d'appartenance des objets aux classes : le degré d'appartenance d'un objet o à la k-ième classe est recalculé, en fonction du paramètre f, par :

$$\mu_k(o) = \frac{1}{\sum_{k'=1}^{K} \left(\frac{d(o,c_k)}{d(o,c_{k'})} \right)^{\frac{2}{f-1}}}$$

1.4.3 SOM

La méthode des Cartes Auto-Organisatrices (ang. *Self-Organizing Maps*) [Kohonen, 1982] est un algorithme de classification non supervisée basé sur un réseau de neurones artificiels. Les H neurones de la couche de sortie sont reliés entre eux et forment un réseau, qui reste fixe durant

l'apprentissage. Chaque neurone possède des coordonnées dans l'espace des données et représente un prototype d'une classe. Deux neurones reliés vont s'influencer l'un l'autre au cours de l'apprentissage. Les topologies les plus souvent employées sont des topologies rectangulaires.

Une grille est utilisée pour déterminer à quel point un neurone N_h, $1 \leq h \leq H$ sera influencé par un objet o présenté au réseau de neurones.

À chaque objet présenté o, le neurone vainqueur N_v est déterminé par $N_v = \underset{N_h}{\text{Argmin}}\, d(o, N_h)$. Chaque neurone N_h est alors modifié :

$$N_h = N_h + \gamma(t)\,\Gamma(N_h, N_v, t)\,(o - N_h)$$

où la fonction $\gamma(t)$ est appelée taux d'apprentissage, définie par $\gamma(t) = \gamma_{max}\left(\frac{\gamma_{min}}{\gamma_{max}}\right)^{\frac{t}{t_{max}}}$, $\Gamma(N_h, N_v, t) = e^{\frac{-d^G(N_h,N_v)^2}{2\sigma(t)^2}}$, $\sigma(t) = \sigma_{max}\left(\frac{\sigma_{min}}{\sigma_{max}}\right)^{\frac{t}{t_{max}}}$, $d^G(N_h, N_{h'})$ est la distance de Manhattan sur la grille entre les neurones N_h et $N_{h'}$, t est initialisé à 0 et est incrémenté de 1 à chaque objet présenté.

L'algorithme SOM permet de découvrir le topologie des données, les neurones se stabilisant là où la densité des objets est la plus importante. Plusieurs neurones proches l'un de l'autre dans la topologie peuvent correspondre à une même classe.

1.4.4 EM

L'algorithme EM [Hartley, 1958 ; Dempster *et al.*, 1977] est un un algorithme de classification probabiliste, c'est-à-dire que l'algorithme cherche les paramètres $\Phi = (\pi_1, \ldots, \pi_K, \theta_1, \ldots, \theta_K)$ d'une loi de mélange. L'algorithme EM cherche à maximiser la log-vraisemblance (Définition 1.11). Chaque itération de l'algorithme se décompose en deux phases, une phase E (*Expectation*) et une phase M (*Maximization*).

La phase E consiste à évaluer $P(o, \Phi)$ pour chaque objet o en fonction des paramètres du modèle, ainsi que la probabilité d'appartenance de o à chacune des classes (assimilable à un degré d'appartenance) définie par :

$$t_k(o) = \frac{\pi_k \times P(o, \theta_k)}{\sum_{k'=1}^{K} \pi_{k'} \times P(o, \theta_{k'})}$$

La phase M consiste à estimer, en fonction des probabilités d'appartenance des objets aux classes, les paramètres du modèle qui maximisent la log-vraisemblance des objets. La définition des paramètres θ_k va dépendre du type de distribution de la classe C_k. La définition des paramètres π_k se fait par :

$$\pi_k = \frac{1}{N} \sum_{o \in D} t_k(o)$$

On suppose généralement que chaque classe suit une loi normale multidimensionnelle. Dans [Candillier *et al.*, 2005a ; Candillier *et al.*, 2005b ; Candillier *et al.*, 2005c], les auteurs utilisent un modèle simplifié en considérant les attributs indépendants les uns des autres. Dans ce cas, on peut définir $P(o, \theta_k) = \prod_{j=1}^{n} P(o_j, \theta_{k,j})$. Les paramètres du modèle sont alors simplement les moyennes et écarts types de chacun des attributs numériques et la fréquence des différentes modalités des attributs catégoriels.

1.4.5 COBWEB

L'algorithme COBWEB a été proposé dans [Fisher, 1987]. Il s'agit d'un algorithme de classification incrémental, hiérarchique travaillant sur des données catégorielles. Un algorithme générique de formation de concepts basé sur COBWEB est proposé dans [Ketterlin, 1995].

Les objets sont présentés un par un à la hiérarchie de concepts. Un objet o présenté à la hiérarchie sera tout d'abord inséré dans le concept racine, puis récursivement dans le concept le plus adapté, en modifiant l'arbre par différents opérateurs. Si le concept courant c est une feuille, l'objet est intégré au concept s'il correspond suffisamment. S'il ne correspond pas, un nouveau concept est créé. Si c n'est pas une feuille, différents opérateurs sont alors testés sur le concept c (FIG. 1.4). Ces différents opérateurs sont :

- l'incorporation de o à un sous-concept de c (FIG. 1.4(a)) ;
- la création d'un sous-concept nc de c restreint à o (FIG. 1.4(b)) ;
- la fusion de deux sous-concepts c_k et c_h de c en un concept nc, en intégrant o à nc (FIG. 1.4(c)) ;
- la scission d'un sous-concept de c en ses sous-concepts en intégrant o à l'un de ces sous-concepts (FIG. 1.4(d)).

Les résultats de chacun de ces quatre opérateurs sont évalués selon l'indice PU (Définition 1.12). Le meilleur résultat est appliqué et l'algorithme se poursuit récursivement avec le concept où o a été intégré.

1.4.6 DBSCAN

DBSCAN [Ester *et al.*, 1996] est un algorithme basé sur la densité. Trois notions concernant des types d'objets particuliers selon les relations de voisinages sont introduites :

Coeur d'une classe (ang. core*)* : un objet o est au coeur d'une classe C_k ($o \in core(C_k)$) si $card(\{o' \in D \mid d(o,o') < \varepsilon\}) \geq N_{min}$.

Bordure d'une classes (ang. bound*)* : un objet o est à la bordure d'une classe C_k si $card(\{o' \in D \mid d(o,o') < \varepsilon\}) < N_{min}$ et $\exists o' \in core(C_k) \mid d(o,o') < \varepsilon$.

Bruit (ang. noise*)* : un objet o est considéré comme du bruit si $card(\{o' \in D \mid d(o,o') < \varepsilon\}) < N_{min}$ et $\nexists o' \in core(C_k) \mid d(o,o') < \varepsilon$.

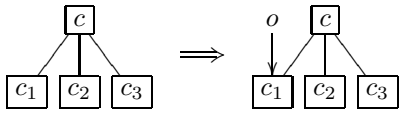
L'algorithme nécessite deux paramètres ε, un réel strictement positif, et N_{min} un entier strictement positif.

L'algorithme consiste à parcourir tous les objets de D. Pour chaque objet rencontré n'ayant pas encore été classifié et appartenant au cœur d'une classe, une classe est créée et s'étend de proche en proche, à la manière de l'algorithme *Seed fill*, tant que que les objets font partie du cœur de la classe. Les objets appartenant à la bordure de la classe y sont également intégrés, mais l'algorithme ne poursuit pas la recherche depuis ces objets. Les objets considérés comme du bruit ne sont intégrés dans aucune des classes.

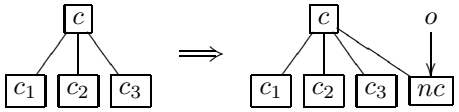
1.4.7 CURE

CURE [Guha *et al.*, 1998] est un algorithme basé sur une distance utilisant des prototypes pour définir les classes. La principale originalité de l'algorithme CURE est qu'il utilise plusieurs prototypes pour décrire une classe, contrairement à K-means par exemple. Les classes peuvent ainsi prendre des formes plus complexes que de simples sphères comme la plupart des algorithmes basés sur des prototypes.

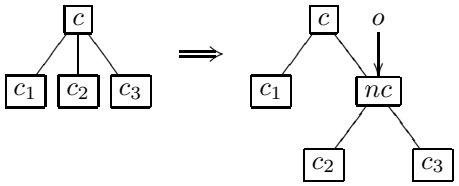
L'ensemble de prototypes $P_k = \{p_k^1, \ldots, p_k^m\}$ d'une classe C_k de centre c_k est défini par :



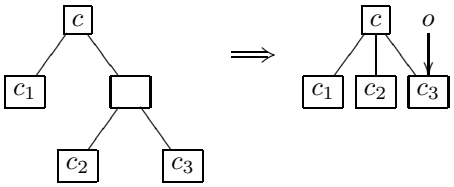
(a) *Incorporation à un sous-concept*



(b) *Création d'un nouveau sous-concept*



(c) *Fusion de deux sous-concepts*



(d) *Scission d'un sous-concept*

FIG. 1.4 : *Opérateurs dans COBWEB*

$$
\begin{aligned}
\hat{p}_k^1 &= \underset{o \in C_k}{\text{Argmax}} \left(d\left(o, c_k\right) \right) \\
\hat{p}_k^i &= \underset{o \in C_k}{\text{Argmax}} \left(\min_{1 \leq j < i} d\left(o, \hat{p}_k^j\right) \right), \forall 1 < i \leq m \\
p_k^i &= \hat{p}_k^i + \alpha \times \left(c_k - \hat{p}_k^i\right), \forall 1 \leq i \leq m
\end{aligned}
$$

La distance entre deux classes est alors définie par $d\left(C_k, C_{k'}\right) = \min_{p \in P_k, p' \in P_{k'}} d\left(p, p'\right)$

L'algorithme est initialisé en créant autant de classes que d'objets à classifier. À chaque itération de l'algorithme, les deux classes les plus proches sont fusionnées, jusqu'à ce qu'il ne reste plus que K classes.

1.4.8 Synthèse

Les caractéristiques de ces méthodes de classification non supervisée présentées dans ce chapitre sont résumées sur la table 1.1.

Méthode	Type de résultat	Critère de regroupement	Fonction à Optimiser	Référence
K-means	Classification dure	Distance, centroïdes	Erreur au carré	[MacQueen, 1965]
Fuzzy-C-means	Classification floue	Distance, centroïdes	Erreur au carré (floue)	[Dunn, 1974a]
SOM	Classification dure	Distance, réseau de neurones	/	[Kohonen, 1982]
EM	Classification floue	Probabilité	Log-vraisemblance	[Hartley, 1958]
COBWEB	Classification hiérarchique	Formation de concepts	/	[Fisher, 1987]
DBSCAN	Classification dure	Distance, densité	/	[Ester *et al.*, 1996]
CURE	Classification dure	Distance, prototypes	/	[Guha *et al.*, 1998]
PoBOC*	Classification douce	/	/	[Cleuziou *et al.*, 2004]

*PoBOC est un algorithme qui transforme une classification floue en classification douce

Tab. 1.1 : *Méthodes de classification non supervisée*

De très nombreuses méthodes de classification ont été définies. Ces méthodes se basent sur divers principes pour définir les classes (distance, loi de distribution, formation de concepts) et peuvent produire des résultats de types très variés (classification dure, douce, floue, ...).

1.5 Conclusion

La classification est une étape importante dans l'analyse de données qui consiste à regrouper les données en classes homogènes. La classification non supervisée consiste à diviser un ensemble de données D en sous-ensembles, appelés classes (ang. *clusters*), tels que les objets d'une classe sont similaires et que les objets de classes différentes sont différents, et ce afin d'en comprendre la structure sous-jacente.

Ce chapitre avait pour but de rappeler les notions de base en classification non supervisée, mais aussi de mettre en évidence la diversité qu'il existe parmi les différentes méthodes. Les algorithmes présentés dans la section 1.4 sont assez simples et sont généralement peu satisfaisants face à des données complexes c'est-à-dire lorsque le volume de données est trop grand, ou, comme nous le verrons dans la suite de ce mémoire, lorsque le nombre d'attributs est trop grand.

Chapitre 2

Algorithmes évolutionnaires

2.1 Introduction

Les algorithmes évolutionnaires sont des méthodes d'optimisation efficaces, connues pour leur robustesse et leur capacité à trouver de bonnes solutions dans un vaste espace de recherche. La particularité de ces méthodes d'optimisation est qu'elles considèrent simultanément une population de plusieurs solutions plutôt qu'une seule solution à la fois.

Nous avons vu dans le chapitre 1 que de nombreuses méthodes de classification consistent en l'optimisation d'une fonction d'évaluation. Comme nous le verrons dans ce chapitre, les algorithmes évolutionnaires sont connus pour leur efficacité et leur robustesse pour les problèmes d'optimisation. Durant cette thèse, nous nous sommes donc intéressés à valider l'utilisation de méthodes évolutionnaires comme technique d'apprentissage pour les algorithmes de classification.

Dans ce chapitre, nous présenterons le principe général des algorithmes évolutionnaires ainsi que les principales variantes. Dans la section 2.2 nous exposerons ce qu'est un problème d'optimisation et nous présenterons les principales méthodes pour résoudre ce type de problèmes, et en particulier les algorithmes génétiques. Nous présenterons ensuite différents modèles d'évolution pouvant être utilisés dans les algorithmes évolutionnaires (section 2.3). Nous discuterons ensuite des méthodes coévolutionnaires consistant à utiliser plusieurs populations en interaction dans la section 2.4. Nous terminerons ce chapitre par une conclusion sur les algorithmes génétiques dans la section 2.5.

Les différents algorithmes présentés dans ce chapitre seront illustrés par un problème d'optimisation d'un critère d'évaluation de classification non supervisée.

2.2 Problème d'optimisation

Définition 2.1 (Problème d'optimisation)
Un problème est un problème d'optimisation s'il peut s'écrire sous la forme :

Étant donnée une fonction $f : A \mapsto \mathbb{R}$ *trouver un élément* $x_0 \in A$ *tel que* $f(x_0) \geq f(x), \forall x \in A$ *(problème de maximisation) ou tel que* $f(x_0) \leq f(x), \forall x \in A$ *(problème de minimisation).*

Un algorithme d'optimisation est un algorithme qui cherche la meilleure solution à un problème d'optimisation. Ces problèmes étant souvent NP-complets, il est très difficile de trouver une solution exacte à ce problème et la plupart des méthodes ne sont que des heuristiques proposant une solution approchée ou « suffisamment bonne », c'est-à-dire une solution dont l'évaluation est suffisamment proche de l'extremum global.

Exemple :

Comme nous l'avons vu dans le chapitre 1, la classification non supervisée peut consister à optimiser une fonction d'évaluation, comme par exemple l'Erreur au carré (Définition 1.1) que nous rappelons ici :

$$cost_{km}(c, C) = \sum_{1 \leq k \leq K} \sum_{o \in C_K} d(o, c_k)^2$$

avec $c = \{c_1, \ldots, c_n\}$ les centres des classes et $C = \{C_1, \ldots, C_n\}$ les classes.
Un algorithme de classification doit donc trouver les paramètres c et C qui minimisent *la fonction de coût $cost_{km}$.*

L'espace de recherche A est souvent contenu dans $\mathbb{Z}^n$ (on parle alors d'optimisation combinatoire) ou dans $\mathbb{R}^n$. Un élément de A est alors un vecteur $\vec{x}$ et l'espace de recherche est délimité par des contraintes de la forme :

$$\begin{aligned} g_i(\vec{x}) &\leq 0, \quad 1 \leq i \leq m \\ h_j(\vec{x}) &= 0, \quad 1 \leq j \leq p \end{aligned}$$

Il existe de nombreuses méthodes pour résoudre un problème d'optimisation. Ces méthodes peuvent être divisées en deux catégories : les méthodes classiques, basées sur des approches mathématiques (section 2.2.1) et les méthodes stochastiques introduisant du hasard, tout en guidant la recherche de solutions (section 2.2.2).

2.2.1 Méthodes classiques

L'optimisation est un problème très ancien qui a donné lieu à de nombreuses méthodes de calcul. Les premières approches sont issues des mathématiques, mais celles-ci font souvent des hypothèses fortes sur la fonction à optimiser, comme par exemple des contrainte de convexité, de différentiabilité ou de continuité [Céa, 1971 ; Culioli, 1994 ; Charon *et al.*, 1996], et sont donc peu utilisables pour des problèmes concrets. Parmi les méthodes les plus connues, on retrouve les méthodes de descente de gradient (utilisées par exemple pour l'apprentissage de réseaux de neurones artificiels), la programmation linéaire ou la programmation dynamique. Les méthodes de *hill-climbing* sont des méthodes itératives qui, à partir d'une solution, trouve une solution meilleure selon une méthode d'exploration donnée. La procédure est alors répétée jusqu'à atteindre un maximum. Les méthodes *gloutonnes* sont des méthodes itératives qui consistent à compléter une solution partielle sans remettre en cause les choix effectués aux étapes précédentes.

Exemple :

L'optimisation de la fonction d'évaluation $cost_{km}$ peut se faire par un algorithme de hill-climbing. *Il est en effet prouvé que :*
- *les centres des classes c_k qui minimisent $cost_{km}(c, C)$ avec C fixé sont les isobarycentres des ensembles C_k ;*
- *les classes C_k qui minimisent $cost_{km}(c, C)$ avec c fixé sont définies par $o \in C_k \Longleftrightarrow d(o, c_{k'}) \leq d(o, c_k), \forall k'$.*

L'optimisation peut donc être réalisée par deux optimisations partielles répétées successivement en fixant l'un des paramètres et en optimisant l'autre. C'est le principe de l'algorithme K-means.

De plus, certains problèmes sont particulièrement difficiles à modéliser sous une forme qui permet l'application de ces méthodes ; celles-ci sont en effet trop coûteuses en temps de calcul ou sont trop souvent bloquées dans un extremum local. L'introduction (contrôlée) de hasard dans les algorithmes de recherche peut permettre de trouver une bonne solution dans un temps raisonnable.

2.2.2 Méthodes stochastiques

Nous présenterons ici tout d'abord la méthode du recuit simulé qui fait évoluer aléatoirement une solution (section 2.2.2.1). Nous présenterons ensuite les méthodes considérant plusieurs solutions simultanément, en particulier les algorithmes génétiques (section 2.2.2.2), puis plus brièvement l'optimisation par essaims particulaires (section 2.2.2.3).

2.2.2.1 Recuit simulé

Le *recuit simulé* est une méthode stochastique itérative inspirée d'un processus utilisé en métallurgie [Kirkpatrick *et al.*, 1983]. La fonction à minimiser représente l'énergie E d'un système et les solutions potentielles sont les différents états du système. Un paramètre T représentant la température est introduit : lorsque la température est élevée, les variations sont plus fréquentes que lorsque la température est basse.

Au début de l'algorithme une solution aléatoire est générée et la température initiale $T = T_0$ est choisie.

À chaque itération de l'algorithme, la solution courante est perturbée. La modification du système entraîne une variation d'énergie ΔE (c'est-à-dire une variation sur la fonction à optimiser). Si la variation est négative (la nouvelle solution est meilleure), elle est acceptée. Si $\Delta E > 0$, la nouvelle solution est acceptée avec une probabilité $e^{-\frac{\Delta E}{T}}$.

La température du système baisse petit à petit (par exemple en faisant $T_{i+1} = \alpha \times T_i$, avec $\alpha < 1$). Ainsi, lorsque la température est élevée (au début de l'algorithme) une solution plus mauvaise que la solution courante est souvent acceptée (ce qui permet de sortir des minimaux locaux). En revanche, lorsque la température est basse une mauvaise solution est rarement acceptée (si la température vaut 0, le recuit simulé devient un algorithme de *hill-climbing*).

L'algorithme s'arrête quand une température minimale est atteinte ou après un certain nombre d'itérations.

2.2.2.2 Algorithmes évolutionnaires

Les *algorithmes évolutionnaires* sont des algorithmes d'exploration fondés sur les mécanismes de la sélection naturelle et de la génétique [Holland, 1975 ; Goldberg, 1989]. Dans une population, les individus les plus adaptés à leur environnement survivent et peuvent se reproduire. Les caractéristiques des individus sont encodées dans leurs chromosomes. Les enfants héritent des caractéristiques de leurs parents. De génération en génération, les individus seront alors de plus en plus adaptés à leur environnement.

Ce mécanisme d'évolution peut être adapté à des systèmes artificiels. L'*environnement* est alors le problème à résoudre et les *individus* des solutions potentielles à ce problème. Les paramètres à optimiser, c'est-à-dire l'encodage des solutions, sont les *gènes* d'un individu, regroupés dans un *chromosome*, également appelé *génotype* de l'individu. Il existe différents *allèles* pour chaque gène, qui sont les différentes valeurs que peut prendre le gène. On appelle *phénotype* le résultat de l'interaction entre le matériel génétique et l'environnement, c'est-à-dire la solution encodée dans l'individu. La fonction d'évaluation (ang. *fitness function*) permet de juger si un individu est adapté à son environnement, c'est-à-dire s'il représente une bonne solution au problème à résoudre. La fonction d'évaluation dépend donc totalement du problème à résoudre, ce qui peut poser bien des problèmes pour sa définition.

Les algorithmes évolutionnaires les plus connues sont les *algorithmes génétiques*, les *stratégies évolutionnaires* et la *programmation évolutionnaire*. Ces approches diffèrent principalement par les méthodes de recombinaison des chromosome [Bäck, 1996]. Nous ne nous intéresserons ici qu'aux algorithmes génétiques et nous ne détaillerons pas les deux autres approches dans ce mémoire.

Dans les premiers algorithmes génétiques, les chromosomes étaient des chaînes de bits, chaque gène étant un bit, c'est-à-dire un paramètre binaire. Il est cependant possible d'utiliser les algorithmes génétiques avec d'autres types de paramètres, comme par exemple des entiers ou des réels. Il est aussi possible d'optimiser des structures plus complexes comme des arbres représentant des programmes. On parle alors de *programmation génétique* [Koza, 1992].

Le fonctionnement des algorithmes génétiques repose sur une heuristique simple : les meilleures solutions se trouvent dans une zone de l'espace de recherche contenant une grande proportion de bonnes solutions [Renders, 1995]. Donc en utilisant plusieurs individus, chacun étant une solution potentielle, et en combinant entre eux ceux qui sont les plus adaptés au problème à résoudre, il est possible de se rapprocher de la solution optimale.

De nouveaux individus sont créés à chaque génération au moyen d'opérateurs génétiques (FIG. 2.1). Les principaux opérateurs sont le *croisement*, c'est-à-dire une recombinaison du matériel génétique provenant de deux individus ou plus (reproduction sexuée) et la *mutation* qui est une perturbation aléatoire du matériel génétique (reproduction asexuée). Les opérateurs génétiques peuvent être de types très variés et dépendent souvent de l'encodage des solutions.

Les croisements ont pour but d'exploiter l'espace des solutions déjà exploré. Un opérateur de croisement va combiner le matériel génétique de plusieurs individus, sans apporter de nouveaux allèles. Dans le cas de chromosomes représentés sous la forme de vecteurs (chaînes de bits, par exemple), plusieurs types de croisement ont été proposés, les plus connus étant le croisement en un point et le croisement uniforme. Dans le cas du croisement en un point (FIG. 2.1(a)), on choisit aléatoirement un point de coupure qui sera appliqué à chacun des chromosomes des parents. Les chromosomes résultant de l'opération sont alors composés d'une partie du chromosome de chacun des parents. Dans ce type de croisement, des gènes éloignés dans le chromosome ont plus de chance d'être séparés par le croisement que des gènes proches, ce qui peut être problématique lorsque l'ordre des gènes a été choisi arbitrairement. C'est pourquoi un croisement uniforme (FIG. 2.1(b)) est souvent préférable. Un croisement uniforme consiste à choisir aléatoirement, pour chaque gène d'un descendant, de quel parent il va être hérité.

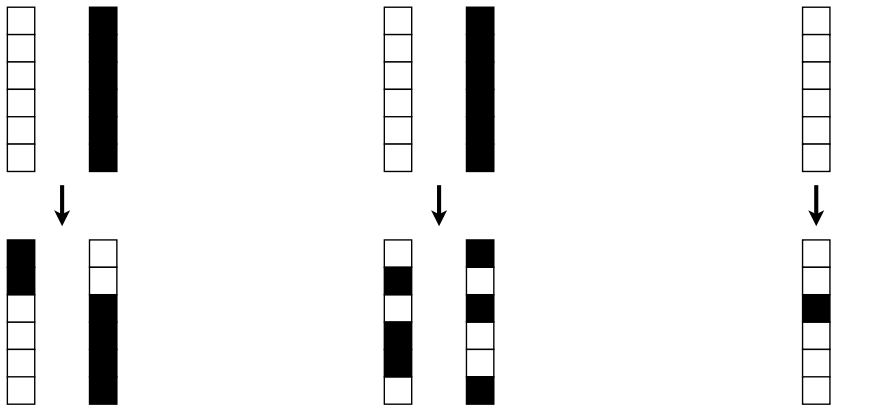
Les mutations sont indispensables pour apporter une diversité et explorer de nouvelles possibilités. En effet, si le nombre d'allèles possibles pour un gène est très grand, il est probable que la valeur optimale ne soit pas apparue dès la première génération. Il est donc nécessaire d'apporter de nouvelles valeurs de temps en temps. En revanche, un taux de mutation trop important risque d'altérer les individus et d'empêcher de converger vers la solution. Dans le cas de chromosomes représentés sous la forme de vecteurs, on va parcourir l'ensemble des gènes. Chacun d'entre eux aura une faible probabilité d'être muté, c'est-à-dire de prendre une nouvelle valeur, par une inversion de bits dans le cas d'une chaîne de bits, ou en prenant une valeur générée aléatoirement (FIG. 2.1(c)).

Un algorithme génétique classique se déroule de la manière suivante : une population d'individus $P = \{I^1, \ldots, I^p\}$ est générée aléatoirement (tous les individus sont différents). À chaque génération, les individus sont évalués selon une fonction d'évaluation (de *fitness*) f. Une nouvelle population est alors générée à partir de la génération courante, en choisissant les meilleurs individus puis en les recombinant entre eux, par des croisements, et en les altérant, par des mutations. Le processus global se répète jusqu'à ce qu'une condition d'arrêt soit atteinte, par exemple après un certain nombre de générations ou quand une solution acceptable est atteinte (Algorithme 2.1)

Exemple :

Un algorithme génétique peut être utilisé pour optimiser la fonction de coût $cost_{km}$. Cette fonction de coût dépend à la fois des centres des classes et de l'affectation des objets aux classes. Étant donné que l'on peut optimiser l'un des deux critères en fixant l'autre, il est possible de redéfinir la fonction de coût $cost_{km}$ en utilisant uniquement l'un des deux paramètres, puisque l'autre peut en être déduit. Par exemple, $cost_{km}$ peut être définie en fonction des centres uniquement par :

$$cost_{km}(c) = \sum_{1 \leq k \leq K} \sum_{o \in C_K} d(o, c_k)$$



(a) *Croisement en un point* (b) *Croisement uniforme* (c) *mutation*

Fig. 2.1 : *Opérateurs génétiques classiques*

Algorithme 2.1 Algorithme génétique

```
1 pour i ← 1 à p faire
2     I^i ← Initialisation ()
3 tant que la condition d'arrêt n'est pas vérifiée faire
4     pour i ← 1 à p faire
5         fitness (I^i) ← f (I^i)
6     pour i ← 1 à p faire
7         Parent_1 ← Slection (P)
8         Parent_2 ← Slection (P)
9         I^i ← Mutation (Croisement (Parent_1, Parent_2))
```

avec $o \in C_k$ si et seulement si $d(o, c_{k'}) \leq d(o, c_k), \forall k'$.
Un algorithme d'optimisation peut donc se limiter à découvrir l'ensemble des centres des classes. Dans un algorithme génétique, un individu représentera l'ensemble des centres des classes, un gène représentant la valeur sur un des attributs pour un des noyaux. Le génotype d'un individu est l'ensemble des centre des classes, son phénotype est la classification produite à partir des centres. Les individus sont évalués selon la fonction de coût $cost_{km}$.

La sélection des individus se fait selon la fonction d'évaluation : plus la qualité d'un individu est élevée, plus il a de chance d'être sélectionné pour se reproduire. Il existe pour cela plusieurs stratégie pour sélectionner les individus [Blickle et Thiele, 1995].

La méthode de *sélection proportionnelle* (ang. *roulette wheel selection*) consiste à choisir un individu avec une probabilité proportionnelle (resp. inversement proportionnelle) à sa qualité pour une fonction d'évaluation à maximiser (resp. pour une fonction d'évaluation à minimiser).

La méthode de *sélection par tournoi* (ang. *tournament selection*) consiste à choisir aléatoirement un sous-ensemble d'individus. Un individu est choisi dans ce sous-ensemble (soit le meilleur, soit avec une sélection proportionnelle). Plus la taille du sous-ensemble est petit, plus un individu de faible qualité a de chance d'être sélectionné.

Une stratégie *élitiste* consiste à conserver intacts les meilleurs individus pour la génération suivante, sans les croiser ni les muter, le reste de la population étant généré de manière classique.

Une autre méthode évolutionnaire pour optimiser des vecteurs réels, proche des algorithmes génétiques, est l'*évolution différentielle* [Storn et Price, 1995]. De manière similaire, on dispose d'une population d'individus (initialisés aléatoirement) qui sont des solutions potentielles au problème d'optimisation.

L'évolution des individus est en revanche différente. À chaque génération, pour chaque individu I^i, un nouvel individu I^i_{tmp} est créé de la manière suivante : $I^i_{tmp} = I^{r_1} + F \times (I^{r_2} - I^{r_3})$, avec r_1, r_2 et r_3, trois entiers distincts différents de i choisis aléatoirement, et $0 < F \leq 1$. Une opération de croisement est alors réalisée entre I^i et I^i_{tmp} pour produire un individu qui est alors évalué et qui remplace I^i s'il est meilleur.

2.2.2.3 Optimisation par essaims particulaires

L'*optimisation par essaims particulaires* (ang. *Particle Swarm Optimization*) est une méthode d'optimisation stochastique inspirée de la biologie qui utilise une autre métaphore que les algorithmes génétiques : la population est un essaim qui évolue dans l'espace et cherche de la nourriture, la fonction d'évaluation indiquant la quantité de nourriture présente en un point de l'espace [Kennedy et Eberhart, 1995].

Les individus se déplacent. À chaque itération de l'algorithme, ils changent de direction. La nouvelle direction est calculée en fonction de trois aspects. Les individus ont chacun une inertie : ils continuent à se déplacer dans la direction qu'ils prenaient à l'itération précédente. Les individus ont tous une mémoire : ils se souviennent de la meilleure solution qu'ils ont découverte et cherchent à y retourner. Les individus communiquent et connaissent le meilleur endroit visité par leurs congénères (soit l'optimum sur toute la population, soit l'optimum sur un voisinage). Les individus tendent donc aussi à se diriger vers le meilleur endroit atteint par l'un de leurs congénères.

On calcule alors la *vélocité* de chacun des individus en fonction de son inertie, de sa mémoire et de l'influence sociale. La vélocité d'un individu I^i à l'itération $t > 0$ est calculée de la manière suivante :

$$\begin{aligned} v_i(t) = v_i(t-1) &+ 2 \times \text{Random}() \times (m_i(t-1) - a_i(t-1)) \\ &+ 2 \times \text{Random}() \times (g_i(t-1) - a_i(t-1)) \end{aligned}$$

où $a_i(t)$ représente les coordonnées de l'individu I^i à l'itération t, $m_i(t)$ représente les coordonnées de la meilleure solution visitée par l'individu I^i, $g_i(t)$ représente les coordonnées de la meilleure solution découverte par la population.

La vélocité sert à modifier les coordonnées d'un individu de la manière suivante :

$$a_i(t) = a_i(t-1) + v_i(t)$$

2.2.3 Synthèse

Les méthodes évolutionnaires sont des méthodes d'optimisation efficaces et robustes qui ont été appliquées avec succès à de nombreuses de diverses applications. Les performances de ces méthodes, comparativement aux approches plus classiques, proviennent, d'une part, de l'introduction de hasard dans la recherche qui permet de s'échapper des optimum locaux, et d'autre part, du fait qu'une population de solution est considérée simultanément, ce qui permet une exploration plus large de l'espace de recherche.

2.3 Modèles d'évolution artificielle

Plusieurs théories concernant l'évolution des espèces ont été développées, la plus célèbre étant la théorie de Charles Darwin. L'une des différences majeures entre les théories que nous allons présenter est l'importance qu'elles accordent à l'influence de l'évolution des individus au cours de leur vie sur l'évolution de l'espèce.

Certaines de ces théories ont été infirmées par les avancées en biologie, mais peuvent cependant être appliquées à des systèmes artificiels.

Nous présenterons tout d'abord la théorie darwinienne de l'évolution, qui ne tient pas compte de l'évolution au cours de la vie (section 2.3.1), puis la théorie lamarckienne qui fait l'hypothèse d'une transmission des caractères acquis à la descendance (section 2.3.2), et enfin la théorie baldwinienne qui tient compte de l'évolution au cours de la vie sans transmission des caractères acquis (section 2.3.3).

2.3.1 Évolution darwinienne

La théorie de l'évolution darwinienne [Darwin, 1859] est la théorie de l'évolution communément admise et validée par la découverte de la génétique. L'évolution darwinienne suit les trois principes suivant :

- les individus sont différents les uns des autres ;
- les enfants héritent des caractères des parents ;
- plus un individu est (génétiquement) adapté à son environnement, plus sa progéniture sera importante.

Dans cette approche, les caractères d'un individu n'évoluent pas au cours de sa vie. Les caractères qu'un parent transmet à ses descendants sont ceux qu'il a lui même hérités de ses propres parents.

Cette théorie peut être traduite dans les systèmes artificiels. À chaque génération de l'algorithme, le phénotype de chaque individu est construit en fonction de son génotype. Les phénotypes des individus sont alors évalués selon la fonction à optimiser. Les meilleurs individus sont sélectionnés pour être recombinés et produire la population de la génération suivante.

Le modèle d'évolution darwinien est le plus employé dans les algorithmes génétiques.

Exemple :

> *Si un algorithme darwinien est utilisé pour l'optimisation de la fonction $cost_{km}$, un individu est évalué selon la qualité de la classification produite à partir des centres des classes qu'il encode.*
> *Les opérateurs de mutations et de croisements s'appliquent directement sur les centres des classes encodés dans les individus.*

2.3.2 Évolution lamarckienne

La théorie de l'évolution proposée par Larmack est basée sur l'idée que les individus cherchent à évoluer au cours de leur vie et qu'ils transmettent leurs caractères acquis à leur descendance [Lamarck, 1809].

L'évolution lamarckienne peut donc être résumée aux préceptes suivant :

- les individus sont différents les uns des autres ;
- les individus évoluent pour s'adapter à leur environnement ;
- les enfants héritent des caractères acquis par leurs parents ;
- plus un individu s'est adapté à son environnement, plus sa progéniture sera importante.

Cette théorie, inexacte dans la réalité, peut cependant être adaptée à des systèmes artificiels. Cela se traduit dans les algorithmes génétiques de manière quasiment identique à l'apprentissage darwinien. À chaque génération, le phénotype de chaque individu est construit en fonction de son génotype. Les phénotypes et génotypes des individus sont alors modifiés par une recherche locale, puis évalués selon la fonction à optimiser. Les meilleurs individus sont sélectionnés pour être recombinés et produire la population de la génération suivante à partir de leur nouveau matériel génétique.

Ce modèle d'évolution, hybride entre approche génétique et approche classique, a été utilisé avec succès dans de nombreuses application [Grefenstette, 1991 ; Paredis, 1996 ; Ross, 1999].

Exemple :

> *Pour définir un algorithme lamarckien d'optimisation de la fonction $cost_{km}$, l'algorithme K-means peut être utilisé comme méthode de recherche locale. Pour chaque individu, une ou plusieurs itérations de l'algorithme sont effectuées à partir des centres encodés dans l'individu. Les nouveaux centres obtenus sont utilisés pour redéfinir le génotype de l'individu. Celui-ci est alors évalué selon la qualité de la classification produite après la recherche locale et les opérations de croisement et de mutation se font également sur le nouveau matériel génétique.*

2.3.3 Évolution baldwinienne

Dans une approche baldwinienne de l'évolution, la sélection des individus ne se fait pas uniquement selon des caractéristiques innées des individus, mais aussi en fonction de leurs expériences [Baldwin, 1896]. L'évolution de l'espèce est donc dirigée par l'expérience, la sélection naturelle dépendant de l'aptitude de l'individu à apprendre et à s'adapter à son milieu. On parle de *plasticité phénotypique*, c'est-à-dire de la capacité des individus à adapter leur phénotype à leur environnement. L'influence des caractères acquis sur l'évolution d'une espèce est appelé *effet Baldwin* (ang. *Baldwin effect*). Les enfants héritent des caractéristiques innées des parents, mais pas de leurs caractéristiques acquises. Ce principe permet de réconcilier les théories de l'évolution darwinienne et lamarckienne. Un système d'évolution peut donc être qualifié de baldwinien si :

- les individus sont différents les uns des autres ;
- les individus évoluent pour s'adapter à leur environnement ;
- les enfants héritent des caractères innées par leurs parents ;
- plus un individu s'est adapté à son environnement, plus sa progéniture sera importante.

Cela se traduit dans les algorithmes génétiques de manière presque identique à l'apprentissage lamarckien. À chaque génération de l'algorithme, le phénotype de chaque individu est construit en fonction de son génotype. Les phénotypes des individus sont alors modifiés par une recherche locale, puis évalués selon la fonction à optimiser. Les meilleurs individus sont sélectionnés pour être recombinés et produire la population de la génération suivante en utilisant le matériel génétique d'origine des individus. Ce modèle d'évolution a souvent été comparé à l'approche lamarckienne [Whitley *et al.*, 1994 ; Ku et Mak, 1997].

Exemple :

> *La définition d'un algorithme baldwinien pour l'optimisation de la fonction $cost_{km}$ est presque identique à celle d'un algorithme lamarckien. L'algorithme K-means peut être utilisé comme méthode de recherche locale. Pour chaque individu, une ou plusieurs itérations de l'algorithme sont effectuées à partir des centres encodés dans l'individu. Celui-ci est alors évalué selon la qualité de la classification produite après la recherche locale. Cependant, les nouveaux centres obtenus ne sont pas utilisés pour redéfinir le génotype de l'individu. Les opérations de croisement et de mutation se font donc sur le matériel génétique originel.*

2.3.4 Comparaison des trois approches

Après avoir présenté les approches darwinienne, lamarckienne et baldwinienne, nous allons comparer les intérêts de chacune. Ses trois approches peuvent être comparées selon leur processus de sélection et d'évaluation des individus et selon leur efficacité.

Les modèles d'évolution lamarckien et baldwinien peuvent être considérés comme des méthodes hybrides d'optimisation, combinant approche génétique et approche classique [Whitley, 1995 ; Goldberg et Voessner, 1999].

L'évolution lamarckienne permet une convergence plus rapide que l'évolution darwinienne, mais le risque d'être bloqué dans un extremum local est plus important.

Dans l'évolution baldwinienne un individu qui peut potentiellement amener à une bonne solution a plus de chance d'être choisi, mais cette approche évite une convergence précipitée vers un extremum local. L'évaluation des individus après une recherche locale a pour effet un lissage de la fonction d'évaluation ce qui rend plus facile la recherche de bonnes solutions.

Il n'est pas toujours possible, après une recherche locale, d'encoder la solution trouvée sous forme de chromosome. Dans ce cas, il est possible d'utiliser une approche baldwinienne, mais pas une approche lamarckienne [Turney, 1996].

Exemple :

Sur la figure 2.2 l'évaluation de différents individus est représentée sur la courbe d'une fonction d'évaluation (à minimiser). I^1 et I^2 indiquent l'évaluation du phénotype de deux individus. $ls\left(I^1\right)$ et $ls\left(I^2\right)$ indiquent l'évaluation du phénotype des deux individus après une recherche locale.
Dans une approche darwinienne, l'individu I^1 aura plus de chance d'être sélectionné pour se reproduire que I^2, car I^1 est mieux adapté à son environnement, sa valeur sur la fonction d'évaluation étant plus basse.
Dans une approche lamarckienne, l'individu $ls\left(I^2\right)$ aura plus de chance d'être sélectionné pour se reproduire que $ls\left(I^1\right)$, car $ls\left(I^2\right)$ est mieux adapté à son environnement, sa valeur sur la fonction d'évaluation étant plus basse.
Dans une approche baldwinienne, l'individu I^2 aura plus de chance d'être sélectionné pour se reproduire que I^1, car $ls\left(I^2\right)$ est mieux adapté à son environnement, sa valeur sur la fonction d'évaluation étant plus basse que $ls\left(I^1\right)$.

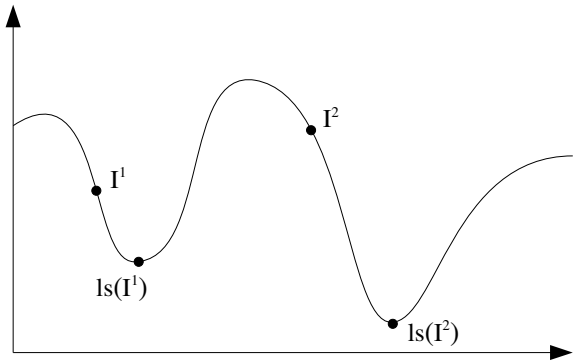


Fig. 2.2 : *Sélection des individus dans les modèles d'évolution darwinien, lamarckien et baldwinien*

Du point de vue de l'efficacité de ces trois approches, différentes études ont été menées afin de les comparer, comme par exemple [Julstrom, 1999]. L'approche darwinienne est généralement moins efficace que les deux autres. Les approches lamarckienne et baldwinienne sont sensiblement équivalentes bien qu'il existe des différences selon les applications.

2.4 Coévolution génétique

Plusieurs méthodes consistant à utiliser plusieurs populations ont été proposées dans la littérature. Les plus connues sont les méthodes d'îlots utilisées pour les algorithmes génétiques distribués [Tanese, 1989 ; Whitley et Starkweather, 1990]. Une population est créée sur chaque site de calcul et les populations évoluent indépendamment. Ces populations cherchent à atteindre le même objectif. De temps en temps surviennent des migrations d'individus : les meilleurs individus des populations sont échangés entre les îlots pour apporter une plus grande diversité génétique. Ce type de méthodes a pour objectif de minimiser le nombre de communication entre les sites de calculs.

La coévolution est une approche très différente des méthodes par îlots qui consiste à faire évoluer plusieurs espèces différentes, c'est-à-dire plusieurs populations indépendantes du point de vue de la reproduction (les individus ne se reproduisent que dans leur propre population), mais en interaction les unes avec les autres, l'évaluation d'un individu d'une population se faisant en fonction d'individus représentatifs des autres populations.

Dans cette section, nous présenterons d'abord les principales caractéristiques d'un algorithme coévolutionnaire (section 2.4.1). Nous présenterons ensuite la coévolution compétitive (section 2.4.2) et la coévolution coopérative (section 2.4.3). Nous discuterons ensuite des difficultés liées à l'utilisation de la coévolution et les solutions qui ont été proposées (section 2.4.4).

2.4.1 Caractéristiques des algorithmes coévolutionnaires

Il existe plusieurs types d'algorithmes de coévolution [Mayer, 1998]. Les principales caractéristiques sont les suivantes :

- la relation entre les populations ;
- le temps de vie des populations ;
- la méthode d'évaluation des individus ;
- la symétrie des populations.

La relation entre les populations peut prendre deux formes, la compétition ou la coopération, selon que l'augmentation de la qualité d'une population dégrade ou améliore celle des autres. Dans le premier cas, les populations sont en compétition (pour l'accès à une ressource, par exemple), dans le second les populations collaborent pour atteindre un but commun.

Le temps de vie peut varier d'une population à l'autre, c'est-à-dire qu'une population peut évoluer pendant plusieurs générations alors que les autres restent inchangées, puis une autre évolue et ainsi de suite.

L'évaluation des individus peut se faire face à un seul individu de chacune des autres populations (environnement simple), face à un ensemble d'individus (environnement multiple) ou face à tous les individus (environnement complet).

Le temps de calcul dans un environnement complet est souvent rédhibitoire. Dans un environnement simple, un seul individu représentatif de la population est choisi alors que dans un environnement multiples, un échantillon de p individus est sélectionné dans chaque population. Les *individus représentatifs* d'une population peuvent être choisis en prenant les p meilleurs individus de la population, en sélectionnant p individus aléatoirement selon leur fitness (*roulette wheel selection*), ou encore, comme proposé dans [Rosin et Belew, 1997], en fonction de la qualité, mais aussi de la diversité des individus.

Dans le cas d'un environnement multiple ou d'un environnement complet, l'agrégation de la qualité peut se faire soit en prenant le meilleur score, soit en faisant une moyenne.

On peut enfin distinguer les approches symétrique et asymétrique. Dans une approche symétrique, les individus de toutes les populations sont identiques (même représentation dans les chromosomes et même expression du matériel génétique), par exemple dans le cas d'une lutte pour des ressources limitées. Dans une approche asymétrique, les individus sont différents d'une population à l'autre, par exemple dans un système prédateur/proie.

2.4.2 Coévolution compétitive

En coévolution compétitive, les populations sont en opposition [Floreano et Nolfi, 1997 ; Haith *et al.*, 1999]. Généralement, deux populations sont utilisées dans une approche asymétrique. La coévolution compétitive s'apparente alors à un contexte prédateur/proie ou hôte/parasite. La population de proies ou de parasites représente les problèmes potentiels et la population de prédateurs ou d'hôtes, les solutions. L'évolution prend alors la forme d'une course à l'armement. Au cours de l'évolution, les problèmes deviennent de plus en plus difficiles, car plus le problème est difficile, moins les solutions ont de chance de les résoudre. Mais dans un même temps, les solutions évoluent également, jusqu'à résoudre les problèmes les plus complexes.

Cependant, il y a un risque d'apprentissage circulaire (définition 2.2). Dans [Rosin et Belew, 1997], les auteurs proposent de mettre en place un *Hall of fame*, c'est-à-dire de conserver les meilleurs individus de chaque génération et d'évaluer les individus face à tous les ancêtres ou un échantillon de ces ancêtres.

Définition 2.2 (Apprentissage circulaire, dans le cas de deux espèces)
On note :

- $P(g)$, *la population constituée des individus d'une espèce* P *à la génération* g ;
- $f(P(g), P'(g'))$, *la qualité du meilleur individu de la population* $P(g)$ *face aux individus de la population* $P'(g')$;
- $P(g) > P'(g')$, *si* $f(P(g), P'(g')) > f(P'(g'), P(g))$.

Soit deux espèces P *et* P'. *Un apprentissage circulaire risque d'apparaître si* $P(g) > P'(g)$ *et* $P(g) < P'(g_0)$, *avec* $g_0 < g$, *c'est-à-dire que des individus des générations anciennes d'une population* P' *sont plus performants face aux individus d'une population* P *de la génération courante. Il se peut dans ce cas qu'il y ait un retour en arrière dans l'évolution plutôt qu'une recherche de solutions nouvelles.*

2.4.3 Coévolution coopérative

Pour résoudre des problèmes complexes d'optimisation, il peut être judicieux de décomposer le problème initial en plusieurs sous-problèmes. Dans un algorithme génétique par *approche parisienne* [Collet *et al.*, 2000 ; Dunn *et al.*, 2005], une seule population est utilisée, mais chaque individu ne représente qu'une partie de la solution. La solution complète est construite en combinant plusieurs individus.

La coévolution coopérative, en revanche, consiste à utiliser plusieurs population et à construire une solution complète en utilisant un individu par population. Ce type d'évolution est également appelé coévolution symbiotique, la symbiose correspondant, en biologie, à une association entre différents organismes vivants (comme par exemple, les lichens qui sont constitués d'une symbiose entre un champignon et une algue). Parmi les premiers travaux sur la coévolution coopérative, on peut citer [Potter et De Jong, 1994 ; Potter *et al.*, 1995 ; Potter et De Jong, 2000]. Les auteurs proposent ainsi une approche utilisant plusieurs espèces pour résoudre des problèmes pouvant se diviser en plusieurs sous-problèmes. Les auteurs ont montré que le simple fait de faire évoluer les populations les unes en fonction des autres permet à chaque espèce d'occuper une niche spécifique. La spécialisation de chaque population se fait ainsi sans contrainte initiale mais en fonction des spécialisations des autres populations.

De plus, ils ont observé que lorsque le nombre de populations n'est pas pas assez élevé, des sous-populations se créent au sein des populations, des individus ayant des phénotypes très différents se développent dans une même population, une espèce pouvant alors occuper plusieurs niches. Dans [Potter et De Jong, 1995], ils appliquent leur méthode de coévolution coopérative à l'apprentissage de réseaux de neurones en cascade. Chaque population évolue afin d'optimiser un neurone du réseau. Les individus sont évalués en collaboration avec les meilleurs individus des autres populations. Le nombre d'espèces est dynamique : une population est rajoutée si l'apprentissage stagne, mais le rôle de chaque espèce est déterminé à l'avance par sa position dans le réseau de neurone.

Dans le cas d'une approche symétrique, si aucune contrainte sur la spécialisation des populations n'est explicitée, celle-ci se fait naturellement, chaque population trouvant une niche différente des autres. Dans [Potter *et al.*, 2001], les auteurs montrent que ce n'est pas la difficulté de la tâche, mais la nécessité d'avoir un nombre important de compétences différentes qui induit une spécialisation des individus.

L'utilité d'une telle approche est montrée dans [Yong et Miikkulainen, 2001]. Les auteurs montrent en effet que des agents indépendants sont plus aptes à résoudre la tâche demandée (un problème prédateurs/proies) que des agents contrôlés par un système central, c'est-à-dire qu'il est plus facile de résoudre un problème en le fractionnant en sous-problèmes plus simples.

Exemple :

> *La coévolution coopérative peut être utilisée pour optimiser la fonction d'évaluation $cost_{km}$. Si l'on cherche à classifier les données en K classes, il est possible de décomposer le problème en K sous-problèmes symétriques qui sont l'identification des K centres des classes.*
> *Dans une approche évolutionnaire classique, un ensemble de centres est encodé dans chaque individu. Dans une approche coévolutionnaire, chaque individu encode un centre de classe. K populations (une par classe) sont utilisées. Le phénotype d'un individu d'une population donnée est la classification obtenue en combinant le centre encodé dans l'individu et les centres encodés dans les individus représentatifs des autres populations.*

2.4.4 Difficulté liées à la coévolution

La décomposition d'un problème en plusieurs population, que ce soit par coévolution compétitive ou coopérative entraîne cependant quelques difficultés, en particulier l'effet *Red Queen* qui peut bloquer l'apprentissage (section 2.4.4.1), et la définition des individus représentatifs à la première génération (section 2.4.4.2).

2.4.4.1 Effet *Red Queen*

En coévolution, contrairement aux approches évolutionnaires classiques, l'environnement d'une population est défini en fonction des autres populations. Or toutes les populations évoluent, donc l'environnement de chacune évolue également. Ainsi, si une population s'est adaptée aux autres au cours d'une génération, elle ne le sera plus nécessairement à la génération suivante. Ce phénomène est appelé effet *Red Queen*, en référence au personnage d'*Alice au pays des merveille* qui court très vite et très longtemps avec Alice, jusqu'à ce que celle-ci en soit exténuée, pour finalement rester au même endroit [Carroll, 1871].

Ceci pose un premier problème qui est l'observation de l'évolution dans le cas où les espèces sont évaluées par des critères différents (en particulier dans la coévolution compétitive, où la qualité d'une espèce fait baisser la qualité des autres espèces). Dans [Cliff et Miller, 1995], les auteurs proposent alors d'observer la qualité d'un individu par rapport à toutes les générations précédentes, ainsi que les distances génétiques entre les individus aux cours des générations, c'est-à-dire la différences entre les individus en terme de phénotype ou de génotype.

Ainsi, l'effet *Red Queen* peut détériorer l'apprentissage. Dans [Paredis, 1997] plusieurs expériences montre que l'effet *Red Queen* peut empêcher l'apprentissage dans le cas de la coévolution compétitive, chaque espèce se contentant de contrer les autres et restant alors bloqué dans un apprentissage circulaire au lieu de proposer des solutions optimales.

2.4.4.2 Initialisation des individus représentatifs

Lors de la première évaluation, il n'existe pas encore d'individus représentatifs. La fonction d'évaluation des individus d'une population n'est donc pas définie. La qualité des individus est alors souvent évaluée sur des *collaborations arbitraires* (c'est-à-dire sans utilisation de connaissance) [Krawiec et Bhanu, 2003 ; Ioro et Li, 2004]. On peut par exemple simplement faire collaborer le i-ième individu d'une population avec le i-ième individu de chacune des $K-1$ autres populations.

Dans une approche constructiviste de l'apprentissage, les connaissances sont construites sur ce que l'apprenant connaît déjà [Gréco et Piaget, 1959] : il semble en effet difficile d'apprendre des concepts complexes en partant de rien. Un processus d'apprentissage efficace consiste à confronter ce que l'apprenant sait déjà à de nouvelles informations.

Nous proposons donc une approche basée sur ce principe afin d'accélérer le processus d'optimisation. On peut en effet espérer de meilleures capacités d'apprentissage en utilisant une *méthode d'initialisation* des individus représentatifs, capable de trouver rapidement des individus correspondant à chacune des sous-tâches de l'algorithme coévolutionnaire. Une méthode d'initialisation est un algorithme d'apprentissage, basé sur un autre paradigme, comme par exemple un algorithme de *hill-climbing*, qui propose une solution au problème global. Cette solution est alors utilisée pour définir les individus représentatifs de la première génération. La méthode d'initialisation des individus représentatifs peut être une méthode globale, qui propose une solution globale au problème, ou un ensemble de méthodes qui proposent une solution à chaque sous-problème. Dans les deux cas, la solution obtenue est décomposée en plusieurs parties, correspondant chacune à un des sous-problèmes de l'algorithme coévolutionnaire : chacune de ces parties doit permettre de définir un individu représentatif pour une des populations. Ainsi, l'algorithme coévolutionnaire pourra se baser sur des connaissances existantes, et l'apprentissage consistera à améliorer la solution initiale.

La procédure qui permet de passer de la solution proposée par la méthode d'initialisation aux individus représentatifs dépend à la fois du problème à résoudre et de la forme que prend la solution initiale. On remarque cependant que l'environnement d'une population, qui définit la fonction d'évaluation de celle-ci, est basé uniquement sur le phénotype des individus. Il n'est donc pas nécessaire que la méthode d'initialisation découvre le génotype des individus représentatifs.

2.5 Conclusion

Les algorithmes évolutionnaires sont des méthodes d'optimisation efficaces et robustes. Il existe de nombreuses variantes qui permettent de s'adapter au problème à résoudre pour permettre une convergence plus rapide vers une bonne solution. Les approches lamarckienne et baldwinienne sont des méthodes hybrides qui combinent algorithme génétique classique et algorithme de recherche locale. La coévolution coopérative consiste en une décomposition d'un problème complexe en sous-problèmes simples permettant ainsi une résolution plus efficace.

De nombreuses méthodes de classification non supervisée consistent en l'optimisation d'une fonction d'évaluation. Les algorithmes évolutionnaires peuvent donc être utilisés comme méthode d'apprentissage pour obtenir des classifications pertinentes. Les algorithmes que nous proposerons dans la suite de ce mémoire seront basés sur des algorithmes évolutionnaires. Nous utiliserons en particulier les approches lamarckienne et baldwinienne ainsi que la coévolution coopérative.

Chapitre 3

Sélection et pondération d'attributs

3.1 Problèmes liés à la dimensionnalité des données

Pour que des observations soient représentatives d'un phénomène étudié, leur nombre doit augmenter de manière exponentielle par rapport à la dimensionnalité de l'espace dans lequel elles sont représentées [Bellman, 1961]. La *malédiction de la dimensionnalité* désigne les problèmes liés à cette augmentation du nombre de dimensions. Or, dans la perspective d'obtenir une classification plus précise, on cherche souvent à décrire les données de la manière la plus détaillée possible, les données étant alors représentées par de nombreux attributs. En conséquence, le nombre d'observations doit être important, ce qui n'est pas toujours réalisable. De plus, les attributs qui décrivent les objets à classifier ne sont donc pas toujours directement utilisables.

De fait, plusieurs problèmes peuvent apparaître lors de la classification, dès lors que les objets sont représentés par de nombreux attributs :

- manque de pertinence : un attribut non pertinent n'apporte aucune information permettant de discriminer les classes entre elles ;
- bruit : un attribut bruité porte des informations incorrectes ;
- corrélations : si des attributs sont corrélés entre eux, une même information est redondante et a donc plus de poids qu'une information portée par un attributs indépendant ;
- ordre de grandeur : une même mesure peut s'exprimer selon différentes unités, or le choix de l'unité peut avoir une influence majeure sur le résultat de la classification, alors que l'information portée est intrinsèquement la même ;
- coût : la saisie de données sur de nombreux attributs peut avoir un coût important (en termes financier ou en termes de temps), il est donc nécessaire de déterminer quels sont les attributs indispensables pour classifier les données.

Exemples :

Les données représentées sur la figure 3.1, sont composées de trois classes de 200 objets. Chaque objet est représenté par deux attributs. On voit aisément que l'attribut 1 n'est pas pertinent et n'apporte aucune information utile à la classification des données. Une classification n'utilisant que l'attribut 2 sera plus efficace.
Les données représentées sur la figure 3.2, sont composées de trois classes de 200 objets. Chaque objet est représenté par cinq attributs. On voit sur la figure 3.2(a) que l'attribut 2 permet de discriminer la classe 3 des deux autres classes, mais confond les classes 1 et 2 entre elles et que l'attribut 1 apporte l'information nécessaire pour discriminer la classe 1 de la classe 2. On voit sur les figures 3.2(b) et 3.2(c) que

les attributs 2, 3, 4 et 5 sont corrélés. Ils portent tous une information similaire. L'information portée par l'attribut 1 risque donc de ne pas être prise en compte par les algorithmes de classification.

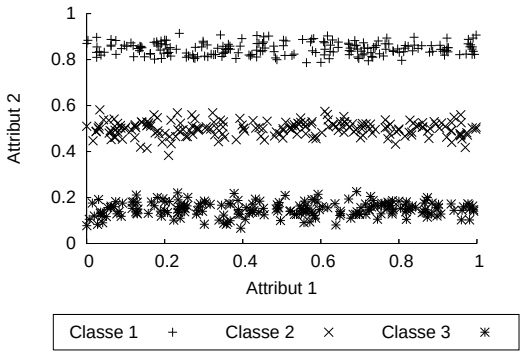
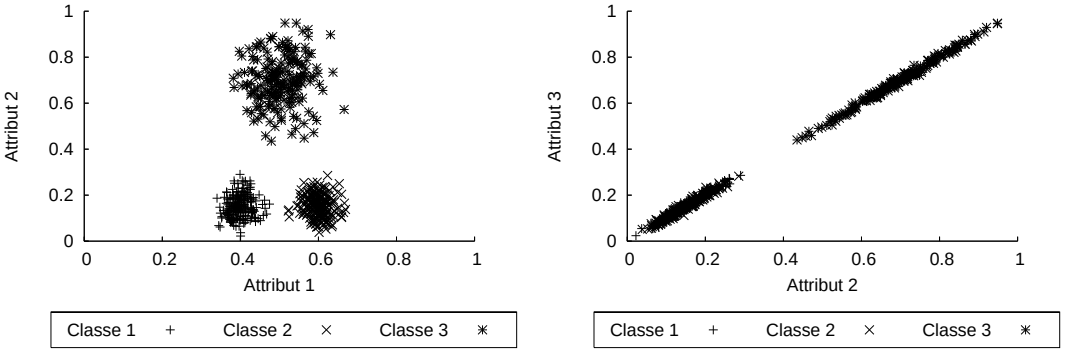
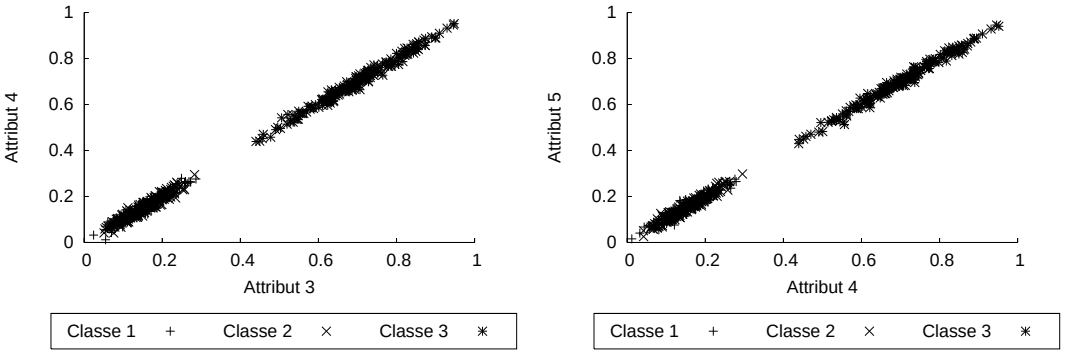


Fig. 3.1 : *Données avec un attribut non pertinent*



(a) *Projection des données sur les attributs 1 et 2*

(b) *Projection des données sur les attributs 2 et 3*



(c) *Projection des données sur les attributs 3 et 4*

(d) *Projection des données sur les attributs 4 et 5*

Fig. 3.2 : *Données avec des attributs corrélés*

Pour remédier à cela, différentes méthodes existent et sont classées en trois catégories selon [Motoda et Liu, 2003]. Elles consistent à adapter les données aux algorithmes de classification, en supprimant le bruit, en réajustant les différentes échelles et éventuellement en supprimant certains attributs. Elles peuvent être catégorisées suivant le traitement qu'elles appliquent aux données :

- l'*extraction d'attributs* consiste à transformer, en projetant les données sur de nouvelles dimensions, l'ensemble d'attributs de départ en un nouvel ensemble d'attributs, généralement plus petit, qui porte la même information ;
- la *construction d'attributs* consiste à créer de nouveaux attributs basés sur les relations entre les attributs existants ;
- la *sélection et la pondération d'attributs* consistent à chercher des poids binaires (sélection d'attributs) ou réels (pondération d'attributs) afin de faire varier l'importance relatives des attributs.

Les deux premières catégories de méthodes peuvent être regroupées sous le terme *transformation d'attributs*.

Les méthodes d'extraction d'attributs (telles que l'Analyse en Composantes Principales) sont généralement des méthodes statistiques applicables uniquement sur des données numériques. Les méthodes de construction d'attributs nécessitent des traitements spécifiques entre chaque type d'attributs pour en extraire des relations pertinentes et demandent donc un travail laborieux. De plus, les méthodes de transformation d'attributs combinent l'ensemble des attributs d'origine : elles conservent donc implicitement l'ensemble de ces attributs, qu'ils soient pertinents ou non. Ainsi, si le nombre d'attributs non pertinents est trop grand, les résultats de classification resteront décevants [Parsons *et al.*, 2004a ; Parsons *et al.*, 2004b]. Ces méthodes doivent donc impérativement être combinées avec des méthodes de sélection ou de pondération d'attributs pour être pleinement efficaces. Il est à noter que les méthodes de sélection/pondération d'attributs permettent de traiter aisément des attributs de types hétérogènes (attributs numériques, catégoriels, histogrammes, intervalles, ...).

Dans nos travaux, nous nous sommes ainsi intéressés à la sélection et à la pondération d'attributs. Dans ce chapitre, nous présenterons d'abord les caractéristiques qui distinguent les différentes méthodes de sélection/pondération d'attributs (section 3.2). Nous expliquerons ensuite comment un sous-ensemble d'attributs obtenu par un algorithme de sélection d'attributs et comment les poids obtenus par un algorithme de pondération d'attributs sont utilisés par un algorithme de classification (section 3.3). Différents algorithmes de sélection (section 3.4) et de pondération d'attributs (section 3.5) seront ensuite exposés. Enfin, nous détaillerons plusieurs critères d'évaluation de l'importance d'un attribut (section 3.6) ou de la pertinence d'une pondération des attributs (section 3.7) pour la classification non supervisée avant de conclure (section 3.8).

