Thèse de doctorat présentée
à l'Université de La Rochelle
pour l'obtention du grade de

DOCTEUR EN INFORMATIQUE

par

STÉPHANIE GUILLAS

RECONNAISSANCE D'OBJETS GRAPHIQUES DÉTÉRIORÉS : APPROCHE FONDÉE SUR UN TREILLIS DE GALOIS

soutenue le 21 novembre 2007 devant le jury composé de

Karl Tombre	Professeur d'université	Président
Yves Lecourtier	Professeur d'université	Rapporteur
Engelbert Mephu Nguifo	Maître de conférences HDR	Rapporteur
Vincent Duquenne	Chargé de recherche au CNRS	Examinateur
Josep Lladós	Professeur d'université	Examinateur
Jean-Marc Ogier	Professeur d'université	Examinateur
Karell Bertet	Maître de conférences	Examinateur

©Stéphanie Guillas, 2007

Résumé

La méthode de classification présentée dans ce manuscrit, se base sur l'utilisation du treillis de Galois. Elle est dédiée à la reconnaissance d'objets graphiques détériorés, et plus particulièrement d'images de symboles. Le treillis de Galois est un graphe dont la structure est proche de celle de l'arbre de décision. Il donne une représentation de toutes les correspondances possibles entre un ensemble d'objets (symboles) et un ensemble d'attributs (descripteurs). Pour atteindre une classe (type de symbole), dans l'arbre de décision il n'y a qu'un seul chemin de la racine vers une feuille, alors que dans le treillis de Galois de nombreuses possibilités de chemins sont définies et représentées. Toutes ces possibilités sont autant de scénarii de classification offrant un réel avantage dans le contexte de reconnaissance de données bruitées.

Un système de reconnaissance itératif a été mis en place, dans le but d'exploiter l'information complémentaire pouvant être fournie par les signatures statistiques et structurelles. Ce système intègre différents types de descriptions des objets (appelées signatures), afin de tirer parti de la complémentarité de ces informations, et ainsi obtenir un apprentissage plus performant. En cas d'ambiguïté sur les données de l'apprentissage, lors de la navigation dans le treillis, il est possible de stopper la progression et d'éviter ainsi certaines erreurs de classification. La description des objets concernés étant ambiguë pour la signature courante, elle peut être renouvelée par des attributs provenant d'un autre type de signature.

Abstract

In this thesis, a classification method based on the use of a concept lattice is presented. It is dedicated to the recognition of noisy graphic objects, and more precisely to symbols recognition. Concept lattice is a graph the structure of which is similar to the decision tree one. It gives a representation of the whole possible correspondences between a set of objects (symbols) and a set of attributes (descriptors). In the decision tree, for reaching a class (symbol type), there is an only path from the root to a leaf, whereas in the concept lattice there is a large number of possibilities of paths. All these possibilities represent scenarii of classification and offer a real advantage in the context of noisy data recognition.

An iterative recognition system has been implemented, in order to use the complementary information given by the statistical and the structural signatures. This system integrates several description types of the objects (called signatures), in order to exploit this complementarity of the data and thus to obtain a more efficient learning stage. In case of ambiguity on the learning data, it is possible to stop the progression in the concept lattice and avoid some classification errors. The description of the objects being ambiguous, it can thus be updated by attributes provided by another type of signature.

Remerciements

Je souhaite tout d'abord adresser mes plus vifs remerciements aux personnes qui me font l'honneur de faire partie de ce jury de thèse.
Monsieur Karl Tombre, professeur à l'École des Mines de Nancy et directeur du Centre de Recherche INRIA de Nancy - Grand Est et du LORIA, de me faire l'honneur de présider ce jury.
Monsieur Yves Lecourtier, professeur à l'université de Rouen, pour avoir accepter d'être rapporteur de ma thèse et pour ses commentaires avisés.
Monsieur Engelbert Mephu Nguifo, Maître de Conférence à l'université d'Artois et chercheur au CRIL Centre de Recherche en Informatique de Lens, pour l'intérêt porté à ce travail et sa lecture attentive du manuscrit.
Monsieur Josep Lladós, chercheur au CVC Computer Vision Center de l'université de Barcelone, pour l'attention qu'il a porté à ce travail.
Monsieur Vincent Duquenne, Chargé de Recherche au CNRS (UMR 7090) à l'université Pierre et Marie Curie (Paris 6), pour le grand intérêt qu'il a montré à ces travaux.

Je tiens à exprimer toute ma reconnaissance et mon amitié à Karell Bertet et Jean-Marc Ogier qui m'ont encadrée de la meilleure des manières au cours de ces trois années. Karell, mon encadrante principale, a toujours su m'épauler et m'a guidée quotidiennement dans mon travail de recherche, tout en me laissant une grande autonomie. J'ai beaucoup apprécié sa disponibilité, son enthousiasme, sa rigueur et sa grande gentillesse. Elle a su me redonner confiance dans les moments de doutes. Quant à Jean-Marc, mon directeur de thèse, il s'est également rendu très disponible malgré ses nombreuses responsabilités. Toujours bienveillant, il a été l'élément moteur pour notamment réaliser certaines publications et fixer les grandes orientations à suivre. Ils possèdent tous les deux de grandes qualités humaines, et m'ont donné l'envie de poursuivre mon objectif de devenir enseignant-chercheur.

Je souhaite également remercier chaleureusement les étudiants qui ont contribué à l'élaboration de ce travail de thèse, au cours de leur stage de master :
Simon Bernard qui a réalisé un état de l'art sur l'extraction de primitives graphiques

et le développement de l'extraction de segments par transformée de Hough pour la construction de la signature structurelle.
Nathalie Girard pour avoir effectué une étude de trois méthodes de classification basées sur le treillis de Galois.
Mickaël Coustaty qui a réalisé un état de l'art sur les relations topologiques entre segments et a achevé le développement de la signature structurelle.
Et pour finir, Antoine Mercier et Philippe Sachot pour avoir réalisé le développement de l'interface du prototype de reconnaissance. Ils ont permis une bonne mise en valeur de ce travail de thèse.
J'ai beaucoup apprécié de les guider dans leurs travaux avec l'aide de Karell et Jean-Marc.

Je tiens à remercier tout particulièrement Muriel Visani, pour ses précieux conseils, sa bonne humeur et le temps qu'elle m'a consacré, Elodie Bichot, pour sa bienveillance et pour les fréquentes discussions amicales que l'on a partagées, Mathieu Delalandre, pour son importante contribution aux encadrements de stage, Christophe Saint-Jean, pour ses explications concernant l'utilisation du logiciel R, Nathalie Blanchard, pour sa gentillesse et son efficacité, ainsi que tous les membres du laboratoire L3I avec qui j'ai pu partager de bons moments.

Côté famille, je souhaite exprimer toute ma gratitude à mes parents, qui m'ont très souvent incitée à aller de l'avant et réconfortée dans les moments difficiles. Sans eux, il n'est pas certain que j'aurais eu la volonté d'entreprendre des études aussi longues. Je vous remercie sincèrement de m'avoir encouragée à poursuivre dans cette voie et surtout de la confiance indéfectible que vous n'avez cessée de me témoigner.
Un grand merci à ma soeur Sandrine pour son soutien et ses nombreuses attentions (culinaires) à mon égard.
Pour finir, une pensée toute particulière pour Ludo, qui m'a soutenue et supportée dans mon travail et au quotidien tout au long de ces trois années et surtout pendant ces derniers mois. Je te remercie vivement pour ta patience, tes conseils et tes motivations toujours très efficaces.

Table des matières

Introduction générale **1**

1 Etat de l'art **4**
1.1 Introduction 4
1.2 Reconnaissance d'images 5
1.2.1 Approches statistiques 6
1.2.2 Approches structurelles 12
1.2.3 Utilisation conjointe des approches statistiques et structurelles . . 23
1.2.4 Classifieurs usuels en reconnaissance d'images 26
1.3 Treillis et Analyse Formelle des Concepts 36
1.3.1 Fondements mathématiques 37
1.3.2 Aspects algorithmiques 46
1.3.3 Classification avec le treillis de Galois 49
1.3.4 Classification basée sur les règles d'association 59
1.4 Conclusion 69

2 Signatures et paramétrage pour les symboles **72**
2.1 Introduction 72
2.2 Description des bases GREC 72
2.2.1 La base GREC 2003 72
2.2.2 La base GREC 2005 74
2.3 Signatures statistiques 76
2.4 Signature structurelle 77
2.4.1 Extraction des segments par transformée de Hough 77
2.4.2 Construction de la signature structurelle à partir des segments extraits 82
2.5 Conclusion 91

3 Reconnaissance à l'aide du treillis de Galois **93**
3.1 Introduction 93
3.2 Apprentissage 94
3.2.1 Normalisation des données 95

3.2.2 Discrétisation des données . . . 96
3.2.3 Construction du treillis de Galois . . . 105
3.3 Classification . . . 109
3.3.1 Principe de navigation . . . 109
3.3.2 Description d'une étape élémentaire de classification . . . 110
3.3.3 Mesure de distance . . . 111
3.3.4 Critère de choix . . . 112
3.4 Reconnaissance itérative . . . 113
3.4.1 Principe . . . 114
3.4.2 Critère d'arrêt de la classification . . . 115
3.5 Paramétrage pour la reconnaissance de symboles . . . 116
3.5.1 Signatures . . . 117
3.5.2 Critère de coupe de la discrétisation . . . 118
3.5.3 Visualisation des symboles de l'apprentissage . . . 119
3.5.4 Comparaison du treillis et des systèmes de règles . . . 123
3.5.5 Comparaison de la construction du treillis avec / sans génération à la demande . . . 125
3.5.6 Mesure de distance et critère de choix . . . 126
3.5.7 Visualisation des étapes de classification . . . 127
3.5.8 Etude des types d'erreurs de classification . . . 145
3.6 Conclusion . . . 148

4 Résultats, discussion et perspectives 149
4.1 Introduction . . . 149
4.2 Etude expérimentale . . . 149
4.2.1 Résultats expérimentaux concernant NAVIGALA . . . 150
4.2.2 Comparaison avec quelques classifieurs usuels sur la reconnaissance de symboles . . . 156
4.2.3 Comparaison avec quelques classifieurs issus de l'AFC sur d'autres bases . . . 170
4.2.4 Conclusion . . . 177
4.3 Discussion et perspectives . . . 178
4.3.1 Phase d'apprentissage . . . 178
4.3.2 Phase de validation . . . 179
4.3.3 Phase de reconnaissance . . . 181
4.4 Conclusion . . . 185

Conclusion générale 186

A NAVIGALA 205
A.1 Description du prototype . . . 205

A.2 Interface graphique 206

Liste des tableaux

1.1 Synthèse des propriétés des signatures statistiques étudiées 13
1.2 Synthèse des propriétés des méthodes d'extraction de primitives graphiques 15
1.3 Comparaison des signatures structurelles sous la forme d'une liste de valeurs numériques 19
1.4 Comparaison des signatures structurelles sous forme de graphes 23
1.5 Propriétés des signatures structurelles 24
1.6 Récapitulatif des avantages et inconvénients des différents classifieurs . . 35
1.7 Récapitulatif des avantages et inconvénients des différents classifieurs (suite) 36
1.8 Exemple de contexte 41
1.9 Propriétés des méthodes de classification basées sur les treillis de Galois . 59
1.10 Propriétés des méthodes de classification basées sur les treillis de Galois (suite) 60
1.11 Propriétés des méthodes de classification basées sur les règles d'association 69
1.12 Propriétés des méthodes de classification basées sur les règles d'association (suite) 70

2.1 Matrice d'adjacence associée au graphe de vecteurs 86
2.2 Matrice d'adjacence élevée au carré 87
2.3 Exemple de signature structurelle obtenue 88
2.4 Taille des signatures obtenues pour différentes longueurs de chemins . . . 91
2.5 Fréquence d'apparition des chemins par symbole (chemins de longueur 1 et 2) 92

3.1 Signatures des 10 objets 99
3.2 Intervalles obtenus après initialisation de la relation d'appartenance R . . 100
3.3 Table discrétisée selon un critère d'entropie 101
3.4 Dimensions et taux de reconnaissance obtenus en validation croisée (5 blocs) 123
3.5 Evolution de la taille des structures en fonction de la taille de S ($|\Sigma| = 15$) 124
3.6 Evolution de la taille des structures en fonction de la taille de Σ ($|S| = 7$) 124
3.7 Apprentissage avec 25 symboles modèles - Reconnaissance de 10 symboles bruités 126

4.1 Evolution des taux de reconnaissance (en %) obtenus aux 5 tirages (v0 à v4) en fonction de la taille de l'ensemble d'apprentissage sans /avec les symboles modèles . . . 152
4.2 Evolution de la taille des treillis (nombre de concepts) obtenus aux 5 tirages (v0 à v4) en fonction de la taille de l'ensemble d'apprentissage avec / sans les symboles modèles . . . 153
4.3 Evolution des taux de reconnaissance (en %) obtenus aux 5 tirages (v0 à v4) en fonction de la taille de l'ensemble d'apprentissage avec / sans analyse discriminante . . . 155
4.4 Evolution de la taille des treillis (nombre de concepts) obtenus aux 5 tirages (v0 à v4) en fonction de la taille de l'ensemble d'apprentissage avec / sans analyse discriminante . . . 155
4.5 Comparaison des taux de reconnaissance et taille des treillis entre reconnaissance simple et itérative . . . 157
4.6 Signatures des 10 objets (à gauche) et table discrétisée selon un critère d'entropie (à droite) . . . 163
4.7 Exemple de table discrétisée . . . 166
4.8 Reconnaissance des différents graphes . . . 167
4.9 Propriétés des méthodes de classification basées sur les treillis de Galois (S=Supervisé, NS=Non Supervisé) . . . 173
4.10 Propriétés des méthodes de classification basées sur les treillis de Galois (suite) . . . 174
4.11 Bases de données du ML Repository . . . 175
4.12 Nombre d'attributs et nombre de concepts utilisés en moyenne par NAVIGALA sur chaque base . . . 175
4.13 Résultats obtenus en validation croisée sur quelques bases de données du ML Repository . . . 176
4.14 Exemple de table discrétisée . . . 182

Table des figures

1 Exemples de symboles 2

1.1 Exemple d'arbre de décision 27
1.2 Exemple d'hyperplan obtenu pour un problème à deux classes ω_1 et ω_2 . 33
1.3 Exemple de treillis 39
1.4 Exemple d'un treillis de Galois et de son diagramme de Hasse 44
1.5 Liens entre contextes, bases et treillis 46
1.6 Points communs / différences des méthodes 61

2.1 Symboles modèles de la base GREC 2003 (dégradations binaires) 73
2.2 Symboles bruités de la base GREC 2003 (dégradations binaires) 74
2.3 Symboles modèles de la base GREC 2003 (distorsions vectorielles) 74
2.4 Symboles bruités de la base GREC 2003 (distorsions vectorielles) 75
2.5 Quelques symboles modèles de la base GREC 2005 75
2.6 Symboles bruités de la base GREC 2005 75
2.7 Représentation graphique d'un accumulateur contenant des groupes de droites (entourés d'un cercle). Certains groupes sont scindés en deux. . . 79
2.8 Projection des pixels du voisinage le long de la droite (en bleu) 80
2.9 Distance de matching entre le symbole modèle (à gauche) et le symbole bruité avec les segments extraits (à droite). Plusieurs cas d'appariement sont représentés : (1) one-to-one : un segment modèle correspond à un segment extrait, (2) many-to-one : plusieurs segments modèles correspondent à un segment extrait, (3) one-to-many : un segment modèle correspond à plusieurs segments extraits, (4) fausse alarme : un segment extrait ne correspond à aucun segment modèle, (5) oubli : un segment modèle ne correspond à aucun segment. 81
2.10 Histogramme des pourcentages d'association et courbe de distance de matching en fonction des classes de symboles 82
2.11 Histogramme des pourcentages d'association et courbe de distance de matching en fonction des types de dégradation appliqués aux symboles . 82
2.12 Types de relation entre deux segments s et s' : V, X, Y, P et O (dans l'ordre de gauche à droite) 84

2.13 Segments extraits par transformée de Hough sur une image de symbole et le graphe de vecteurs associé . . . 85
2.14 Proportion d'apparition des types de relation X, Y, V, P et O . . . 89
2.15 Proportion d'apparition des types de relation X, Y, T, V, L, P et O . . . 89
2.16 Proportion d'apparition des angles . . . 90
2.17 Proportion d'apparition des longueurs relatives . . . 90

3.1 Description schématique de l'apprentissage . . . 95
3.2 Exemple de nombre flou . . . 103
3.3 Cas particuliers de nombres flous . . . 103
3.4 Exemple de nombre flou formé à partir d'un intervalle . . . 104
3.5 Diagramme de Hasse du treillis de Galois . . . 106
3.6 Description schématique de la classification . . . 109
3.7 Navigation dans le treillis de Galois . . . 110
3.8 Schéma de description du processus de reconnaissance itératif . . . 115
3.9 Zone d'ambiguïté d'un intervalle représenté par un nombre flou . . . 116
3.10 Taux de reconnaissance obtenus pour les différentes signatures . . . 117
3.11 Tailles des treillis obtenus pour les différentes signatures . . . 117
3.12 Taux de reconnaissance obtenus selon le critère de coupe (classes 1 à 10) 118
3.13 Taux de reconnaissance obtenus selon le critère de coupe (classes 11 à 20) 119
3.14 Taille du treillis selon le critère de coupe (classes 1 à 10) . . . 119
3.15 Taille du treillis selon le critère de coupe (classes 11 à 20) . . . 120
3.16 Légende . . . 120
3.17 Symboles de l'apprentissage décrits par les 50 attributs de la signature de Radon . . . 121
3.18 Symboles de l'apprentissage décrits par 7 attributs de la signature de Radon sélectionnés par le critère de Hotelling . . . 122
3.19 Légende . . . 128
3.20 Exemple de classification d'un symbole de la classe 3 : étape 1 . . . 128
3.21 Exemple de classification d'un symbole de la classe 3 : étape 2 . . . 129
3.22 Exemple de classification d'un symbole de la classe 3 : étape 3 . . . 130
3.23 Exemple de classification d'un symbole de la classe 3 : étape 4 . . . 131
3.24 Exemple de classification d'un symbole de la classe 3 : étape 5 . . . 132
3.25 Légende . . . 133
3.26 Exemple de classification d'un symbole de la classe 2 : étape 1 . . . 133
3.27 Exemple de classification d'un symbole de la classe 2 : étape 2 . . . 134
3.28 Exemple de classification d'un symbole de la classe 2 : étape 3 . . . 135
3.29 Exemple de classification d'un symbole de la classe 2 : étape 4 . . . 136
3.30 Exemple de classification d'un symbole de la classe 2 : étape 5 . . . 137
3.31 Exemple de classification d'un symbole de la classe 2 : étape 6 . . . 138
3.32 Légende . . . 139

3.33 Exemple d'erreur de classification d'un symbole de la classe 4 : étape 1 . 139
3.34 Exemple d'erreur de classification d'un symbole de la classe 4 : étape 2 . 140
3.35 Exemple d'erreur de classification d'un symbole de la classe 4 : étape 3 . 141
3.36 Exemple d'erreur de classification d'un symbole de la classe 4 : étape 4 . 142
3.37 Exemple d'erreur de classification d'un symbole de la classe 4 : étape 5 . 143
3.38 Exemple d'erreur de classification d'un symbole de la classe 4 : étape 6 . 144
3.39 Types d'erreurs de classification 145
3.40 Nombre d'occurrences des différents types d'erreurs 146
3.41 Nombre d'occurrences des tailles des intervalles ayant généré des erreurs . 147
3.42 Nombre d'occurrences des écarts entre les intervalles ayant généré des erreurs 147

4.1 Vue d'ensemble de l'interface et visualisation d'un treillis de Galois . . . 150
4.2 Evolution du taux de reconnaissance (en %) en fonction de la taille de l'ensemble d'apprentissage 151
4.3 Evolution de la taille du treillis de Galois (nombre de concepts) en fonction de la taille de l'ensemble d'apprentissage 151
4.4 Treillis de Galois 164
4.5 Arbre de décision 164
4.6 Inclusion de l'arbre de décision (en gras) dans le treillis de Galois 165
4.7 Reconnaissance avec les arbres de décision 1 et 2 166
4.8 Reconnaissance avec les arbres de décision 3 et 4 167
4.9 Reconnaissance avec les arbres de décision 5 et 6 168
4.10 Reconnaissance avec le treillis de Galois 169
4.11 Taux de reconnaissance obtenus par l'arbre de décision et le treillis de Galois sur plusieurs signatures pour les classes de symboles 1 à 10 (graphique de gauche) et 11 à 20 (graphique de droite) 170
4.12 Taux de reconnaissance obtenus en validation croisée pour les 3 classifieurs 171
4.13 Points communs / différences des méthodes 172
4.14 Reconnaissance locale vs globale dans le treillis de Galois 183

A.1 Menu principal (à gauche) et contextuel (à droite) 207
A.2 Arborescence contenant différentes expérimentations 208
A.3 Onglets de couleur de la fenêtre principale et visualisation d'une signature 209
A.4 Vue d'ensemble de l'interface et visualisation d'un treillis de Galois . . . 209

Introduction générale

Les travaux décrits dans ce manuscrit concernent, comme son titre l'indique, la reconnaissance d'objets graphiques détériorés, réalisée à l'aide d'un treillis de Galois. Ce sujet se compose de deux parties principales : la reconnaissance d'objets et l'utilisation d'un treillis de Galois qui caractérisent chacune un domaine de recherche à part entière. Pour préciser l'intérêt de ces travaux, il paraît important d'expliquer l'objectif et les applications liés à la reconnaissance d'images, mais également de motiver l'utilisation d'un treillis de Galois dans ce contexte.

Pourquoi reconnaître ?