3.2 Caractéristiques des méthodes de sélection ou de pondération d'attributs

De nombreuses méthodes de sélection ou de pondération d'attributs ont été développées. On peut distinguer plusieurs caractéristiques à ces méthodes [John *et al.*, 1994 ; Wettschereck et Aha, 1995 ; Wettschereck *et al.*, 1997 ; Blum et Langley, 1997 ; Aha, 1998 ; Raman et Ioerger, 2003] :

- l'utilisation des connaissances : l'algorithme de pondération d'attributs peut être supervisé ou non supervisé (section 3.2.1) ;
- l'espace de recherche : les pondérations peuvent être binaires, dans le cas de la sélection d'attributs, ou réelles, dans le cas de la pondération d'attributs (section 3.2.2) ;
- la relation entre sélection/pondération d'attributs et classification : l'algorithme de sélection ou de pondération d'attributs peut être indépendant (approche filtre) ou lié (approches enveloppe et intégrée) à l'algorithme de classification (section 3.2.3) ;
- la portée des attributs sélectionnés ou des pondérations : les attributs sélectionnés ou les pondérations obtenues peuvent être appliqués pour l'ensemble des données ou être spécifiques à chacune des classes (section 3.2.4) ;
- le type d'évaluation : l'algorithme de sélection ou de pondération d'attributs peut se baser sur l'évaluation de l'importance relative de chacun des attributs ou sur la pertinence d'un sous-ensemble ou d'une pondération des attributs (section 3.2.5) ;

– le mode opératoire : il existe un grand nombre d'algorithmes de sélection ou de pondération d'attributs (section 3.5).

Dans la suite, nous détaillerons ces différentes caractéristiques.

3.2.1 Utilisation des connaissances

La sélection ou la pondération des attributs peut être réalisée en utilisant des exemples dont les classes sont connues, c'est-à-dire de manière supervisée, ou bien sans utiliser de connaissance, c'est-à-dire de manière non supervisée.

Les méthodes de sélection ou de pondération d'attributs supervisées sont toujours utilisées dans le cadre de la classification supervisée. En revanche, les méthodes de sélection ou de pondération d'attributs non supervisées, généralement utilisées dans le cadre de la classification non supervisée, sont parfois utilisées pour une classification supervisée en prétraitement de l'algorithme de classification [Groves et Bajcsy, 2003 ; Bajcsy et Groves, 2004].

3.2.2 Espace de recherche

La sélection d'attributs est souvent considérée comme un cas particulier de pondération d'attributs, dans lequel des poids binaires (0 ou 1) sont affectés. Mais les deux approches sont en réalité assez différentes l'une de l'autre, car elles ont des objectifs différents et des algorithmes spécifiques à chaque approche peuvent être mis en oeuvre.

3.2.2.1 Sélection d'attributs

La sélection d'attributs consiste à chercher le plus petit sous-ensemble d'attributs $F' \subset F$ portant la même information que F, sans nécessairement chercher à améliorer les résultats de classification. La suppression d'un attribut non pertinent peut cependant faciliter la classification des données.

Dans [John *et al.*, 1994], les auteurs distinguent trois niveaux de pertinence pour les attributs :

– les attributs non pertinents, qui n'apportent aucune information ;
– les attributs faiblement pertinents, qui contribuent à une bonne discrimination des classes mais qui peuvent être éliminés sous certaines conditions (il s'agit principalement des attributs redondants) ;
– les attributs fortement pertinents, indispensables à une bonne classification.

Le but de la sélection d'attributs est de simplifier les données en utilisant un minimum d'attributs pour décrire les objets, tout en conservant un maximum d'information, c'est-à-dire que la classification avec les attributs $F' \subset F$ est au moins aussi bonne que la classification avec les attributs F. Les données représentées par un espace de dimensionnalité moins élevée seront ainsi plus facilement interprétables. De plus, en utilisant moins d'attributs, les temps de calcul seront améliorés [John *et al.*, 1994 ; Dash et Liu, 2000 ; Søndberg-Madsen *et al.*, 2003]. La suppression d'attributs trop bruités, corrélés ou non pertinents va généralement permettre une amélioration des résultats de classification.

3.2.2.2 Pondération d'attributs

La pondération d'attributs est la recherche d'un ou plusieurs vecteurs de poids qui modifieront le degré d'utilisation des attributs dans un algorithme de classification. Plus un attribut F_j a un poids élevé, plus l'algorithme de classification tiendra compte des valeurs sur l'attribut F_j.

La pondération d'attributs consiste donc à trouver un vecteur de poids $W = (w_1, \ldots, w_n)$, avec $w_j > 0, \forall j$ et $\sum_{j=1}^{n} w_j = 1$.

La pondération d'attributs a pour but l'amélioration des résultats de la classification en utilisant principalement les attributs les plus importants, mais en conservant tous les détails des données. Dans [Wettschereck *et al.*, 1997], les auteurs montrent que les algorithmes de pondération d'attributs sont plus efficaces que les algorithmes de sélection d'attributs en terme de qualité de classification. Ceci est confirmé dans [Kohavi *et al.*, 1997], cependant les auteurs précisent qu'il est parfois préférable de chercher des pondérations discrètes, en nombre limité (par exemple deux valeurs non nulles).

3.2.3 Relation entre l'algorithme de sélection ou de pondération d'attributs et l'algorithme de classification

Généralement, les algorithmes de pondération d'attributs sont catégorisés en trois approches selon leurs liens avec l'algorithme de classification employé.

3.2.3.1 Approche filtre

L'approche *filtre* (ang. *filter*) consiste à pondérer les attributs en prétraitement, en se basant uniquement sur les données. Un algorithme de sélection ou de pondération d'attributs par approche filtre va permettre de découvrir un vecteur de poids W (binaires ou réels) à partir de l'ensemble de données D ; ce vecteur de poids et ensuite utilisé par un algorithme de classification M pour obtenir un ensemble de classes C (FIG. 3.3).

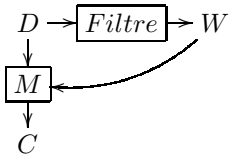


FIG. 3.3 : *Approche filtre pour la sélection/pondération d'attributs*

Dans ce type d'approche, les attributs sont évalués selon des critères qui ne dépendent pas de l'algorithme de classification qui sera utilisé, mais uniquement des propriétés des données. De nombreux critères ont ainsi été définis dans le cadre des méthodes de sélection ou de pondération d'attributs. Ces critères ont été définis pour la classification supervisée (c'est à dire des critères qui se base sur des exemples dont la classe est connue) et pour la classification non supervisée (c'est à dire qui se basent uniquement sur des propriétés intrinsèques aux données).

3.2.3.2 Approche enveloppe

Dans un algorithme de sélection ou de pondération d'attributs par approche *enveloppe* (ang. *wrapper*), les sous-ensembles d'attributs ou les pondérations obtenues par la méthode de recherche sont évalués en fonction de la qualité de la classification des données réalisée en les utilisant [John *et al.*, 1994 ; Kohavi et John, 1998]. L'approche enveloppe consiste donc à effectuer une classification par sous-ensemble ou pondération des attributs testé et à utiliser un ou plusieurs critères d'évaluation de la qualité d'une classification pour évaluer la qualité des attributs et modifier les poids en conséquence. Un algorithme de sélection ou de pondération d'attributs par approche enveloppe consiste donc à découvrir un vecteur de poids W (binaires ou réels) qui sera appliqué à une méthode de classification M pour obtenir un ensemble de classes C ; ces classes sont alors évaluées (selon un critère d'évaluation de la qualité d'une classification), et cette évaluation sera utilisée pour remettre en cause le vecteur de poids proposé, l'opération étant ainsi itéré jusqu'à atteindre le meilleur résultat possible (FIG. 3.4).

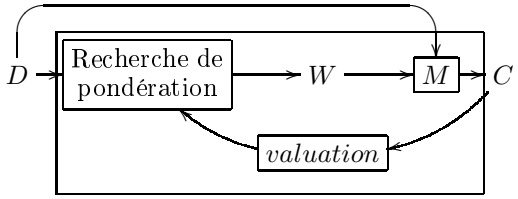


FIG. 3.4 : *Approche enveloppe pour la sélection/pondération d'attributs*

L'approche enveloppe nécessite de réaliser une classification par sous-ensemble ou pondération des attributs testé. Le temps de calcul d'un algorithme de pondération d'attributs par approche enveloppe est donc bien plus long que celui d'un algorithme par approche filtre. Cependant, une telle approche permet de s'adapter aux différents biais induits par les algorithmes de classification et donc d'obtenir de meilleurs résultats. Les pondérations obtenues sont en revanche spécifiques à la méthode de classification utilisée.

3.2.3.3 Approche intégrée

L'approche *intégrée* (ang. *embedded*) consiste à réaliser la sélection ou la pondération des attributs de manière conjointe à la classification. C'est le par exemple des algorithmes de construction d'arbre de décision comme ID3 et C4.5. Un algorithme de sélection ou de pondération d'attributs par approche intégrée propose à la fois un ensemble de classes et un vecteur de poids sur les attributs expliquant ces classes (FIG. 3.5).

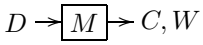


FIG. 3.5 : *Approche intégrée pour la sélection/pondération d'attributs*

L'approche intégrée a l'avantage d'être plus rapide que l'approche enveloppe tout en proposant un sous-ensemble ou une pondération adapté à la méthode de classification employée. Mais il s'agit souvent de méthodes de sélection/pondération très spécifiques à une méthode de classification et difficile, voire impossible, à généraliser à d'autres algorithmes.

3.2.4 Portée des attributs sélectionnés et des pondérations

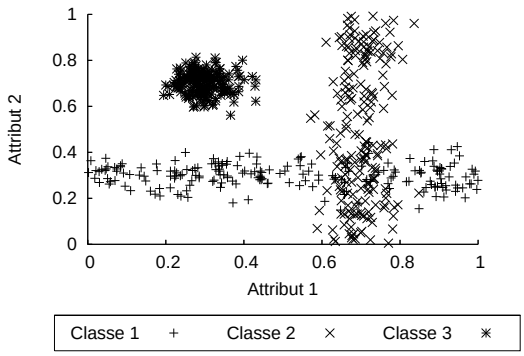
Les premières méthodes de sélection et de pondération d'attributs cherchaient un sous-ensemble ou une pondération qui s'applique à l'ensemble des objets à classifier (on parle alors de sélection/pondération globale des attributs). Or, dans certains cas, l'importance des attributs dépend de la classe à mettre en évidence. Ainsi, dans [Howe et Cardie, 1997 ; Howe et Cardie, 1999] les auteurs proposent de chercher un sous-ensemble ou des pondérations spécifiques à chaque classe.

Un algorithme de sélection/pondération locale des attributs permettra donc de définir un vecteur $W = (W_1, \ldots, W_K)$, où $W_k = (w_{k,1}, \ldots, w_{k,n})$ est le vecteur de poids pour la k-ième classe, permettant de discriminer au mieux la k-ième classe du reste des données.

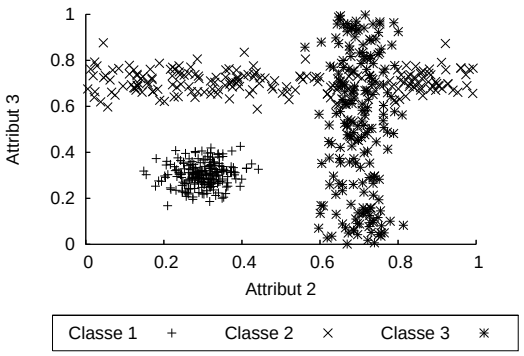
On appelle généralement *Subspace Clustering* ou *Projected Clustering* les méthodes de classification non supervisée avec sélection locale d'attributs pour la classification [Parsons *et al.*, 2004b ; Parsons *et al.*, 2004a].

Exemple :

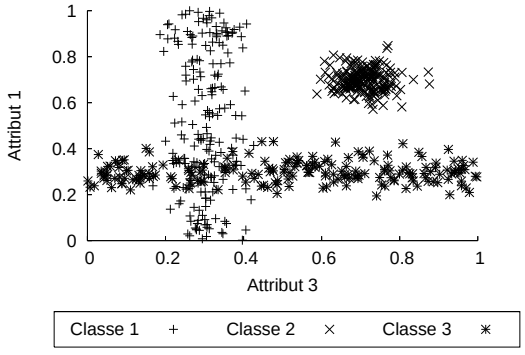
Les données représentées sur la figure 3.6, sont composées de trois classes de 200 objets. Chaque objet est représenté par trois attributs. On voit aisément que l'attribut 1 n'est pas pertinent pour la classe 1, que l'attribut 2 n'est pas pertinent pour la classe 2 et que l'attribut 3 n'est pas pertinent pour la classe 3. Ainsi, l'ensemble des attributs sont nécessaires pour discriminer les classes entre elles, mais il est préférable d'utiliser un sous-ensemble des attributs pour mettre en évidence chacune des classes.



(a) *Projection des données sur les attributs 1 et 2*



(b) *Projection des données sur les attributs 2 et 3*



(c) *Projection des données sur les attributs 1 et 3*

Fig. 3.6 : *Données pour lesquelles une sélection locale des attributs est préférable*

L'utilisation de sous-ensembles locaux ou de pondérations locales est cependant difficilement applicable dans le cas de certaines méthodes de classification, en particulier lorsque la méthode fait intervenir des distances inter-classes, car chaque classe est alors définie selon une métrique différente. La plupart des algorithmes de sélection/pondération locale des attributs sont des méthodes intégrées.

3.2.5 Type d'évaluation

Les méthodes de sélection ou de pondération d'attributs se basent toutes sur une évaluation de la pertinence des attributs. Il existe cependant différents critères d'évaluation qui se distinguent selon l'objectif visé et selon le paradigme utilisé.

3.2.5.1 Objectifs de l'évaluation

Dans [Lai *et al.*, 2006], les auteurs distinguent les approches mono-variables (ang. *univariate*) et les approches multi-variables (ang. *multivariate*).

Dans une *approche mono-variable*, les critères cherchent à déterminer l'importance des attributs individuellement (par leur qualité discriminatoire et leurs liens avec les autres attributs). Chaque attribut est évalué en fonction de son importance pour une tâche de classification. Les critères de ce type ne sont utilisables que dans le cadre de la sélection d'attributs. Des critères de ce type seront présentés, dans le cadre de la classification non supervisée, dans la section 3.6.

Dans une *approche multi-variables*, les critères cherchent à évaluer la qualité d'un sous-ensemble des attributs ou d'une pondération d'attributs. Dans ce cas c'est la combinaison des attributs qui est évaluée. Les critères de ce type peuvent cependant être utilisés pour évaluer l'importance d'un attribut F_j, en évaluant la qualité du sous-ensemble $F \setminus F_j$: plus le sous-ensemble est « mauvais », plus l'attribut est important. Des critères de ce type seront présentés, dans le cadre de la classification non supervisée, dans la section 3.7.

3.2.5.2 Paradigmes utilisés

On distingue généralement quatre catégories de critères pour la sélection/pondération d'attributs par approche filtre [Dash et Liu, 2003] :

- les critères basés sur une mesure de distance intra-classe et inter-classes ;
- les critères basés sur une mesure de consistance entre les attributs et les classes des objets ;
- les critères basés sur une évaluation de la quantité d'information contenue dans les attributs, par des mesures d'entropie ;
- les critères basés sur la dépendance et les corrélations entre les attributs.

Les critères basés sur la distance nécessitent de connaître les classes par avance. Les critères de ce type ne peuvent donc être définis que dans le cadre de la classification supervisée. Les critères de cette catégorie consistent à évaluer la compacité et la séparabilité des classes selon un attribut, un sous-ensemble des attributs ou une pondération des attributs.

Les critères basés sur la consistance cherchent à évaluer si un sous-ensemble des attributs contient toutes les informations nécessaires à la discrimination des classes. Ce type de critères ne peut donc être utilisé que pour la sélection d'attributs en classification supervisée.

Les critères basés sur l'entropie peuvent aussi bien permettre l'évaluation d'un attribut seul ou d'une pondération des attributs et cela de manière supervisée ou non supervisée.

Les critères basés sur la dépendance entre les attributs permettent l'évaluation d'un attribut seul et peuvent être aussi bien supervisés que non supervisés. La plupart des critères de cette catégorie comparent les attributs deux par deux. Pour faire l'évaluation selon un critère de dépendance dep d'un attribut F_j, il est nécessaire de faire l'agrégation des résultats des comparaisons avec chacun des autres attributs. Très souvent, une moyenne est utilisée :

$$dep\,(F_j) = \frac{1}{n-1} \sum_{F_{j'} \neq F_j} dep\,(F_j, F_{j'})$$

où $dep\,(F_j, F_{j'})$ est un critère d'évaluation de la dépendance entre deux attributs F_j et $F_{j'}$.

D'autres auteurs utilisent un algorithme de sélection ou de pondération d'attributs pour évaluer l'importance de chacun des attributs et utilisent ensuite ces mesures d'importance pour un autre algorithme. Par exemple, une méthode de pondération d'attributs qui trouve un vecteur de poids $W = (w_1, \ldots, w_n)$ permettra de définir l'importance d'un attribut F_j par w_j. Dans [Cardie, 1993 ; Cardie, 1996], les attributs utilisés dans C4.5 sont sélectionnés pour être utilisés dans un algorithme de raisonnement à base de cas. Dans [Schuschel et Hsu, 1998 ; Hsu *et al.*, 2002] l'importance d'un attribut est évaluée en fonction des poids des connections dépendant de cet attributs dans un réseau de neurones artificiels.

Enfin, la qualité de la classification est utilisée pour évaluer la qualité d'un sous-ensemble ou d'une pondération des attributs dans une approche enveloppe.

3.3 Utilisation des pondérations par les différentes méthodes de classification

Les pondérations des attributs indiquent à un utilisateur l'importance relative de chacun des attributs pour la discrimination des classes. Ils correspondent aux degrés d'utilisation des attributs dans le processus de classification. L'utilisation de pondérations dans un algorithme de classification dépend bien évidemment de la méthode de classification employée.

Dans le cas d'un sous-ensemble des attributs $F' \subset F$ issu d'un algorithme de sélection d'attributs, on définira une pondération par un vecteur de poids tel que $w_j = \frac{1}{card(F')}$ si $F_j \in F'$ et $w_j = 0$ sinon.

Nous allons présenter ici comment sont utilisés les poids obtenus par un algorithme de pondération d'attributs par des méthodes basées sur une distance (section 3.3.1), par des méthodes probabilistes (section 3.3.2) ou par l'algorithme de formation de concepts COBWEB (section 3.3.3).

3.3.1 Méthodes basées sur une distance

Dans les méthodes basées sur une mesure de distance (ou plus généralement sur une mesure de dissimilarité) comme K-means, DBSCAN ou SOM, les pondérations sont utilisées pour définir la distance (ou la dissimilarité) entre deux objets :

Définition 3.1 (Distance pondérée)
La distance pondérée selon un vecteur de poids $W = (w_1, \ldots, w_n)$ entre deux objets o et o' peut être définie par :

$$d_W(o, o') = \left(\sum_{1 \leq j \leq n} w_j \times (d_j(o, o'))^p \right)^{\frac{1}{p}}$$

où $d_j(o, o')$ est la distance entre o et o' selon le j-ième attribut.

Très souvent, les méthodes de pondération d'attributs ne travaillent pas sur les données brutes, mais en les normalisant de sorte que chaque attribut soit à la même échelle que les autres avant la pondération. Il existe plusieurs façons de normaliser les données [Scherf et Brauer, 1997 ; Howe et Cardie, 1999] :

Définition 3.2 (Normalisation dans $[0;1]$)
Un objet $o = (o_1, \ldots, o_n)$ composé d'attributs numériques peut être normalisé dans $[0;1]$ en un objet $\tilde{o} = (\tilde{o}_1, \ldots, \tilde{o}_n)$, avec :

$$\tilde{o}_j = \frac{o_j - o_j^{min}}{o_j^{max} - o_j^{min}}$$

où o_j^{min} (respectivement o_j^{max}) est la valeur minimale (respectivement maximale) que peut prendre un objet sur le j-ième attribut parmi tous les objets de D.

Définition 3.3 (Normalisation linéaire)
Un objet $o = (o_1, \ldots, o_n)$ composé d'attributs numériques peut être normalisé en un objet $\tilde{o} = (\tilde{o}_1, \ldots, \tilde{o}_n)$, de sorte que la moyenne (respectivement l'écart type) de chaque attribut soit de 0 (respectivement de 1), avec :

$$\tilde{o}_j = \frac{o_j - \overline{o}_j}{\sigma_j}$$

où $\overline{o}_j$ (respectivement σ_j) est la moyenne (respectivement l'écart type) des valeurs du j-ième attribut sur tous les objets de D.

DÉFINITION 3.4 (NORMALISATION DE DONNÉES HÉTÉROGÈNES)
Un objet $o = (o_1, \ldots, o_n)$ composé d'attributs pour lesquels une distance est définie peut être normalisé en un objet $\tilde{o} = (\tilde{o}_1, \ldots, \tilde{o}_n)$, de sorte que la moyenne des distance sur toutes les paires d'objets soit égale à 1, avec :

$$\tilde{o}_j = \frac{o_j}{\overline{d}_j}$$

où $\overline{d}_j$ est la moyenne des distances sur toutes les paires d'objets de D pour le j-ième attribut.

3.3.2 Méthodes probabilistes

Dans [Kim *et al.*, 2002 ; Cord *et al.*, 2006] des méthodes de sélection d'attributs sont appliquées à l'algorithme EM. Un attribut qui n'a pas été sélectionné n'a simplement pas d'influence dans le calcul de $P(o, \theta_k)$. Dans la version simplifié de EM présentée dans [Candillier *et al.*, 2005b], on suppose que tous les attributs sont indépendants, ce qui simplifie le calcul de $P(o, \theta_k)$. Pour un sous-ensemble des attributs F' on définit $P(o, \theta_k) = \prod_{j|F_j \in F'} P(o_j, \theta_{k,j})$.

Cette définition peut être étendue à la pondération d'attributs par $P(o, \theta_k) = \prod_{j=1}^{n} (P(o_j, \theta_{k,j}))^{|w_j|_{max}}$ où $|w_j|_{max} = \frac{w_j}{\max_{i \in [1;n]} w_i}$.

3.3.3 COBWEB

Comme nous l'avons vu dans la section 1.4.5, l'algorithme COBWEB utilise une mesure de prédictivité des concepts. La prédictivité d'un concept se calcule par la somme des prédictivités sur chacun des attributs. Cette somme peut alors simplement être pondérée selon le vecteur de poids utilisé.

3.4 Méthodes de sélection d'attributs

Une recherche exhaustive du meilleur sous-ensemble d'attributs est généralement impossible et n'a été que rarement utilisée [Almuallim et Dietterich, 1991], le temps de calcul devenant rapidement rédhibitoire.

Les deux approches généralement utilisées pour la sélection d'attributs sont le classement des attributs selon un critère d'évaluation (section 3.4.1) et l'optimisation d'une fonction d'évaluation d'un sous-ensemble d'attributs (section 3.4.2). Nous présenterons également des algorithmes de classification non supervisée avec sélection locale des attributs par approche intégrée (section 3.4.3).

3.4.1 Classement des attributs

Dans les méthodes de sélection d'attributs basées sur un classement des attributs (ang. *attribute ranking*), les attributs sont ordonnés du plus pertinent au moins pertinent selon un critère d'évaluation de la pertinence d'un attribut Q_f [Scherf et Brauer, 1997 ; Liu *et al.*, 1998 ; Dash et Liu, 2000 ; Dash et Liu, 2003]. Plusieurs méthodes de sélection d'attributs sont alors possibles :

- sélection des m meilleurs attributs : les m meilleurs attributs selon Q_f sont sélectionnés (cela nécessite cependant de connaître le nombre d'attributs à utiliser, ce qui n'est généralement pas le cas) ;
- sélection selon un seuil : tous les attributs dont l'évaluation selon Q_f est supérieure (ou inférieure, selon le critère utilisé) à un seuil sont sélectionnés ; ce seuil dépend grandement des données à classifier ;
- sélection par algorithme glouton : les attributs sont sélectionnés un à un (en commençant par le meilleur selon Q_f). Chaque ensemble ainsi obtenu est alors évalué selon un critère d'évaluation d'un sous-ensemble d'attributs Q_w ; l'algorithme s'arrête lorsque Q_w cesse de s'améliorer, ou bien lorsque tous les attributs ont été ajoutés, le meilleur ensemble obtenu selon Q_w étant alors retenu comme sous-ensemble optimal.

Dans le cas de l'utilisation de critères basés sur la dépendance entre les attributs, cet algorithme peut être légèrement modifié. L'attribut le moins corrélé aux autres est sélectionné en premier. Puis les attributs les moins corrélés à ceux déjà choisis sont itérativement sélectionnés, jusqu'à vérifier la condition d'arrêt.

Cette approche peut être problématique étant données qu'elle n'utilise qu'une évaluation monovariable, qui, par définition ne tient pas compte de toutes les relations entre les attributs. Dans le cas des indices d'évaluation des attributs basés sur la dépendances, ce n'est pas car un attributs est indépendant des autres qu'il est nécessairement important pour la classification des données. Les indices basés sur d'autres notions (distance, consistance ou entropie) ne tiennent pas compte des corrélations ce qui peut nuire au résultat : en effet un attributs indispensable mais indépendant des autres risque de ne pas être sélectionné par un algorithme de ce type.

3.4.2 Optimisation d'une fonction d'évaluation

La majorité des algorithmes de sélection d'attributs sont des algorithmes d'optimisation qui cherchent à optimiser un critère d'évaluation d'un sous-ensemble d'attributs.

3.4.2.1 Algorithmes gloutons

L'ensemble des sous-ensembles d'attributs peut être vu comme un graphe, où chaque nœud du graphe est un sous-ensemble de l'ensemble des attributs, et deux nœuds sont adjacents lorsque les sous-ensembles ne diffèrent que sur un attribut [Kohavi, 1994 ; Caruana et Freitag, 1994 ; Kohavi et Sommerfield, 1995 ; Domingos, 1997]. On cherche alors dans le graphe le nœud qui optimise un critère de qualité.

Les méthodes les plus employées sont des algorithmes gloutons où, à chaque itération de l'algorithme, le meilleur nœud voisin du nœud courant est choisi, en gardant en mémoire le meilleur nœud visité. Le parcours des nœuds varie suivant la méthode employée. Les méthodes les plus simples consistent à ajouter ou à supprimer un par un les attributs.

L'adjonction progressive d'attributs (ang. *Forward Selection, FS*) débute avec un ensemble F' des attributs utilisés vide ($F' = \varnothing$). À chaque étape, pour chaque attribut F_j qui n'a pas encore été sélectionné on évalue $F' \cup \{F_j\}$, et l'attribut qui maximise le critère d'évaluation est ajouté à F'. L'algorithme s'arrête lorsque l'ensemble de tous les attributs est atteint.

La suppression progressive d'attributs (ang. *Backward Elimination, BE*) est l'inverse de FS. L'algorithme BE débute avec l'ensemble de tous les attributs et les supprime un par un. Il s'arrête quand il n'y a plus d'attributs.

Les algorithmes *Forward Stepwise Selection* (FSS) et *Backward Stepwise Elimination* (BSE) consistent à une recherche bidirectionnelle. À chaque étape un attribut est ajouté ou supprimé à l'ensemble courant des attributs (en évitant de passer deux fois sur le même nœud). L'algorithme FSS débute avec un ensemble vide et l'algorithme BSE avec l'ensemble de tous les attributs.

Il est possible d'accélérer ces algorithmes en réalisant à chaque étape plusieurs adjonctions ou suppressions d'attributs, par exemple en ajoutant à chaque étape les deux attributs qui maximisent le critère dans un algorithme FS [Kohavi et Sommerfield, 1995].

Ces méthodes sont cependant critiquables car, comme nous l'avons vu dans le chapitre 2, ce type de méthode d'optimisation a de grand risque d'être bloqué dans un optimum local et de ne pas pouvoir trouver une solution satisfaisante.

3.4.2.2 Méthodes stochastiques

Afin de permettre d'obtenir des solutions plus satisfaisante qu'avec de simples algorithmes gloutons, de auteurs ont étudiés l'utilisation de méthodes stochastiques pour la sélection d'attributs. Différentes méthodes ont ainsi été employées avec succès depuis une simple génération aléatoire d'ensembles d'attributs aux algorithmes génétiques [Vafaie et Jong, 1993 ; Liu et Setiono, 1996 ; Yang et Honavar, 1997 ; Yang et Honavar, 1998 ; Kim *et al.*, 2000 ; Kim *et al.*, 2002 ; Cantù-Paz, 2002 ; Kim *et al.*, 2003].

Dans le cas d'un algorithme génétique, les individus représentent généralement des sous-ensembles des attributs. Les sous-ensemble d'attributs sont encodés par des chaînes de bits. Les individus sont évalués selon la fonction à optimiser.

3.4.3 Classification non supervisée avec sélection locale des attributs

Depuis quelques années, des méthodes de classification non supervisée avec sélection locale des attributs par approche intégrée ont été développées. On distingue deux catégories de méthodes : les méthodes ascendantes (ang. *bottom-up*) et des méthodes descendantes (ang. *top-down*) [Parsons *et al.*, 2004a ; Parsons *et al.*, 2004b].

Les méthodes ascendantes sont des méthodes basées sur la densité. Ces algorithmes consistent à chercher des *zones denses* dans les sous-espaces de faible dimensionnalité pour former les classes. Ce sont des méthodes par approche intégrée qui produisent généralement des classifications douces partielles, c'est-à-dire qu'il y peut exister des superposition entre les classes ainsi que des objets non classifiés.

Les méthodes descendantes consistent à produire une classification des données utilisant tous les attributs puis à déterminer les attributs les plus pertinents pour les classes ainsi obtenues. Ces attributs seront alors utilisés pour produire une nouvelle classification et ce processus sera réitéré jusqu'à stabilisation de la classification produite. Les méthodes descendantes sont généralement plus lentes que les méthodes ascendantes mais sont plus efficaces pour les données représentées dans un espace de dimensionnalité élevée.

Nous détaillerons les méthodes de classification non supervisée avec sélection locale des attributs CLIQUE (section 3.4.3.1), PROCLUS (section 3.4.3.2) et FINDIT (section 3.4.3.3). Une discussion de ces approches sera faites dans la section 3.4.3.4.

3.4.3.1 L'algorithme CLIQUE

Le premier algorithme de classification non supervisée avec sélection locale des attributs est l'algorithme CLIQUE [Agrawal *et al.*, 1998]. Il s'agit d'une méthode ascendante par approche intégrée.

CLIQUE est un algorithme basé sur une grille (cf. section 1.2.2.2). Chacune des dimensions est divisée en intervalles de taille fixée au début de l'algorithme. Une *unité* se définit comme un point de l'espace discret ainsi construit ou comme un hyperrectangle construit par l'intersection d'un intervalle de chacune des dimensions de l'espace de données considéré. Une *unité dense* est une unité contenant au moins τ objets (τ étant un paramètre de l'algorithme).

L'algorithme procède à une recherche d'unités denses dans des sous-ensembles d'attributs par une méthode basée sur l'algorithme APRIORI [Agrawal *et al.*, 1994] : les unités denses sont d'abord cherchées dans les espaces de dimension 1. Puis les unités denses de dimension n' sont construites à partir des unités denses de dimension $n' - 1$. Les classes sont alors produites en agrégeant les unités denses voisines.

Plusieurs extensions ont été proposées à cet algorithme. L'algorithme ENCLUS est similaire à CLIQUE mais utilise une mesure d'entropie à la place d'une mesure de densité [Cheng *et al.*, 1999]. L'algorithme MAFIA consiste à chercher une grille optimale pour l'algorithme CLIQUE, la taille des intervalles sur un même attribut n'étant pas nécessairement fixe [Nagesh *et al.*, 1999].

3.4.3.2 L'algorithme PROCLUS

L'algorithme PROCLUS [Aggarwal *et al.*, 1999] est une méthode de classification non supervisée avec sélection locale des attributs par approche intégrée basée sur des médoïdes (prototypes de classes appartenant à l'ensemble de données à classifier). Il s'agit d'une méthode descendante basée sur l'algorithme de classification non supervisée CLARANS [Ng et Han, 1994 ; Ng et Han, 2002]. L'algorithme PROCLUS a pour but de découvrir un médoïde et un sous-ensemble des attributs pour chacune des classes.

Tout d'abord, un sous-ensemble de données M est choisi comme ensemble des médoïdes potentiels par une méthode d'échantillonnage. Cet ensemble reste fixe tout au long de l'algorithme. À chaque itération de l'algorithme, un sous-ensemble des attributs est déterminé pour chacun des médoïdes courants. Pour cela, un voisinage $V(c_k) \in D$ est défini pour chaque médoïde, en fonction des autres médoïdes courants et d'une distance définie en utilisant tous les attributs. L'importance d'un attribut est alors évaluée pour chacun des médoïdes par un critère d'évaluation supervisée en utilisant les objets du voisinage de chacun des médoïdes comme exemples de l'ensemble d'apprentissage.

Les meilleurs attributs de chaque classe sont ensuite choisis. Le nombre d'attributs utilisés par chaque médoïde n_k n'est pas identique pour toutes les classes, mais il faut en revanche vérifier que $\sum_{k=1}^{K} n_k = K \times l$, où l est un paramètre de l'algorithme. Une seconde contrainte doit également être respectée : deux attributs au moins doivent être choisis pour chacune des classes. Les classes associées à chacun des médoïdes sont alors construites et évaluées selon un critère de compacité. Les médoïdes les plus « mauvais » sont alors remplacés par d'autres médoïdes potentiels choisis aléatoirement. Le processus est alors itéré.

3.4.3.3 L'algorithme FINDIT

L'algorithme FINDIT est une méthode de classification non supervisée avec sélection locale des attributs [Woo *et al.*, 2005]. Il s'agit d'une méthode descendante par approche intégrée.

Cet algorithme se base sur l'idée que deux objets peuvent être considérés comme proches l'un de l'autre s'ils le sont dans de nombreuses dimensions. Pour cela, une mesure de dissimilarité dissymétrique entre deux objets o et o', paramétrée par un seuil ϵ, est définie. Cette distance est définie pour deux objets placés dans deux sous-espaces des données F_o et $F_{o'}$:

$$d_{\epsilon,o}(o') = card(F_o) - card(\{F_j \mid d_j(o, o') \leq \epsilon, F_j \in F_o \cap F_{o'}\})$$

Une mesure de dissimilarité symétrique peut alors être définie par :

$$d_\epsilon(o, o') = \max(d_{\epsilon,o}(o'), d_{\epsilon,o'}(o))$$

Deux échantillons de données sont sélectionnés pour l'apprentissage, un l'ensemble de médoïdes, noté M, et un sous-ensemble des données, noté S. L'algorithme FINDIT se divise en trois

étapes : la sélection des dimensions associées à chaque médoïde, la formation des classes et l'évaluation du résultat.

Pour chaque médoïde $m \in M$, on définit un voisinage $V(m)$ en sélectionnant les plus proches voisins selon la mesure de dissimilarité d_ϵ. On calcule alors $c_j(m) = card(\{o \in V(m) \mid d_j(o,m) \leq \epsilon\})$ pour chaque attribut F_j. Les attributs pour lesquels la mesure $c_j(m)$ est assez élevée sont considérés comme pertinents pour le médoïde m.

Chaque objet $o \in S$ est alors affecté au médoïde m qui vérifie $d_{\epsilon,m}(o) = 0$ et qui a le plus grand nombre d'attributs associés. On obtient alors une classe pour chacun des médoïdes $m \in M$. Les classes ainsi définies sont alors regroupées en utilisant un algorithme de classification hiérarchique utilisant la distance d_ϵ. On obtient alors une nouvelle classification, notée C^ϵ, comportant moins de classes. Les classes $C_k^\epsilon \in C^\epsilon$ sont définies chacune par plusieurs médoïdes. Les attributs pertinents pour une classes $C_k^\epsilon \in C^\epsilon$ sont calculés en fonction des attributs pertinents des médoïdes qui définissent la classe. Les classes trop petites sont supprimées. On note m_k^ϵ les médoïdes associés à la classe C_k^ϵ et n_k^ϵ le nombre d'attributs associés.

Les auteurs considèrent que les classes doivent être décrites par le plus grand nombre de prototypes et d'attributs possibles afin de contenir le plus d'information possible. Ainsi, une classification C^ϵ est évaluée par le critère suivant :

$$Q(C^\epsilon) = \sum_{C_k^\epsilon \in C^\epsilon} (card(m_k^\epsilon) \times n_k^\epsilon)$$

L'algorithme est appliqué avec différentes valeurs pour le paramètre ϵ et le meilleur résultat selon le critère d'évaluation est retenu. La classification est alors étendue au reste de l'ensemble de données D. On remarque que l'algorithme FINDIT est le seul algorithme de sélection d'attribut qui cherche à maximiser le nombre d'attributs sélectionnés.

3.4.3.4 Synthèse

Les méthodes présentées dans cette section fonctionnent toutes selon une approche intégrée et se limitent donc à une certaine définition de la structure des classes. De plus, elles n'autorisent qu'une sélection des attributs. Or, nous avons vu que les méthodes de pondération d'attributs produisent de meilleures résultats en terme de classification dans le cadre supervisée. Il est donc légitime de faire l'hypothèse que cela se vérifie également dans le cadre non supervisée, et donc de chercher à développer des méthodes de pondération d'attributs non supervisée.

Ces méthodes ont cependant montré la faisabilité et la pertinence de la réduction de dimensionnalité pour la classification non supervisée.

3.5 Méthodes de pondération d'attributs

Nous exposons ici différentes méthodes de pondération d'attributs. Nous commençons par présenter des méthodes qui consistent à calculer les pondérations directement à partir des données (section 3.5.1). Nous discuterons ensuite brièvement des méthodes d'optimisation (section 3.5.2). Nous détaillerons enfin différents algorithmes de classification non supervisée intégrant une pondération locale des attributs basés sur K-means (section 3.5.3).

3.5.1 Calcul direct

Les méthodes de pondération d'attributs par calcul direct consistent à utiliser une mesure de l'importance d'un attribut comme poids pour cet attribut.

Dans [Akkus et Güvenir, 1996], pour chaque attribut F_j, les objets de l'ensemble d'apprentissage L sont classés selon leur ordre sur l'attribut F_j de sorte que $i < i' \Leftrightarrow o_j^i < o_j^{i'}$. Le poids sur le j-ième attribut est alors défini de la manière suivante :

$$w_j = \frac{1}{N-1} \sum_{i=2}^{N} \alpha_j^i$$

où $\alpha_j^i = 1$ si o^i et o^{i-1} appartiennent à la même classe et $\alpha_j^i = 0$ sinon.

Toujours dans [Akkus et Güvenir, 1996], une autre méthode est proposée. Une classification par attribut est effectuée. Chacune de ces classifications est évaluée selon le même critère (un critère à maximiser). On note q_j l'évaluation de la classification n'utilisant que le j-ième attribut. La qualité moyenne d'une classification aléatoire selon le même critère d'évaluation q_r est également calculée. Les auteurs définissent alors $w_j = q_j - q_r$.

Dans [Howe et Cardie, 1999], les auteurs proposent de définir les pondérations locales de la manière suivante :

$$w_{k,j} = \max\left(\Delta_{k,j}^{inter} - \Delta_{k,j}^{intra}, 0\right)$$

où $\Delta_{k,i}^{inter}$ est la moyenne des distances sur le j-ième attribut entre les objets de la k-ième classe et les autres objets et $\Delta_{k,i}^{intra}$ est la moyenne des distances sur le j-ième attribut entre les objets de la k-ième classe.

Ces méthodes peuvent être considérées comme l'équivalent pour la pondérations d'attributs des méthodes basées sur une évaluation mono-variable des attributs en sélection d'attributs. Les poids sur chacun des attributs sont en effet calculés indépendamment les uns des autres, ce qui laisse présager une grande sensibilité aux corrélations.

3.5.2 Pondération d'attributs par optimisation d'une fonction d'évaluation

La plupart des méthodes de pondération sont des méthodes d'optimisation d'une mesure de qualité d'une pondération des attributs. Pour cela, différentes méthodes peuvent être employées, des réseaux de neurones [Yeung et Wang, 2002 ; Wang *et al.*, 2004] aux algorithmes stochastiques [Punch *et al.*, 1993 ; Inza *et al.*, 2000]. Dans le cas de pondérations discrètes, la pondération d'attributs peut être traitée comme cela a été vu dans la section 3.4.2.1 pour la sélection d'attributs en utilisant un graphe [Kohavi *et al.*, 1997].

Dans le cas d'un algorithme génétique, les individus représentent généralement des pondérations des attributs et sont évalués par le critère d'évaluation de pondération d'attributs utilisé.

3.5.3 Méthodes par approche intégrée basées sur K-means

Une famille de méthodes de pondération d'attributs intégrées à K-means a été définie [Chan *et al.*, 2004 ; Frigui et Nasraoui, 2004 ; Huang *et al.*, 2005] : les pondérations pouvent être globales ou locales et s'appliquer à K-means pour une classification dure ou à Fuzzy-C-means pour une classification floue. De telles méthodes cherchent un ensemble de poids et un ensemble de centres de classes qui minimisent une fonction de coût dérivée de celles utilisées dans les algorithmes K-means et Fuzzy-C-means. Ces fonctions de coût tiennent compte à la fois de la distribution des objets dans les classes et de la pondération des attributs

Dans le cas de pondérations globales, on note $\widetilde{W} = (\widetilde{w}_1, \ldots, \widetilde{w}_n)$ les pondérations à optimiser. Dans le cas de pondérations locales, on note $\widetilde{W} = \left(\widetilde{W}_1, \ldots, \widetilde{W}_K\right)$, où $\widetilde{W}_k = (\widetilde{w}_{k,1}, \ldots, \widetilde{w}_{k,n})$ sont les pondérations locales à optimiser.

Les algorithmes de pondération d'attributs par approche intégrée basés sur K-means sont les suivants :

– Global-Attribute-Weighting-K-means (GAW-K-means), un algorithme de pondération globale basé sur K-means, la fonction de coût à optimiser est alors :

$$cost_{gaw}\left(c, C, \widetilde{W}\right) = \sum_{1\leq k\leq K}\sum_{o\in C_k}\sum_{1\leq j\leq n}\left(\widetilde{w}_j\right)^{\beta} d_j\left(o, c_k\right)^2$$

– Local-Attribute-Weighting-K-means (LAW-K-means), un algorithme de pondération locale basé sur K-means, la fonction de coût à optimiser est alors :

$$cost_{law}\left(c, C, \widetilde{W}\right) = \sum_{1\leq k\leq K}\sum_{o\in C_k}\sum_{1\leq j\leq n}\left(\widetilde{w}_{k,j}\right)^{\beta} d_j\left(o, c_k\right)^2$$

– Global-Attribute-Weighting-Fuzzy-C-means (GAW-Fuzzy-C-means), un algorithme de pondération globale basé sur Fuzzy-C-means, la fonction de coût à optimiser est alors :

$$cost_{fgaw}\left(c, \mu, \widetilde{W}\right) = \sum_{1\leq k\leq K}\sum_{o\in D}\left(\mu_k\left(o\right)\right)^f\sum_{1\leq j\leq n}\left(\widetilde{w}_j\right)^{\beta} d_j\left(o, c_k\right)^2$$

– Local-Attribute-Weighting-Fuzzy-C-means (LAW-Fuzzy-C-means), un algorithme de pondération locale basé sur Fuzzy-C-means, la fonction de coût à optimiser est alors :

$$cost_{flaw}\left(c, \mu, \widetilde{W}\right) = \sum_{1\leq k\leq K}\sum_{o\in D}\left(\mu_k\left(o\right)\right)^f\sum_{1\leq j\leq n}\left(\widetilde{w}_{k,j}\right)^{\beta} d_j\left(o, c_k\right)^2$$

Le paramètre f est un réel défini comme dans la méthode Fuzzy-C-means. Le paramètre β (appelé *exposant de discrimination*) est un réel tel que $\beta > 1$. Dans [Frigui et Nasraoui, 2004], les auteurs ont montré que lorsque β est proche de 1, les pondérations s'approchent de pondérations binaires et que, lorsque β est très grand, les poids sur les attributs sont très similaires.

On a de plus les contraintes suivantes :

– $0 \leq \widetilde{w}_j \leq 1$ et $\sum_{1\leq j\leq n}\widetilde{w}_j = 1$

– $0 \leq \widetilde{w}_{k,j} \leq 1$ et $\sum_{1\leq j\leq n}\widetilde{w}_{k,j} = 1$, $\forall k = 1 \ldots K$

– $0 \leq \mu_k\left(o\right) \leq 1$ et $\sum_{1\leq k\leq K}\mu_k\left(o\right) = 1$

Les pondérations finales retenues pour les méthodes globales sont $W = \left(\left(\widetilde{w}_1\right)^{\beta}, \ldots, \left(\widetilde{w}_n\right)^{\beta}\right)$. Les pondérations finales retenues pour les méthodes locales sont $W_k = \left(\left(\widetilde{w}_{k,1}\right)^{\beta}, \ldots, \left(\widetilde{w}_{k,n}\right)^{\beta}\right)$.

Pour les méthodes basées sur K-means, il est possible d'optimiser l'un des paramètres c, C ou $\widetilde{W}$ si les deux autres sont fixés, et pour les méthodes basées sur Fuzzy-C-means, il est possible d'optimiser l'un des paramètres c, μ ou $\widetilde{W}$ si les deux autres sont fixés.

De fait, les algorithmes sont eux-mêmes dérivés de K-means ou Fuzzy-C-means et consistent en trois optimisations successives : l'appartenance de objets aux classes en fonction des centres et des pondérations, l'optimisation des centres des classes en fonction de l'appartenance des objets et des pondérations et enfin l'optimisation des pondérations en fonction des centres des classes et de l'appartenance des objets aux classes (Algorithme 3.1).

Algorithme 3.1 Algorithme de pondération d'attributs basé sur K-means

```
1 Initialiser les centres des classes
2 while la condition d'arrêt n'est pas vérifiée do
3 |   Modifier l'appartenance des objets aux classes
4 |   Modifier les centres des classes
5 |   Modifier les pondérations
```

La modification de l'appartenance des objets aux classes se fait de la même manières que pour les algorithmes K-means et Fuzzy-C-means. Dans le cas des méthodes basées sur K-means les objets appartiennent à la classe dont le centre est le plus proche selon la distance pondérée. Dans le cas des méthodes basées sur Fuzzy-C-means, le degré d'appartenance des objets aux classes est calculé de la même façon que dans l'algorithme d'origine, mais en utilisant des distances pondérées.

La modification du centre d'une classe se fait en calculant la moyenne des objets appartenant à la classe (pour les méthodes basées sur K-means) ou en calculant la moyenne pondérée par le degré d'appartenance des objets à la classe (pour les méthodes basées sur Fuzzy-C-means).

La modification des pondérations consiste à donner d'autant plus de poids à un attribut que les classes sont compactes pour celui-ci. La compacité des classes pour un attribut peut être calculée sur tout l'ensemble des données (pour les méthodes cherchant des pondérations globales) ou pour une classe (pour les méthodes cherchant des pondérations locales).

En fixant c et C pour GAW-K-means ou c et μ pour GAW-Fuzzy-C-means, les pondérations globales $\widetilde{w}_j$ optimales peuvent être définies quelque soit $j \in [1;n]$ par :

$$\widetilde{w}_j = \frac{1}{\sum_{l=1}^{n} \left[\frac{sum_j}{sum_l}\right]^{\frac{1}{\beta-1}}}$$

avec :

- $sum_j = \sum_{1\leq k\leq K} \sum_{o\in C_k} d_j\left(c_k, o\right)^2$, pour GAW-$K$-means ;
- $sum_j = \sum_{1\leq k\leq K} \sum_{o\in D} \left(\mu_k\left(o\right)\right)^f d_j\left(c_k, o\right)^2$, pour GAW-Fuzzy-$C$-means.

En fixant c et C pour LAW-K-means ou c et μ pour LAW-Fuzzy-C-means, les pondérations globales $\widetilde{w}_{k,j}$ optimales peuvent être définies quelque soit $j \in [1;n]$ et quelque soit $k \in [1;K]$ par :

$$\widetilde{w}_{k,j} = \begin{cases} \frac{1}{nbNuls_k} & \text{si } sum_{k,j} = 0 \\ 0 & \text{si } sum_{k,j} \neq 0 \text{ et } nbNuls_k \neq 0 \\ \frac{1}{\sum_{l=1}^{n} \left[\frac{sum_{k,j}}{sum_{k,l}}\right]^{\frac{1}{\beta-1}}} & \text{si } nbNuls_k = 0 \end{cases}$$

avec :

- $sum_{k,j} = \sum_{o\in C_k} d_j\left(c_k, o\right)^2$, pour LAW-$K$-means ;
- $sum_{k,j} = \sum_{o\in D} \left(\mu_k\left(o\right)\right)^f d_j\left(c_k, o\right)^2$, pour LAW-Fuzzy-$C$-means ;
- $nbNuls_k = card\left(\{j \mid sum_{k,j} = 0\}\right)$.

Ces calculs pour la redéfinition des pondérations ne sont valables que dans le cas de l'utilisation de la distance euclidienne. Si une autre distance était utilisée, les définitions des fonctions de coûts en seraient légèrement modifiées et la redéfinition des pondérations en serait grandement compliquée.

Ces quatre méthodes sont parmi les plus récentes dans le domaine. Elles serviront d'ailleurs de base à une partie de notre travail (présentée dans le chapitre 4). De plus, elles permettent la pondération globale et locale des attributs, ce qui nous permettra d'évaluer l'importance de choisir les attributs spécifiquement à chaque classe.

Ces méthodes présentent cependant plusieurs limites. En premier lieu, elles sont basées sur K-means, ce qui implique que le nombre de classes doit être fixé au début de l'algorithme, mais surtout que les classes sont définies par un centre et une mesure de distance. Seules des classes sphériques dans une certaine métrique (dépendant de la pondération des attributs) peuvent donc être découvertes. En second lieu, on remarque que si plusieurs attributs ont les mêmes valeurs,

la méthode d'optimisation va leur donner le même poids. Cette méthode d'optimisation étant totalement dépendante du critère d'évaluation utiliser, celui-ci est probablement très sensible aux corrélations entre les attributs.

3.6 Évaluation de l'importance d'un attribut

La sélection d'attributs consiste à choisir les meilleurs attributs à utiliser pour la classification. Comme cela a été exposé dans la section 3.2.5, la recherche de ces meilleurs attributs peut se faire en évaluant un par un leur qualité discriminatoire. Dans cette section sont présentés les principaux critères d'évaluation de l'importance d'un attributs pour une tâche de classification non supervisée. Des critères d'évaluation supervisée sont présentés dans l'annexe B.

3.6.1 Critères basés sur l'entropie

Dans [Groves et Bajcsy, 2003] une mesure d'entropie est proposée pour évaluer l'importance d'un attribut catégoriel ou numérique discrétisé en m valeurs discrètes :

$$H_j(D) = -\sum_{i=1}^{m} p_i^j(D) \ln p_i^j(D)$$

où $p_i^j(D)$ est la probabilité que le j-ième attribut d'un objet $o \in D$ prenne la i-ième valeur discrète.

Une forte valeur sur l'indice H indique une grande importance de l'attribut pour la classification.

3.6.2 Critères basés sur la dépendance

Alors que dans la plupart des travaux, on cherche à supprimer les informations redondantes, dans [Søndberg-Madsen *et al.*, 2003], les auteurs font l'hypothèse que les attributs pertinents sont ceux qui présentent le plus de corrélations avec les autres attributs. Deux scores de dépendance entre les attributs discrets f et f' sont proposés :

$$SD\left(F_j, F'_{j'}\right) = H(F_j) - H(F_j \mid F_{j'})$$

$$\text{ou} \quad SD(F_j, F_{j'}) = 1 - \frac{1}{2}\left(\frac{\max\limits_{F_{j,i}} p(F_{j,i})}{\sum\limits_{F_{j',i'}} p(F_{j',i'}) \max\limits_{F_{j,i}} p(F_{j,i} \mid F_{j',i'})} + \frac{\max\limits_{F_{j',i'}} p(F_{j',i'})}{\sum\limits_{F_{j,i}} p(F_{j,i}) \max\limits_{F_{j',i'}} p(F_{j',i'} \mid F_{j,i})}\right)$$

où $H(F_j)$ est une mesure de l'entropie des données selon l'attribut F_j, $H(F_j \mid F_{j'})$ est l'entropie conditionnelle de F_j sachant $F_{j'}$ et $F_{j,i}$ est la i-ième valeur discrète sur le j-ième attribut.

Une forte valeur sur ces indices indique que les attributs sont indépendants. Les attributs importants, selon les auteurs, sont donc ceux qui minimisent ces indices.

Dans [Groves et Bajcsy, 2003] trois mesures utilisées pour éliminer les attributs (numériques) redondants, en faisant l'hypothèse que le j-ième attribut et le $j+1$-ième attributs sont liés (comme c'est le cas, par exemple, dans les bandes d'une image hyperspectrale), sont présentées :

$$D_j = \sum_{o \in D} |o_j - o_{j+1}|$$
$$D'_j = \sum_{o \in D} |o_{j-1} - 2o_j + o_{j+1}|$$
$$RatioM_j = \sum_{o \in D} \left| \frac{o_j}{o_{j+1}} - \mu\left(\frac{o_j}{o_{j+1}}\right) \right|$$

Une quatrième mesure, toujours sur des attributs numériques, ne fait pas cette hypothèse et permet de vérifier si deux attributs sont corrélés :

$$CorM_{i,j} = \frac{\mu(o_i \times o_j) - \mu(o_i) \times \mu(o_j)}{\sigma(o_i) \times \sigma(o_j)}$$

où $\mu(x)$ et $\sigma(x)$ représente la moyenne de x sur tous les objets o de D.

3.6.3 Synthèse

De nombreux critères ont été définis dans la littérature. Ces critères sont basés soit sur l'entropie, soit sur la dépendance entre les attributs. Les caractéristiques des principaux critères d'évaluation sont résumées sur la table 3.1.

Indice	Type de critère	Type de données	Référence
H	entropie	nominal, discret	[Groves et Bajcsy, 2003]
SD	dépendance	nominal, discret	[Søndberg-Madsen *et al.*, 2003]
D^*	dépendance	numérique	[Groves et Bajcsy, 2003]
D'^*	dépendance	numérique	[Groves et Bajcsy, 2003]
$RatioM^*$	dépendance	numérique	[Groves et Bajcsy, 2003]
$CorM$	dépendance	numérique	[Groves et Bajcsy, 2003]

*cette mesure suppose que les attributs F_j et F_{j+1} sont liés

Tab. 3.1 : *Critères d'évaluation non supervisée de l'importance d'un attribut*

Les différents indices de dépendance présentées dans [Groves et Bajcsy, 2003] ont pour but de détecter, et donc de supprimer les redondances d'information. Il est évident que ce type de critère ne peut pas être utilisé seul, car un attributs indépendants n'est pas nécessairement pertinent pour la classification. Dans [Søndberg-Madsen *et al.*, 2003] font l'hypothèse inverse et cherchent les attributs les plus corrélés dans les données, ce qui peut amener à supprimer des attributs cruciaux pour la classification.

3.7 Évaluation de la pertinence d'un sous-ensemble ou d'une pondération des attributs

Comme cela a été exposé dans la section 3.2.5, certains critères ne consistent pas à évaluer l'importance des attributs indépendamment les uns des autres, mais à évaluer un sous-ensemble ou une pondération des attributs. Ici encore, de nombreux critères ont été définis dans la littérature. Dans cette section, nous présentons les critères qui s'appliquent dans le cadre de la classification non supervisée. Les critères qui s'appliquent dans le cadre de la classification supervisée sont présentés dans l'annexe B.

3.7.1 Critères basés sur l'entropie

Des critères d'évaluation basés sur un score d'entropie sont présentés dans [Dash et Liu, 2000] et dans [Yeung et Wang, 2002 ; Wang *et al.*, 2004]. L'entropie pour deux objets o et o' peut-être définie par :

$$\begin{aligned} E_W(o,o') &= -S_W(o,o')\log(S_W(o,o')) - (1-S_W(o,o'))\log(1-S_W(o,o')) \\ E'_W(o,o') &= \frac{1}{2}(S_W(o,o')(1-S_1(o,o')) + S_1(o,o')(1-S_W(o,o'))) \end{aligned}$$

où $S_W(o,o')$ est une mesure de similarité entre deux objets selon un vecteur de poids W et S_1 une mesure de similarité en utilisant tous les attributs avec le même poids. $S_W(o,o') = e^{-\alpha \times d_W(o,o')}$ ou $S_W(o,o') = \frac{1}{1+\beta+d_W(o,o')}$, avec α est choisi de sorte que $\overline{S_W(o,o')} = 1/2$ et β de sorte que $\overline{S_1(o,o')} = 1/2$.

La valeur est basse si les objets sont très proches ou très éloignés (c'est-à-dire qu'ils appartiennent probablement à la même classe ou probablement à deux classes différentes). L'entropie pour l'ensemble des données peut alors être définie par la somme des entropies pour toutes les paires d'objets. Plus l'entropie est basse, plus il y a d'information et donc plus le sous-ensemble d'attributs (ou la pondération) est pertinent.

3.7.2 Critères basés sur les résultats de classification

L'approche enveloppe consiste à évaluer la qualité d'un sous-ensemble ou d'une pondération des attributs en fonction de la qualité d'une classification réalisée en utilisant ce sous-ensemble ou cette pondération. Les critères d'évaluation classiques sont donc utilisés (cf. section 1.3).

Cependant, pour les méthodes non supervisées cette approche n'est pas triviale. En effet, les critères d'évaluation d'une classification non supervisée, par exemple ceux basés sur une distance, sont dépendants de la métrique utilisée. Ainsi, dans [Dy et Brodley, 2000], afin de comparer la qualité de deux classifications C^1 et C^2 obtenues en utilisant deux pondérations W^1 et W^2, selon un critère $q_W(C)$ qui évalue la qualité d'une classification C avec une pondération W, les qualités sont normalisées de la manière suivante :

$$\begin{aligned} q\left(C^1\right) &= q_{W^1}\left(C^1\right) \times q_{W^2}\left(C^1\right) \\ q\left(C^2\right) &= q_{W^2}\left(C^2\right) \times q_{W^1}\left(C^2\right) \end{aligned}$$

Néanmoins, cette normalisation ne permet pas d'avoir un critère de qualité absolue, mais juste de comparer deux pondérations d'attributs. D'autres approches plus simples et plus générales consistent à normaliser les pondérations de sorte que la somme des poids d'un vecteur de poids fasse 1 ou encore à utiliser un critère indépendant de la métrique utilisée.

D'autres méthodes ne cherchent pas à optimiser un seul critère, mais une série de critères. Par exemple, en sélection d'attributs, en plus de chercher à maximiser la qualité de la classification, certaines méthodes cherchent à minimiser le nombre d'attributs utilisés [Kim *et al.*, 2000 ; Kim *et al.*, 2002 ; Kim *et al.*, 2003 ; Morita *et al.*, 2003].

3.7.3 Synthèse

Les critères d'évaluation multi-variable, permettant d'évaluer la qualité d'un sous-ensemble ou d'une pondération des attributs dans le cadre de la classification non supervisée sont de deux natures : basés sur l'entropie dans une approche filtre ou basés sur la qualité d'une classification produite dans une approche enveloppe (TAB. 3.2).

Indice	Type de critère	Type de données	Référence
E	entropie	quelconque	[Dash et Liu, 2000]
E'	entropie	quelconque	[Wang *et al.*, 2004]
Qualité de classification*	évaluation de classification	dépend de la méthode	[Kohavi et John, 1998]

*approche enveloppe

Tab. 3.2 : *Critères d'évaluation non supervisée de la pertinence d'une pondération des attributs*

3.8 Conclusion

La classification de données de dimensionnalité élevée est un problème important qui a donné lieu à de nombreux travaux de recherche.

En particulier, des auteurs ont montré, dans le cadre de la classification supervisée, que :

- des poids réels (pondération d'attributs) permettent d'obtenir de meilleurs résultats que des poids binaires (sélection d'attributs) ;
- une pondération d'attributs par approche filtre (poids indépendants de la méthode de classification), bien que plus rapide, produit de moins bons résultats que les approches enveloppe ou intégrée (poids liés à la méthode de classification) ;
- une pondération locale des attributs est plus efficace qu'une pondération globale.

Nous pensons cependant que ces hypothèses, démontrées dans le cas de la classification supervisée, se vérifient également en classification non supervisée. Dans ce cadre, le domaine de recherche est plus récent et n'a donné lieu, à notre connaissance, qu'à peu de publications. Il n'est pas possible à l'heure actuelle de comparer expérimentalement les différentes approches car les méthodes existantes sont trop peu nombreuses.

Notre attention s'est particulièrement portée sur quatre méthodes de classification non supervisée avec pondération d'attributs basées sur K-means. Bien que ces méthodes présentent des limites liées à l'algorithme K-means, elles permettent la pondération globale ou locale pour une classification dure ou floue. Ces méthodes utilisent une technique de *hill-climbing* basée sur des optimisations partielles pour minimiser une fonction de coût, ce qui ne garantit pas d'atteindre le minimum global.

C'est pourquoi nous souhaitons développer de nouveaux algorithmes de pondération locale d'attributs utilisant des algorithmes évolutionnaires comme méthode d'optimisation.

Méthodes proposées

Chapitre 4

Approches génétiques pour l'amélioration des algorithmes de pondération d'attributs basés sur K-means

4.1 Motivations

Dans le chapitre 3, nous avons présenté différents algorithmes de classification non supervisée avec pondération d'attributs. Notre attention s'est tout particulièrement portée sur une famille de méthodes issues de [Frigui et Nasraoui, 2004 ; Chan *et al.*, 2004 ; Huang *et al.*, 2005] et détaillée dans la section 3.5.3. Ces méthodes présentent en effet des caractéristiques intéressantes : elles permettent une pondération globale ou locale par approches intégrées. Ces méthodes sont basées sur les algorithmes K-means et Fuzzy-C-means et permettent d'obtenir une classification dure ou floue. Elles consistent en trois optimisations partielles (classes des objets, centres des classes et pondération des attributs) itérées dans le but d'optimiser une fonction de coût dérivée de celle utilisée dans K-means. Nous proposons dans ce chapitre une amélioration de ces algorithmes en remplaçant cette approche par *hill-climbing* par une approche génétique. En effet, les algorithmes génétiques sont des méthodes d'optimisation efficaces, comme nous l'avons présenté dans le chapitre 2.

Cette idée a déjà été proposée dans [Murthy et Chowdhury, 1996] où un algorithme génétique, présenté comme alternative à la méthode de *hill-climbing* classique de K-means, est proposé. Une solution est un partitionnement des données en K classes, évaluée selon la fonction de coût de K-means. D'une manière similaire à cette approche, il est envisageable d'utiliser un algorithme génétique pour optimiser l'une des fonctions de coût utilisées par les algorithmes de pondération d'attributs basés sur K-means.

Une autre approche est proposée dans [Krishna et Narasimha Murty, 1999] où les auteurs définissent une méthode hybride, combinant K-means et un algorithme évolutionnaire : l'opérateur classique de croisement est remplacé par une étape de recherche locale.

De manière plus générale, il est possible de définir un algorithme évolutionnaire lamarckien ou baldwinien en utilisant K-means comme méthode de recherche locale. Cela est également applicable pour chacune des méthodes de classification avec pondération d'attributs basées sur K-means : l'approche classique, par trois optimisations partielles, peut être utilisée comme méthode de recherche locale pour une approche lamarckienne ou baldwinienne.

Dans le cas de la recherche de pondérations locales, chaque classe est associée à un centre représentatif de la classe et un vecteur de poids indiquant les attributs pertinents pour discriminer la classe des autres classes. Le problème de classification en K classes se découpe alors naturellement en K sous-problèmes d'identification des centres des classes et des pondérations associées. Une approche par coévolution coopérative est donc possible, en utilisant une population par classe cherchée.

Nous allons donc proposer de nouvelles méthodes de pondérations utilisant des méthodes évolutionnaires. Il est possible de définir des méthodes par approche darwinienne, lamarckienne ou baldwinienne pour la pondération globale ou locale des attributs. De plus, la coévolution coopérative peut être utilisée dans le cadre des méthodes de pondération locale. Enfin, il est possible de chercher des classes dures ou floues. C'est ainsi, comme indiqué sur la table 4.1, que 18 méthodes basées sur des algorithmes génétiques ont été définies, selon qu'il s'agisse d'une méthode évolutionnaire ou coévolutionnaire, d'une stratégie darwinienne, lamarckienne ou baldwinienne, avec des pondérations globales ou locales et retournant une classification dure ou floue.

Algorithme	Type d'évolution	Stratégie	Pondérations	Type de classification
DE-GAW-K-means	Évolution	Darwinienne	Globales	Dure
LE-GAW-K-means	Évolution	Lamarckienne	Globales	Dure
BE-GAW-K-means	Évolution	Baldwinienne	Globales	Dure
DE-LAW-K-means	Évolution	Darwinienne	Locales	Dure
LE-LAW-K-means	Évolution	Lamarckienne	Locales	Dure
BE-LAW-K-means	Évolution	Baldwinienne	Locales	Dure
DC-LAW-K-means	Coévolution	Darwinienne	Locales	Dure
LC-LAW-K-means	Coévolution	Lamarckienne	Locales	Dure
BC-LAW-K-means	Coévolution	Baldwinienne	Locales	Dure
DE-GAW-Fuzzy-C-means	Évolution	Darwinienne	Globales	Floue
LE-GAW-Fuzzy-C-means	Évolution	Lamarckienne	Globales	Floue
BE-GAW-Fuzzy-C-means	Évolution	Baldwinienne	Globales	Floue
DE-LAW-Fuzzy-C-means	Évolution	Darwinienne	Locales	Floue
LE-LAW-Fuzzy-C-means	Évolution	Lamarckienne	Locales	Floue
BE-LAW-Fuzzy-C-means	Évolution	Baldwinienne	Locales	Floue
DC-LAW-Fuzzy-C-means	Coévolution	Darwinienne	Locales	Floue
LC-LAW-Fuzzy-C-means	Coévolution	Lamarckienne	Locales	Floue
BC-LAW-Fuzzy-C-means	Coévolution	Baldwinienne	Locales	Floue

Tab. 4.1 : *Algorithmes génétiques pour la pondération d'attributs dans K-means*

Dans la suite de ce chapitre seront présentés ces différents algorithmes : tout d'abord les méthodes évolutionnaires (section 4.2) puis les méthodes coévolutionnaires (section 4.3). Dans la section 4.4, nous évaluerons l'efficacité des algorithmes proposés sur différents ensembles de données avant de conclure sur ces méthodes (section 4.5).

Notations

- $\left(c', \widetilde{W}'\right) = \text{RL}\left(c, \widetilde{W}, s\right)$ est le résultat (les centres et les pondérations) de l'algorithme de recherche locale RL initialisé avec les centres c et les pondérations $\widetilde{W}$ après s étapes (où RL peut être GAW-K-means, LAW-K-means, GAW-Fuzzy-C-means ou LAW-Fuzzy-C-means) ;
- *cost* est la fonction de coût à minimiser (*cost* est l'une des quatre fonctions de coût $cost_{gaw}$, $cost_{law}$, $cost_{fgaw}$ ou $cost_{flaw}$ selon que l'on cherche des pondérations globales ou locales pour une classification dure ou floue).
- c^g et $\widetilde{W}^g$ sont respectivement l'ensemble des centres des classes et les pondérations (globales ou locales) de la meilleure solution obtenue lors des g premières générations ;
- m est le nombre d'individus dans une population.

4.2 Approche évolutionnaire

Nous commencerons par exposer comment sont encodées les solutions dans nos algorithmes évolutionnaires, ainsi que les principales opérations génétiques définies (section 4.2.1). Nous présenterons ensuite les trois versions darwinienne (section 4.2.2), lamarckienne (section 4.2.3) et baldwinienne (section 4.2.4).

4.2.1 Encodage des solutions et opérations génétiques

Les algorithmes proposés ont pour objectif de déterminer les centres des classes et les pondérations qui minimisent une fonction de coût. Un chromosome est alors composé de deux parties. L'une représente les centres et l'autre les pondérations. Il est donc nécessaire de définir les opérations génétiques pour chacune des deux parties.

4.2.1.1 Centres des classes

Dans le cas où tous les attributs sont numériques, le centre d'une classe peut être représenté par un vecteur de n valeurs réelles, l'ensemble des K centres est alors représenté simplement par un vecteur de $K \times n$ valeurs réelles.

Mais dans un cas plus général où les attributs sont de types plus hétérogènes, la représentation des centres est moins triviale. Dans [Murthy et Chowdhury, 1996 ; Krishna et Narasimha Murty, 1999], une classification est encodée par un vecteur d'entiers indiquant la classe de chaque objet. Cette représentation n'a cependant pas été retenue car, d'une part, la taille des chromosomes (égale au nombre d'objets de l'ensemble à classifier) risque d'être trop grande, et d'autre part, elle interdit une décomposition du problème selon les classes, nécessaire à l'application d'un algorithme de coévolution coopérative.

Nous avons donc décidé d'encoder les solution par les centres des classes. On note $c^{i,g} = \left(c_1^{i,g}, \ldots, c_K^{i,g}\right)$ la partie codant les centres des classes dans le chromosome de l'individu $I^{i,g}$ (le i-ième individu de la g-ième population). Chaque gène $c_k^{i,g}$ est de même structure qu'un objet dans l'espace de données considéré. Nous avons ensuite défini les diverses opérations génétiques pour les centres des classes : initialisation des individus, croisement, mutation.

L'initialisation de chaque centre se fait en calculant une moyenne pondérée aléatoirement de tous les objets de D, afin de générer des centres que l'on peut espérer être répartis dans l'espace des données.

Les croisements se font en calculant une moyenne entre chacun des centres encodés dans les chromosomes des deux parents. Afin de permettre plus de diversité dans les croisements, la moyenne sera pondérée différemment pour chacun des centres. Ainsi, le k-ième centre du résultat du croisement sera défini par $r \times c_k^{i,g} + (1-r) \times c_k^{i',g}$, où $c_k^{i,g}$ et $c_k^{i',g}$ les centres de la k-ième classe pour deux parents I^i et $I^{i'}$ à la g-ième génération et $r \in [0;1]$ est une valeur aléatoire.

La mutation d'un centre se fait par un opérateur de croisement entre la valeur courante du centre et un autre centre généré aléatoirement. Ainsi, une mutation sur le centre $c_k^{i,g}$ de la k-ième classe du chromosome d'un individu I^i à la g-ième génération se fait en générant aléatoirement un centre c_{rand}. Le résultat de la mutation est alors défini par $(1-r) \times c_k^{i,g} + r \times c_{rand}$ où $r \in [0;1]$ une valeur aléatoire. Définies de cette façon, les mutations peuvent perturber très fortement les solutions. Par contre, il est possible de modifier cette définition de sorte que la valeur maximale de r décroisse au cours de l'apprentissage afin de rendre les mutations de moins en moins importantes au fur et à mesure que l'algorithme converge vers une solution satisfaisante, cette solution étant inspirée du recuit simulé.

4.2.1.2 Pondérations

Qu'il s'agisse de pondérations globales ou des pondérations locales pour une classe, la somme des poids (avant application de l'exposant β) doit être égale à 1, comme cela a été vu dans la description des algorithmes (section 3.5.3). Ainsi, pour n attributs, il est possible de déterminer les n poids à partir de $n-1$ paramètres. On distingue trois encodages possibles pour un vecteur de poids $(\widetilde{w}_1, \ldots, \widetilde{w}_n)$:

- par un vecteur $\left(w'_1, \ldots, w'_{n-1}\right)$, avec $0 \leq w_j \leq 1$, $\widetilde{w}_j = w'_j, \forall 1 \leq j < n$ et $\widetilde{w}_n = 1 - \sum_{j=1}^{n-1} w'_j$;
- par un vecteur $\left(w'_1, \ldots, w'_{n-1}\right)$, avec $0 \leq w_j \leq 1$, $\widetilde{w}_1 = w'_1$, $\widetilde{w}_j = w'_j \times \left(1 - \sum_{l=1}^{j-1} \widetilde{w}_l\right)$ et $\widetilde{w}_n = 1 - \sum_{j=1}^{n-1} \widetilde{w}_j$;
- par un vecteur $(w'_1, \ldots, w'_n)$, avec $0 \leq w_j \leq 1$ et $\widetilde{w}_j = \frac{w'_j}{\sum_{j=1}^{n} w'_j}$.

Dans la première approche, il est possible d'encoder des solutions incohérentes si $\sum_{j=1}^{n-1} w'_j > 1$, ce qui demande une gestion des contraintes au niveau de l'algorithme génétique. Dans la deuxième approche, le premier gène w'_1 a une influence sur tous les poids $\widetilde{w}_j$, alors que le dernier gène n'a d'influence que sur les poids $\widetilde{w}_{n-1}$ et $\widetilde{w}_n$. Les mutations et les croisements auront donc un impact différent selon l'endroit où l'opération est effectuée : les modifications du phénotype seront plus importantes si l'on modifie les premiers gènes plutôt que les derniers. Enfin, la troisième approche utilise une dimension de plus que nécessaire et donc toutes les solutions sont représentées plusieurs fois dans l'espace de recherche. Pour cette dernière approche, il est possible de normaliser les poids au moment du calcul du phénotype (c'est-à-dire sans modifier le génotype) ou au moment de la reproduction, en modifiant le génotype.

Pour les trois types d'encodage, la valeur d'un gène est comprise entre 0 et 1, et la somme des poids correspondants est égale à 1.

Quelque soit le type d'encodage retenu, on note $\widetilde{W}^{i,g}$ la partie du chromosome représentant les pondérations utilisées. L'encodage varie qu'il s'agisse de pondérations globales, auquel cas $\widetilde{W}^{i,g}$ est un vecteur de poids (FIG. 4.1), ou locales, auquel cas $\widetilde{W}^{i,g} = \left(\widetilde{W}_1^{i,g}, \ldots, \widetilde{W}_K^{i,g}\right)$ est composé des vecteurs de poids de chaque classe (FIG. 4.2). Le gène $\widetilde{w}_j^{i,g}$ représente le poids sur le j-ième attribut d'un individu I^i à la g-ième génération dans le cas de pondérations globales. Le gène $\widetilde{w}_{k,j}^{i,g}$ représente le poids sur le j-ième attribut pour la k-ième classe d'un individu I^i à la g-ième génération dans le cas de pondérations locales.

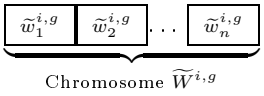


FIG. 4.1 : *Partie du chromosome correspondant aux pondérations dans l'approche évolutionnaire dans le cas de pondérations globales*

Il est nécessaire de définir les opérations génétiques pour la partie des chromosomes encodant les poids. Nous avons utilisé les définitions suivantes :

- initialisation : une valeur réelle (entre 0 et 1) est choisie aléatoirement pour chaque gène ;
- croisement : un croisement uniforme est utilisé car sinon l'ordre des attributs (choisi arbitrairement) aurait une influence sur les croisements ;

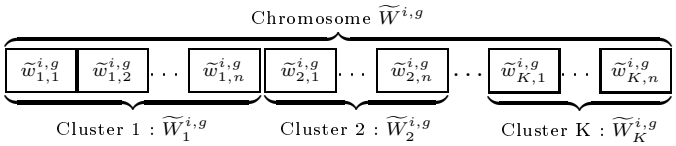


Fig. 4.2 : *Partie du chromosome correspondant aux pondérations dans l'approche évolutionnaire dans le cas de pondérations locales*

– mutation : la valeur d'un gène mutant est remplacée par une nouvelle valeur réelle générée aléatoirement.

4.2.2 Algorithme darwinien évolutionnaire

L'algorithme darwinien évolutionnaire (Algorithme 4.1), qu'il s'agisse de pondérations globales ou locales pour une classification dure ou floue, est le suivant :

Étape 0 – initialisation (lignes 1 à 5)

– initialisation des centres des classes de la solution courante c^1 (choisis aléatoirement parmi les objets de D) ;
– initialisation des poids de la solution courante $\widetilde{W}^1$ (initialisés à $1/n$) ;
– les individus sont initialisés aléatoirement.

À chaque génération g, la procédure suivante est répétée jusqu'à atteindre un critère d'arrêt (évaluation satisfaisante, nombre de génération maximale) :

Étape 1 – évaluation des individus (lignes 7 et 8)

Chaque solution (individu) est évaluée selon la fonction de coût *cost*.

Étape 2 – définition de la meilleure solution courante (lignes 9 à 13)

La meilleure solution obtenue à cette génération remplace la meilleure solution courante si son évaluation est inférieure selon la fonction *cost*.

Étape 3 – reproduction des individus (ligne 14)

Une nouvelle population est créée par reproduction (par les opérateurs de sélection, de croisement et de mutation).

4.2.3 Algorithme lamarckien évolutionnaire

L'algorithme présenté dans la section 4.2.2 a été étendu à un algorithme lamarckien en utilisant l'algorithme classique correspondant (c'est-à-dire GAW-K-means, LAW-K-means, GAW-Fuzzy-C-means ou LAW-Fuzzy-C-means) comme méthode de recherche locale. Durant sa vie, avant son évaluation, chaque individu va faire évoluer son génotype et son phénotype par une recherche locale. L'individu est évalué après cette recherche locale. Le nouveau matériel génétique est utilisé lors de la phase de reproduction.

L'algorithme lamarckien évolutionnaire (Algorithme 4.2) est presque identique à l'algorithme darwinien évolutionnaire sauf que :

– s_L étapes de l'algorithme de recherche locale sont effectuées, avec $s_L \in \mathbb{N}^*$ (ligne 8) ;
– les individus sont évalués selon leur résultat après la recherche locale (ligne 9) ;
– le matériel génétique des individus est modifié (lignes 10 et 11) ;
– s_G étapes de l'algorithme de recherche locale sont effectuées sur la meilleure solution de la génération courante, avec $s_G \in \mathbb{N}^*$ (ligne 13). Ce calcul est rajouté pour correspondre à l'algorithme coévolutionnaire présenté dans la section 4.3.

Algorithme 4.1 Algorithme darwinien évolutionnaire

1 initialisation de c^1 à partir de D
2 initialisation de $\widetilde{W}^1$
3 **pour** $i \leftarrow 1$ **à** m **faire**
4 ⌊ initialisation de $I^{i,1}$
5 $g \leftarrow 1$
6 **tant que non** *condition de fin* **faire**
7 ⌊ **pour** $i \leftarrow 1$ **à** m **faire**
8 ⌊ ⌊ $Eval^{i,g} \leftarrow cost\left(c^{i,g}, \widetilde{W}^{i,g}\right)$
9 ⌊ $b \leftarrow Argmax_{i\in[1,m]}\left(Eval^{i,g}\right)$
10 ⌊ **si** $cost\left(c^{b,g}, \widetilde{W}^{b,g}\right) < cost\left(c^{g}, \widetilde{W}^{g}\right)$ **alors**
11 ⌊ ⌊ $\left(c^{g+1}, \widetilde{W}^{g+1}\right) \leftarrow \left(c^{b,g}, \widetilde{W}^{b,g}\right)$
12 ⌊ **sinon**
13 ⌊ ⌊ $\left(c^{g+1}, \widetilde{W}^{g+1}\right) \leftarrow \left(c^{g}, \widetilde{W}^{g}\right)$
14 ⌊ $\left\{I^{i,g+1}\right\}_{i\in[1,m]} \leftarrow reproduction\left(\left\{\left(Eval^{i,g}, I^{i,g}\right)\right\}_{i\in[1,m]}\right)$
15 ⌊ $g \leftarrow g+1$

Algorithme 4.2 Algorithme lamarckien évolutionnaire

1 initialisation de c^1 à partir de D
2 initialisation de $\widetilde{W}^1$
3 **pour** $i \leftarrow 1$ **à** m **faire**
4 ⌊ initialisation de $I^{i,1}$
5 $g \leftarrow 1$
6 **tant que non** *condition de fin* **faire**
7 ⌊ **pour** $i \leftarrow 1$ **à** m **faire**
8 ⌊ ⌊ $\left(\hat{c}^{i,g}, \hat{W}^{i,g}\right) \leftarrow \mathrm{RL}\left(c^{i,g}, \widetilde{W}^{i,g}, s_L\right)$
9 ⌊ ⌊ $Eval^{i,g} \leftarrow cost\left(\hat{c}^{i,g}, \hat{W}^{i,g}\right)$
10 ⌊ ⌊ $c^{i,g} \leftarrow \hat{c}^{i,g}$
11 ⌊ ⌊ $\widetilde{W}^{i,g} \leftarrow \hat{W}^{i,g}$
12 ⌊ $b \leftarrow Argmax_{i\in[1,m]}\left(Eval^{i,g}\right)$
13 ⌊ $\left(c, \widetilde{W}\right) \leftarrow \mathrm{RL}\left(\hat{c}^{b,g}, \hat{W}^{b,g}, s_G\right)$
14 ⌊ **si** $cost\left(c, \widetilde{W}\right) < cost\left(c^{g}, \widetilde{W}^{g}\right)$ **alors**
15 ⌊ ⌊ $\left(c^{g+1}, \widetilde{W}^{g+1}\right) \leftarrow \left(c^{b,g}, \widetilde{W}^{b,g}\right)$
16 ⌊ **sinon**
17 ⌊ ⌊ $\left(c^{g+1}, \widetilde{W}^{g+1}\right) \leftarrow \left(c^{g}, \widetilde{W}^{g}\right)$
18 ⌊ $\left\{I^{i,g+1}\right\}_{i\in[1,m]} \leftarrow reproduction\left(\left\{\left(Eval^{i,g}, I^{i,g}\right)\right\}_{i\in[1,m]}\right)$
19 ⌊ $g \leftarrow g+1$

4.2.4 Algorithme baldwinien évolutionnaire

L'algorithme présenté dans la section 4.2.3 a été modifié pour devenir un algorithme baldwinien. Cet algorithme est presque identique à l'algorithme lamarckien, mais seul le phénotype est modifié lors de la recherche locale. C'est uniquement le matériel génétique d'origine qui est utilisé lors la phase de reproduction : les lignes 10 et 11 de l'algorithme lamarckien ont été supprimées.

4.3 Approche coévolutionnaire

La coévolution coopérative consiste à diviser un problème en plusieurs sous-problèmes, chacun étant associé à une population de l'algorithme génétique. Dans le cas de la classification en K classes avec pondération locale des attributs, il existe une décomposition naturelle en K problèmes symétriques, consistant chacun à découvrir le centre et les poids des attributs associés à une classe. Dans les algorithmes proposés, seule une évaluation dans un environnement simple a été implantée, bien qu'il eût été possible d'étendre les méthodes à des environnements multiples. Aux vues des premiers résultats de nos algorithmes, cela ne nous a pas paru nécessaire pour valider notre approche.

Après avoir exposé comment étaient encodées les solutions dans les algorithmes coévolutionnaires (section 4.3.1), nous présenterons les trois versions darwinienne (section 4.2.2), lamarckienne (section 4.2.3) et baldwinienne (section 4.2.4).

4.3.1 Encodage des solutions et opérations génétiques

L'encodage des individus est similaire à celle de l'approche évolutionnaire, à la différence que seul un centre et un vecteur de poids sont inclus dans chaque chromosome. On note $c^{i,k,g}$ la partie représentant le centre de la classe du chromosome de l'individu $I_k^{i,g}$ (le i-ième individu de la k-ième population à la g-ième génération) et $W^{i,k,g}$ la partie représentant le vecteur de pondérations locales (FIG. 4.3). Les opérations génétiques sont identiques à celles utilisées dans l'approche évolutionnaire.

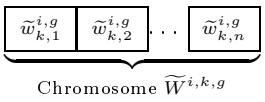


FIG. 4.3 : *Partie du chromosome correspondant aux pondérations dans l'approche coévolutionnaire*

4.3.2 Algorithme darwinien coévolutionnaire

L'algorithme darwinien évolutionnaire (Algorithme 4.3), qu'il s'agisse d'une classification dure ou floue, est le suivant :

Étape 0 – initialisation (lignes 1 à 6)
- initialisation des centres des classes c^1 (choisis aléatoirement parmi les objets de D) ;
- initialisation des poids de $\widetilde{W}^1$ (initialisés à $1/n$) ;
- les individus sont initialisés aléatoirement.

À chaque génération g, la procédure suivante est répétée jusqu'à atteindre un critère d'arrêt (évaluation satisfaisante, nombre de génération maximale) :

Étape 1 – évaluation des individus (lignes 9 à 12)
Chaque solution (individu) est évaluée selon la fonction de coût *cost*.

Étape 2 – sélection des meilleures solutions locales (lignes 13 et 14)
Les meilleurs individus de chaque population sont sélectionnés : $c^* = \{c_1^*, c_2^*, \ldots, c_K^*\}$, où c_k^* est le centre du meilleur individu de la k-ième population et $W^* = \{W_1^*, W_2^*, \ldots, W_K^*\}$, où W_k^* est le vecteur de pondérations locales du meilleur individu de la k-ième population.

Étape 3 – définition de la meilleure solution courante (lignes 15 à 19)
Les 2^K combinaisons possibles obtenues en choisissant, pour chaque classe, le centre et les poids de la meilleure solution locale ou ceux de la meilleure solution courante, sont évaluées selon la fonction de coût *cost*. La meilleure combinaison est retenue comme meilleure solution courante et sera utilisée dans les générations suivantes.

Étape 4 – reproduction des individus (lignes 20 et 21)
De nouvelles populations sont créées par reproduction (par les opérateurs de sélection, de croisement et de mutation), indépendamment pour chaque population.

Algorithme 4.3 Algorithme darwinien coévolutionnaire

1 initialisation de c^1 à partir de D
2 initialisation de $\widetilde{W}^1$
3 **pour** $k \leftarrow 1$ **à** K **faire**
4 **pour** $i \leftarrow 1$ **à** m **faire**
5 initialisation de $I^{i,k,1}$
6 $g \leftarrow 1$
7 **tant que non** *condition de fin* **faire**
8 **pour** $k \leftarrow 1$ **à** K **faire**
9 **pour** $i \leftarrow 1$ **à** m **faire**
10 $\check{c}^{i,k,g} \leftarrow \left\{c_1^g, c_2^g, \ldots, c^{i,k,g}, \ldots, c_K^g\right\}$
11 $\check{W}^{i,k,g} \leftarrow \left\{\widetilde{W}_1^g, \widetilde{W}_2^g, \ldots, \widetilde{W}^{i,k,g}, \ldots, \widetilde{W}_K^g\right\}$
12 $Eval^{i,k,g} \leftarrow cost\left(\check{c}^{i,k,g}, \check{W}^{i,k,g}\right)$
13 $b \leftarrow Argmax_{i\in[1,m]}\left(Eval^{i,k,g}\right)$
14 $(c_k^*, W_k^*) \leftarrow \left(c_k^{b,k,g}, \widetilde{W}_k^{b,k,g}\right)$
15 $Comb \leftarrow \left\{\left(c, \widetilde{W}\right) \mid \left(c_k, \widetilde{W}_k\right) = \left(c_k^*, \widetilde{W}_k^*\right) \text{ ou } \left(c_k, \widetilde{W}_k\right) = \left(c_k^g, \widetilde{W}_k^g\right)\right\}$
16 $\left(c^{g+1}, \widetilde{W}^{g+1}\right) \leftarrow \left(c^g, \widetilde{W}^g\right)$
17 **pour chaque** $\left(c, \widetilde{W}\right) \in Comb$ **faire**
18 **si** $cost\left(c, \widetilde{W}\right) < cost\left(c^{g+1}, \widetilde{W}^{g+1}\right)$ **alors**
19 $\left(c^{g+1}, \widetilde{W}^{g+1}\right) \leftarrow \left(c, \widetilde{W}\right)$
20 **pour** $k \leftarrow 1$ **à** K **faire**
21 $\left\{I^{i,g+1}\right\}_{i\in[1,m]} \leftarrow reproduction\left(\left\{\left(Eval^{i,g}, I^{i,g}\right)\right\}_{i\in[1,m]}\right)$
22 $g \leftarrow g + 1$

4.3.3 Algorithme lamarckien coévolutionnaire

L'algorithme présenté dans la section 4.3.2 a été étendu à un algorithme lamarckien en utilisant l'algorithme classique correspondant (c'est-à-dire LAW-K-means ou LAW-Fuzzy-C-means) comme méthode de recherche locale. Durant sa vie, avant son évaluation, chaque individu va faire évoluer son génotype et son phénotype par une recherche locale. L'individu est évalué après cette recherche locale. Le nouveau matériel génétique est utilisé lors de la phase de reproduction.

L'algorithme lamarckien coévolutionnaire (Algorithme 4.2) est presque identique à l'algorithme darwinien coévolutionnaire sauf que :

- s_L étapes de l'algorithme de recherche locale sont effectuées, avec $s_L \in \mathbb{N}^*$ (ligne 12) ;
- les individus sont évalués selon leur résultat après la recherche locale (ligne 13) ;
- le matériel génétique des individus est modifié (lignes 14 et 15) ;
- s_G étapes de l'algorithme de recherche locale sont effectuées sur chacune des combinaisons, avec $s_G \in \mathbb{N}^*$ (ligne 21). Ce calcul a pour but d'unifier les solutions partielles obtenues dans chacune des populations.

Il est important de noter que la méthode de recherche locale employée (c'est-à-dire LAW-K-means ou LAW-Fuzzy-C-means) modifie les centres et les poids de chacune des classes, alors que les individus ne correspondent chacun qu'à une des classes. Il est en effet impossible, avec une méthode basée sur K-means, de modifier une classe sans influencer les autres. Ce qui signifie que modifier le phénotype d'un individu avec cette méthode de recherche locale va également modifier son environnement.

Algorithme 4.4 Algorithme lamarckien coévolutionnaire

1 initialisation de c^1 à partir de D
2 initialisation de $\widetilde{W}^1$
3 **pour** $k \leftarrow 1$ **à** K **faire**
4 **pour** $i \leftarrow 1$ **à** m **faire**
5 initialisation de $I^{i,k,1}$
6 $g \leftarrow 1$
7 **tant que non** *condition de fin* **faire**
8 **pour** $k \leftarrow 1$ **à** K **faire**
9 **pour** $i \leftarrow 1$ **à** m **faire**
10 $\check{c}^{i,k,g} \leftarrow \left\{c_1^g, c_2^g, \ldots, c^{i,k,g}, \ldots, c_K^g\right\}$
11 $\check{W}^{i,k,g} \leftarrow \left\{\widetilde{W}_1^g, \widetilde{W}_2^g, \ldots, \widetilde{W}^{i,k,g}, \ldots, \widetilde{W}_K^g\right\}$
12 $\left(\hat{c}^{i,k,g}, \hat{W}^{i,k,g}\right) \leftarrow \mathrm{RL}\left(\check{c}^{i,k,g}, \check{W}^{i,k,g}, s_L\right)$
13 $Eval^{i,k,g} \leftarrow cost\left(\hat{c}^{i,k,g}, \hat{W}^{i,k,g}\right)$
14 $c^{i,k,g} \leftarrow \hat{c}_k^{i,k,g}$
15 $\widetilde{W}^{i,k,g} \leftarrow \hat{W}_k^{i,k,g}$
16 $b \leftarrow Argmax_{i\in[1,m]}\left(Eval^{i,k,g}\right)$
17 $(c_k^*, W_k^*) \leftarrow \left(c_k^{b,k,g}, \widetilde{W}_k^{b,k,g}\right)$
18 $Comb \leftarrow \left\{\left(c, \widetilde{W}\right) \mid \left(c_k, \widetilde{W}_k\right) = \left(c_k^*, \widetilde{W}_k^*\right) \text{ ou } \left(c_k, \widetilde{W}_k\right) = \left(c_k^g, \widetilde{W}_k^g\right)\right\}$
19 $\left(c^{g+1}, \widetilde{W}^{g+1}\right) \leftarrow \left(c^g, \widetilde{W}^g\right)$
20 **pour chaque** $\left(c, \widetilde{W}\right) \in Comb$ **faire**
21 $\left(c, \widetilde{W}\right) \leftarrow \mathrm{RL}\left(c, \widetilde{W}, s_G\right)$
22 **si** $cost\left(c, \widetilde{W}\right) < cost\left(c^{g+1}, \widetilde{W}^{g+1}\right)$ **alors**
23 $\left(c^{g+1}, \widetilde{W}^{g+1}\right) \leftarrow \left(c, \widetilde{W}\right)$
24 **pour** $k \leftarrow 1$ **à** K **faire**
25 $\left\{I^{i,g+1}\right\}_{i\in[1,m]} \leftarrow reproduction\left(\left\{\left(Eval^{i,g}, I^{i,g}\right)\right\}_{i\in[1,m]}\right)$
26 $g \leftarrow g + 1$

Cependant, comme on pouvait s'y attendre, l'expérience a montré que si s_L est suffisamment petit, les différences entre les classes avant et après la recherche locale seront suffisamment mineures pour ne pas nuire au résultat de la classification. De plus, les s_G étapes de recherche locale sur les combinaisons de classes obtenues pendant la génération permet d'unifier les solutions partielles obtenues dans chacune des populations et de minimiser encore les différences obtenues entre les différentes solutions locales à chaque population.

4.3.4 Algorithme baldwinien coévolutionnaire

L'algorithme présenté dans la section 4.3.3 a été modifié pour devenir un algorithme baldwinien. Cet algorithme est presque identique à l'algorithme lamarckien, mais seul le phénotype est modifié lors de la recherche locale. Le matériel génétique d'origine est utilisé lors la phase de reproduction : les lignes 14 et 15 de l'algorithme lamarckien ont été supprimées.

4.4 Évaluation des algorithmes

Afin d'évaluer l'efficacité de ces nouveaux algorithmes, nous allons les comparer avec les méthodes classiques par *hill-climbing*.

Nous allons tout d'abord présenter les ensembles de données sur lesquels les algorithmes ont été testés (section 4.4.1) ainsi que les paramètres utilisés avec chacun d'entre eux (section 4.4.2). Les différentes méthodes sont comparées selon différents critères. Nous allons tout d'abord comparer les différents algorithmes selon la fonction de coût qu'ils cherchent tous à minimiser (section 4.4.3). Nous évaluerons ensuite la pertinence des classes obtenues par les différents algorithmes en les comparant aux classes réelles des objets (section 4.4.4). Nous évaluerons également la stabilité des résultats dans la section 4.4.5. Nous comparerons les pondérations obtenues, ce qui nécessitera un traitement particulier dans le cas de pondérations locales, dans la section 4.4.6. Enfin les temps de calcul seront évalués dans la section 4.4.7.

Dans ce chapitre seront présentés les résultats des algorithmes de classification dure. Pour chacun des critères, nous avons comparé les méthodes de pondération globale entre elles (c'est-à-dire GAW-K-means, DE-GAW-K-means, LE-GAW-K-means et BE-GAW-K-means) et les méthodes de pondération locale (c'est-à-dire LAW-K-means, DE-LAW-K-means, LE-LAW-K-means, BE-LAW-K-means, DC-LAW-K-means, LC-LAW-K-means et BC-LAW-K-means). Toutes ces méthodes ont aussi été comparées à l'algorithme K-means afin d'évaluer l'influence de la pondération d'attributs sur la qualité de la classification.

4.4.1 Données

Nous avons testé nos algorithmes sur différents ensembles de données pour lesquels la classe réelle de chaque objet est connue, afin de vérifier la qualité des classes obtenues. De plus, pour certains ensembles de données, une information concernant la pertinence des attributs est fournie. Il est possible de comparer cette information avec les pondérations obtenues par les différents algorithmes.

Nous avons tout d'abord testé les différents algorithmes sur des ensembles de données artificiels (section 4.4.1.1). Nous avons également testé les algorithmes sur des ensembles de données classiquement utilisés en fouille de données (section 4.4.1.2).

4.4.1.1 Données artificielles

Nous avons mis au point un générateur aléatoire de données. Il est possible d'y configurer le nombre de classes et d'attributs ainsi que le nombre d'objets dans chacune des classes. Il est également possible, pour chaque attribut, de décider pour quelles classes il est pertinent :

- un attribut non pertinent pour une classe suit une loi de distribution uniforme entre 0 et 1 ;
- un attribut pertinent suit une loi de distribution normale, configurée par sa moyenne et son écart type.

Il est enfin possible de dupliquer un attribut (en perturbant légèrement les duplicata par un bruit gaussien) afin de créer des attributs fortement corrélés.

Les données présentées dans le chapitre 3, que nous utilisons dans nos tests, ont été produites par ce générateur. Ces données présentent les principaux problèmes auxquels nous souhaitsons trouver une solution. Nous allons donc pouvoir évaluer les algorithmes selon leur capacité à classifier les données lorsqu'un attribut n'est pas pertinent (DA1), lorsque des attributs sont corrélés (DA2) ou lorsque les attributs pertinents diffèrent d'une classe à l'autre (DA3).

L'ensemble de données DA1 (FIG. 4.4) est composé de trois classes de 200 objets chacune. Les objets sont représentés selon deux attributs. L'attribut 1 n'est pertinent pour aucune classe. L'attribut 2 suit une distribution normale pour chacune des trois classes, avec pour moyenne 0,85

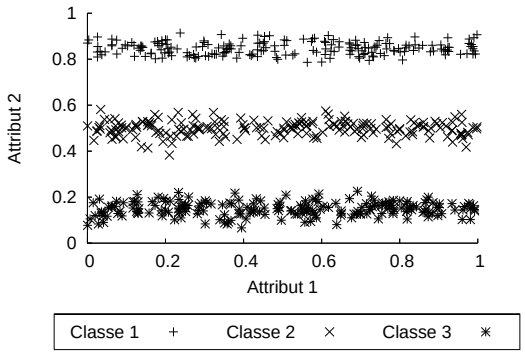


FIG. 4.4 : *Ensemble de données* DA1

(respectivement 0,5 et 0,15) pour la classe 1 (respectivement pour les classes 2 et 3), et pour écart type 0,03 pour chacune des classes.

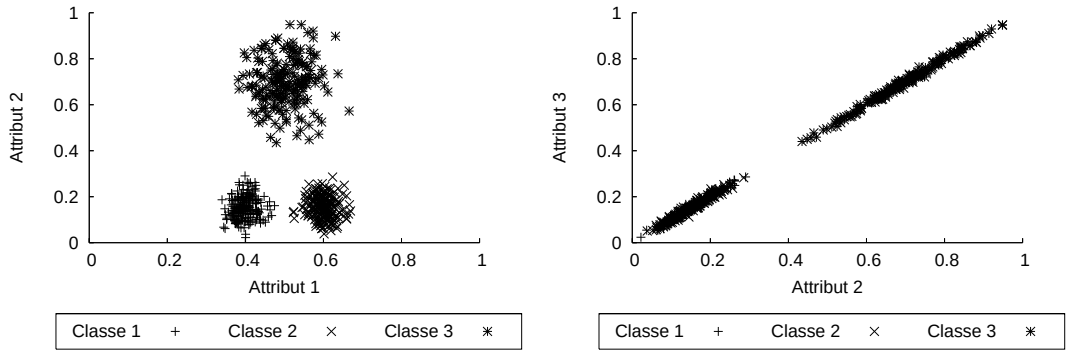
L'ensemble de données DA2 (FIG. 4.5) est composé de trois classes de 200 objets chacune. Les objets sont représentés selon cinq attributs. Tous les attributs sont pertinents, mais les attributs 2, 3, 4 et 5 sont corrélés entre eux. L'attribut 1 suit une distribution normale pour chacune des trois classes, avec pour moyenne 0,4 (respectivement 0,6 et 0,5) pour la classe 1 (respectivement pour les classes 2 et 3), et pour écart type 0,025 (respectivement 0,025 et 0,05) pour la classe 1 (respectivement pour les classes 2 et 3). L'attribut 2 suit une distribution normale pour chacune des trois classes, avec pour moyenne 0,15 (respectivement 0,15 et 0,7) pour la classe 1 (respectivement pour les classes 2 et 3), et pour écart type 0,05 (respectivement 0,05 et 0,1) pour la classe 1 (respectivement pour les classes 2 et 3). Ainsi les attributs 1 et 2 sont tous deux indispensables pour discriminer efficacement les classes entre elles. En particulier, si seul l'attribut 2 était utilisé, les classes 1 et 2 ne serait pas séparées. Comme l'attribut 2 est dupliqué plusieurs fois, l'information qu'il porte aura une importance relative accentuée par rapport à l'information portée par l'attribut 1.

L'ensemble de données DA3 (FIG. 4.6) est composé de trois classes de 200 objets chacune. Les objets sont représentés selon trois attributs. L'attribut 1 n'est pas pertinent pour la classe 1, et suit une distribution normale de moyenne 0,7 (respectivement 0,3) pour la classe 2 (respectivement pour la classe 3). L'attribut 2 n'est pas pertinent pour la classe 2, et suit une distribution normale de moyenne 0,3 (respectivement 0,7) pour la classe 1 (respectivement pour la classe 3). L'attribut 3 n'est pas pertinent pour la classe 3, et suit une distribution normale de moyenne 0,3 (respectivement 0,7) pour la classe 1 (respectivement pour la classe 1). L'écart type sur chacun des attributs et chacune des classes est de 0,05. Ainsi, la classe 1 peut être discriminée des deux autres grâce aux attributs 2 et 3, la classe 2 grâce aux attributs 1 et 3 et la classe 3 grâce aux attributs 1 et 2.

4.4.1.2 Données de l'UCI

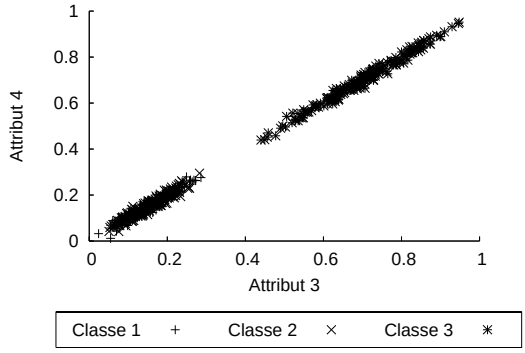
L'Université de Californie à Irvine offre gracieusement à la communauté de fouille de données une série d'ensemble de données présentant des caractéristiques variées pour le validation d'algorithmes de classification [Blake et Merz, 1998]. Nous avons choisi de tester nos algorithmes sur l'ensemble de données IRIS pour lequel des informations détaillées sur les attributs sont fournies, et sur les ensembles DIABETES, IONOSPHERE et SONAR qui comportent un grand nombre d'attributs.

L'ensemble de données IRIS PLANTS DATABASE est l'un des ensembles de données les plus connus dans ce cadre. Cet ensemble est composé de trois classes de 50 objets, chacune des classes représentant un type d'iris (*Iris Setosa*, *Iris Versicolor* et *Iris Virginica*). L'une des classes est linéairement séparable des autres, les deux autres ne le sont pas. Les données sont représentées par quatre attributs numériques (*sepal length*, *sepal width*, *petal length* et *petal width*). On sait que les deux derniers attributs sont particulièrement pertinents.

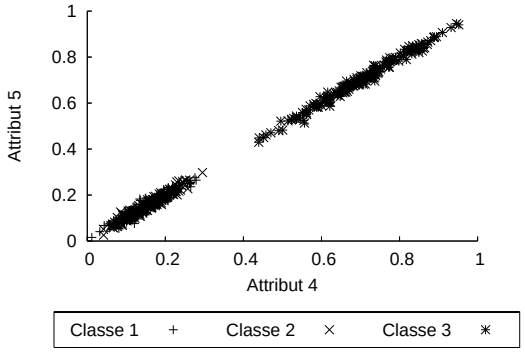


(a) *Projection des données sur les attributs 1 et 2*

(b) *Projection des données sur les attributs 2 et 3*



(c) *Projection des données sur les attributs 3 et 4*



(d) *Projection des données sur les attributs 4 et 5*

FIG. 4.5 : *Ensemble de données* DA2

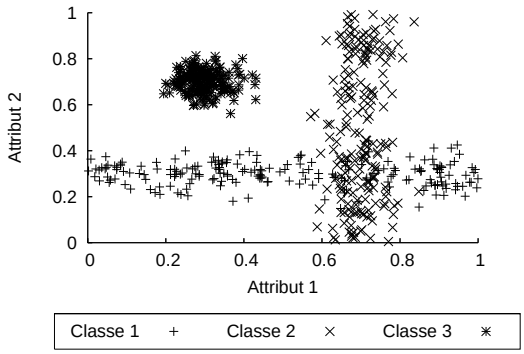
L'ensemble de données PIMA INDIANS DIABETES DATABASE contient des informations concernant le diagnostique du diabète. Les données sont séparées en deux classes, selon que la personne ait été testée positivement au diabète ou pas. Il y a 268 cas positifs pour 500 cas négatifs. Les données sont représentées sous la forme de huit attributs numériques, représentant entre autres l'âge, l'indice de masse corporelle ou encore la pression sanguine.

L'ensemble de données JOHNS HOPKINS UNIVERSITY IONOSPHERE DATABASE est constitué de données radar permettant la détection du structures particulières dans la ionosphère. Il s'agit d'un problème de classification binaire, avec 225 instances positives et 126 instances négatives. Les données sont représentées par 34 attributs numériques.

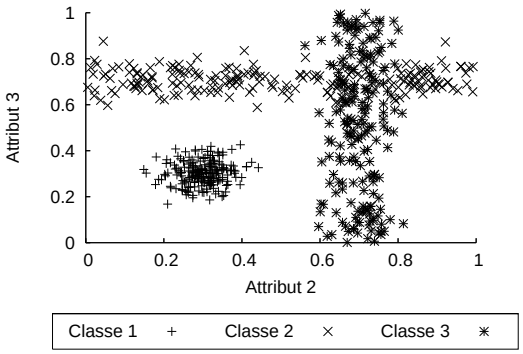
L'ensemble de données SONAR, MINES VS. ROCKS est constitué de deux classes de données. La première classe est composée de 111 objets représentant un cylindre métallique observé selon différents angles et différentes conditions par un sonar. La seconde est composée de 97 observations d'une pierre selon les même conditions. Les objets sont représentés par 60 attributs numériques.

4.4.2 Configuration des algorithmes

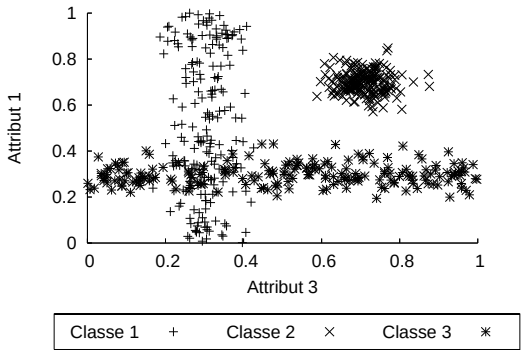
Les paramètres des algorithmes sont de différentes natures : issus des méthodes de classifications avec pondérations d'origine, issus des algorithmes génétiques et apportés par notre approche. Afin de valider notre approche, nous avons choisi de n'évaluer les algorithmes qu'en utilisant des valeurs standard pour les différents paramètres et de ne pas étudier l'influence de chacun d'eux.



(a) *Projection des données sur les attributs 1 et 2*



(b) *Projection des données sur les attributs 2 et 3*



(c) *Projection des données sur les attributs 1 et 3*

FIG. 4.6 : *Ensemble de données* DA3

On considère tout d'abord les paramètres des méthodes GAW-K-means, LAW-K-means, GAW-Fuzzy-C-means et LAW-Fuzzy-C-means :

- le nombre de classes, défini selon l'ensemble de données traité ;
- le nombre d'itérations (ou de générations), fixé à 100 ;
- l'exposant discriminatif β, fixé à 1,8, valeur par défaut proposée dans [Chan *et al.*, 2004] ;
- l'exposant discriminatif f pour les méthodes floues, classiquement fixé à 2.

Viennent ensuite les paramètres spécifiques aux algorithmes génétiques. Nous n'avons utilisé que des valeurs utilisées généralement dans la littérature :

- le taux de croisement, fixé à 70 % ;
- le taux de mutation, fixé à 5 % ;
- la méthode de sélection, par sélection proportionnelle.

Le nombre d'individus a été fixé à 20 par population pour les algorithmes par coévolution coopérative et à $20 \times K$ pour les méthodes évolutionnaires classiques, afin d'avoir le même nombre d'individus évalués à chaque génération pour toutes les méthodes.

Restent enfin les paramètres propres aux approches lamarckienne et baldwinienne :

- le nombre d'étapes dans la recherche locale au niveau des individus s_L, fixé à 1 ;
- le nombre d'étapes dans la recherche locale au niveau de la meilleure solution courante s_G, fixé à 1.

Comme nous en avions discuter dans la section 4.3.3 une faible valeur semble préférable. Une étude plus poussée sur l'influence de ces deux paramètres reste cependant à faire.

Il est également nécessaire de choisir la méthode d'encodage des poids comme cela a été présenté dans la section 4.2.1.2. Les premières expériences ont montré que l'encodage par n valeurs réelles était le plus efficace.

On voit qu'au final, seul le nombre de classes doit être configuré en fonction des données traitées, tous les autres paramètres pouvant être fixés à des valeurs par défaut.

Les méthodes testées étant toutes non déterministes, toutes les valeurs présentées correspondent à une moyenne sur 100 exécutions (l'écart type est précisé entre parenthèses).

4.4.3 Comparaison selon la fonction d'évaluation

Nous allons tout d'abord comparer les algorithmes selon leur capacité à minimiser la fonction d'évaluation, afin de s'assurer que les méthodes évolutionnaires sont plus efficaces que les méthodes classiques, et de vérifier si les méthodes lamarckiennes et baldwiniennes sont plus performantes que les méthodes darwiniennes pour ce critère.

4.4.3.1 Comparaison des méthodes de pondération globale

Sur la table 4.2 sont présentées les valeurs finales de la fonction $cost_{gaw}$ (c'est-à-dire de l'erreur en carré pondérée de façon globale) des différents algorithmes avec pondération globale, sur les différents ensemble de données. On remarque que la méthode GAW-K-means fournit toujours un meilleur résultat que la méthode K-means.

On remarque également que tous les algorithmes génétiques ont de meilleurs résultats que K-means et GAW-K-means. Plus particulièrement, les méthodes lamarckienne et baldwinienne produisent des résultats de qualité relativement similaires et sont nettement plus efficaces que les trois autres méthodes, excepté sur l'ensemble de données IONOSPHERE où l'algorithme baldwinien n'est pas très efficace.

4.4.3.2 Comparaison des méthodes de pondération locale

Sur la table 4.3 sont présentées les valeurs finales la fonction $cost_{law}$ (c'est-à-dire de l'erreur en carré pondérée de façon locale) selon les différents algorithmes avec pondération locale, sur les différents ensembles de données. On remarque que la méthode LAW-K-means fournit toujours un meilleur résultat que la méthode K-means.

L'algorithme DE-LAW-K-means ne produit pas toujours un meilleur résultat que LAW-K-means, comme on le voit sur les ensembles DA3, DIABETES et SONAR. L'espace de recherche peut être trop grand ou la fonction d'évaluation trop complexe pour permettre une convergence rapide de l'algorithme darwinien. En revanche, les résultats de l'algorithme DC-LAW-K-means sont toujours meilleurs que ceux des algorithmes LAW-K-means et DE-LAW-K-means, excepté sur l'ensemble de données DA1.

Les méthodes lamarckiennes et baldwiniennes sont toujours meilleures que les méthodes darwiniennes, excepté sur l'ensemble DA1. Les algorithmes lamarckiens sont toujours aussi bons, sinon meilleurs que les algorithmes baldwiniens. Il n'y a que peu de différences entre les approches évolutionnaires et coévolutionnaires pour les modèles d'évolution lamarckien et baldwinien : les méthodes coévolutionnaires sont moins efficaces pour les ensembles de données DA1 et DIABETES, mais meilleures pour les ensembles de données IONOSPHERE et SONAR.

Algorithme	$cost_{gaw}$
DA1	
K-means	97,92 (1,35)
GAW-K-means	37,45 (31,06)
DE-GAW-K-means	26,26 (16,35)
LE-GAW-K-means	5,10 (0,00)
BE-GAW-K-means	5,10 (0,00)
DA2	
K-means	17,37 (7,39)
GAW-K-means	10,65 (1,92)
DE-GAW-K-means	9,75 (2,07)
LE-GAW-K-means	7,64 (0,00)
BE-GAW-K-means	7,64 (0,00)
DA3	
K-means	74,82 (7,29)
GAW-K-means	50,67 (11,33)
DE-GAW-K-means	49,03 (2,86)
LE-GAW-K-means	42,07 (0,48)
BE-GAW-K-means	42,22 (0,52)
IRIS	
K-means	9,84 (1,38)
GAW-K-means	5,01 (2,52)
DE-GAW-K-means	4,10 (0,12)
LE-GAW-K-means	3,78 (0,00)
BE-GAW-K-means	3,78 (0,00)
DIABETES	
K-means	112,56 (2,07)
GAW-K-means	91,59 (8,05)
DE-GAW-K-means	88,25 (2,12)
LE-GAW-K-means	85,08 (0,00)
BE-GAW-K-means	85,08 (0,00)
IONOSPHERE	
K-means	15,89 (0,85)
GAW-K-means	12,77 (2,50)
DE-GAW-K-means	12,91 (0,00)
LE-GAW-K-means	0,00 (0,00)
BE-GAW-K-means	11,49 (4,04)
SONAR	
K-means	5,83 (0,04)
GAW-K-means	5,40 (0,23)
DE-GAW-K-means	5,37 (0,06)
LE-GAW-K-means	5,11 (0,00)
BE-GAW-K-means	5,21 (0,02)

TAB. 4.2 : *Évaluation des algorithmes de pondération globale selon fonction $cost_{gaw}$*

4.4.3.3 Conclusion sur l'optimisation de la fonction de coût

Les résultats expérimentaux ont montré une grande efficacité des algorithmes proposés pour l'optimisation de la fonction de coût, que ce soit dans le cadre de la pondération globale ou de la pondération locale. Les approches hybrides, et particulièrement les algorithmes lamarckien se sont montrés très performants, obtenant un résultat toujours largement inférieur à celui de GAW-K-means et LAW-K-means. Les résultats ont également mis en évidence l'efficacité de la coévolution coopération par rapport l'évolution classique dans le cas du modèle d'évolution darwinien. Les résultats de l'approche coévolutionnaires sont plus mitigés dans le cas des approches lamarckienne et baldwinienne.

4.4.4 Comparaison selon des critères externes

La classe réelle de chaque objet est connue pour tous les ensembles de données présentés dans la section 4.4.1. Les résultats des différents algorithmes peuvent donc être comparés à la classification réelle des données, afin de vérifier la pertinences des classes obtenues. Pour évaluer les algorithmes, nous utilisons différents critères de comparaison de résultats de classification. Ce type d'évaluation est souvent appelée évaluation par critères externes.

Cette comparaison n'est pas aisée, étant donné qu'il n'est pas possible de faire directement la correspondance d'une classe du résultat de la classification non supervisée avec une classe réelle des

Algorithme	$cost_{law}$
DA1	
K-means	97,92 (1,35)
LAW-K-means	47,46 (14,20)
DE-LAW-K-means	7,61 (9,47)
LE-LAW-K-means	5,10 (0,00)
BE-LAW-K-means	5,10 (0,00)
DC-LAW-K-means	12,36 (16,04)
LC-LAW-K-means	13,64 (17,08)
BC-LAW-K-means	14,07 (17,39)
DA2	
K-means	17,37 (7,39)
LAW-K-means	10,09 (1,63)
DE-LAW-K-means	8,25 (0,97)
LE-LAW-K-means	7,56 (0,00)
BE-LAW-K-means	7,56 (0,00)
DC-LAW-K-means	7,60 (0,04)
LC-LAW-K-means	7,56 (0,00)
BC-LAW-K-means	7,56 (0,00)
DA3	
K-means	74,82 (7,29)
LAW-K-means	15,24 (12,79)
DE-LAW-K-means	23,12 (8,38)
LE-LAW-K-means	10,74 (0,00)
BE-LAW-K-means	10,74 (0,00)
DC-LAW-K-means	10,75 (0,00)
LC-LAW-K-means	10,74 (0,00)
BC-LAW-K-means	10,74 (0,00)
IRIS	
K-means	9,84 (1,38)
LAW-K-means	4,51 (1,87)
DE-LAW-K-means	3,86 (0,24)
LE-LAW-K-means	3,62 (0,00)
BE-LAW-K-means	3,62 (0,00)
DC-LAW-K-means	3,65 (0,02)
LC-LAW-K-means	3,62 (0,00)
BC-LAW-K-means	3,62 (0,00)
DIABETES	
K-means	112,56 (2,07)
LAW-K-means	54,79 (12,80)
DE-LAW-K-means	77,75 (2,18)
LE-LAW-K-means	44,63 (0,00)
BE-LAW-K-means	44,68 (0,04)
DC-LAW-K-means	49,53 (10,53)
LC-LAW-K-means	46,38 (6,41)
BC-LAW-K-means	45,63 (4,92)
IONOSPHERE	
K-means	15,89 (0,85)
LAW-K-means	9,90 (4,93)
DE-LAW-K-means	4,05 (0,03)
LE-LAW-K-means	0,00 (0,00)
BE-LAW-K-means	3,28 (1,49)
DC-LAW-K-means	0,36 (2,03)
LC-LAW-K-means	0,00 (0,00)
BC-LAW-K-means	0,00 (0,00)
SONAR	
K-means	5,83 (0,04)
LAW-K-means	4,50 (0,25)
DE-LAW-K-means	4,94 (0,07)
LE-LAW-K-means	4,46 (0,04)
BE-LAW-K-means	4,53 (0,05)
DC-LAW-K-means	4,39 (0,02)
LC-LAW-K-means	4,33 (0,00)
BC-LAW-K-means	4,33 (0,00)

TAB. 4.3 : *Évaluation des algorithmes de pondération locale selon la fonction* $cost_{law}$

données. Les méthodes couramment employées consistent à utiliser des critères de comparaison de partition.

Tous ces critères sont détaillés dans l'annexe C. Pour chacun de ces critères, une valeur élevée indique une forte ressemblance entre les classes obtenues par l'algorithme et les classes réelles des données et donc une forte pertinence des classes.

4.4.4.1 Comparaison des méthodes de pondération globale

Sur la table 4.4 on voit que les classes proposées par les méthodes de classification intégrant un processus de pondération globale des attributs ne sont pas toujours plus proches des classes réelles que celles proposées par l'algorithme K-means :

- sur les ensemble de données DA1, IRIS et IONOSPHERE, les résultats sont nettement améliorés. Le critère d'évaluation $cost_{gaw}$ est donc pertinent pour ces deux ensembles de données. Sur l'ensemble de données DA3, les résultats ne sont que très légèrement améliorés par les algorithmes avec pondération globale des attributs ;
- sur les ensembles DA2, DIABETES et SONAR, les résultats sont dégradé par rapport à l'algorithme K-means classique : les algorithmes les plus efficaces pour minimiser la fonction de coût sont ceux qui produisent les classes les plus éloignées de la réalité. Les mauvais résultats sur l'ensemble de données DA2 semble indiquer que ces algorithmes ne sont pas efficaces face aux attributs corrélés.

	Critères d'évaluation						
Algorithme	WG	R	J	FM	$F-M.$	$\overline{\Gamma}$	κ
			DA1				
K-means	0,50 (0,02)	0,77 (0,02)	0,49 (0,02)	0,66 (0,02)	0,66 (0,02)	0,49 (0,04)	0,73 (0,02)
GAW-K-means	0,54 (0,41)	0,77 (0,21)	0,57 (0,37)	0,66 (0,31)	0,65 (0,31)	0,48 (0,46)	0,72 (0,24)
DE-GAW-K-means	0,78 (0,38)	0,89 (0,19)	0,80 (0,34)	0,83 (0,29)	0,83 (0,29)	0,75 (0,43)	0,87 (0,23)
LE-GAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BE-GAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
			DA2				
K-means	0,88 (0,22)	0,93 (0,12)	0,87 (0,22)	0,92 (0,14)	0,92 (0,14)	0,87 (0,23)	0,92 (0,13)
GAW-K-means	0,76 (0,25)	0,88 (0,13)	0,75 (0,25)	0,84 (0,16)	0,84 (0,17)	0,75 (0,26)	0,86 (0,15)
DE-GAW-K-means	0,66 (0,22)	0,82 (0,12)	0,66 (0,22)	0,78 (0,14)	0,77 (0,15)	0,65 (0,23)	0,78 (0,14)
LE-GAW-K-means	0,50 (0,00)	0,72 (0,00)	0,50 (0,00)	0,68 (0,00)	0,67 (0,00)	0,47 (0,00)	0,69 (0,00)
BE-GAW-K-means	0,50 (0,00)	0,72 (0,00)	0,50 (0,00)	0,68 (0,00)	0,67 (0,00)	0,47 (0,00)	0,69 (0,00)
			DA3				
K-means	0,29 (0,02)	0,67 (0,01)	0,34 (0,02)	0,51 (0,02)	0,51 (0,02)	0,26 (0,03)	0,61 (0,01)
GAW-K-means	0,45 (0,08)	0,74 (0,03)	0,47 (0,06)	0,64 (0,06)	0,64 (0,06)	0,44 (0,08)	0,69 (0,04)
DE-GAW-K-means	0,45 (0,05)	0,73 (0,02)	0,46 (0,04)	0,64 (0,04)	0,63 (0,04)	0,43 (0,05)	0,69 (0,03)
LE-GAW-K-means	0,49 (0,02)	0,75 (0,01)	0,51 (0,02)	0,68 (0,02)	0,67 (0,02)	0,49 (0,02)	0,72 (0,01)
BE-GAW-K-means	0,49 (0,02)	0,75 (0,01)	0,50 (0,02)	0,67 (0,02)	0,67 (0,02)	0,48 (0,02)	0,72 (0,01)
			IRIS				
K-means	0,62 (0,06)	0,81 (0,04)	0,57 (0,04)	0,73 (0,03)	0,72 (0,04)	0,58 (0,06)	0,58 (0,07)
GAW-K-means	0,81 (0,17)	0,90 (0,09)	0,78 (0,15)	0,87 (0,11)	0,87 (0,11)	0,80 (0,18)	0,79 (0,19)
DE-GAW-K-means	0,88 (0,02)	0,95 (0,01)	0,85 (0,03)	0,92 (0,02)	0,92 (0,02)	0,88 (0,03)	0,88 (0,03)
LE-GAW-K-means	0,89 (0,00)	0,95 (0,00)	0,86 (0,00)	0,92 (0,00)	0,92 (0,00)	0,89 (0,00)	0,89 (0,00)
BE-GAW-K-means	0,89 (0,00)	0,95 (0,00)	0,86 (0,00)	0,92 (0,00)	0,92 (0,00)	0,89 (0,00)	0,89 (0,00)
			DIABETES				
K-means	0,40 (0,03)	0,56 (0,01)	0,43 (0,03)	0,60 (0,03)	0,60 (0,02)	0,12 (0,04)	0,36 (0,08)
GAW-K-means	0,35 (0,07)	0,54 (0,03)	0,41 (0,04)	0,58 (0,05)	0,58 (0,04)	0,06 (0,06)	0,32 (0,07)
DE-GAW-K-means	0,34 (0,05)	0,53 (0,02)	0,39 (0,02)	0,56 (0,03)	0,56 (0,03)	0,06 (0,04)	0,34 (0,01)
LE-GAW-K-means	0,25 (0,00)	0,50 (0,00)	0,36 (0,00)	0,53 (0,00)	0,53 (0,00)	-0,00 (0,00)	0,28 (0,00)
BE-GAW-K-means	0,25 (0,00)	0,50 (0,00)	0,36 (0,00)	0,53 (0,00)	0,53 (0,00)	-0,00 (0,00)	0,28 (0,00)
			IONOSPHERE				
K-means	0,43 (0,03)	0,59 (0,02)	0,46 (0,05)	0,63 (0,05)	0,62 (0,05)	0,18 (0,04)	0,17 (0,02)
GAW-K-means	0,42 (0,02)	0,59 (0,01)	0,44 (0,03)	0,61 (0,03)	0,61 (0,03)	0,17 (0,04)	0,17 (0,04)
DE-GAW-K-means	0,42 (0,00)	0,59 (0,00)	0,43 (0,00)	0,60 (0,00)	0,60 (0,00)	0,17 (0,00)	0,17 (0,00)
LE-GAW-K-means	0,50 (0,00)	0,62 (0,00)	0,56 (0,00)	0,73 (0,00)	0,72 (0,00)	0,25 (0,00)	0,21 (0,00)
BE-GAW-K-means	0,43 (0,03)	0,59 (0,01)	0,45 (0,04)	0,62 (0,04)	0,62 (0,04)	0,18 (0,02)	0,18 (0,01)
			SONAR				
K-means	0,30 (0,01)	0,51 (0,01)	0,37 (0,02)	0,55 (0,03)	0,54 (0,02)	0,02 (0,02)	0,02 (0,02)
GAW-K-means	0,28 (0,03)	0,50 (0,01)	0,37 (0,03)	0,54 (0,03)	0,54 (0,03)	0,01 (0,02)	0,01 (0,02)
DE-GAW-K-means	0,27 (0,01)	0,50 (0,00)	0,34 (0,00)	0,51 (0,00)	0,51 (0,00)	-0,00 (0,01)	-0,00 (0,01)
LE-GAW-K-means	0,27 (0,00)	0,50 (0,00)	0,35 (0,00)	0,51 (0,00)	0,51 (0,00)	-0,00 (0,00)	-0,00 (0,00)
BE-GAW-K-means	0,26 (0,01)	0,50 (0,00)	0,35 (0,00)	0,52 (0,00)	0,52 (0,00)	-0,00 (0,00)	-0,00 (0,00)

TAB. 4.4 : *Évaluation des algorithmes de pondération globale par critères externes*

4.4.4.2 Comparaison des méthodes de pondération locale

On voit sur la table 4.5, que la pertinence des classes découvertes par les divers algorithmes varient de la même façon que pour les méthodes de pondérations globales. Les résultats sont

nettement améliorés lorsque la fonction d'évaluation est minimisée pour les ensembles de données DA1, IRIS et IONOSPHERE, et sont dégradés pour les ensemble de données DA2, DIABETES et SONAR.

Les résultats sont cependant nettement supérieurs sur l'ensemble de données DA3, ce qui confirme l'intérêt de l'utilisation des algorithmes de pondération locale. En effet, cet ensemble a été construit de telle sorte que tous les attributs soient pertinents, mais pas tous pour les mêmes classes.