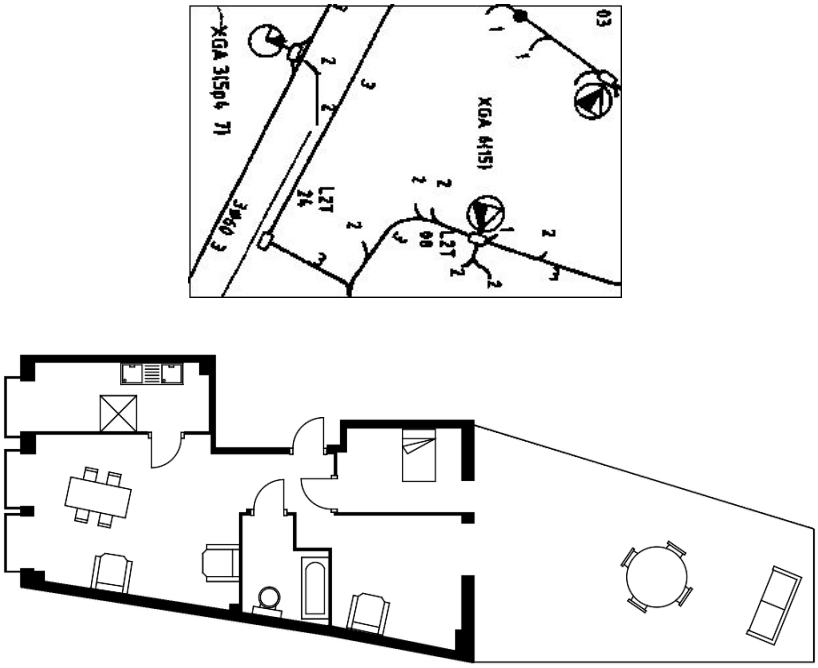
Avec l'essor de l'informatique et de l'Internet, une masse considérable de données est créée, échangée et conservée sur des supports numériques. Dans ce contexte de surabondance, il est nécessaire, pour faciliter la navigation au sein de ces données et les rendre exploitables, de les organiser en regroupant entre elles les données similaires. Cette indexation ne peut raisonnablement pas s'effectuer à la main et doit être automatisée. Pour les documents textuels, l'indexation est réalisée à l'aide de mots-clés ou directement à partir des mots du texte. Mais s'agissant des images, des graphiques, des schémas ou des photographies, l'opération se révèle plus complexe. Comment extraire des indices objectifs qui soient les mêmes pour toute personne ? Des mots-clés ne semblent pas offrir une solution satisfaisante car ils ne peuvent retranscrire l'intégralité des éléments d'une image, et sont très subjectifs d'une personne à l'autre. Seule une indexation par le contenu paraît réellement objective et exploitable, à condition que l'extraction des indices visuels ne soit pas trop coûteuse. Dans ce genre d'applications, la reconnaissance des images et plus généralement des objets graphiques trouve donc tout son intérêt.

Cependant, bien que l'on observe un engouement général pour les outils numériques, une grande quantité de documents papiers est toujours exploitée. Parmi ces ouvrages, on trouve en particulier, les documents du patrimoine, dont le contenu riche et va-

rié suscite l'intérêt grandissant du grand public. En vue d'une meilleure conservation, l'accès à certains de ces documents anciens est restreint. Des campagnes de numérisation ont déjà eu lieu et se poursuivent pour les rendre disponibles à un plus grand nombre. La détérioration du papier (trous, taches, plis, effacement de l'encre, ...) rend la reconnaissance des objets graphiques d'autant plus délicate.

Parmi tous ces objets graphiques, il existe un domaine plus particulier qui concerne la reconnaissance de symboles. Au regard de l'étymologie grecque de ce mot, il est amusant de parler de "reconnaissance de symboles". En effet, à l'origine le mot symbole est un signe de reconnaissance entre deux hôtes. Il s'agissait d'un objet séparé en deux parties, que conservait la famille de chaque hôte. En rapprochant les deux parties de l'objet, on pouvait ainsi prouver les liens sacrés d'hospitalité entre les deux familles. Cette référence étymologique rend ainsi cette problématique fort sympathique. En effet, il est certain que la reconnaissance des symboles facilitera l'usage et la navigation dans ces masses de données, réunissant ainsi comme dans l'antiquité, les personnes autour du partage de la connaissance.

FIG. 1 – Exemples de symboles



Pourquoi utiliser un treillis de Galois ?

Le treillis de Galois est issu d'une recherche théorique abondante et relativement ancienne. Après une première définition algébrique à partir des opérations de borne sup et borne inf [Bir67], il fut décrit de manière structurelle sous la forme d'un graphe [BM70]. Il faut savoir que, dans le pire des cas, la taille de ce graphe peut être exponentielle. C'est pourquoi, le développement d'applications le concernant, n'a été rendu possible que récemment, suite à la montée en puissance des ordinateurs. Dès lors, son utilisation n'a cessé de progresser, notamment dans les domaines de la fouille, de la classification et de la visualisation des données.

Sa structuration sous forme de graphe lui confère une grande lisibilité et une organisation des données très intuitive. Ses qualités ont séduit bon nombre de chercheurs qui mettent en œuvre des applications surprenantes. Par exemple, J. Ducrou et P. Eklund [DE05] ont mis au point un système pour rechercher les meilleurs spots de surf sur les côtes australiennes à partir des données météorologiques et des connaissances géographiques du terrain. De même, V. Duquenne [Duq07] a récemment employé le treillis de Galois pour aider les enseignants à adapter l'apprentissage des mathématiques en fonction des acquis des élèves. A notre connaissance, le treillis de Galois n'a jamais été utilisé en reconnaissance de symboles. Pourtant, il semble posséder des propriétés intéressantes pour son exploitation dans ce domaine. En effet, l'une de ses propriétés principales concerne l'existence dans le treillis de Galois de nombreux chemins de classification qui permettent d'aboutir à une même classe. Ces différents chemins offrent ainsi une grande souplesse dans le cadre d'une reconnaissance d'objets graphiques détériorés.

Le premier chapitre de ce manuscrit fournit un état de l'art consacré aux approches statistiques et structurelles de descripteurs de formes et aux classifieurs communément utilisés en reconnaissance d'images. Cet état de l'art comprend également une partie dédiée aux fondements mathématiques du treillis de Galois et à plusieurs applications le concernant. Le second chapitre contient une description des bases de données et des descripteurs de formes utilisés pour l'évaluation de notre système de reconnaissance. Le classifieur basé sur le treillis de Galois, principale contribution de ces travaux, est ensuite présenté dans le troisième chapitre. Enfin, différentes évaluations ainsi qu'une discussion proposant quelques pistes de recherche sont détaillées dans le dernier chapitre.

Chapitre 1

Etat de l'art

1.1 Introduction

Cet état de l'art comprend deux parties principales correspondant chacune à l'un des deux domaines étudiés et mis en application conjointement dans ce travail de thèse. Ces deux domaines sont la reconnaissance d'images (§ 1.2) d'une part, et le domaine des treillis et de l'analyse formelle des concepts (§ 1.3) d'autre part.

Nous débutons la partie reconnaissance d'images par la description des signatures caractérisant des objets graphiques, qui consiste à distinguer les approches statistiques (§ 1.2.1), des approches structurelles (§ 1.2.2). Ensuite, nous présentons différentes méthodes de classification utilisées classiquement en reconnaissance de formes (§ 1.2.4) : l'arbre de décision, les C-Moyennes floues, les k-plus proches voisins, le classifieur Bayésien, les réseaux de neurones et les machines à vecteur de support.

Dans un second temps, nous développons la partie concernant les treillis qui comprend la description des fondements mathématiques et de l'analyse formelle des concepts (§ 1.3.1), puis quelques aspects algorithmiques (§ 1.3.2). Enfin, nous présentons différents travaux réalisés en classification avec les treillis de Galois (§ 1.3.3) et les règles d'association (§ 1.3.4).

1.2 Reconnaissance d'images

Dans la littérature, plusieurs états de l'art décrivent la problématique de la reconnaissance d'images, et plus particulièrement de symboles issus de documents : [CP98, CV00, LVSM02, LVS03, TL03, VD04].

La *reconnaissance* ou *classification* consiste à attribuer un même label (une même étiquette) à un ensemble d'objets identiques. Pour cela, il est souvent nécessaire de passer par une phase d'*apprentissage* qui permet au classifieur, c'est-à-dire au système chargé de classer les objets, de connaître et d'apprendre les descriptions d'un ensemble d'objets. Par la suite, lors de la *classification*, il devra être en mesure de classer correctement de nouveaux objets inconnus, mais qui ressemblent à ceux de l'ensemble d'apprentissage.

En reconnaissance d'images, pour décrire les objets graphiques, nous utilisons une *signature*. Il s'agit d'une information condensée de l'objet qui permet de le décrire mais aussi surtout de le caractériser. Cette information peut être de nature différente : numérique (l'objet se compose de 4 segments), symbolique (l'objet est bleu et jaune), elle peut décrire la forme dans sa globalité (l'objet est un triangle) ou de manière plus fine (l'objet comporte 34 pixels noirs), ...

Il existe un grand nombre de signatures dans la littérature. Elles représentent un caractère fondamental dans un processus de reconnaissance de formes [BSA91]. Un bon état de l'art de ces signatures se trouve dans les références [LVSM01, Ada01, Zuw06, TWT03, TW03a].

Nous allons présenter ces signatures suivant deux types d'approches classiquement utilisées en reconnaissance de symboles : les approches *statistiques* et les approches *structurelles*. Les approches statistiques sont obtenues par comptage d'une mesure considérée comme caractéristique de la forme. En revanche, les approches structurelles décrivent un type de structure caractéristique de la forme. Plus précisément, elles décrivent les relations spatiales ou topologiques entre les *primitives élémentaires*. Ainsi, il semble approprié de calculer une signature structurelle sur des images de symboles car ces derniers comportent une certaine organisation : leur structure est caractéristique.

En reconnaissance d'objets graphiques, les signatures décrivant les symboles se doivent, selon le contexte, de respecter certaines contraintes, telles que par exemple l'invariance à la translation, au changement d'échelle ou à l'orientation. De plus, les signatures doivent être le plus possible robustes au bruit.

Les valeurs de la signature seront par la suite nommées *attributs*, *primitives*, *descripteurs* ou *caractéristiques*. La description de certaines approches statistiques est donnée dans la partie suivante, et celle d'approches structurelles dans la partie 1.2.2.

1.2.1 Approches statistiques

Une signature *statistique* donne une description de l'objet graphique en fonction de son contenu pixellaire. Elle peut également caractériser des propriétés fréquentielles. Dans tous les cas, nous obtenons un vecteur numérique, autrement dit un ensemble de valeurs numériques calculées suivant certaines propriétés.

Parmi les références sur ce type de signature, il existe l'étude de Trier, Jain et Taxt [TJT96] qui décrit plusieurs méthodes d'extraction de primitives sur des caractères segmentés. Différentes représentations des caractères sont testées : caractères binaires ou en niveau de gris, approche contours ou squelettes. Les propriétés d'invariance des méthodes, les distorsions et variabilités des caractères attendues, et la complétude (reconstruction des caractères à partir des descripteurs) sont discutées. Les auteurs insistent sur l'importance d'avoir un ensemble de descripteurs adapté à l'application visée, et sur le fait que le calcul de descripteurs est un maillon essentiel à une bonne reconnaissance. Le livre de Marcus Rodrigues [Rod00] fait également référence. Il regroupe des articles sur de nombreuses signatures statistiques : projection des invariants, invariants à la convolution et à la rotation, invariants de Fourier-Mellin, descripteurs robustes au bruit, descripteurs d'images couleur, ...

Les méthodes de reconnaissance basées sur l'utilisation de signatures statistiques offrent des résultats de classification acceptables lorsqu'elles s'appuient sur des jeux d'apprentissage significatifs. En revanche, elles traitent une information bas niveau et nécessitent souvent d'être appliquées sur des images de symboles segmentés.

Dans la suite, nous décrivons plus particulièrement le principe du calcul des signatures basées sur la transformée de Fourier-Mellin, la transformée de Radon, les moments de Zernike et pour finir sur les ondelettes. Ces signatures sont souvent utilisées dans la littérature en reconnaissance de formes où elles ont fait leurs preuves [CV00, LVS03, VD04, Ada01] et elles respectent les contraintes d'invariance à la translation, rotation et au changement d'échelle.

Intuitivement, l'invariance à la translation s'obtient en positionnant l'origine du repère pour le calcul de la signature au centre de gravité de la forme. L'invariance à la rotation peut être obtenue, soit en se projetant dans un espace où il y a possibilité de

travailler sur l'invariance (domaine polaire), soit en travaillant directement dans l'espace cartésien sous forme de projection circulaire, le cercle étant par définition invariant à la rotation. Enfin, l'invariance au changement d'échelle est souvent obtenue par normalisation par un facteur "général" permettant de rendre la mesure indépendante de la taille ou du poids de la forme dans l'image. Ce facteur peut être le nombre de pixels de la forme, le rayon de giration de l'ellipse englobante, ...

Signature de Fourier-Mellin

Présentation La transformée de Fourier-Mellin est une méthode utilisée principalement dans la littérature en reconnaissance de formes, reconstruction d'images et également pour l'estimation du mouvement. En pratique, les invariants de Fourier-Mellin sont calculés à partir de filtres appliqués sur l'image du symbole.

Description mathématique La *transformée de Fourier-Mellin* [DDG99, AOC+00] est définie pour une image décrite par une fonction $f(r,\theta)$ réelle et positive en coordonnées polaires de la manière suivante :

$$M_{pq} = \int_{r=0}^{\infty} \int_{\theta=0}^{2\Pi} r^{p-1} e^{-iq\theta} f(r,\theta) dr d\theta \qquad \text{avec } p = \sigma + iv \in \mathbb{C} \text{ et } q \in \mathbb{Z}$$

Les invariants de Fourier-Mellin sont alors donnés par la formule :

$$I_{pq} = M_{pq} [M_{0\sigma}]^{-p/\sigma} [M_{1\sigma}]^{-q} |M_{1\sigma}|^q$$

Cette signature est paramétrable suivant deux termes p et q relatifs à l'invariance à l'orientation et à l'échelle : p est un coefficient radial et q un coefficient circulaire. Les moments de Fourier-Mellin sont invariants à la translation, à la rotation et au changement d'échelle [DDG99].

Exemples d'application Les applications sont nombreuses et majoritairement sur des images en niveaux de gris. On peut citer les travaux de R. Milanese et al. (1998) [MCP98] qui ont travaillé sur la recherche d'images décrites par la transformée de Fourier-Mellin, tout comme S. Derrode, M. Daoudi et F. Ghorbel dans leurs travaux en 1999 [DDG99], ou encore S. Adam et al. en 2001 [AOC+01]. En 2002, C. Kan et M.D. Srinath [KS02] ont utilisé les moments orthogonaux de Fourier-Mellin associés aux moments de Zernike pour la reconnaissance de caractères de différentes tailles. Enfin en 2004, A.T.B. Jin et al. ont développé un système de vérification des empreintes digitales [JLS04].

Signature de Radon

Présentation C'est en 1917 que J. Radon [Rad17] a créé la transformée qui porte son nom. Cette transformée donne une représentation de la forme par un ensemble de projections suivant différentes orientations.

Description mathématique La *transformée de Radon* [Rad17, TW03b] d'une image est déterminée par un ensemble de projections de l'image le long de droites suivant différentes orientations. Soit $f(x, y)$ une image, la transformée de Radon est alors définie par :

$$T_{R_f}(\rho, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y)\delta(xcos(\theta) + ysin(\theta) - \rho)dxdy$$

avec δ la fonction Dirac telle que $\delta(x) = 1$ si $x = 0$ et 0 sinon, $\theta \in [0, \Pi[$ et $\rho \in [-\infty, \infty[$. Autrement dit, la T_{R_f} est l'intégrale de f le long de la droite $L(\rho, \theta)$ définie par $\rho = xcos(\theta) + ysin(\theta)$. A partir de cette formule, on calcule la R-signature comme suit :

$$R_f(\theta) = \int_{-\infty}^{\infty} (T_{R_f}(\rho, \theta))^2 d\rho$$

Cette signature est invariante à la translation, aux changements d'échelle et si on inclut les permutations dans son calcul de distance, à la rotation [TW03b]. Pour obtenir l'invariance à la rotation, il est nécessaire d'intégrer dans la mesure de distance entre deux signatures, la comparaison avec les différentes permutations possibles des valeurs de la signature. Autrement dit, il s'agit de réaliser une permutation cyclique de la signature. Une autre possibilité pour obtenir l'invariance à la rotation, consiste à calculer la transformée de Fourier de la signature.

Exemples d'application Dans la littérature, on trouve de nombreuses applications en reconnaissance de formes et en reconstruction d'images médicales. En 1992, VF Leavers [Lea92] a appliqué la transformée de Radon pour décrire les propriétés géométriques et les relations spatiales au sein de la forme. L'objectif de cette description était la reconnaissance d'objets. E. Magli, G. Olmo et L. Lo Presti (1998) se sont servis de la transformée de Radon associée à la transformée continue en ondelettes pour détecter le sillage en forme de "V" des bateaux sur des images aériennes maritimes [MOLP99]. En 2000, J. Waldemark, M. Millberg, T. Lindblad et K. Waldemark l'ont appliquée dans le but de reconnaître des missiles [WMLW00]. A. Kadyrov et M. Petrou(2003) [KP03] ont généralisé la transformée de Radon en créant une transformée nommée "Trace" pour construire une signature invariante d'un objet. Elle a également été appliquée en 2003 par J.S. Seo, J. Haitsma, T. Kalker et C.D. Yoo pour obtenir des images robustes d'empreintes digitales [SHKY04]. Enfin en 2003, S. Tabbone et L. Wendling [TW03b] ont

étendu l'utilisation de cette transformée aux images en niveaux de gris et en couleur dans le cadre d'un travail sur la reconnaissance de formes.

Signature de Zernike

Présentation Les moments invariants permettent de décrire le contenu d'une image en le projetant dans un espace, l'espace des polynômes de Zernike. Il est possible de classer les moments invariants en deux catégories : les moments non-orthogonaux et les orthogonaux. Parmi les moments orthogonaux, on trouve les moments de Legendre et de Zernike (1934).

Description mathématique Les *polynômes de Zernike* ont été introduits par Teague [Tea03] en analyse d'images. Ils forment un ensemble orthogonal complet intérieur au cercle unité $x^2 + y^2 = 1$. La fonction de Zernike dépend de l'ordre (p, q) et est définie selon le système de coordonnées polaires (r, θ) tel que :

$$W_{pq}(r, \theta) = R_{pq}(r)e^{iq\theta}$$

où

$$R_{p,\pm q}(r) = \sum_{k=q, p-|k| pair}^{p} B_{pqk} r^k$$

$$B_{pqk} = \frac{(-1)^{(p-k)/2}((p+k)/2)!}{((p-k)/2)!((q+k)/2)!((k-q)/2)!}$$

Les moments de Zernike d'une image sont les projections de la fonction image suivant ces fonctions à base orthogonale. Le moment de Zernike d'ordre p avec répétition de q pour une image est défini par :

$$Z_{pq} = \frac{p+1}{\Pi} \sum_x \sum_y f(x, y) W^{\star}_{pq}(r, \theta) \Delta_x \Delta_y$$

où r est la longueur du vecteur partant de l'origine et allant au pixel (x, y), θ est l'angle entre ce vecteur et l'axe des abscisses dans le sens inverse des aiguilles d'une montre, $x^2 + y^2 \leq 1$, $x = rcos(\theta)$ et $y = rsin(\theta)$.

Exemples d'application Dans la littérature, les moments de Zernike ont souvent été utilisés pour faire de la reconnaissance de formes. On peut citer les travaux de P.C. Hew (1997) [Hew97] qui traitent de la reconnaissance de chiffres imprimés ayant subi des rotations bruitées, ou ceux de Novotni et Klein [NK04] sur la reconnaissance

de formes en 3D. Leur popularité tient en majeure partie au fait qu'ils sont robustes au bruit. Une autre application des moments concerne la reconstruction d'images. De nombreuses recherches sont encore menées sur les moments de Zernike pour améliorer la rapidité de leur calcul par des approximations [PBKM07], ou encore pour mieux expliciter leurs propriétés d'invariance [BHO07]. Ils ont été dernièrement combinés aux ondelettes dans le cadre de la reconnaissance de caractères [BS07].

Signature à base d'ondelettes

Présentation La théorie des ondelettes a été développée dès les années 1900. Etudiées en mathématiques, physique quantique, statistiques et plus récemment en traitement du signal (1980), les ondelettes sont basées sur une extension de l'idée du filtrage passe-haut (qui permet de filtrer les basses fréquences pour ne laisser passer que les hautes fréquences). Un contraste important dans le domaine spatial (ciel clair au-dessus du sol sombre) s'exprime par des valeurs élevées dans le domaine fréquentiel. L'information fréquentielle peut être extraite par transformée de Fourier ; cependant, avec l'utilisation de cette transformée, il manque l'association avec le domaine spatial. L'intérêt des ondelettes réside dans le fait qu'elles sont basées sur des fonctions localisées à la fois dans le domaine spatial et fréquentiel. Ces fonctions représentent des signaux dans différentes bandes de fréquence, chacune suivant une résolution adaptée à l'échelle.

Description mathématique Les ondelettes peuvent être décrites selon deux groupes : les *fonctions d'ondelette* appelées *ondelettes mères* sont notées $\psi(x)$ et les *fonctions d'échelle* appelées *ondelettes pères* sont notées $\phi(x)$. On se réfère à ces deux groupes par le terme de famille d'ondelettes. Les transformations appliquées aux ondelettes mères et pères sont respectivement appelées ondelettes filles et fils. La première ondelette, qui est également la plus simple, a été développée par Alfred Haar en 1909. Elle appartient au groupe d'ondelettes connu sous le nom d'ondelettes de Daubechies, dont les fonctions d'échelle vérifient des propriétés intéressantes.

$$\phi_{Haar}(x) = \begin{cases} 1 & \text{si } 0 \leq x < 1 \\ 0 & \text{sinon} \end{cases} \qquad \psi_{Haar}(x) = \begin{cases} 1 & \text{si } 0 \leq x < 1/2 \\ -1 & \text{sinon si } 1/2 \leq x < 1 \\ 0 & \text{sinon} \end{cases}$$

Une famille d'ondelettes est généralement décrite par sa *fonction d'ondelette* ou *ondelette mère*. Elle doit vérifier certaines propriétés pour que la transformation en ondelettes soit inversible de manière stable :

$$\int |\psi(x)|^2 dx = 1 \quad (1) \qquad \int |\psi(x)| dx < \infty \quad (2) \qquad \int \psi(x) dx = 0 \quad (3)$$

La fonction doit être un élément de $L^2(\mathbb{R})$ et avoir une énergie normalisée, elle doit être un élément de $L^1(\mathbb{R})$ et avoir une moyenne nulle, enfin elle doit permettre l'addition des coefficients en ondelettes sans changer le flux total du signal. D'autres propriétés peuvent être spécifiques aux applications. Par exemple, il peut être nécessaire que la fonction d'ondelette soit continue, ou que le support soit compact sur un intervalle spécifique.

Un facteur de normalisation est généralement appliqué à la fonction d'ondelette afin que l'ondelette fille hérite de toutes les propriétés de sa mère. Une ondelette fille est définie par :

$$\psi_{a,b}(x) = a^{-1/2}\psi((x-b)/a) \qquad \text{avec } a, b \in \mathbb{R} \text{ et } a \neq 0$$

où a est le facteur d'échelle ou facteur de dilatation, et b est le facteur de translation. Le choix généralement fait est le suivant : $a = 2^{-j}$ et $b = 2^{-j}k$ avec j et k des entiers. L'ondelette fille devient ainsi :

$$\psi_{j,k}(x) = 2^{-1/2}\psi(2^j x - k)$$

Dans les transformées en ondelettes discrètes, une *fonction d'échelle* ou *ondelette père* est nécessaire pour couvrir les basses fréquences. Il est utile de spécifier que l'ondelette père doit avoir une moyenne égale à 1 :

$$\int \phi(x) dx = 1$$

Les fonctions d'ondelettes de Haar étant discontinues, il n'est pas souhaitable de les utiliser pour l'analyse de fonctions aux dérivées continues. Elles ne sont donc pas préconisées pour le traitement des images. Les *ondelettes de Daubechies* [Dau92] font partie des nombreux types d'ondelettes développés pour améliorer la transformée en ondelette de Haar. Pour chaque entier r, la base orthonormée [Mal89] pour $L^2(\mathbb{R})$ est définie par :

$$\psi_{r,j,k}(x) = 2^{j/2}\psi_r(2^j x - k) \qquad j,k \in \mathbb{Z}$$

où la fonction $\psi_r(x)$ dans $L^2(\mathbb{R})$ est telle que $\{\psi_r(x-k)|k \in \mathbb{Z}\}$ est une suite orthonormée dans $L^2(\mathbb{R})$. Ici, j est le facteur d'échelle, k est le facteur de translation et r le facteur du filtre.

En reconnaissance de formes, on trouve différentes méthodes de calcul des invariants basés sur les ondelettes. L'approche de Shen et Ip [SI99] présente une formule générale pour obtenir n'importe quels moments invariants à la rotation et ils en déduisent sous certaines contraintes une formulation des moments à base d'ondelettes. L'approche de Chen et al. [CBK03] combine l'utilisation de la transformée de Fourier et de la transformée en ondelettes.

Exemples d'application Les nombreuses applications concernent la reconnaissance de caractères ou de symboles [CBK03, TV06, RM07], la reconnaissance de visages [KP05], la compression et le débruitage d'images [BV07], la fusion d'images [AZD07]. Les ondelettes ont été combinées avec succès aux réseaux de neurones dans un objectif de classification [CBK03, ATP05].

Conclusion

Comme en attestent les références citées précédemment dans les différents exemples d'application, les signatures statistiques utilisées pour la description des symboles vérifient toutes une bonne robustesse au bruit et les propriétés d'invariance nécessaires au contexte de reconnaissance d'objets graphiques détériorés issus de documents (voir Tab. 1.1). De plus, elles proposent un large panel de descripteurs statistiques et offrent ainsi une information de caractérisation des symboles relativement complémentaire.

1.2.2 Approches structurelles

Par définition, une signature *structurelle* vise à représenter la "structure" de la forme. Souvent, cette signature repose sur une décomposition de la forme en primitives élémentaires et sur une description des relations entre ces primitives. On appelle primitive élémentaire ou primitive graphique tout regroupement de pixels de la forme : lignes, arcs de cercle, régions, vecteurs, ... Techniquement, une signature structurelle peut

TAB. 1.1 – Synthèse des propriétés des signatures statistiques étudiées

Signatures statistiques	Représentation	Invariance et propriétés	Rapidité du traitement	Implémentation simple	Pouvoir de discrimination
Invariants de Fourier-Mellin	Domaine fréquentiel	Translation Rotation Changement d'échelle Gestion des formes connectées	+	+	-
R-signature (Radon)	Domaine spatial	Translation Rotation Changement d'échelle	+	+	+
Moments de Zernike	Domaine spatial	Translation Rotation Changement d'échelle	++	++	+
Moments à base d'ondelettes	Domaines spatial et fréquentiel	Translation Rotation Changement d'échelle	+	+	++

contenir, soit un ensemble de primitives, soit un ensemble de relations topologiques (jonctions, parallélisme, ...), ou bien encore les deux à la fois. Dans ce dernier cas, la signature structurelle est souvent représentée par un graphe relationnel où les nœuds du graphe sont, par exemple, les primitives graphiques, et les arcs du graphe sont les relations topologiques entre ces regroupements. Selon les éléments intégrés dans la signature structurelle, la description de l'objet graphique obtenue peut être plus ou moins élaborée. Outre les graphes, il existe aussi des approches syntaxiques basées sur l'utilisation de grammaires. Une grammaire est constituée d'un alphabet d'éléments terminaux (ensemble d'éléments de base), d'un alphabet d'éléments non-terminaux (ensemble d'éléments plus complexes composés d'éléments de base), et d'un ensemble de règles de grammaires. Ces dernières modélisent la construction des éléments complexes de manière hiérarchique, à partir des éléments de base. Plusieurs types de grammaire sont utilisés en reconnaissance de formes : grammaire de chaînes, grammaire bidimensionnelle, grammaire stochastique, ...

Les méthodes de reconnaissance basées sur l'utilisation de signatures structurelles intègrent donc une information haut niveau et peuvent ainsi être appliquées sur des images de symboles en contexte, c'est-à-dire sur des symboles non segmentés. Elles possèdent également l'avantage d'être invariantes à l'échelle et à la rotation. Cependant, elles sont assez sensibles au bruit et aux déformations et nécessitent souvent l'intégration de connaissances a priori sur les symboles traités. Leur limite est essentiellement liée au caractère NP-complet des isomorphismes qui implique souvent d'introduire des heuristiques pour élaguer les graphes.

Dans son étude bibliographique, M. Delalandre [Del05] présente une organisation des méthodes d'extraction des primitives graphiques en sept catégories. En collaboration avec S. Bernard et M. Delalandre [Ber06], nous avons agrémenté cette classification d'une huitième catégorie :

Détection de contours. Cette catégorie de méthodes a pour but la détection des points de contours (opérateurs morphologiques, suivi de contours ou encore décomposition en plages) puis le chaînage de ces points de contours (codage de Freeman [Fre61]).

Squelettisation. Il s'agit d'obtenir le squelette, c'est-à-dire les axes médians d'une forme par amincissements successifs des traits, ou encore par transformée en distance.

Parcours de forme. L'objectif est également d'obtenir les axes médians de la forme par des techniques de suivi de traits, ou de détection de jonction.

Décomposition en plages. Le principe est de décomposer une forme en trois catégories de bandes verticales (ou horizontales) : plage "extrémité", plage "ligne" ou plage "jonction" afin de construire un graphe représentant l'organisation de ces plages au sein de la forme.

Segmentation en régions. Il s'agit d'identifier et d'étiqueter chaque pixel d'une forme comme appartenant à une région "extrémité", "ligne" ou "jonction" (calcul de "distances d'orientation" dans toutes les directions pour former une courbe dont les pics permettent d'identifier le type de région).

Sous-échantillonnage. Cette catégorie de méthodes consiste en l'application d'un maillage (sous-échantillonnage) sur la forme, puis en la recherche de correspondance entre chaque maille de la forme et des modèles de mailles référencées dans une bibliothèque. A partir des mailles reconnues, il est possible de déterminer la structure des primitives graphiques et leurs relations topologiques.

Graphes de composantes. Il s'agit de segmenter les pixels de la forme en composantes connexes (ensemble de pixels interconnectés) pour obtenir les primitives graphiques correspondantes à ces composantes connexes et leurs relations topologiques (propagation, balayage de lignes, suivi de contours).

Transformée de Hough/Radon. [Hou62, Rad17] L'objectif de ces méthodes est de détecter dans une image un ensemble de formes géométriques simples connaissant leurs équations paramétriques, comme par exemple des droites, des cercles, des ellipses, ...

Dans le tableau synthétique 1.2, nous présentons les différentes caractéristiques de ces huit catégories de méthodes. Tout d'abord, les différentes méthodes offrent des qualités de représentation qui vont du simple ensemble de pixels à des représentations plus riches sémantiquement tels que les vecteurs. Elles peuvent comporter des informations sur les jonctions ou les axes médians des formes et proposent des propriétés de robustesse et d'invariance plus ou moins intéressantes.

TAB. 1.2 – Synthèse des propriétés des méthodes d'extraction de primitives graphiques

Méthodes	Qualité de la représentation	Robustesse	Invariance
Détection de contours	+ Représentation exacte - Jonctions, axes médians	++	++
Squelettisation	+ Axes médians - Distorsions de jonctions	-	+
Parcours de formes	+ Primitives vecteurs + Jonctions	+	-
Décomposition en plages	+ Représentation riche + Jonctions, axes médians	-	-
Segmentations en régions	+ Jonctions - Sémantiquement faible	++	++
Sous-échantillonnage	+ Primitives graphiques de types variés + Sémantiquement riche	-	-
Graphe de composantes	+ Primitives composantes connexes - Jonctions, axes médians	++	++
Transformée de Hough/Radon	+ Primitives de haut niveau + Jonctions, axes médians	++	+

Comme le précise S. Bernard [Ber06], la plupart des catégories de méthodes font l'objet d'un compromis entre qualité de représentation et propriétés de robustesse et d'invariance. Il semble ainsi que la méthode basée sur les transformées de Hough/Radon, offre le meilleur compromis, de par sa représentation sémantiquement riche, et ses performances en terme de robustesse et d'invariance. Grâce à son caractère global, cette méthode semble en effet pouvoir absorber différents types de bruit. Les capacités de reconnaissance d'un système dépendent, en majeure partie, de la représentation structurelle choisie. C'est pourquoi, il est essentiel, d'extraire des primitives graphiques de

bonne qualité, qui constitueront les briques élémentaires de la structure finale représentant le symbole.

Dans cet état de l'art, les différentes signatures structurelles présentées vont être ordonnées selon trois catégories :

- les signatures structurelles sous forme de liste de valeurs numériques
- les signatures composées de modèles de vecteurs, c'est-à-dire composés d'ensembles de vecteurs ou d'agencements simples de vecteurs (tels que des jonctions particulières, des formes rectangulaires, triangulaires, ...)
- les signatures sous forme de graphe

A - Signatures structurelles sous la forme d'une liste de valeurs numériques

Dans ses travaux [Hua97], Huang utilise un code binaire pour représenter les objets ainsi que leur agencement au sein des images. Ce code, appelé "2D C^+-string", est un vecteur numérique contenant une suite de 0 et de 1. L'auteur dénombre 13 relations spatiales entre 2 objets et 9 types de similarité entre 2 paires d'objets qui lui permettent de caractériser les objets des images. L'objectif est d'effectuer l'indexation des images de grandes bases de données ou de rechercher une image dans la base. Il est possible à partir des données contenues dans la signature, de reconstruire les objets d'une image.

Dans leurs travaux [GW96, GSW02], Geibel et Wysotzki proposent une signature structurelle sous la forme d'une liste de valeurs numériques, obtenue à partir de graphes relationnels. En effet, le principe de leur méthode est de construire un graphe relationnel par symbole à décrire. Les nœuds du graphe représentent les primitives graphiques et les arcs sont les relations topologiques entre ces primitives graphiques. Pour chaque graphe relationnel, les auteurs calculent la matrice d'adjacence correspondante, qu'ils élèvent au carré, au cube, ou plus, ce qui leur permet d'obtenir tous les chemins de longueur 2, 3 ou plus du graphe relationnel correspondant. Ces chemins de longueurs n représentent des modèles d'agencement des primitives graphiques. Chaque symbole est alors représenté par une signature sous forme d'un vecteur numérique. Dans cette signature, la i-ème valeur correspond au nombre de fois où le modèle d'agencement des primitives graphiques i est présent dans le symbole. Pour la reconnaissance, les auteurs utilisent par la suite un arbre de décision. Ainsi, ils valident au fur et à mesure de la progression dans l'arbre, la présence ou l'absence des différents modèles d'agencement des primitives graphiques, ce qui leur permet de déterminer une classe pour le symbole à reconnaître. Cette signature a l'avantage d'allier une représentation haut niveau, adaptable à la reconnaissance en contexte, et une facilité d'utilisation conjointe avec une signature statistique. De plus, les modèles de vecteurs ne sont pas fixés à l'avance

comme c'est le cas d'ordinaire pour ce type de signature, mais sont calculés à partir des symboles. En plus, il est possible de choisir le degré de précision des modèles de vecteurs en choisissant des chemins plus ou moins longs.

L'approche de Dosch et Lladós [DL04] consiste à décrire les relations spatiales entre les paires de segments voisins qui constituent le symbole. Chaque paire est décrite par une relation spatiale et un facteur de qualité traduisant la pertinence de la relation. Les 5 relations considérées sont les suivantes : parallélisme avec ou sans chevauchement, colinéarité, jonction en L et jonction en V. Pour les relations spatiales comme pour le facteur de qualité, des seuils doivent être fixés afin de sélectionner seulement les relations pertinentes. La signature est alors obtenue en comptant le nombre d'occurences de chacune des 5 relations spatiales considérées dans le symbole. Les auteurs obtiennent de bons résultats en contexte, c'est-à-dire sur des symboles non segmentés (par exemple sur des plans architecturaux). Cependant, les résultats dépendent beaucoup de la vectorisation qui doit faire face à certaines difficultés (problème des petites dimensions et des arcs de cercles) et entraine un nombre important de fausses détections.

Dans [RnL05], Rusinol et Lladós ont basé leur approche sur la méthode de Dosch et Lladós. L'image du symbole est vectorisée, puis, le graphe attribué représentant le symbole est construit. Les nœuds du graphe sont les segments et les arcs sont les relations structurelles. Il est alors possible de calculer la matrice d'adjacence correspondante au graphe. On examine l'ensemble des sous-matrices contenant 2, 3 ou 4 nœuds. Les auteurs considèrent qu'une seule ligne de la matrice permet de caractériser la forme codée par la matrice. La ligne est mise sous la forme d'un mot. La signature est alors composée du nombre d'occurences de chacune des formes codées par les sous-matrices, c'est-à-dire du nombre d'occurences de chaque mot. Un dictionnaire est nécessaire pour mémoriser les équivalences entre les mots, car plusieurs mots peuvent représenter la même forme. Les tests ont été réalisés sur les symboles segmentés de la base GREC2003 [GRE03] (données synthétisées), sur les symboles de plans architecturaux en contexte (données réelles) et également sur des symboles dessinés à la main. Comme dans [DL04], les résultats obtenus en reconnaissance sont bons et les auteurs rencontrent les mêmes difficultés. Le passage aux données réelles entraine notamment un grand nombre de fausses détections. Cette approche suit également le même principe que celui de l'approche de Geibel et Wysotzki énoncée précédemment : dans leur cas, ils utilisaient les chemins de longueur n du graphe relationnel, alors qu'ici, les auteurs passent par les sous-matrices de la matrice d'adjacence.

Fonseca et al. [FFJ05] proposent une méthode de classification de formes génériques décrites par 2 types de descripteurs. L'objectif est d'estimer la ressemblance entre une forme donnée et un ensemble restreint de formes simples. Ils utilisent des descripteurs

topologiques (caractéristique globale du dessin : inclusion, adjacence) et géométriques (caractéristique locale : ratios des aires, ratios des périmètres de certains polygones). Les auteurs ont comparé cette méthode à 5 autres méthodes considérées comme les plus performantes en description de formes dont les moments de Zernike, les descripteurs de Fourier, la triangulation de Delaunay, ... Ils obtiennent les meilleurs résultats en rappel-précision[1] excepté lorsque l'objet à reconnaître et d'autres objets de la base sont très ressemblants (rappel inférieur ou égal à 20%). Dans ce cas seulement, les moments de Zernike sont plus perfomants.

Les travaux de Surapong et al. [SHG+05] présentent une signature structurelle calculée à partir de l'organisation spatiale de différentes couches de segmentation, dans le but de caractériser des images de lettrines. En effet, il s'agit de réaliser la segmentation d'une même image à différents seuils pour obtenir des couches d'informations distinctes. Les auteurs obtiennent alors l'organisation spatiale des informations contenues dans chaque couche en calculant la longueur de l'arbre recouvrant minimum (Minimum Spanning Tree), ainsi que les relations (angle et distance) entre les deux axes principaux d'inertie.

Le tableau 1.3 propose un récapitulatif des caractéristiques des signatures structurelles sous la forme d'une liste de valeurs numériques qui viennent d'être présentées.

B - Signatures structurelles composées de modèles de vecteurs

Dans leurs travaux [PP00], Parker et Pivovarov proposent une méthode de reconnaissance basée sur les modèles de vecteurs. A partir des symboles de l'ensemble d'apprentissage, ils construisent des modèles de vecteurs par squelettisation. Le squelette est débarrassé de ses petites imperfections pour obtenir un ensemble de segments d'un pixel d'épaisseur. Pour obtenir un modèle comportant la même épaisseur de trait que le symbole d'origine, les auteurs approximent alors l'épaisseur des traits et ajoutent sur le squelette des rectangles dont la largeur correspond à cette épaisseur estimée. Les rectangles sont remplis et forment ainsi le modèle du symbole. Pour gérer l'invariance à la rotation, les auteurs proposent de déterminer l'orientation de l'axe principal du symbole par le calcul de moments ou de la matrice de covariance, afin de tourner le modèle selon la bonne orientation avant d'effectuer l'appariement. De la même manière, la gestion de l'invariance à l'échelle est réalisée en évaluant l'échelle du symbole au moyen

[1]Rappel : rapport du nombre d'objets pertinents trouvés au nombre total d'objets pertinents. Précision : rapport du nombre d'objets pertinents trouvés au nombre total d'objets sélectionnés. Soient S l'ensemble des objets qu'un processus considère comme vérifiant la propriété recherchée, et V l'ensemble des objets qui vérifient effectivement cette propriété, alors $R = |S \bigcap V|/|V|$ et $P = |S \bigcap V|/|S|$

TAB. 1.3 – Comparaison des signatures structurelles sous la forme d'une liste de valeurs numériques

Méthodes	Construction d'un graphe	Recherche de formes simples	Construction de la signature	Taille de la signature
[Hua97]		Par paire d'éléments	Présence / absence d'agencements	Dépend des données
[GW96, GSW02]	Graphe de vecteurs		Comptage de chemins dans le graphe	Dépend des données
[DL04]		Par paire d'éléments	Comptage d'agencements	5 valeurs
[RnL05]	Graphe de vecteurs		Comptage des chemins (utilisation d'un dictionnaire)	Dépend des données
[FFJ05]	Graphe de régions		Ratios d'aires, de périmètres des polygones ; Valeurs propres de la matrice d'adjacence	Descripteur topologique : dépend des données ; Descripteur géométrique : 11 valeurs
[SHG+05]	Graphe de régions (MST)		Poids de l'arbre recouvrant minimum	3 valeurs par couche de segmentation

d'une boîte englobante et en recalant le modèle à cette échelle. Les résultats obtenus sur des symboles électroniques sont très encourageants, étant donné que la méthode a remporté le concours de reconnaissance de symboles à la conférence ICPR 2000 (International Conference on Pattern Recognition), avec seulement un modèle généré à partir d'un symbole de l'ensemble d'apprentissage. La méthode a également été testée sur des symboles manuscrits avec trois modèles générés par symbole et obtient de bons résultats. Dans cette approche, les auteurs ont fait l'hypothèse que les symboles sont d'une épaisseur constante. Une variation de l'épaisseur pourrait donc entrainer des erreurs de classification, de même qu'une épaisseur de traits des symboles trop fine.

Huang et al. [HTL03] ont mis en place un système de reconnaissance de diagrammes basé sur des modèles. Les graphiques décrits par des modèles sont de quatre types : histogramme, diagramme circulaire, graphique sous forme de courbes et graphique en barres. La reconnaissance est réalisée à deux niveaux : un appariement au niveau caractéristique, pour localiser les objets de base, et un appariement au niveau objet, pour déterminer s'il correspond bien à un modèle de diagramme. La première étape

du traitement consiste à séparer le texte des éléments graphiques. La partie texte est traitée par un OCR (Optical Character Recognition), mais le résultat n'est pas très convaincant du fait que les polices des diagrammes sont souvent de très petite taille et suivent différentes orientations. Une vectorisation est appliquée à la partie graphique pour extraire les segments, les arcs de cercle et les contours. Elle consiste à partir d'un ensemble de pixels alignés, à examiner son voisinage horizontalement et verticalement, pour prolonger cet ensemble jusqu'à retrouver un arc de cercle ou un segment. Parmi les éléments vectorisés, les auteurs recherchent les propriétés de parallélisme, perpendicularité, ou convergence qui permettent de décrire les barres, les camemberts, les lignes, ... Les éléments recherchés étant connus, de nombreuses contraintes sont établies pour les caractériser. Ensuite, une mesure de vraisemblance permet de déterminer si un diagramme est présent dans l'image, et si c'est le cas, de connaître son type. Enfin, les données numériques sont extraites en examinant les attributs du diagramme. Les tests de reconnaissance ont été effectués sur des diagrammes issus d'Internet ou de documents scannés. Le type de diagramme est parfaitement reconnu sur cet ensemble de test et l'extraction des données numériques est relativement précise.

C - Signatures structurelles sous forme de graphe

Parmi les signatures sous forme de graphe, nous distinguerons :
- le graphe des vecteurs : où les nœuds sont des vecteurs (segments, cercles, arcs de cercles) et les arcs sont les relations topologiques entre ces vecteurs.
- le graphe des régions : où les nœuds sont des régions et les arcs sont les relations topologiques entre ces régions.

Ces approches à base de graphe sont limitées par les contraintes liés à l'isomorphisme de graphe. En effet, pour reconnaître les symboles, il est possible de réaliser une comparaison des graphes deux à deux. Or, cette comparaison est un problème NP-complet, c'est-à-dire que les algorithmes employés sont exponentiels, il n'existe pas d'algorithme polynomial. Une autre possibilité, pour réaliser l'étape de reconnaissance, consiste à utiliser des grammaires. Le principe est d'appliquer des règles de grammaire pour transformer le graphe à reconnaître en graphe modèle (issu de l'étape d'apprentissage). Enfin, il est également possible de réaliser la reconnaissance en utilisant une distance d'édition. En cas de présence de bruit, des nœuds ou arcs supplémentaires peuvent apparaître dans le graphe qui peuvent ainsi perturber l'analyse des graphes. Pour être assez robuste, un bon appariement doit pouvoir intégrer une certaine tolérance aux erreurs.

Dans la référence [DTOL05], Delalandre et al. proposent une approche à base de graphe pour représenter les symboles. Des pré-traitements (opérations morphologiques et filtrage), adaptés en fonction de l'épaisseur des traits et la surface occupée par la forme, sont tout d'abord appliqués. Ensuite, les auteurs ont la possibilité de construire plusieurs types de graphes : un graphe de régions qui peut contenir différents types d'information (relations d'inclusion, relations de voisinage entre les occlusions et/ou les éléments de base) ; un graphe de vecteurs qui peut combiner plusieurs approches (squelettisation, détection de contours, vectorisation, ...). Ces deux types de graphes sont adaptables selon le contexte de reconnaissance. Un formalisme de connaissances structurelles est mis en place pour justement permettre cette adaptabilité au contexte. La reconnaissance est effectuée par un algorithme d'appariement basé sur un critère de similarité des nœuds et des arcs des graphes.