Algorithme	WG	R	J	FM	$F-M.$	$\overline{\Gamma}$	κ
			DA1				
K-means	0,50 (0,02)	0,77 (0,02)	0,49 (0,02)	0,66 (0,02)	0,66 (0,02)	0,49 (0,04)	0,73 (0,02)
LAW-K-means	0,51 (0,20)	0,78 (0,10)	0,52 (0,18)	0,67 (0,15)	0,67 (0,15)	0,50 (0,22)	0,74 (0,12)
DE-LAW-K-means	0,96 (0,17)	0,98 (0,09)	0,97 (0,16)	0,97 (0,13)	0,97 (0,13)	0,96 (0,20)	0,98 (0,10)
LE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
DC-LAW-K-means	0,85 (0,33)	0,92 (0,17)	0,86 (0,30)	0,89 (0,25)	0,89 (0,25)	0,83 (0,38)	0,91 (0,20)
LC-LAW-K-means	0,82 (0,35)	0,91 (0,18)	0,84 (0,32)	0,87 (0,27)	0,87 (0,27)	0,80 (0,40)	0,89 (0,21)
BC-LAW-K-means	0,81 (0,36)	0,91 (0,18)	0,83 (0,33)	0,86 (0,27)	0,86 (0,27)	0,79 (0,41)	0,89 (0,21)
			DA2				
K-means	0,88 (0,22)	0,93 (0,12)	0,87 (0,22)	0,92 (0,14)	0,92 (0,14)	0,87 (0,23)	0,92 (0,13)
LAW-K-means	0,62 (0,21)	0,81 (0,11)	0,62 (0,21)	0,75 (0,14)	0,75 (0,14)	0,61 (0,21)	0,78 (0,12)
DE-LAW-K-means	0,51 (0,03)	0,74 (0,02)	0,50 (0,01)	0,68 (0,01)	0,67 (0,01)	0,48 (0,02)	0,70 (0,02)
LE-LAW-K-means	0,50 (0,00)	0,72 (0,00)	0,50 (0,00)	0,68 (0,00)	0,67 (0,00)	0,47 (0,00)	0,69 (0,00)
BE-LAW-K-means	0,50 (0,00)	0,72 (0,00)	0,50 (0,00)	0,68 (0,00)	0,67 (0,00)	0,47 (0,00)	0,69 (0,00)
DC-LAW-K-means	0,50 (0,02)	0,72 (0,00)	0,50 (0,00)	0,68 (0,00)	0,67 (0,00)	0,47 (0,00)	0,68 (0,02)
LC-LAW-K-means	0,50 (0,00)	0,72 (0,00)	0,50 (0,00)	0,68 (0,00)	0,67 (0,00)	0,47 (0,00)	0,69 (0,00)
BC-LAW-K-means	0,50 (0,00)	0,72 (0,00)	0,50 (0,00)	0,68 (0,00)	0,67 (0,00)	0,47 (0,00)	0,69 (0,00)
			DA3				
K-means	0,29 (0,02)	0,67 (0,01)	0,34 (0,02)	0,51 (0,02)	0,51 (0,02)	0,26 (0,03)	0,61 (0,01)
LAW-K-means	0,92 (0,22)	0,96 (0,10)	0,93 (0,20)	0,95 (0,15)	0,95 (0,15)	0,92 (0,23)	0,96 (0,12)
DE-LAW-K-means	0,76 (0,15)	0,89 (0,07)	0,73 (0,15)	0,84 (0,10)	0,83 (0,10)	0,75 (0,16)	0,87 (0,08)
LE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
DC-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
LC-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BC-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
			IRIS				
K-means	0,62 (0,06)	0,81 (0,04)	0,57 (0,04)	0,73 (0,03)	0,72 (0,04)	0,58 (0,06)	0,58 (0,07)
LAW-K-means	0,82 (0,15)	0,91 (0,08)	0,79 (0,14)	0,88 (0,09)	0,88 (0,10)	0,81 (0,15)	0,81 (0,16)
DE-LAW-K-means	0,87 (0,02)	0,94 (0,01)	0,83 (0,03)	0,91 (0,02)	0,91 (0,02)	0,86 (0,03)	0,86 (0,03)
LE-LAW-K-means	0,89 (0,00)	0,95 (0,00)	0,86 (0,00)	0,92 (0,00)	0,92 (0,00)	0,89 (0,00)	0,89 (0,00)
BE-LAW-K-means	0,89 (0,00)	0,95 (0,00)	0,86 (0,00)	0,92 (0,00)	0,92 (0,00)	0,89 (0,00)	0,89 (0,00)
DC-LAW-K-means	0,89 (0,01)	0,95 (0,00)	0,85 (0,01)	0,92 (0,01)	0,92 (0,01)	0,88 (0,01)	0,88 (0,01)
LC-LAW-K-means	0,89 (0,00)	0,95 (0,00)	0,86 (0,00)	0,92 (0,00)	0,92 (0,00)	0,89 (0,00)	0,89 (0,00)
BC-LAW-K-means	0,89 (0,00)	0,95 (0,00)	0,86 (0,00)	0,92 (0,00)	0,92 (0,00)	0,89 (0,00)	0,89 (0,00)
			DIABETES				
K-means	0,40 (0,03)	0,56 (0,01)	0,43 (0,03)	0,60 (0,03)	0,60 (0,02)	0,12 (0,04)	0,36 (0,08)
LAW-K-means	0,30 (0,05)	0,52 (0,02)	0,37 (0,02)	0,54 (0,02)	0,54 (0,02)	0,03 (0,04)	0,34 (0,02)
DE-LAW-K-means	0,29 (0,01)	0,51 (0,00)	0,36 (0,00)	0,53 (0,00)	0,53 (0,00)	0,01 (0,01)	0,34 (0,01)
LE-LAW-K-means	0,26 (0,00)	0,50 (0,00)	0,35 (0,00)	0,52 (0,00)	0,52 (0,00)	-0,00 (0,00)	0,33 (0,00)
BE-LAW-K-means	0,26 (0,00)	0,50 (0,00)	0,35 (0,00)	0,52 (0,00)	0,52 (0,00)	-0,00 (0,00)	0,33 (0,00)
DC-LAW-K-means	0,28 (0,03)	0,51 (0,02)	0,36 (0,02)	0,53 (0,02)	0,53 (0,02)	0,01 (0,03)	0,34 (0,02)
LC-LAW-K-means	0,27 (0,03)	0,50 (0,01)	0,36 (0,01)	0,52 (0,01)	0,52 (0,01)	0,01 (0,02)	0,33 (0,01)
BC-LAW-K-means	0,27 (0,02)	0,50 (0,01)	0,35 (0,01)	0,52 (0,01)	0,52 (0,01)	0,00 (0,02)	0,33 (0,00)
			IONOSPHERE				
K-means	0,43 (0,03)	0,59 (0,02)	0,46 (0,05)	0,63 (0,05)	0,62 (0,05)	0,18 (0,04)	0,17 (0,02)
LAW-K-means	0,41 (0,05)	0,57 (0,03)	0,45 (0,07)	0,62 (0,07)	0,61 (0,06)	0,14 (0,07)	0,13 (0,06)
DE-LAW-K-means	0,50 (0,00)	0,62 (0,00)	0,56 (0,00)	0,73 (0,00)	0,72 (0,00)	0,25 (0,00)	0,21 (0,00)
LE-LAW-K-means	0,50 (0,00)	0,62 (0,00)	0,56 (0,00)	0,73 (0,00)	0,72 (0,00)	0,25 (0,00)	0,21 (0,00)
BE-LAW-K-means	0,50 (0,00)	0,62 (0,00)	0,56 (0,00)	0,73 (0,00)	0,72 (0,00)	0,25 (0,00)	0,21 (0,00)
DC-LAW-K-means	0,50 (0,02)	0,62 (0,01)	0,56 (0,03)	0,73 (0,03)	0,72 (0,02)	0,25 (0,02)	0,21 (0,02)
LC-LAW-K-means	0,50 (0,00)	0,62 (0,00)	0,56 (0,00)	0,73 (0,00)	0,72 (0,00)	0,25 (0,00)	0,21 (0,00)
BC-LAW-K-means	0,50 (0,00)	0,62 (0,00)	0,56 (0,00)	0,73 (0,00)	0,72 (0,00)	0,25 (0,00)	0,21 (0,00)
			SONAR				
K-means	0,30 (0,01)	0,51 (0,01)	0,37 (0,02)	0,55 (0,03)	0,54 (0,02)	0,02 (0,02)	0,02 (0,02)
LAW-K-means	0,28 (0,02)	0,50 (0,01)	0,34 (0,01)	0,51 (0,01)	0,51 (0,01)	0,01 (0,01)	0,01 (0,01)
DE-LAW-K-means	0,29 (0,02)	0,51 (0,01)	0,35 (0,02)	0,52 (0,02)	0,52 (0,02)	0,01 (0,02)	0,01 (0,02)
LE-LAW-K-means	0,30 (0,02)	0,51 (0,01)	0,34 (0,01)	0,51 (0,01)	0,51 (0,01)	0,02 (0,02)	0,02 (0,02)
BE-LAW-K-means	0,27 (0,01)	0,50 (0,00)	0,34 (0,01)	0,50 (0,01)	0,50 (0,01)	-0,00 (0,00)	-0,00 (0,00)
DC-LAW-K-means	0,26 (0,01)	0,50 (0,00)	0,33 (0,00)	0,50 (0,00)	0,50 (0,00)	-0,00 (0,00)	-0,00 (0,00)
LC-LAW-K-means	0,27 (0,00)	0,50 (0,00)	0,33 (0,00)	0,50 (0,00)	0,50 (0,00)	-0,00 (0,00)	-0,00 (0,00)
BC-LAW-K-means	0,27 (0,00)	0,50 (0,00)	0,33 (0,00)	0,50 (0,00)	0,50 (0,00)	-0,00 (0,00)	-0,00 (0,00)

TAB. 4.5 : *Évaluation des algorithmes de pondération locale par critères externes*

4.4.4.3 Conclusion sur l'efficacité des méthodes à découvrir les classes réelles des données

Les résultats concernant l'évaluation des algorithmes par critères externes sont plus mitigés. Les algorithmes de pondération globale et locale améliore les résultats, comparativement à l'algorithme K-means, sur les ensemble de données DA1, DA3, IRIS et IONOSPHERE. Sur l'ensemble de données DA3, les méthodes de pondération locale produisent un bien meilleur résultat que les méthodes de pondération globale, ce qui montre l'importance de l'utilisation de pondérations spécifiques à chaque classe.

Les résultats sont en revanche dégradés, comparativement aux résultats de l'algorithme K-means, sur les ensembles de données DA2, DIABETES et SONAR. Ces résultats remettent en cause, non pas les méthodes d'optimisation que nous avons développées, mais le critère d'évaluation de la qualité d'une classification. En effet, nous avions déjà fait remarqué dans la section 3.5.3 que la fonction d'évaluation limitait la forme des classes dans l'espace des données à des sphères et semblait relativement sensible aux corrélations entre les attributs.

Ainsi, bien que nos algorithmes soient capables d'optimiser avec une grande efficacité la fonction de coût, celle-ci est pertinentes que dans des cas très limitées et amènera donc souvent à des résultats incohérents.

4.4.5 Stabilité des résultats

Les algorithmes génétiques sont des méthodes d'optimisation non déterministes. Les résultats obtenus par les algorithmes de classification proposés peuvent donc différer d'une exécution à l'autre. C'est le cas également pour d'autres méthodes de classification non supervisée, comme par exemple K-means où les centres des classes sont initialisés aléatoirement.

Afin d'évaluer la stabilité des résultats produits par une méthode de classification non supervisée non déterministe, nous utilisons des indices de comparaison de résultats (détaillés dans l'annexe C), afin d'évaluer la similarité entre les différents résultats. Chaque méthode est exécutée plusieurs fois, et les résultats sont comparés deux à deux. Le temps de calcul pouvant devenir excessivement long si chaque résultat était comparé à chaque autre, la moyenne est calculé en comparant chaque résultat avec le suivant. Une forte valeur indique une forte similarité entre les différents résultats.

4.4.5.1 Comparaison des méthodes de pondération globale

On voit sur la table 4.6 que l'algorithme GAW-K-means est souvent beaucoup moins stable que l'algorithme K-means pour la plupart des ensembles de données (excepté sur l'ensemble de données IRIS). Les algorithmes évolutifs sont en revanche généralement beaucoup plus stables, excepté sur l'ensemble de données DA3. Les méthodes intégrant un processus d'apprentissage au cours de la vie donnent les résultats les plus stables, et tout particulièrement l'algorithme lamarckien.

4.4.5.2 Comparaison des méthodes de pondération locale

On voit sur la table 4.7 que l'algorithme LAW-K-means est plus ou moins stable que l'algorithme K-means selon les ensemble de données.

L'algorithme DE-LAW-K-means n'est pas plus stable que l'algorithme LAW-K-means. En revanche, les résultats de l'algorithme DC-LAW-K-means sont beaucoup plus stables que ceux des algorithmes LAW-K-means et DE-LAW-K-means.

Les méthodes lamarckiennes et baldwiniennes sont toutes plus stables que les méthodes darwiniennes, les algorithmes lamarckiens étant parfois plus stables que les algorithmes baldwiniens. Il n'y a que peu de différences entre les approches évolutionnaires et coévolutionnaires pour les méthodes lamarckiennes et baldwiniennes : les méthodes coévolutionnaires sont moins stables pour l'ensemble de données DIABETES, mais plus stables pour l'ensemble de données SONAR.

	Critères d'évaluation						
Algorithme	WG	R	J	FM	$F-M.$	$\overline{\Gamma}$	κ
			DA1				
K-means	0,53 (0,35)	0,79 (0,16)	0,57 (0,31)	0,68 (0,23)	0,68 (0,23)	0,52 (0,35)	0,75 (0,18)
GAW-K-means	0,48 (0,38)	0,74 (0,19)	0,51 (0,34)	0,62 (0,28)	0,62 (0,28)	0,42 (0,43)	0,69 (0,23)
DE-GAW-K-means	0,63 (0,43)	0,81 (0,22)	0,66 (0,39)	0,72 (0,32)	0,72 (0,32)	0,58 (0,49)	0,78 (0,26)
LE-GAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BE-GAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
			DA2				
K-means	0,80 (0,24)	0,89 (0,14)	0,80 (0,25)	0,87 (0,16)	0,87 (0,16)	0,79 (0,26)	0,88 (0,15)
GAW-K-means	0,64 (0,22)	0,81 (0,12)	0,63 (0,22)	0,76 (0,15)	0,75 (0,15)	0,62 (0,23)	0,78 (0,13)
DE-GAW-K-means	0,70 (0,19)	0,85 (0,12)	0,73 (0,22)	0,83 (0,14)	0,82 (0,15)	0,72 (0,23)	0,82 (0,14)
LE-GAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BE-GAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
			DA3				
K-means	0,94 (0,16)	0,97 (0,07)	0,94 (0,15)	0,96 (0,10)	0,96 (0,10)	0,94 (0,16)	0,97 (0,09)
GAW-K-means	0,40 (0,23)	0,71 (0,13)	0,49 (0,21)	0,63 (0,16)	0,63 (0,16)	0,39 (0,28)	0,66 (0,16)
DE-GAW-K-means	0,49 (0,23)	0,77 (0,14)	0,58 (0,22)	0,71 (0,18)	0,71 (0,18)	0,51 (0,30)	0,74 (0,16)
LE-GAW-K-means	0,53 (0,36)	0,76 (0,20)	0,61 (0,32)	0,71 (0,24)	0,71 (0,24)	0,50 (0,42)	0,72 (0,23)
BE-GAW-K-means	0,47 (0,34)	0,74 (0,19)	0,57 (0,29)	0,68 (0,23)	0,68 (0,23)	0,46 (0,39)	0,70 (0,21)
			IRIS				
K-means	0,78 (0,20)	0,88 (0,12)	0,76 (0,20)	0,85 (0,13)	0,85 (0,14)	0,76 (0,22)	0,76 (0,23)
GAW-K-means	0,81 (0,24)	0,90 (0,13)	0,82 (0,23)	0,89 (0,15)	0,88 (0,16)	0,81 (0,25)	0,81 (0,26)
DE-GAW-K-means	0,95 (0,04)	0,97 (0,02)	0,93 (0,06)	0,96 (0,03)	0,96 (0,03)	0,94 (0,05)	0,94 (0,05)
LE-GAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BE-GAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
			DIABETES				
K-means	0,66 (0,30)	0,76 (0,20)	0,69 (0,25)	0,80 (0,17)	0,79 (0,18)	0,52 (0,42)	0,64 (0,34)
GAW-K-means	0,45 (0,25)	0,61 (0,18)	0,52 (0,22)	0,66 (0,16)	0,66 (0,16)	0,21 (0,36)	0,41 (0,27)
DE-GAW-K-means	0,60 (0,32)	0,73 (0,23)	0,63 (0,29)	0,74 (0,22)	0,74 (0,22)	0,45 (0,45)	0,63 (0,30)
LE-GAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BE-GAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
			IONOSPHERE				
K-means	0,76 (0,31)	0,83 (0,23)	0,80 (0,25)	0,87 (0,16)	0,87 (0,17)	0,64 (0,46)	0,64 (0,46)
GAW-K-means	0,89 (0,23)	0,92 (0,18)	0,91 (0,19)	0,94 (0,12)	0,94 (0,13)	0,84 (0,35)	0,83 (0,35)
DE-GAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,01)	1,00 (0,00)	1,00 (0,00)	1,00 (0,01)	1,00 (0,01)
LE-GAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BE-GAW-K-means	0,87 (0,25)	0,90 (0,19)	0,89 (0,21)	0,93 (0,14)	0,92 (0,15)	0,80 (0,38)	0,80 (0,38)
			SONAR				
K-means	0,59 (0,20)	0,71 (0,15)	0,62 (0,18)	0,75 (0,13)	0,75 (0,13)	0,41 (0,30)	0,41 (0,30)
GAW-K-means	0,51 (0,26)	0,65 (0,19)	0,57 (0,22)	0,70 (0,16)	0,70 (0,16)	0,29 (0,38)	0,29 (0,38)
DE-GAW-K-means	0,84 (0,07)	0,89 (0,05)	0,81 (0,09)	0,89 (0,05)	0,89 (0,05)	0,78 (0,11)	0,78 (0,11)
LE-GAW-K-means	0,99 (0,01)	0,99 (0,01)	0,99 (0,01)	0,99 (0,01)	0,99 (0,01)	0,99 (0,01)	0,99 (0,01)
BE-GAW-K-means	0,96 (0,03)	0,97 (0,02)	0,95 (0,03)	0,97 (0,02)	0,97 (0,02)	0,94 (0,04)	0,94 (0,04)

TAB. 4.6 : *Stabilité des algorithmes de pondération globale*

4.4.5.3 Conclusion sur la stabilité des résultats

Les résultats expérimentaux semblent indiquer une grande stabilité des résultats comparativement aux algorithmes par *hill-climbing*, très sensibles aux conditions initiales (initialisation aléatoire des centres des classes). L'algorithme K-means (et donc les méthodes qui en dérivent) est connu pour sa grande instabilité et des travaux ont été menés pour améliorer sa stabilité comme par exemple la méthode présentée dans [Likas *et al.*, 2003]. Une méthode évolutionnaire hybride (par approche lamarckienne par exemple) semble être une bonne solution.

4.4.6 Comparaison des pondérations

Il est intéressant de comparer quels attributs ont été mis en valeur par les différents algorithmes de classification avec sélection/pondération d'attributs, afin de vérifier si les algorithmes de sélection/pondération d'attributs mettent en évidence les attributs effectivement pertinents.

Dans le cas de la sélection/pondération globale d'attributs, il est trivial de comparer différents sous-ensembles d'attributs et différents vecteurs de poids. Dans le cas de la sélection/pondération locale pour la classification supervisée, il est également évident de comparer différents résultats, en les comparant classe par classe. La comparaison des pondérations est cependant plus complexe, dans le cas de la sélection/pondération locale d'attributs, car il n'est pas possible de faire la correspondance entre les classes.

Algorithme	WG	R	J	FM	$F-M.$	$\overline{\Gamma}$	κ
			DA1				
K-means	0,53 (0,35)	0,79 (0,16)	0,57 (0,31)	0,68 (0,23)	0,68 (0,23)	0,52 (0,35)	0,75 (0,18)
LAW-K-means	0,56 (0,34)	0,80 (0,16)	0,59 (0,31)	0,69 (0,24)	0,69 (0,24)	0,54 (0,35)	0,76 (0,19)
DE-LAW-K-means	0,93 (0,24)	0,96 (0,12)	0,94 (0,22)	0,95 (0,18)	0,95 (0,18)	0,92 (0,27)	0,96 (0,14)
LE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
DC-LAW-K-means	0,74 (0,40)	0,87 (0,20)	0,76 (0,37)	0,80 (0,31)	0,80 (0,31)	0,70 (0,46)	0,84 (0,24)
LC-LAW-K-means	0,75 (0,40)	0,88 (0,20)	0,78 (0,36)	0,81 (0,30)	0,81 (0,30)	0,72 (0,45)	0,85 (0,24)
BC-LAW-K-means	0,68 (0,42)	0,84 (0,21)	0,71 (0,38)	0,76 (0,32)	0,76 (0,32)	0,64 (0,48)	0,81 (0,25)
			DA2				
K-means	0,80 (0,24)	0,89 (0,14)	0,80 (0,25)	0,87 (0,16)	0,87 (0,16)	0,79 (0,26)	0,88 (0,15)
LAW-K-means	0,66 (0,21)	0,84 (0,11)	0,70 (0,21)	0,81 (0,14)	0,80 (0,14)	0,69 (0,22)	0,82 (0,13)
DE-LAW-K-means	0,77 (0,20)	0,90 (0,12)	0,82 (0,22)	0,89 (0,14)	0,88 (0,15)	0,81 (0,23)	0,88 (0,14)
LE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
DC-LAW-K-means	0,93 (0,07)	0,99 (0,01)	0,98 (0,02)	0,99 (0,01)	0,99 (0,01)	0,98 (0,02)	0,99 (0,01)
LC-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BC-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
			DA3				
K-means	0,94 (0,16)	0,97 (0,07)	0,94 (0,15)	0,96 (0,10)	0,96 (0,10)	0,94 (0,16)	0,97 (0,09)
LAW-K-means	0,86 (0,28)	0,94 (0,13)	0,87 (0,26)	0,90 (0,19)	0,90 (0,19)	0,85 (0,29)	0,92 (0,15)
DE-LAW-K-means	0,64 (0,16)	0,82 (0,09)	0,61 (0,17)	0,75 (0,12)	0,75 (0,12)	0,61 (0,19)	0,79 (0,10)
LE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
DC-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
LC-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BC-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
			IRIS				
K-means	0,78 (0,20)	0,88 (0,12)	0,76 (0,20)	0,85 (0,13)	0,85 (0,14)	0,76 (0,22)	0,76 (0,23)
LAW-K-means	0,87 (0,21)	0,94 (0,11)	0,88 (0,20)	0,92 (0,13)	0,92 (0,13)	0,88 (0,21)	0,87 (0,22)
DE-LAW-K-means	0,95 (0,05)	0,98 (0,03)	0,93 (0,06)	0,96 (0,04)	0,96 (0,04)	0,95 (0,06)	0,95 (0,06)
LE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
DC-LAW-K-means	0,98 (0,02)	0,99 (0,01)	0,97 (0,02)	0,99 (0,01)	0,99 (0,01)	0,98 (0,02)	0,98 (0,02)
LC-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BC-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
			DIABETES				
K-means	0,66 (0,30)	0,76 (0,20)	0,69 (0,25)	0,80 (0,17)	0,79 (0,18)	0,52 (0,42)	0,64 (0,34)
LAW-K-means	0,65 (0,34)	0,76 (0,24)	0,68 (0,32)	0,76 (0,24)	0,76 (0,24)	0,52 (0,48)	0,68 (0,32)
DE-LAW-K-means	0,63 (0,14)	0,74 (0,10)	0,60 (0,13)	0,74 (0,10)	0,74 (0,10)	0,48 (0,20)	0,65 (0,13)
LE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BE-LAW-K-means	0,99 (0,00)	1,00 (0,00)	0,99 (0,01)	1,00 (0,00)	1,00 (0,00)	0,99 (0,01)	1,00 (0,00)
DC-LAW-K-means	0,82 (0,31)	0,87 (0,21)	0,83 (0,28)	0,87 (0,21)	0,87 (0,21)	0,75 (0,42)	0,83 (0,28)
LC-LAW-K-means	0,90 (0,24)	0,93 (0,17)	0,91 (0,23)	0,93 (0,17)	0,93 (0,17)	0,86 (0,34)	0,91 (0,23)
BC-LAW-K-means	0,94 (0,19)	0,96 (0,13)	0,95 (0,18)	0,96 (0,13)	0,96 (0,13)	0,92 (0,26)	0,95 (0,18)
			IONOSPHERE				
K-means	0,76 (0,31)	0,83 (0,23)	0,80 (0,25)	0,87 (0,16)	0,87 (0,17)	0,64 (0,46)	0,64 (0,46)
LAW-K-means	0,56 (0,29)	0,68 (0,21)	0,62 (0,23)	0,75 (0,17)	0,74 (0,17)	0,34 (0,42)	0,33 (0,42)
DE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
LE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BE-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
DC-LAW-K-means	0,96 (0,16)	0,97 (0,12)	0,97 (0,13)	0,98 (0,09)	0,98 (0,09)	0,94 (0,23)	0,94 (0,23)
LC-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BC-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
			SONAR				
K-means	0,59 (0,20)	0,71 (0,15)	0,62 (0,18)	0,75 (0,13)	0,75 (0,13)	0,41 (0,30)	0,41 (0,30)
LAW-K-means	0,76 (0,25)	0,83 (0,18)	0,75 (0,23)	0,84 (0,17)	0,84 (0,17)	0,66 (0,36)	0,66 (0,36)
DE-LAW-K-means	0,63 (0,09)	0,74 (0,07)	0,60 (0,09)	0,75 (0,07)	0,75 (0,07)	0,47 (0,14)	0,47 (0,14)
LE-LAW-K-means	0,82 (0,08)	0,87 (0,06)	0,78 (0,09)	0,87 (0,06)	0,87 (0,06)	0,74 (0,12)	0,74 (0,12)
BE-LAW-K-means	0,86 (0,12)	0,90 (0,10)	0,84 (0,13)	0,90 (0,09)	0,90 (0,09)	0,80 (0,19)	0,80 (0,19)
DC-LAW-K-means	0,94 (0,04)	0,96 (0,03)	0,93 (0,05)	0,96 (0,03)	0,96 (0,03)	0,92 (0,05)	0,92 (0,05)
LC-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
BC-LAW-K-means	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)

TAB. 4.7 : *Stabilité des algorithmes de pondération locale*

Afin de pouvoir comparer les pondérations locales obtenues, nous définissons un degré d'utilisation des attributs, dépendant des pondérations locales.

DÉFINITION 4.1 (DEGRÉ D'UTILISATION D'UN ATTRIBUT)
Soit $W = (w_1, \ldots, w_n)$ un vecteur global de poids. On défini le degré d'utilisation *par $U_j = w_j$.*

Soit $W = \{W_1, \ldots, W_K\}$ *un vecteur de pondérations locales, avec* $W_k = (w_{k,1}, \ldots, w_{k,n})$ *et* $\sum_{j=1}^{n} w_{k,j} = 1$. *On défini le* degré d'utilisation *de l'attribut* F_j *par* $U_j = \max_{k=1}^{K} w_{k,j}$.

De façon évidente, le degré d'utilisation d'un attribut F_j dans une classification est élevé s'il existe une classe C_k pour lequel le poids $w_{k,j}$ est élevé par rapport aux poids des autres attributs.

Exemple :

> *Soient un ensemble d'attributs* $F = \{F_1, \ldots, F_6\}$ *et les sous-ensembles d'attributs locaux* $F^1 = \{F_1, F_2\}$ *et* $F^2 = \{F_2, F_3, F_4, F_5\}$ *correspondant à deux classes différentes d'une même classification. Les degrés d'utilisation des attributs sont alors* $U_1 = 1/2$, $U_2 = 1/2$, $U_3 = 1/4$, $U_4 = 1/4$, $U_5 = 1/4$ *et* $U_6 = 0$.

Le degré d'utilisation d'un attribut exprime à quel point celui-ci est pris en compte par un algorithme de classification, relativement aux autres attributs. Le degré d'utilisation d'un attribut F_j étant relatif à ceux des autres attributs, les vecteurs $(c \times U_1, \ldots, c \times U_n)$ sont tous équivalents entre eux quelque soit $c > 0$.

Les algorithmes étant souvent non déterministes, il est nécessaire de comparer des moyennes entre plusieurs résultats, ceux-ci pouvant différer d'une exécution à l'autre. Un calcul de moyenne entre différents vecteurs de degrés d'utilisation se fera en normalisant ces vecteurs de sorte qu'ils aient tous la même norme.

Afin de facilité la lecture d'un vecteur de degrés d'utilisation, celui sera normalisé de sorte que le degré maximum soit égal à 1.

4.4.6.1 Comparaison des méthodes de pondération globale

Sur la table 4.8 est représenté le degré d'utilisation des attributs par les algorithmes de pondération globale pour l'ensemble de données DA1. On constate que les méthodes les plus efficaces ne donnent d'importance qu'à l'attribut 2 qui est en effet le seul attribut pertinent.

	Attributs	
Algorithme	attribut 1	attribut 2
GAW-K-means	0,87 (0,86)	1,00 (0,89)
DE-GAW-K-means	0,37 (0,55)	1,00 (0,55)
LE-GAW-K-means	0,00 (0,00)	1,00 (0,00)
BE-GAW-K-means	0,00 (0,00)	1,00 (0,00)

TAB. 4.8 : *Degré d'utilisation des attributs par les algorithmes de pondération globale pour l'ensemble de données* DA1

Sur la table 4.9 est représenté le degré d'utilisation des attributs par les algorithmes de pondération globale pour l'ensemble de données DA2. On constate que toutes les méthodes donnent une grande importance à chacun des attributs corrélés, mais ne prennent pas en compte les valeurs sur l'attribut 1 qui est pourtant indispensable pour bien discriminer les classes 1 et 2. Cela peut expliquer les mauvais résultats obtenues par ces méthodes.

	Attributs				
Algorithme	attribut 1	attribut 2	attribut 3	attribut 4	attribut 5
GAW-K-means	0,12 (0,12)	1,00 (0,02)	0,95 (0,00)	0,95 (0,01)	0,96 (0,02)
DE-GAW-K-means	0,07 (0,11)	1,00 (0,00)	0,97 (0,01)	0,96 (0,01)	0,97 (0,01)
LE-GAW-K-means	0,00 (0,00)	1,00 (0,00)	0,92 (0,00)	0,94 (0,00)	0,91 (0,00)
BE-GAW-K-means	0,00 (0,00)	1,00 (0,00)	0,92 (0,00)	0,94 (0,00)	0,91 (0,00)

TAB. 4.9 : *Degré d'utilisation des attributs par les algorithmes de pondération globale pour l'ensemble de données* DA2

Sur la table 4.10 est représenté le degré d'utilisation des attributs par les algorithmes de pondération globale pour l'ensemble de données DA3. On constate que les méthodes les plus efficaces donnent un maximum d'importance à l'attribut 2, une importance moyenne à l'attribut 1 et ne tiennent pas compte de l'attribut 3. Or pour cet ensemble de données, tous les attributs sont pertinents, mais pas tous pour la même classe.

	Attributs		
Algorithme	attribut 1	attribut 2	attribut 3
GAW-K-means	0,71 (0,84)	1,00 (0,89)	0,57 (0,79)
DE-GAW-K-means	0,19 (0,40)	1,00 (0,46)	0,09 (0,26)
LE-GAW-K-means	0,79 (0,87)	1,00 (0,88)	0,01 (0,00)
BE-GAW-K-means	0,48 (0,68)	1,00 (0,68)	0,01 (0,00)

TAB. 4.10 : *Degré d'utilisation des attributs par les algorithmes de pondération globale pour l'ensemble de données* DA3

Sur la table 4.11 est représenté le degré d'utilisation des attributs par les algorithmes de pondération globale pour l'ensemble de données IRIS. On constate que toutes les méthodes donnent plus d'importance aux deux derniers attributs qui, comme nous l'avons vu dans la section 4.4.1.2 sont bien les plus pertinents.

	Attributs			
Algorithme	sepallength	sepalwidth	petallength	petalwidth
GAW-K-means	0,04 (0,04)	0,03 (0,14)	1,00 (0,17)	0,87 (0,20)
DE-GAW-K-means	0,05 (0,01)	0,02 (0,00)	1,00 (0,06)	0,69 (0,10)
LE-GAW-K-means	0,02 (0,00)	0,01 (0,00)	0,83 (0,00)	1,00 (0,00)
BE-GAW-K-means	0,02 (0,00)	0,01 (0,00)	0,83 (0,00)	1,00 (0,00)

TAB. 4.11 : *Degré d'utilisation des attributs par les algorithmes de pondération globale pour l'ensemble de données* IRIS

Ces résultats montrent que ces méthodes sont capables de découvrir les attributs pertinents pour la classification des données. En revanche, des problèmes se posent lorsque de trop fortes corrélations sont présentes. De plus, comme nous pouvions nous y attendre, lorsque les attributs pertinents sont différents d'une classe à l'autre, les méthodes globales sont inefficaces.

4.4.6.2 Comparaison des méthodes de pondération locale

Sur la table 4.12 est représenté le degré d'utilisation des attributs par les algorithmes de pondération locale pour l'ensemble de données DA1. On constate que, comme dans le cas des méthodes de pondération globale, les méthodes les plus efficaces n'utilisent que l'attribut 2.

	Attributs	
Algorithme	attribut 1	attribut 2
LAW-K-means	0,85 (0,32)	1,00 (0,34)
DE-LAW-K-means	0,05 (0,21)	1,00 (0,20)
LE-LAW-K-means	0,00 (0,00)	1,00 (0,00)
BE-LAW-K-means	0,00 (0,00)	1,00 (0,00)
DC-LAW-K-means	0,20 (0,45)	1,00 (0,45)
LC-LAW-K-means	0,25 (0,50)	1,00 (0,49)
BC-LAW-K-means	0,27 (0,51)	1,00 (0,51)

TAB. 4.12 : *Degré d'utilisation des attributs par les algorithmes de pondération locale pour l'ensemble de données* DA1

Sur la table 4.13 est représenté le degré d'utilisation des attributs par les algorithmes de pondération locale pour l'ensemble de données DA2. Les résultats ne sont pas meilleurs que ceux obtenus par les méthodes de pondération globale, seuls les quatre attributs corrélés entre eux sont pris en compte, les algorithmes ne tenant pas compte de l'attribut 1.

Sur la table 4.14 est représenté le degré d'utilisation des attributs par les algorithmes de pondération locale pour l'ensemble de données DA3. On voit que tous les attributs sont utilisés

	Attributs				
Algorithme	attribut 1	attribut 2	attribut 3	attribut 4	attribut 5
LAW-K-means	0,58 (0,71)	1,00 (0,25)	0,92 (0,22)	0,90 (0,22)	0,90 (0,22)
DE-LAW-K-means	0,14 (0,11)	1,00 (0,05)	0,95 (0,05)	0,94 (0,06)	0,95 (0,08)
LE-LAW-K-means	0,03 (0,00)	1,00 (0,00)	0,94 (0,00)	0,97 (0,00)	0,98 (0,00)
BE-LAW-K-means	0,03 (0,00)	1,00 (0,00)	0,94 (0,00)	0,97 (0,00)	0,98 (0,00)
DC-LAW-K-means	0,02 (0,00)	0,99 (0,01)	0,92 (0,01)	1,00 (0,03)	0,99 (0,04)
LC-LAW-K-means	0,03 (0,00)	1,00 (0,00)	0,94 (0,00)	0,97 (0,00)	0,98 (0,00)
BC-LAW-K-means	0,03 (0,00)	1,00 (0,00)	0,94 (0,00)	0,97 (0,00)	0,98 (0,00)

TAB. 4.13 : *Degré d'utilisation des attributs par les algorithmes de pondération locale pour l'ensemble de données* DA2

pour la classification. En effet, pour cet ensemble de données, tous les attributs sont pertinents mais pas tous pour la même classe, ce qui explique les résultats obtenus par les algorithmes de pondération locale.

	Attributs		
Algorithme	attribut 1	attribut 2	attribut 3
LAW-K-means	1,00 (0,11)	0,93 (0,11)	0,94 (0,08)
DE-LAW-K-means	1,00 (0,45)	0,83 (0,47)	0,96 (0,43)
LE-LAW-K-means	1,00 (0,00)	0,91 (0,00)	0,92 (0,00)
BE-LAW-K-means	1,00 (0,00)	0,91 (0,00)	0,92 (0,00)
DC-LAW-K-means	1,00 (0,01)	0,86 (0,02)	0,93 (0,01)
LC-LAW-K-means	1,00 (0,00)	0,91 (0,00)	0,92 (0,00)
BC-LAW-K-means	1,00 (0,00)	0,91 (0,00)	0,92 (0,00)

TAB. 4.14 : *Degré d'utilisation des attributs par les algorithmes de pondération locale pour l'ensemble de données* DA3

Sur la table 4.15 est représenté le degré d'utilisation des attributs par les algorithmes de pondération locale pour l'ensemble de données IRIS. On constate que, comme dans le cas des méthodes de pondération globale, toutes les méthodes donnent plus d'importance aux deux derniers attributs.

	Attributs			
Algorithme	sepallength	sepalwidth	petallength	petalwidth
LAW-K-means	0,12 (0,30)	0,05 (0,14)	1,00 (0,11)	0,79 (0,17)
DE-LAW-K-means	0,05 (0,08)	0,06 (0,02)	1,00 (0,08)	0,92 (0,11)
LE-LAW-K-means	0,02 (0,00)	0,02 (0,00)	1,00 (0,00)	0,83 (0,00)
BE-LAW-K-means	0,02 (0,00)	0,02 (0,00)	1,00 (0,00)	0,83 (0,00)
DC-LAW-K-means	0,02 (0,00)	0,02 (0,00)	1,00 (0,03)	0,81 (0,04)
LC-LAW-K-means	0,02 (0,00)	0,02 (0,00)	1,00 (0,00)	0,83 (0,00)
BC-LAW-K-means	0,02 (0,00)	0,02 (0,00)	1,00 (0,00)	0,83 (0,00)

TAB. 4.15 : *Degré d'utilisation des attributs par les algorithmes de pondération locale pour l'ensemble de données* IRIS

Ces résultats montre que ces méthodes sont capables de découvrir les attributs pertinents pour la classification des données, que les attributs soient pertinents pour l'ensemble des données ou spécifiques à certaines classes. En revanche, des problèmes se posent toujours en cas de trop fortes corrélations.

4.4.6.3 Conclusion sur les pondérations

Les pondérations découvertes sont utilisées pour définir les classes, mais sont aussi une information supplémentaire fourni à l'utilisateur pour l'aider à comprendre le résultat de la classification. Les résultats expérimentaux montrent que les pondérations des attributs ont une grande influence sur la qualité des résultats.

En effet, sur les ensembles de données DA1 et IRIS, les algorithmes de pondération globale et locale sont capable de mettre en évidences les attributs les plus pertinents. En particulier,

les algorithmes évolutionnaires ont tendances à éliminer complètement l'influence des attributs non pertinent. Sur ces deux ensembles de données, la qualité de la classification est grandement améliorée comparativement à celle des résultats de l'algorithme K-means.

Sur l'ensemble de données DA3, les algorithmes de pondérations locales sont capables de mettre en évidence que chacun des attributs sont pertinent, bien qu'ils ne le soient pas tous pour la même classe, alors que les méthodes de pondération globale en sont incapable. Sur cet ensemble de données, la qualité des résultats de classification des méthodes de pondération locale est grandement améliorée par rapport à celle des résultats de l'algorithme K-means et des méthodes de pondération globale.

Sur l'ensemble de données DA2, les méthodes de pondération donnent une importance maximale à chacun des attributs corrélés et néglige totalement l'attribut indépendant, poutant indispensable pour discriminer deux des classes. La qualité des résultats sur cet ensemble est grandement dégradé par rapport à celle des résultats de l'algorithme K-means.

4.4.7 Temps de calcul

Il est connu que les algorithmes génétiques sont bien plus lents à converger vers une solution qu'un algorithme de *hill-climbing* (cette solution est toutefois bien souvent meilleure que celle trouvée par un algorithme de *hill-climbing*).

Les approches lamarckienne et baldwinienne sont également plus coûteuses en temps que l'approche darwinienne étant donné qu'à l'algorithme génétique s'ajoute une recherche locale. Les méthodes coévolutionnaires sont également plus coûteuse en temps car il est nécessaire de combiner les résultats de chacune des populations pour obtenir la solution globale.

Nous allons donc comparer le temps d'exécution des différentes méthodes. Les tests ont tous été réalisés sur des machines dotées de processeurs Opteron à 2,4 Ghz avec 4 Go de Ram. Ces machines n'étaient cependant pas utilisées exclusivement pour l'exécution de ces algorithmes de classification, ce qui peut expliquer certaines variations incohérentes.

4.4.7.1 Comparaison des méthodes de pondération globale

Sur la table 4.16 est présenté le temps de calcul des différents algorithmes de pondération globale exprimé en ms. Les algorithmes génétiques sont effectivement plus lents que les algorithmes classiques. Les algorithmes lamarckien et baldwinien sont légèrement plus lents que l'algorithme darwinien.

Cette différence en temps de calcul est cependant largement compensée par la qualité et la stabilité des résultats obtenus. En effet, bien qu'une génération d'un des algorithmes génétiques soit plus coûteuse en temps de calcul qu'une itération d'un algorithme classique, elle apporte plus en terme de qualité.

Sur la figure 4.7 est représentée l'évolution de la fonction d'évaluation au cours du temps pour les différents algorithmes de pondération globales.

On voit très clairement que les algorithmes proposés sont en réalité très intéressants en terme de temps de calcul. Il ne suffit que de quelques générations pour obtenir un résultat quasi optimal. On remarque également que le temps de calcul utilisé pour la recherche locale des algorithmes LE-GAW-K-means et LE-GAW-K-means est largement compensé par un gain de qualité par rapport à l'algorithme darwinien.

4.4.7.2 Comparaison des méthodes de pondération locale

Sur la table 4.17 est présenté le temps de calcul des différents algorithmes de pondération locale exprimé en ms. Comme c'était le cas pour les algorithmes de pondération globale, les méthodes évolutionnaires sont plus lentes que les méthodes classiques. Les algorithmes lamarckiens et

DA1	
K-means	1 557 (12)
GAW-K-means	1 554 (69)
DE-GAW-K-means	5 349 (82)
LE-GAW-K-means	7 910 (73)
BE-GAW-K-means	7 815 (70)
DA2	
K-means	1 513 (18)
GAW-K-means	1 565 (33)
DE-GAW-K-means	6 478 (126)
LE-GAW-K-means	10 336 (83)
BE-GAW-K-means	10 289 (89)
DA3	
K-means	1 957 (533)
GAW-K-means	1 881 (568)
DE-GAW-K-means	6 048 (582)
LE-GAW-K-means	9 131 (755)
BE-GAW-K-means	9 174 (801)
IRIS	
K-means	179 (33)
GAW-K-means	181 (9)
DE-GAW-K-means	1 659 (59)
LE-GAW-K-means	2 535 (19)
BE-GAW-K-means	2 499 (11)
DIABETES	
K-means	3 582 (2 617)
GAW-K-means	3 030 (487)
DE-GAW-K-means	6 678 (544)
LE-GAW-K-means	10 513 (546)
BE-GAW-K-means	10 630 (579)
IONOSPHERE	
K-means	735 (16)
GAW-K-means	878 (331)
DE-GAW-K-means	7 170 (226)
LE-GAW-K-means	13 620 (80)
BE-GAW-K-means	13 629 (173)
SONAR	
K-means	436 (98)
GAW-K-means	612 (9)
DE-GAW-K-means	8 032 (52)
LE-GAW-K-means	18 136 (49)
BE-GAW-K-means	18 001 (53)

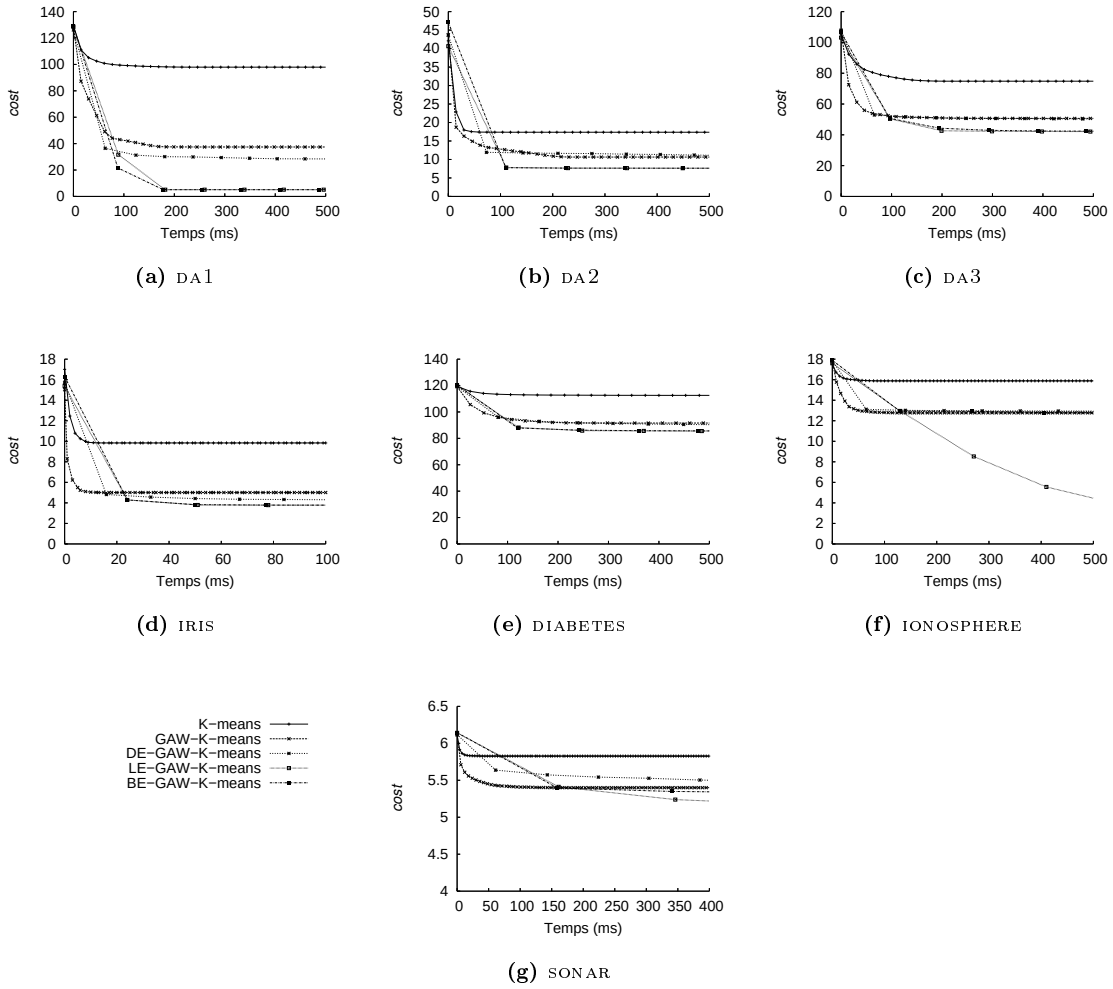
TAB. 4.16 : *Temps de calcul des algorithmes de pondération globale*

baldwiniens sont légèrement plus lents que les algorithmes darwiniens. On remarque aussi que les algorithmes coévolutionnaires sont très légèrement plus lents que les algorithmes évolutionnaires classiques.

Comment dans le cas des algorithmes de pondération globale, cette différence en temps de calcul est largement compensée par la qualité et la stabilité des résultats obtenus. En effet, bien qu'une génération d'un des algorithmes génétiques soit plus coûteuse en temps de calcul qu'une itération d'un algorithme classique, elle apporte plus en terme de qualité.

Sur la figure 4.8 est représentée l'évolution de la fonction d'évaluation au cours du temps pour les différents algorithmes de pondération locales. Dans un souci de lisibilité, seuls sont affichés les courbes des algorithmes K-means, LAW-K-means, DE-LAW-K-means, LE-LAW-K-means et DC-LAW-K-means.

On voit très clairement que, tout comme c'était le cas pour les algorithmes de pondération globale, les méthodes proposées sont en réalité très intéressantes en terme de temps de calcul (excepté pour l'ensemble de données SONAR pour lequel les algorithmes évolutifs ont du mal à converger). Il ne suffit, pour la plupart des ensembles de données, que de quelques générations pour obtenir un résultat quasi optimal. On remarque également que le temps de calcul utilisé pour la recherche locale dans l'algorithme LE-LAW-K-means et pour l'unification résultats dans l'algorithme DC-LAW-K-means est largement compensé par un gain de qualité par rapport à l'algorithme DE-LAW-K-means.



(a) DA1 (b) DA2 (c) DA3

(d) IRIS (e) DIABETES (f) IONOSPHERE

(g) SONAR

FIG. 4.7 : *Évolution de la fonction d'évaluation au cours du temps pour les algorithmes de pondération globale*

4.4.7.3 Conclusion sur le temps de calcul

Les résultats expérimentaux ont montré que les algorithmes évolutionnaires proposés pour l'optimisation des fonctions de coût pour la classification avec pondération d'attributs étaient relativement efficace en terme de temps de calcul. En effet, sur la majorité des ensembles de données testés, les algorithmes les plus efficaces convergeaient en moins de dix générations.

On constate que le temps de calcul nécessaire à la recherche locale à chaque génération dans les approches hybrides est largement compensé par un fort gain en terme de qualité par rapport à un algorithme darwinien. De même, le temps de calcul utilisé pour unifier les différentes solutions locales dans l'approche coévolutionnaire (darwinienne) est compensé par le gain en terme de qualité par rapport à l'approche évolutionnaire (darwinienne).

4.5 Conclusion

Les méthodes de classification avec pondération d'attributs par optimisations partielles basées sur K-means peuvent être améliorées par des algorithmes génétiques efficaces (algorithmes

Algorithme	Temps de calcul (ms)
DA1	
K-means	1 557 (12)
LAW-K-means	1 561 (35)
DE-LAW-K-means	5 413 (104)
LE-LAW-K-means	7 960 (55)
BE-LAW-K-means	7 936 (114)
DC-LAW-K-means	5 871 (83)
LC-LAW-K-means	8 516 (148)
BC-LAW-K-means	8 460 (100)
DA2	
K-means	1 513 (18)
LAW-K-means	1 574 (35)
DE-LAW-K-means	6 669 (183)
LE-LAW-K-means	10 614 (455)
BE-LAW-K-means	10 773 (624)
DC-LAW-K-means	7 669 (731)
LC-LAW-K-means	11 432 (547)
BC-LAW-K-means	11 513 (747)
DA3	
K-means	1 957 (533)
LAW-K-means	1 758 (506)
DE-LAW-K-means	6 178 (751)
LE-LAW-K-means	9 242 (642)
BE-LAW-K-means	9 227 (869)
DC-LAW-K-means	6 641 (523)
LC-LAW-K-means	9 674 (442)
BC-LAW-K-means	9 597 (426)
IRIS	
K-means	179 (33)
LAW-K-means	176 (13)
DE-LAW-K-means	1 712 (15)
LE-LAW-K-means	2 698 (26)
BE-LAW-K-means	2 645 (28)
DC-LAW-K-means	1 839 (24)
LC-LAW-K-means	2 801 (15)
BC-LAW-K-means	2 813 (131)
DIABETES	
K-means	3 582 (2 617)
LAW-K-means	3 059 (501)
DE-LAW-K-means	6 769 (563)
LE-LAW-K-means	10 752 (555)
BE-LAW-K-means	11 059 (569)
DC-LAW-K-means	7 003 (469)
LC-LAW-K-means	11 068 (698)
BC-LAW-K-means	10 797 (461)
IONOSPHERE	
K-means	735 (16)
LAW-K-means	891 (163)
DE-LAW-K-means	7 266 (264)
LE-LAW-K-means	13 901 (475)
BE-LAW-K-means	15 058 (413)
DC-LAW-K-means	7 340 (346)
LC-LAW-K-means	13 839 (271)
BC-LAW-K-means	13 832 (341)
SONAR	
K-means	436 (98)
LAW-K-means	744 (15)
DE-LAW-K-means	8 487 (33)
LE-LAW-K-means	23 693 (47)
BE-LAW-K-means	23 512 (48)
DC-LAW-K-means	8 303 (170)
LC-LAW-K-means	23 556 (23)
BC-LAW-K-means	23 463 (93)

TAB. 4.17 : *Temps de calcul des algorithmes de pondération locale*

lamarckien et baldwinien, approche par coévolution coopérative), permettant ainsi d'obtenir des résultats meilleurs, de manière plus robuste.

Il nous semble également possible de décomposer le problème de manière dissymétrique, en séparant la recherche des poids de la recherche des centres. Il devient alors possible d'utiliser des méthodes de coévolution coopérative dans le cas de la recherche de pondérations globales, en utilisant K populations pour les centres et une population pour les pondérations, ou encore

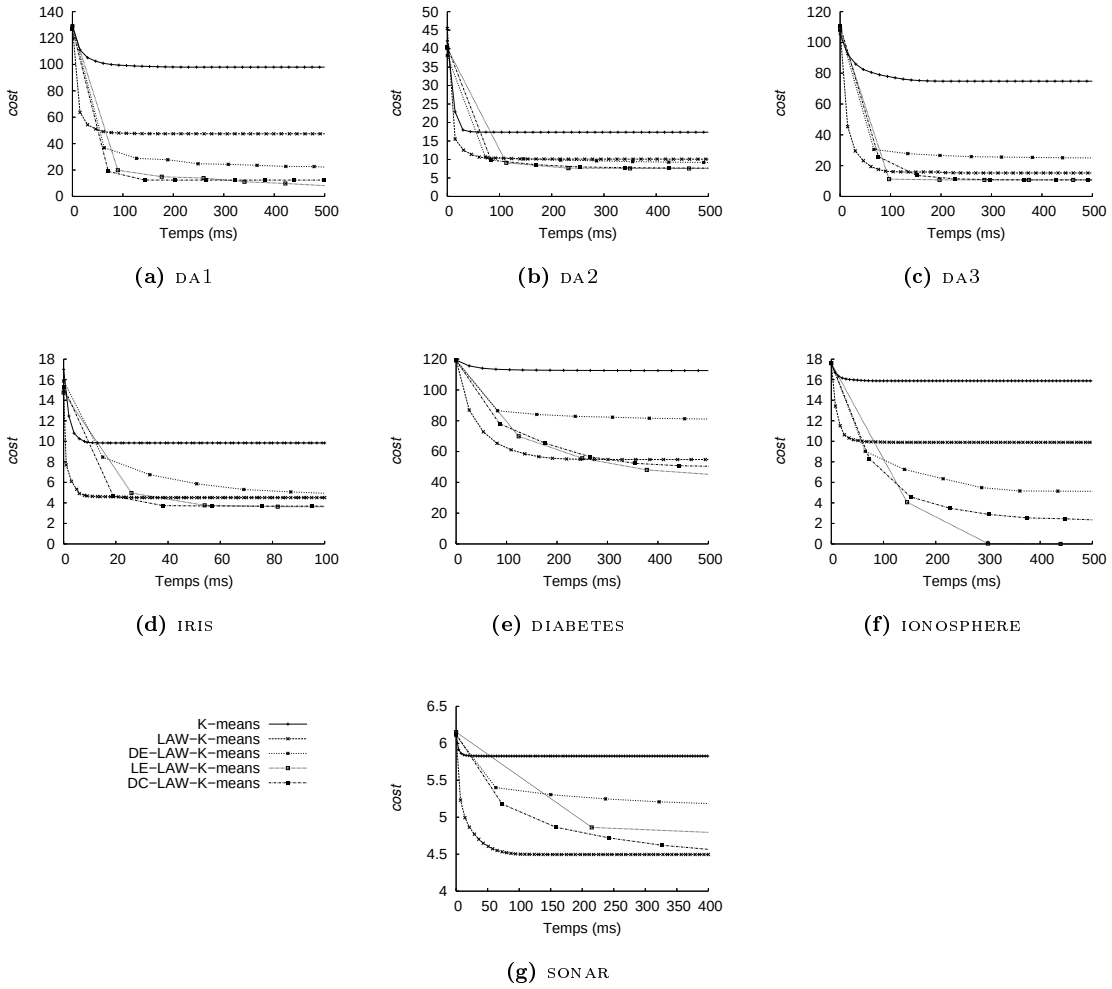


FIG. 4.8 : *Évolution de la fonction d'évaluation au cours du temps pour les algorithmes de pondération locale*

d'utiliser K populations pour les centres et K populations pour les pondérations dans le cas de pondérations locales. Ces possibilités n'ont cependant pas été étudiées dans ce mémoire.

Les résultats expérimentaux nous ont montré l'efficacité des algorithmes évolutionnaires (en particulier des algorithmes lamarckiens et baldwiniens) pour l'optimisation de la fonction de coût utilisée dans les méthodes de classification basées sur K-means. Nous avons également pu mettre en évidence l'importance de la pondération locale d'attributs pour obtenir des résultats pertinents.

L'utilisation des méthodes présentées dans ce chapitre reste cependant limitée à la découverte de classes sphériques dans une certaine métrique et définies à l'aide d'un prototype. De plus, nous avons vu que les performances était particulièrement mauvaises en cas de fortes corrélations entre les attributs. Ce type d'algorithmes n'est pas efficace sur tous les types de données et il nous est apparu indispensable de définir un cadre plus général permettant la classification non supervisée avec pondération locale d'attributs pour des types de classes différents.

Chapitre 5

MACLAW : un algorithme par approche modulaire pour la classification non supervisée avec pondération d'attributs

5.1 Introduction

Nous avons vu dans le chapitre 3 que la plupart des méthodes de classification non supervisée avec pondération locale des attributs étaient des méthodes intégrées. Ces méthodes, tout en étant efficaces, sont limitées par la définition qu'elles ont des classes (classes basées sur un prototype ou basées sur la densité) et ne sont donc pas adaptées à tous les types de données. La famille de méthodes de classification avec pondération des attributs basées sur K-means présentée dans le chapitre 4 est, par exemple, limitée à des classes sphériques dans une certaine métrique et définies en fonction d'un prototype. Or, nous avons vu dans le chapitre 1 qu'il existait une grande variété dans les méthodes de classification selon le type de classes que l'on souhaite découvrir dans les données. Il est donc intéressant de développer une méthode de pondération d'attributs générique pouvant être utilisée avec n'importe quelle méthode de classification non supervisée.

Nous avons également vu qu'il est difficile pour certaines méthodes d'utiliser des pondérations locales, en particulier dans le cas de méthodes de classification basées sur une distance inter-classes comme par exemple l'algorithme CURE.

Dans nos travaux, nous nous sommes donc intéressés à définir une méthode de classification non supervisée intégrant un mécanisme de pondération locale des attributs. Cette méthode doit permettre une flexibilité maximale. Elle doit, d'une part, permettre d'utiliser n'importe quel type de méthode d'extraction de classes (classifieurs, algorithmes *ad hoc*, etc.) supervisée ou non. D'autre part, des méthodes différentes doivent pouvoir être utilisées simultanément pour réaliser une même classification, notre approche pouvant ainsi être qualifiée de multi-stratégique dans la mesure où l'on considère que chaque méthode utilisée correspond à une stratégie d'extraction différente.

La méthode que nous proposons s'inscrit dans un cadre plus général de l'approche modulaire pour la classification non supervisée. Nous avons défini cette nouvelle approche qui consiste à décomposer le problème de classification en K classes en K sous-problèmes d'extraction d'une classe. Nous commencerons donc par détailler l'architecture générale de l'approche modulaire dans la section 5.2.

Nous présenterons ensuite comment cette approche modulaire peut être utilisée pour définir une méthode de classification non supervisée avec pondération locale des attributs par approche

enveloppe, en définissant les méthodes d'extraction des classes en fonction de méthodes de classification classiques et des pondérations globales. Cette méthode, appelée MACLAW sera exposée dans la section 5.3. L'algorithme MACLAW sera évalué dans la section 5.4, sur différents ensembles de données.

Nous terminerons ce chapitre par une conclusion dans la section 5.5

5.2 Architecture générale de l'approche modulaire

Une nouvelle approche pour la classification non supervisée appelée *approche modulaire pour la classification non supervisée* (ou *classification modulaire*) est proposée ici. Elle consiste à décomposer le problème de classification en K classes en K sous-problèmes d'extraction d'une classe. Le problème de classification revient alors à rechercher K *extracteurs*, mettant en évidence une classe chacun, permettant d'obtenir des classes pertinentes et complémentaires. Les données sont alors présentées à chacun des K extracteurs. Les K classes obtenues sont alors utilisées pour former une classification globale des données. La recherche des extracteurs se fait par optimisation d'une fonction d'évaluation par un algorithme de coévolution coopérative. La fonction d'évaluation est basée sur la complémentarité des classes ainsi que sur la qualité de chacune d'entre elles.

Dans cette section, nous présenterons en détail l'approche modulaire pour la classification non supervisée. Nous répondrons en particulier à trois questions :

- Comment les classes sont-elles extraites et la classification construite ? (section 5.2.1)
- Comment est évalué le résultat d'une classification ? (section 5.2.2)
- Quel est l'algorithme utilisé pour optimiser la fonction de coût ? (section 5.2.3)

Un descriptif général de l'algorithme sera présenté dans la section 5.2.4.

5.2.1 Notions de base

Les extracteurs (Définition 5.1) sont les briques de base de l'approche modulaire pour la classification non supervisée. Un extracteur extrait un seul ensemble d'objets, appelé sa *classe extraite* de l'ensemble à classifier. Un extracteur peut être vu comme un classifieur binaire, car il cherche deux classes dans les données : la classe extraite et la classe regroupant tous les objets n'appartenant pas à la classe extraite. Une telle fonction peut être définie par différents procédés comme, par exemple, un seuillage ou une méthode de classification. Selon l'application qui en sera faite, il est possible de définir les extracteurs *ad hoc*, comme par exemple des algorithmes d'extraction des routes dans les images de télédétections.

Définition 5.1 (Extracteur et classe extraite)
Un extracteur X est une fonction qui retourne un sous-ensemble $X(D)$ d'un ensemble d'objet D.

$X(D)$ est la classe extraite de X.

Plusieurs extracteurs sont utilisés en parallèle dans un classifieur modulaire (Définition 5.2) pour produire une classification en K classes.

Définition 5.2 (Classifieur modulaire)
Un classifieur modulaire est un ensemble d'extracteur $\mathcal{X} = \{X_1, \ldots, X_K\}$ où K est le nombre de classes à extraire.

L'approche modulaire pour la classification non supervisée peut être considérée comme une approche multi-stratégique. En effet, chaque extracteur du classifieur modulaire peut être défini selon une stratégie qui lui est propre.

Il est à noter que le nombre d'extracteurs (c'est-à-dire le nombre de classes) n'est pas nécessairement fixe : une phase d'apprentissage peut en effet ajouter ou supprimer des extracteurs afin de

déterminer le nombre de classes dans l'ensemble de données. Dans ce mémoire nous n'étudieront cependant pas la possibilité de faire évoluer le nombre de classes.

Les extracteurs étant indépendants les uns des autres, les classes extraites forment une classification douce partielle (CDP) : les classes ne sont pas nécessairement disjointes et tous les objets n'appartiennent pas nécessairement à une classe.

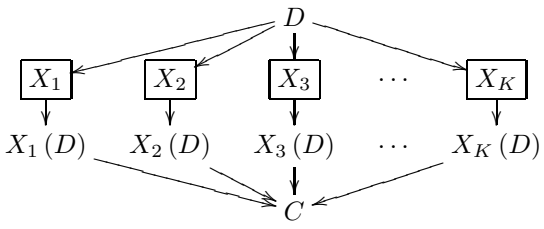


Fig. 5.1 : *Classifieur modulaire*

Comme indiqué sur la figure 5.1, les données à classifier D sont présentées à chacun des K extracteurs. Chaque extracteur X_k extrait une classe $X_k(D)$ depuis les données selon une méthode et des paramètres qui lui sont propres. Les différentes classes extraites sont alors regroupées dans une classification douce partielle C, c'est-à-dire qu'il peut y avoir des intersections entre les classes et des objets non classés.

La notion d'extracteur peut-être étendue à une définition floue de l'appartenance à une classe, ce qui permet d'avoir une définition plus fine des classes extraites (Définition 5.3). L'extracteur produit alors un sous-ensemble flou (ang. *fuzzy set*) auquel correspond un degré d'appartenance $\mu(o)$ pour chaque objet o [Zadeh, 1965].

Définition 5.3 (Extracteur flou)
Un extracteur flou est une fonction qui retourne un sous-ensemble flou $X(D)$ d'un ensemble d'objets D.

La classe extraite $X(D)$ est alors une classe floue.

A contrario, les extracteurs présentés dans la définition 5.1 peuvent être qualifiés d'*extracteurs durs*.

Un classifieur modulaire composé d'extracteurs flous, appliqué à un ensemble d'objets D, produira une classification floue. Contrairement à beaucoup d'algorithmes de classification floue, l'équation $\sum_{k=1}^{K} \mu_k(o) = 1$ n'est pas nécessairement vérifiée, ce qui signifie que des objets peuvent avoir un degré d'appartenance proche de 1 pour plusieurs classes ou au contraire un degré d'appartenance proche de 0 quelque soit la classe.

5.2.2 Évaluation de la qualité de la classification

Notre méthode de classification prévoie une phase d'optimisation (section 2.2) afin d'obtenir la meilleure classification possible. Pour pouvoir la mettre en place, il est nécessaire de définir comment sont évalués les résultats d'une classification par approche modulaire.

Dans le cas d'extracteurs durs, la classification produite est une CDP. Or, à notre connaissance, aucun critère d'évaluation de la qualité d'une CDP n'a encore été défini. Dans le cas d'extracteurs flous, la classification produite est une classification floue. Cependant, les critères classiques d'évaluation d'une classification floue ne sont pas adaptés à une classification modulaire définie à partir

d'extracteurs flous, car ils imposent que $\sum_{k=1}^{K} \mu_k(o) = 1$. Il est donc nécessaire de définir de nouveaux critères d'évaluation de la qualité adaptés à la classification modulaire.

Dans une CDP, les classes sont indépendantes et peuvent donc se superposer. Il se peut aussi que des objets n'appartiennent à aucune classe. Or, on attend d'une classification que les classes soient les plus séparées possibles et qu'il y ait un minimum d'objets atypiques. Les classes doivent être le plus complémentaires possible et recouvrir un maximum d'objets de l'ensemble de données à classifier, c'est-à-dire qu'idéalement une CDP doit être le plus proche possible d'un partitionnement des données. Nous définissons pour cela un degré de partitionnement qui sera discuté dans la section 5.2.2.1.

Cependant, avoir des classes complémentaires n'est pas toujours suffisant pour garantir une bonne classification. Il est en effet possible qu'une classe englobe la quasi totalité des objets et que le résultat forme une partition parfaite, mais sémantiquement incohérente. Un second critère est donc utilisé afin de tenir compte de la qualité interne des classes extraites afin d'assurer que les classes soient le plus homogènes possible. Ce critère sera discuté dans la section 5.2.2.2.

Les résultats de classification par approche modulaire sont donc évalués selon deux critères qui devront être combinés d'une manière qui dépendra, entre autres, de l'algorithme d'optimisation mis en œuvre.

5.2.2.1 Degré de partitionnement

Nous allons proposer ici un critère qui a pour but d'évaluer la complémentarité des classes. Ce critère se base sur le nombre de chevauchements entre les classes ainsi que sur la quantité d'objets classifiés. De manière similaire aux mesures de chevauchement dans les classifications floues, comme par exemple l'indice de Bezdek (Définition 1.9), nous souhaitons définir un critère qui est maximum lorsque la classification est une partition.

Pour cela nous allons d'abord introduire les notions de degré de classification d'un objet dans une CDP (Définition 5.4) et la notion d'objet k-classifié (Définition 5.5).

Définition 5.4 (Degré de classification)
On note $deg_C(o) = card(\{C_i \mid o \in C_i\})$ le degré de classification d'un objet, c'est-à-dire le nombre de classes auxquelles il appartient.

Définition 5.5 (Objet k-classifié)
Un objet k-classifié dans une CDP C est défini comme un objet tel que $deg_C(o) = k$, c'est-à-dire un objet qui appartient à k classes de C et a donc été extrait par k extracteurs.

$D_k(C)$ est défini comme l'ensemble de tous les objets k-classifiés d'une CDP C.

De façon évidente, une CDP C sur un ensemble d'objets D est une partition si et seulement si $D_1(C) = D$. Le degré de partitionnement doit être maximal dans ce cas là. Moins il y a d'objets 1-classifiés dans C plus bas sera le degré de partitionnement de C. Réciproquement, le degré de partitionnement d'une CDP doit être élevé si le nombre d'objets appartenant à plusieurs classes et le nombre d'objets non classifiés est minimal.

On pourrait ainsi définir le degré de partitionnement d'une CDP C simplement par le taux d'objets 1-classifiés par rapport au nombre d'objets dans D : $Q_P(C) = \frac{card(D_1(C))}{card(D)}$. Cependant, cette définition n'est pas satisfaisante car aucune différence n'est faite entre des objets k-classifiés et k'-classifiés avec $2 \leq k < k'$.

Une autre définition du degré de partitionnement est donc proposée. Pour cela, un degré d'unicité d'appartenance pour chaque objet est d'abord défini (Définition 5.6).

DÉFINITION 5.6 (DEGRÉ D'UNICITÉ D'APPARTENANCE D'UN OBJET)
Le degré d'unicité d'un objet o dans une CDP $C = \{C_1, \ldots, C_K\}$ est défini par :

$$Q_O(o, C) = \frac{1}{1 + |deg_C(o) - 1|} = \begin{cases} 1/2 \text{ si } o \in D_0(C) \\ 1 \quad \text{si } o \in D_1(C) \\ 1/k \text{ si } o \in D_k(C), \text{ avec } k \geq 2 \end{cases}$$

où $deg_C(o)$ est le degré de classification de o dans C (Définition 5.4)

Ainsi, si un objet o est 1-classifié dans C alors $Q_O(o, C) = 1$, sinon $0 < Q_O(o, C) < 1$. De plus, pour deux objets o et o', respectivement k-classifié et k'-classifié, $2 \leq k < k' \Rightarrow Q_O(o, C) > Q_O(o', C)$.

Nous pouvons alors définir la qualité sur l'ensemble des objets (Définition 5.7) par la moyenne du degré d'unicité d'appartenance sur tous les objets de D.

DÉFINITION 5.7 (DEGRÉ DE PARTITIONNEMENT)
On définit le degré de partitionnement d'une CDP C par :

$$Q_P(C) = \frac{1}{N} \sum_{o \in D} Q_O(o, C)$$

Exemple :

Sur la table 5.1 sont représentées trois classes extraites $C_1 = \{o_1, o_2, o_3, o_4, o_5\}$, $C_2 = \{o_3, o_4, o_5, o_6, o_7\}$ et $C_3 = \{o_4, o_5, o_6, o_7, o_8, o_9\}$ d'un ensemble de données $D = \{o_1, \ldots, o_{10}\}$.
L'objet o_{10} est 0-classifié et a donc un degré d'unicité de 1/2. Les objets o_1, o_2, o_8 et o_9 sont 1-classifiés et ont un degré d'unicité de 1. Les objets o_3, o_6 et o_7 sont 2-classifiés et ont un degré d'unicité de 1/2. Les objets o_4 et o_5 sont 3-classifiés et ont un degré d'unicité de 1/3.
Le degré de partitionnement de la CDP $C = \{C_1, C_2, C_3\}$ vaut donc 2/3.

	o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8	o_9	o_{10}
C_1	√	√	√	√	√					
C_2			√	√	√	√	√			
C_3				√	√	√	√	√	√	
deg_C	1	1	2	3	3	2	2	1	1	0
Q_O	1	1	1/2	1/3	1/3	1/2	1/2	1	1	1/2

TAB. 5.1 : *Évaluation du degré de partitionnement dans une classification dure*

PROPRIÉTÉ 5.1
Le degré de partitionnement d'une CDP C sur un ensemble de données D est d'autant plus élevé que C est proche d'une partition de D et réciproquement.

Preuve :

Prouvons que si une CDP C est plus proche d'être une partition de D alors le degré de partitionnement sera plus élevé.
On voit facilement que $Q_P(C) = 1$ si et seulement si C est une partition de D. En effet, si C est une partition, alors $D = D_1(C)$ et donc $Q_O(o, D) = 1, \forall o \in D$. Ainsi $Q_P(C) = 1$. Si C n'est pas une partition, il existe un objet o tel que $o \notin D_1(C)$ et donc $Q_O(o, C) < 1$. Ainsi $Q_P(C) < 1$.
Il n'existe pas de définition exacte de ce que c'est d'être « proche d'une partition ». Nous allons cependant montrer que notre critère est pertinent pour décider si une CDP C est

plus proche d'être une partition qu'une autre CPD C' en étudiant trois cas pour lesquels cela est intuitivement vrai. Dans chacun de ces trois cas, les CDP C et C' ne diffèrent que pour un objet o. Dans chacun des cas, on considère que la classification C est plus proche d'être une partition que C'.

Soit deux CDP C et C' sur un ensemble D à N éléments. On considère que le taux d'objet k-classifiés dans C (respectivement C') est de p_k (respectivement p'_k), avec $\sum_{k=0}^{K} p_k = \sum_{k=0}^{K} p'_k = 1$. On a alors $Q_P(C) = \frac{p_0}{2} + p_1 + \sum_{k=2\ldots K} \frac{p_k}{k}$ (respectivement $Q_P(C') = \frac{p'_0}{2} + p'_1 + \sum_{k=2\ldots K} \frac{p'_k}{k}$).

cas 1 : C et C' sont identiques, à la différence que o est 1-classifié dans C, mais 0-classifié dans C'. Donc :

- $p_0 + \frac{1}{N} = p'_0$
- $p_1 = p'_1 + \frac{1}{N}$
- $p_k = p'_k, k \in [2; K]$

Alors, $Q_P(C) - Q_P(C') = \frac{p_0}{2} + p_1 - \frac{p'_0}{2} - p'_1 = \frac{1}{2N}$, et donc $Q_P(C) > Q_P(C')$.

cas 2 : C et C sont identiques, à la différence que o est 1-classifié dans C, mais k-classifié (avec $k \geq 2$) dans C'. Donc :

- $p_0 = p'_0$
- $p_1 = p'_1 + \frac{1}{N}$
- $p_k + \frac{1}{N} = p'_k$
- $p_h = p'_h, h \in [2; K] \setminus \{k\}$

Alors, $Q_P(C) - Q_P(C') = p_1 + \frac{p_k}{k} - p'_1 - \frac{p'_k}{k} = \frac{k-1}{k \times N}$, et donc $Q_P(C) > Q_P(C')$.

cas 3 : C et C' sont identiques, à la différence que o est k-classifié dans C, mais k'-classifié dans C', avec $2 \leq k < k'$. Donc :

- $p_0 = p'_0$
- $p_1 = p'_1$
- $p_k = p'_k + \frac{1}{N}$
- $p_{k'} + \frac{1}{N} = p'_{k'}$
- $p_h = p'_h, h \in [2; K] \setminus \{k, k'\}$

Alors, $Q_P(C) - Q_P(C') = \frac{p_k}{k} + \frac{p_{k'}}{k'} - \frac{p'_k}{k} - \frac{p'_{k'}}{k'} = \frac{k'-k}{k \times k' \times N}$, et donc $Q_P(C) > Q_P(C')$.

Prouvons maintenant la réciproque, c'est-à-dire qu'une CDP est d'autant plus proche d'être une partition que son degré de partitionnement est élevé.

Soit deux CDP C et C', avec $Q(C) > Q(C')$. Supposons que C' est plus proche d'être une partition que C. Alors, comme nous l'avons montré, $Q(C') > Q(C)$ ce qui est une contradiction.

Le degré de partitionnement peut être étendu à une définition floue de l'appartenance à une classe. Ce critère se différencie des mesures classiques de chevauchement dans une classification floue, car il n'est pas nécessaire de vérifier $\sum_{k=1}^{K} \mu_k(o) = 1$.

Définition 5.8 (Degré d'unicité d'appartenance floue d'un objet)
On définit le degré d'unicité d'un objet o dans une CDP floue $C = \{C_1, \ldots, C_K\}$ par :

$$Q_O(o, C) = \frac{1}{2 - \left(2 \times \max_{C_k \in C} \mu_k(o)\right) + \sum_{C_k \in C} \mu_k(o)}$$

Cette définition floue du degré d'unicité d'appartenance d'un objet est bien une extension à la définition floue de l'appartenance d'un objet à une classe de la définition 5.6. En effet, on voit facilement que si le degré d'appartenance $\mu_k(o)$ ne peut valoir que 0 ou 1, on retombe sur la première définition.

La définition de degré de partitionnement d'une classification floue est identique à la définition pour une classification dure (Définition 5.7).

Exemple :

Sur la table 5.2 sont représentées trois classes extraites floues C'_1, C'_2 et C'_3 (avec pour fonction d'appartenance μ'_1, μ'_2 et μ'_3) d'un ensemble de données $D = \{o_1, \ldots, o_{10}\}$. Le degré de partitionnement flou de la classification $C' = \{C'_1, C'_2, C'_3\}$ vaut donc environ $0,52$.

	o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8	o_9	o_{10}
μ'_1	0,9	0,8	0,6	0,7	0,8	0,4	0,1	0,4	0,1	0,3
μ'_2	0,1	0,2	0,6	0,6	0,9	0,7	0,6	0,2	0,2	0,1
μ'_3	0,1	0,2	0,4	0,9	0,9	0,8	0,8	0,6	0,7	0,4
Q_O (flou)	0,77	0,63	0,42	0,42	0,36	0,43	0,53	0,5	0,63	0,5

Tab. 5.2 : *Évaluation du degré de partitionnement dans une classification floue*

Propriété 5.2
Le degré de partitionnement d'une classification floue C sur un ensemble de données D est d'autant plus élevé que C est proche d'une partition de D et réciproquement.

Preuve :

Cette démonstration se base sur le même principe que la démonstration de la propriété 5.1.
Prouvons que si une CDP C est plus proche d'être une partition de D alors de degré de partitionnement sera plus élevé.
Soit deux classifications floues C et C' sur un ensemble D de N éléments. On note $\mu(o)$ et $\mu'(o)$ les degrés d'appartenance de o dans C et dans C'.
Dans chacun des deux cas présentés ci-dessous, les distributions C et C' ne diffèrent que pour un objet o. Dans chacun des cas, on considère intuitivement que la classification C est plus proche d'être une partition de D que C'.

cas 1 : C et C' sont identiques sauf pour un objet o : $\max_{C_k \in C} \mu_k(o) > \max_{C'_k \in C'} \mu'_k(o)$. On note
$\delta = \max_{C_k \in C} \mu_k(o) - \max_{C'_k \in C'} \mu'_k(o)$ *(donc $\delta > 0$).*
Comme $\frac{1}{Q_O(o,C)} = 2 - \left(2 \times \max_{C_k \in C} \mu_k(o)\right) + \sum_{C_k \in C} \mu_k(o) = 2 - \left(2 \times \max_{C'_k \in C'} \mu'_k(o)\right) + \sum_{C'_k \in C'} \mu'_k(o) - \delta$, *on a $Q_O(o,C) > Q_O(o,C')$.*
Or $Q_P(C) - Q_P(C') = \frac{1}{N} Q_o(o,C) - \frac{1}{N} Q_o(o,C')$, *donc $Q_P(C) > Q_P(C')$.*

cas 2 : C et C' sont identiques sauf pour un objet o : $\max_{C_k \in C} \mu_k(o) = \max_{C'_k \in C'} \mu'_k(o)$ et $\exists k \mid \mu_k(o) < \mu'_k(o)$. On note $\delta = \mu_k(o) - \mu'_k(o)$ (donc $\delta < 0$).
Comme $\frac{1}{Q_O(o,C)} = 2 - \left(2 \times \max_{C_k \in C} \mu_k(o)\right) + \sum_{C_k \in C} \mu_k(o) = 2 - \left(2 \times \max_{C'_k \in C'} \mu'_k(o)\right) + \sum_{C'_k \in C'} \mu'_k(o) + \delta$, *on a $Q_O(o,C) > Q_O(o,C')$.*
Or $Q_P(C) - Q_P(C') = \frac{1}{N} Q_o(o,C) - \frac{1}{N} Q_o(o,C')$, *donc $Q_P(C) > Q_P(C')$.*

La preuve de la réciproque est identique à celle de la démonstration de la propriété 5.1.

5.2.2.2 Qualité interne des classes

Le degré de partitionnement n'est pas toujours suffisant pour évaluer de manière fiable la qualité d'une classification. Il est en effet possible qu'une classe englobe la quasi totalité des objets et que le résultat forme une partition parfaite, mais sémantiquement incohérente. Nous définissons donc un second critère basé sur la qualité interne des classes extraites afin d'assurer que les classes soient les plus homogènes possible. Nous n'utilisons pas ici un critère d'évaluation inter-classes afin de garder notre approche générale. En effet, chaque classe a pu être extraite selon une définition différente de ce qu'est une classe, ce qui rend impossible l'utilisation d'une mesure inter-classes.

L'évaluation de la qualité interne des classes va dépendre des méthodes d'extraction utilisées. N'importe quel critère, pour peu qu'il prenne ses valeurs $[0;1]$, est utilisable dans notre méthode, comme par exemple la compacité d'une classe. Ce critère est totalement dépendant de l'application et ne peut être choisi a priori. Partant de ce critère, nous pouvons définir un indice de qualité interne des classes par une moyenne géométrique sur l'ensemble des classes (Définition 5.9).

Définition 5.9 (Qualité interne des classes)
On définit la qualité des classes d'une répartition par :

$$Q_I(C) = \sqrt[K]{\prod_{C_i \in C} q_I(C_i)}$$

où $q_I(C_i)$ est un critère de qualité d'une classe qui prend ses valeurs dans $[0;1]$.

5.2.3 Optimisation

Les critères d'évaluation définis dans la section 5.2.2 permettent de juger de la qualité d'une classification réalisée à partir d'un classifieur modulaire. Un algorithme d'optimisation peut alors être utilisé pour « découvrir » des extracteurs produisant des classes pertinentes et complémentaires. Cette méthode doit faire évoluer simultanément les K extracteurs, de sorte que les classes extraites maximisent les deux critères d'évaluation.

Les extracteurs pouvant être définis de manières très variés, il est difficile de définir une méthode d'optimisation générale par *hill-climbing*, descente de gradient, ou programmation linéaire car la fonction d'optimisation ne peut pas être aisément exprimable par une fonction mathématique. Nous avons donc choisi d'utiliser les algorithmes évolutionnaires, plus précisément les algorithmes de coévolution coopérative, comme méthode d'optimisation.

Nous présenterons dans cette section comment sont encodés les individus (section 5.2.3.1), quelle est la fonction à optimiser (section 5.2.3.2) et comment sont évalués les individus (section 5.2.3.3). Nous présenterons également comment sont initialisés (section 5.2.3.4) et modifiés (section 5.2.3.5) les individus représentatifs de chaque population.

5.2.3.1 Encodage des individus : génotype et phénotype

L'encodage d'un extracteur dans un chromosome va dépendre de la méthode d'extraction employée. Si l'extraction est réalisée par des seuillages sur les différents attributs, l'extracteur peut-être alors encodé simplement par l'ensemble des seuils. Pour des méthodes d'extraction plus complexes, un extracteur peut être encodé par un arbre syntaxique représentant la fonction à utiliser pour extraire la classe (programmation génétique).

Dans une approche évolutionnaire, le génotype d'un individu contient l'encodage de l'ensemble des extracteurs. Le phénotype d'un tel individu est toujours une CDP (dans le cas d'extracteurs durs) ou une classification floue (dans le cas d'extracteurs flous). La classification est simplement obtenue par l'application des extracteurs encodés dans l'individu à un ensemble de données D.

Cependant, le principe même de l'approche modulaire pour la classification non supervisée permet une décomposition naturelle du problème et une approche par coévolution coopérative symétrique peut être mise en œuvre de manière évidente. Pour cela, K populations d'extracteurs sont utilisées. Dans cette approche, le génotype d'un individu contient uniquement l'encodage d'un extracteur. Le phénotype d'un tel individu est alors une classe extraite dure (dans le cas d'extracteurs durs) ou floue (dans le cas d'extracteurs flous), obtenue par application de l'extracteur encodé dans l'individu à l'ensemble de données D.

5.2.3.2 Fonction d'évaluation

L'évaluation d'une CDP se fait par deux critères : le degré de partitionnement (Définition 5.7) et la qualité interne des classes (Définition 5.9). Nous cherchons donc à optimiser simultanément ces deux critères. Or la plupart des méthodes d'optimisation, en particulier les algorithmes génétiques, permettent d'optimiser une seule fonction d'évaluation. L'optimisation multiobjectif n'est pas un problème trivial car l'amélioration d'une solution sur un critère d'évaluation ne garantit pas une améliorations sur les autres critères. De nombreuses recherches ont été menées à ce sujet [Fonseca et Fleming, 1995 ; Tan *et al.*, 2005] et plusieurs approches ont été proposées, comme par exemple :

- l'agrégation de plusieurs critères en un seul peut se faire par une moyenne arithmétique ou géométrique pondérée, par un simple produit ou par toute autre fonction ;
- l'utilisation de priorités dans les objectifs consiste à trier les individus dans l'ordre lexicographique selon la priorité de chacun des objectifs. Dans notre cas, il n'existe aucune priorité entre les critères d'évaluations ;
- un individu $I^{i,g}$ est dit *dominé au sens de Pareto* s'il existe au moins un individu $I^{j,g}$ meilleur sur tous les critères d'évaluation. On considère alors tous les individus non dominés au sens de Pareto comme les meilleurs individus de la population, en leur affectant une qualité maximale pour cette génération. On recherche alors parmi les individus restant les individus non dominés, pour leur affecter une qualité un peu moins élevée. Le processus est alors répété jusqu'à ce que tous les individus soient évalués ;
- une population peut être divisée en plusieurs sous-populations (une par objectif). Les individus d'une population sont évalués selon un seul des objectifs (qui diffère bien entendu d'une sous-population à l'autre). Par moment, les meilleurs individus de chaque sous-population sont mélangés avec les autres pour permettre l'optimisation de tous les critères. Cette méthode est cependant difficile à mettre en oeuvre, en particulier dans le cas de la coévolution qui utilise déjà plusieurs populations.

Bien que chacune de ces méthodes aurait pu être employée, afin de simplifier l'implantation de notre méthode et de vérifier sa faisabilité, nous avons choisi d'agréger les deux critères en un seul par un simple produit : ainsi un individu c fois meilleur sur un des deux critères d'évaluation aura c fois plus de chance d'être sélectionné dans un processus de sélection par roue de loterie. L'algorithme aura donc pour objectif de maximiser l'indice de qualité d'une CDP (Définition 5.10).

Définition 5.10 (Indice de qualité d'une CDP)
La qualité d'une CDP C est définie par :

$$Q_{CDP}(C) = Q_P(C) \times Q_I(C)$$

5.2.3.3 Évaluation des individus

Dans la section 5.2.3.2, nous avons présenté la fonction que l'algorithme coévolutionnaire doit optimiser. Il reste cependant à définir comment les individus sont évalués.

Dans une approche évolutionnaire, un individu est évalué en fonction de l'indice de qualité d'une CDP (Définition 5.10). Les K classes sont extraites à partir d'un ensemble de données D en fonction des K extracteurs encodés dans l'individu. L'individu est alors évalué selon la qualité de la CDP formée à partir des différentes classes extraites.

Cependant, dans le cas de notre approche par coévolution coopérative, l'évaluation d'un individu est plus complexe. Chaque individu ne produit qu'une partie de la solution (une classe). La solution globale (une classification en K classes) est construite en fonction des autres populations. Comme cela a été présenté dans la section 2.4, il est préférable d'évaluer un individu dans un environnement simple ou multiple, c'est-à-dire en fonction d'un ou plusieurs individus représentatifs de chacune des autres populations, définis par les meilleurs individus découverts dans chacune des populations.

Soit p le nombre d'individus représentatifs d'une population. On note alors $\Delta^g = \left\{\Delta^{1,g}, \ldots, \Delta^{p,g}\right\}$ l'environnement à la g-ième génération avec $\Delta^{i,g} = \left\{\Delta_1^{i,g}, \ldots, \Delta_K^{i,g}\right\}$ où $\Delta_k^{i,g}$ est la classe extraite par le i-ième individu représentatif de la k-ième population à la g-ième génération.

DÉFINITION 5.11 (ÉVALUATION D'UN INDIVIDU)
Un individu $I_k^{i,g}$ encode un extracteur $X_k^{i,g}$ qui permet d'obtenir une classe extraite $X_k^{i,g}(D)$. L'individu $I_k^{i,g}$ est évalué par :

$$Q\left(I_k^{i,g}\right) = \max_{C \in \Delta^g\left(X_k^{i,g}(D)\right)} \left(Q\left(C\right)\right)$$

où $\Delta^g\left(X_k^{i,g}(D)\right) = \left\{\left\{\Delta_1^{p_1,g}, \ldots, X_k^{i,g}(D), \ldots, \Delta_K^{p_K,g}\right\} \mid p_h \in [1;p], \forall h \in [1;K] \setminus \{k\}\right\}$.

Plus simplement dans le cas d'un environnement simple (c'est-à-dire si $p = 1$), la qualité d'un individu $I_k^{i,g}$ est donnée par :

$$Q\left(I_k^{i,g}\right) = Q\left(\Delta^{1,g}\left(X_k^{i,g}(D)\right)\right)$$

où $\Delta^{1,g}\left(X_k^{i,g}(D)\right) = \left\{\Delta_1^{1,g}, \ldots, X_k^{i,g}(D), \ldots, \Delta_K^{1,g}\right\}$.

Exemple :

Soit un ensemble $D = \{o_1, \ldots, o_{10}\}$ et $\Delta_1^{1,g} = \{o_1, o_2, o_3, o_4\}$, $\Delta_2^{1,g} = \{o_3, o_4, o_5, o_6\}$ et $\Delta_3^{1,g} = \{o_5, o_6, o_7\}$ les classes extraites dures des individus représentatifs, avec $q_I\left(\Delta_1^{1,g}\right) = 0,8$, $q_I\left(\Delta_2^{1,g}\right) = 0,7$ et $q_I\left(\Delta_3^{1,g}\right) = 0,6$.
L'évaluation d'un individu $I_2^{i,g}$ permettant d'obtenir une classe extraite $X_2^{i,g}(D) = \{o_7, o_8, o_9\}$ ayant une qualité interne $q_I\left(X_2^{i,g}(D)\right) = 0,8$ se fait en évaluant la qualité de la CDP $\Delta^{1,g}\left(X_2^{i,g}(D)\right) = \left\{\Delta_1^{1,g}, X_2^{i,g}(D), \Delta_3^{1,g}\right\}$, comme indiqué sur la table 5.3, $Q_P\left(I_2^{i,g}\right) = 0,9$. Ainsi $Q\left(I_2^{i,g}\right) = 0,9 \times \sqrt[3]{0,8 \times 0,8 \times 0,6} \simeq 0,58$.

	o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8	o_9	o_{10}
$\Delta_{1,1}^g$	√	√	√	√						
$X_2^{i,g}(D)$							√	√	√	
$\Delta_{3,1}^g$					√	√	√			
deg_C	1	1	1	1	1	1	2	1	1	0
Q_O	1	1	1	1	1	1	1/2	1	1	1/2

TAB. 5.3 : *Évaluation d'un individu dans une approche coévolutionnaire*

5.2.3.4 Initialisation des individus représentatifs

Comme cela a été exposé dans la section 2.4.4.2, les individus représentatifs ne sont pas définis à la première génération. Pour définir Δ^1, nous avons vu qu'il existe deux méthodes : les collaborations arbitraires et l'utilisation d'une méthode d'initialisation. Les collaborations arbitraires consistent à évaluer un certain nombre de collaborations entre individus de chaque population et de choisir les meilleurs comme individus représentatifs. L'utilisation d'une méthode d'initialisation consiste à utiliser le résultat d'un autre algorithme d'optimisation afin de se placer dans une zone favorable de l'espace de recherche.

Une méthode d'initialisation des individus représentatifs n'a besoin que de découvrir le phénotype des individus représentatifs, pas leur génotype. Ainsi, dans un algorithme de classification modulaire, le génotype correspond à l'encodage des extracteurs et le phénotype aux classes extraites. La méthode d'initialisation des individus représentatifs doit donc découvrir un ensemble de classes. Une méthode de classification classique peut être utilisée comme méthode d'initialisation des individus représentatifs, comme par exemple les méthodes K-means, COBWEB ou EM.

Comme cela est illustré sur la figure 5.2, une classification est effectuée à partir des données D, avec un algorithme M_{init}. Une classification C_{init} est alors obtenue. S'il est possible de définir le nombre de classes de l'algorithme (comme c'est le cas pour K-means par exemple), celui-ci doit être configuré pour découvrir K classes (c'est-à-dire une classe par population) ; chaque classe de C_{init} sera alors utilisée pour définir le phénotype d'un individu représentatif d'une population. Dans le cas contraire (comme c'est le cas pour COBWEB), l'algorithme doit être configuré pour trouver au moins K classes ; K classes seront alors choisies pour définir le phénotype d'un individu représentatif d'une population.

Si la méthode d'initialisation utilisée est un algorithme produisant une classification dure et que cet algorithme découvre K classes (ce qui est cas par exemple si l'on utilise l'algorithme K-means comme méthode d'initialisation), le degré de partitionnement de la classification initiale est alors maximal. Cela n'est cependant pas gênant étant donné que les classifications sont évaluées selon deux critères, le degré de partitionnement et la qualité interne des classes.

L'utilisation d'un environnement multiple permet de garantir une diversité et donc une meilleure évaluation des individus. Cependant les individus représentatifs d'une population P_k doivent être représentatifs de la classe C_k. Les classes des individus représentatifs doivent être suffisamment similaires entre elles (elles doivent correspondre à la même niche de l'écosystème). Si deux individus représentatifs d'une même population sont trop différents, c'est que l'un d'eux correspond plutôt à une autre population. Il est difficile de mettre en œuvre une méthode d'initialisation permettant d'obtenir des individus représentatifs représentants des classes similaires en utilisant des méthodes de classification non supervisée. Dans le cas d'un environnement multiple, il est donc préférable que l'évaluation des individus de la première génération ne se fasse que dans un environnement simple.

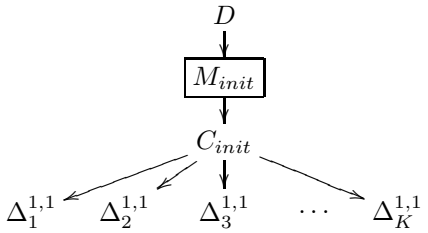


Fig. 5.2 : *Initialisation des individus représentatifs par une méthode de classification*

5.2.3.5 Modification des individus représentatifs

À la fin de chaque génération, les p individus représentatifs de chaque population sont mis à jour en fonction des p meilleurs individus de chacune des populations. On note $B_k^{1,g}, \ldots, B_k^{p,g}$ les classes extraites par les p meilleurs individus de la population P_k^g à la g-ième génération. $Comb^g$ est l'ensemble des CDP qu'il est possible de construire en utilisant pour la k-ième classe les classes extraites $\Delta_k^{i,g}$ ou $B_k^{i,g}$ avec $i = 1, \ldots, p$ (on a donc $card\,(Comb^g) = (2 \times p)^K$).

$\Delta^{1,g+1}$ est définie comme la meilleure CDP de $Comb^g$. Si $p > 1$, les classes des meilleures combinaisons sont utilisées. Cependant, afin d'assurer la diversité des individus représentatifs, une même classe ne peut pas servir à définir plusieurs individus représentatifs d'une population. Donc, s'il y a p individus représentatifs dans chaque population, les p meilleures combinaisons au moins seront étudiées pour définir ces individus représentatifs à la génération suivante.

Exemple :

Dans un problème de classification en trois classes, avec deux individus représentatifs par population, l'évaluation des combinaisons est présentée sur la table 5.4 (dans l'ordre décroissant de qualité). Les individus représentatifs choisis sont alors $B_1^{1,g}$, $\Delta_2^{1,g}$, $\Delta_3^{1,g}$, $\Delta_1^{1,g}$, $B_2^{2,g}$ *et* $B_3^{2,g}$ *(en gras dans la table). Ainsi* $\Delta^{1,g+1} = \left\{B_1^{1,g}, \Delta_2^{1,g}, \Delta_3^{1,g}\right\}$ *et* $\Delta^{2,g+1} = \left\{\Delta_1^{1,g}, B_2^{2,g}, B_3^{1,g}\right\}$.

Combinaison	Qualité
$\left\{\boldsymbol{B_1^{1,g}}, \boldsymbol{\Delta_2^{1,g}}, \boldsymbol{\Delta_3^{1,g}}\right\}$	$0,56$
$\left\{\boldsymbol{\Delta_1^{1,g}}, \Delta_2^{1,g}, \Delta_3^{1,g}\right\}$	$0,54$
$\left\{B_1^{1,g}, \boldsymbol{B_2^{2,g}}, \Delta_3^{1,g}\right\}$	$0,53$
$\left\{\Delta_1^{1,g}, B_2^{1,g}, \Delta_3^{1,g}\right\}$	$0,52$
$\left\{\Delta_1^{2,g}, \Delta_2^{2,g}, \boldsymbol{B_3^{1,g}}\right\}$	$0,46$
$\vdots$	$\vdots$

Tab. 5.4 : *Évaluation des combinaisons*

5.2.4 Descriptif de l'algorithme

En nous basant sur les notions présentées dans les sections précédentes, nous pouvons décrire notre algorithme modulaire de classification non supervisée.

Étape 0 – initialisation

- les individus représentatifs sont initialisés (par coopérations arbitraires ou par l'utilisation d'une méthode d'initialisation) ;
- les individus sont initialisés aléatoirement.

À chaque génération g, la procédure suivante est répétée jusqu'à atteindre un critère d'arrêt (évaluation satisfaisante, nombre de génération maximale) :

Étape 1 – évaluation des individus

- les classes sont extraites à partir de chacun des individus ;
- les individus sont évalués en fonction de leur classe extraite et des individus représentatifs ;
- les meilleurs individus de chaque population sont sélectionnés.

Étape 2 – mise à jour des individus représentatifs

Les individus représentatifs sont mis à jours en fonction des meilleurs individus de chacune des populations et des individus représentatifs de la génération courante.

Étape 3 – reproduction

De nouveaux individus sont créés par reproduction (par les opérateurs de sélection, de croisement et de mutation), indépendamment dans chaque population.

5.3 Approche modulaire pour la classification non supervisée avec pondération d'attribut

L'approche modulaire pour la classification non supervisée présentée dans la section 5.2 peut être utilisée pour définir une méthode de classification avec pondération locale des attributs par approche enveloppe. Dans cette section nous allons présenter une nouvelle méthode appelée MACLAW (*Modular Approach for Clustering with Local Attribute Weighting*).

MACLAW est basé sur l'observation suivante : l'utilisation, de manière globale, de pondérations pertinentes pour une classe C_k peut permettre de mettre en évidence la classe C_k, les autres classes pouvant être confondues entre elles. Ainsi, si l'on produit différentes classifications, en utilisant pour chacune des pondérations globales différentes, des classes pertinentes vont apparaître dans certaines classifications mais pas dans d'autres. Une combinaison de ces différentes classifications, en choisissant pour chacune les classes pertinentes mises en évidence, pourrait amener à la construction d'une classification judicieuse des données. Cette approche a donc l'avantage de n'utiliser que des pondérations globales pour découvrir les classes, ce qui permet d'utiliser n'importe quel type d'algorithme de classification, tout en permettant aussi de construire une classification avec pondération locale des attributs.

Ce principe peut être mis en œuvre dans une approche modulaire : l'extraction d'une classe se fait alors en réalisant une classification des données selon une méthode quelconque et en utilisant des pondérations globales, puis en choisissant, selon un certain critère, une des classes obtenues comme classe extraite.

Dans cette section, nous exposons la méthode MACLAW construite sur des extracteurs basés sur des classifieurs. Cette méthode peut produire des classifications dures ou floues (section 5.3.1). Nous présenterons ensuite l'encodage des extracteurs dans l'algorithme génétique intégré dans MACLAW ainsi que les opérations génétiques qui sont utilisées (section 5.3.2).

5.3.1 Extracteurs basés sur des classifieurs

Dans MACLAW, un extracteur est défini en fonction d'une méthode de classification donnée par l'utilisateur (Définition 5.12).

Définition 5.12 (Extracteur basé sur un classifieur)
Un extracteur basé sur un classifieur est un triplet $X = (M, w, s)$, où M est une méthode de classification, w un vecteur de poids $\{w_1, \ldots, w_n\}$ et s un critère de sélection d'une classe.

Le processus d'extraction d'une classe est présenté sur la figure 5.3. L'application de la méthode M sur un ensemble de données D en utilisant le vecteur global de poids w permet d'obtenir un ensemble de classes $M^w(D) = \left\{C_1, C_2, \ldots, C_{K_M^w}\right\}$ (le nombre de classes obtenues K_M^w dépend à la fois de la méthode employée et du vecteur de poids utilisé). La classe extraite $X(D)$ est alors la classe qui maximise le critère s, c'est-à-dire $X(D) = \underset{C_k \in M^w(D)}{\text{Argmax}} \left(s(C_k)\right)$.

Le critère s peut être défini de différentes manières. Nous proposons ici 2 critères de sélection.

La *sélection de la classe extraite par un critère fixe* se fait en utilisant un critère de qualité de classe, comme par exemple l'indice de qualité de Wemmert et Gançarski pour une classe [Wemmert

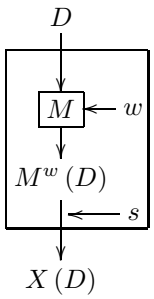


FIG. 5.3 : *Extracteur basé sur un classifieur*

et al., 2000]. Cette approche consiste à choisir la meilleure classe parmi celles obtenues par la méthode de classification M. Le désavantage de ce critère est qu'il néglige toutes les autres classes qui peuvent s'avérer être plus adaptée aux autres classes de la CDP et donc produire un meilleur résultat.

La *sélection de la classe extraite en fonction d'autres classes* se fait en utilisant des classes de référence $C_1, \ldots, C_{k-1}, C_{k+1}, \ldots, C_K$. On considère alors $X_k(M_k, w_k, s)$ un extracteur défini par un classifieur. On note $M_k^w(D) = \left\{C_{k,1}, C_{k,2}, \ldots, C_{k,K_M^w}\right\}$ le résultat de la classification obtenue, sur les données D, par la méthode M_k et les poids w. La classe extraite par X_k est alors :

$$X_k(D) = \underset{\left\{C_{k,h} \in M_k^w(D)\right\}}{\text{Argmax}} Q\left(\{C_1, \ldots, C_{k-1}, C_{k,h}, C_{k+1}, \ldots, C_K\}\right)$$

La classe extraite est donc choisie en fonction de la qualité de la classification obtenue avec les classes de référence. Cette méthode de sélection s'étend facilement à un nombre quelconque de classes de référence.

Cette méthode de sélection peut être utilisée de deux manières différentes selon qu'il s'agisse d'évolution classique ou de coévolution coopérative. Dans le cas d'une évolution classique, la classe extraite d'un extracteur X_k va dépendre des classes obtenues par les classifieurs définissant les extracteurs pour les autres classes. Dans le cas de la coévolution coopérative, la classe extraite sera définie en fonction des classes des individus représentatifs.

Pour étendre la notion d'extracteurs définis par des classifieurs à des extracteurs flous, il faut définir une fonction d'appartenance selon la méthode employée.

Pour les méthodes produisant des classifications floues (comme Fuzzy-C-means), on peut utiliser les degrés d'appartenance aux classes données par l'algorithme, mais souvent ce degré d'appartenance est dépendant du nombre de classes car $\sum_{k=1}^{K} \mu_k(o) = 1$. En particulier, un objet atypique aura $1/K$ comme degré d'appartenance, cette valeur dépendant du nombre de classes.

Il est en fait possible de définir des extracteurs flous à partir de toute méthode de classification basée sur un centre de classe, même pour des méthodes proposant des classes dures (comme par exemple K-means, Fuzzy-C-means ou SOM). Nous proposons que le degré d'appartenance d'un objet o à un classe C_k dans une classification C soit défini en fonction du rapport entre la distance entre o et le centre de C_k et celle entre o et les centres des autres classes de l'extracteur (Définition 5.13).

DÉFINITION 5.13 (DEGRÉ D'APPARTENANCE À UNE CLASSE DÉFINIE PAR UN CENTRE)
Soit $C = \{C_1, \ldots, C_K\}$ une classification dont toutes les classes sont définies dans le même espace, c'est-à-dire selon la même pondération des attributs, et $c = \{c_1, \ldots, c_K\}$ les centres des classes de C (c_k est le centre de la classe C_k).

Le degré d'appartenance d'un objet o à la classe C_k est défini par :

$$\mu_k(o) = \begin{cases} \exp\left(-\ln\left(\frac{1}{\varphi}\right) \times \left(\frac{d(o,c_k)}{d(o,c_{min})}\right)^{\alpha}\right), \text{ si } d(o,c_{min}) \neq 0 \\ 0, \text{ sinon} \end{cases}$$

avec $c_{min} = \underset{\substack{c_i \in c \\ i \neq k}}{\text{Argmin}}\left(d(o,c_i)\right)$ *le centre de classe le plus proche de o différent de c_k et avec* $\alpha > 0$ *et* $0 < \varphi < 1$.

Le paramètre φ est le degré d'appartenance de o à C_k lorsque o se trouve à la frontière de la classe (donc que $d(o,c_k) = d(o,c_{min})$). Si l'objet o appartient à la classe C_k, c'est-à-dire que $d(o,c_k) < d(o,c_{min})$, alors $\mu_{C_k}(o) > \varphi$. Si au contraire o n'appartient pas à la classe C_k, c'est-à-dire que $d(o,c_k) > d(o,c_{min})$, alors $\mu_{C_k}(o) < \varphi$.

Le paramètre α va modifier le niveau de flou de la fonction d'appartenance : plus α est élevé, plus la classe sera «dure» (quand α tend vers $+\infty$, $\mu_k(o)$ tend vers une définition binaire de l'appartenance aux classes).

Un extracteur flou $X = (M, w, s)$ avec M une méthode basée sur les centres des classes va calculer le degré d'appartenance des objets à la classe extraite en fonction des classes non sélectionnées. Le degré d'appartenance des objets à la classe extraite se fait simplement en calculant le degré d'appartenance à $X(D)$ dans $M^w(D)$.

5.3.2 Génotype des individus et opérations génétiques

Dans une première étude, afin de valider notre approche et d'en vérifier la faisabilité et l'efficacité, nous utilisons une méthode unique, qui reste fixe au cours de l'apprentissage, pour tous les extracteurs. Seules les pondérations des attributs évoluent, les individus ne seront donc représentés que par les pondérations utilisées.

Dans une approche évolutionnaire, chaque individu encode un ensemble de K extracteurs. Le chromosome du i-ième individu à la g-ième génération $W^{i,g} = \left(W_1^{i,g}, \ldots, W_K^{i,g}\right)$ est alors constitué des poids sur chaque attributs pour chaque extracteur $W_k^{i,g} = \left(w_{k,1}^{i,g}, \ldots, w_{k,n}^{i,g}\right)$ (FIG. 5.4).

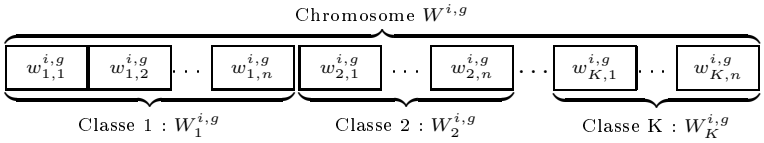


FIG. 5.4 : *Un chromosome dans une approche évolutionnaire*

Dans notre approche coévolutionnaire, chaque individu est un extracteur. Un chromosome est alors constitué des poids sur chaque attribut pour cet extracteur. On note $W_k^{i,g} = \left(w_{k,1}^{i,g}, \ldots, w_{k,n}^{i,g}\right)$ le chromosome du i-ième individu de la k-ième population à la génération g (FIG. 5.5).

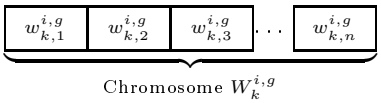


FIG. 5.5 : *Un chromosome dans une approche coévolutionnaire*

Les opérations génétiques utilisées sont les suivantes :

– initialisation : une valeur réelle (entre 0 et 1) est choisie aléatoirement pour chaque gène ;

- croisement : un croisement uniforme est utilisé car sinon l'ordre des attributs (choisi arbitrairement) aurait une influence sur les croisements, comme cela a été dit dans le chapitre 2 ;
- mutation : une nouvelle valeur réelle est générée aléatoirement pour chaque gène mutant.

5.4 Évaluation de la méthode MACLAW

Nous avons évalué l'efficacité de la méthode de la même façon que les algorithmes évolutionnaires basés sur K-means dans le chapitre 4. Nous avons réalisé les tests sur les mêmes ensembles de données selon les mêmes critères d'évaluation.

Nous allons tout d'abord présenter les paramètres utilisés dans les différentes variantes de la méthode (section 5.4.1), puis nous comparerons les différents algorithmes selon la fonction d'évaluation que nous avons défini dans ce chapitre (section 5.4.2). Nous évaluerons ensuite la pertinence des résultats de classification obtenus en les comparant à une classification optimale (section 5.4.3). Nous évaluerons également la stabilité des résultats dans la section 5.4.4, les pondérations obtenues dans la section 5.4.5, ainsi que le temps de calcul dans la section 5.4.6.

5.4.1 Configuration des algorithmes

L'algorithme MACLAW a été testé en utilisant des extracteurs définis à partir de la méthode de classification K-means ou de la méthode EM. Dans chacun des cas, le nombre de générations a été fixé à 100 et le nombre d'individus par population à 20. Le nombre de classes (et donc de populations) est fixé au nombre de classes présentes dans l'ensemble de données testé. Afin de valider notre approche, nous avons choisi de n'évaluer les algorithmes qu'en utilisant des valeurs standard pour les différents paramètres spécifiques aux algorithmes génétiques et de ne pas étudier l'influence de chacun d'eux. Nous avons donc utilisés les paramètres suivant :

- le taux de croisement est fixé à 70 % ;
- le taux de mutation est fixé à 5 % ;
- la sélection proportionnelle (*roulette wheel selection*) est utilisée.

Les autres paramètres de l'algorithme sont les suivants :

- l'algorithme utilisé pour définir les extracteurs (K-means ou EM) ;
- le type d'extracteurs (durs ou flous) ;
- le nombre d'individus représentatifs ;
- le critère de sélection de la classe extraite (par un critère fixe WG_q ou en fonction des individus représentatifs) ;
- la technique d'initialisation des individus représentatifs (par collaborations arbitraires ou par un algorithme de classification) ;
- la distance utilisée pour l'algorithme K-means (distance euclidienne, L^2, ou distance de Manhattan, L^1).

Nous avons décidé de faire varier la distance utilisée car les classes peuvent prendre des formes plus facile à découvrir dans une distance plutôt qu'une autre.

Afin d'évaluer l'influence des divers paramètres, nous les avons fait varier à partir d'une configuration standard.

Dans la configuration standard (notée *cfg. stand.*), les extracteurs sont basés sur K-means et proposent des classes dures. Le nombre d'individus représentatifs par population est fixé à 3 et la sélection de la classe extraite se fait en fonction des individus représentatifs. Ceux-ci sont initialisés par les classes obtenues par l'algorithme K-means. La distance L^2 est utilisée. Le critère de qualité interne des classes utilisé est celui de Wemmert et Gançarski.

Afin de vérifier l'importance de chacun des paramètres que nous avons définis, cinq autres configurations sont dérivées de la configuration *cfg. stand.* :

- dans la configuration *cfg. arb.*, les individus représentatifs sont initialisés par des collaborations arbitraires ;
- dans la configuration *cfg. WG*, la classe extraite est sélectionnée en fonction d'un critère fixe WG_q ;
- dans la configuration *cfg. 1 rep.*, un seul individu représentatif par population est utilisé ;
- dans la configuration *cfg. flou*, les classes extraites sont floues en utilisant la définition 5.13, avec $\varphi = 1/2$ et $\alpha = 3$ (les premières expériences ont montré que ces paramètres avaient peu d'influence sur les résultats) ;
- dans la configuration *cfg.* L^1, la distance L^1 est utilisée.

Une dernière configuration a été testée (notée *cfg. EM*). Les extracteurs sont basés sur EM et proposent des classes floues. Le nombre d'individus représentatifs par population est fixé à 3 et la sélection de la classe extraite se fait en fonction des individus représentatifs. Ceux-ci sont initialisés par les classes obtenues par l'algorithme EM. Il n'existe pas à notre connaissance un critère normalisé entre 0 et 1 permettant d'évaluer les classes fournies par EM. Les classifications ont donc été uniquement évaluées en fonction du degré de partitionnement, sans tenir compte de la qualité interne des classes. Le degré d'appartenance des objets aux classes est basé sur celui proposé par EM.

Les différentes configurations testées sont présentées sur la table 5.5.

Configuration	Algorithme	Type d'extracteurs	Nombre d'individus représentatifs	Sélection de la classe extraite	Initialisation des individus représentatifs	Distance
cfg. stand.	K-means	durs	3	individus représentatifs	K-means	L^2
cfg. arb.	K-means	durs	3	individus représentatifs	collaborations arbitraires	L^2
cfg. WG	K-means	durs	3	WG_q	K-means	L^2
cfg. 1 rep.	K-means	durs	1	individus représentatifs	K-means	L^2
cfg. flou	K-means	flous	3	individus représentatifs	K-means	L^2
cfg. L^1	K-means	durs	3	individus représentatifs	K-means	L^1
cfg. EM	EM	floue	3	individus représentatifs	EM	/

Tab. 5.5 : *Configurations de MACLAW testées*

Ces différentes configurations ont été comparées à l'algorithme K-means selon les distances L^2 et L^1 ou à l'algorithme EM (selon la méthode utilisée pour définir les extracteurs). Les méthodes testées étant toutes non déterministes, nous présentons des résultats correspondant à une moyenne sur 100 exécutions (l'écart type est précisé entre parenthèses).

5.4.2 Comparaison selon la fonction d'évaluation

Nous allons tout d'abord tester différentes configurations selon la fonction d'évaluation utilisée dans l'algorithme génétique, c'est-à-dire selon le degré de partitionnement, la qualité interne des classes et la qualité d'une CDP qui est le produit des deux précédents (Tab. 5.6). Dans cette section, nous allons comparer les différentes configurations basées sur K-means, produisant des classes dures et utilisant la distance L^2. Les autres configurations ne sont en effet pas comparables, l'évaluation du degré de partitionnement par des extracteurs flous (configuration *cfg. flou*) étant trop différents des valeurs obtenues avec des extracteurs dures et la distance utilisée pouvant également avoir une influence (configuration *cfg.* L^1).

On remarque que la configuration *cfg. WG*, où la sélection des classes se fait en fonction du critère fixe WG_q, produit toujours un moins bon résultat selon le degré de partitionnement, la

qualité interne des classes et donc sur la qualité totale (sauf sur les ensembles de données DA2 et DA3).

On voit également que, sur l'ensemble de données IRIS, la configuration *cfg. arb.*, où l'initialisation des individus représentatifs se fait par des collaborations arbitraires, et la configuration *cfg. WG* produisent un moins bon résultat pour le degré de partitionnement que les configurations *cfg. 1 rep.* et *cfg. stand.*, mais un meilleur résultat selon la qualité interne des classes. Cela semble indiquer que les deux critères d'évaluation peuvent être contradictoires sur certaines données. Il serait donc nécessaire de tester des méthodes d'optimisation multiobjectif plus complexes qu'une simple agglomération en un seul critère. La qualité totale donne ici l'avantage aux configurations *cfg. arb.* et *cfg. WG*

Sur l'ensemble DA3, la configuration *cfg. 1 rep.*, qui n'utilise qu'un seul individu représentatif, produit des résultats légèrement inférieurs aux autres configurations pour l'indice de qualité interne des classes et donc pour la qualité totale.

Algorithme	Critères d'évaluation		
	Degré de partitionnement	Qualité interne	Qualité totale
	DA1		
cfg. stand.	1,00 (0,00)	0,90 (0,01)	0,90 (0,01)
cfg. arb.	1,00 (0,00)	0,90 (0,02)	0,90 (0,02)
cfg. WG	0,92 (0,03)	0,72 (0,03)	0,66 (0,02)
cfg. 1 rep.	1,00 (0,00)	0,90 (0,01)	0,90 (0,01)
	DA2		
cfg. stand.	1,00 (0,00)	0,85 (0,00)	0,85 (0,00)
cfg. arb.	1,00 (0,00)	0,85 (0,00)	0,85 (0,00)
cfg. WG	1,00 (0,00)	0,85 (0,00)	0,85 (0,00)
cfg. 1 rep.	1,00 (0,00)	0,85 (0,00)	0,85 (0,00)
	DA3		
cfg. stand.	0,99 (0,01)	0,73 (0,03)	0,73 (0,03)
cfg. arb.	0,99 (0,01)	0,74 (0,03)	0,74 (0,03)
cfg. WG	0,99 (0,01)	0,75 (0,02)	0,75 (0,02)
cfg. 1 rep.	0,99 (0,01)	0,70 (0,02)	0,70 (0,03)
	IRIS		
cfg. stand.	0,98 (0,03)	0,73 (0,04)	0,72 (0,03)
cfg. arb.	0,95 (0,02)	0,79 (0,02)	0,75 (0,03)
cfg. WG	0,95 (0,02)	0,80 (0,02)	0,75 (0,03)
cfg. 1 rep.	0,98 (0,02)	0,73 (0,05)	0,71 (0,04)
	DIABETES		
cfg. stand.	1,00 (0,01)	0,72 (0,03)	0,71 (0,03)
cfg. arb.	1,00 (0,01)	0,71 (0,04)	0,71 (0,03)
cfg. WG	1,00 (0,01)	0,70 (0,04)	0,70 (0,04)
cfg. 1 rep.	1,00 (0,01)	0,72 (0,04)	0,71 (0,03)
	IONOSPHERE		
cfg. stand.	1,00 (0,00)	0,78 (0,01)	0,78 (0,01)
cfg. arb.	1,00 (0,00)	0,78 (0,01)	0,78 (0,01)
cfg. WG	0,97 (0,03)	0,74 (0,02)	0,72 (0,01)
cfg. 1 rep.	1,00 (0,00)	0,78 (0,01)	0,78 (0,01)
	SONAR		
cfg. stand.	1,00 (0,01)	0,79 (0,05)	0,79 (0,06)
cfg. arb.	1,00 (0,01)	0,79 (0,05)	0,79 (0,06)
cfg. WG	0,95 (0,03)	0,66 (0,06)	0,63 (0,07)
cfg. 1 rep.	1,00 (0,01)	0,80 (0,04)	0,79 (0,04)

TAB. 5.6 : *Évaluation de degré de partitionnement (extracteurs basés sur K-means)*

Les résultats sur le critère de qualité globale semble indiquer que la configuration *cfg. arb.* est la plus efficace pour l'optimisation de cette fonction d'évaluation. Cependant, le degré de partitionnement est optimal lorsque les individus représentatifs sont initialisés par la méthode K-means. Les deux critères d'évaluation (le degré de partitionnement et la qualité interne des classes) étant parfois contradictoire, il reste difficile de déterminer la meilleure configuration.

5.4.3 Comparaison selon des critères externes

Afin de vérifier la pertinence des classes obtenues, nous allons comparer les résultats des différents algorithmes avec les classes réelles des données, par les critères présentés dans l'annexe C,

de manière similaire à ce qui a été fait dans le chapitre 4 pour les algorithmes basés sur K-means. Nous rappelons qu'une valeur élevée indique une forte ressemblance entre le résultat de l'algorithme de classification et les classes réelles des données, et donc une forte pertinence des classes découverte par l'algorithme.

5.4.3.1 Extracteurs basés sur K-means

Sur la table 5.7 est présentée l'évaluation de K-means et des différentes configurations de MACLAW pour lesquelles les extracteurs sont définis à partir de la méthode K-means. On voit que sur les ensembles DA1, DA2 et DA3, MACLAW produit toujours de meilleurs résultats que K-means quelque soit la configuration utilisée. On remarque cependant que l'utilisation de la distance L^1 permet d'obtenir de meilleurs résultats sur l'ensemble DA3. Cela montre que la méthode MACLAW peut être appliquée pour diverses mesures de distance. La distance L^2 semble être généralement plus efficace pour mettre en évidence les classes.

Sur l'ensemble de données IRIS, les résultats avec les configurations *cfg. 1 rep.* et *cfg. stand.* sont meilleurs qu'avec les configuration *cfg. arb.* et *cfg. WG*. Cela semble indiquer qu'il est inutile de maximiser la qualité interne des classes si le degré de partitionnement n'est pas maximum. Or, en initialisant les individus représentatifs avec K-means, le degré de partitionnement est très élevé (il est égal à 1 dans le cas d'extracteurs durs), l'algorithme MACLAW cherche alors à maximiser la qualité des classes tout en conservant un fort degré de partitionnement, ce qui au final produit un meilleur résultat. Ce résultat laisse cependant penser qu'il est nécessaire de retravailler la fonction d'évaluation, en donnant, par exemple plus de poids au degré de partitionnement.

Sur l'ensemble de données DIABETES, les indices WG, J, FM et $F-M.$ indiquent que les résultats de MACLAW sont meilleurs que ceux de K-means. Les autres indices indiquent qu'il n'y a pas de différence ou qu'il y a une perte de qualité comparable au gain sur les premiers indices.

Sur les ensembles de données IONOSPHERE et SONAR, les indices J, FM et $F-M.$ indiquent que les résultats de MACLAW sont meilleurs que ceux de K-means. Les autres indices indiquent qu'il n'y a pas de différence ou une légère perte de qualité.

On remarque enfin que les extracteurs flous basés sur K-means produisent des résultats similaires aux extracteurs durs.

Ceci montre que l'algorithme MACLAW est efficace pour extraire les classes réelles, bien que les résultats soient mitigés pour les ensembles de données DIABETES, IONOSPHERE et SONAR. Ceci est probablement dû à l'utilisation de l'algorithme K-means qui n'est pas adapté à la structure des classes réelles dans ces ensembles de données. Des résultats médiocres avaient d'ailleurs été obtenus dans le chapitre 4 avec les méthodes de pondérations basées sur K-means pour les ensembles de données DIABETES et SONAR.

Les résultats montrent l'intérêt d'utiliser plusieurs individus représentatifs plutôt qu'un seul et de sélectionner la classe extraite d'un extracteur en fonction de ces individus représentatifs. Ils montrent également l'importance d'utiliser une méthode d'initialisation des individus représentatifs, comme l'algorithme K-means ici, plutôt que des collaborations arbitraires.

5.4.3.2 Extracteurs basés sur EM

Sur la table 5.8 est présentée l'évaluation de EM et de MACLAW en utilisant EM pour définir les extracteurs. On voit que sur les ensembles DA1, IRIS et particulièrement sur l'ensemble DA3, MACLAW produit de meilleurs résultat que EM.

Sur l'ensemble DIABETES, l'indice WG indique un meilleur résultat pour MACLAW, mais les indices J, FM et $F-M.$ indiquent le contraire.

Les résultats sont clairement plus mauvais avec MACLAW sur l'ensemble de données DA2 et similaires à ceux de EM sur les autres ensembles.

	Critères d'évaluation						
Algorithme	WG	R	J	FM	$F-M.$	$\overline{\Gamma}$	κ
				DA1			
K-means L^2	0,50 (0,02)	0,77 (0,02)	0,49 (0,02)	0,66 (0,02)	0,66 (0,02)	0,49 (0,04)	0,73 (0,02)
K-means L^1	0,45 (0,13)	0,75 (0,06)	0,47 (0,13)	0,63 (0,10)	0,63 (0,10)	0,45 (0,15)	0,71 (0,08)
cfg. stand.	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. arb.	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. WG	0,58 (0,14)	0,78 (0,04)	0,57 (0,09)	0,74 (0,06)	0,73 (0,06)	0,59 (0,09)	0,74 (0,06)
cfg. 1 rep.	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. flou	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. L^1	0,99 (0,09)	1,00 (0,04)	0,99 (0,08)	0,99 (0,07)	0,99 (0,07)	0,99 (0,10)	0,99 (0,05)
				DA2			
K-means L^2	0,88 (0,22)	0,93 (0,12)	0,87 (0,22)	0,92 (0,14)	0,92 (0,14)	0,87 (0,23)	0,92 (0,13)
K-means L^1	0,75 (0,24)	0,87 (0,13)	0,74 (0,25)	0,84 (0,16)	0,83 (0,16)	0,74 (0,25)	0,85 (0,14)
cfg. stand.	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. arb.	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. WG	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. 1 rep.	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. flou	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. L^1	0,95 (0,13)	0,97 (0,07)	0,95 (0,13)	0,97 (0,08)	0,97 (0,08)	0,95 (0,13)	0,97 (0,09)
				DA3			
K-means L^2	0,29 (0,02)	0,67 (0,01)	0,34 (0,02)	0,51 (0,02)	0,51 (0,02)	0,26 (0,03)	0,61 (0,01)
K-means L^1	0,57 (0,35)	0,80 (0,16)	0,60 (0,32)	0,70 (0,24)	0,70 (0,24)	0,55 (0,36)	0,76 (0,19)
cfg. stand.	0,62 (0,18)	0,81 (0,09)	0,60 (0,17)	0,74 (0,12)	0,74 (0,12)	0,60 (0,19)	0,78 (0,10)
cfg. arb.	0,68 (0,18)	0,84 (0,09)	0,65 (0,18)	0,78 (0,12)	0,77 (0,12)	0,65 (0,19)	0,81 (0,10)
cfg. WG	0,79 (0,17)	0,89 (0,08)	0,75 (0,18)	0,85 (0,11)	0,85 (0,12)	0,76 (0,18)	0,87 (0,10)
cfg. 1 rep.	0,46 (0,10)	0,74 (0,04)	0,46 (0,08)	0,63 (0,07)	0,63 (0,07)	0,43 (0,10)	0,69 (0,05)
cfg. flou	0,66 (0,14)	0,83 (0,06)	0,62 (0,12)	0,76 (0,08)	0,76 (0,08)	0,63 (0,13)	0,80 (0,07)
cfg. L^1	0,98 (0,08)	0,99 (0,04)	0,98 (0,09)	0,99 (0,05)	0,99 (0,05)	0,98 (0,08)	0,99 (0,05)
				IRIS			
K-means L^2	0,62 (0,06)	0,81 (0,04)	0,57 (0,04)	0,73 (0,03)	0,72 (0,04)	0,58 (0,06)	0,58 (0,07)
K-means L^1	0,64 (0,08)	0,82 (0,05)	0,60 (0,06)	0,75 (0,05)	0,75 (0,05)	0,61 (0,09)	0,61 (0,10)
cfg. stand.	0,77 (0,14)	0,88 (0,08)	0,73 (0,13)	0,84 (0,09)	0,84 (0,09)	0,76 (0,14)	0,74 (0,15)
cfg. arb.	0,57 (0,12)	0,77 (0,02)	0,57 (0,02)	0,75 (0,01)	0,73 (0,01)	0,59 (0,03)	0,54 (0,03)
cfg. WG	0,55 (0,12)	0,76 (0,01)	0,57 (0,02)	0,75 (0,02)	0,73 (0,01)	0,59 (0,03)	0,54 (0,02)
cfg. 1 rep.	0,72 (0,16)	0,86 (0,08)	0,70 (0,14)	0,82 (0,09)	0,81 (0,09)	0,72 (0,14)	0,71 (0,16)
cfg. flou	0,77 (0,17)	0,88 (0,09)	0,74 (0,13)	0,85 (0,08)	0,85 (0,09)	0,77 (0,14)	0,75 (0,16)
cfg. L^1	0,83 (0,12)	0,92 (0,06)	0,80 (0,10)	0,89 (0,06)	0,89 (0,07)	0,83 (0,11)	0,82 (0,12)
				DIABETES			
K-means L^2	0,40 (0,03)	0,56 (0,01)	0,43 (0,03)	0,60 (0,03)	0,60 (0,02)	0,12 (0,04)	0,36 (0,08)
K-means L^1	0,38 (0,01)	0,55 (0,01)	0,41 (0,01)	0,59 (0,01)	0,59 (0,01)	0,10 (0,02)	0,36 (0,01)
cfg. stand.	0,48 (0,00)	0,54 (0,00)	0,52 (0,00)	0,71 (0,00)	0,68 (0,00)	0,02 (0,00)	0,11 (0,00)
cfg. arb.	0,47 (0,03)	0,54 (0,00)	0,52 (0,00)	0,71 (0,00)	0,68 (0,00)	0,02 (0,00)	0,10 (0,02)
cfg. WG	0,48 (0,03)	0,54 (0,00)	0,52 (0,00)	0,71 (0,00)	0,69 (0,00)	0,02 (0,00)	0,10 (0,01)
cfg. 1 rep.	0,47 (0,04)	0,54 (0,00)	0,52 (0,00)	0,71 (0,00)	0,68 (0,00)	0,02 (0,00)	0,10 (0,02)
cfg. flou	0,47 (0,04)	0,54 (0,00)	0,52 (0,00)	0,71 (0,01)	0,69 (0,00)	0,02 (0,01)	0,10 (0,02)
cfg. L^1	0,47 (0,03)	0,54 (0,00)	0,51 (0,02)	0,70 (0,02)	0,68 (0,02)	0,02 (0,02)	0,12 (0,06)
				IONOSPHERE			
K-means L^2	0,43 (0,03)	0,59 (0,02)	0,46 (0,05)	0,63 (0,05)	0,62 (0,05)	0,18 (0,04)	0,17 (0,02)
K-means L^1	0,37 (0,00)	0,55 (0,00)	0,40 (0,00)	0,58 (0,00)	0,58 (0,00)	0,11 (0,00)	0,11 (0,00)
cfg. stand.	0,42 (0,00)	0,54 (0,00)	0,54 (0,00)	0,73 (0,00)	0,70 (0,00)	0,03 (0,01)	0,01 (0,00)
cfg. arb.	0,42 (0,00)	0,54 (0,00)	0,54 (0,00)	0,73 (0,00)	0,70 (0,00)	0,03 (0,01)	0,01 (0,00)
cfg. WG	0,45 (0,02)	0,57 (0,03)	0,54 (0,01)	0,73 (0,00)	0,70 (0,01)	0,11 (0,07)	0,07 (0,07)
cfg. 1 rep.	0,42 (0,00)	0,54 (0,00)	0,54 (0,00)	0,73 (0,00)	0,70 (0,00)	0,03 (0,01)	0,01 (0,00)
cfg. flou	0,43 (0,00)	0,54 (0,00)	0,54 (0,00)	0,73 (0,00)	0,70 (0,00)	0,03 (0,01)	0,01 (0,00)
cfg. L^1	0,39 (0,01)	0,56 (0,01)	0,41 (0,01)	0,58 (0,01)	0,58 (0,01)	0,12 (0,01)	0,12 (0,01)
				SONAR			
K-means L^2	0,30 (0,01)	0,51 (0,01)	0,37 (0,02)	0,55 (0,03)	0,54 (0,02)	0,02 (0,02)	0,02 (0,02)
K-means L^1	0,30 (0,03)	0,50 (0,01)	0,39 (0,04)	0,56 (0,04)	0,56 (0,04)	0,01 (0,01)	0,01 (0,01)
cfg. stand.	0,29 (0,00)	0,50 (0,00)	0,50 (0,00)	0,70 (0,00)	0,66 (0,00)	-0,01 (0,00)	-0,00 (0,00)
cfg. arb.	0,29 (0,00)	0,50 (0,00)	0,50 (0,00)	0,70 (0,00)	0,66 (0,00)	-0,01 (0,00)	-0,00 (0,00)
cfg. WG	0,30 (0,01)	0,50 (0,00)	0,48 (0,01)	0,68 (0,01)	0,65 (0,01)	-0,01 (0,00)	-0,00 (0,00)
cfg. 1 rep.	0,29 (0,00)	0,50 (0,00)	0,50 (0,00)	0,70 (0,00)	0,66 (0,00)	-0,01 (0,00)	-0,00 (0,00)
cfg. flou	0,29 (0,00)	0,50 (0,00)	0,50 (0,00)	0,70 (0,00)	0,66 (0,00)	-0,01 (0,00)	-0,00 (0,00)
cfg. L^1	0,30 (0,01)	0,50 (0,00)	0,48 (0,02)	0,68 (0,03)	0,65 (0,02)	-0,01 (0,01)	-0,00 (0,00)

TAB. 5.7 : *Évaluation de MACLAW par critères externes (extracteurs basés sur K-means)*

On remarque également une très forte variance des résultats, en particulier les ensembles de données artificiels. Cela semble indiquer une grande sensibilité aux paramètres d'initialisation (déterminés aléatoirement) de notre implantation de l'algorithme EM.