Dans leurs travaux [YW03, WZY06], Wenyin et al. proposent une description des symboles par un graphe contenant les primitives graphiques structurelles des symboles ainsi que leurs relations (parmi quatre types de contraintes géométriques). L'apprentissage consiste en la construction du graphe avec la description des primitives graphiques (taille, position et orientation) et l'intégration des types de contraintes vérifiées par le symbole (intersection, parallélisme, perpendicularité, arcs de cercle et cercles). Par cette méthode, le graphe de vecteurs obtenu n'est pas unique. La reconnaissance ne peut donc pas être réalisée par un appariement de graphes. Pour reconnaître un symbole, les auteurs recherchent dans le dessin la présence des primitives graphiques et leurs relations. Pour améliorer la reconnaissance, l'utilisateur peut modifier les graphes pour indiquer les primitives graphiques manquantes ou mal reconnues.

Dans l'approche de Qureshi et al. [QJH06], le graphe de vecteurs utilisé est un peu particulier. En effet, chacun de ses nœuds représente un quadrilatère et ses arcs correspondent aux relations spatiales entre paire de quadrilatères. Les quadrilatères sont une forme originale de vectorisation des segments, car généralement les segments sont représentés par leurs contours ou leur squelette. Un quadrilatère est décrit par plusieurs attributs, parmi lesquels on trouve la longueur de son axe médian et une zone d'influence. Cette zone d'influence est de forme rectangulaire et dépend des dimensions du quadrilatère. Seuls les quadrilatères dont les zones d'influence se chevauchent sont en relation dans le graphe de vecteurs. Dans cette application, les nœuds du graphe contiennent la longueur relative du quadrilatère par rapport à la plus grande longueur observée sur l'image, et les relations spatiales se résument à l'angle formé entre les axes médians des quadrilatères. L'appariement des graphes est obtenu en calculant une mesure de similarité. Cette mesure évalue la distance entre les attributs décrivant les nœuds et la distance entre les attributs décrivant les arcs du graphe. De plus, cette mesure intègre des pénalités lorsque un nœud ou un arc du graphe G_1 est associé à

plusieurs autres dans le graphe G_2. Les résultats de reconnaissance obtenus par cette approche sur la base GREC 2003 [GRE03] sont très satisfaisants.

Dans la référence [DHJ06], Deruyver, Hodé et Jolion mettent en place un processus de segmentation en régions, dont le principe est d'associer un graphe d'adjacence de régions et un graphe sémantique. Ces deux types de graphes utilisent un formalisme assez proche. Le graphe sémantique représente les relations entre les composants élémentaires sémantiques de l'image. Il s'agit alors de mettre en correspondance le graphe de régions représentant une segmentation de l'image et le graphe sémantique représentant l'objet recherché. La mise en correspondance utilisée est basée sur un critère de relaxation contrôlant la consistance d'arc du graphe sémantique à deux niveaux de contraintes (inter-nœuds et intra-nœud). Le principe est de faire évoluer le graphe d'adjacence de régions pour le faire correspondre au graphe sémantique. Pour fusionner deux régions du graphe d'adjacence, il faut pouvoir évaluer si elles sont assez similaires, ce qui est déterminé au moyen d'un seuil. La meilleure valeur de seuil permet d'obtenir un minimum de régions et une compatibilité avec le graphe sémantique. L'algorithme est itératif et se poursuit tant que la segmentation permet d'obtenir un graphe arc-consistant. Des tests ont été effectués sur des visages humains pour reconnaître les zones correspondant aux cheveux, aux yeux, à la bouche, ...

Zuwala et Tabbone [ZT06] présentent une méthode de segmentation de symboles dans les documents graphiques sans connaissance a priori. Le traitement débute par la construction d'un graphe de jonctions où les nœuds représentent les chaînes de points reliant deux points de jonction. Le but est ensuite d'isoler les chaînes de points, qui caractérisent un symbole, en fusionnant successivement les nœuds du graphe de jonctions, selon un critère d'aggrégation. Ce critère dépend de la compacité de l'ensemble de chaînes de points et de son degré de convexité. A chaque étape de fusion, le graphe représente un symbole potentiel. Pour la reconnaissance d'un symbole, les candidats sont tout d'abord filtrés à partir de critères géométriques simples. Ensuite, le descripteur statistique ART (Angular Radial Transform) est calculé parmi les symboles candidats restants.

Le tableau 1.4 propose un récapitulatif des caractéristiques des signatures structurelles sous forme de graphes qui viennent d'être présentées.

Conclusion

Les propriétés des trois types de signatures structurelles présentés dans cet état de l'art sont données dans le tableau 1.5. On constate que lorsque la qualité de la description

TAB. 1.4 – Comparaison des signatures structurelles sous forme de graphes

Méthodes	Type de graphe	Principe de la reconnaissance
[DTOL05]	Graphe de régions et graphe de vecteurs	Appariement de graphes selon un critère de similarité des nœuds et des arcs
[YW03, WZY06]	Graphe de vecteurs	Recherche des primitives graphiques et de leurs relations dans les images
[QJH06]	Graphe de vecteurs (quadrilatères)	Appariement de graphes selon un critère de similarité des nœuds et des arcs
[DHJ06]	Graphe de régions et graphe sémantique	Mise en correspondance selon un critère de relaxation contrôlant la consistance d'arc à deux niveaux de contraintes du graphe sémantique
[ZT06]	Graphe de vecteurs (points de jonction et points terminaux)	Filtrage des candidats selon des critères géométriques simples, puis application du descripteur ART

s'améliore, la complexité du traitement augmente. Les méthodes les plus coûteuses offrent aussi des perspectives d'applications intéressantes, telle que la reconnaissance en contexte, c'est-à-dire lorsque les objets graphiques à reconnaître ne sont pas segmentés (isolés).

1.2.3 Utilisation conjointe des approches statistiques et structurelles

Dans les parties précédentes, nous avons présenté des exemples applicatifs d'approches statistiques et structurelles. Ces deux types d'approches possèdent leurs propres avantages. Les signatures statistiques fournissent une information bas niveau relativement robuste au bruit, et les signatures structurelles proposent une représentation plus ou moins complexe qui intègre une description sémantique des objets, plus proche de la perception visuelle humaine. Plusieurs auteurs ont trouvé un intérêt à combiner approches statistiques et structurelles, afin de tirer partie de leurs qualités respectives.

TAB. 1.5 – Propriétés des signatures structurelles

Signatures	Qualité	Précision	Invariance	Complexité	Divers
Sous la forme d'une liste de valeurs numériques	-	-	+	Peu complexe	Liaison simple avec un classifieur statistique
Composée de modèles de vecteurs	+	++	-	Complexité moyenne	Les plus performantes actuellement
Graphe	++	-	+	Très complexe	Simple à adapter pour de la reconnaissance en contexte (ie. symboles non segmentés)

Comme nous l'avons précisé dans le tableau 1.5, l'ensemble des signatures structurelles mises sous forme de listes de valeurs numériques peuvent être intégrées dans un classifieur statistique et ainsi être utilisées conjointement avec une signature statistique. C'est cette approche qui a été choisie par [HPM+98] pour combiner information statistique et structurelle. De leurs côtés, les auteurs des travaux [DHA+02] et [LAT+05] ont choisi un autre procédé : ils ont introduit une description statistique des régions dans une structure de graphe.

Heutte et al. présentent une méthode de reconnaissance de caractères manuscrits [HPM+98] (chiffres, lettres en capitale et graphèmes). A partir de 7 familles de descripteurs structurels et statistiques : moments invariants, projections verticales et horizontales, profils selon quatre directions, intersections avec des lignes droites, occlusions (une occlusion est une fermeture dans la forme, par exemple un "8" comporte deux occlusions, un "0" comporte une occlusion, un "m" n'en comporte aucune) et arcs concaves, extremum, points de terminaison et de jonction, ils composent une signature statistico-structurelle de 124 caractéristiques. L'apprentissage est réalisé en partitionnant les classes deux à deux par des hyperplans. Pour la classification d'un caractère, il faut observer sa position par rapport à chaque hyperplan séparant les paires de classes. Lorsqu'une classe correspond mieux au caractère que les autres classes, un niveau de confiance lui est attribué (distance séparant cette classe de la 2nde meilleure classe). L'étape de reconnaissance renvoie un vecteur de toutes les classes ayant obtenu un niveau de confiance non nul classées par ordre décroissant. Les bons résultats obtenus sur les caractères suggèrent que l'utilisation de classifieurs multiples n'est pas la seule

alternative pour améliorer les taux de reconnaissance.

Delalandre et al. [DHA+02] présentent une approche statistico-structurelle de reconnaissance de symboles. Cette méthode se décompose en plusieurs étapes : extraction des occlusions, extraction des composantes invariantes à l'orientation et au changement d'échelle, reconnaissance statistique des occlusions, construction des graphes d'occlusions composant les symboles, reconnaissance structurelle des symboles. Pour la partie statistique, les occlusions sont caractérisées par trois types de descripteurs : les moments de Zernike, les invariants de Fourier-Mellin et les sondes circulaires. A partir des résultats de reconnaissance statistique des occlusions, un graphe d'occlusions, vérifiant une contrainte de distance et/ou de nombre de connexions entre les occlusions, est ensuite construit. La reconnaissance structurelle des symboles est ensuite calculée par appariement de graphes. L'appariement dépend d'un critère de similarité basé sur le recouvrement entre les graphes. Les tests applicatifs sont réalisés sur des symboles extraits de plans de réseaux France Télécom et des symboles météorologiques.

Locteau et al. [LAT+05] proposent une méthode de classification combinant des descripteurs statistiques et structurels. Chaque symbole est décrit par un graphe d'adjacence de régions. Les nœuds représentent les régions (occlusions du symboles) qui sont décrites selon deux formalismes différents : une description des frontières des régions au moyen de primitives vectorielles et une description du contenu des régions au moyen de moments invariants. Pour l'apprentissage, les auteurs génèrent plusieurs symboles bruités (bruit caractérisant l'impression, la photocopie ou la numérisation) à partir d'un symbole modèle pour obtenir un ensemble plus conséquent de représentants par classe. Au commencement, chaque occlusion se voit attribuer un label, puis le nombre de dimensions est réduit grâce à une méthode d'analyse en composantes principales. Chaque nuage de points est approximé par une distribution gaussienne multivariée. La reconnaissance d'un symbole est alors réalisée par application de la règle de décision de Bayes.

Conclusion

Les signatures statistiques et structurelles qui viennent d'être décrites possèdent des propriétés d'invariance et de robustesse intéressantes. Cependant, les descriptions des symboles qu'elles proposent peuvent s'avérer insuffisantes lorsque le problème de reconnaissance se complexifie. En effet, les systèmes de reconnaissance doivent faire face au problème de passage à l'échelle. Il s'agit de vérifier de quelle manière un système se comporte lorsque le nombre de classes de symboles augmente fortement (plusieurs centaines de classes) et également lorsque le nombre d'objets par classe augmente. Gé-

néralement, les performances de reconnaissance des systèmes ont tendance à chuter avec l'augmentation du nombre de classes. Cette baisse peut s'expliquer par le manque de discrimination des signatures. Mais il faut savoir que plus on cherche à être discriminant, plus la description des symboles doit être précise et complexe, et plus la robustesse au bruit risque de diminuer. Il faut donc trouver le bon compromis entre complexité de la description et robustesse. C'est pourquoi, la robustesse au passage à l'échelle est un sujet de recherche d'importance majeure.

Un autre sujet en pleine expansion concerne la reconnaissance de symboles en contexte. Le problème de reconnaissance est plus complexe car les symboles ne sont pas segmentés. Plusieurs symboles sont disposés sur une même image, et parfois connectés entre eux, comme par exemple sur des schémas électriques. Ce contexte a permis de faire ressortir la dualité du problème [Lla06] : "reconnaître pour segmenter" ou "segmenter pour reconnaître" ? Certains systèmes de reconnaissance fonctionnent bien dans le cas segmenté, mais dans certains contextes, les symboles peuvent être très difficiles à segmenter.

1.2.4 Classifieurs usuels en reconnaissance d'images

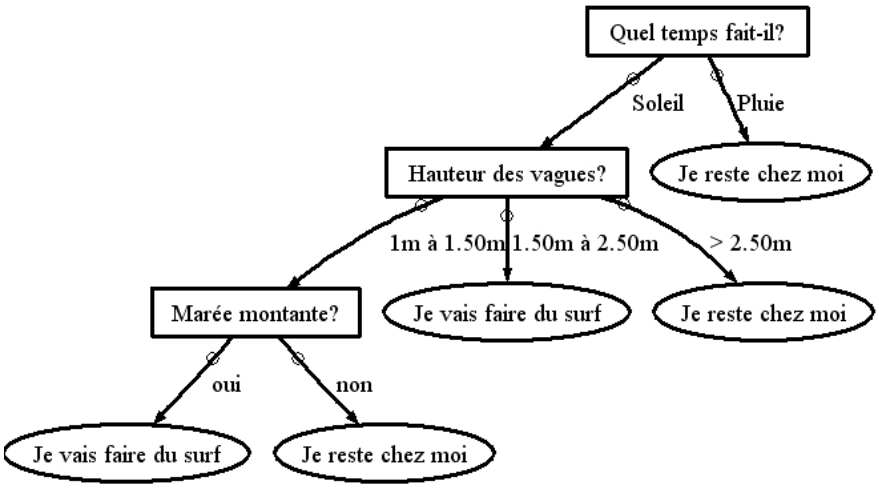
La littérature foisonne de méthodes de classification qui peuvent être réparties en différentes familles selon qu'elles sont basées sur l'utilisation de graphes, de fondements probabilistes, d'une simple distance entre les objets de l'apprentissage, ... Il faut noter qu'elles sont souvent corrélées aux signatures utilisées pour décrire les données.

Les auteurs des travaux présentés dans la référence [LL01] proposent une comparaison du perceptron multi-couches, des réseaux de neurones bayésiens, des machines à vecteur de support et du bayésien naïf. Les résultats expérimentaux sont meilleurs pour les réseaux de neurones bayésiens et les machines à vecteur de support mais ces méthodes nécessitent de trouver la bonne distribution ou le bon paramétrage.

Dans le livre [CMKM02], les auteurs proposent une bonne description des techniques d'apprentissage artificiel et de reconnaissance de formes, et en particulier sur l'utilisation des réseaux de neurones.

La référence [GRC03] donne une comparaison entre les k-plus proches voisins (k-ppv) et les machines à vecteur de support (SVM), dans le domaine de la reconnaissance d'émotions. Ce domaine particulier nécessite un très grand nombre de descripteurs. Les résultats montrent que les SVM perdent leurs avantages face aux k-ppv lorsqu'en amont la sélection des attributs est très efficace.

FIG. 1.1 – Exemple d'arbre de décision



Plus récemment, Gunes, Ménard et Loonis ont présenté un état de l'art sur les classifieurs dans la référence [GMLPR04] ; l'objectif de cette analyse était de les utiliser conjointement.

Nous allons à présent développer plus précisément six méthodes couramment employées en reconnaissance. L'objectif n'est pas de décrire en détail les algorithmes, mais de donner une intuition sur leur fonctionnement et de mettre en avant leurs points forts et leurs faiblesses. Cette étape nous paraît nécessaire pour mieux situer le classifieur utilisé au cours de ce travail de thèse.

Arbre de décision

Les arbres de décision sont largement utilisés dans le domaine de la classification et surtout en classification supervisée. Ils ont notamment fait l'objet de nombreuses recherches par le passé et sont encore actuellement un axe de réflexion important [BK99, CF04, Rak05, Ait08]. Les méthodes de classification basées sur les arbres de décision les plus connues sont CART [BFOS84], ID3 [Qui86] et C4.5 [Qui93].

Un exemple d'arbre de décision est présenté en figure 1.1. Il s'agit de déterminer si la personne va faire du surf ou bien va rester chez elle selon les conditions météorologiques et les marées.

Les nœuds d'un arbre de décision sont appelés *nœuds de décision* car chaque nœud

décrit un test sur l'une des primitives qui doit permettre de partitionner les objets. On peut ainsi associer à chaque nœud de décision l'ensemble D de primitives à tester. Le test doit donc être applicable à l'ensemble des objets. A chaque réponse au test est associée une *branche* qui conduit vers un autre nœud de décision ou bien vers une feuille. Les *feuilles* sont les nœuds terminaux de l'arbre. Chaque feuille correspond à une classe, c'est-à-dire à une prise de décision. Plusieurs feuilles peuvent représenter la même classe.

Lors d'une procédure de classification, on parcourt l'arbre à partir de sa racine (le premier nœud de décision), puis on progresse vers d'autres nœuds de décision en fonction des réponses aux tests, jusqu'à atteindre une feuille, c'est-à-dire trouver la classe de l'objet.

En général, les arbres de décision donnent de bons résultats dans la pratique. Cette structure est intéressante car elle est facilement compréhensible par l'utilisateur. De plus, elle permet une traduction immédiate en règles de décision. Ces règles sont d'ailleurs mutuellement exclusives, c'est-à-dire qu'il n'est pas possible d'avoir deux feuilles différentes possédant les mêmes primitives de description.

Cependant, les arbres de décision possèdent quelques inconvénients. Tout d'abord, les méthodes de construction sont non optimales ; elles ne permettent pas de créer l'arbre, parmi tous les possibles, qui permettra d'obtenir la meilleure classification. De plus, les choix dans la construction ne sont jamais remis en question (pas de backtracking) et l'ordre dans lequel les primitives sont étudiées est figé. L'utilisation d'un ensemble d'échantillonnage peut entraîner un problème de biais inductif. En effet, si l'ensemble d'échantillonnage n'est pas assez représentatif, l'erreur commise sur la classification peut se réveler assez conséquente. Enfin, il peut être difficile de faire un choix parmi les nombreuses heuristiques relative à la construction de l'arbre.

C-Moyennes Floues

Cet algorithme (Fuzzy c-Means) a été créé par Bezdek en 1987 [CDB86]. C'est une méthode de classification non supervisée. Le principe est de former des groupes d'objets d'après leur positionnement et d'essayer de faire converger ces groupes de manière à ce qu'ils correspondent aux classes. Il s'agit d'optimiser un critère qui permet de minimiser l'éparpillement des objets de chaque groupe et de maximiser l'écart entre les groupes. Les groupes sont représentés par leur centre de gravité. Plus précisément, pour obtenir une classification des objets en c classes, il s'agit de sélectionner aléatoirement c points (centroïdes). Ensuite, chaque objet est affecté au centroïde le plus proche en terme de

distance (plus fort degré d'appartenance). Tous les objets affectés à un même centroïde constituent un groupe (cluster). Pour chaque cluster, on peut alors calculer un nouveau centroïde (centre de gravité du cluster). Le processus est itéré jusqu'à obtenir un partitionnement stable, où les objets restent affectés au même cluster pour deux itérations successives. Pour cet algorithme, il est donc nécessaire de fixer le nombre de classes.

Le critère à optimiser est le suivant :

$$J_m(U,V,X) = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^m d^2(x_j, v_i)$$

$$\text{avec } u_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{d(x_j,v_i)}{d(x_j,v_k)}\right)^{\frac{2}{m-1}}} \text{ et } v_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{i=1}^{n} u_{ij}^m}$$

où c est le nombre de classes, $X = \{x_1, \ldots, x_n\}$ représente l'ensemble des objets, $U = \{u_{ij} = u_i(x_j), 1 \leq i \leq c, 1 \leq j \leq n\}$ est la matrice des partitions floues de X avec $u_{ij} \in [0,1]$ le degré d'appartenance de x_j à la classe i, $V = \{v_1, \ldots, v_c\}$ représente les centroïdes des classes, et $m \in [1, +\infty[$ est le facteur flou qui contrôle le recouvrement entre les classes.

L'algorithme 1 présente le déroulement de la classification par les C-Moyennes Floues. Cette méthode permet d'obtenir la matrice U des degrés d'appartenance des objets à chacune des classes. On peut ainsi déterminer la classe la plus plausible (plus fort degré d'appartenance) mais aussi connaître le degré de chevauchement entre les classes. Le problème classique de cette méthode est de pouvoir fixer automatiquement le nombre de classes.

Entrées: c, X, U et V
Sorties: U
début
  Initialiser U et V aléatoirement;
  Initialiser $m > 1$;
  répéter
    Calculer $U = \{u_{ij}$ pour $i = 1, \ldots, c, j = 1, \ldots, n\}$ et
    $V = \{v_i$ pour $i = 1, \ldots, c\}$;
  jusqu'à $|J_m^{t-1} - J_m^t| > \epsilon$;
fin

Algorithm 1: Algorithme des C-Moyennes Floues

K plus proches voisins

Cette méthode de classification très simple et relativement efficace a largement été étudiée [CH67, Das91]. Le principe de cette méthode est de considérer que les objets d'une même classe sont rassemblés au sein de nuages de points et donc censés être proches les uns des autres (au sens de la distance), alors qu'ils seront plus éloignés des objets d'une autre classe. C'est une approche de classification supervisée non paramétrique étant donné qu'elle ne fait pas d'hypothèse sur le modèle des données.

Cette méthode des k plus proches voisins nécessite donc l'utilisation d'une mesure de similarité entre deux objets. Pour désigner la classe à attribuer à un nouvel objet, il faut calculer les distances qui le séparent de tous les autres objets et considérer seulement les classes des objets correspondant aux k plus petites distances obtenues. La classe majoritaire parmi les k objets les plus proches sera affectée à l'objet à reconnaître.

La méthode des k plus proches voisins a le mérite d'être assez simple à mettre en œuvre et les résultats obtenus sont facilement interprétables. En revanche, elle nécessite de conserver l'ensemble des données d'apprentissage, et pour chaque objet à reconnaître de calculer la distance avec tous les objets appris.

Classifieur bayésien

Cette méthode de classification supervisée est dite paramétrique car elle suppose que les données suivent un modèle particulier (une fonction). Après avoir choisi la forme de la fonction, les objets de l'apprentissage permettent d'estimer les paramètres de cette fonction. Comme son nom l'indique, le classifieur bayésien est basé sur la formule de Bayes :

Définition 1 (Formule de Bayes) *Soient A et B deux variables aléatoires. Soient $A = a$ et $B = b$ deux évènements. Par abus de notation, on note $P(a)$ au lieu de $P(A = a)$. La formule de Bayes est donnée par :*

$$P(a|b) = \frac{P(b|a)P(a)}{P(b)}$$

Plus précisément, soit x un objet à classer et $W = \{w_1, \ldots, w_n\}$ un ensemble de n classes. La formule de Bayes s'écrit donc :

$$P(w_i|x) = \frac{P(x|w_i)P(w_i)}{P(x)}$$

Le terme $P(x|w_i)$ représente la probabilité que x appartienne à w_i, $P(w_i)$ est la probabilité a priori de la classe w_i (estimée par la formule $\frac{\text{nombre d'objets de la classe } w_i}{\text{nombre total d'objets}}$ sur la base d'apprentissage), $P(w_i|x)$ est la probabibité a posteriori et $P(x)$ est la probabilité que x se produise ($P(x) = \sum_{i=1}^{n} P(x|w_i)P(w_i)$).

En général, les valeurs de $P(x|w_i)$ et $P(w_i)$ ne sont pas connues et doivent être estimées. L'estimation des $P(w_i)$ ne pose pas de problème. En revanche, celle des $P(x|w_i)$ requiert un ensemble d'apprentissage très grand pour que l'estimation obtenue soit convenable. Dans l'approche du classifieur bayésien naïf, on fait l'hypothèse simplificatrice que l'ensemble des attributs du vecteur de l'objet $x = (x_1, \ldots, x_m)$ sont indépendants connaissant la classe. On obtient alors :

$$P(x_1, \ldots, x_m|w_i) = \prod_{j=1}^{m} P(x_j|w_i)$$

Il faut donc estimer pour chaque attribut x_j de x et pour chaque classe w_i, la proportion d'objets appartenant à la classe w_i qui possède la valeur x_j pour le j-ième attribut de x. On fait alors généralement une hypothèse de normalité sur la répartition des données pour chaque attribut :

$$P(x_j|w_i) \sim \mathcal{N}(x_j; \mu_{x_j w_i}, \sigma^2_{x_j w_i})$$

Finalement, il est nécessaire d'estimer $\mu_{x_j w_i}$ et $\sigma^2_{x_j w_i}$ à partir des objets de l'ensemble d'apprentissage. La classe attribuée à l'objet x à reconnaître sera alors fixée par la probabilité a posteriori pour laquelle l'appartenance est maximale (maximum de vraissemblance) :

$$w_{MAP} = argmax P(w) \prod_{j=1}^{m} \mathcal{N}(x_j; \mu_{x_j w_i}, \sigma^2_{x_j w_i})$$

Le classifieur bayésien naïf a l'intérêt d'être simple à mettre en œuvre. Il est possible d'introduire une fonction de coût associé à un mauvais classement, pour prendre en compte le fait que toutes les erreurs n'ont pas forcément les mêmes conséquences. La règle de décision de Bayes garantit de manière théorique le taux d'erreur de classification minimal. Cependant, dans les problèmes réels, la distribution des classes et les probabilités a posteriori sont inconnues, et ne peuvent qu'être estimées. Cette méthode est basée sur une hypothèse d'indépendance des attributs qui est fausse en général, mais elle offre cependant de bons résultats de classification dans les problèmes réels. Elle fournit ainsi un seuil de performance pour une comparaison avec d'autres méthodes.

Réseaux de neurones

Les réseaux de neurones suivent le principe d'induction, c'est-à-dire d'apprentissage par expérience. Ils permettent de prendre des décisions et trouvent de nombreuses applications en classification, reconnaissance de formes, approximation / modélisation de fonctions, estimations, ... Les réseaux de neurones sont "grossièrement" inspirés du mode de fonctionnement des neurones humains. Ils sont généralement représentés par un graphe qui traduit l'organisation entre les neurones. L'un des modèles les plus utilisé en pratique est le modèle en couches. Chaque couche de neurones reçoit en entrée la sortie des neurones de la couche précédente par le biais de synapses. Entre deux couches de neurones successives, la liaison entre deux neurones sera donc assurée par une synapse qui pondère l'information transférée par un poids synaptique. Un réseau de neurones peut contenir trois types de couches de neurones : la couche d'entrée du réseau de neurones, la couche de sortie et éventuellement des couches cachées qui sont situées entre ces deux couches. Le nombre de couches cachées dépend de la complexité du problème à résoudre.

Le principe d'utilisation d'un réseau de neurones est le suivant : on présente un objet décrit par un ensemble de valeurs au réseau, c'est-à-dire à la couche d'entrée du réseau de neurones. Pour propager l'information de l'objet à un neurone de la couche suivante, chaque valeur d'entrée est multipliée par un poids synaptique et associée aux autres valeurs au moyen d'une *fonction de combinaison*. Le résultat donné par cette combinaison est ensuite envoyé vers une *fonction d'activation* qui détermine la sortie renvoyée par le neurone. Pendant l'apprentissage, les neurones cachés sont entraînés pour reconnaître certains objets et vont diminuer le nombre d'erreurs de classification des neurones de sortie. Les réseaux de neurones permettent d'approximer n'importe quelle fonction bornée. Les paramètres sont estimés par minimisation d'une fonction de moindres carrés.

Il existe différents types de réseaux de neurones qui dépendent de la structure des connexions entre neurones, de la fonction de combinaison, de la fonction d'activation, de l'algorithme d'apprentissage, ... En 1957, le premier réseau capable d'apprendre par expérience est le *perceptron* (Franck Rosenblatt) [Ros58]. Ensuite, le réseau sans doute le plus connu fit son apparition en 1986 : il s'agit du *perceptron multi-couches* (Rumelhart) qui intègre la *rétro-propagation du gradient de l'erreur*. Plus tard, les réseaux ont inclus la théorie de la régularisation statistique (Vladimir Vapnik) pour éviter les problèmes de sur-apprentissage. Les neurones peuvent être à fonction d'activation linéaire ou sphérique (fonction de base radiale), c'est-à-dire qu'ils permettent de réaliser des séparations linéaires ou sphériques de l'espace des données. Les réseaux de neurones à unités sphériques sont également dénommés réseaux à fonction de base radiale ou

réseaux RBF. Leurs fonctions d'activation sont appelées noyaux (le noyau gaussien est le plus utilisé).

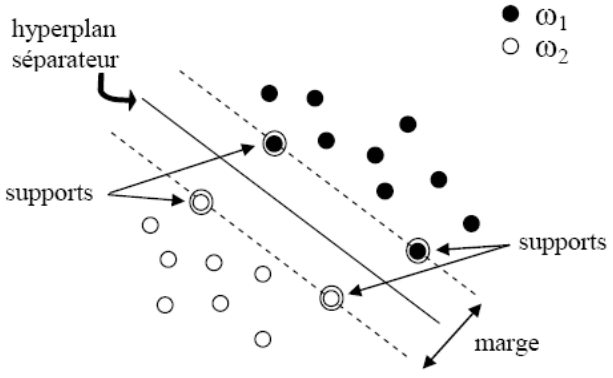
Les réseaux de neurones donnent de très bons résultats en pratique [KTK08, EBM07, TS92]. Les inconvénients des réseaux de neurones sont : l'effet "boîte noire" qui empêche une interprétation du résultat obtenu, la nécessité d'une très grande base d'apprentissage pour les problèmes complexes, et le choix de la famille de fonctions relatif au nombre de neurones cachés.

Machines à vecteur de support

Les machines à vecteurs de support, également appelées séparateur à vaste marge, machine à support vectoriel et en anglais Support Vector Machine (SVM) ont été introduites par V.N. Vapnik en 1992 [BGV92, Vap98].

Le principe est de séparer les classes des objets de l'apprentissage au moyen d'un hyperplan, et ce, de manière optimale (voir Fig. 1.2). En effet, il faut maximiser la taille de l'hyperplan (marge) afin de garantir une grande robustesse au bruit. L'intérêt des machines à vecteurs de support est qu'elles peuvent résoudre des problèmes non linéairement séparables. Pour cela, l'idée est de considérer le problème dans un espace de dimension supérieure où le problème devient linéairement séparable. Pour passer dans un espace de dimension supérieure ou revenir à un espace de dimension inférieure, on utilise un critère de Mercer [Mer09].

FIG. 1.2 – Exemple d'hyperplan obtenu pour un problème à deux classes ω_1 et ω_2



Plus précisément, il s'agit de rechercher un espace optimal (fonctionnelle à minimiser), dans lequel il est possible d'exprimer la solution du problème, sous la forme d'une combinaison linéaire de fonctions noyaux. Le critère de Mercer fournit une méthode de

construction de ces fonctions noyaux. Ces dernières peuvent être vues comme des produits scalaires entre les attributs décrivant les objets de l'apprentissage dans un espace différent de l'espace d'origine.

Cette méthode s'est révélée très performante dans de nombreuses applications (catégorisation de documents [HCT07], reconnaissance d'objets [Ing07], ...), surpassant parfois les réseaux de neurones [SSB+97]. Cependant, les machines à vecteurs de support souffrent tout de même de quelques inconvénients. Pour les problèmes à résoudre dans un espace de grande dimension, cette méthode perd en interprétabilité. De plus, pour traiter de grands ensembles d'apprentissage, la complexité calculatoire est assez élevée. Certains algorithmes ont été mis au point pour décomposer le problème en sous-parties afin de les résoudre en parallèle.

Conclusion

Dans cette partie, nous avons présenté plusieurs méthodes de classification employées en reconnaissance d'images et dans bien d'autres domaines. Elles représentent un panel assez large des possibilités offertes pour la classification automatique. Les tableaux 1.6 et 1.7 présentent un récapitulatif des avantages et inconvénients de ces méthodes.

Nous allons désormais présenter un outil qui n'a, à notre connaissance, jamais été utilisé en reconnaissance d'images. Il s'agit du treillis de Galois. Nous verrons bien que certains travaux ont bien été réalisés dans ce domaine, mais le treillis de Galois n'était pas réellement utilisé comme classifieur, il servait à faire de la sélection d'objets ou d'attributs pertinents. Il semble posséder des propriétés intéressantes comme classifieur :

- une bonne lisibilité : les résultats sont facilement interprétables
- la possibilité de mélanger des données continues et discrètes
- la possibilité d'intégrer des données issues de signatures statistiques et structurelles
- une certaine robustesse au bruit offerte par l'utilisation de plusieurs scénarii de reconnaissance (l'ordre de traitement des attributs n'est pas figé)
- le partitionnement des données est moins important qu'avec un arbre de décision
- la construction obtenue est unique (pas d'heuristiques)

TAB. 1.6 – Récapitulatif des avantages et inconvénients des différents classifieurs

Classifieurs	Avantages	Inconvénients
Arbre de décision	- technique arrivée à maturité - intuitive et résultats facilement interprétables - performances comparables aux autres méthodes supervisées - pas d'hypothèse a priori sur les distributions des données - gestion des données continues et discrètes - robuste aux données atypiques - intègre un procédé de sélection des attributs	- nombreuses heuristiques impliquent un choix difficile - incapacité à gérer les combinaisons d'attributs - nécessite un ensemble d'apprentissage de taille importante - peut engendrer un fort partitionnement des données - la sélection des attributs peut manquer d'efficacité lorsqu'ils sont en grand nombre et en présence de bruit important - l'ordre de traitement des attributs est figé
C-Moyennes floues	- intuitive et résultats facilement interprétables	- fixer le nombre de classes - dimensionnalité : la distance peut être dominée par des attributs non pertinents
k-PPV	- très bons résultats en général - intuitive et résultats facilement interprétables - robuste aux données bruitées - très simple à mettre en œuvre	- dimensionnalité : la distance peut être dominée par des attributs non pertinents - calcul fastidieux nécessite une méthode d'indexation efficace
Bayésien	- garantit théoriquement l'obtention du taux d'erreur de classification minimal - par ces bons résultats, elle est utilisée comme référence pour situer les autres méthodes - apprentissage rapide	- suppose que les données suivent une certaine distribution - nécessite un grand ensemble d'apprentissage pour estimer convenablement les probabilités a posteriori

TAB. 1.7 – Récapitulatif des avantages et inconvénients des différents classifieurs (suite)

Classifieurs	Avantages	Inconvénients
Réseaux de neurones	- très bons résultats en général - robuste aux données bruitées - prise en compte des combinaisons entre les attributs - aptitude à modéliser des problèmes très variés	- effet "boîte noire", difficilement compréhensible - apprentissage long et difficulté d'intégrer des connaissances - fixer le nombre de neurones - sensibilité à un trop grand nombre d'attributs non discriminants
SVM	- très bons résultats en général - robuste aux données bruitées - pas d'hypothèse a priori sur les distribution des données - intègre un procédé de sélection des attributs - résolution des problèmes non linéairement séparables	- interprétabilité difficile pour les problèmes à grande dimension - grande complexité calculatoire pour traiter de grands ensembles d'apprentissage, mais possibilité de décomposer le problème - choix de la fonction noyau

1.3 Treillis et Analyse Formelle des Concepts

Historiquement, les premiers travaux pouvant être rattachés au treillis datent des années 1850 à 1920. Plusieurs chercheurs tels que Boole, Schröder et Dedekind ont ainsi contribué pendant cette période à l'émergence de la théorie des treillis. Cette dernière a par la suite vu le jour de manière effective dans les années 1930-1940. Le treillis a tout d'abord été défini algébriquement à partir des opérations de borne sup et de borne inf pour un ensemble d'éléments [Bir67]. Ensuite, il fut décrit structurellement sous la forme d'un graphe possédant la propriété d'être un ordre, ainsi que des éléments particuliers appelés borne sup et borne inf [BM70]. Depuis une vingtaine d'années, les recherches concernant cette structure ne se limitent plus au cadre théorique structurel et algorithmique. En effet, la structure de treillis est de plus en plus utilisée dans le cadre applicatif, notamment en classification [MNN05] où le treillis de Galois est construit à partir d'une table de données. Ces applications utilisent pleinement les différents résultats structurels et algorithmiques mis au point en recherche théorique. Elles n'ont que récemment été rendues possibles par l'augmentation de la puissance des ordinateurs, car un treillis peut être de taille exponentielle.

L'*analyse formelle des concepts (AFC)* [GW99] a ainsi été introduite [Wil82] pour fournir un cadre théorique au treillis de Galois et à ses applications nombreuses, notamment en traitement des données et des connaissances, domaine qui comprend la visualisation des données, la fouille des données, l'apprentissage, l'analyse des données et la gestion des connaissances.

Le *treillis de Galois*, ou encore *treillis des concepts* est une relation de treillis définie sur un ensemble d'éléments appelés concepts. Il a fait l'objet de premiers travaux formels en théorie des graphes et des structures ordonnées [Bir67, BM70, DP91]. Il a ensuite été introduit en analyse de données et classification [Wil82, GD86, MNN05] où il a rapidement montré son utilité : la structure de treillis, basée sur la notion de *concept*, permet de décrire les données tout en conservant leur diversité, mais aussi leur complexité.

Dans cette partie, nous donnerons les fondements mathématiques du treillis et de l'AFC (voir § 1.3.1), quelques aspects algorithmiques (voir § 1.3.2), puis nous décrirons quelques méthodes de classification basées sur l'AFC qui utilisent le treillis de Galois (voir § 1.3.3) ou les règles d'association (voir § 1.3.4). Dans le domaine de l'AFC, très peu d'études ont été réalisées sur l'utilisation de ces outils en reconnaissance d'images et encore moins en reconnaissance de symboles bruités, c'est pourquoi nous décrirons plus généralement des études menées en classification supervisée.

1.3.1 Fondements mathématiques

Comme nous venons de le préciser, la notion de treillis a d'abord fait l'objet d'une définition algébrique, puis d'une définition structurelle. Algébriquement, un treillis est un ensemble X muni de deux opérateurs binaires algébriques appelés borne sup et borne inf [Bir67]. Structurellement, un treillis est un ensemble X muni d'une *relation d'ordre* tel que toute paire $\{x, y\}$ de ses éléments admet une *borne inférieure* $x \wedge y$ et une *borne supérieure* $x \vee y$.

Pour bien comprendre cette dernière définition, il est nécessaire de décrire les notions de relation d'ordre, de borne inférieure et de borne supérieure, elles-mêmes définies à partir des minorants et des majorants.

Une relation binaire O sur un ensemble X est une *relation d'ordre* sur X si elle vérifie les trois propriétés suivantes :

1. réflexivité : pour tout $x \in X$, $x\, O\, x$,

2. antisymétrie : pour tous x, $y \in X$, ($x\,O\,y$ et $y\,O\,x$) impliquent $x = y$,
3. transitivité : pour tous x, y et $z \in X$, ($x\,O\,y$ et $y\,O\,z$) impliquent $x\,O\,z$.

Notons qu'un ensemble X muni d'une relation d'ordre O est appelé *ensemble ordonné*.

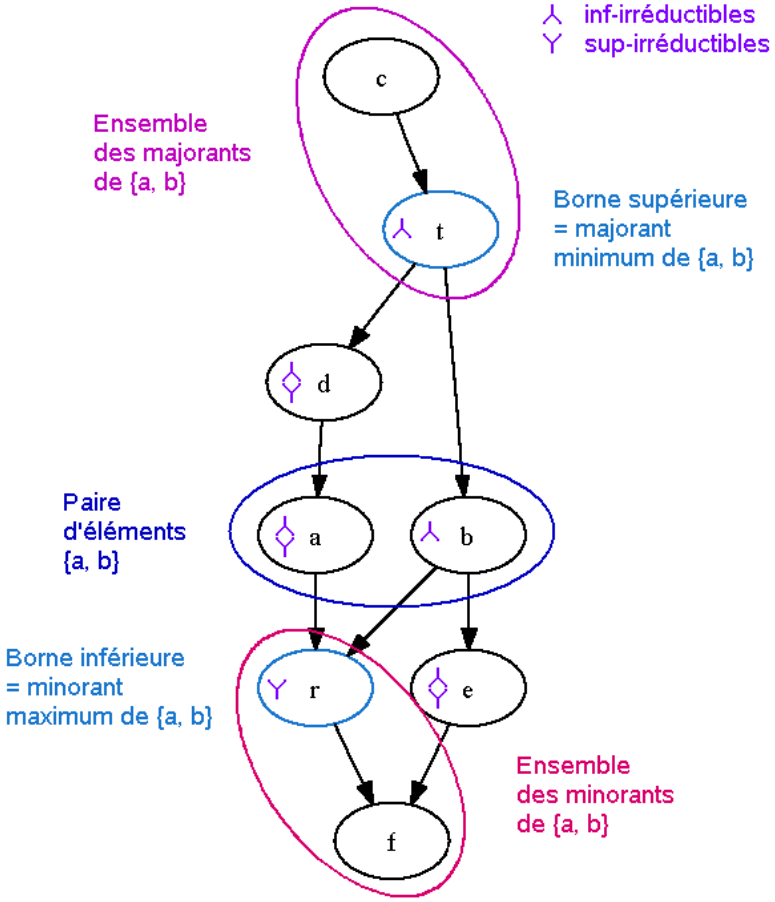
Pour des besoins de lisibilité, lorsque l'on souhaite représenter graphiquement un ensemble ordonné, on présente le plus souvent son diagramme de Hasse. Le *diagramme de Hasse* d'un graphe, est la réduction réflexive et transitive du graphe. De plus, on associe au diagramme de Hasse la *relation de couverture* notée $\prec$.

Soit Y une partie d'un ensemble ordonné X. Un *minorant* (respectivement, un *majorant*) de Y est un élément m de X tel que $m \leq x$ (respectivement, $m \geq x$) pour tout $x \in Y$. La partie Y est dite *minorée* (respectivement, *majorée*) si elle admet au moins un minorant (respectivement, un majorant). On dit que $r \in X$ est la *borne inférieure* de Y (ou son *infimum*, conjonction, meet) si Y est minorée et si l'ensemble de ses minorants admet r pour maximum. De même, Y a une *borne supérieure* (ou *supremum*, union, join) t si elle est majorée et si l'ensemble de ses majorants admet t pour minimum. Un treillis admet un unique élément minimal appelé *bottom* et noté $\bot$, ainsi qu'un unique élément maximal appelé *top* et noté $\top$. Par contradiction, s'il existait deux éléments minimaux (resp. maximaux) distincts, alors ces deux éléments n'auraient pas de borne inférieure (resp. supérieure), ce qui est contradictoire avec la propriété de treillis.

Prenons l'exemple de l'ensemble $X = \{a, b, c, d, e, f, r, t\}$ muni d'une relation d'ordre possédant la propriété de treillis. La figure 1.3 illustre les définitions précédentes. Elle présente également les notions d'inf-irréductible et sup-irréductible définies de la manière suivante. Soit Y une partie d'un ensemble ordonné X. $x \in Y$ est *inf-irréductible* s'il n'est borne inférieure d'aucune partie ne le contenant pas. De même, $x \in Y$ est *sup-irréductible* s'il n'est borne supérieure d'aucune partie ne le contenant pas. De plus, il est possible de caractériser un inf-irréductible (respectivement sup-irréductible) comme un élément x qui ne possède qu'un seul arc entrant (respectivement sortant) dans le diagramme de Hasse de l'ensemble ordonné.

Une relation d'ordre classique est la relation d'inclusion. En effet, elle vérifie les propriétés de réflexivité, de transitivité et d'antisymétrie. On appelle *système* sur un ensemble X, une famille de sous-ensembles munie de la relation d'inclusion sur X. Un système vérifie la propriété de treillis si et seulement si il est stable par intersection ($F_1, F_2 \in \mathbb{F}$ implique $F_1 \cap F_2 \in \mathbb{F}$) et s'il contient X. Il est alors appelé *système de fermeture* ou *famille de Moore*. Les sous-ensembles appartenant à un système de fermeture $\mathbb{F}$ sont appelés les *fermés* de $\mathbb{F}$. L'ensemble des fermés reliés par inclusion forme le *treillis des fermés*. La borne inférieure et la borne supérieure s'expriment alors à partir

FIG. 1.3 – Exemple de treillis



de la relation d'inclusion : pour F_1 et $F_2 \in \mathbb{F}$:
- $F_1 \bigwedge F_2 = F_1 \bigcap F_2$
- $F_1 \bigvee F_2$ est le plus petit fermé $F \in \mathbb{F}$ tel que F contient $F_1 \bigcup F_2$

Les opérations de borne inférieure et de borne supérieure sont définies "en dualité". Cette propriété fondamentale permet de représenter la spécificité du domaine étudié.

On associe à tout système de fermeture un *opérateur de fermeture* h. Un opérateur de fermeture sur S vérifie les propriétés suivantes :
- $X \subseteq h(X), \forall X \subseteq S$ (extensivité)
- si $X \subseteq X'$, alors $h(X) \subseteq h(X'), \forall X, X' \subseteq S$ (monotonie)
- $h(X) = h(h(X)), \forall X \subseteq S$ (idempotence)

Treillis de Galois

Partant de la définition d'un treillis et plus particulièrement de celle d'un treillis des fermés, il est désormais possible de donner une description du *treillis de Galois*. Contrairement au treillis des fermés, le treillis de Galois est défini à partir, non pas d'un, mais de deux ensembles (objets et attributs). La mise en correspondance d'une partie de l'ensemble des objets avec une partie de l'ensemble des attributs forment un élément du treillis de Galois.

Un treillis de Galois est une représentation sous forme de graphe de toutes les correspondances maximales possibles entre un ensemble d'*objets* (ou individus, instances, exemples) O et un ensemble d'*attributs* (ou primitives, caractéristiques) I. Les relations (ou correspondances) qui lient l'ensemble des objets à celui des attributs sont décrites par un *contexte formel*. Précisons que l'ensemble de ces relations entre objets et attributs constituent les données à partir desquelles le treillis de Galois est défini. Ces données sont rassemblées sous la forme d'une table appelé contexte formel. Un *contexte formel* C est un triplet $C = (O, I, R)$, où O est un ensemble d'objets, I est un ensemble d'attributs et R une *relation binaire* $R \subseteq O \times I$ entre O et I.