	Critères d'évaluation						
Algorithme	WG	R	J	FM	$F-M.$	$\overline{\Gamma}$	κ
				DA1			
EM	0,51 (0,00)	0,78 (0,00)	0,51 (0,00)	0,67 (0,00)	0,67 (0,00)	0,50 (0,00)	0,74 (0,00)
cfg. EM	0,56 (0,18)	0,78 (0,10)	0,55 (0,16)	0,71 (0,12)	0,70 (0,12)	0,54 (0,20)	0,73 (0,12)
				DA2			
EM	0,86 (0,22)	0,92 (0,12)	0,86 (0,22)	0,91 (0,14)	0,91 (0,15)	0,85 (0,24)	0,91 (0,14)
cfg. EM	0,74 (0,22)	0,86 (0,12)	0,74 (0,22)	0,85 (0,13)	0,84 (0,14)	0,75 (0,21)	0,83 (0,15)
				DA3			
EM	0,34 (0,20)	0,70 (0,09)	0,39 (0,18)	0,55 (0,14)	0,55 (0,14)	0,32 (0,20)	0,64 (0,11)
cfg. EM	1,00 (0,04)	1,00 (0,03)	0,99 (0,05)	1,00 (0,03)	1,00 (0,03)	0,99 (0,05)	1,00 (0,03)
				IRIS			
EM	0,77 (0,10)	0,88 (0,05)	0,72 (0,08)	0,84 (0,05)	0,84 (0,06)	0,75 (0,09)	0,75 (0,10)
cfg. EM	0,79 (0,14)	0,90 (0,07)	0,77 (0,12)	0,87 (0,07)	0,86 (0,08)	0,80 (0,12)	0,79 (0,14)
				DIABETES			
EM	0,32 (0,09)	0,54 (0,01)	0,47 (0,07)	0,65 (0,08)	0,64 (0,06)	0,04 (0,06)	0,20 (0,20)
cfg. EM	0,37 (0,07)	0,53 (0,02)	0,42 (0,07)	0,60 (0,07)	0,59 (0,06)	0,04 (0,04)	0,28 (0,12)
				IONOSPHERE			
EM	0,50 (0,00)	0,62 (0,00)	0,56 (0,00)	0,73 (0,00)	0,72 (0,00)	0,25 (0,00)	0,21 (0,00)
cfg. EM	0,50 (0,02)	0,63 (0,02)	0,52 (0,04)	0,69 (0,04)	0,68 (0,03)	0,25 (0,04)	0,23 (0,06)
				SONAR			
EM	0,28 (0,01)	0,50 (0,00)	0,34 (0,00)	0,50 (0,00)	0,50 (0,00)	0,01 (0,01)	0,01 (0,01)
cfg. EM	0,28 (0,01)	0,50 (0,01)	0,35 (0,02)	0,52 (0,03)	0,52 (0,02)	0,01 (0,01)	0,01 (0,01)

TAB. 5.8 : *Évaluation de MACLAW par critères externes (extracteurs basés sur EM)*

Les résultats obtenus en utilisant MACLAW avec l'algorithme EM sont moins significatifs que pour K-means, mais restent cependant encourageants. En effet, il n'existait pas à ce jour, à notre connaissance, de méthode permettant une pondération locale des attributs pour l'algorithme EM. De plus, ces résultats ont été obtenus sans utiliser de notion de distance, mais uniquement en fonction des calculs probabilistes de EM.

Les résultats pourront probablement être grandement améliorés en tenant compte d'un second critère de qualité interne des classes adapté au type de résultat obtenus par l'algorithme EM.

5.4.4 Stabilité des résultats

L'algorithme MACLAW étant non déterministe, les résultats peuvent différer d'une exécution à l'autre. Il est donc intéressant d'évaluer la stabilité des résultats obtenus. Comme cela a déjà été fait dans le chapitre 4 pour la famille de méthodes de pondération basées sur K-means, les résultats sont comparés deux à deux par des critères de comparaison de partitions.

5.4.4.1 Extracteurs basés sur K-means

On voit sur la table 5.9 la stabilité des différents algorithmes basés sur K-means. On voit que, excepté sur les ensembles DA3 (lorsque la distance L^2 est utilisée) et IONOSPHERE, les résultats obtenus par MACLAW sont beaucoup plus stables que ceux obtenus par K-means.

5.4.4.2 Extracteurs basés sur EM

On voit sur la table 5.10 la stabilité de EM et de MACLAW. On voit que, contrairement au cas où les extracteurs sont définis par K-means, les résultats de MACLAW en définissant les extracteurs par EM sont moins stables que les résultats de EM.

5.4.5 Comparaison des pondérations

Comme nous l'avons fait pour les méthodes de pondérations basées sur K-means dans le chapitre 4, nous allons comparer les attributs mis en valeur par l'algorithme MACLAW avec les attributs effectivement pertinents.

	Critères d'évaluation						
Algorithme	WG	R	J	FM	$F-M.$	$\overline{\Gamma}$	κ
				DA1			
K-means L^2	0,53 (0,35)	0,79 (0,16)	0,57 (0,31)	0,68 (0,23)	0,68 (0,23)	0,52 (0,35)	0,75 (0,18)
K-means L^1	0,59 (0,34)	0,81 (0,16)	0,61 (0,31)	0,71 (0,24)	0,71 (0,24)	0,57 (0,36)	0,77 (0,19)
cfg. stand.	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. arb.	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. WG	0,54 (0,26)	0,77 (0,17)	0,62 (0,25)	0,75 (0,19)	0,74 (0,19)	0,55 (0,34)	0,71 (0,21)
cfg. 1 rep.	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. flou	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. L^1	0,98 (0,12)	0,99 (0,06)	0,98 (0,11)	0,99 (0,09)	0,99 (0,09)	0,98 (0,14)	0,99 (0,07)
				DA2			
K-means L^2	0,80 (0,24)	0,89 (0,14)	0,80 (0,25)	0,87 (0,16)	0,87 (0,16)	0,79 (0,26)	0,88 (0,15)
K-means L^1	0,69 (0,23)	0,84 (0,13)	0,69 (0,24)	0,80 (0,15)	0,80 (0,16)	0,68 (0,25)	0,82 (0,14)
cfg. stand.	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. arb.	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. WG	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. 1 rep.	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. flou	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. L^1	0,94 (0,15)	0,97 (0,08)	0,94 (0,15)	0,96 (0,09)	0,96 (0,10)	0,94 (0,15)	0,96 (0,11)
				DA3			
K-means L^2	0,94 (0,16)	0,97 (0,07)	0,94 (0,15)	0,96 (0,10)	0,96 (0,10)	0,94 (0,16)	0,97 (0,09)
K-means L^1	0,61 (0,36)	0,82 (0,16)	0,64 (0,33)	0,73 (0,25)	0,73 (0,25)	0,60 (0,37)	0,79 (0,19)
cfg. stand.	0,57 (0,20)	0,78 (0,11)	0,57 (0,19)	0,71 (0,14)	0,71 (0,14)	0,53 (0,23)	0,75 (0,13)
cfg. arb.	0,59 (0,21)	0,78 (0,12)	0,58 (0,20)	0,72 (0,15)	0,72 (0,15)	0,54 (0,25)	0,75 (0,14)
cfg. WG	0,69 (0,20)	0,83 (0,11)	0,66 (0,21)	0,78 (0,14)	0,78 (0,14)	0,65 (0,24)	0,81 (0,13)
cfg. 1 rep.	0,71 (0,22)	0,85 (0,12)	0,70 (0,22)	0,80 (0,15)	0,80 (0,15)	0,69 (0,25)	0,83 (0,13)
cfg. flou	0,57 (0,22)	0,78 (0,13)	0,58 (0,21)	0,72 (0,16)	0,72 (0,16)	0,53 (0,26)	0,74 (0,15)
cfg. L^1	0,96 (0,11)	0,98 (0,05)	0,96 (0,12)	0,98 (0,07)	0,98 (0,07)	0,96 (0,12)	0,98 (0,06)
				IRIS			
K-means L^2	0,78 (0,20)	0,88 (0,12)	0,76 (0,20)	0,85 (0,13)	0,85 (0,14)	0,76 (0,22)	0,76 (0,23)
K-means L^1	0,82 (0,24)	0,91 (0,13)	0,82 (0,23)	0,89 (0,15)	0,88 (0,16)	0,81 (0,25)	0,81 (0,26)
cfg. stand.	0,74 (0,20)	0,87 (0,10)	0,73 (0,19)	0,84 (0,11)	0,83 (0,12)	0,75 (0,18)	0,73 (0,20)
cfg. arb.	0,62 (0,21)	0,95 (0,05)	0,91 (0,09)	0,95 (0,05)	0,95 (0,05)	0,91 (0,09)	0,90 (0,10)
cfg. WG	0,66 (0,21)	0,98 (0,02)	0,96 (0,03)	0,98 (0,01)	0,98 (0,01)	0,95 (0,03)	0,95 (0,03)
cfg. 1 rep.	0,69 (0,19)	0,85 (0,09)	0,70 (0,18)	0,82 (0,11)	0,81 (0,11)	0,72 (0,17)	0,70 (0,19)
cfg. flou	0,75 (0,21)	0,87 (0,11)	0,76 (0,19)	0,86 (0,11)	0,85 (0,12)	0,78 (0,19)	0,75 (0,21)
cfg. L^1	0,85 (0,20)	0,93 (0,10)	0,86 (0,19)	0,92 (0,11)	0,91 (0,12)	0,86 (0,18)	0,85 (0,20)
				DIABETES			
K-means L^2	0,66 (0,30)	0,76 (0,20)	0,69 (0,25)	0,80 (0,17)	0,79 (0,18)	0,52 (0,42)	0,64 (0,34)
K-means L^1	0,60 (0,30)	0,72 (0,22)	0,62 (0,27)	0,73 (0,20)	0,73 (0,20)	0,43 (0,44)	0,59 (0,31)
cfg. stand.	0,97 (0,04)	1,00 (0,01)	1,00 (0,01)	1,00 (0,00)	1,00 (0,00)	0,97 (0,04)	0,97 (0,04)
cfg. arb.	0,95 (0,14)	0,99 (0,02)	0,99 (0,02)	1,00 (0,01)	1,00 (0,01)	0,94 (0,17)	0,94 (0,17)
cfg. WG	0,95 (0,08)	0,99 (0,01)	0,99 (0,01)	1,00 (0,01)	1,00 (0,01)	0,95 (0,09)	0,95 (0,10)
cfg. 1 rep.	0,96 (0,07)	0,99 (0,02)	0,99 (0,02)	1,00 (0,01)	1,00 (0,01)	0,95 (0,14)	0,94 (0,15)
cfg. flou	0,92 (0,18)	0,99 (0,02)	0,99 (0,02)	0,99 (0,01)	0,99 (0,01)	0,91 (0,22)	0,90 (0,23)
cfg. L^1	0,85 (0,23)	0,94 (0,13)	0,93 (0,14)	0,96 (0,09)	0,96 (0,09)	0,82 (0,31)	0,83 (0,28)
				IONOSPHERE			
K-means L^2	0,76 (0,31)	0,83 (0,23)	0,80 (0,25)	0,87 (0,16)	0,87 (0,17)	0,64 (0,46)	0,64 (0,46)
K-means L^1	0,99 (0,01)	1,00 (0,01)	0,99 (0,01)	1,00 (0,01)	1,00 (0,01)	0,99 (0,01)	0,99 (0,01)
cfg. stand.	0,39 (0,41)	0,98 (0,05)	0,98 (0,05)	0,99 (0,02)	0,99 (0,03)	0,34 (0,39)	0,33 (0,39)
cfg. arb.	0,31 (0,42)	0,99 (0,01)	0,99 (0,01)	0,99 (0,00)	0,99 (0,00)	0,29 (0,41)	0,29 (0,40)
cfg. WG	0,43 (0,35)	0,90 (0,08)	0,90 (0,08)	0,95 (0,05)	0,94 (0,05)	0,25 (0,32)	0,23 (0,32)
cfg. 1 rep.	0,42 (0,43)	0,99 (0,01)	0,99 (0,01)	0,99 (0,00)	0,99 (0,00)	0,40 (0,42)	0,39 (0,42)
cfg. flou	0,62 (0,36)	0,99 (0,01)	0,99 (0,01)	1,00 (0,00)	1,00 (0,00)	0,58 (0,36)	0,56 (0,36)
cfg. L^1	0,91 (0,03)	0,94 (0,02)	0,89 (0,03)	0,94 (0,02)	0,94 (0,02)	0,89 (0,04)	0,89 (0,04)
				SONAR			
K-means L^2	0,59 (0,20)	0,71 (0,15)	0,62 (0,18)	0,75 (0,13)	0,75 (0,13)	0,41 (0,30)	0,41 (0,30)
K-means L^1	0,55 (0,25)	0,67 (0,19)	0,61 (0,22)	0,74 (0,15)	0,74 (0,15)	0,32 (0,38)	0,32 (0,38)
cfg. stand.	0,95 (0,10)	1,00 (0,02)	1,00 (0,02)	1,00 (0,01)	1,00 (0,01)	0,93 (0,16)	0,92 (0,19)
cfg. arb.	0,96 (0,09)	1,00 (0,01)	1,00 (0,01)	1,00 (0,00)	1,00 (0,00)	0,95 (0,12)	0,94 (0,14)
cfg. WG	0,75 (0,07)	0,94 (0,03)	0,94 (0,03)	0,97 (0,02)	0,97 (0,02)	0,52 (0,26)	0,49 (0,28)
cfg. 1 rep.	0,94 (0,16)	1,00 (0,01)	1,00 (0,01)	1,00 (0,00)	1,00 (0,00)	0,93 (0,19)	0,92 (0,20)
cfg. flou	0,94 (0,11)	1,00 (0,01)	1,00 (0,01)	1,00 (0,01)	1,00 (0,01)	0,92 (0,16)	0,90 (0,19)
cfg. L^1	0,68 (0,26)	0,93 (0,07)	0,92 (0,07)	0,96 (0,04)	0,96 (0,04)	0,48 (0,33)	0,44 (0,35)

TAB. 5.9 : *Stabilité de MACLAW (extracteurs basés sur K-means)*

Nous rappelons que, dans le cas de pondérations locales, il est nécessaire de calculer un degré global d'utilisation des attributs en fonction des pondérations locales.

Sur la table 5.11 est représenté le degré d'utilisation des attributs obtenu par l'algorithme MACLAW suivant les différentes configurations, pour l'ensemble DA1. On voit que MACLAW

	Critères d'évaluation						
Algorithme	WG	R	J	FM	$F-M.$	$\overline{\Gamma}$	κ
				DA1			
EM	0,66 (0,36)	0,84 (0,17)	0,68 (0,33)	0,76 (0,25)	0,76 (0,25)	0,64 (0,37)	0,81 (0,20)
cfg. EM	0,48 (0,24)	0,71 (0,15)	0,52 (0,21)	0,67 (0,17)	0,66 (0,17)	0,43 (0,31)	0,63 (0,21)
				DA2			
EM	0,80 (0,24)	0,89 (0,14)	0,80 (0,25)	0,87 (0,16)	0,87 (0,16)	0,79 (0,26)	0,88 (0,15)
cfg. EM	0,47 (0,23)	0,71 (0,19)	0,60 (0,24)	0,73 (0,18)	0,72 (0,18)	0,42 (0,38)	0,57 (0,31)
				DA3			
EM	0,86 (0,28)	0,94 (0,13)	0,87 (0,25)	0,91 (0,19)	0,91 (0,19)	0,86 (0,28)	0,93 (0,15)
cfg. EM	0,78 (0,25)	0,89 (0,12)	0,78 (0,24)	0,86 (0,16)	0,85 (0,16)	0,77 (0,25)	0,87 (0,14)
				IRIS			
EM	0,79 (0,19)	0,89 (0,10)	0,77 (0,19)	0,86 (0,12)	0,86 (0,12)	0,78 (0,19)	0,78 (0,20)
cfg. EM	0,75 (0,24)	0,87 (0,15)	0,78 (0,22)	0,87 (0,13)	0,86 (0,15)	0,77 (0,26)	0,75 (0,27)
				DIABETES			
EM	0,61 (0,29)	0,73 (0,24)	0,72 (0,24)	0,84 (0,15)	0,82 (0,16)	0,42 (0,45)	0,41 (0,45)
cfg. EM	0,51 (0,24)	0,64 (0,18)	0,57 (0,20)	0,72 (0,14)	0,71 (0,15)	0,28 (0,36)	0,40 (0,32)
				IONOSPHERE			
EM	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)	1,00 (0,00)
cfg. EM	0,66 (0,29)	0,73 (0,22)	0,69 (0,24)	0,81 (0,15)	0,80 (0,16)	0,40 (0,44)	? (?)
				SONAR			
EM	0,80 (0,29)	0,86 (0,20)	0,80 (0,28)	0,86 (0,20)	0,86 (0,20)	0,72 (0,41)	0,72 (0,41)
cfg. EM	0,49 (0,20)	0,64 (0,14)	0,52 (0,16)	0,68 (0,13)	0,67 (0,13)	0,26 (0,29)	0,26 (0,29)

TAB. 5.10 : *Stabilité de MACLAW (extracteurs basés sur EM)*

donne plus d'importance à l'attribut 2 qu'à l'attribut 1 qui est en effet le seul attribut pertinent. On remarque cependant que cette différence est moins flagrante dans la configuration *cfg. EM*, c'est-à-dire en définissant les extracteurs en fonction de l'algorithme EM.

	DA1	
Algorithme	attribut 1	attribut 2
cfg. stand.	0,01 (0,01)	1,00 (0,00)
cfg. arb.	0,02 (0,01)	1,00 (0,00)
cfg. WG	0,29 (0,18)	1,00 (0,07)
cfg. 1 rep.	0,01 (0,01)	1,00 (0,00)
cfg. flou	0,02 (0,01)	1,00 (0,00)
cfg. L^1	0,02 (0,10)	1,00 (0,10)
cfg. EM	0,76 (0,19)	1,00 (0,16)

TAB. 5.11 : *Degré d'utilisation des attributs par MACLAW pour l'ensemble de données* DA1 *(extracteurs basés sur K-means)*

Sur la table 5.12 est représenté le degré d'utilisation des attributs obtenu par l'algorithme MACLAW suivant les différentes configurations, pour l'ensemble DA2. On constate que MACLAW tient compte de l'attribut 1 et donne moins d'importance aux quatre attributs corrélés entre eux. Cela semble indiquer que MACLAW est capable de gérer efficacement le cas des dépendances fortes entre les attributs. On remarque cependant que dans la configuration *cfg. EM*, l'importance donnée aux attributs 2, 3, 4 et 5 est plus grande, ce qui peut expliquer les moins bon résultats en utilisant EM.

	DA2				
Algorithme	attribut 1	attribut 2	attribut 3	attribut 4	attribut 5
cfg. stand.	1,00 (0,05)	0,55 (0,15)	0,44 (0,17)	0,47 (0,17)	0,49 (0,16)
cfg. arb.	1,00 (0,06)	0,59 (0,16)	0,43 (0,19)	0,46 (0,17)	0,46 (0,18)
cfg. WG	1,00 (0,06)	0,54 (0,12)	0,40 (0,17)	0,49 (0,13)	0,48 (0,18)
cfg. 1 rep.	1,00 (0,06)	0,57 (0,15)	0,44 (0,19)	0,47 (0,20)	0,48 (0,15)
cfg. flou	1,00 (0,06)	0,61 (0,16)	0,41 (0,19)	0,52 (0,19)	0,50 (0,17)
cfg. L^1	1,00 (0,28)	0,47 (0,16)	0,49 (0,20)	0,44 (0,20)	0,52 (0,20)
cfg. EM	1,00 (0,28)	0,82 (0,20)	0,84 (0,17)	0,83 (0,23)	0,78 (0,17)

TAB. 5.12 : *Degré d'utilisation des attributs par MACLAW pour l'ensemble de données* DA2 *(extracteurs basés sur K-means)*

Sur la table 5.13 est représenté le degré d'utilisation des attributs obtenu par l'algorithme MACLAW suivant les différentes configurations, pour l'ensemble DA3. On constate que les confi-

gurations les plus efficaces (c'est-à-dire les configurations *cfg.* L^1 et *cfg. EM*) donnent autant d'importance à chacun des attributs, les attributs étant tous pertinents, mais pas tous pour la même classe.

DA3			
Algorithme	attribut 1	attribut 2	attribut 3
cfg. stand.	0,91 (0,46)	1,00 (0,43)	0,51 (0,48)
cfg. arb.	0,86 (0,51)	1,00 (0,49)	0,70 (0,46)
cfg. WG	1,00 (0,47)	0,98 (0,47)	0,81 (0,43)
cfg. 1 rep.	0,70 (0,30)	1,00 (0,27)	0,33 (0,41)
cfg. flou	0,72 (0,55)	1,00 (0,47)	0,62 (0,47)
cfg. L^1	1,00 (0,10)	0,96 (0,16)	0,88 (0,20)
cfg. EM	0,98 (0,06)	1,00 (0,05)	0,94 (0,06)

TAB. 5.13 : *Degré d'utilisation des attributs par MACLAW pour l'ensemble de données* DA3 *(extracteurs basés sur K-means)*

Enfin, sur la table 5.14 est représenté le degré d'utilisation des attributs obtenu par l'algorithme MACLAW suivant les différentes configurations, pour l'ensemble IRIS. On constate que MACLAW donne plus d'importance aux deux derniers l'attribut qui sont effectivement les plus pertinents. On remarque ici aussi cependant que cette différence est moins flagrante dans la configuration *cfg. EM*.

IRIS				
Algorithme	sepallength	sepalwidth	petallength	petalwidth
cfg. stand.	0,30 (0,37)	0,27 (0,29)	1,00 (0,21)	0,92 (0,35)
cfg. arb.	0,21 (0,17)	0,61 (0,22)	1,00 (0,13)	0,56 (0,20)
cfg. WG	0,20 (0,19)	0,70 (0,19)	1,00 (0,15)	0,63 (0,19)
cfg. 1 rep.	0,41 (0,42)	0,30 (0,32)	1,00 (0,21)	0,90 (0,36)
cfg. flou	0,23 (0,19)	0,25 (0,25)	0,96 (0,21)	1,00 (0,29)
cfg. L^1	0,11 (0,10)	0,15 (0,19)	0,95 (0,16)	1,00 (0,23)
cfg. EM	0,68 (0,19)	0,70 (0,14)	0,89 (0,08)	1,00 (0,16)

TAB. 5.14 : *Degré d'utilisation des attributs par MACLAW pour l'ensemble de données* IRIS *(extracteurs basés sur K-means)*

Ces résultats montrent que l'algorithme MACLAW est une méthode efficace de pondération d'attributs, capable de découvrir les attributs pertinents pour correctement discriminer les classes entre elles, bien que les résultats en utilisant EM pour définir les extracteurs soient moins significatifs.

5.4.6 Temps de calcul

Le coût algorithmique de MACLAW est important. En effet, à chaque génération $K \times m$ classifications sont réalisées, où m est le nombre d'individus par population. De plus, si chaque extracteur trouve K classes et que la sélection de la classe extraite se fait en fonction des individus représentatifs, on compte $p^{K-1} \times m \times K^2 + (2 \times p)^K$ CDP évaluées à chaque génération, où p est le nombre d'individus représentatifs par population.

Sur les tables 5.15 et 5.16 est présenté le temps de calcul selon différentes configurations de l'algorithme MACLAW exprimé en ms. Les tests ont tous été réalisés sur des machines dotées de processeurs Opteron à 2,4 Ghz avec 4 Go de Ram. Ces machines n'étaient cependant pas utilisées exclusivement pour l'exécution de ces algorithmes de classification ce qui peut expliquer certaines variations incohérentes.

Sans surprise, l'algorithme MACLAW est bien plus lent que les algorithmes de classification classiques. On remarque que la sélection de la classe en fonction des individus représentatif est très coûteuse en temps, en effet, le temps de calcul dans la configuration *cfg. WG*, c'est-à-dire avec sélection de la classe par un critère fixe, est le plus court.

Algorithme	Temps de calcul (ms)
DA1	
K-means L^2	2 344 (35)
K-means L^1	1 979 (2 771)
cfg. stand.	41 064 (1 390)
cfg. arb.	42 414 (1 693)
cfg. WG	35 626 (1 705)
cfg. 1 rep.	39 936 (2 332)
cfg. flou	44 909 (1 139)
cfg. L^1	43 008 (211)
DA2	
K-means L^2	2 286 (224)
K-means L^1	1 564 (259)
cfg. stand.	52 669 (486)
cfg. arb.	52 802 (863)
cfg. WG	45 174 (1 975)
cfg. 1 rep.	50 856 (426)
cfg. flou	56 567 (490)
cfg. L^1	55 360 (981)
DA3	
K-means L^2	2 448 (35)
K-means L^1	1 519 (75)
cfg. stand.	45 199 (2 070)
cfg. arb.	45 308 (434)
cfg. WG	38 296 (2 015)
cfg. 1 rep.	43 982 (4 140)
cfg. flou	49 070 (1 494)
cfg. L^1	46 393 (1 222)
IRIS	
K-means L^2	160 (5)
K-means L^1	195 (155)
cfg. stand.	12 275 (506)
cfg. arb.	12 336 (522)
cfg. WG	10 331 (354)
cfg. 1 rep.	11 567 (276)
cfg. flou	13 655 (2 493)
cfg. L^1	13 094 (2 279)
DIABETES	
K-means L^2	2 998 (1 376)
K-means L^1	2 624 (13)
cfg. stand.	41 994 (365)
cfg. arb.	42 321 (1 196)
cfg. WG	38 945 (188)
cfg. 1 rep.	41 870 (1 764)
cfg. flou	42 210 (95)
cfg. L^1	44 928 (1 853)
IONOSPHERE	
K-means L^2	772 (13)
K-means L^1	797 (18)
cfg. stand.	43 094 (1 220)
cfg. arb.	43 551 (915)
cfg. WG	40 381 (1 615)
cfg. 1 rep.	43 001 (1 395)
cfg. flou	43 328 (1 436)
cfg. L^1	47 537 (1 533)
SONAR	
K-means L^2	458 (6)
K-means L^1	505 (157)
cfg. stand.	41 563 (768)
cfg. arb.	42 265 (1 213)
cfg. WG	39 221 (841)
cfg. 1 rep.	41 477 (379)
cfg. flou	41 756 (869)
cfg. L^1	46 477 (1 057)

TAB. 5.15 : *Temps de calcul de MACLAW (extracteurs basés sur K-means)*

Ces temps de calcul très long sont compensés par le fait que la structure modulaire permet une parallélisation aisée. Une étude concernant la parallélisation de MACLAW est en cours de validation.

Algorithme	Temps de calcul (ms)
DA1	
EM	1 666 (180)
cfg. EM	105 397 (589)
DA2	
EM	1 857 (363)
cfg. EM	298 896 (3 773)
DA3	
EM	1 729 (44)
cfg. EM	167 987 (594)
IRIS	
EM	255 (139)
cfg. EM	56 764 (1 688)
DIABETES	
EM	3 033 (226)
cfg. EM	280 825 (2 505)
IONOSPHERE	
EM	1 688 (1 301)
cfg. EM	525 382 (15 931)
SONAR	
EM	1 398 (414)
cfg. EM	556 088 (9 523)

TAB. 5.16 : *Temps de calcul de MACLAW (extracteurs basés sur EM)*

5.5 Conclusion

Dans ce chapitre, nous avons proposé une nouvelle approche pour la classification non supervisée appelée approche modulaire. Cette approche consiste à diviser un problème de classification en K classes en K sous-problèmes d'extraction d'une classe. Les classes sont extraites par des extracteurs définis chacun par une stratégie qui lui est propre. L'apprentissage est alors réalisé par un algorithme de coévolution coopérative, en optimisant un critère d'évaluation basé sur la complémentarité des classes et leur qualité interne.

Cette approche a été utilisée pour réaliser une classification non supervisée avec pondération locale des attributs par approche enveloppe. Pour cela, les extracteurs ont été définis à partir de méthodes de classification classiques. Bien que chacun des extracteurs utilise une pondération globale des attributs différente, chaque classe du résultat global est définie en fonction d'une pondération des attributs qui lui est spécifique. Ainsi, il est possible de découvrir des pondérations locales quelque soit la méthode de classification utilisée.

Les résultats expérimentaux ont montré l'efficacité de l'algorithme MACLAW, malgré des temps de calcul très longs. En effet, notre algorithme a permis d'obtenir de meilleurs résultats sur la plupart des ensembles de données sur lesquels il a été testé et a pu mettre en évidence les attributs les plus importants pour la discrimination des classes. L'algorithme MACLAW s'est montré particulièrement efficace face à des données fortement corrélées. De plus, nous avons montré que MACLAW peut être utilisé avec diverses méthodes de classification : il n'utilise pas de notion de distance et n'utilise que des pondérations globales pour extraire les classes.

L'approche modulaire semble donc être une solution adéquate pour la classification non supervisée. Des travaux futurs pourront être menés sur d'autres méthodes de classification basées sur l'approche modulaire.

Application

Chapitre 6

Utilisation de MACLAW dans le cadre de l'observation de la Terre

6.1 Introduction

Cette thèse s'inscrit dans le cadre d'une collaboration entre le LSIIT et le Laboratoire Image et Ville (LIV, ULP/CNRS UMR 7011), en particulier dans le cadre de l'ACI « Masse de Données » FoDoMuSt[1]. L'objectif premier de ce projet est d'étudier et de définir des méthodes et outils permettant une utilisation conjointe de plusieurs sources de connaissances et d'images lors de l'identification, la localisation et la formalisation des éléments du tissu urbain (surfaces minéralisées, végétation, eau). Dans le cadre de cette ACI, l'objectif global est de proposer un processus complet de sélection, d'extraction et d'interprétation de connaissances à partir de bases de données d'images et de connaissances du domaine considéré.

Une image de télédétection est composée d'un ensemble d'images en niveaux de gris en deux dimensions. Le niveau de gris d'un pixel correspond à la réponse spectrale de la surface observée, sur une bande spectrale, c'est-à-dire sur la plage de longueurs d'onde captée. Une description de la télédétection et de ses applications est présentée dans l'annexe D. L'extraction de l'information contenue dans une image de télédétection peut être réalisée manuellement par un photo-interprète. Ce processus d'interprétation visuelle est cependant consommateur de temps, d'autant plus que le volume de données augmente avec les nouvelles technologies. Il est, de plus, particulièrement subjectif. L'automatisation de l'extraction de l'information devient alors une nécessité.

Cette extraction automatique d'informations se fait principalement par des techniques de traitement d'images et de fouille de données, en particulier la classification. La classification peut être réalisée à deux niveaux différents [Puissant, 2003] :

- classification spectrale (au niveau des pixels) : la classification spectrale consiste à découvrir la classe de chaque pixel de l'image en fonction de ses caractéristiques spectrales (et éventuellement en fonction de celles des pixels de son voisinage) ;
- classification zonale (au niveau des objets) : la classification zonale consiste à découvrir la classe de zones (également appelées régions ou segments) obtenues par un processus de segmentation, correspondant chacune à un objet de la scène, et caractérisées selon différents attributs (information sur la radiométrie, la forme ou la texture).

L'arrivée de la très haute résolution spatiale, pour une étude plus fine du tissu urbain, a mis à mal les méthodologies classiquement employées par les géographes dans le cadre de l'observation de la Terre. En effet, celles-ci sont principalement axées sur l'utilisation de méthodes supervisées. Or,

[1] FoDoMuSt regroupe des chercheurs du LSIIT, du LIV et du laboratoire ERIC (de l'Université Lumière – Lyon2).

la définition d'exemples est un processus fastidieux, d'une part à cause du manque d'informations de l'expert sur la nature des classes dans l'image, dû à l'apparition de nombreux détails à ce niveau de résolution (voitures, maisons individuelles), et, d'autre part, à cause de la taille des images qui imposent de définir un grand nombre d'exemples, ce qui est particulièrement consommateur de temps.

Une nouvelle méthodologie proposée en observation de la Terre, et en particulier par les experts du LIV, consiste à utiliser des mécanismes de classification non supervisée comme première analyse des données. Cela permet à l'expert d'obtenir des informations sur la structure des classes et peut l'aider au paramétrage d'algorithmes supervisés. De plus, des classes découvertes par un algorithme de classification non supervisée et identifiées par l'expert peuvent être utilisées pour sélectionner des pixels (ou des régions) caractéristiques des classes recherchées. Ces pixels ou régions peuvent alors être utilisés comme exemples d'apprentissage pour un algorithme de classification supervisée.

Parallèlement, l'analyse d'images de télédétection s'est grandement complexifiée avec l'apparition de nombreux capteurs à résolution décamétrique et hyperspectraux. Les images multispectrales (capteurs Landsat, IRS, SPOT, Quickbird, Ikonos) ne comportent que quelques bandes discontiguës, avec une résolution spectrale large, allant jusqu'à 1000 nm. La résolution spectrale dans les images hyperspectrales (capteurs AVIRIS, HyMap, CASI, DAIS) peut descendre jusqu'à 2 nm avec plusieurs dizaines de bandes contiguës. Les capteurs hyperspectraux sont généralement sensibles sur une région spectrale allant du visible à l'infrarouge lointain. Ces images ont généralement une résolution spatiale assez fine (3 m ou moins). De telles images apportent des problèmes nouveaux en télédétection :

- une augmentation forte de la dimensionnalité des données, c'est-à-dire du nombre de bandes spectrales ;
- des corrélations fortes entre les bandes liées à la contiguïté de celles-ci.

De même, dans le cadre de la classification zonale, la volonté (voire la nécessité) de décrire les segments d'une image par des attributs nombreux (apportant des informations sur la radiométrie, la forme ou encore la texture) nous ramène au problème de leur pertinence pour la classification.

Dans ce contexte, nous proposons de vérifier la validité de l'algorithme MACLAW dont l'objectif est justement de traiter ce type de problèmes. Pour cela, nous allons présenter les résultats de quatre expériences qui ont été menées.

Dans la section 6.2, nous présenterons une expérience préliminaire réalisée avec l'une des premières implantations de notre algorithme. Cette section a pour but unique d'illustrer le comportement de notre algorithme.

Dans la section 6.3, nous comparerons l'algorithme MACLAW avec les méthodes habituellement utilisées au LIV. En particulier, nous évaluerons sa capacité à déterminer les bandes spectrales les plus pertinentes pour discriminer les classes, et ce, malgré les corrélations entre les différentes bandes.

Dans la section 6.4, nous montrerons la capacité de l'algorithme MACLAW à classifier des régions, malgré des imperfections de la segmentation et l'hétérogénéité des attributs décrivant ces régions.

Enfin, dans la section 6.5, nous présenterons une expérience mettant en évidence une faiblesse du critère d'évaluation utilisé dans l'algorithme MACLAW.

Contexte des expériences

Ces expériences ont été menées au sein du LIV à partir de données fournies par leurs experts. Les extraits des images utilisés dans nos tests ont été choisis de sorte qu'ils présentent les caractéristiques type des scènes habituellement traitées par ces experts. De plus, chaque résultat a été vu, étudié et validé par ces mêmes experts. Il est à noter que l'expérience présentée dans la section 6.3 a donné lieu à une publication dans une conférence internationale en télédétection, en collaboration avec des membres du LIV [Wania *et al.*, 2006].

Nous nous sommes restreints à comparer l'algorithme MACLAW aux outils habituellement utilisés par les membres du LIV, en particulier les algorithmes intégrés dans le logiciel ENVI 4.2 (*Environment for Visualizing Images*, *Research Systems Inc.*). D'autres méthodes de fouille de données, probablement plus efficaces, ne sont que peu utilisées par les géographes car elles ne sont pas intégrées dans les logiciels d'analyse d'images de télédétection. De plus, la complexité (réelle ou supposée) de leur paramétrisation demande un certaine connaissance de chacune des méthodes. C'est d'ailleurs pourquoi, nous avons fait en sorte, dans nos expériences, de n'utiliser aucun paramétrage autre que le nombre de classes. Les autres paramètres sont fixés aux valeurs par défaut définies dans le chapitre 5.

6.2 Expérience préliminaire

Nous allons commencer par présenter les résultats obtenus par l'une des premières versions de notre implantation de MACLAW sur un extrait d'une image DAIS. Nous avons choisi de présenter ces résultats car ils permettent de visualiser comment notre algorithme fait évoluer les différentes classes pour arriver au résultat final.

Dans cette implantation, un seul individu représentatif par population était utilisé. Les individus représentatifs étaient initialisés par collaborations arbitraires. La sélection de la classe extraite ne se fait que par critère fixe.

L'algorithme a été paramétré pour rechercher cinq classes. Chacune des populations est composée de 150 individus. Les extracteurs sont basés sur l'algorithme K-means. L'apprentissage s'est déroulé sur 50 générations.

Nous n'analyserons pas les résultats en détail, mais uniquement l'évolution des classes au fil des générations, afin d'aider à comprendre le comportement de l'algorithme. Les caractéristiques des données obtenues par un capteur DAIS seront présentées en détail dans la section 6.3. Nous préciserons seulement qu'il s'agit d'un extrait de 152×156 pixel, constitué de 44 bandes qui ont été sélectionnées parmi un total de 79. Certaines de ces bandes sont bruitées. De plus, au moment de l'expérience, une erreur de lecture rendait certaines bandes incohérentes. Deux des bandes sont présentées sur la figure 6.1 : une bande non bruitée (Fig. 6.1(a)) et une bande bruitée par une erreur de lecture (Fig. 6.1(b)). Sur cet extrait se trouve une zone de bâti sur la gauche, un petit stade au centre et un cours d'eau, traversé par un pont, sur la droite.

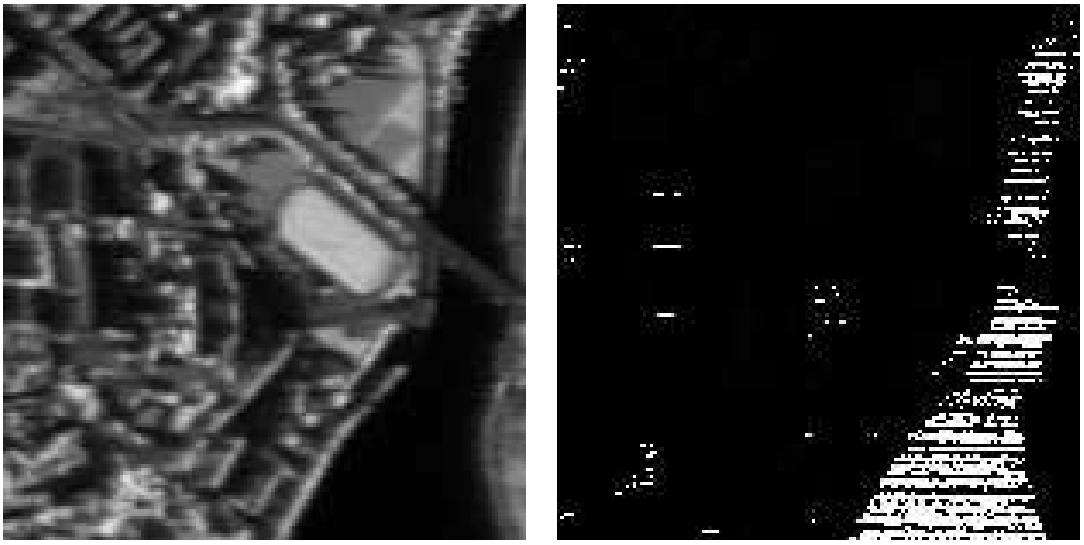
Sur la figure 6.2, nous pouvons observer l'évolution du critère de qualité au cours des générations. Nous remarquons que l'évolution présente quatre phases :

- de l'initialisation à la génération 12, une forte amélioration de la qualité de la classification ;
- de la génération 13 à 28, une amélioration très lente, voire une stagnation de cette qualité ;
- de la génération 29 à 37, une nouvelle augmentation très forte de la qualité ;
- de la génération 38 à 50, à nouveau une augmentation très lente.

Sur la figure 6.3 sont représentés les résultats correspondant à ces différents points d'inflexion. Les classes correspondant aux individus représentatifs de chaque population sont représentées. L'ensemble de ses classes forment le meilleur résultat courant.

Nous expliquons cette évolution de la manière suivante :

- dans la première période (de l'initialisation à la génération 12), l'amélioration provient principalement de la spécialisation de chacune des populations de classifieurs. On constate que les individus représentatifs des populations 1 et 4 se « partagent » la classe (réelle) d'eau ;
- dans la seconde période (de la génération 13 à la génération 28), les spécialisations trouvées dans l'étape précédente n'ont plus permis d'évolution significative de la qualité, ceci étant vraisemblablement dû à la présence d'un maximum local ;
- à la génération 29, la population 3 se spécialise dans une classe radicalement nouvelle par rapport à la génération précédente, mais aussi, et surtout, par rapport aux autres classes de la génération 29. Ceci explique la première inflexion de la courbe après cette phase de



(a) *Bande non bruitée* **(b)** *Bande bruitée*

Fig. 6.1 : *Bandes de l'image DAIS*

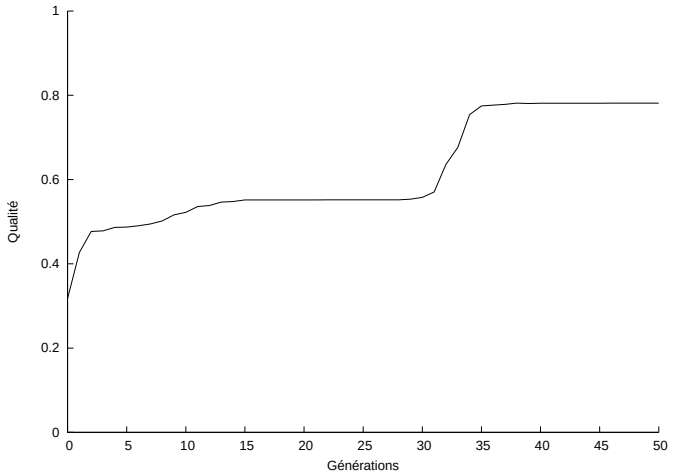


Fig. 6.2 : *Évolution de la qualité*

stagnation. En effet cette nouvelle spécialisation a permis de classifier des objets qui ne l'étaient pas encore ;

- cette amélioration est immédiatement suivie à la génération 31 par une nouvelle amélioration encore plus forte due au fait que la population 1 propose aussi une classe radicalement différente. Cette spécialisation a, de plus, supprimé le « conflit » entre les individus représentatifs de la populations 1 et de la population 4 ;
- les différentes populations ont plus ou moins conservé leur niche jusqu'à la fin de l'apprentissage, qui n'a plus consisté alors qu'à affiner les classes proposées. Par exemple la population 4 a pu regrouper l'ensemble des pixels que nous savons être de l'eau.

D'une façon plus générale, nous observons à la génération 46 que des classes intéressantes ont été découvertes. Ainsi, la population 1 a bien identifié les routes, la population 2 les ombres, la population 4 l'eau et la population 5 la végétation (le stade et des espaces verts). La population 3

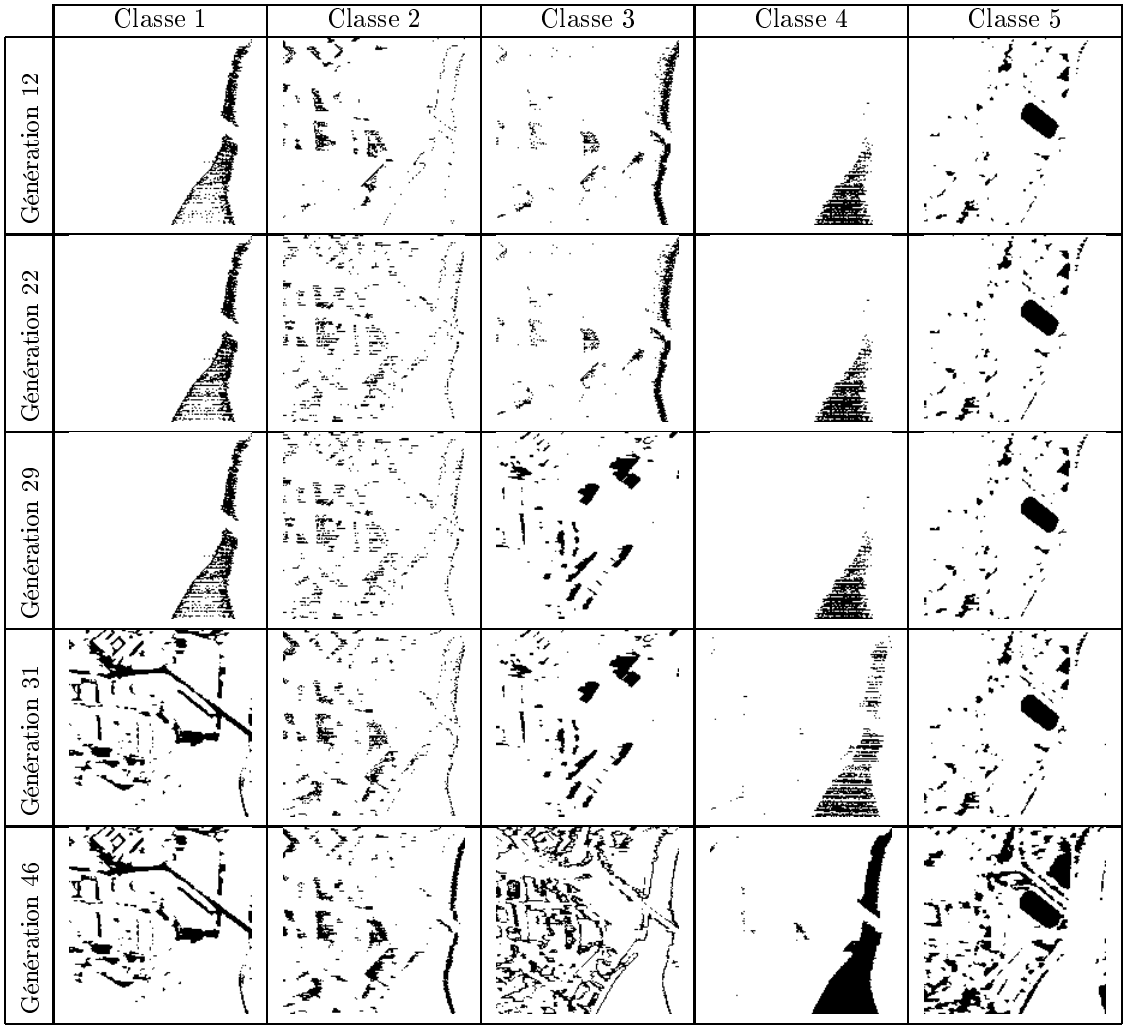


Fig. 6.3 : *Évolution des classes extraites*

a mis en évidence les pixels de bordure, difficiles à classifier. Le bâti n'a pas été mis en évidence et correspond à l'ensemble des pixels non classifiés.

6.3 Expérimentations sur une image DAIS

Le but de cette série de tests est triple. Il s'agit tout d'abord de vérifier la capacité de MACLAW à extraire des classes identifiables par l'expert et sémantiquement correctes, en fonction d'une vérité-terrain (section 6.3.2). Nous allons également vérifier la validité des pondérations trouvées par MACLAW par une analyse d'expert et par comparaison avec d'autres méthodes d'évaluation des attributs (section 6.3.3). Enfin, nous allons tester la possibilité d'inclure MACLAW dans un processus plus large d'extraction de connaissances à partir d'une image de télédétection, en montrant que les pondérations découvertes peuvent être utilisées pour améliorer la qualité d'une classification supervisée[2] (section 6.3.4).

Nous commencerons cependant par présenter les données (section 6.3.1).

[2]Les classifications supervisées ont été réalisées par Annett Wania du LIV.

6.3.1 Description des données

Cette première série de tests concerne une image DAIS (*Digital Airborne Imaging Spectrometer*, *Spaceimaging*) de la ville de Strasbourg acquise le 17 juillet 1999 à 16 h 50 GMT. L'image est composée de 79 bandes allant du spectre visible à l'infrarouge thermique (477–14 208 nm). La résolution spatiale est de 3 m. La résolution spectrale varie de 2 nm dans le spectre visible à 1 000 nm dans l'infrarouge lointain. Il s'agit d'une image brute, non corrigée, la valeur des pixels représentant une mesure de radiance. Cette image a été fournie par l'équipe Télédétection, Radiométrie et Imagerie Optique (TRIO) du LSIIT.

Le nombre de bandes a cependant été réduit à 40. En effet, 39 bandes étaient inutilisables et ont été retirées manuellement. Les tests ont été réalisés sur un extrait de l'image de 100×65 pixels (les images en niveaux de gris de quatre bandes sont représentées sur la figure 6.4). Cet extrait correspond à une zone dans la banlieue de Strasbourg correspondant à une transition entre une zone résidentielle (sur le haut de l'image) et une zone industrielle (bâtiments au centre et excavations pleines d'eau au bas de l'image). Les deux zones sont séparées par une route.

La bande 3 (Fig. 6.4(a)) correspond à une région du visible (vert) : les surfaces les plus claires correspondent aux toits et à l'asphalte de la route, indiquant une forte réflectance de ces matériaux dans le visible.

Sur les deux bandes du proche et du moyen infrarouge 22 et 29 (Fig. 6.4(b) et 6.4(c)) on voit que la réflectance est particulièrement élevée sur les zones de végétation et particulièrement basse sur les autres zones (bâtiments, route, eau).

Sur la bande d'infrarouge lointain 38 (Fig. 6.4(d)), les surfaces chaudes (toits et routes) sont les plus visibles.

(a) *Bande 3, VIS (534 nm)*

(b) *Bande 22, NEAR–IR (886 nm)*

(c) *Bande 29, MID–IR (1 690 nm)*

(d) *Bande 38, FAR–IR (11 266 nm)*

Fig. 6.4 : *Extrait de l'image DAIS sur quatre bandes*

Sur la figure 6.5 est représentée la photographie aérienne correspondant à l'extrait utilisé dans ces tests, provenant de la base de données d'orthophotographie de l'IGN (Institut Géographique

National) du 5 mai 1998. Il s'agit d'une photographie, correspondant donc à la lumière visible (450–900 nm), la résolution spatiale est de 0,5 m. Cette photographie n'est pas utilisée par les algorithmes de classification, mais uniquement pour faciliter l'évaluation visuelle des résultats. Il faut d'ailleurs remarquer que les ombres présentes sur une image dépendent de la date et de l'heure d'acquisition.

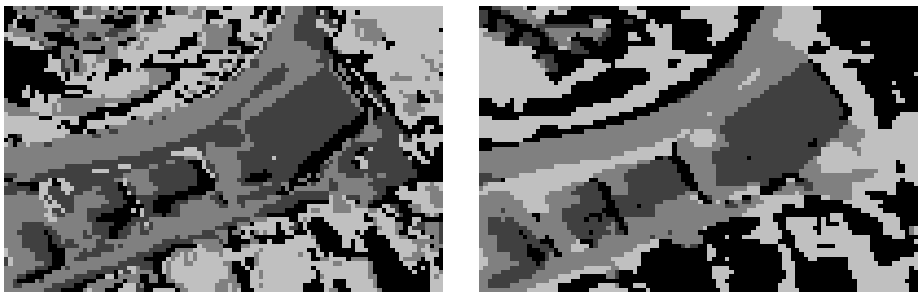
Fig. 6.5 : *Photographie aérienne correspondant à l'extrait de l'image DAIS*

Nous disposons de plus, pour cette image, d'une vérité-terrain pour quatre classes. La vérité-terrain est composée de 48 pixels pour une classe d'ombre/d'eau, 550 pixels pour une classe de bâti, 122 pour une classe de route et 526 pour une classe de végétation. Cette vérité-terrain nous permettra d'estimer l'efficacité des algorithmes.

6.3.2 Évaluation du résultat de classification de MACLAW

L'algorithme MACLAW a été appliqué sur l'extrait de l'image DAIS avec 40 bandes. Il a été configuré pour utiliser des extracteurs basés sur K-means avec trois individus représentatifs par population initialisés par l'algorithme K-means. Les classes extraites ont été sélectionnées en fonction des individus représentatifs. Il y avait 20 individus dans chaque population et 100 générations. Il a été convenu avec les experts que l'algorithme devait chercher quatre classes dans l'image.

Sur la figure 6.6 sont représentés les résultats de classification initial et final obtenus par MACLAW. Sur la table 6.1 sont représentées les matrices de confusion entre les résultats de classification initial et final de MACLAW et la vérité-terrain. Le résultat initial (Fig. 6.6(a)) est celui obtenu par l'algorithme K-means lors de l'initialisation des individus représentatifs. Le résultat final est celui obtenu après optimisation de la fonction d'évaluation (Fig. 6.6(b)).



(a) *Résultat initial (obtenu par K-means)*

(b) *Résultat final (obtenu par MACLAW)*

Fig. 6.6 : *Résultat de l'algorithme MACLAW sur l'image DAIS*

Malgré certaines erreurs de classification, les classes obtenues par MACLAW sont plus facilement identifiables que les classes initiales obtenues par K-means. La classe 1 correspond à un

	Vérité-terrain (en %)				
	ombre eau	bâti	route	vég.	total
classe 1	35,4	4,7	23,0	47,3	25,7
classe 2	0,0	81,1	9,0	2,5	37,7
classe 3	35,4	14,0	67,2	7,8	17,4
classe 4	29,2	0,2	0,8	42,4	19,2

(a) *Résultat initial*

	Vérité-terrain (en %)				
	ombre eau	bâti	route	vég.	total
classe 1	68,8	0,4	2,5	33,5	17,2
classe 2	31,3	87,5	27,9	0,4	42,7
classe 3	0,0	12,2	67,2	0,2	12,0
classe 4	0,0	0,0	2,5	66,0	28,1

(b) *Résultat final*

TAB. 6.1 : *Matrices de confusion entre les résultats de classification initial et final de MACLAW et la vérité-terrain*

mélange de deux classes difficilement séparables : l'ombre et l'eau. La classe 2 a été identifiée par un expert comme étant du bâti, la classe 3 comme représentant les routes et la classe 4 comme étant une classe de végétation.

Sur la table 6.2 est représentée l'évaluation de la qualité par comparaison avec la vérité-terrain selon les divers indices présentés dans l'annexe C. Nous rappelons que pour chacun de ces indices, une valeur élevée indique une forte ressemblance entre le résultat de l'algorithme de classification et les classes réelles des données, et donc une forte pertinence des classes découvertes par un algorithme.

	Critères d'évaluation						
Algorithme	WG	R	J	FM	$F-M.$	$\overline{\Gamma}$	κ
K-means	0,38	0,76	0,47	0,65	0,64	0,48	0,74
MACLAW	0,48	0,82	0,59	0,75	0,74	0,62	0,80

TAB. 6.2 : *Évaluation des résultats de K-means et de MACLAW sur l'image DAIS par critères externes*

Sur chacun des indices, on voit une très nette amélioration entre le résultat initial (obtenu par l'algorithme K-means) et le résultat final après optimisation des pondérations par l'algorithme MACLAW, même s'il reste des imperfections. Ces résultats confirment la comparaison visuelle.

6.3.3 Évaluation des pondérations obtenues par MACLAW

Sur la table 6.3 sont représentées les pondérations locales obtenues par l'algorithme MACLAW sur l'image DAIS ainsi que le degré d'utilisation globale de chacune des bandes. Pour chaque classe, les poids des quatre bandes les plus utilisées sont indiquées en gras. De plus, le degré d'utilisation des neuf bandes les plus utilisées sont également indiquées en gras.

Ces pondérations ont été analysées par un expert du LIV (section 6.3.3.1) et comparées à des mesures d'évaluation de la pertinence des attributs (section 6.3.3.2).

6.3.3.1 Analyse par un expert

Pour la première classe, l'eau reflète très peu la lumière, en particulier dans l'infrarouge. Cette faible réflectance dans l'infrarouge permet de bien distinguer l'eau des autres classes. Les poids les plus importants obtenus par MACLAW correspondent aux bandes de l'infrarouge. L'importance donnée aux bandes 34, 36 et 40 correspond à la faible température des zones ombragées.

Pour la classe de bâti, les bandes les plus utilisées sont des bandes de l'infrarouge lointain (FAR–IR, bandes 34 et 36). Ce type de surface absorbe le rayonnement solaire d'où une température élevée, ce qui a pour effet de réfléchir particulièrement dans l'infrarouge lointain (souvent appelé

	Bande	Moyenne en nm	Pondérations locales Ombre/eau	Bâti	Route	Végétation	Degré d'utilisation
VIS	1	498	0,69	0,78	0,73	0,43	0,75
	2	516	0,76	0,15	0,87	**0,95**	**0,95**
	3	534	0,18	0,02	0,51	0,44	0,44
	4	553	0,06	0,78	0,63	**1,00**	**1,00**
	5	571	0,20	0,32	0,28	0,28	0,31
	6	589	0,09	0,11	0,48	0,24	0,39
	7	607	0,77	0,21	0,21	0,79	0,78
	8	625	0,74	0,29	0,65	0,23	0,73
	9	642	0,05	0,68	**0,96**	0,83	0,81
	10	659	0,59	0,12	0,77	0,41	0,63
	11	678	0,24	0,30	0,33	0,30	0,29
	12	696	0,69	0,05	0,16	0,29	0,68
NEAR–IR	13	711	0,64	0,13	0,30	0,04	0,63
	14	745	0,80	0,80	**1,00**	0,36	0,81
	15	764	0,63	**0,92**	0,57	0,94	**0,93**
	16	781	**1,00**	0,71	0,79	0,41	**0,98**
	17	799	0,30	0,47	0,33	0,36	0,44
	18	815	0,05	**0,90**	0,53	0,09	0,85
	19	833	0,13	0,61	0,26	0,54	0,58
	20	850	0,65	0,36	0,11	0,62	0,64
	21	868	0,31	0,67	0,10	0,63	0,64
	22	886	0,12	0,51	0,44	0,17	0,49
	23	904	0,08	0,46	0,78	0,50	0,64
	24	922	0,52	0,11	0,88	0,26	0,73
	25	993	0,65	0,87	0,42	0,59	0,81
	26	1 014	0,38	0,84	0,57	**0,99**	**0,98**
	27	1 024	0,06	0,14	0,73	0,09	0,59
	28	1 037	0,76	0,18	0,54	**0,96**	**0,95**
MID–IR	29	1 690	0,09	0,00	0,75	0,69	0,68
	30	1 740	0,33	0,13	0,27	0,12	0,32
	31	2 124	0,07	0,49	0,03	0,06	0,46
	32	2 141	0,05	0,08	0,71	0,06	0,59
	33	2 171	0,15	0,06	**0,93**	0,09	0,76
FAR–IR	34	4 371	**0,87**	**1,00**	0,45	0,46	**0,95**
	35	8 747	0,73	0,44	0,88	0,22	0,73
	36	9 648	**0,91**	**0,96**	0,27	0,30	**0,92**
	37	10 482	0,09	0,48	**0,97**	0,07	0,80
	38	11 266	0,38	0,65	0,28	0,20	0,63
	39	11 997	0,46	0,75	0,12	0,19	0,71
	40	12 668	**0,98**	0,32	0,06	0,91	**0,97**

Tab. 6.3 : *Pondérations obtenues par MACLAW sur l'image DAIS*

infrarouge thermique). Les résultats de MACLAW semblent donc cohérents. L'algorithme donne également beaucoup d'importance à deux bandes du proche infrarouge, qui permettent de séparer efficacement le bâti de la végétation. On remarque par contre que les bandes du visible sont très peu utilisées, ce qui semble indiquer qu'elles sont moins importantes pour la discrimination du bâti.

Pour la classe de route, l'algorithme a mis en évidence une bande du spectre visible, une du proche infrarouge, une du moyen infrarouge et une de l'infrarouge lointain. Tout comme c'était le cas pour la classe de bâti, le proche infrarouge et l'infrarouge lointain sont particulièrement importants pour identifier la classe.

Pour la classe de végétation, la bande la plus utilisée est la bande 4 qui correspond au spectre vert (et donc à la couleur verte de la végétation). Des bandes de proche infrarouge sont également beaucoup utilisées. En effet, la végétation présente une forte réflectance dans le proche infrarouge en raison de la présence de chlorophylle. On remarque également, de façon moins marquée, qu'une des bandes rouges (la bande 9) a également été mise en évidence. Or la végétation présente une réflectance particulièrement faible dans le rouge. L'algorithme MACLAW semble donc avoir identifié cette différence entre rouge et proche infrarouge, spécifique à la végétation. Cette différence est généralement utilisée pour définir l'indice de végétation NDVI (*Normalized Difference Vegetation Index*) par les télédétecteurs [Rouse *et al.*, 1973].

Ces résultats indiquent que MACLAW est capable de mettre en évidence les bandes intéressantes généralement utilisées en télédétection pour discriminer les différentes classes entre elles. On remarque de plus que les bandes voisines (donc très corrélées) ont souvent des poids très différents. Ceci indique la robustesse de l'algorithme MACLAW face aux données fortement corrélées.

6.3.3.2 Comparaison avec d'autres critère d'évaluation des attributs

Nous avons comparé les résultats obtenus par MACLAW avec différents indices de pertinence des attributs utilisés généralement en prétraitement dans une approche filtre pour la sélection d'attributs. Nous avons calculé quatre indices supervisés I, DM, IS et J, présentés dans l'annexe B, et un indice non supervisé E présenté dans le chapitre 3. Les divers indices supervisés sont basés sur la notion de distance entre les classes. L'indice non supervisé E est basé sur une mesure d'entropie. Les indices DM et IS permettent d'évaluer la pertinence des attributs pour chacune des classes et de manière globale. Les indices J et E sont des indices de pertinence d'une pondération. L'évaluation de l'importance d'un attribut F_j se fait en évaluant la qualité du sous-ensemble $F \setminus F_j$: plus le sous-ensemble est « mauvais », plus l'attribut est important.

Les tables 6.4 et 6.5 montrent le classement des bandes, ordonnées de la plus pertinente à la moins pertinente selon les indices DM et IS. Les bandes sont classées spécifiquement pour chacune des classes, mais aussi de manière globale pour l'ensemble des données. Les neuf bandes les plus importantes sont mises en évidence (pour chaque classe et pour l'ensemble des données).

	Classements locaux			Classement
Ombre/Eau	Bâti	Route	Végétation	global
13	40	35	37	37
29	1	37	40	35
12	2	40	35	40
5	37	38	21	1
3	39	39	38	2
4	35	36	19	38
11	38	1	34	39
7	34	34	39	3
9	3	2	20	34
33	9	10	24	9
15	10	9	22	10
6	8	8	23	11
14	7	11	18	7
10	11	7	17	8
2	4	3	36	36
8	36	5	16	5
32	5	6	15	4
30	6	4	1	6
1	12	12	2	17
31	15	33	25	15
16	17	21	28	18
17	18	20	27	19
18	16	19	26	16
35	19	24	14	21
19	21	18	10	20
20	20	17	9	24
37	14	23	8	12
21	24	22	3	23
40	22	16	7	22
24	23	15	11	14
22	31	31	5	33
23	33	32	6	31
36	32	14	4	32
38	25	25	12	25
39	28	13	31	13
34	28	26	32	28
25	27	27	33	27
27	13	28	29	29
28	26	29	30	26
26	30	30	13	30

Tab. 6.4 : *Classement des bandes selon l'indice DM*

La table 6.6 montre le classement des bandes, ordonnées de la plus pertinente à la moins pertinente pour l'ensemble des données selon les indices I, J et E. Les neuf bandes les plus importantes sont mises en évidence (pour chacun des indices).

On voit que, sur chacune de ces mesures, des bandes proches les unes des autres (et donc très corrélées) ont un classement à peu près similaire. En effet, ces mesures prennent en compte les attributs indépendamment, négligeant les dépendances entre eux.

	Classements locaux			Classement
Ombre/Eau	Bâti	Route	Végétation	global
33	31	31	31	31
32	32	32	32	32
31	33	33	33	33
30	30	34	34	34
29	34	30	30	30
34	29	29	40	29
13	24	40	29	40
24	25	39	39	24
12	26	38	1	39
1	23	25	35	1
11	27	36	2	2
25	28	26	38	25
4	22	24	37	3
14	21	27	10	26
3	2	28	36	38
2	19	35	9	11
7	20	37	11	12
15	18	23	3	4
23	3	2	7	10
9	17	1	4	27
10	1	3	12	23
17	4	22	8	9
18	39	12	5	7
26	14	4	6	35
27	15	13	13	13
16	16	11	24	36
28	36	7	25	28
19	13	10	14	37
22	40	21	26	22
5	12	19	27	14
20	7	9	23	19
40	11	18	28	18
21	10	14	15	21
8	38	17	17	17
6	9	20	18	15
39	6	6	22	20
35	35	15	19	6
38	37	16	21	8
36	5	5	16	5
37	8	8	20	16

Tab. 6.5 : *Classement des bandes selon l'indice IS*

De plus, les résultats semblent influencés par l'ordre de grandeur des attributs. Un grande importance est accordée aux bandes du proche infrarouge et de l'infrarouge lointain. Or, sur ses bandes, l'ordre de grandeur des valeurs (et par conséquence des différences entre les valeurs) est plus élevée que sur les bandes du visible. On remarque enfin qu'il n'y a que peu de différence entre les « meilleurs » attributs spécifiques à chaque classe (mise à part la classe d'ombre et d'eau pour l'indice DM).

Ces résultats diffèrent grandement de ceux obtenus par MACLAW. En effet, notre algorithme est peu sensible aux corrélations entre les bandes et donne rarement un poids important à deux bandes trop corrélées.

6.3.4 Résultats avec une méthode de classification supervisée

Une classification supervisée a été réalisée en utilisant une approche proposée par le logiciel ENVI.

La classification se base sur des pixels purs ou spectres de référence (ang. *endmembers*) obtenus par une technique de réduction des dimensions spatiales et spectrales [Kruse et Boardman, 2004 ; RIS, 2004]. Les spectres de référence sont des signatures spectrales pures, c'est-à-dire d'un seul objet et non pas d'une composition de plusieurs objets comme c'est le cas des pixels mixtes. Ces spectres de référence serviront d'exemples d'apprentissage pour un algorithme de classification supervisée.

La méthodologie de classification suit la procédure suivante :

Indice I	Indice J	Indice E
20	20	20
21	21	21
19	19	16
16	22	19
18	18	22
22	16	18
17	17	17
23	23	23
15	15	15
8	37	24
28	8	14
24	24	25
27	28	26
14	6	27
26	5	28
5	35	31
25	27	32
37	38	33
6	26	34
10	36	30
9	14	29
1	9	13
7	25	2
38	10	3
11	2	12
35	1	4
2	7	10
3	3	11
4	11	7
36	39	9
40	4	40
39	40	9
12	12	39
13	34	6
29	13	5
34	31	38
30	32	8
33	33	36
31	30	35
32	29	37

TAB. 6.6 : *Classement des bandes selon les indices I, J et E*

Étape 1 – décorrélation des données

Les données sont décorrélées par l'application d'une transformation par MNF (*Minimum Noise Fraction*). Une transformation par MNF consiste à réaliser deux ACP en cascade. La première ACP va décorréler les données et supprimer le bruit. La seconde est appliquée aux données nettoyées du bruit. Les dix bandes obtenues présentant les plus fortes valeurs propres sont sélectionnées pour les autres étapes.

Étape 2 – identification des spectres de référence

Les pixels les plus purs sont sélectionnés selon un indice de pureté pour définir les spectres de référence et sont projetés sur les dix bandes sélectionnées après la transformation par MNF. Un filtre MTMF (*Mixture Tuned Matched Filtering*) est appliqué pour identifier les différents spectres de référence [Boardman, 1998].

Étape 3 – classification

La classification est réalisée par l'algorithme SAM (*Spectral Angle Mapper*) qui consiste à affecter la classe correspondant au spectre de référence le plus proche selon une mesure d'angle spectral [Yuhas *et al.*, 1992]. L'utilisateur doit décider d'un seuil sur cette mesure de dissimilarité pour décrire les classes. Ce seuil peut être différent d'une classe à une autre. Il peut rester des objets non classifiés après cette phase de classification.

Des tests ont été réalisés en utilisant les 40 bandes (section 6.3.4.1) ou différents sous-ensembles d'attributs (section 6.3.4.2). Les sous-ensembles d'attributs ont été définis soit selon les pondérations obtenues par l'algorithme MACLAW, soit selon l'un des indices d'évaluation DM, IS, I, J ou E.

6.3.4.1 Résultat de classification supervisée avec les 40 bandes

Le résultat de la classification supervisée sur l'image DAIS en utilisant les 40 bandes est présenté sur la figure 6.7. Les pixels non classifiés sont représentés en blanc.

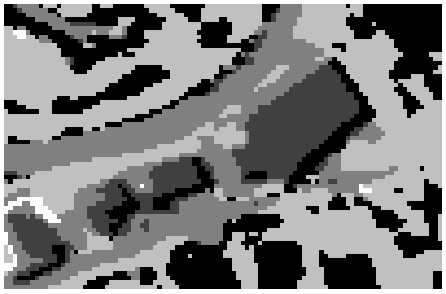


Fig. 6.7 : *Résultat de la classification supervisée avec les 40 bandes*

Sur la table 6.7 est représentée la matrice de confusion entre la classification supervisée et les exemples d'apprentissage, ainsi que le pourcentage d'erreur de commission et d'omission pour chaque classe. Comme c'est souvent le cas sur ce type de données, les confusions les plus importantes concernent le bâti et la route, ainsi que l'ombre et toutes les autres classes. On remarque que 1 % des pixels de la vérité-terrain ne sont pas classifiés. Il s'agit de pixels présentant une radiance très forte dans le visible. La signature spectrale de ces pixels est trop éloignée de chacun des spectres de référence, ils ne sont donc pas classifiés.

	Vérité-terrain (en %)				
	ombre eau	bâti	route	vég.	total
ombre eau	95,8	3,6	20,5	16,0	14,1
bâti	2,1	86,0	9,8	0,2	39,1
route	0,0	10,4	55,7	0,4	10,2
vég.	2,1	0,0	3,3	83,4	35,6
non classifié	0,0	0,0	10,7	0,0	1,0
Commission	73,7	2,9	46,5	1,1	
Omission	4,2	14,0	44,3	16,5	

Tab. 6.7 : *Matrice de confusion entre le résultat de classification supervisée avec les 40 bandes et les exemples d'apprentissage*

Nous avons comparé le résultat de la classification supervisée avec les exemples d'apprentissage selon les mêmes indices que ceux utilisés pour comparer les résultats des classifications non supervisées, présentés dans l'annexe C (Tab. 6.8).

Critères d'évaluation						
WG	R	J	FM	$F-M.$	$\overline{\Gamma}$	κ
0,46	0,87	0,68	0,81	0,81	0,72	0,84

Tab. 6.8 : *Évaluation des résultats de l'algorithme supervisé avec 40 bandes sur l'image DAIS par critères externes*

Ces résultats de classification peuvent paraître faibles pour un algorithme supervisé, comparativement à ce que l'on observe généralement dans la littérature. Il s'agit cependant d'une méthodologie récente, développée dans le cadre des images hyperspectrales, et utilisée au sein du LIV. Nous n'avons donc pas cherché à vérifier si d'autres méthodologies supervisées étaient plus

efficaces, mais nous nous sommes restreint à étudier l'intérêt de l'algorithme MACLAW dans le cadre des outils utilisés classiquement en observation de la Terre.

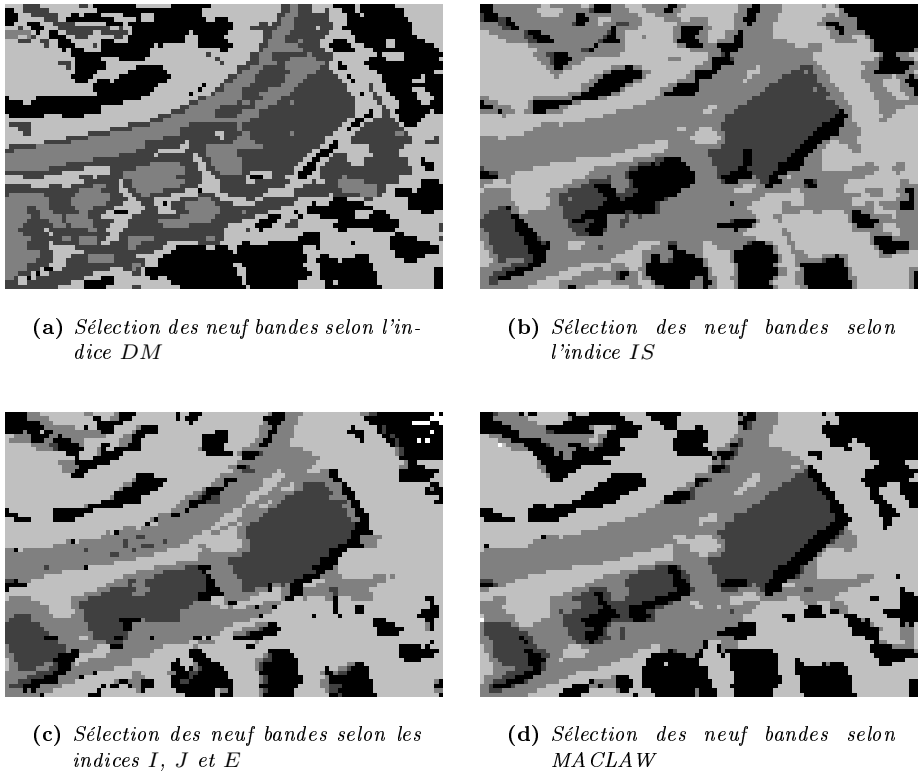
Sans surprise, le résultat de la classification supervisée est de meilleure qualité que celui de l'algorithme MACLAW. Cependant, si l'on tient compte du fait que notre algorithme est non supervisé, le résultat de celui-ci est relativement satisfaisant.

6.3.4.2 Résultats de classification supervisée avec différents sous-ensembles des bandes

Nous avons essayé de vérifier la pertinence des pondérations obtenues par MACLAW et l'efficacité des indices d'évaluation des attributs, en appliquant les divers résultats à l'algorithme de classification supervisée.

Cependant l'algorithme de classification supervisée utilisé ne permet d'utiliser qu'une sélection globale des attributs. Il ne sera donc pas possible de tester les pondérations obtenues par MACLAW, mais uniquement une sélection de bandes basée sur les pondérations obtenues. Il ne sera pas possible non plus de réaliser une sélection des bandes différente pour chaque classe, mais uniquement une sélection globale en fonction du degré d'utilisation des bandes dans MACLAW.

Il a été convenu avec les experts de réaliser les différentes classifications en utilisant les neuf meilleures bandes selon les indices DM, IS, I, J et E, et selon le degré d'utilisation dans MACLAW. Les neuf meilleures bandes pour les indices I, J et E sont identiques. L'algorithme de classification a été appliqué sur les mêmes spectres de référence, avec les mêmes paramètres que pour la classification avec les 40 bandes. Les résultats sont présentés sur la figure 6.8. Les résultats des classifications avec neuf bandes sont comparés au résultat de classification avec 40 bandes, selon divers indices de comparaison présentés dans l'annexe C (Tab. 6.9).



(a) *Sélection des neuf bandes selon l'indice DM*

(b) *Sélection des neuf bandes selon l'indice IS*

(c) *Sélection des neuf bandes selon les indices I, J et E*

(d) *Sélection des neuf bandes selon MACLAW*

Fig. 6.8 : *Résultat de la classification supervisée avec neuf bandes*

Méthode de sélection des bandes	Critères d'évaluation						
	WG	R	J	FM	$F-M.$	$\overline{\Gamma}$	κ
DM	0,24	0,71	0,31	0,48	0,48	0,27	0,68
IS	0,47	0,78	0,46	0,63	0,63	0,48	0,76
I, J et E	0,32	0,81	0,54	0,70	0,70	0,57	0,80
MACLAW	0,72	0,94	0,82	0,90	0,90	0,86	0,94

Tab. 6.9 : *Comparaison entre les résultats de classification supervisée avec neuf bandes et le résultat de la classification supervisée avec 40 bandes*

Le résultat obtenu en utilisant les bandes sélectionnées selon l'indice DM (Fig. 6.8(a)) est très incohérent. Une partie du bâti est classée en tant que route, une partie de la végétation en tant que bâti et une partie l'eau en tant que végétation. Le résultat obtenu en utilisant les bandes sélectionnées selon l'indice IS (Fig. 6.8(b)) est meilleur que le précédent, mais une partie du bâti est classée en tant qu'ombre. Dans ces deux résultats aucun pixel ne reste non classifié, mais les résultats sont visiblement moins bons que celui obtenu en utilisant les 40 bandes.

Le résultat obtenu en utilisant les bandes sélectionnées selon les indices I, J et E est bien meilleur que ceux obtenus en utilisant les indices DM et IS (Fig. 6.8(c)). Le résultat de classification est assez différent de celui obtenu avec les 40 bandes. On remarque par endroit des incohérences (pixels de route classés dans le bâti, pixels d'eau classés dans le bâti ou la route) et plusieurs pixels de la classe d'eau non classifiés.

Le résultat obtenu en utilisant les bandes sélectionnées selon le degré d'utilisation dans MACLAW (Fig. 6.8(d)) est très similaire à celui obtenu en utilisant les 40 bandes spectrales. La plupart des pixels non classifiés dans l'image avec 40 bandes sont maintenant affectés à la classe de route.

Sur la table 6.10 sont représentées les matrices de confusion entre les différentes classifications supervisées (avec neuf bandes) et les exemples d'apprentissage, ainsi que le pourcentage d'erreurs de commission et d'omission pour chaque classe. Les matrices de confusion semblent confirmer les observations visuelles. Les résultats de classification obtenus en utilisant les bandes sélectionnées selon les indices DM et IS sont peu satisfaisants.

On remarque que le résultat obtenu en utilisant les bandes sélectionnées selon les indices I, J et E semble meilleur que celui obtenu en utilisant les bandes sélectionnées selon MACLAW.

Nous avons comparé les résultats de ces différentes classifications supervisées avec les exemples d'apprentissage selon les mêmes indices que ceux utilisés pour comparer les résultats des classifications non supervisées, présentés dans l'annexe C (Tab. 6.11).

On voit que les indices DM et IS sont peu pertinents. Les classifications obtenues en sélectionnant les bandes selon ces indices dégradent les résultats par rapport à une classification utilisant toutes les bandes.

Les résultats de classification sont légèrement améliorés en sélectionnant les bandes selon les indices I, J et E. On voit enfin qu'en sélectionnant les bandes selon les résultats de MACLAW, la qualité reste inchangée sur la majorité des critères, excepté sur l'indice WG pour lequel la qualité est sensiblement améliorée.

Ces résultats montrent l'efficacité de notre méthode de classification avec pondération d'attributs dans le cadre des images hyperspectrales. Le résultat de MACLAW est en effet nettement supérieur à celui d'une classification classique par l'algorithme K-means.

De plus, nous avons pu voir que les bandes mises en évidence par MACLAW étaient pertinentes et contiennent suffisamment d'informations pour la discrimination des quatre classes par un algorithme supervisé, bien que d'autres critères semblent plus efficaces. On notera cependant que les tests ont été réalisés avec un algorithme supervisé n'autorisant que la sélection globale d'attributs. Il n'a donc pas été possible d'utiliser toutes les informations fournies par l'algorithme MACLAW, c'est-à-dire une pondération locale des attributs.

	Vérité-terrain (en %)				
	ombre eau	bâti	route	vég.	total
ombre eau	66,7	0,2	7,4	39,0	19,8
bâti	0,0	74,2	8,2	39,0	34,35
route	0,0	23,6	69,7	0,2	17,3
vég.	33,3	2,0	14,8	58,9	28,5
non classifié	0,0	0,0	0,0	0,0	0,0
Commis.	87,0	4,7	60,7	12,7	
Omis.	33,3	25,8	30,3	41,1	

(a) *Sélection des neuf bandes selon l'indice DM*

	Vérité-terrain (en %)				
	ombre eau	bâti	route	vég.	total
ombre eau	79,2	14,9	17,2	2,1	12,2
bâti	10,4	79,6	11,5	0,2	36,8
route	10,4	5,5	66,4	21,3	18,3
vég.	0,0	0,0	4,9	76,4	32,7
non classifié	0,0	0,0	0,0	0,0	0,0
Commis.	75,0	4,4	67,5	1,5	
Omis.	20,8	20,4	33,6	23,6	

(b) *Sélection des neuf bandes selon l'indice IS*

	Vérité-terrain (en %)				
	ombre eau	bâti	route	vég.	total
ombre eau	68,8	0,0	11,5	6,3	6,4
bâti	25,0	93,5	31,2	0,4	45,4
route	0,0	6,6	52,5	4,4	9,9
vég.	6,3	0,0	4,9	89,0	38,3
non classifié	0,0	0,0	0,0	0,0	0,0
Commis.	58,8	9,2	43,1	3,4	
Omis.	2,1	12,2	42,6	18,4	

(c) *Sélection des neuf bandes selon les indices I, J et E*

	Vérité-terrain (en %)				
	ombre eau	bâti	route	vég.	total
ombre eau	97,9	3,3	20,5	17,4	14,6
bâti	2,1	87,8	9,8	0,2	39,9
route	0,0	8,9	57,4	0,8	9,9
vég.	0,0	0,0	12,3	81,6	35,6
non classifié	0,0	0,0	0,0	0,0	0,0
Commis.	74,2	2,8	48	1,9	
Omis.	31,3	6,6	47,5	11,0	

(d) *Sélection des neuf bandes selon MACLAW*

TAB. 6.10 : *Matrice de confusion entre les résultats de classification supervisée avec neuf bandes et les exemples d'apprentissage*

	Critères d'évaluation						
Méthode de sélection des bandes	WG	R	J	FM	$F-M.$	$\overline{\Gamma}$	κ
DM	0,40	0,77	0,48	0,66	0,65	0,51	0,75
IS	0,45	0,82	0,57	0,74	0,73	0,61	0,79
I, J et E	0,48	0,88	0,73	0,85	0,85	0,75	0,87
MACLAW	0,51	0,86	0,67	0,81	0,80	0,70	0,84

TAB. 6.11 : *Évaluation des résultats de l'algorithme supervisé avec neuf bandes sur l'image DAIS par critères externes*

6.4 Expérimentations sur une image Quickbird

Cette seconde série de tests a été réalisée dans le cadre de la classification de régions, sur une image Quickbird segmentée afin de montrer la capacité de notre méthode à classifier des régions, malgré des imperfections de la segmentation et l'hétérogénéité des attributs décrivant ses régions. Nous commencerons par présenter les données (section 6.4.1). Nous présenterons ensuite les résultats de classification avec les algorithmes K-means et MACLAW (section 6.4.2).

6.4.1 Description des données

Cette dernière série de tests concerne une image Quickbird de la ville de Strasbourg acquise le 2 mai 2002. Il s'agit d'une image fusionnée à partir d'une image panchromatique (résolution

spatiale de 0,7 m) et quatre bandes spectrales (450–520 nm, 530–590 nm, 630–690 nm et 770–900 nm, résolution spatiale de 2,8 m) selon une technique décrite dans [Ranchin *et al.*, 2003]. La fusion permet d'obtenir une image multispectrale à 0,7 m de résolution spatiale. Les tests ont été réalisés sur un extrait de 900×900 sur lequel se trouve un parc avec de la végétation et un plan d'eau en haut à gauche de l'image, un cours d'eau en bas à droite et un quartier résidentiel entre les deux (FIG. 6.9(a)).

Cette image a été segmentée par l'algorithme Probashed. Il s'agit d'un algorithme de ligne de partage des eaux basé sur les degrés d'appartenance des pixels dans une classification floue [Derivaux *et al.*, 2006]. L'image segmentée est présentée sur la figure 6.9(b)[3].

(a) *Image Quickbird fusionnée* **(b)** *Image Quickbird segmentée*

FIG. 6.9 : *Image Quickbird*

La segmentation est une des principales opérations de traitement d'image. Elle consite à découper une image en zones homogènes (selon un certain critère) connexes, également appelées régions ou segments [Gonzalez et Woods, 1993 ; Cocquerez et Philipp, 1995]. Le critère d'homogénéité est généralement basé sur le niveau de gris, la couleur ou la réponse spectrale (suivant le type d'images), ou sur la texture. Les différentes régions peuvent être caractérisées par divers attributs.

Des informations spectrales peuvent être utilisées pour décrire une région, comme par exemple la moyenne des réponses spectrales des pixels qui composent la région sur chacune des bandes de l'image.

Les indices de forme donnent des indications sur la géométrie des régions, en particulier sur l'élongation ou la compacité de celles-ci. Les indices les plus couramment utilisés dans le cadre des images de télédétection sont [Cauvin et Rimbert, 1976 ; Puissant, 2003] :

- le coefficient de compacité de Gravélius : $\frac{P}{2\sqrt{\pi S}}$;
- l'indice de circularité de Miller : $\frac{4\pi S}{P^2}$;
- l'indice de forme de Morton : $\frac{S}{\pi\left(\frac{L}{2}\right)^2}$;
- l'indice d'élongation « simple » : $\frac{S}{L^2}$.

Ces indices sont basés sur des caractéristiques simples qui sont l'élongation L de la région (distance entre les deux points les plus éloignés), sa surface S et son périmètre P.

[3] La segmentation a été réalisée par Sébastien Derivaux de l'équipe AFD du LSIIT.

Une dernière famille d'indices concerne la texture des régions. Dans [Marceau *et al.*, 1990], les auteurs proposent d'utiliser des indices basés sur une matrice de cooccurrence des niveaux de gris des pixels d'une région. Ces indices ont été généralisés aux images multispectrales en construisant la matrice de cooccurence à partir d'une quantification vectorielle des données, réalisée, par exemple, par l'algorithme K-means [Hauta-Kasari *et al.*, 1996]. On considère quatre indices calculés à partir d'une matrice de cooccurrence $Cooc$ de taille m :

- l'homogénéité : $\sum_{i=0}^{m-1}\sum_{j=0}^{m-1}\frac{Cooc(i,j)}{1+|i-j|}$;
- le contraste : $\sum_{i=0}^{m-1}\sum_{j=0}^{m-1} Cooc(i,j)\times(i-j)^2$;
- l'entropie : $\sum_{i=0}^{m-1}\sum_{j=0}^{m-1} Cooc(i,j)\times ln\left(Cooc(i,j)\right)$;
- le second moment angulaire : $\sum_{i=0}^{m-1}\sum_{j=0}^{m-1} Cooc(i,j)^2$.

Ces caractéristiques apportent des informations variées pour la classification des régions. Il existe cependant des corrélations entre certains attributs (entre les différents indices de forme ou entre ceux de texture, par exemple). De plus, les différences d'échelle entre les attributs ne sont pas nécessairement adaptées à une bonne classification. Certains attributs sont peut-être inutiles, d'autres bruités car la segmentation n'est pas parfaite. Une solution possible à ces problèmes est la pondération d'attributs.

L'algorithme MACLAW a été appliqué sur les régions de l'image segmentée, en fusionnant les régions de taille inférieure à 50 pixels à une des régions voisines, formant ainsi 501 régions. Une classification non supervisée par l'algorithme K-means et une fenêtre de 11×11 pixels ont été utilisées pour calculer les indices de texture.

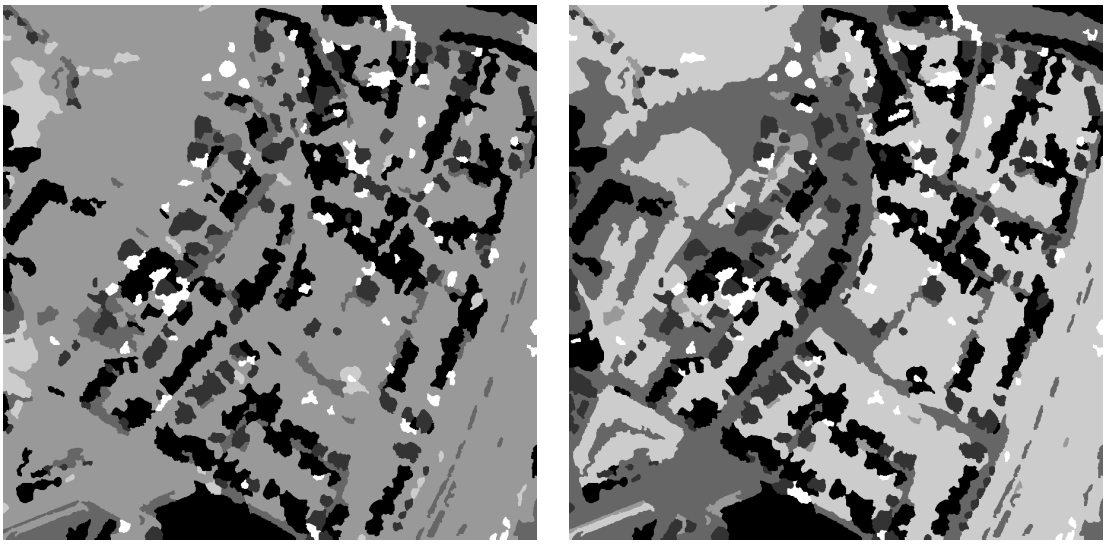
6.4.2 Résultats avec MACLAW

L'algorithme MACLAW a été appliqué sur les régions de l'image Quickbird segmentée. L'algorithme a été configuré pour utiliser des extracteurs basés sur K-means avec trois individus représentatifs initialisés par K-means. Les classes extraites ont été sélectionnées en fonction des individus représentatifs. L'algorithme cherchait six classes. Il y avait 20 individus dans chaque population et 100 générations.

Sur la figure 6.10 sont représentés les résultats de classification initial et final obtenus par MACLAW. Le résultat initial (Fig. 6.10(a)) est celui obtenu par l'algorithme K-means en initialisant les individus représentatifs. Le résultat final est celui obtenu après optimisation de la fonction d'évaluation (Fig. 6.10(b)).

On observe une très nette différence entre ces deux résultats. Exceptée une classe de bâti, les classes ne sont pas identifiables sur le résultat de classification avec K-means (Fig. 6.10(a)). En revanche, sur le résultat de classification de MACLAW, malgré des erreurs de classification, il est possible d'identifier deux classes de bâti, deux classes de route et une classe de végétation. La dernière classe obtenue n'est cependant pas identifiable. L'algorithme n'a pas réussi à mettre en évidence la classe d'eau, confondue avec la végétation (le canal en bas à droite) ou la route (le plan d'eau en haut à gauche) (Fig. 6.10(b)).

Sur la table 6.12 sont indiquées les pondérations découvertes pour chacune des classes. On remarque avec étonnement que la radiométrie n'intervient que très peu dans la discrimination des classes. En revanche un attribut de forme et un attribut de texture sont mis en évidence pour chacune des classes (exceptée la deuxième classe de route qui utilise très peu les indices de forme). L'indice de forme de Miller et plus particulièrement le contraste ont des pondérations très faibles quel que soit la classe. Cela semble indiquer que ces deux attributs ne sont pas indispensables pour



(a) *Résultat initial* **(b)** *Résultat final*

Fig. 6.10 : *Résultat de l'algorithme MACLAW sur l'image Quickbird*

la classification des régions (soit ils ne sont pas pertinents, soit ils sont trop corrélés avec d'autres attributs).

Attribut	Pondérations locales					
	Bâti (1)	Bâti (2)	Route (1)	Route (2)	Végétation	Non identifiée
Moyenne radiométrique	0,17	0,27	0,05	0,16	0,31	0,62
Coefficient de compacité de Gravélius	0,84	0,98	0,13	0,15	1,00	0,72
Indice de circularité de Miller	1,00	0,62	0,25	0,25	0,16	0,15
Indice de forme de Morton	0,20	0,22	0,41	0,26	0,05	0,22
Indice d'élongation « simple »	0,00	0,17	1,00	0,10	0,55	0,84
Homogénéité	0,50	0,34	0,52	0,25	0,26	1,00
Contraste	0,02	0,11	0,03	0,07	0,07	0,00
Entropie	0,88	0,42	0,88	0,36	0,82	0,77
Second moment angulaire	0,19	1,00	0,79	1,00	0,19	0,93

Tab. 6.12 : *Pondérations obtenues par MACLAW sur l'image Quickbird*

6.5 Expérimentations sur une image CASI

Cette dernière série de tests a été réalisée sur une image hyperspectrale (sur un extrait d'une image CASI) et a mis en évidence une faiblesse du critère d'évaluation qui est utilisé dans notre algorithme. Nous commencerons par présenter les données (section 6.5.1). Nous présenterons ensuite les résultats de classification de l'algorithme MACLAW (section 6.5.2).

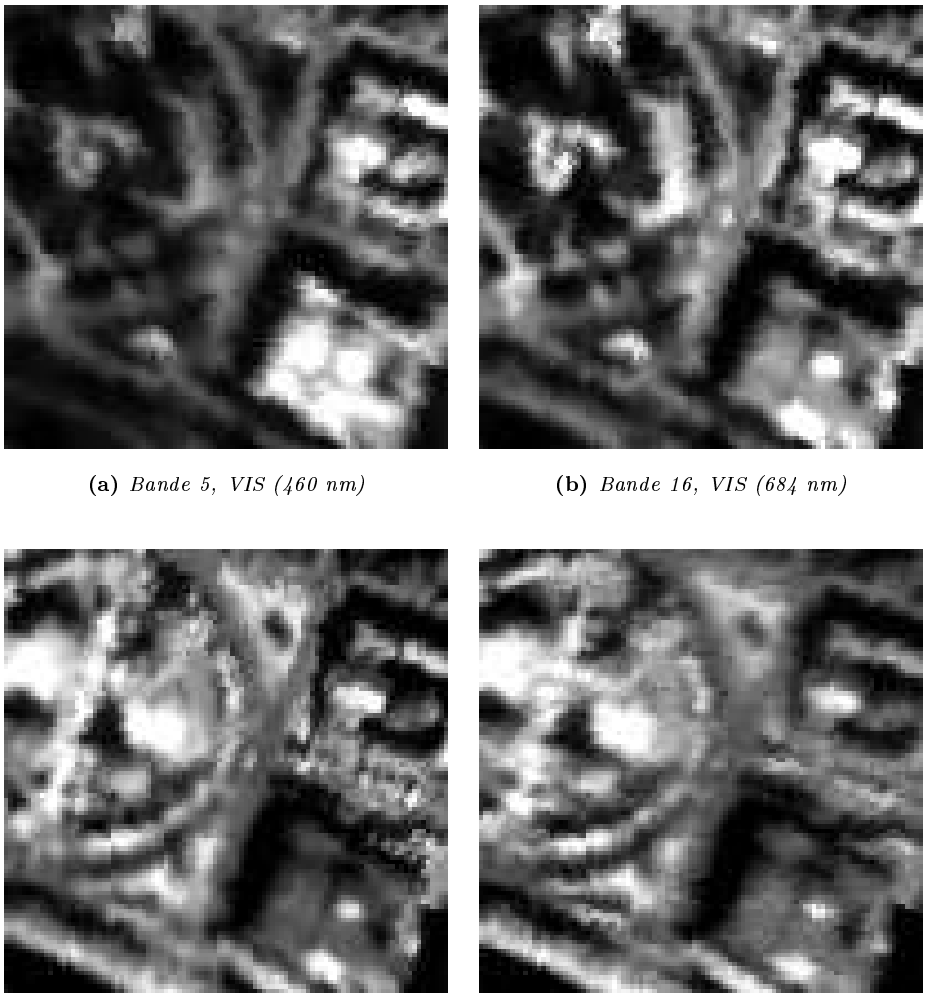
6.5.1 Description des données

La dernière série de tests concerne une image CASI–2 (*Compact Airborne Spectrographic Imager*) de la ville de Strasbourg acquise le 21 septembre 2005. L'image est composée de 32 bandes allant du spectre visible au proche infrarouge (420–960 nm). La résolution spatiale est de 2 m. La

résolution spectrale est d'environ 11 nm. Il s'agit de données calibrées en réflectance apparente au sol et corrigées géométriquement. Cette image a été acquise dans le cadre de FoDoMuSt.

Les tests ont été réalisés sur un extrait de l'image de 100×100 pixels (les images en niveaux de gris de quatre des bandes sont représentées sur la figure 6.11). Cet extrait correspond à une place recouverte de végétation (pelouse et arbres) et entourée d'une route et de bâtiments. Au bas de l'image se trouve un cours d'eau bordé de végétation.

Les bandes 5 et 16 (FIG. 6.11(a) et 6.11(b)) correspondent à deux régions du spectre visible (bleu et rouge). Les deux bandes 19 et 32 (FIG. 6.11(c) et 6.11(d)) correspondent au proche infrarouge.



(a) *Bande 5, VIS (460 nm)*

(b) *Bande 16, VIS (684 nm)*

(c) *Bande 19, NEAR–IR (736 nm)*

(d) *Bande 32, NEAR–IR (955 nm)*

FIG. 6.11 : *Extrait de l'image CASI sur quatre bandes*

Sur la figure 6.12 est représentée la photographie aérienne correspondant à l'extrait utilisé dans ces tests, provenant toujours de la base de données d'orthophotographie de l'IGN (Institut Géographique National) du 5 mai 1998. Il s'agit d'une photographie, correspondant donc à la lumière visible (450–900 nm), la résolution spatiale est de 0,5 m. Cette photographie n'est pas utilisée par les algorithmes de classification, mais uniquement pour faciliter l'évaluation visuelle des résultats. Rappelons également que les ombres présentes sur une image dépendent de la date et de l'heure d'acquisition.

Fig. 6.12 : *Photographie aérienne correspondant à l'extrait de l'image CASI*

6.5.2 Résultats avec MACLAW

L'algorithme MACLAW a été appliqué sur l'extrait de l'image CASI avec 32 bandes. L'algorithme a été configuré pour utiliser des extracteurs basés sur K-means avec trois individus représentatifs initialisés par K-means. Les classes extraites ont été sélectionnées en fonction des individus représentatifs. Plusieurs tests ont été réalisés en cherchant quatre à six classes. Il y avait 20 individus dans chaque population et 100 générations.

Sur la figure 6.13 sont représentés les résultats de classification obtenus par MACLAW.

Sur le résultat avec quatre classes (Fig. 6.13(a)), chacune d'entre elles a pu être identifiée : ombre/eau, bâti, route et végétation. On remarque cependant des confusions entre les classes de bâti et de route (en particulier à proximité des zones d'ombre). Sur le résultat avec cinq classes (Fig. 6.13(b)), on retrouve les quatre classes précédentes, et une classe correspondant à des surfaces très réfléchissantes apparaît. Sur le résultat avec six classes (Fig. 6.13(c)), on retrouve les cinq classes précédentes. La classe supplémentaire correspond à la même classe de surfaces réfléchissantes déjà mise en évidence. Des tests avec un nombre supérieur de classes font encore apparaître cette même classe.

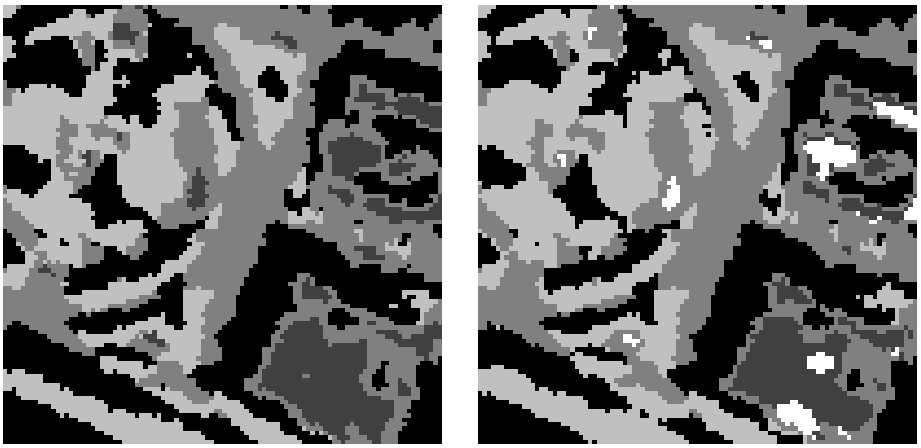
Ces résultats montrent les limites du critère d'évaluation qui a été proposé. En effet, plusieurs extracteurs ont mis en évidence plusieurs fois une même classe, très compacte et de cardinalité très faible. Cette classe apporte une grande qualité interne car elle est très homogène. En revanche, la multiplicité de cette classe n'est pas pénalisée par le degré de partitionnement car le cardinal de la classe est trop petit.

Constatant cette limite de notre algorithme, nous n'avons pas poussé l'analyse des résultats plus loin pour cette image. En revanche, ces résultats indiquent qu'il est nécessaire d'améliorer le critère d'évaluation utilisé, en prenant d'avantage en compte le degré de partitionnement, par exemple.

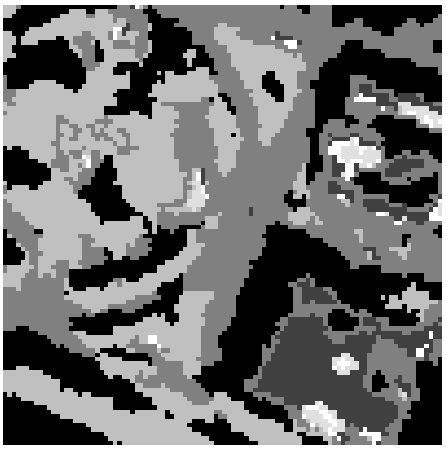
6.6 Conclusion

Nous avons vu dans ce chapitre que la méthode de classification non supervisée avec pondération locale d'attributs MACLAW pouvait être appliquée efficacement pour la construction de cartes thématiques à partir d'images de télédétection.

Nous avons en effet constaté que sur des images hyperspectrales, comportant un nombre important de bandes présentant de nombreuses corrélations, notre algorithme était capable de découvrir des classes identifiables par un expert. De plus, l'algorithme est capable de mettre en évidence les



(a) *Résultat avec quatre classes* **(b)** *Résultat avec cinq classes*



(c) *Résultat avec six classes*

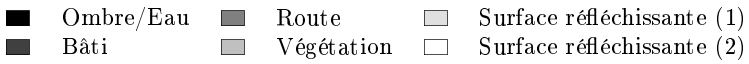


Fig. 6.13 : *Résultat de l'algorithme MACLAW sur l'image CASI*

bandes les plus discriminantes pour chacune des classes. Les bandes mises en évidence par notre méthode ont pu être utilisées dans un autre algorithme (une méthode de classification supervisée du logiciel ENVI) : en utilisant moins du quart des attributs, l'algorithme supervisé était capable de produire un résultat légèrement meilleur que celui obtenu en utilisant toute l'information.

L'algorithme MACLAW s'est également montré efficace pour la classification de régions obtenues par un processus de segmentation. Les caractéristiques des régions apportent des informations très différentes (informations sur la radiométrie, la forme ou la texture). De plus certains attributs sont corrélés entre eux (par exemple, les différents attributs de forme) et bruités en raison des imperfections des régions construites par segmentation. Un algorithme de classification classique est incapable de découvrir des classes identifiables, alors que les résultats de MACLAW sont satisfaisants.

Ces résultats ont cependant souligné une faiblesse du critère d'évaluation que nous avons proposé. En effet, celui-ci est mis en défaut par des classes très homogènes et de cardinalité faible. Une remise en cause du critère d'évaluation d'une CDP est à faire, probablement en donnant plus d'importance au degré de partitionnement qu'à la qualité interne des classes.

L'application de MACLAW pour la construction de cartes thématiques à partir d'images de télédétection nécessite cependant une étude plus approfondie. En particulier, des tests sur des images plus grandes et avec un plus grand nombre de classes restent à faire.

Conclusion

Dans la perspective d'obtenir une classification plus précise, on cherche souvent à décrire les données de la manière la plus détaillée possible, celles-ci étant alors représentées par de nombreux attributs. Or, plusieurs problèmes peuvent apparaître lors de la classification, lorsque les objets sont représentés par de trop nombreux attributs : manque de pertinence de certains attributs, bruit, corrélations, différence d'échelle et coût d'acquisition ou de temps de calcul. Face à des données de plus en plus complexes, les méthodes classiques ne sont plus efficaces. Une approche communément utilisée consiste alors à adapter les données aux algorithmes de classification.

Dans cette thèse, nous avons étudié la pondération d'attributs pour la classification non supervisée. La pondération d'attributs consiste à faire varier l'influence relative des attributs lors de la classification. Il n'existe à ce jour que peu de travaux concernant la pondération d'attributs dans le cadre de la classification non supervisée. Nous avons proposé deux familles de méthodes utilisant les algorithmes génétiques comme technique d'optimisation.

Contributions

Nous nous sommes particulièrement intéressés à un groupe de méthodes de pondération globale ou locale basées sur l'algorithme K-means. Ces méthodes utilisent trois optimisations partielles réitérées successivement afin de minimiser une fonction de coût. Il est connu que ce type de méthodes d'optimisation est sensible aux paramètres initiaux et risque de ne découvrir qu'un minimum local et non pas la solution optimale. Nous avons donc amélioré ces méthodes en remplaçant la technique d'optimisation classique par un algorithme génétique afin d'optimiser la même fonction d'évaluation. Nous avons utilisé divers algorithmes génétiques évolutionnaires ou coévolutionnaires, ainsi que des méthodes hybrides combinant approche classique et approche génétique (approches lamarckienne et baldwinienne). Ces nouveaux algorithmes se sont révélés très efficaces pour minimiser la fonction d'évaluation. Plus particulièrement, nous avons pu vérifier l'importance de la pondération locale des attributs, c'est-à-dire la nécessité d'utiliser des pondérations différentes selon les classes. Néanmoins, nous avons pu mettre en évidence plusieurs limites à ces approches. En premier lieu, ces algorithmes semblent sensibles aux dépendances entre les attributs et produisent de mauvais résultats lorsqu'il y a trop de corrélations entre les attributs. En second lieu, la fonction de coût utilisée implique que ces algorithmes ne peuvent découvrir que des classes sphériques (dans une certaine métrique) construites en fonction d'une distance et d'un prototype.

C'est pourquoi nous avons cherché à définir une nouvelle méthode générique de pondération d'attributs par approche enveloppe pouvant être utilisée avec toute méthode de classification classique.

Pour cela, nous avons d'abord proposé une approche modulaire pour la classification non supervisée. Cette approche consiste à décomposer un problème de classification en K classes en K problèmes d'extraction d'une classe. Le problème de classification revient alors à identifier K extracteurs permettant d'obtenir des classes pertinentes et complémentaires, chaque extracteur

produisant une unique classe. Une phase d'apprentissage consiste à construire un ensemble d'extracteurs produisant un bonne classification des données par une optimisation réalisée par un algorithme de coévolution coopérative.

Afin d'évaluer la qualité du résultat obtenu par un ensemble d'extracteurs, nous avons été amenés à définir un critère d'évaluation de la qualité d'une classification douce partielle. Ce critère est basé, d'une part, sur la complémentarité des classes (il doit y avoir le moins d'intersections possible entre les classes et le moins d'objets atypiques possible) et, d'autre part, sur la qualité interne des classes (les classes doivent être homogènes). L'expérience a montré que ces deux aspects du critère étaient antagonistes. De plus, ces deux critères ont montré une faiblesse face à des classes très homogènes et de faible cardinalité. Il semble indispensable de maintenir le degré de partitionnement le plus élevé possible avant de chercher à maximiser la qualité interne des classes. Une meilleure gestion de l'optimisation multiobjectif ou une redéfinition de la fonction d'évaluation sont à étudier.

La méthode MACLAW est une application de l'approche modulaire pour la classification avec pondération des attributs : l'extraction d'une classe se fait en réalisant une classification des données par une méthode classique, en utilisant des pondérations globales et en choisissant l'une des classes obtenues comme classe extraite, selon un critère défini.

L'algorithme MACLAW peut être appliqué avec des méthodes de classification qui n'utilisent pas la notion de distance entre objets pour construire les classes. Il autorise par ailleurs l'utilisation de pondérations locales pour tout algorithme de classification non supervisée. La méthode peut également être considérée comme une approche multi-stratégique, car les méthodes d'extraction peuvent être différentes pour chacune des classes.

Des tests réalisés sur différents ensembles de données ont montré sa capacité à trouver des classes valides et à mettre en évidence les attributs pertinents. Nous avons pu voir que la méthode MACLAW est particulièrement efficace face à des attributs nombreux et corrélés entre eux. Les temps de calcul des différentes méthodes sont très longs ce qui rend problématique leur utilisation. Cependant, la structure modulaire de notre approche permet une parallélisation aisée. Une étude concernant la parallélisation de MACLAW est en cours de validation[4].

L'algorithme MACLAW a également été appliqué efficacement à la construction de cartes thématiques à partir d'images de télédétection. Notre méthode s'est montrée efficace à la fois dans le cadre de la classification au niveau des pixels d'images hyperspectrales, mais aussi dans le cadre de la classification au niveau des objets construits à partir d'un algorithme de segmentation. Cependant des tests sur des images plus volumineuses et avec un nombre de classes plus conséquent sont à faire.

Ces travaux de recherche ont donné lieu à des publications aussi bien au niveau international [Blansché *et al.*, 2005a ; Blansché *et al.*, 2005b ; Blansché *et al.*, 2005c ; Blansché *et al.*, 2006 ; Wania *et al.*, 2006] qu'au niveau national [Blansché *et al.*, 2004 ; Blansché et Gançarski, 2004a ; Blansché et Gançarski, 2004b ; Blansché et Gançarski, 2005 ; Gançarski et Blansché, 2006].

Perspectives

Les travaux présentés dans cette thèse ouvrent de nombreuses pistes de recherche.

En premier lieu, il sera intéressant d'étudier plus en détail l'apport de l'algorithme MACLAW dans le cadre de l'observation de la Terre. Des tests sur des images plus grandes et avec un nombre de classes plus élevé doivent être réalisés. Il serait également important d'étudier l'intégration de l'algorithme MACLAW dans un processus plus général d'extraction des connaissances à partir d'images de télédétection.

Parallèlement, une étude plus approfondie et une extension de l'approche modulaire sont à faire. Une piste de recherche concerne la recherche automatique du nombre de classes qui est un paramètre déterminant de l'approche modulaire. Dans [Potter *et al.*, 1995], les auteurs ont observé

[4]Travail en cours, en collaboration avec Damien Vouriot, Stéphane Genaud et Pierre Gançarski.

que lorsque le nombre de populations n'est pas assez élevé, des sous-populations se créent au sein des populations, c'est-à-dire des individus ayant des phénotypes très différents se développent dans une même population, une espèce pouvant alors occuper plusieurs niches. Ainsi, dans notre cas, cela revient à diviser une population en deux, afin d'ajouter une classe. Réciproquement, si le nombre de classes est trop élevé, il est envisageable de supprimer les populations occupant la même niche. Des travaux préliminaires sur l'adaptation dynamique du nombre de classes ont été menés mais nécessitent encore une étude théorique plus poussée.

D'autres pistes de recherche concernent la nature des extracteurs utilisés dans l'approche modulaire. Tout en utilisant des extracteurs définis par des méthodes de classification, comme dans MACLAW, il est envisageable de faire varier la méthode de classification utilisée en même temps que les pondérations des attributs, afin de découvrir le meilleur algorithme pour extraire chacune des classes. Il est également concevable de modifier les chromosomes afin de définir une méthode de construction d'attributs. Les chromosomes seraient alors des arbres décrivant les attributs construits par des relations entres les attributs existants (programmation génétique).

Il serait également intéressant de développer de nouvelles stratégies d'extraction. Par exemple, dans une approche floue, il est possible de définir des extracteurs calculant directement le degré d'appartenance des objets aux classes par une fonction arithmétique sur les attributs définie par le chromosome. Des approches moins génériques peuvent également être employées. En particulier, l'utilisation d'extracteurs *ad hoc*, en exploitant l'aspect multi-stratégique de l'approche modulaire, semble particulièrement prometteuse. Des méthodes d'extraction spécifiques à chaque classe que l'on souhaite mettre en évidence peuvent être utilisées. L'algorithme permettrait par exemple de découvrir les paramètres optimaux pour ces différentes méthodes. De plus, comme il n'existe pas nécessairement de méthode spécifique à chaque classe, une combinaison entre extracteurs *ad hoc* et extracteurs génériques (comme ceux basés sur des méthodes de classification que nous avons présentées dans cette thèse) est tout à fait possible.

L'application de l'approche modulaire à des données différentes, en utilisant la spécificité de ces données, est également concevable. Une étude concernant des extracteurs basés sur des opérateurs de morphologie mathématique pour la classification des pixels d'une image est d'ailleurs en cours au sein de LSIIT.

Enfin, il serait intéressant d'étudier l'intégration de connaissances dans le processus d'apprentissage, par des exemples d'apprentissage, une ontologie représentant ces connaissances ou encore une intervention de l'utilisateur au cours du processus.

Bibliographie

[Aggarwal *et al.*, 1999] C.C. Aggarwal, J.L. Wolf, P.S. Yu, C. Procopiuc, et J.S. Park. Fast algorithms for projected clustering. Dans *Proceedings of SIGMOD*, pages 61–72, 1999.

[Agrawal *et al.*, 1994] R. Agrawal, T. Imielinski, et A. Swami. Mining association rules between sets of items in large databases. Dans *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, 1994.

[Agrawal *et al.*, 1998] R. Agrawal, J. Gehrke, D. Gunopulos, et P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. Dans *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 94–105, 1998.

[Aha, 1998] D.W. Aha. Feature weighting for lazy learning algorithms. Dans *Feature Extraction, Construction and Selection : A Data Mining Perspective*, éditeurs H. Liu et H. Motoda, chapitre 1. Kluwer, 1998.

[Akkus et Güvenir, 1996] A. Akkus et H.A. Güvenir. Weighting features in k nearest neighbor classification on feature projections, 1996.

[Almuallim et Dietterich, 1991] H. Almuallim et T.G. Dietterich. Learning with many irrelevant features. Dans *Proceedings of the Ninth National Conference on Artificial Intelligence*, volume 2, pages 547–552, Anaheim, California, 1991. AAAI Press.

[Bäck, 1996] T. Bäck. *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, 1996.

[Bajcsy et Groves, 2004] P. Bajcsy et P. Groves. Methodology for hyperspectral band selection. *Photogrammetric Engineering and Remote Sensing journal*, 70(7) : 793–802, 2004.

[Baldwin, 1896] J.M. Baldwin. A new factor in evolution. *The American Naturalist*, 30(354) : 441–451, 1896.

[Bel Mufti et Bertrand, 1997] G. Bel Mufti et P. Bertrand. Validation d'une classe par rééchantillonnage. Dans *Cinquième rencontres de la Société Francophone de Classification, Lyon*, pages 251–254, 1997.

[Bell et Wang, 2000] D.A. Bell et H. Wang. A formalism for relevance and its application in feature subset selection. *Machine Learning*, 41 : 175–195, 2000.

[Bellman, 1961] R.E. Bellman. *Adaptive Control Processes*. Princeton University Press, Princeton, NJ, 1961.

[Benediktsson *et al.*, 1995] J.A. Benediktsson, J.R. Sveinsson, et K. Arnason. Classification and feature extraction of AVIRIS data. *IEEE Transactions on Geoscience and Remote Sensing*, 33 : 1194–1205, 1995.

[Bezdek et Pal, 1998] J.C. Bezdek et N.R. Pal. Some new indexes of cluster validity. *IEEE Transactions on Systems, Man and Cybernetics*, 28(3) : 301–315, 1998.

[Bezdek, 1974] J.C. Bezdek. Numerical taxonomy with fuzzy sets. *Journal of Mathematical Biology*, 1 : 57–71, 1974.

[Bezdek, 1981] J.C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algoritms*. Plenum Press, New York, 1981.

[Blake et Merz, 1998] C.L. Blake et C.J. Merz. UCI repository of machine learning databases, 1998. http ://www.ics.uci.edu/∼mlearn/MLRepository.html.

[Blansché *et al.*, 2004] A. Blansché, P. Gançarski, et J.J. Korczak. Algorithme génétique de pondération d'attributs pour une classification non supervisée. Dans *Proceedings of SFC'04*, Bordeaux, France, 2004.

[Blansché et Gançarski, 2004a] A. Blansché et P. Gançarski. Application aux images hyperspectrales d'une nouvelle méthode de sélection d'attributs pour la classification d'objets complexes. Dans *Proceedings of the 1st Workshop on Fouille de Données Complexes dans un processus d'extraction de connaissances in EGC'04*, pages 103–114, Clermont-Ferrand, France, 2004.

[Blansché et Gançarski, 2004b] A. Blansché et P. Gançarski. Sélection d'attributs et classification d'objets complexes. Dans *Proceedings of Extraction et Gestion des Connaissances (Poster session)*, page 203, Clermont-Ferrand, France, 2004.

[Blansché *et al.*, 2005a] A. Blansché, P. Gançarski, et J.J. Korczak. A coevolutionary approach for clustering with feature weighting : Application to image analysis. Dans *Proceedings of the 7th European Workshop on Evolutionary Computation in Image Analysis and Signal Processing in EvoWorkshops2005*, volume 3449 de *LNCS*, pages 254–264, Lausanne, Switzerland, 2005.

[Blansché *et al.*, 2005b] A. Blansché, P. Gançarski, et J.J. Korczak. Genetic algorithms for feature weighting : Evolution vs. coevolution and darwin vs. lamarck. Dans *Proceedings of the 4th Mexican International Conference on Artificial Intelligence, Monterrey, Mexique*, volume 3789 de *LNCS*, pages 682–691, 2005.

[Blansché *et al.*, 2005c] A. Blansché, P. Gançarski, et J.J. Korczak. Representative individuals initialization in cooperative coevolution. Dans *Proceedings of the 2nd Indian International Conference on Artificial Intelligence, Pune, Inde*, pages 2748–2758, 2005.

[Blansché *et al.*, 2006] A. Blansché, P. Gançarski, et J.J. Korczak. MACLAW : A modular approach for clustering with local attribute weighting. *Pattern Recognition Letters*, 27(11) : 1299–1306, 2006.

[Blansché et Gançarski, 2005] A. Blansché et P. Gançarski. Algorithme génétique de pondération d'attributs pour une classification non supervisée d'objets complexes. *RNTI*, 2005.

[Blickle et Thiele, 1995] T. Blickle et L. Thiele. A comparison of selection schemes used in genetic algorithms. Rapport Technique 11 Version 2, Computer Engineering and Communicaion Netwok Lab, Swiss Federal Institute of Technology, Zurich, 1995.

[Blum et Langley, 1997] Avrim Blum et Pat Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1–2) : 245–271, 1997.

[Boardman, 1998] J. Boardman. Leveraging the high dimensionality of aviris data for improved subpixel target unmixing and rejection of false positives : mixure tuned matched filtering. Dans *Summaries of the Seventh Annual JPL Airborne Geoscience Workshop*, 1998.

[Bolshakova et Azuaje, 2003] N. Bolshakova et F. Azuaje. Cluster validation techniques for genome expression data. *Signal Processing*, 83(4) : 825–833, 2003.

[Bonn et Rochon, 1992] F. Bonn et G. Rochon. *Précis de télédétection*, volume 1. Presses de l'Université du Québec, 1992.

[Candillier *et al.*, 2005a] L. Candillier, I. Tellier, F. Torre, et O. Bousquet. SSC : Statistical Subspace Clustering. Dans *5èmes journées francophones d'Extraction et Gestion des Connaissances (EGC'2005)*, éditeurs Suzanne Pinson et Nicole Vincent, volume 1, pages 177–182, 2005.

[Candillier *et al.*, 2005b] Laurent Candillier, Isabelle Tellier, Fabien Torre, et Olivier Bousquet. SSC : Statistical Subspace Clustering. Dans *4th International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM'2005)*, volume LNAI 3587 de *LNCS*, pages 100–109, Leipzig, Germany, july 2005. Springer Verlag.

[Candillier *et al.*, 2005c] Laurent Candillier, Isabelle Tellier, Fabien Torre, et Olivier Bousquet. SSC : Statistical Subspace Clustering. Rapport Technique GRAPPA–0105, GRAppA - Université Charles de Gaulle - Lille 3, 2005.

[Cantù-Paz, 2002] E. Cantù-Paz. Feature subset selection by estimation of distribution algorithms. Dans *Genetic and Evolutionary Computation Conference (GECCO'02)*, pages 303–310, 2002.

[Cardie, 1993] C. Cardie. Using decision trees to improve case-based learning. Dans *Proceedings of the Tenth International Conference on Machine Learning*, pages 25–32, 1993.

[Cardie, 1996] C. Cardie. Automating feature set selection for case-based learning of linguistic knowledge. Dans *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 1996.

[Carroll, 1871] Lewis Carroll. *Through the Looking-Glass, and What Alice Found There*, 1871.

[Caruana et Freitag, 1994] Rich Caruana et Dayne Freitag. Greedy attribute selection. Dans *International Conference on Machine Learning*, pages 28–36, 1994.

[Cauvin et Rimbert, 1976] C. Cauvin et S. Rimbert. *La lecture numérique des cartes thématiques.* Éditions universitaires, 1976.

[CCRS, 2001] Canada Centre for Remote Sensing CCRS, 2001. http ://www.ccrs.nrcan.gc.ca/.

[Céa, 1971] J. Céa. *Optimisation : théorie et algorithmes.* Dunod, 1971.

[Cha et Srihari, 2000a] S.-H. Cha et S.N. Srihari. Distance between histograms of angular measurements and itsapplication to handwritten character similarity. Dans *Proceedings of the 15th International Conference on Pattern Recognition*, pages 21–24, 2000.

[Cha et Srihari, 2000b] S.-H. Cha et S.N. Srihari. On measuring the distance between histograms. *Pattern Recognition*, 35 : 1355–1370, 2000.

[Chan *et al.*, 2004] E.Y. Chan, W.K. Ching, M.K. Ng, et J.Z. Huang. An optimization algorithm for clustering using weighted dissimilarity measures. *Pattern Recognition*, 37 : 943–952, 2004.

[Charon *et al.*, 1996] I. Charon, A. Germa, et O. Hudry. *Méthodes d'optimisation combinatoire.* Masson, 1996.

[Cheng *et al.*, 1999] C.-H. Cheng, A.W.-C. Fu, et Y. Zhang. Entropy-based subspace clustering for mining numerical data. Dans *Knowledge Discovery and Data Mining*, pages 84–93, 1999.

[Cleuziou *et al.*, 2004] G. Cleuziou, L. Martin, et C. Vrain. PoBOC : un algorithme de «soft-clustering». applications à l'apprentissage de règles et au traitement de donnée textuelles. Dans *Proceedings of Extraction et Gestion des Connaissances*, pages 217–228, 2004.

[Cliff et Miller, 1995] Dave Cliff et Geoffrey F. Miller. Tracking the red queen : Measurements of adaptive progress in co-evolutionary simulations. Dans *European Conference on Artificial Life*, pages 200–218, 1995.

[Cocquerez et Philipp, 1995] J.-P. Cocquerez et S. Philipp. *Analyse d'images : filtrage et segmenation.* Masson, 1995.

[Collet *et al.*, 2000] P. Collet, E. Lutton, F. Raynal, et M. Schoenauer. Polar IFS+parisian genetic programming=efficient IFS inverse problem solving. *Genetic Programming and Evolvable Machines*, 1(4) : 339–361, 2000.

[Cord *et al.*, 2006] A. Cord, C. Ambroise, et J.-P. Cocquerez. Feature selection in robust clustering based on Laplace mixture. *Pattern Recognition Letters*, 27(6) : 627–635, 2006.

[Culioli, 1994] J.-C. Culioli. *Introduction à l'optimisation.* Ellipses, 1994.

[Darwin, 1859] C. Darwin. *On the origin of species by means of natural selection.* London, John Murray, 1859.

[Dash et Liu, 2000] M. Dash et H. Liu. Feature selection for clustering. Dans *Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2000.

[Dash et Liu, 2003] M. Dash et H. Liu. Consistency-based search in feature selection. *Artificial Intelligence*, 151 : 155–176, 2003.

[Davies et Bouldin, 1979] D. Davies et D. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2) : 224–227, 1979.

[Dempster *et al.*, 1977] A. Dempster, N. Laird, et D Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1) : 1–38, 1977.

[Derivaux *et al.*, 2006] S. Derivaux, S. Lefèvre, C. Wemmert, et J.J. Korczak. Watershed segmentation of remotely sensed images based on a supervised fuzzy pixel classification. Dans *Proceeding of the 2006 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2006.

[Domingos, 1997] P. Domingos. Contextsensitive feature selection for lazy learners. *Artificial Intelligence Review*, 11 : 227–253, 1997.

[Dunn *et al.*, 2005] E. Dunn, G. Olague, et E. Lutton. Automated photogrammetric network design using the parisian approach. Dans *Proceedings of the 7th European Workshop on Evolutionary Computation in Image Analysis and Signal Processing in EvoWorkshops2005*, pages 356–365, 2005.

[Dunn, 1974a] J.C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact, well separated clusters. *Journal of Cybernetics*, 3 : 32–57, 1974.

[Dunn, 1974b] J.C. Dunn. Well separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 4 : 95–104, 1974.

[Dy et Brodley, 2000] Jennifer G. Dy et Carla E. Brodley. Feature subset selection and order identification for unsupervised learning. Dans *Proceedings 17th International Conf. on Machine Learning*, pages 247–254. Morgan Kaufmann, San Francisco, CA, 2000.

[Ester *et al.*, 1996] M. Ester, H.-P. Kriegel, J. Sander, et X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. Dans *Proceedings of 2nd International Conference of Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[Fisher, 1987] D. H. Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine learning*, 2 : 139–172, 1987.

[Floreano et Nolfi, 1997] D. Floreano et S. Nolfi. God save the red queen! competition in coevolutionary robotics. Dans *Genetic Programming 1997 : Proceedings of the Second Annual Conference*, pages 398–406, 1997.

[Fonseca et Fleming, 1995] C.M. Fonseca et P.J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1) : 1–16, 1995.

[Frigui et Nasraoui, 2004] H. Frigui et O. Nasraoui. Unsupervised learning of prototypes and attribute weights. *Pattern Recognition*, 34 : 567–581, 2004.

[Gançarski et Blansché, 2006] P. Gançarski et A. Blansché. Trois stratégies d'évolution pour la pondération automatique d'attributs en classification non supervisée d'objets complexes. Dans *Proceedings of the 3rd Workshop on Fouille de Données Complexes dans un processus d'extraction de connaissances in EGC'04*, pages 71–82, 2006.

[Goldberg et Voessner, 1999] D.E. Goldberg et S. Voessner. Optimizing global-local search hybrids. Dans *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 220–228, 1999.

[Goldberg, 1989] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., 1989.

[Gonzalez et Woods, 1993] R.C. Gonzalez et R.E. Woods. *Digital image processing*. Addison-Wesley Publishing Company, 1993.