Exemple 1 *La table 1.8 donne un exemple de contexte formel décrivant les réponses de 5 personnes à un sondage sur les activités extra-scolaires. L'ensemble O correspond donc aux personnes interrogées et l'ensemble I aux réponses données. La relation binaire entre O et I est la relation : l'individu $o \in O$ "a donné la réponse" $i \in I$.*

TAB. 1.8 – Exemple de contexte

Réponses	Sexe		Age		Loisirs			
Individus	H	F	[11-15]	[16-20]	Sports d'équipe	Sports individuel	Activités artistiques	TV
1		X	X		X	X		X
2		X		X	X		X	X
3	X		X			X	X	X
4	X		X		X			X
5	X			X	X	X		X

Pour obtenir les attributs qui décrivent un même ensemble d'objets, ou inversement les objets qui partagent un même ensemble d'attributs, il est nécessaire de définir deux fonctions : l'une pour le passage de O vers I et l'autre pour le passage de I vers O. Plus formellement, on associe à un ensemble d'objets $A \subseteq O$ l'ensemble $f(A)$ des attributs en relation R avec les objets de A :

$$f(A) = \{x \in I \mid pRx \ \forall \ p \in A\}$$

Dualement, pour un ensemble d'attributs $B \subseteq I$, on définit l'ensemble $g(B)$ des objets en relation avec les attributs de B :

$$g(B) = \{p \in O \mid pRx \ \forall \ x \in B\}$$

Ces deux fonctions f et g définies entre objets et attributs forment une *correspondance de Galois*.

Chaque correspondance maximale objets-attributs en relation selon R ainsi définie est représentée par un nœud du treillis appelé concept formel. Un *concept formel* est un couple (A, B) avec $A \subseteq O$ et $B \subseteq I$, qui vérifie $f(A) = B$ et $g(B) = A$. L'ensemble des concepts formels est noté $\beta(C)$. Notons que l'*extension* (respectivement, l'*intention*) d'un concept (A, B) est le sous-ensemble d'objets A (respectivement, le sous-ensemble d'attributs B) qui le compose.

Exemple 2 *En reprenant l'exemple précédent sur le sondage (voir Ex. 1) dont le contexte formel est décrit par la table 1.8, on peut constater que la fonction f permet d'obtenir à partir d'un ensemble d'individus, toutes les réponses au sondage qui sont communes à ces individus. A l'inverse, la fonction g permet d'obtenir pour un ensemble donné de réponses, tous les individus qui les ont formulées. Ainsi, si l'on recherche les individus masculins qui ont entre 11 et 15 ans, on obtient : $g(H \text{ et } [11-15]) =$*

individus 3 *et* 4. *A l'inverse, pour connaître les réponses partagées par ces deux individus, on applique :* f(*individus 3 et 4*) = *H, [11-15] et TV. En appliquant le résultat de la fonction* g *à la fonction* f, *nous venons en fait d'utiliser l'opérateur de fermeture* $f \circ g$. *Cet opérateur nous permet d'identifier la présence d'un concept formel :* $(\{$*individus 3 et 4*$\}, \{$*H, [11-15] et TV*$\})$, *où* $\{$*individus 3 et 4*$\}$ *est l'extension du concept et* $\{$*H, [11-15] et TV*$\}$ *son intention. Il est possible de vérifier que l'on a bien* f(*individus 3 et 4*) = $\{$*H, [11-15] et TV*$\}$ *et* g(*H, [11-15] et TV*) = $\{$*individus* 3 *et* 4$\}$.

Nous avons vu précédemment qu'un *treillis* est un ensemble muni d'une *relation d'ordre*. Dans le cas du treillis de Galois, cette relation d'ordre est la relation $\leq$ basée sur l'inclusion. Ainsi, deux concepts formels (A_1, B_1) et (A_2, B_2) seront reliés par un arc dans le treillis s'ils vérifient :

$$(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow \left\| \begin{array}{l} A_2 \subseteq A_1 \\ \text{(équivalent à } B_1 \subseteq B_2\text{)} \end{array} \right.$$

Le *treillis de Galois* associé à un contexte C est l'ensemble de tous les concepts $\beta(C)$ muni de la relation d'ordre $\leq$. Il est donc défini par la paire $(\beta(C), \leq)$. Notons que le treillis de Galois du contexte C est noté $Gal(C)$.

Intuitivement, les concepts formels sont donc ordonnés selon une double inclusion : au fur et à mesure que l'on "descend" dans le graphe, les concepts contiennent de moins en moins d'objets mais une description des objets de plus en plus précise étant donné que le nombre d'attributs augmente.

Exemple 3 *Pour illustrer ces différentes définitions, reprenons l'exemple du sondage (voir Ex. 1 et 2) où le contexte formel est décrit par la table 1.8. Rappelons que nous avons obtenu le concept formel suivant :* $(\{$*individus 3 et 4*$\}, \{$*H, [11-15] et TV*$\})$. *En suivant le même pincipe que celui énoncé précédemment (voir Ex. 2), nous construisons un second concept formel :* $(\{$*individus 3, 4 et 5*$\}, \{$*H et TV*$\})$. *Or, ces deux concepts vérifient la double inclusion :* $\{$*individus 3 et 4*$\} \subseteq \{$*individus 3, 4 et 5*$\}$ *et* $\{$*H et TV*$\} \subseteq \{$*H, [11-15] et TV*$\}$, *ils seront donc reliés par un arc dans le treillis.*

La restriction à l'ensemble I des concepts du treillis de Galois est un système de fermeture défini sur I. L'opérateur de fermeture est alors $f \circ g$. Dualement, on peut retrouver un système de fermeture défini sur O dont l'opérateur de fermeture est $g \circ f$.

On définit la borne inférieure et la borne supérieure de deux concepts formels (A_1, B_1) et (A_2, B_2), respectivement par :

– $(A_1, B_1) \bigwedge (A_2, B_2) = (A_3, B_3)$ tel que $A_1 \bigcup A_2 \subseteq A_3$ et $(B_1 \bigcap B_2) = B_3$
– $(A_1, B_1) \bigvee (A_2, B_2) = (A_4, B_4)$ tel que $(A_1 \bigcap A_2) = A_4$ et $(B_1 \bigcup B_2) \subseteq B_4$

Comme tout treillis, un treillis de Galois contient un unique *concept minimal* $\top = (O, f(O))$ et un unique *concept maximal* $\bot = (g(I), I))$, selon la relation $\leq$.

Pour améliorer la lisibilité du graphe associé à un ensemble ordonné, nous représentons le plus souvent son *diagramme de Hasse* (défini précédemment), auquel on associe la relation de couverture $\prec$. On obtient alors le diagramme de Hasse $(\beta(C), \prec)$ du treillis des concepts $(\beta(C), \leq)$ (voir Fig. 1.4).

Nous avons déjà précisé que les relations entre les ensembles d'objets et d'attributs à partir desquels le treillis de Galois est défini, sont représentées par une table également appelée contexte formel. A un treillis de Galois donné, peuvent correspondre plusieurs contextes formels équivalents. Cependant, il existe parmi ces contextes équivalents un unique contexte réduit, appelé le *contexte réduit*, qui offre une représentation condensée des données. $Red(C)$, le *contexte réduit* d'un contexte formel $C = (O, I, R)$, s'obtient en supprimant :

1. un objet partageant les mêmes attributs qu'un autre objet du contexte
2. un attribut partagé par les mêmes objets qu'un autre attribut du contexte
3. un objet partageant les mêmes attributs qu'une combinaison d'autres objets du contexte
4. un attribut partagé par les mêmes objets qu'une combinaison d'autres attributs du contexte

$Red(C)$ est équivalent à C car $Gal(C) \cong Gal(Red(C))$: le treillis de Galois associé au contexte C est isomorphe au treillis de Galois associé au contexte réduit de C. Notons qu'il existe une bijection entre les attributs et les objets du contexte réduit de C et les inf et sup-irréductibles du treillis de Galois associé à C :

$$Red(C) = (O_{irr}, I_{irr}, R \cap O_{irr} \times I_{irr})$$

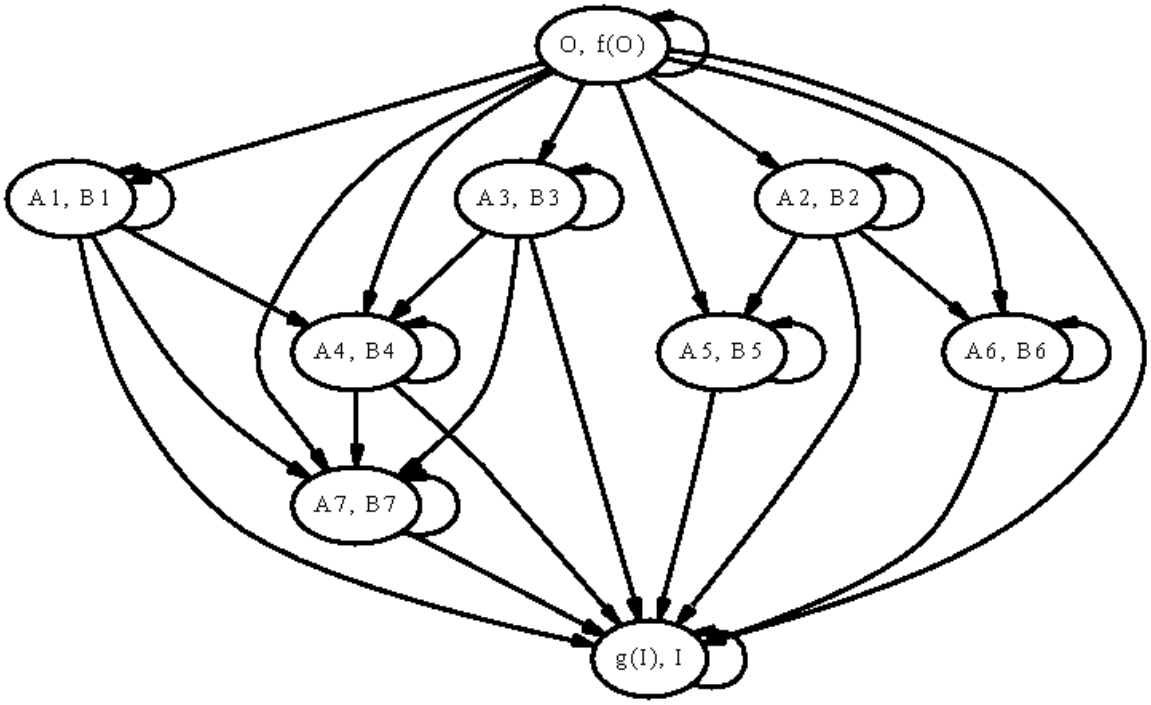
avec $O_{irr} = \{\vee\text{-irréductibles } (Gal(C))\}$ et $I_{irr} = \{\wedge\text{-irréductibles } (Gal(C))\}$.

Système implicatif

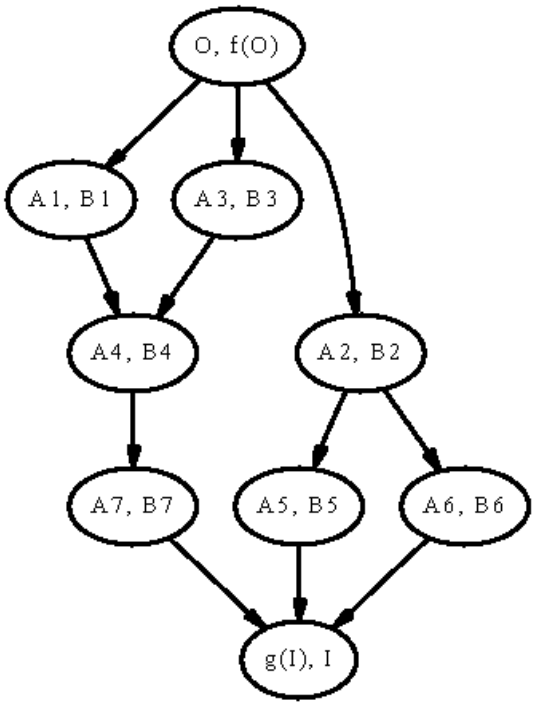
Un treillis de Galois est une extension d'un treillis des fermés. On peut également définir à partir d'un treillis des fermés un *système implicatif* ou *base de règles*.

Les *règles d'implication* ou *règles exactes* sont de la forme $R : X \rightarrow Y$ où X et $Y \subseteq S$. Une partie $X' \subseteq S$ vérifiant une règle $X \rightarrow Y$ est telle que $X \subseteq X'$ implique

FIG. 1.4 – Exemple d'un treillis de Galois et de son diagramme de Hasse



(a) treillis de Galois



(b) diagramme de Hasse du treillis

$Y \subseteq X'$. L'ensemble de toutes les règles est un système de fermeture. Les ensembles X et Y sont respectivement appelés *premisse* (ou hypothèse) et *conclusion* (ou conséquence) de la règle R.

Plus généralement, une règle d'association sur S est une règle $X \rightarrow Y$ telle que $X \subseteq S$, $Y \subseteq S$, vérifiant une mesure de qualité. Les mesures de qualité les plus populaires sont le *support* et la *confiance*. Soit un contexte binaire (O, I, R) où O est un ensemble d'objets, I un ensemble d'attributs et R une relation binaire entre O et I. Soient X et Y des ensembles d'attributs tels que $X \in I$ et $Y \in I$. Le support de X correspond au rapport entre le nombre d'objets qui vérifient tous les attributs de X et l'ensemble de tous les objets $|O|$ du contexte, c'est-à-dire : $Supp(X) = \frac{|X'|}{|O|}$ avec $X' = g(X)$. Le support d'une règle d'association $X \rightarrow Y$ est donné par $Supp(X \rightarrow Y) = Supp(X \cup Y)$. Quant à elle, la confiance d'une règle $X \rightarrow Y$ est la proportion d'objets vérifiant tous les attributs de Y parmi les objets vérifiant tous les attributs de X. Plus précisément, la confiance est donnée par : $Conf(X \rightarrow Y) = \frac{Supp(X \rightarrow Y)}{Supp(X)}$.

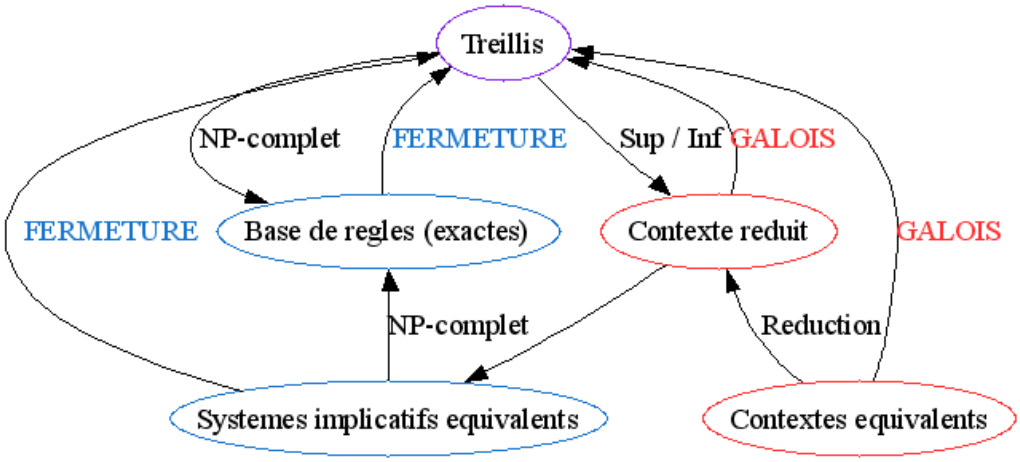
Remarquons qu'une *règle d'implication* ou règle exacte a une confiance égale à 100%. Une *règle d'association* est une règle approximée ou exacte, c'est-à-dire une règle dont la confiance est inférieure ou égale à 100%. Une règle d'association est dite valide si son support et sa confiance sont respectivement supérieurs ou égaux aux deux seuils $minsup$ et $minconf$ représentant respectivement un support minimal et une confiance minimale. Une *règle de classification* comprend en prémisse un ensemble d'attributs et en conclusion un label de classe. Ainsi une règle de classification s'écrit : $R : X \rightarrow c$ où c est un label de classe.

Conclusion

Dans cette première partie, nous avons présenté plusieurs définitions concernant les treillis, et notamment, une caractérisation du treillis à partir d'un contexte formel. Notons cependant qu'il existe d'autres moyens pour obtenir un treillis. La figure 1.5 représente les relations existantes entre un contexte (une table), une base de règles (un système implicatif) et un treillis. Pour un ensemble de contextes équivalents, l'opération de réduction nous permet d'obtenir un unique contexte réduit. Outre cette unicité (canonicité), l'intérêt de ce contexte réduit est qu'il contient les inf et sup-irréductibles. Ensuite, à partir d'un contexte, on peut construire un treillis (de Galois) en utilisant la correspondance de Galois. De manière similaire, pour un ensemble de systèmes implicatifs équivalents, il est possible d'obtenir une unique base de règles, la base canonique. A partir d'un système implicatif ou de cette base de règles, on peut construire un treillis (des fermés) par opération de fermeture. Un treillis peut donc être obtenu soit à partir

d'un contexte (treillis de Galois), soit à partir d'une base de règles (treillis des fermés).

FIG. 1.5 – Liens entre contextes, bases et treillis



1.3.2 Aspects algorithmiques

Parmi les nombreux algorithmes permettant de générer le treillis de Galois, citons ceux de Chein [Che69], Norris [Nor78] (premier algorithme incrémental), Ganter [Gan84] (NextClosure), Bordat [Bor86], Godin et al. [GMA91] (proche de celui de Norris, incrémental), Carpineto et Romano [CR93] (incrémental), Stumme et al. [STB+02] (Titanic), Valtchev et al. [VML02] et celui de Nourine et Raynaud [NR99] qui a la meilleure complexité théorique (complexité quadratique par élément du treillis).

Des études comparatives ont également été publiées telles que Godin et al. [GMM95] en 1995, et Kuznetsov et Obiedkov [KO02] en 2002, Fu et Mephu Nguifo [FMN04] en 2004. Une comparaison [KO01] a récemment été menée entre plusieurs algorithmes de génération du treillis de Galois et sur un même jeu de données. Ces algorithmes ont une complexité polynomiale (au mieux quadratique dans [NR99]) par concept généré, et dépendent donc de la taille du treillis de Galois. La taille du treillis est bornée par $2^{|O+I|}$ dans le pire des cas, et par $|O + I|$ dans le meilleur des cas. Des études de complexité en moyenne sont extrêmement difficiles à mener, car la taille du treillis dépend à la fois de la taille des données à classifier, mais aussi de leur organisation et de leur diversité. Notons cependant que sa taille reste raisonnable en pratique, comme l'illustrent les expérimentations qui en ont déjà été faites [MNN05]. De récents travaux [Gel05] proposent un algorithme générique permettant à la fois d'unifier les algorithmes existants dans un même cadre, mais aussi de les comparer en fonction des propriétés des données, et par conséquent du treillis.

Nous allons à présent décrire plus précisément les deux algorithmes de construction du treillis de Galois les plus connus et les plus utilisés : NextClosure [Gan84] et l'algorithme de Bordat [Bor86]. Le premier calcule les concepts dans un ordre lectique (proche de l'ordre lexicographique), puis les ordonne par inclusion. Quant au second, il calcule directement le diagramme de Hasse en utilisant la relation successeur.

Algorithme NextClosure

Le principe général de l'algorithme NextClosure de Ganter [Gan84] est de générer les concepts selon l'ordre lectique, un ordre proche de l'ordre lexicographique.

Soit S un ensemble d'éléments. L'ordre lectique, noté $\leq_i$, est un ordre total sur 2^S qui étend l'inclusion : $X \subseteq Y \Rightarrow X \leq_i Y$:

$$X <_i Y \Leftrightarrow \exists i \in Y - X \text{ tel que } X|i = Y|i$$

où $X|i$ est la restriction de X aux (i-1) premiers éléments de S.

L'algorithme Alg. `NextClosure` présente la génération des fermés à partir d'un ensemble d'éléments S. Sa complexité algorithmique est $O(|S|^3)$. Dans cet algorithme, h est l'opérateur de fermeture $g \circ f$ ou $f \circ g$.

Nom : `NextClosure`
Données: Un ensemble d'éléments S ($S = I$ ou $S = O$), un fermé X
Résultat: Un fermé Y
début
  $Z = S \backslash X$ trié par ordre décroissant;
  pour chaque $x_i \in Z$ **faire**
    $Y = h(X|i \bigoplus x_i)$;
    si $X <_i Y$ **alors** NextClosure(Y);
  fin
fin

Algorithme de Bordat

L'algorithme de Bordat est issu du théorème 1 qui définit la relation de couverture d'un concept.

Théorème 1 (Bordat [Bor86]) *Soient (A, B) et (A', B') deux concepts d'un contexte (O, I, R). Alors $(A, B) \prec (A', B')$ si et seulement si A' est maximalement inclus dans l'ensemble suivant F_A défini sur O :*

$$F_A = \{g(b) \cap A \ : \ b \in I - B\} \tag{1.1}$$

Corollaire 1 (Bordat [Bor86]) *Soit (A, B) un concept. Il y a une correspondance exacte entre $Couverture((A, B))$ et les sous-ensembles maximalement inclus de F_A où :*

$$Couverture((A, B)) = \{(A', B') \ : \ (A, B) \prec (A', B')\} \tag{1.2}$$

L'algorithme de Bordat (voir Alg. `Treillis de Galois`) calcule tous les concepts de $\mathbb{C}$ en appelant $Couverture(A, B)$ pour chaque concept (A, B), à partir du concept minimal $\perp = (O, f(O))$, jusqu'à ce que tous les concepts soient générés. Sa complexité est $O(|\mathbb{C}||I|^{\alpha})$, avec $2,5 \leq \alpha \leq 3$, puisque chaque concept est issu du calcul de $Couverture((A, B))$ dont la complexité est $O(|I|^{\alpha})$.

Nom : `Treillis de Galois`
Données: Un contexte $C = (O, I, R)$
Résultat: Un diagramme de Hasse du treillis de Galois de C
début
  $\mathbb{C} = \{(O, f(O))\}$;
  pour chaque $(A, B) \in \mathbb{C}$ *non marqués* **faire**
    F=`Couverture` $(K, (A, B))$;
    pour chaque $A' \in F$ **faire**
      $B' = f(A')$;
      si $(A', B') \notin \mathbb{C}$ **alors** Ajouter (A', B') dans $\mathbb{C}$;
      Ajouter la relation de couverture $(A, B) \prec (A', B')$;
    fin
    Marquer (A, B);
  fin
  Retourner $(\mathbb{C}, \prec)$;
fin

L'algorithme de couverture (voir Alg. `Couverture des objets`) décrit les deux étapes du calcul de $Couverture((A, B))$: l'ensemble F_A doit tout d'abord être généré en temps linéaire (en particulier, F_A peut être calculé de manière incrémentale) ; ensuite, les sous-ensembles maximalement inclus de F_A peuvent être facilement calculés en $O(|I|^3)$. Notons tout de même que ce calcul des sous-ensembles maximalement inclus est solvable avec une complexité plus faible ($O(|I|^{2,5})$) en utilisant des structures de données sophistiquées [FM71, I.71].