[Grabmeier et Rudolph, 2002] J. Grabmeier et A. Rudolph. Techniques of cluster algorithms in data mining. *Data Mining and Knowledge Discovery*, 6(4) : 303–360, 2002.

[Gréco et Piaget, 1959] P. Gréco et J. Piaget. *Apprentissage et connaissance*. Presses universitaires de France, 1959.

[Grefenstette, 1991] J.J. Grefenstette. Lamarckian learning in multi-agent environments. Dans *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 303–310, 1991.

[Groves et Bajcsy, 2003] P. Groves et P. Bajcsy. Methodology for hyperspectral band and classification model selection. Dans *IEEE Workshop on Advances in Techniques for Analysis of Remotely Sensed Data*, 2003.

[Guha *et al.*, 1998] S. Guha, R. Rastogi, et K. Shim. CURE : an efficient clustering algorithm for large databases. pages 73–84, 1998.

[Günter et Burke, 2001] S. Günter et H. Burke. Validation indices for graph clustering. Dans *Proc. 3rd IAPR- TC15 Workshop on Graph-based Representations in Pattern Recognition*, pages 229–238. J.-M. Jolion, W. Kropatsch, M. Vento, 2001.

[Haith *et al.*, 1999] Gary L. Haith, Silvano P. Colombano, Jason D. Lohn, et Dimitris Stassinopoulos. Coevolution for problem simplification. Dans *Proceedings of the Genetic and Evolutionary Computation Conference*, éditeurs Wolfgang Banzhaf, Jason Daida, Agoston E. Eiben, Max H. Garzon, Vasant Honavar, Mark Jakiela, et Robert E. Smith, volume 1, pages 244–251, Orlando, Florida, USA, 13-17 1999. Morgan Kaufmann.

[Halkidi *et al.*, 2001a] M. Halkidi, Y. Batistakis, et M. Vazirgiannis. Clustering algorithms and validity measures. Dans *Proceedings of the 13th International Conference on Scientific and Statistical Database Management*, pages 3–22, 2001.

[Halkidi *et al.*, 2001b] M. Halkidi, Y. Batistakis, et M. Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2-3) : 107–145, 2001.

[Hammadi-Mesmoudi, 1995] F. Hammadi-Mesmoudi. *Classifieur neuronal d'images de télédétection.* Thèse de doctorat, Université Louis Pasteur, Strasbourg, France, 1995.

[Hartley, 1958] H. Hartley. Maximum likelihood estimation from incomplete data. *Biometrics*, 14 : 174–194, 1958.

[Hauta-Kasari *et al.*, 1996] M. Hauta-Kasari, J. Parkkinen, T. Jaaskelainen, et R. Lenz. Generalized cooccurrence matrix for multispectral texture analysis. pages 785–789, 1996.

[Holland, 1975] J.H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, 1975.

[Howe et Cardie, 1997] N. Howe et C. Cardie. Examining locally varying weights for nearest neighbor algorithms. Dans *ICCBR*, pages 455–466, 1997.

[Howe et Cardie, 1999] N. Howe et C. Cardie. Weighting unusual feature types. Rapport Technique TR99-1735, Ithaca, 1999.

[Hsu *et al.*, 2002] C. Hsu, H. Huang, et D. Schuschel. The ANNIGMA-wrapper approach to fast feature selection for neural nets. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 32(2) : 207–212, 2002.

[Huang *et al.*, 2005] J.Z. Huang, M.K. Ng, H. Rong, et Z. li. Automated variable weighting in k-means type clustering. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 27(2) : 657–668, 2005.

[Hubert *et al.*, 1985] L. Hubert, , et P. Arabie. Comparing partitions. *Journal of Classification*, 2 : 193–218, 1985.

[Inza *et al.*, 2000] I. Inza, P. Larraaga, et B. Sierra. Feature weighting for nearest neighbor by estimation of bayesian networks algorithms. Rapport Technique EHU-KZAA-IK-3/00, University of the Basque Country, 2000.

[Ioro et Li, 2004] A. Ioro et X. Li. A cooperative coevolutionary multiobjective algorithm using non-dominated sorting. Dans *Proceeding of Genetic and Evolutionary Computation Conference 2004 (GECCO'04), Lecture Notes in Computer Science (LNCS 3102)*, pages 537–548, 2004.

[Jain *et al.*, 1999] A.K. Jain, M.N. Murty, et P.J. Flynn. Data clustering : a review. *ACM Computing Surveys*, 31(3) : 264–323, 1999.

[John *et al.*, 1994] G.H. John, R. Kohavi, et K. Pfleger. Irrelevant features and the subset selection problem. Dans *Proceedings of the Eleventh International Conference on Machine Learning*, pages 121–129, 1994.

[Julstrom, 1999] B.A. Julstrom. Comparing darwinian, baldwinian, and lamarckian search in a genetic algorithm for the 4-cycle problem. Dans *Late Breaking Papers at the 1999 Genetic and Evolutionary Computation Conference*, éditeurs S. Brave et A.S. Wu, pages 134–138, 1999.

[Kennedy et Eberhart, 1995] J. Kennedy et R. Eberhart. Particle swarm optimization. Dans *Proceedings of the 1995 IEEE Int. Conf. on Neural Networks*, pages 1942–1948, 1995.

[Ketterlin, 1995] A. Ketterlin. *Découverte de concepts structurés dans les bases de données.* Thèse de doctorat, Université Louis Pasteur, Strasbourg, 1995.

[Kim *et al.*, 2000] Y.S. Kim, W.N. Street, et F. Menczer. Feature Selection in Unsupervised Learning via Evolutionary Search. Dans *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 365–369, 2000.

[Kim *et al.*, 2002] Y. Kim, W.N. Street, et F. Menczer. Evolutionary model selection in unsupervised learning. *Intelligent Data Analysis*, 6 : 531–536, 2002.

[Kim *et al.*, 2003] Y.S. Kim, W.N. Street, et F. Menczer. Feature selection in data mining. *Data mining : opportunities and challenges*, pages 80–105, 2003.

[Kira et Rendell, 1992] K. Kira et L.A. Rendell. A practical approach to feature selection. Dans *Proceedings of International Conference on Machine Learning*, pages 249–256, 1992.

[Kirkpatrick *et al.*, 1983] S. Kirkpatrick, C.D. Gelatt Jr., et M.P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598) : 671–680, 1983.

[Kittler, 1998] J. Kittler. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3) : 226–239, 1998.

[Kohavi *et al.*, 1997] R. Kohavi, P. Langley, et Y. Yun. The utility of feature weighting in nearest-neighbor algorithms. Dans *Proceedings of the Ninth European Conference on Machine Learning, Prague*, 1997.

[Kohavi et John, 1998] R. Kohavi et G.H. John. The wrapper approach. Dans *Feature Extraction, Construction and Selection : A Data Mining Perspective*, éditeurs Huan Liu et Hiroshi Motoda. Springer-Verlag, 1998.

[Kohavi et Sommerfield, 1995] R. Kohavi et D. Sommerfield. Feature subset selection using the wrapper method : Overfitting and dynamic search space topology. Dans *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, pages 192–197, 1995.

[Kohavi, 1994] R. Kohavi. Feature subset selection as search with probabilistic estimates. Dans *AAAI Fall Symposium on Relevance*, pages 122–126, 1994.

[Kohonen, 1982] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43 : 59–69, 1982.

[Koller et Sahami, 1996] Daphne Koller et Mehran Sahami. Toward optimal feature selection. Dans *International Conference on Machine Learning*, pages 284–292, 1996.

[Kononenko, 1994] I. Kononenko. Estimating attributes : Analysis and extensions of relief. Dans *Proceedings of the European Conference on Machine Learning*, pages 171–182, 1994.

[Koza, 1992] J.R. Koza. Genetic programming : On the programming of computers by means of natural selection. *Journal of Artificial Intelligence Research*, 4 : 237–285, 1992.

[Krawiec et Bhanu, 2003] K. Krawiec et B. Bhanu. Coevolutionary computation for synthesis of recognition systems. Dans *Proceedings of Computer Vision and Pattern Recognition Conference, Workshop on Learning in Computer Vision and Pattern Recognition CVPR 2003*, 2003.

[Krishna et Narasimha Murty, 1999] K. Krishna et M. Narasimha Murty. Genetic K-means algorithm. *IEEE Transactions on Systems, Man and Cybernetics*, 29(3) : 433–439, 1999.

[Kruse et Boardman, 2004] F.A. Kruse et J.W. Boardman. Using hyperspectral data for urban baseline studies. Dans *Proceedings of the 13th JPL Airborne Geoscience Workshop*, 2004.

[Ku et Mak, 1997] K. Ku et M. Mak. Exploring the effects of lamarckian and baldwinian learning in evolving recurrent neural networks. Dans *Proceedings of the IEEE International Conference on Evolutionary Computation*, pages 617–621, 1997.

[Kumar *et al.*, 2001] S. Kumar, J. Ghosh, et M.M. Crawford. Best-bases feature extraction algorithms for classification of hyperspectral data. *IEEE Transactions on Geoscience and Remote Sensing*, 39 : 1368–79, 2001.

[Kuo et Landgrebe, 2001] B.-C. Kuo et D. Landgrebe. Improved statistics estimation and feature extraction for hyperspectral data classification. Rapport Technique TR-ECE-01-6, School of Electrical and Computer Engineering, Purdue University, West Lafayette, Indiana, 2001.

[Lai *et al.*, 2006] C. Lai, M.J.T. Reinders, et L. Wessels. Random subspace method for multivariate feature selection. *Pattern Recognition Letters*, 27 : 1067–1076, 2006.

[Lamarck, 1809] J.-B. Lamarck. *Philosophie zoologique*. Dentu Paris, 1809.

[Lennon, 2002] M. Lennon. *Méthodes d'analyse d'images hyperspectrales*. Thèse de doctorat, Université de Rennes I, 2002.

[Levine et Domany, 2001] E. Levine et E. Domany. Resampling method for unsupervised estimation of cluster validity. *Neural Computation*, 13(11) : 2573–2593, 2001.

[Likas *et al.*, 2003] A. Likas, N. Vlassis, et J.J. Verbeek. The global k-means clustering algorithm. *Pattern Recognition*, 36(2) : 451–461, 2003.

[Lillesand et Kiefer, 1987] T.M. Lillesand et R.W. Kiefer. *Remote sensing and image interpretation*, page 17. Wiley, New York, 1987.

[Lipinski, 2004] P. Lipinski. *Discovering and developing intelligent agents for financial forecasting and analyzing financial time*. Thèse de doctorat, Université Louis Pasteur, Strasbourg, 2004.

[Liu *et al.*, 1998] H. Liu, H. Motoda, et M. Dash. A monotonic measure for optimal feature selection. Dans *Proceedings of European Conference of Machine Learning (ECML)*, 1998.

[Liu et Setiono, 1996] H. Liu et R. Setiono. Feature selection and classification - a probabilistic wrapper approach. Dans *Proceedings of the 9th International Conference on Industrial and Engineering Applications of AI and ES*, 1996.

[MacQueen, 1965] J. MacQueen. Some methods for classification and analysis of multivariate observations. Dans *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, Berkeley, CA, 1965. University of California Press.

[Marceau *et al.*, 1990] D.J. Marceau, P.J. Howarth, J.-M.M. Dubois, et D.J. Gratton. Evaluation of the grey-level co-occurence matrix method for land-cover classification using SPOT imagery. *IEEE Transactions on geoscience and remote sensing*, 28(4) : 513–519, 1990.

[Mayer, 1998] H.A. Mayer. Symbiotic coevolution of artificial neural networks and training data sets. Dans *Proceedings of the Fifth International Conference on Parallel Problem Solving from Nature*, pages 511–520, 1998.

[Morita *et al.*, 2003] M. Morita, R. Sabourin, F. Bortolozzi, et C.Y. Suen. Unsupervised feature selection using multi-objective genetic algorithms for handwritten word recognition. Dans *ICDAR03*, pages 666–670, 2003.

[Motoda et Liu, 2003] H. Motoda et H. Liu. Feature selection, extraction and construction. Dans *The Handbook of Data Mining*, pages 409–423. Lawrence Erlbaum Associates, Inc. Publishers, 2003.

[Murthy et Chowdhury, 1996] C.A. Murthy et N. Chowdhury. In search of optimal clusters using genetic algorithms. *Pattern recognition letters*, 17 : 825–832, 1996.

[Nagesh *et al.*, 1999] H. Nagesh, S. Goil, et A. Choudhary. Mafia : Efficient and scalable subspace clustering for very large data sets. Rapport Technique CPDC–TR–9906–010, Northwestern University, 1999.

[Ng et Han, 1994] R.T. Ng et J. Han. Efficient and effective clustering methods for spatial data mining. Dans *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 144–155, 1994.

[Ng et Han, 2002] R.T. Ng et J. Han. CLARANS : A method for clustering objects for spatial data mining. *IEEE Transactions on Knowledge and Data Engineering*, 14(5) : 1003–1016, 2002.

[Nicoloyannis *et al.*, 1997] N. Nicoloyannis, M. Terrenoireand, et D. Tounissoux. Pertinence d'une classification. Dans *Cinquième rencontres de la Société Francophone de Classification, Lyon*, pages 257–259, 1997.

[Novak, 2000] J.P. Novak. *Méthodes neuronales pour la segmentation d'images de télédétection et l'apprentissage de concepts*. Thèse de doctorat, Université Louis Pasteur, Strasbourg, 2000.

[Paredis, 1996] Jan Paredis. Coevolutionary life-time learning. Dans *Parallel Problem Solving from Nature (PPSN IV)*, pages 72–80. Springer, 1996.

[Paredis, 1997] J. Paredis. Coevolving cellular automata : Be aware of the red queen! Dans *7th Int. Conference on Genetic Algorithms (ICGA 97)*, pages 393–400, 1997.

[Parsons *et al.*, 2004a] L. Parsons, E. Haque, et H. Liu. Evaluating subspace clustering algorithms. *Workshop on Clustering High Dimensional Data and its Applications, SIAM International Conference on Data Mining (SDM 2004)*, pages 48–56, 2004.

[Parsons *et al.*, 2004b] L. Parsons, E. Haque, et H. Liu. Subspace clustering for high dimensional data : a review. *SIGKDD Explorations Newsletter*, 6(1) : 90–105, 2004.

[Potter *et al.*, 1995] M.A. Potter, K.A. De Jong, et J.J. Grefenstette. A coevolutionary approach to learning sequential decision rules. Dans *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 366–372, 1995.

[Potter *et al.*, 2001] M.A. Potter, L.A. Meeden, et A.C. Schultz. Heterogeneity in the coevolved behaviors of mobile robots : The emergence of specialists. Dans *In Proceedings of The Seventeenth International Conference on Artificial Intelligence*, pages 1337–1343, 2001.

[Potter et De Jong, 1994] M.A. Potter et K.A. De Jong. A cooperative coevolutionary approach to function optimization. Dans *Proceedings of the Third Conference on Parallel Problem Solving from Nature*, pages 249–257, Berlin, 1994. Springer.

[Potter et De Jong, 1995] M.A. Potter et K.A. De Jong. Evolving neural networks with collaborative species. Dans *Proceedings of the 1995 Summer Computer Simulation Conf.*, pages 340–345. The Society of Computer Simulation, 1995.

[Potter et De Jong, 2000] M.A. Potter et K.A. De Jong. Cooperative coevolution : an architecture for evolving coadaptative subcomponents. *Evolutionary Computation*, 8 : 1–29, 2000.

[Puissant, 2003] A. Puissant. *Information géographique et images à très haute résolution.* Thèse de doctorat, Université Louis Pasteur, Strasbourg, 2003.

[Punch *et al.*, 1993] W. F. Punch, E. D. Goodman, M. Pei, L. Chia-Shun, P. Hovland, et R. Enbody. Further research on feature selection and classification using genetic algorithms. Dans *Proc. of the Fifth Int. Conf. on Genetic Algorithms*, éditeur Stephanie Forrest, pages 557–564, San Mateo, CA, 1993. Morgan Kaufmann.

[Quirin, 2005] A. Quirin. *Découverte de règles de classification par approche évolutive : application aux images de télédétection.* Thèse de doctorat, Université Louis Pasteur, Strasbourg, 2005.

[Ralambondrainy, 1995] H. Ralambondrainy. A conceptual version of the k-means algorithm. *Pattern Recognition Letters*, 16, 1995.

[Raman et Ioerger, 2003] B. Raman et T.R. Ioerger. Enhancing learning using feature and example selection. Master's thesis, Department of Computer Science, Texas A&M University, 2003.

[Ranchin *et al.*, 2003] T. Ranchin, B. Aiazzi, L. Alparone, S. Baronti, et L. Wald. Image fusion—the ARSIS concept and some successful implementation schemes. *ISPRS Journal of Photogrammetry and Remote Sensing*, 58(1–2) : 4–18, 2003.

[Renders, 1995] J.-M. Renders. *Algorithmes génétiques et réseaux de neurones.* Hermes, 1995.

[RIS, 2004] Research Systems Inc. RIS. Spectral analysis with ENVI, RSI training series, 2004.

[Rosenblatt, 1958] F. Rosenblatt. The perceptron : A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6) : 386–407, 1958.

[Rosin et Belew, 1997] C.D. Rosin et R.K. Belew. New methods for competitive coevolution. *Evolutionary Computation*, 5(1) : 1–29, 1997.

[Ross, 1999] B.J. Ross. A lamarckian evolution strategy for genetic algorithms. Dans *Practical Handbook of Genetic Algorithms : Complex Coding Systems*, éditeur L.D. Chambers, volume 3, pages 1–16. CRC Press, Boca Raton, Florida, 1999.

[Rouse *et al.*, 1973] J.W. Rouse, R.H. Haas, J.A. Schell, et D.W. Deering. Monitoring vegetation systems in the great plains with ERTS. Dans *the Third ERTS Symposium, NASA SP-351 I*, pages 309–317, 1973.

[Scherf et Brauer, 1997] M. Scherf et W. Brauer. Feature selection by means of a feature weighting approach. Rapport Technique FKI-221-97, Forschungsberichte kunstliche Intelligenz, Institut fur Informatik, Technische Universitat Munchen, 1997.

[Schuschel et Hsu, 1998] D. Schuschel et C. Hsu. A weight analysis-based wrapper approach to neural nets feature selection. Dans *Proceedings of the 10th IEEE International Conference on Tools with AI (ICTAI-98)*, pages 89–96, 1998.

[Serpico et Bruzzone, 2001] S.B. Serpico et L. Bruzzone. A new search algorithm for feature selection in hyperspectralremote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 39(7) : 1360–1367, 2001.

[Shiv Naga Prasad *et al.*, 2002] V. Shiv Naga Prasad, A.G. Faheema, et S. Rakshit. Feature selection in example based image retrieval systems. Dans *ICVGIP 2002*, 2002.

[Søndberg-Madsen *et al.*, 2003] N. Søndberg-Madsen, C. Thomsen, et J.M. Pe na. Unsupervised feature subset selection. Dans *Proceedings of the Workshop on Probabilistic Graphical Models for Classification (within ECML/PKDD 2003)*, pages 71–82, 2003.

[Storn et Price, 1995] R. Storn et K. Price. Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. Rapport Technique TR-95-012, ICSI, 1995.

[Tan *et al.*, 2005] K.C. Tan, E.F. Khor, et T.H. Lee. *Multiobjective evolutionary algorithms and applications*. Springer-Verlag, 2005.

[Tanese, 1989] R. Tanese. Distributed genetic algorithms. Dans *Proceedings of the Third International Conference on Genetic Algorithms*, pages 434–439, 1989.

[Tibshirani *et al.*, 2000] R. Tibshirani, G. Walther, et T. Hastie. Estimating the number of clusters in a dataset via the Gap statistic. Rapport Technique 208, Department of Statistics, Stanford University, 2000.

[Turney, 1996] P. Turney. Myths and legends of the baldwin effect. Dans *Proceedings Workshop on Evolutionary Computation and Machine Learning at the 13th International Conference on Machine Learning*, pages 135–142, 1996.

[Vafaie et Jong, 1993] H. Vafaie et K. De Jong. Robust feature selection algorithms. Dans *Proceedings of the Fifth Conference onTools for Artificial Intelligence*, pages 356–363, 1993.

[van Rijsbergen, 1979] C.J. van Rijsbergen. *Information Retrieval*. London, Butterworths, 1979.

[Wang *et al.*, 2004] X.Z. Wang, Y. Wang, et L. Wang. Improving fuzzy c-means clustering based on feature weight learning. *Pattern Recognition Letters*, 25 : 1123–1132, 2004.

[Wang et Zhang, 2001] J. Wang et K. Zhang. Algorithms for new distance measures between histograms. Dans *IGARSS International Geoscience and Remote Sensing Symposium*, pages 1907–1909, 2001.

[Wang, 2005] C. Wang. *Apport de la télédétection hyperspectrale et de l'altimétrie Lidar avionnées à la classification des espèces végétales de la lagune de Venise*. Thèse de doctorat, Université Louis Pasteur, Strasbourg, 2005.

[Wania *et al.*, 2006] A. Wania, A. Blansché, C. Weber, et P. Gançarski. Hyperspectral data : band selection algorithms comparison. Dans *First Workshop of the EARSeL Special Interest Group on Urban Remote Sensing (Poster session)*, page 105, 2006.

[Warner et Shank, 1997] T.A. Warner et M.C. Shank. Spatial autocorrelation analysis of hyperspectral imagery for feature selection. *Remote Sensing of Environment*, 60 : 58–70, 1997.

[Wemmert *et al.*, 2000] C. Wemmert, P. Gançarski, et J.J. Korczak. A collaborative approach to combine multiple learning methods. *International Journal on Artificial Intelligence Tools*, 9(1) : 59–78, 2000.

[Wemmert, 2000] C. Wemmert. *Classification hybride distribuée par collaboration de méthodes non supervisées*. Thèse de doctorat, Université Louis Pasteur, Strasbourg, 2000.

[Wettschereck *et al.*, 1997] D. Wettschereck, D.W. Aha, et T. Mohri. A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review*, 11(1-5) : 273–314, 1997.

[Wettschereck et Aha, 1995] D. Wettschereck et D.W. Aha. Weighting features. Dans *Case-Based Reasoning, Research and Development, First International Conference*, éditeurs Manuela Veloso et Agnar Aamodt, pages 347–358, Berlin, 1995. Springer Verlag.

[Whitley *et al.*, 1994] D. Whitley, V.S. Gordon, et K. Mathias. Lamarckian evolution, the baldwin effect and function optimization. Dans *Parallel Problem Solving from Nature (PPSN III)*, pages 6–16, 1994.

[Whitley et Starkweather, 1990] D. Whitley et T. Starkweather. GENITOR II. : a distributed genetic algorithm. *Journal of Experimental & Theoretical Artificial Intelligence*, 2(3) : 189–214, 1990.

[Whitley, 1995] Darrell L. Whitley. Modeling hybrid genetic algorithms. Dans *Genetic Algorithms*, éditeurs G. Winter, J. Périaux, M. Galán, et P. Cuesta, pages 191–201. John Wiley and Sons, Chichester, 1995.

[Woo *et al.*, 2005] K.-G. Woo, J.-H. Lee, M.-H. Kim, et Y.-J. Lee. FINDIT : a fast and intelligent subspace clustering algorithm using dimension voting. *Information and Software Technology*, 46(4) : 255–271, 2005.

[Xie et Beni, 1991] X. Xie et G. Beni. A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(8) : 841–847, 1991.

[Xu et Wunsch, 2005] R. Xu et D. Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3) : 645–678, 2005.

[Yang et Honavar, 1997] J. Yang et V. Honavar. Feature subset selection using a genetic algorithm. Rapport Technique 97–02a, Iowa Sate University, Artificial Intelligence Research Group, Department of Computer Science, 1997.

[Yang et Honavar, 1998] J. Yang et V. Honavar. Feature subset selection using a genetic algorithm. *IEEE Intelligent Systems*, 13 : 44–49, 1998.

[Yeung et Wang, 2002] D.S. Yeung et X.Z. Wang. Improving performance of similarity-based clustering by feature weight learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24 : 556–561, 2002.

[Yong et Miikkulainen, 2001] C.H. Yong et R. Miikkulainen. Cooperative coevolution of multi-agent systems. Rapport technique, University of Texas at Austin Department of Computer Sciences, 2001.

[Yuhas *et al.*, 1992] R.H. Yuhas, A.F.H. Goetz, et J.W. Boardman. Discrimination among semi-arid landscape endmembres using the spectral angle mapper (sam) algorithm. Dans *Summaries of the 3rd annual JPL Airborne Geoscicence Workshop*, pages 147–149, 1992.

[Zadeh, 1965] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8(3) : 338–353, 1965.

Annexes

Annexe A

Mesures de distance et de similarité

A.1 Distance et similarité

De nombreuses méthodes de classification sont basées sur une mesure de distance entre les objets, comme par exemple la méthode K-means. Les mesures de distance les plus fréquemment utilisées sont la distance euclidienne et la distance de Manhattan. Des méthodes peuvent également utiliser une mesure de similarité (Définition A.2), généralement basée sur une mesure de dissimilarité.

Définition A.1 (Distance et dissimilarité)
On appelle distance *sur un ensemble D une application $d : D \times D \to \mathbb{R}^+$ telle que :*

- $d(o, o') = d(o', o), \forall o, o' \in D$ *(symétrie) ;*
- $d(o, o') = 0 \Leftrightarrow o = o', \forall o, o' \in D$ *(séparation) ;*
- $d(o, o'') \leq d(o, o') + d(o', o''), \forall o, o', o'' \in D$ *(inégalité triangulaire).*

On appelle dissimilarité *sur un ensemble D une application $d : D \times D \to \mathbb{R}^+$ telle que :*

- $d(o, o') = d(o', o), \forall o, o' \in D$ *(symétrie) ;*
- $d(o, o') = 0 \Leftrightarrow o = o', \forall o, o' \in D$ *(séparation).*

La distance est donc un cas particulier de mesure de dissimilarité.

La distance entre deux objets o et o' selon la norme L^p peut être définie par :

$$d(o, o') = \left(\sum_{1 \leq j \leq n} (d_j(o, o'))^p \right)^{\frac{1}{p}}$$

où $d_j(o, o')$ est la distance entre o et o' pour le j-ième attribut.

La distance de Manhattan est définie selon la norme L^1, la distance euclidienne selon la norme L^2.

Définition A.2 (Mesure de similarité)
Une mesure de similarité *entre deux objets o et o' peut être définie par :*

$$S(o, o') = e^{-\alpha \times d(o, o')}$$

ou par :

$$S(o, o') = \frac{1}{1 + \beta + d(o, o')}$$

où $d(o, o')$ est une mesure de dissimilarité entre o et o' et α et β sont deux réels positifs.

Ces mêmes méthodes de classification nécessitent également de réaliser des opérations algébriques sur les données, en particulier des calculs de moyennes. Ces opérations sont généralement réalisées indépendamment sur chacun des attributs.

Nous étudierons donc plus précisément les spécificités des principaux types de données utilisés en classification non supervisée :

- les données numériques (section A.2) ;
- les données catégorielles (section A.3) ;
- les histogrammes (section A.4).

A.2 Attributs numériques

On appelle *attribut numérique* un attribut qui prend ses valeurs dans un intervalle de $\mathbb{R}$. On appelle *attribut numérique discret* un attribut qui prend un nombre fini de valeurs (généralement dans $\mathbb{R}$). On note $V(F_j) = \left\{F_j^1, \ldots, F_j^m\right\}$ l'ensemble des valeurs discrètes que peut prendre un attribut numérique discret F_j. La particularité des données numériques (discrètes ou non) est qu'une relation d'ordre total y est définie.

Ces deux types d'attributs utilisent la même mesure de distance (Définition A.3).

Définition A.3 (Distance sur un attribut numérique)
La distance entre deux objets o et o' sur un attribut numérique ou numérique discret F_j peut être définie par :

$$d_j(o, o') = \left|o_j - o'_j\right|$$

Les opérations algébriques se font de manière triviale. Il faut cependant noter que dans le cas d'un attribut numérique discret F_j, les opérations telles que la moyenne peuvent produire une valeur en dehors de $V(F_j)$.

A.3 Attributs catégoriels

On appelle *attribut catégoriel* un attribut qui prend ses valeurs dans un ensemble fini de valeurs sans relation d'ordre appelées *modalités*. Un attribut de ce type est parfois aussi appelé *attribut symbolique* ou *attribut nominal*. On note $V(F_j) = \left\{F_j^1, \ldots, F_j^m\right\}$ l'ensemble des modalités que peut prendre un attribut catégoriel F_j.

La valeur d'un objet o sur un attribut catégoriel F_j peut être représentée par un vecteur $o_j = (o_{j,1}, \ldots, o_{j,m})$, avec $o_{j,i} = 1$ si l'attribut prend la i-ième valeur et $o_{j,i} = 0$ sinon.

Plusieurs mesures de distance peuvent être définies sur un attributs catégoriel (Définitions A.4 et A.5).

Définition A.4 (Distance sur un attribut catégoriel)
La distance entre deux objets o et o' sur un attribut catégoriel F_j peut être définie par :

$$d_j(o, o') = \begin{cases} 1 \text{ si } o_j = o'_j \\ 0 \text{ sinon} \end{cases}$$

En représentant un attribut catégoriel par un vecteur, la distance entre deux objets o et o' sur un attribut catégoriel F_j peut être définie par :

$$d_j(o, o') = \sum_{i=0}^{m} \left|o_{j,i} - o'_{j,i}\right|$$

La distance de Ralambondrainy permet que des modalités ayant une faible fréquence d'apparition dans D aient une influence plus grande sur la distance entre deux objets que des modalités ayant une forte fréquence d'apparition [Ralambondrainy, 1995].

DÉFINITION A.5 (DISTANCE DE RALAMBONDRAINY SUR UN ATTRIBUT CATÉGORIEL)
La distance de Ralambondrainy entre deux objets o et o' sur un attribut catégoriel F_j est définie par :

$$d_j(o, o') = \sum_{i=0}^{m} \frac{\left|o_{j,i} - o'_{j,i}\right|}{n_{j,i}(D)}$$

où $n_{j,i}(D)$ est le nombre d'occurrences de la i-ième valeur catégorielle de j-ième attribut dans l'ensemble de données D.

Les opérations algébriques deviennent triviales en utilisant la représentation par vecteur d'un attribut catégoriel. Les valeurs des éléments du vecteur ne seront plus uniquement égales à 0 ou 1 uniquement, mais pourront représenter la probabilité qu'un objet prennent l'une ou l'autre des modalités.

Exemple :

Un attribut catégoriel F_j défini par 3 modalités a, b ou c peut être représenté par 3 attributs numériques. La modalité a est alors représenté par le vecteur $(1, 0, 0)$, la modalité b par le vecteur $(0, 1, 0)$ et la modalité c par $(0, 0, 1)$.
Soit c_k le centre d'une classe composée de 6 objets qui ont pour valeur pour l'attribut F_j : a pour trois d'entre eux, b pour deux autres et c pour le dernier. La valeur de c_k pour le j-ième attribut est alors un vecteur $(1/2, 1/3, 1/6)$.

A.4 Histogrammes

Un *histogramme* est fréquemment utilisé comme attribut pour caractériser des données. Un histogramme est construit à partir d'un ensemble d'observations $X = \{x_1, \ldots, x_b\}$. Chacune de ses observations prend une valeur parmi un nombre fini de modalités $V = \{v^1, \ldots, v^m\}$. On calcule alors $H_i(X) = card(x_l \in X \mid x_l = v_i)$. On défini alors l'histogramme $H(X) = [H_1(X), \ldots, H_m(X)]$.

Ce type d'attribut est plus complexe et a donné lieu à de nombreux travaux de recherche concernant des mesures de distance pouvant être utilisées pour la classification non supervisée [Cha et Srihari, 2000a ; Wang et Zhang, 2001 ; Cha et Srihari, 2000b].

Trois types d'histogrammes ont été définis, selon la relation d'ordre qu'il existe entre les différentes modalités de V. Dans un *histogramme nominal*, il n'y a pas de relation d'ordre entre les différentes modalités V. Dans un *histogramme ordinal*, il existe une relation d'ordre, comme c'est le cas pour les niveaux de gris des pixels d'une image. Dans un *histogramme modulo*, il existe une relation d'ordre cyclique, comme c'est le cas par exemple pour une mesure d'angle discrétisée.

Il existe plusieurs mesures de distance pour chacun des types d'histogrammes. Nous ne présentons ici qu'une définition simple de la distance entre deux histogrammes nominaux.

DÉFINITION A.6 (DISTANCE ENTRE HISTOGRAMMES NOMINAUX)
La distance entre deux objets o et o' sur un attribut F_j défini par un histogramme nominal de m valeurs peut être définie par :

$$d_j(o, o') = \sum_{i=1}^{m} \left|o_{j,i} - o'_{j,i}\right|$$

Les opérations algébriques doivent être traitées en fonction de la mesure de distance utilisée. Si l'on utilise la mesure présentée dans la définition A.6, il suffit de traiter un histogramme comme un vecteur de valeurs réelles.

Annexe B

Critères d'évaluation supervisée pour la sélection ou la pondération d'attributs

B.1 Évaluation de l'importance d'un attribut

La sélection d'attributs consite à choisir les meilleurs attributs à utiliser pour la classification. Comme cela a été exposé dans la section 3.2.5, la recherche de ces meilleurs attributs peut se faire en évaluant un par un leur qualité discriminatoire. Dans cette section seront présentés différents critères d'évaluation de l'importance d'un attribut pour une tâche de classification supervisée.

B.1.1 Critères basés sur la distance

Dans [Kira et Rendell, 1992 ; Kononenko, 1994] une mesure d'importance d'un attribut, basée sur la distance entre objets d'une même classe et entre objets de classes différentes, est proposée :

$$I_j(D) = \frac{1}{N} \sum_{o \in D} \left(-d_j\left(o, v_{o,C(o)}\right) + \sum_{C_k \neq C(o)} p(C_k)\, d_j\left(o, v_{o,C_k}\right) \right)$$

où v_{o,C_k} est un objet de la classe C_k, le plus proche de o, d_j est une mesure de distance sur le j-ième attribut, normalisée entre 0 et 1 et $p(C_k)$ est la probabilité qu'un objet appartienne à la classe C_k.

Plus l'indice I_j est élevé, plus l'attribut numérique F_j est important pour classifier les données.

Dans [Aggarwal *et al.*, 1999], l'importance d'un attribut F_j pour une classe est définie de la manière suivante :

- pour chaque classe, la distance moyenne $\mu_{k,j}$ entre les objets qui la composent et le centre de la classe est calculée pour chaque attribut ;
- on définit alors les valeurs $\mu_k = \frac{1}{n} \sum_{j=1}^{n} \mu_{k,j}$ et $\sigma_k = \sqrt{\frac{1}{n-1} \sum_{j=1}^{n} \left(\mu_{k,j} - \mu_k\right)^2}$;
- une mesure de compacité du j-ième attribut pour la k-ième classe est alors défini par :

$$Z_j(C_k) = \frac{\mu_{k,j} - \mu_k}{\sigma_k}$$

Une faible valeur de $Z_j(C_k)$ indique une forte compacité de la k-ième classe sur le j-ième attribut. L'attribut sera alors considéré comme pertinent pour cette classe.

Dans [Shiv Naga Prasad *et al.*, 2002] plusieurs mesures de compacité des classes sont proposées pour évaluer l'importance d'un attributs (numérique) pour la classification en deux classes C_k et $C_{k'}$:

$$\begin{aligned}
DM_j(C_k, C_{k'}) &= \frac{|\mu_{k,j} - \mu_{k',j}|}{\sigma_{k,j} + \sigma_{k',j}} \\
IS_j(C_k) &= \frac{1}{\sigma_{k,j}} \\
MC_j(C_k, C_k) &= \frac{1}{N}\left(card\left(\left\{o \in C_k \mid \frac{|\mu_{k,j} - o_j|}{\sigma_{k,j}} < \theta\right\}\right) + \right. \\
&\qquad \left. card\left(\left\{o \in C_{k'} \mid \frac{|\mu_{k',j} - o_j|}{\sigma_{k',j}} < \theta\right\}\right)\right)
\end{aligned}$$

où $\mu_{k,j}$ et $\sigma_{k,j}$ sont respectivement les moyennes et l'écart type de la classe C_k pour le j-ième attribut et θ est un réel strictement positif.

Une valeur élevée sur les trois indices DM_j, IS_j ou MC_j indique une grande importance de l'attribut F_j pour la discrimination des classes C_k et $C_{k'}$.

B.1.2 Critères basés sur l'entropie

Dans [Koller et Sahami, 1996 ; Shiv Naga Prasad *et al.*, 2002] plusieurs mesures d'entropie sont proposées pour évaluer l'importance d'un attribut catégoriel, ou numérique discrétisé en m valeurs discrètes, pour la classification en deux classes C_k et $C_{k'}$:

$$\begin{aligned}
E_j(C_k) &= \log m + \sum_{i=1}^{m}\left(p_i^{k,j} \log p_i^{k,j}\right) \\
E_j(C_k, C_{k'}) &= 2\log m + \sum_{i=1}^{m}\left(p_i^{k,j} \log p_i^{k,j}\right) + \sum_{i=1}^{m}\left(p_i^{k',j} \log p_i^{k',j}\right) \\
KL_j(C_k, C_{k'}) &= \sum_{i=1}^{m}\left(p_i^{k,j} \log \frac{p_i^{k,j}}{p_i^{k',j}}\right) \\
KL_j(C_k, C_{k'}) &= \sum_{i=1}^{m}\left(p_i^{k,j} \log \frac{p_i^{k,j}}{p_i^{k',j}}\right) + \sum_{i=1}^{m}\left(p_i^{k',j} \log \frac{p_i^{k',j}}{p_i^{k,j}}\right)
\end{aligned}$$

où $p_i^{k,j}$ est la probabilité que le j-ième attribut d'un objet de la k-ième classe prenne la i-ième valeur discrète.

Ces quatre mesures sont des mesures d'entropie, c'est-à-dire d'incertitude. Plus l'entropie est basse pour un attribut, plus celui-ci contient d'information et plus il sera important pour la classification.

B.1.3 Critères basés sur la dépendance

Dans [Bell et Wang, 2000] un critère de pertinence d'un attribut basé sur la dépendance est proposé. Ce critère utilise la notion d'entropie pour définir la dépendance entre deux attributs :

$$r_{j,j'}(D) = \frac{H(F_{j'}) - H(F_{j'}|F_j)}{H(F_{j'})}$$

où $H(F_{j'})$ est une mesure de l'entropie des données selon l'attribut $F_{j'}$ et $H(F_{j'} \mid F_j)$ est l'entropie conditionnelle de $F_{j'}$ sachant F_j.

Une forte valeur sur l'indice r indique que les deux attributs sont indépendants.

B.1.4 Synthèse

De nombreux critères ont été définis dans la littérature. Ces critères sont basés sur des paradigmes différents et s'appliquent sur des types de données différents. Les caractéristiques de ces différents critères d'évaluation sont résumées sur la table B.1.

Indice	Type de critère	Type de données	Référence
I	distance	quelconque	[Kononenko, 1994]
Z	distance	numérique	[Aggarwal *et al.*, 1999]
DM	distance	numérique	[Shiv Naga Prasad *et al.*, 2002]
IS	distance	numérique	[Shiv Naga Prasad *et al.*, 2002]
MC^*	distance	numérique	[Shiv Naga Prasad *et al.*, 2002]
E	entropie	nominal, discret	[Shiv Naga Prasad *et al.*, 2002]
KL	entropie	nominal, discret	[Shiv Naga Prasad *et al.*, 2002]
r	dépendance	nominal, discret	[Bell et Wang, 2000]

*cette mesure nécessite un paramètre

Tab. B.1 : *Critères d'évaluation supervisée de l'importance d'un attribut*

B.2 Évaluation de la pertinence d'un sous-ensemble ou d'une pondération des attributs

Comme cela a été exposé dans la section 3.2.5 certain critères ne consistent pas à évaluer l'importance des attributs indépendamment les uns des autres, mais à évaluer un sous-ensemble ou une pondération des attributs. Ici encore, de nombreux critères ont été définis dans la littérature. Dans cette section, nous présentons les critères qui s'appliquent dans le cadre de la classification supervisée.

B.2.1 Critères basés sur la distance

Dans [Scherf et Brauer, 1997], une mesure de pertinence d'un sous-ensemble d'attributs ou d'un vecteur de poids, basée sur la compacité des classes de l'ensemble d'apprentissage L, est définie par :

$$J_W(D) = \sum_{o,o' \in D} \delta(o,o') \frac{d_W(o,o')}{N_s} - (1 - \delta(o,o')) \frac{d_W(o,o')}{N_v}$$

avec N_s le nombre de paires d'objets dans la même classe, N_v le nombre de paires d'objets de classes différentes et $\delta(o,o') = 1$ si o et o' sont dans la même classe et $\delta(o,o') = 0$ sinon.

Une faible valeur indique une forte séparabilité des classes et donc que la pondération est pertinente.

Dans [Wang, 2005] la qualité discriminante d'un sous-ensemble d'attributs (numériques) entre deux classes C_k et $C_{k'}$ est évaluée par la distance de Jeffries-Matusita :

$$JM_W\left(C_k, C'_k\right) = \sqrt{2\left(1 - e^{-\alpha_{k,k'}}\right)}$$
$$\text{avec } \alpha_W\left(k, k'\right) = \frac{1}{8}\left(c_k - c_{k'}\right)^T \left(\frac{Cov_k + Cov_{k'}}{2}\right)^{-1} \left(c_k - c_{k'}\right) + \frac{1}{2}\ln\left(\frac{\left|\frac{1}{2}\left(Cov_k + Cov_{k'}\right)\right|}{\sqrt{|Cov_k| \times |Cov_{k'}|}}\right)$$

où c_k et $c_{k'}$ sont les centres respectifs des classes C_k et $C_{k'}$ et Cov_k et $Cok_{k'}$ sont les matrices de covariance des classes C_k et $C_{k'}$

Ce critère est une mesure de distance entre les classes. Une forte valeur indique donc une forte séparabilité des classes et une forte pertinence du sous-ensemble d'attributs ou de la pondération évaluée.

B.2.2 Critères basés sur la consistance

Dans [Liu *et al.*, 1998 ; Dash et Liu, 2003], une mesure d'inconsistance d'un ensemble d'attributs pour des données nominales est proposée. Une inconsistance apparaît entre deux objets o et o' s'ils ont la même valeurs sur tous les attributs sélectionnés, mais des classes différentes. L'inconsistance d'un sous-ensemble des attributs F' est définie de la manière suivante :

$$Inc_{F'}\left(L\right) = \frac{1}{N} \sum_{\{p|n_p \neq 0\}} \left(n_p - n_p^{max}\right)$$

où n_p est le nombre d'objets égaux à p dans l'ensemble d'apprentissage L, $n_p = \sum_{k=1}^{K} n_p^k$, où n_p^k est le nombre d'objets de la k-ième classe égaux à p et $n_p^{max} = \max_{k=1\ldots K} n_p^k$.

Ce critère consiste donc à compter la proportion d'objets inconsistants dans l'ensemble de données selon le sous-ensemble d'attributs considéré. Plus ce ratio sera bas, plus le sous-ensemble d'attributs sera pertinent.

B.2.3 Synthèse

De nombreux critères ont été définis dans la littérature. Ces critères sont basés sur des paradigmes différents et s'appliquent sur des types de données différents. Les caractéristiques de ces différents critères d'évaluation sont résumées sur la table B.2.

Indice	Type de critère	Type de données	Référence
J	distance	quelconque	[Scherf et Brauer, 1997]
JM	distance	numérique	[Wang, 2005]
Inc	consistance	nominal	[Dash et Liu, 2003]
Qualité de classification*	évaluation de classification	dépend de la méthode	[Kohavi et John, 1998]

*approche enveloppe

Tab. B.2 : *Critères d'évaluation supervisée de la pertinence d'une pondération des attributs*

Annexe C

Évaluation d'une classification non supervisée par des critères externes

C.1 Comparaison de résultats de classification

Il est possible d'évaluer la pertinence du résultat d'une classification sur un ensemble de données pour lequel la classe réelle de chaque objet est connue, en comparant le résultat obtenu avec la classification réelle. Ce type d'évaluation est souvent appelé évaluation par critères externes.

Cette comparaison n'est pas aisée, étant donné qu'il n'est pas possible de faire directement la correspondance d'une classe du résultat de la classification non supervisée avec une classe réelle des données. Les méthodes couramment employées consistent à utiliser des critères de comparaison de partitions. Pour cela, il est nécessaire de transformer tous les résultats de classification en partitions, c'est-à-dire en classifications dures, sans hiérarchie de classes.

Nous allons présenter différents critères de comparaison entre deux résultats de classifications $C^1 = \{C_1^1, C_2^1, \ldots, C_{K_1}^1\}$ et $C^2 = \{C_1^2, C_2^2, \ldots, C_{K_2}^2\}$ sur un ensemble D de N éléments.

Nous présentons ici des critères classiques, tous basés sur un même principe (section C.2), et l'indice de Wemmert et Gançarski qui se distingue de tous les autres critères de comparaison (section C.3).

C.2 Critères classiques

La plupart des critères se basent sur des critères utilisés en classification supervisée [van Rijsbergen, 1979 ; Halkidi *et al.*, 2001b]. Comme il n'est pas possible de faire une correspondance entre les classes des deux résultats, les critères ne sont pas évalués directement sur les classes des objets, mais sur des paires d'objets : dans une classification dure, deux objets peuvent appartenir à la même classe ou à deux classes différentes.

On notera $\mathcal{P}_2(D)$ l'ensemble des paires de D, c'est-à-dire l'ensemble des sous-ensembles de deux éléments de D. $M = card\,(\mathcal{P}_2(D)) = 1/2 \times N \times (N-1)$ désigne le nombre de paires d'objets.

On calcule alors mm le nombre de paires d'objets qui sont dans la même classe dans C^1 et dans C^2, dd le nombre de paires d'objets qui sont dans deux classes différentes dans C^1 et dans C^2, md le nombre de paires d'objets qui sont dans la même classe dans C^1 mais dans deux classes

différentes dans C^2 et dm le nombre de paires d'objets qui sont dans deux classes différentes dans C^1 mais dans la même classe dans C^2. On a $mm + dd + md + dm = M$.

Plus mm et dd sont élevés et md et dm sont bas, plus on considère que les partitions sont similaires.

Plusieurs indices ont ainsi été définis (Définitions C.1 à C.8). Sur chacun de ces critères une valeur forte indique une forte ressemblance entre les partitions, et une valeur faible indique une faible ressemblance. En dehors de la précision (Définition C.1) et du rappel (Définition C.2), tous les indices sont symétriques.

Définition C.1 (Précision)
La précision *est la probabilité pour que deux objets soient dans la même classe dans C^2 s'il le sont dans C^1 :*

$$prec = \frac{mm}{mm + md}$$

La précision prend ses valeurs sur $[0; 1]$. Mais une valeur de 1 ne garantie pas que les deux classifications sont identiques. Ce critère suppose que C^2 représente la classification réelle des données.

Définition C.2 (Rappel)
Le rappel *est la probabilité pour que deux objets soient dans la même classe dans C^1 s'il le sont dans C^2 :*

$$rapp = \frac{mm}{mm + dm}$$

Le rappel prend ses valeurs sur $[0; 1]$. Mais une valeur de 1 ne garantie pas que les deux classifications sont identiques. Ce critère suppose que C^2 représente la classification réelle des données.

Définition C.3 (Statistique de Rand)
La statistique de Rand *est définie par :*

$$R = \frac{mm + dd}{M}$$

La statistique de Rand prend ses valeurs sur $[0; 1]$. La statistique de Rand vaut 1 si et seulement si les deux classifications sont identiques.

Définition C.4 (Coefficient de Jaccard)
Le coefficient de Jaccard *est défini par :*

$$J = \frac{mm}{mm + md + dm}$$

Le coefficient de Jaccard prend ses valeurs sur $[0; 1]$. Le coefficient de Jaccard vaut 1 si et seulement si les deux classifications sont identiques.

Définition C.5 (Indice de Folkes et Mallows)
*L'*indice de Folkes et Mallows *est défini par la moyenne géométrique entre la précision (Définition C.1) et le rappel (Définition C.2) :*

$$FM = \sqrt{prec \times rapp}$$

L'indice de Folkes et Mallows prend ses valeurs sur $[0; 1]$. L'indice de Folkes et Mallows vaut 1 si et seulement si la précision et le rappel valent 1 tous les deux, et donc si et seulement si les deux classifications sont identiques.

Définition C.6 (F-measure)
La F-measure *est définie par :*

$$F - M. = \frac{2 \times prec \times rapp}{prec + rapp}$$

La F-measure prend ses valeurs sur $[0;1]$. La F-measure vaut 1 si et seulement si la précision et le rappel valent 1 tous les deux, et donc si et seulement si les deux classifications sont identiques.

Définition C.7 (Statistique de Hubert)
La statistique de Hubert est définie par :

$$\Gamma = \frac{1}{M} \sum_{\{o,o'\} \in \mathcal{P}_2(D)} S^1(o,o') S^2(o,o')$$

avec $S^i(o,o') = 1$ si o et o' sont dans la même classe dans la classification C^i et $S^i(o,o') = 0$ sinon.

Il est possible de normaliser cet indice pour qu'il prenne ses valeurs entre -1 et 1 :

$$\overline{\Gamma} = \frac{1}{M} \times \frac{1}{\sigma_1 \sigma_2} \times \sum_{\{o,o'\} \in \mathcal{P}_2(D)} \big(S^1(o,o') - \mu_1\big)\big(S^2(o,o') - \mu_2\big)$$

avec μ_i et σ_i représentent la moyenne et la variance de S^i.

Définition C.8 (Coefficient κ)
Le coefficient κ est défini par :

$$\kappa = \frac{P_0 - P_e}{1 - P_e}$$

avec $P_0 = \frac{mm+dd}{M}$ et $P_e = \frac{1}{M^2} \times (mm + md) \times (mm + dm) + (md + dd) \times (dm + dd)$.

Le coefficient κ prend ses valeurs sur $[-1;1]$. Une forte valeur sur le coefficient κ indique une forte similarité entre les classification.

C.3 Indice de Wemmert et Gançarski

L'indice de Wemmert et Gançarski (Définition C.12) évalue la ressemblance entre deux résultats de classification sur un tout autre principe que celui des indices classiques. Il se base sur la répartition des classes d'une des classifications dans celles de l'autre classification [Wemmert, 2000].

Définition C.9 (Matrice de confusion et coefficient de confusion)
La matrice de confusion *entre deux classifications C^1 et C^2 est définie par :*

$$\mathcal{C}^{i,j} = \begin{pmatrix} \alpha^{i,j}_{1,1} & \cdots & \alpha^{i,j}_{1,K_j} \\ \vdots & \ddots & \vdots \\ \alpha^{i,j}_{K_i,1} & \cdots & \alpha^{i,j}_{K_i,K_j} \end{pmatrix}$$

où $\alpha^{i,j}_{k,l} = \frac{|C^i_k \cap C^j_l|}{|C^i_k|}$ est appelé le coefficient de confusion *entre C^i_k et C^j_l.*

Définition C.10 (Coefficient de répartition)
Le coefficient de répartition *d'une classe C_k^i dans une classification C^j est défini par :*

$$\rho_k^{i,j} = \sum_{l=1}^{K_j} \left(\alpha_{k,l}^{i,j}\right)^2$$

Plus $\rho_k^{i,j}$ est proche de 1, plus cela signifie que les objets de la classe C_k^i se trouvent dans une même classe de C^j. Au contraire, plus $\rho_k^{i,j}$ est proche de 0, plus cela implique que les objets de C_k^i sont répartis dans les classes de C^j.

Définition C.11 (Critère local de similitude)
Le critère local de similitude *d'une classe C_k^i dans une classification C^j évalue si la classe C_k^i est similaire à une classe de C^j. Sa définition est :*

$$\omega_k^{i,j} = \rho_k^{i,j} \times \max_{l=1,\ldots,K_j} \left(\alpha_{k,l}^{i,j}\right)$$

Définition C.12 (Indice de Wemmert de Gançarski)
*L'*indice de Wemmert de Gançarski *de similarité entre deux classifications C^1 et C^2 est défini par :*

$$WG = \frac{1}{2}\left(\frac{1}{K_1}\sum_{k=1}^{K_1}\omega_k^{1,2} + \frac{1}{K_2}\sum_{k=1}^{K_2}\omega_k^{2,1}\right)$$

Une forte valeur indique une grande ressemblance entre les classes. Ce critère d'évaluation est symétrique.

Annexe D

Observation de la Terre

D.1 Télédétection

La télédétection est une discipline scientifique ayant pour objet l'observation de la Terre. Elle regroupe tout un ensemble de processus : capter et enregistrer le rayonnement électromagnétique émis ou réfléchi par les surfaces observées sous forme d'images, mais aussi traiter et analyser les données [Bonn et Rochon, 1992 ; CCRS, 2001].

L'information obtenue par chaque capteur est un ensemble d'images en niveaux de gris en deux dimensions. Le niveau de gris d'un pixel correspond à la réponse spectrale de la surface observée, sur la plage de longueurs d'onde captée. Un capteur *panchromatique* est un capteur sensible à l'ensemble du spectre visible et produit une image en niveaux de gris. Un capteur *multispectral* (ou *multibande*) est un capteur sensible à un ensemble de bandes spectrales, produisant simultanément plusieurs images en niveaux de gris, correspondant chacune à une des bandes spectrales. On appelle généralement capteur *hyperspectral* un capteur sensible à un grand nombre de bandes spectrales (plusieurs dizaines).

L'information brute obtenue est une information de *radiance* (ou *luminance énergétique*) correspondant à l'énergie émise ou réfléchie par la surface. Cette information est transformée en *réflectance* qui est le rapport du flux réfléchi par la surface au flux incident.

Les images de télédétection peuvent être de natures très différentes. De telles images peuvent être prises à partir de plateformes aériennes (ballons, avions) ou spatiales (satellites SPOT, satellites Landsat, ...). Chaque type d'image a des caractéristiques différentes. La résolution spatiale, c'est-à-dire la taille des pixels, varie de plusieurs dizaines de mètres (parfois même quelques kilomètres) à une résolution inférieure au mètre. On distingue généralement, dans le domaine civil, la basse résolution (1000 m), la moyenne résolution (80 m), la haute résolution (10 à 30 m) et la très haute résolution (inférieure à 5 m).

La résolution spectrale, c'est-à-dire la taille de l'intervalle de chaque bande spectrale varie de 1000 nm à 2 nm. Le nombre de bandes peut aller jusqu'à plusieurs centaines. On distingue généralement quatre régions spectrales [Bonn et Rochon, 1992] :

- le visible (VIS) : 400 nm à 700 nm ;
- le proche infrarouge (NEAR–IR) : 700 nm à 1 500 nm ;
- le moyen infrarouge (MID–IR) : 1 500 nm à 3 000 nm ;
- l'infrarouge lointain (FAR–IR) : 3 000 nm à 15 000 nm ;

Les informations obtenues sont de nature physique (propriétés radiométriques et spatiales des surfaces observées). Ces informations physiques peuvent cependant être mises en relation avec l'aspect fonctionnel des surfaces, c'est-à-dire l'utilisation qui est faite du sol. Sur la figure D.1 sont représentées les signatures spectrales de trois types de surfaces (sol nu sec, végétation et

eau), c'est-à-dire la réponse spectrale de ces surfaces selon la longueur d'onde. On remarque un pic de réflectance de la végétation dans le vert (aux environs de 500 nm), ainsi qu'une très forte réflectance dans le proche infrarouge. Le sol nu (et plus généralement les surfaces minéralisées) a une forte réponse spectrale dans le moyen infrarouge. L'eau présente une très faible réflectance, tout particulièrement dans l'infrarouge. On voit également sur cette figure que l'ordre de grandeur des valeurs de réflectance est bien plus faible dans le visible que dans l'infrarouge.

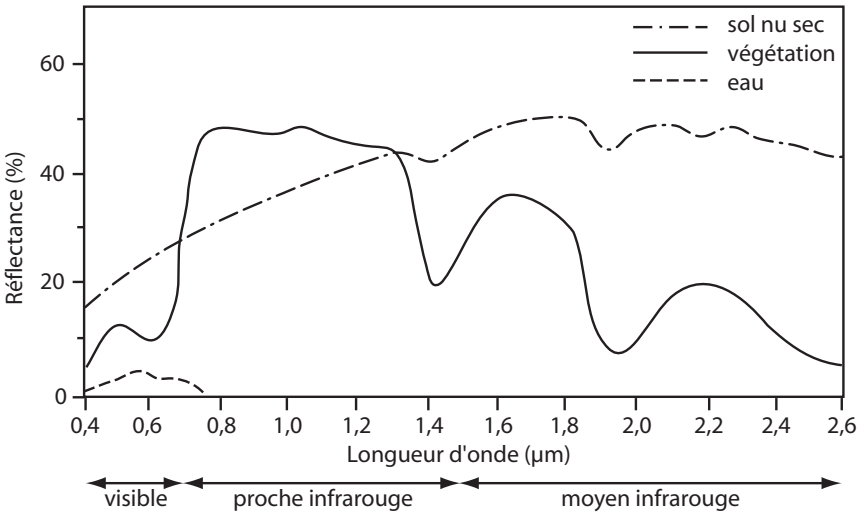


Fig. D.1 : *Signatures spectrales de trois types de surfaces, d'après la figure présentée dans [Lillesand et Kiefer, 1987]*

D.2 Applications de la télédétection

Les images de télédétection constituent une source d'information importante pour l'étude de notre planète. Elles sont utilisées en urbanisme, en sciences environnementales, en météorologie ou encore en géologie. Elles apportent des informations pour la cartographie de l'occupation et de l'utilisation du sol afin de permettre la détection des espaces verts, des surfaces imperméables, ou encore l'analyse des changements. Les applications sont alors diverses : aménagement du territoire, planification urbaine, gestion de l'environnement, renseignements, transport et télécommunication [Puissant, 2003].

D.3 Extraction de l'information dans les images de télédétection

L'extraction de l'information contenue dans une image de télédétection peut être réalisée manuellement par un photo-interprète. Ce processus d'interprétation visuelle est cependant consommateur de temps, d'autant plus que le volume de données augmente avec les nouvelles technologies. Il est, de plus, particulièrement subjectif. L'automatisation de l'extraction de l'information devient alors une nécessité.

Cette extraction automatique d'informations se fait principalement par des techniques de traitement d'images et de fouille de données, en particulier la classification. Plusieurs problèmes, spécifiques aux images de télédétection, rendent la classification dans les images relativement complexe.

En effet, un pixel peut représenter une zone composée de plusieurs objets de natures très différentes (et cela d'autant plus que la résolution spatiale est basse). On parle alors de pixel *mixte* ou de *mixel*.

Exemple :

Sur la figure D.2, on voit que le pixel au centre de l'image sera composé en partie d'une route (en haut), de végétation (à gauche) et d'eau (à droite). Il est alors difficile de déterminer la classe d'appartenance d'un tel pixel.

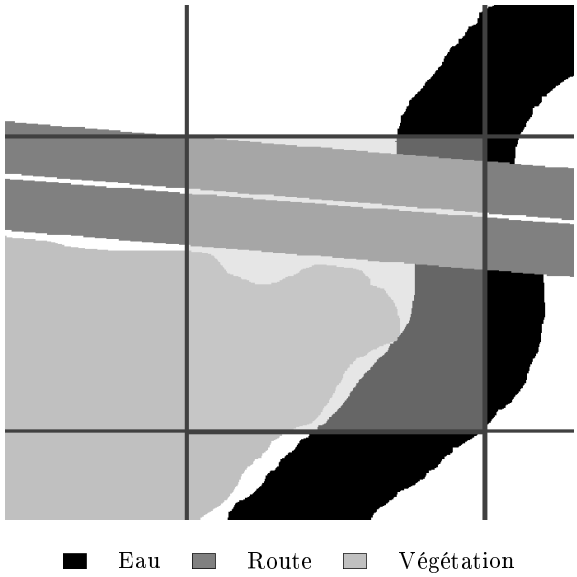


Fig. D.2 : *Pixel mixte dans une image de télédétection*

Le problème inverse peut également survenir, en particulier sur des images à (très) haute résolution. Il est possible qu'un même objet soit composé de pixels très différents. Il est alors nécessaire de tenir compte de la texture, ainsi que du contexte sémantique et topologique, pour classifier correctement les données.

La classification peut être réalisée à deux niveaux différents :

- classification spectrale (au niveau des pixels) : la classification spectrale consiste à découvrir la classe de chaque pixel de l'image en fonction de ses caractéristiques spectrales (et éventuellement en fonction de celles des pixels de son voisinage) ;
- classification zonale (au niveau des objets) : la classification zonale consiste à découvrir la classe de zones (également appelées régions ou segments) obtenues par un processus de segmentation, correspondant chacune à un objet de la scène, et caractérisées selon différents attributs (information sur la radiométrie, la forme ou la texture).

L'analyse d'image hyperspectrales, composées d'un grand nombre de bandes contiguës présentent les problèmes mis en évidence dans le chapitre 3 : corrélations entre les bandes, manque de pertinence pour certaines classes. Ainsi, des méthodes de transformation d'attributs [Benediktsson *et al.*, 1995 ; Kumar *et al.*, 2001 ; Kuo et Landgrebe, 2001 ; Lennon, 2002 ; Kruse et Boardman, 2004] et de sélection d'attributs [Warner et Shank, 1997 ; Serpico et Bruzzone, 2001 ; Groves et Bajcsy, 2003 ; Bajcsy et Groves, 2004] ont été appliquées. Cependant, toutes ces méthodes sont des méthodes supervisées et aucune méthode non supervisée n'a encore été appliquée à notre connaissance.

Résumé

La *classification non supervisée* consiste à diviser un ensemble de données D en sous-ensembles, appelés classes, tel que les objets d'une classe sont similaires et que les objets de classes différentes sont différents, et ce afin d'en comprendre la structure sous-jacente. Les algorithmes de classification non supervisée sont souvent utilisés pour étudier des données pour lesquelles peu d'informations sont disponibles (trop peu d'exemples pour un apprentissage supervisé).

Dans la perspective d'obtenir une classification plus précise, on cherche souvent à décrire les données de la manière la plus détaillée possible, les données étant alors représentées par de nombreux attributs. Cependant, l'augmentation de la dimensionnalité peut parfois nuire à la qualité de la classification car les méthodes classiques de classification ne sont pas adaptées à des données de grande dimensionnalité. Une solution consiste à adapter les données aux algorithmes de classification par une sélection ou une pondération des attributs : des poids binaires (sélection d'attributs) ou réels (pondération d'attributs). Cela permet de faire varier l'influence relative des attributs lors de la classification.

Dans le cadre des méthodes de classification non supervisée, le domaine de recherche est récent et n'a donné lieu qu'à peu de publications. Beaucoup d'algorithmes proposé pour la classification avec sélection/pondération d'attributs sont basés sur l'optimisation d'une fonction d'évaluation. Mais jusqu'à présent, les approches évolutionnaires ont été très peu utilisées, alors qu'elles sont connues pour leur efficacité à résoudre les problèmes d'optimisation.

Deux nouvelles familles de méthodes de classification non supervisée avec pondération d'attributs sont proposées dans cette thèse. La première est une amélioration, par intégration de méthodes évolutionnaires efficaces, de méthodes actuelles basées sur K-means. La seconde repose sur une approche nouvelle en classification non supervisée, la classification modulaire, qui consiste à découper le problème de classification en K classes en K problèmes d'extraction d'une classe. Cette méthode permet de découvrir des pondérations locales, bien que chacun des extracteurs n'utilise que des pondérations globales des données. Enfin, les extracteurs pouvant être définis à partir d'algorithmes de classification différents, cette méthode peut être qualifiée de multi-stratégique. Des expérimentations sur des ensembles de données divers (données artificielles, données de l'UCI) ont montré des résultats satisfaisant de nos algorithmes.

Les images de télédétection, utilisées pour l'observation de la Terre, présentent les caractéristiques correspondant à la problématique étudiée. Dans une image hyperspectrale, chaque pixel est caractérisé par sa réflectance sur un grand nombre de bandes spectrales (plusieurs dizaines) présentant de nombreuses corrélations. Une image peut être découpée en segments (ou régions) par un processus appelé *segmentation*. Ces segments peuvent être caractérisés par des attributs nombreux (informations sur la couleur, la texture ou la forme) et de types hétérogènes (valeurs numériques, valeurs catégorielles, histogrammes ou intervalles de valeurs). Des expériences ont montrés l'intérêt des méthodes proposées dans cette thèse pour ce type de données.