Nom : `Couverture des objets`
Données: Un contexte K ; Un concept (A, B) de K
Résultat: Les sous-ensembles maximalement inclus de F_A
début

1. Calculer F_A : $F_A = \{g(b) \cap A \ : \ b \in I - B\}$;
2. Calculer F : les sous-ensembles maximalement inclus de F_A;

Retourner F;
fin

1.3.3 Classification avec le treillis de Galois

Nous allons maintenant présenter sept méthodes de classification basées sur le treillis de Galois et une méthode d'indexation d'images (voir § 1.3.3-A). Cette comparaison est motivée par une étude récente [MNN05] qui recense quelques méthodes de classification supervisée basées sur un treillis de Galois. Elle apporte des éléments de comparaisons et les résultats y sont semblables voire meilleurs que ceux d'approches plus classiques.

Dans notre étude comparative, les méthodes de classification présentées utilisent le treillis pour choisir les éléments (objets, attributs) les plus pertinents parmi les données de l'apprentissage. Ensuite, la phase de classification est réalisée à partir des éléments précédemment sélectionnés et consiste à appliquer des classifieurs usuels de type k-plus proches voisins, bayésien, ou encore vote majoritaire. Nous proposons ensuite une comparaison théorique de ces sept méthodes de classification (voir § 1.3.3-B).

Les données utilisées dans les expérimentations ne sont pas pour la plupart des données issues d'images. Elles proviennent en majorité des bases de données du Machine Learning Repository [MM96, BKM98]. On trouve également un jeu de données issu de la biologie moléculaire GenBank 64.1. Seule la méthode de Zenou et Samuelides est en fait réalisée sur des images.

A - Description des méthodes

GRAND

La méthode GRAND (GRAph iNDuction) a été développée par Oosthiuzen [Oos88]. La particularité de cette méthode est de construire de manière incrémentale le treillis de Galois. Habituellement, pour construire le treillis, on détermine l'ensemble des nœuds

de manière exhaustive, puis on recherche l'ensemble des arcs reliant ces nœuds. L'intérêt de l'algorithme GRAND est qu'il construit de manière incrémentale des sous-treillis de la structure finale.

Apprentissage A partir d'un objet, on crée le graphe orienté entre cet objet et les attributs de la table. Le graphe obtenu doit vérifier un certain nombre de propriétés d'invariance pour être considéré comme étant le treillis de Galois uniquement déterminé par ce premier objet. Dans le treillis, seules les intentions des concepts sont représentées, les concepts min et max sont supprimés s'ils sont vides. Les nœuds sont séparés en trois types :

les nœuds-attributs : nœuds contenant un attribut singleton

les nœuds-objets : nœuds contenant un objet singleton

les nœuds intermédiaires : tous les autres nœuds du treillis

Pour insérer un second objet dans le graphe, il faut ajouter des éléments au treillis de l'étape précédente, puis vérifier que les propriétés d'invariance sont toujours respectées. On obtient alors un nouveau treillis de Galois déterminé par les deux objets. On reproduit ainsi ce procédé jusqu'à ce que tous les objets soient insérés. La vérification à chaque insertion des propriétés d'invariance permet de garantir l'obtention d'une structure de treillis. Un élagage est proposé pour supprimer les concepts statistiquement insignifiants, c'est-à-dire les concepts vérifiés par un nombre trop faible d'objets.

La classe des objets est un attribut parmi les autres. A partir du treillis final, les auteurs extraient des règles de classification pour chaque classe, suivant le principe d'inférence sur les attributs. Le principe est le suivant : pour une classe donnée, on recherche le concept intermédiaire le plus général qui permette d'inférer la classe. En d'autres termes, le concept recherché est la borne inférieure contenant cette classe. Les attributs du concept deviennent l'hypothèse de la règle et la classe devient sa conclusion.

Classification Pour le classement d'un objet, il faut appliquer les règles les plus spécifiques à l'objet. Toutes les règles de classification vérifiées par l'objet sont collectées, puis un vote majoritaire est appliqué.

Expérimentation GRAND a été comparé sur deux jeux de données Breast Cancer et Lymphography [BKM98] à d'autres méthodes : Assistant [CKB87], AQ15 [MMHL86], AQR [MMHL86], Bayes et CN2 [CN89]. Les tests de reconnaissance sont effectués en validation croisée. Le résultat final est la moyenne des taux de précision obtenus pour chaque bloc. GRAND obtient des taux de précision comparables aux autres méthodes.

Dans [VOR97], les auteurs présentent DATAMAP, un programme permettant la visualisation et la navigation au sein d'un treillis de Galois. L'utilisateur choisit un ensemble d'attributs et il visualise alors la borne inférieure correspondante à ces attributs ainsi que tous les concepts qui spécifient ou généralisent la borne inférieure. La borne inférieure est représentée de manière particulière (par un carré) par rapport aux autres concepts. Elle est considérée comme le concept courant. Tous les autres concepts qui n'ont aucun lien avec le concept n'apparaissent pas. A partir du concept courant, on peut ajouter / supprimer de nouveaux attributs et ainsi progresser vers un nouveau concept courant. L'objectif de cet outil est de rendre les liens existants entre les différents attributs d'un ensemble de données, compréhensibles et intuitifs pour l'utilisateur.

LEGAL et LEGAL-E

Le système LEGAL a été développé par Liquière et Mephu Nguifo [LMN90] et son extension LEGAL-E par Mephu Nguifo [MN93]. Ces deux systèmes sont basés sur la construction d'un *sup-demi treillis* (treillis tronqué à partir d'un concept donné). La description du système est faite pour un problème à deux classes. Les objets sont alors séparés en deux sous-ensembles : les exemples et les contre-exemples. La différence entre LEGAL et LEGAL-E se situe au niveau de l'apprentissage. En effet, LEGAL utilise tous les objets de l'apprentissage (exemples et contre-exemples) pour construire le treillis alors que LEGAL-E ne conserve qu'une seule classe (celle des exemples).

Apprentissage Un sup-demi treillis est tout d'abord construit sur un critère de pertinence. Seuls les concepts pertinents sont insérés dans le treillis et serviront à la reconnaissance. Les critères de pertinence proposés sont construits à partir des propriétés suivantes :

Validité : un concept est *valide* s'il contient au moins α exemples.

Quasi-cohérence : un concept est *quasi-cohérent* s'il est valide et contient au plus β contre-exemples.

Maximalité : un concept valide et quasi-cohérent est *maximal* s'il n'est couvert par aucun autre concept valide et quasi-cohérent.

Pertinence 1 : concepts valides, quasi-cohérents et maximaux

Pertinence 2 : concepts valides et quasi-cohérents

LEGAL utilise le critère de pertinence 1, et LEGAL-E est testé pour les deux variantes de critères de pertinence (LEGAL-E1 et LEGAL-E2). Les seuils de validité et de quasi-cohérence sont fixés à : $\alpha = 2$ et $\beta = 1$.

Classification Pour la classification, les auteurs utilisent un vote majoritaire. Si l'objet à reconnaître vérifie un nombre suffisant de concepts pertinents (seuil de justification suffisamment élevé), il reçoit la classe majoritaire parmi ces concepts.

Expérimentation Les systèmes sont testés expérimentalement sur un jeu de données issu de la biologie moléculaire GenBank 64.1 et comparés à des méthodes d'apprentissage symbolique (ID3 [Qui86], COBWEB [Fis87]) et neuronal (KBANN [TS92], BACKPROP [RHW86], PEBLS [CS93], PERCEPTRON [Ros58]). En terme de résultats, LEGAL-E est aussi efficace que les méthodes neuronales et plus performant que les méthodes symboliques. De plus, lorsque le seuil de justification est bas, LEGAL-E2 est plus efficace, et lorsqu'il augmente, c'est LEGAL-E1 qui se comporte le mieux. L'auteur pense que "les résultats de LEGAL-E sont probablement imputables à l'exhaustivité du treillis, mais aussi au fait que le treillis de Galois peut être assimilable à l'architecture simple et dynamique d'un réseau de neurones". Cette approche possède tout de même un inconvénient : un sous-ensemble d'objets de l'apprentissage ayant des attributs très similaires, ou des objets de l'apprentissage isolés, vérifieront très peu de concepts pertinents et ne pourront donc pas être reconnus.

GALOIS

Le système GALOIS, développé par Carpineto et Romano [CR93], propose une construction incrémentale du treillis et deux modes d'apprentissage : découverte de classe et prédiction de classe. Dans tous les cas, il y a une recherche de *concepts pertinents* qui vont permettre de caractériser les classes. L'intérêt de cette construction incrémentale du treillis vient du fait qu'elle est invariante à l'ordre d'insertion des objets (contrairement aux constructions incrémentales de hiérarchies de concepts).

Apprentissage Le mode découverte de classe, correspond à un apprentissage supervisé. Les objets sont insérés dans le treillis en tenant compte des valeurs de chaque attribut et de la classe. Plus on insère d'objets dans le treillis, plus la description des classes est précise et plus le partitionnement des objets tend vers le partitionnement naturel des classes. En revanche, dans le mode prédiction de classe, les objets sont insérés sans tenir compte de leur classe. Les concepts regroupant différents objets sont alors formés, sans certitude d'obtenir au final au moins un concept représentant chaque classe, c'est-à-dire un concept dont tous les objets appartiennent à une même classe.

Classification Il faut ensuite sélectionner les concepts pertinents (concepts cohérents et maximalement complets) du treillis. En d'autres termes, un concept est pertinent

pour un objet donné, si les attributs de l'objet sont égaux ou inclus dans les attributs du concept. Deux types de classification d'un nouvel objet sont proposés :

- Le concept pertinent obtenant la similarité la plus élevée affecte sa classe à l'objet. La similarité étant donnée par le nombre d'attributs du concept partagés par l'objet.
- L'objet reçoit la classe majoritaire parmi tous les concepts pertinents.

Expérimentation L'expérimentation consiste à tester GALOIS sur quatre jeux de données : Soybean, Congressional Voting, Breast Cancer et Iris [BKM98]. Le mode découverte de classes permet de caractériser les données de l'apprentissage selon la convergence (ou non) des concepts pertinents vers un partitionnement naturel des classes. Lorsqu'il n'y a pas cette convergence, cela dénote un certain chevauchement entre les classes. Quant au mode prédiction de classes, il propose des résultats de classification intéressants pour ces quatre jeux de données.

RULEARNER

Sahami a développé la méthode RULEARNER [Sah95] où la classification est basée sur l'utilisation de *règles de classification* extraites à partir des concepts du treillis de Galois.

Apprentissage Pour un contexte donné, le treillis de Galois est construit en appliquant l'algorithme GRAND. On distingue dans le treillis trois types de nœuds :

- nœud d'instance : concept obtenu directement à partir d'un objet du contexte.

- nœud d'attribut : concept obtenu directement à partir d'un attribut du contexte.

- nœud interne : concept situé entre les nœuds d'instance et les nœuds d'attribut.

Ensuite, chaque concept du treillis va se voir attribuer un label en commençant par les nœuds d'instance. En effet, ils sont simples à labelliser étant donné qu'ils reçoivent le label de la classe correspondant à l'objet du concept. Ensuite, les nœuds internes sont labellisés : soit par un label de classe, si une classe est suffisamment représentée (seuil de pureté), soit par le label MIXTE si aucune classe n'est majoritaire. Parmi tous les nœuds labellisés par une classe, on recherche celui qui contient un nombre minimal d'attributs et un nombre maximal d'objets. Une règle de classification est extraite à partir du nœud sélectionné (attributs du nœud -> label de classe du nœud). Ensuite, ce nœud est inactivé ainsi que tous ses nœuds successeurs dans le treillis. Du fait de ces modifications, il est possible que d'autres nœuds du treillis changent de label. Avec cette nouvelle règle créée, on est en mesure de classer une partie des objets de l'apprentissage. Il reste donc à reproduire le même procédé (recherche du nœud labellisé, extraction de

la règle, mise à jour des labels) jusqu'à ce qu'on puisse attribuer une classe à l'ensemble des objets, plus précisément lorsque qu'il ne reste plus de nœud pouvant être labellisé par une classe dans le treillis.

Classification La classification d'un nouvel objet est réalisée : soit en appliquant la première règle vérifiée par l'objet (attributs de l'objet inclus dans les attributs de l'hypothèse de la règle) parmi une liste des règles triées, soit en attribuant à l'objet la classe majoritaire parmi les règles (non triées) qu'il vérifie.

Expérimentation L'auteur présente des résultats expérimentaux obtenus avec RULEARNER sur les jeux de données MONK'S et BREAST CANCER, et une comparaison avec les classifieurs CN2 [CN89] et C4.5 [Qui93]. Plusieurs configurations des trois classifieurs ont été testées : liste de règles triées ou non triées pour CN2 et RULEARNER, élagage ou non pour C4.5 et différentes tolérances au bruit pour les trois. Il en résulte que les taux de précision du système RULEARNER sont relativement compétitifs par rapport aux autres classifieurs. Plus exactement, il obtient le second taux de reconnaissance derrière la version élaguée de C4.5. Ces tests prouvent qu'il est adapté à la reconnaissance d'objets bruités et qu'un élagage au sein des règles pourrait être envisagé pour améliorer la classification.

CIBLe

La méthode de classification CIBLe [NMN99], développée par Njiwoua et Mephu Nguifo, est basée sur l'induction de concepts. Elle prend en compte à la fois des attributs symboliques et numériques. A partir d'un contexte initial, un *sup-demi treillis* est tout d'abord construit pour effectuer une sélection des *concepts pertinents*. Le but est d'obtenir une *redescription du contexte* en fonction des attributs issus des concepts pertinents. Pour toute la démarche de sélection des concepts pertinents, et de redescription des données, seuls les attributs symboliques sont utilisés. Après la redescription, les attributs numériques sont ajoutés au contexte obtenu.

Apprentissage Au départ, il faut donc construire un sup-demi treillis dans lequel les concepts pertinents vont être recherchés. La pertinence est donnée par une fonction Laplacien à maximiser. Il s'agit d'obtenir des concepts où une classe est bien représentée, en d'autres termes, la classe doit contenir beaucoup d'objets par rapport aux autres classes (pureté du concept). Chaque attribut qui apparaît dans au moins un concept pertinent est un attribut pertinent et il sera conservé pour la redescription. A partir

des attributs pertinents, tous les objets du contexte initial sont ensuite redécrits. On compte le nombre d'occurences de chaque paire (objet, attribut pertinent) dans les concepts pertinents et cette valeur est insérée dans le nouveau contexte. Les attributs numériques sont ensuite ajoutés au contexte redécrit. Une sélection est alors effectuée sur les objets pour éviter les redondances et les incohérences au sein du contexte. Les objets sélectionnés sont appelés prototypes.

Classification Pour la classification d'un nouvel objet, il est nécessaire de calculer sa redescription pour le comparer aux prototypes du nouveau contexte. La classe affectée à l'objet sera la classe majoritaire parmi les k plus proches prototypes du contexte redécrit.

Expérimentation L'expérimentation propose une comparaison de CIBLe avec d'autres classifieurs tels que : IBi (IB1, IB2 et IB3) [AKA91], K^* [CT95] et Pebls [CS93] sur les bases de données de l'UCI Machine Learning Repository [BKM98]. Les résultats obtenus montre une supériorité en terme taux de prédiction de l'algorithme CIBLe par rapport aux autres méthodes.

CLNN et CLNB

Les méthodes CLNN et CLNB, développées par Xie et al. [XHLL02], utilisent les treillis des concepts pour transformer le problème de classification en sous-problèmes plus facilement résolvables. Chaque concept représente un sous-problème à résoudre. Un classifieur dédié à la classification de ce sous-problème apprend à reconnaître les objets du concept en fonction des attributs qui les décrivent. Pour la méthode CLNN, le classifieur en question est le plus proche voisin (NN pour Nearest Neighbor), et pour CLNB, il s'agit du classifieur bayésien naïf (NB pour Naive Bayes).

Apprentissage A partir des concepts, un ensemble de *règles contextuelles* sont construites. Une règle contextuelle permet en fonction d'un ensemble d'attributs connu d'appliquer le classifieur adapté au sous-problème. Une règle contextuelle doit vérifier trois types de contraintes afin de limiter les redondances :

- être vérifiée par un nombre minimal d'objets ; avoir un nombre d'objets vérifiant la règle suffisamment élevé par rapport au nombre d'objets vérifiant la règle contextuelle issue du concept successeur dans le treillis.
- avoir une précision suffisamment éloignée de celle de la règle contextuelle issue du concept successeur dans le treillis.

– avoir un nombre d'attributs en hypothèse de la règle suffisamment élevé par rapport au nombre d'attributs en hypothèse de la règle contextuelle issue du concept successeur dans le treillis.

Un test du χ^2 est ensuite appliqué sur l'ensemble des règles contextuelles pour conserver les règles de meilleure précision.

Classification La classification d'un nouvel objet est réalisée en appliquant les règles contextuelles dont les attributs en hypothèse correspondent aux attributs de l'objet. Chaque règle renvoie le résultat de classification donné par son classifieur. Un vote majoritaire est alors appliqué sur l'ensemble des résultats retournés. En cas d'égalité, la règle la plus précise permet de trancher.

Expérimentation Dans l'expérimentation, CLNB et CLNN sont comparés à plusieurs classifieurs, ceux sur lesquels ils sont basés : NB et NN, et également le NBTree [Koh96], le CBA [LHM98] et le C4.5Rules [Qui93] sur 26 jeux de données de l'UCI Machine Learning Repository [MM96]. Les méthodes CLNB et CLNN améliorent les résultats de classification de leurs méthodes de base respectives, et CLNB devance même les autres classifeurs NBTree, CBA et C4.5Rules.

Zenou et Samuelides

La méthode implémentée par Zenou et Samuelides [ZS04, SZ04] concerne l'orientation d'un robot dans un environnement structuré comprenant plusieurs pièces. Le robot dispose d'un ensemble d'images décrivant chaque pièce.

Apprentissage A partir des images sont extraites des caractéristiques visuelles, c'est-à-dire des éléments permettant de distinguer chacune des pièces. Ces caractéristiques sont des *amers visuels* qui servent de point de repère pour le robot. Parmi toutes les caractéristiques envisageables (structurelles, colorimétriques ou photogrammétriques), il est nécessaire d'effectuer une sélection pour ne conserver que les plus fréquentes. L'ensemble des images et de leurs caractéristiques sélectionnées sont organisées au sein d'un *treillis de Galois*. Le graphe contient alors de manière structurée toutes les correspondances entre caractéristiques visuelles et les pièces étudiées. Ensuite, les auteurs *sélectionnent des concepts* contenant des images provenant toutes d'une même pièce. Les concepts successeurs et prédécesseurs de ces concepts sélectionnés et qui vérifient des critères de pureté sont également utilisés pour *extraire les amers visuels*.

Classification Il est alors possible pour le robot, si on lui présente une nouvelle image d'une des pièces, d'extraire les amers visuels de cette image pour rechercher de quelle pièce il s'agit. Pour proposer au robot de nouvelles images, les photographies des pièces sont obtenues selon différents point de vue. Les signatures doivent donc respecter des contraintes d'invariance à la translation, à la rotation et au changement d'échelle et être robustes au bruit. Lorsque la classification n'est pas possible (aucune correspondance avec les amers visuels de l'apprentissage), le robot peut se déplacer, prendre de nouvelles photographies et retenter une nouvelle reconnaissance de la pièce.

Expérimentation Les auteurs précisent que cette approche donne de bons résultats sur de petites bases de données (moins de 25 images par caractéristiques), mais que la complexité algorithmique de certains algorithmes ne permet pas de traiter de plus grandes bases de cette manière. Pour passer outre ce problème, les auteurs proposent une solution où chaque pièce est caractérisée par son propre treillis de Galois. Les résultats obtenus avec cette nouvelle approche sont identiques à ceux obtenus avec l'ancienne méthode, mais la complexité est nettement moindre.

Eklund

Nous avons trouvé dans la littérature une méthode d'indexation concernant la fouille de données. Bien que le contexte d'application de cette méthode est différent du nôtre (fouille de données au lieu de la classification et utilisation données non bruitées), l'auteur a porté un intérêt particulier à l'utilisation du treillis de Galois pour la structuration intéressante qu'il offre entre les objets (généralisation ou spécialisation selon que l'on progresse vers le haut ou le bas du graphe).

Il s'agit de l'étude menée par Eklund [Ekl06] concernant l'utilisation du treillis de Galois pour l'indexation d'images MPEG-7. Ces images ont la particularité de pouvoir contenir des descriptions telles que la forme des images, leur couleur et luminance, ou encore un contenu plus sémantique. De la même manière que dans l'étude précédente, ces descriptions sont utilisées en tant que caractéristiques de chaque image et sont organisées de manière structurée dans le treillis de Galois. Le treillis de Galois obtenu peut ensuite être utilisé de deux manières différentes :

- soit en entrée directe, l'utilisateur doit alors introduire un ensemble de caractéristiques et il obtiendra en réponse le concept du treillis le plus spécifique contenant tous les attributs demandés.
- soit pour une simple exploration, par validation / suppression de caractéristiques. Lorsque l'utilisateur valide une nouvelle caractéristique, il demande une description plus précise des images, ce qui conduit à une réduction du nombre d'images

correspondant à cette description (spécialisation). Au contraire, lorsque l'utilisateur supprime une caractéristique, la description des images sera moins précise et correspondra donc à un nombre plus important d'images (généralisation).

Pour réduire la complexité, l'auteur propose d'utiliser un treillis par type de caractéristiques, par exemple, un treillis pour la couleur, un autre pour la forme, ... avec des liaisons entre les treillis pour permettre un basculement d'un treillis vers un autre.

Cet outil a plus pour vocation de permettre une navigation intuitive et rapide au sein d'une base d'images que de réaliser une classification des images. C'est pourquoi nous n'avons pas intégré cette méthode dans l'étude comparative qui suit (voir § 1.3.3).

B - Comparaison des méthodes

Les tableaux 1.9 et 1.10 proposent une synthèse détaillée des méthodes en indiquant le type d'apprentissage proposé (S=Supervisé, NS=Non Supervisé), les constructions obtenues, le procédé de classification, la complexité et les résultats expérimentaux obtenus par les auteurs des méthodes.

La figure 1.6 permet de situer les méthodes de classification, présentées précédemment. Toutes comportent au moins une étape de sélection sur les attributs et / ou les objets. Elles peuvent être regroupées selon qu'elles utilisent seulement les concepts, les concepts et les règles de classification, une double sélection, ou simplement des règles contextuelles.

Le fait d'utiliser une construction du treillis incrémentale est intéressant pour la réduction du temps de traitement. Ainsi, pour prendre en compte un nouvel objet, il n'est pas nécessaire de tout reconstruire le treillis. En revanche, un algorithme de construction du treillis incrémental n'apporte rien de plus pour la classification.

Les descriptions des méthodes de classification basée sur les treillis de Galois montrent qu'elles n'utilisent pas réellement le treillis comme classifieur pour reconnaître les objets. Le principe est de sélectionner des concepts du treillis pertinents ou des règles pertinentes extraites du treillis afin d'attribuer une classe à l'objet, le plus souvent, par un vote majoritaire.

A partir de cette étude de la littérature, il semble donc que l'exploration du treillis de Galois soit plus employée pour la fouille de données qu'en classification. C'est un moyen intuitif pour l'utilisateur d'observer les dépendances entre les attributs de la table de données.

TAB. 1.9 – Propriétés des méthodes de classification basées sur les treillis de Galois

Méthodes	Apprentissage	Construction	Classification
GRAND	S et NS	Treillis Sélection des concepts les plus généraux de chaque classe Extraction des règles de classification	Vote majoritaire parmi les règles les plus spécifiques pour l'objet
LEGAL	S	Sup-demi treillis contenant les concepts pertinents	Vote majoritaire parmi les concepts pertinents
GALOIS	S et NS	Treillis Sélection des concepts pertinents	Concept le plus similaire pour l'objet ou vote majoritaire
RULE-ARNER	S	Treillis Sélection des concepts pertinents Extraction des règles de classification	Application de la première règle vérifiée par l'objet à classer ou vote majoritaire parmi les règles vérifiées
CIBLe	S	Sup-demi treillis Sélection des concepts pertinents Redescription du contexte Sélection de prototypes	Vote majoritaire parmi les plus proches voisins
CLNN et CLNB	S	Treillis Extraction des règles contextuelles Sélection des règles pertinentes	Vote majoritaire parmi les classes retournées par le classifieur NN ou NB des règles contextuelles vérifiées par l'objet
Zenou et Samuelides	S	Treillis Concepts pertinents d'où sont extraits des amers visuels (intentions des concepts)	Classement si tous les amers visuels vérifiés appartiennent à une même pièce

1.3.4 Classification basée sur les règles d'association

Dans cette partie, nous décrivons sept méthodes de classification basées sur les règles d'association (voir § 1.3.4-A) et présentons une comparaison théorique (voir § 1.3.4-B).

TAB. 1.10 – Propriétés des méthodes de classification basées sur les treillis de Galois (suite)

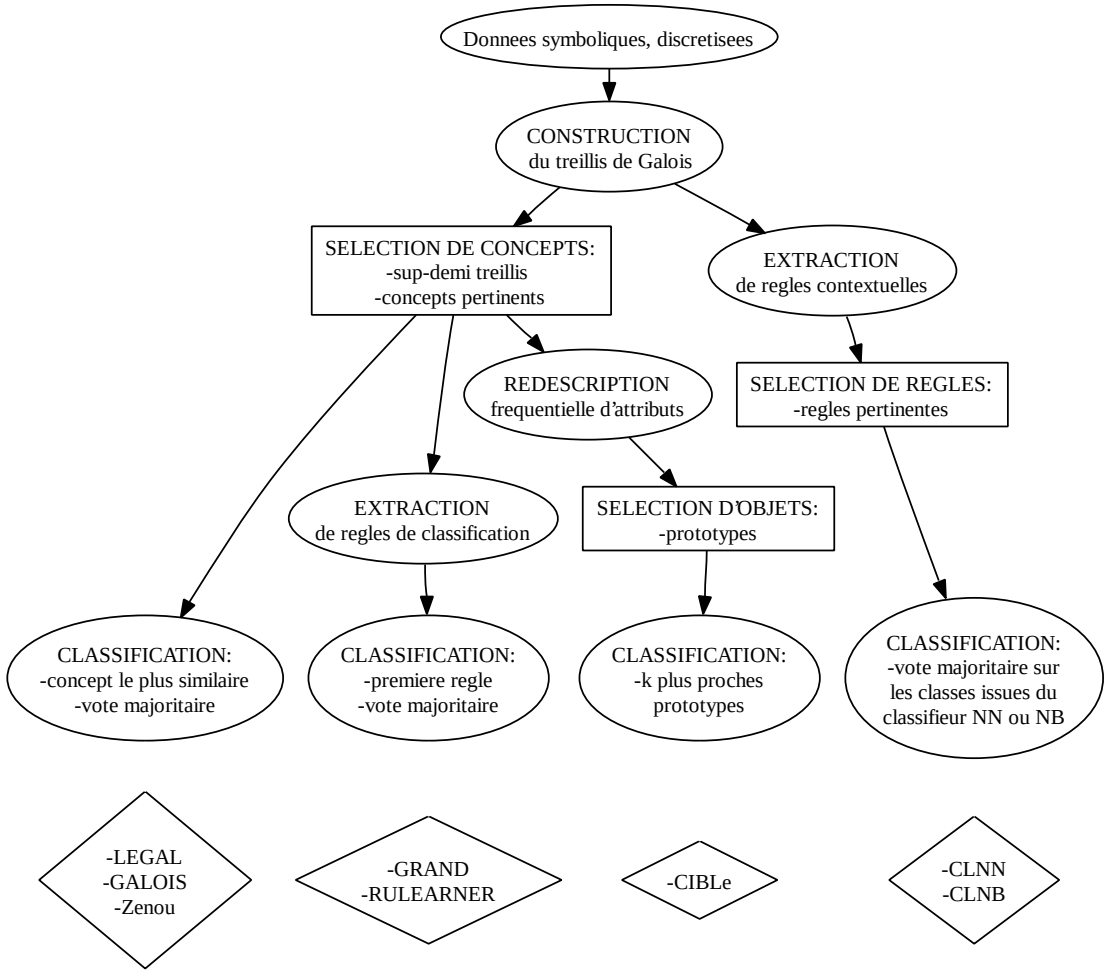
Méthodes	Complexité	Comparaison expérimentale
GRAND	$O(2^n n^4)$ avec n minimum entre le nombre d'exemples et le nombre d'attributs	Taux comparables aux autres méthodes : Assistant, AQ15, AQR, Bayes et CN2
LEGAL	$O((\|L\|n(1-\alpha))$ avec $\|L\|$ le nombre de concepts du treillis	Taux comparables à GLUE et supérieurs C4.5
GALOIS	$O(3^m 2^m n) < O(3^{2m} n)$ avec m le nombre d'attributs et n le nombre d'exemples	Taux comparables aux méthodes de la littérature
RULE-ARNER	$O(2^n n^4)$ avec n minimum entre le nombre d'exemples et le nombre d'attributs	Taux comparables à C4.5 et CN2, voire légèrement supérieurs
CIBLe	$O(min(n-1, m-1)^{h+1} m^3)$ construction du sup-demi treillis + $O(min(n-1, m-1)^{h+1})$ recherche du seuil	Taux supérieurs aux autres méthodes :IBi, K^* et Pebls
CLNN et CLNB	$O(\|L\|\|E\|^3 + \|L\|m + \|L'\|) + O(\|L'\|m).O(NN/NB)$ avec $\|E\|$ le nombre de fils d'un nœud $\|L'\| < \|L\|$ et $O(NN/NB)$ les complexités des classifieurs NN ou NB	Taux en moyenne supérieurs à ceux de NBTree, CBA et C4.5Rules
Zenou et Samuelides	$O(3^m 2^m n + \|L\|^2 m + \|L\|m)$ avec n le nombre d'exemples et m le nombre d'attributs	Taux très encourageants (18 photographies reconnues sur 20)

A - Description des méthodes

CBA

L'algorithme CBA [LHM98], implémenté par Liu, Hsu et Ma, est l'un des premiers algorithmes utilisant les règles d'association pour la classification. La génération de l'ensemble des règles de classification est basée sur l'algorithme Apriori [AS94]. Un petit ensemble de ces règles est sélectionné à l'aide de l'ensemble d'apprentissage et

FIG. 1.6 – Points communs / différences des méthodes



constitue le classifieur. Au sein du classifieur, les règles sont triées selon une notion de précédence (notée $\prec$) basée sur les valeurs de confiance (et en cas d'égalité, de support) des règles. Le classement d'un nouvel objet est donné par la première règle du classifieur qui le vérifie.

Apprentissage Plus précisément, la première étape de la méthode CBA concernant la génération des règles est réalisée à partir d'une adaptation d'Apriori aux cas des règles de classification. Les règles sont construites de manière croissante : à une étape k donnée, les règles candidates sont construites à partir des règles obtenues à l'étape $k-1$. Parmi les règles candidates, on recherche les règles ayant un support supérieur à *minsup*. Un élagage basé sur le calcul d'un taux d'erreur [Qui93] permet également de supprimer certaines règles. Quant à elle, la construction de l'ensemble final des règles

pertinentes est réalisée en trois étapes :

- Recherche pour chaque objet de l'ensemble d'apprentissage de la règle de classification ayant la meilleure précédence qui classe correctement l'objet (notée *cRule*) ainsi que la règle de classification de meilleure précédence qui classe mal l'objet (notée *wRule*). Si *cRule* a une meilleure précédence que *wRule*, l'objet sera correctement classé, sinon il faut faire des ajustements.
- Recherche de chacune des règles *wRule* ayant une précédence supérieure à la règle *cRule* correspondante. Si une règle *wRule* d'un objet est à la fois une règle *cRule* pour un autre objet, on recherche des règles pour la remplacer.
- Les règles *cRule* et les règles de remplacement sont triées par ordre de précédence. La construction du classifieur se fait alors en ajoutant une à une les règles triées et en calculant l'erreur de classification sur les objets de l'ensemble d'apprentissage. Tant que l'ajout d'une nouvelle règle permet de faire baisser l'erreur de classification, on poursuit les insertions. Dès qu'on observe une augmentation de l'erreur de classification, le processus est terminé. Toutes les règles restantes situées après la règle courante ne seront pas utilisées dans le classifieur.

Classification Un nouvel objet se voit attribuer un label de classe par la première règle vérifiée par l'objet, dans l'ensemble des règles classées par ordre de précédence.

Expérimentation Elle est réalisée sur 26 jeux de données de l'UCI ML Repository [MM96] et offre une comparaison avec la méthode C4.5 (arbre et règles). Les taux d'erreur moyens obtenus en validation croisée montrent une plus grande efficacité de CBA par rapport à C4.5.

CMAR

Li, Han et Pei ont développé la méthode CMAR [LHP01] qui base son extraction des règles de classification de la base d'apprentissage sur l'algorithme FP-growth. Selon les auteurs, cet algorithme de fouille de motifs fréquents, est plus rapide que les méthodes du type Apriori [AS94] pour de grands ensembles de données, des supports faibles, et/ou des motifs de taille importantes.

Apprentissage Les règles de classification générées selon l'algorithme FP-growth sont stockées dans une structure CR-tree. Cette structure d'arbre est compacte et indexée. Elle permet de réduire la taille de stockage des règles et de les retrouver efficacement. Ainsi l'élagage et l'utilisation des règles pour la classification sont facilités. L'élagage des règles de classification est réalisé si une règle ne vérifie pas l'une de ces trois conditions :

- une confiance minimale supérieure à $minconf$.
- une corrélation minimale entre les règles basée sur une métrique du χ^2.
- un seuil minimal de couverture de la base d'apprentissage d'au moins un objet. C'est-à-dire qu'au moins un objet de la base doit vérifier la règle.

Les règles de classification obtenues après élagage sont regroupées pour constituer le classifieur.

Classification La classification s'effectue ensuite de la manière suivante : toutes les règles du classifieur vérifiant l'objet à classer sont considérées. Si elles impliquent le même label de classe, ce label est attribué à l'objet. Sinon, les règles sont regroupées selon le label de classe qu'elles impliquent. Pour chaque regroupement, on calcule l'effet combiné de l'ensemble des règles d'après une mesure pondérée du χ^2. Le groupe obtenant un effet maximal attribue son label à l'objet.

Expérimentation La méthode CMAR est comparée aux méthodes C4.5 et CBA sur les 26 jeux de données de l'UCI ML Repository [MM96]. CMAR obtient la meilleure précision sur 13 jeux de données. Par rapport à CBA, CMAR permet un gain d'espace mémoire d'environ 77% sur les 6 plus grands jeux de données étudiés dans cette expérimentation. Le seuil minimal de couverture de la base d'apprentissage et le seuil de confiance sont les deux paramètres essentiels pour la précision de classification de CMAR. Ces seuils contrôlent le nombre de règles insérées dans le classifieur. Si ce nombre est trop faible, il risque de manquer des règles performantes. Au contraire, si le nombre est trop important, il y a risque de surapprentissage.

ARC-AC et ARC-BC

Les méthodes ARC-AC et ARC-BC [AZ02b, AZ02a], développées par Antonie et Zaiane en 2002, sont dédiées à la catégorisation de document textuel.

Apprentissage Elles utilisent l'algorithme Apriori [AS94] pour générer l'ensemble des règles de classification de support suffisant. La différence entre les deux méthodes est que ARC-AC génère des règles de classification pour toutes les catégories (i.e. toutes les classes) de texte prises dans leur ensemble, alors que ARC-BC les génère catégorie par catégorie. Les règles sont ensuite triées des plus générales au plus spécifiques. Plus précisément, pour deux règles $R1 : T1 \rightarrow c$ et $R2 : T2 \rightarrow c$, $R1$ est plus générale que $R2$ si $T1 \subseteq T2$. La sélection des règles est alors effectuée de la manière suivante : pour chaque règle de classification, on recherche les règles les plus spécifiques correspondantes

et on élimine celles de confiance faible (notion de précédence définie précédemment dans la méthode CBA). Parmi les règles résultantes de ce premier élagage, on conserve celles qui sont vérifiées par au moins un objet de la base d'apprentissage.

Classification Les règles de classification obtenues constituent le classifieur. Celles ayant le même attribut de classe en conclusion sont regroupées dans un même ensemble. On calcule ensuite la confiance moyenne par ensemble et on attribue la classe qui obtient la meilleure confiance moyenne.

Expérimentation L'expérimentation est réalisée sur la base Reuters-21578 [LI87] qui regroupe une collection de documents textuels. Les auteurs ont choisi de travailler sur les 10 catégories de texte les plus populaires. Une comparaison est réalisée avec les méthodes Bayes, Rocchio, C4.5, k-plus proches voisins, bigrams, SVM. Les résultats proposés montrent que ARC-BC est plus efficace que la plupart des méthodes excepté les deux méthodes k-plus proches voisins et SVM. De plus, les phases d'apprentissage et de reconnaissance sont relativement rapides. Une expérience de comparaison sur les types d'élagage (sans élagage, avec élagage des règles spécifiques, et avec élagage des règles spécifiques et vérification d'au moins un objet de la base d'apprentissage) indique que les résultats de reconnaissance ne sont pas améliorés en appliquant un élagage.

CPAR

Xiaoxin Yin a développé le système CPAR [XY03] qui reprend la stratégie de l'algorithme FOIL [QCJ93] pour la génération des règles. Contrairement aux trois méthodes (CBA, ARC-AC et ARC-BC, CMAR) présentées précédemment, où l'on construit l'ensemble de règles pour ensuite sélectionner les plus pertinentes, cette méthode utilise une heuristique pour générer directement les règles de classification pertinentes.

Apprentissage Cette heuristique est basée sur une fonction de gain à maximiser. Pour un ensemble d'objets appartenant à la même classe, on attribue un poids équivalent à chacun d'entre eux. Ensuite, on construit une règle en ajoutant petit à petit les attributs qui maximisent la fonction de gain. Tous les objets qui vérifient cette nouvelle règle créée voient leur poids diminuer. Il faut ensuite construire de la même manière une nouvelle règle pour les objets restants (ceux dont le poids n'a pas diminué), et ainsi de suite jusqu'à ce que tous les objets soient au moins vérifiés par l'une des règles de classification créées. Ainsi, plusieurs règles proches de la meilleure règle peuvent être construites.

Classification Les règles obtenues sont ensuite réparties en paquet selon la classe qu'elles contiennent en conclusion. De plus, elles sont triées en fonction de leur précision (formule de l'estimation de l'erreur attendue de Laplace). On calcule la précision moyenne des k meilleures règles au sein de chaque paquet, et le paquet obtenant la meilleure précision moyenne attribue sa classe à l'objet.

Expérimentation Les résultats expérimentaux montrent que CPAR obtient une meilleure classification que CBA, CMAR, C4.5 et RIPPER sur 9 des 26 jeux de données de l'UCI ML Repository [MM96] et permet de diminuer de manière conséquente les temps d'exécution et le nombre de règles générées par rapport à CBA et CMAR.

HARMONY

Wang et Karypis, les auteurs d'HARMONY [WK05], proposent une extraction de règles de classification dont le point de vue est centré sur les objets. En d'autres termes, ils cherchent pour chaque objet de la base d'apprentissage à extraire au moins une des règles couvrant un grand nombre d'objets et de confiance la plus élevée.

Apprentissage Pour cela, les motifs fréquents sont recherchés et ordonnés non pas par ordre de support décroissant, mais selon l'un des trois ordres définis ci-dessous :

- par ordre de confiance décroissant
- par ordre d'entropie croissant
- par ordre de coefficient de corrélation croissant

Le dernier ordre est utilisé par défaut dans le système HARMONY.

Une fois, les motifs fréquents triés, plusieurs élagages sont mis en place :

- Suppression des motifs ayant un support équivalent au motif m.
- Suppression des attributs non prometteurs. Les règles issues du motif fréquent auquel on ajoute un attribut non prometteur ne peuvent pas avoir une confiance supérieure à la règle de meilleure confiance du motif courant.
- Suppression des motifs non prometteurs. Les règles issues d'un motif fréquent non prometteur auquel on ajoute un attribut ne peuvent pas avoir une confiance supérieure à la règle de meilleure confiance du motif courant.

Les motifs résultant de ces élagages sont utilisés pour la génération des règles de classification.

Classification L'ensemble de ces règles constitue le classifieur. Elles sont regroupées par paquet en fonction de l'attribut de classe qu'elles contiennent en conclusion, et triées

au sein de chaque paquet ainsi obtenu selon leur précédence basée sur la confiance (définie dans la méthode CBA décrite précédemment). Pour la reconnaissance, trois modes sont proposés pour attribuer une classe à l'objet à reconnaître :

- la classe de la meilleure règle vérifiée (précédence)
- la classe du paquet où les k meilleures règles (précédence) obtiennent le meilleur score
- la classe du paquet obtenant le meilleur score

Le score est en fait la somme des confiances des règles au sein du paquet. Le dernier mode est celui utilisé par défaut pour les expérimentations.

Expérimentation HARMONY est comparé à FOIL, CPAR et SVM sur 10 jeux de données de l'UCI ML Repository [MM96] de taille importante. Les résultats montrent que cette méthode est la plus efficace en terme de génération des règles et de classification dans 7 cas sur 10.

Barbu et al.

Dans leurs travaux [BHAT05], Barbu et al. proposent une approche non-supervisée de recherche de symboles dans les images de documents. Cette méthode est intéressante car elle présente une utilisation des règles d'association dans le contexte de détection de symboles bruités fréquents. Il ne s'agit pas de reconnaissance, mais bien de recherche de symboles fréquents et des relations fréquentes entre ces symboles. Un symbole est considéré comme une représentation graphique comportant de la connaissance. Les images de documents sont décrites par des graphes relationnels attribués. Cette méthode utilise la recherche de sous-graphes fréquents pour obtenir une représentation graphique et également les règles d'association pour retrouver la connaissance relative au symbole.

Apprentissage Pour réduire la complexité en temps de la recherche de sous-graphes fréquents, les auteurs utilisent un réseau de graphes non-isomorphes calculé "hors-ligne". Ce réseau est un graphe orienté acyclique, où les nœuds sont des graphes non-isomorphes et les arcs représentent l'inclusion entre ces graphes. A partir d'un graphe non-isomorphe contenant un seul nœud, on ajoute des nœuds un à un. Si le nouveau graphe obtenu avec l'ajout du nœud est fréquent (c'est-à-dire qu'il apparaît plus souvent que la valeur d'un seuil fixé), il est inséré dans le réseau. Sinon, il n'est pas nécessaire de poursuivre la croissance de ce graphe, car tous les graphes obtenus ne seront pas fréquents. Le réseau est tout de même limité en taille, car les graphes fréquents ne peuvent contenir plus de 9 arcs. L'algorithme utilise l'information contenue dans ce réseau de graphes non-isomorphes pour rechercher efficacement les sous-graphes fréquents. Cet algorithme

de recherche de sous-graphes fréquents permet la détection d'un certain nombre de symboles sur le document.

Classification Il est alors envisageable de rechercher les relations spatiales existantes entre ces différents symboles reconnus. Cette démarche est réalisée en calculant les règles d'association entre symboles à partir de l'algorithme Apriori [AS94], selon un support et une confiance minimaux. Les règles d'association obtenues sont du type : "La présence du symbole 1 dans telle partie du document implique la présence du symbole 2 dans cette même partie" ou "La présence du symbole 3 dans le document implique la présence des symboles 4, 5 et 6 dans ce même document".

Expérimentation Les résultats expérimentaux obtenus sur des images de documents architecturaux montrent que les symboles fréquents sont détectés et des règles d'association intéressantes peuvent être extraites. En revanche, les résultats obtenus sur des images de fax contenant des logos, des tables, de l'écriture manuscrite et imprimée montrent que la recherche des sous-graphes fréquents est plus délicate en présence de bruit. En effet, seuls quelques logos ont été détectés comme fréquents. Dans ces conditions de documents à contenu varié, la représentation graphique ne semble pas appropriée.

GARCm

La toute récente méthode GARCm [BEBY06], implémentée par Bouzouita, Elloumi et Ben Yahia, est basée sur la classification associative. L'originalité de cette méthode est l'extraction de règles de classification directement à partir d'une base générique. Le but est d'obtenir un petit ensemble de règles de meilleure qualité et contenant moins de redondances que les méthodes habituelles de classification associative, et ce sans perte d'information.

Apprentissage Les auteurs utilisent la base générique $\mathcal{IGB}$ [GBYMNS06] car elle est réputée pour être informative et relativement compacte. Les règles générées par la base $\mathcal{IGB}$ ont tendance à avoir des prémisses contenant peu d'éléments. Les auteurs précisent que c'est un avantage car les règles peuvent être ainsi plus facilement vérifiées par les objets. Seules les règles de classification extraites de cette base sont conservées, et parmi elles, seules les règles vérifiées par au moins un objet de l'apprentissage sont intégrées au classifieur.

Classification Un classifieur associatif est un ensemble de règles de classification qui doit prédire de manière la plus précise possible une classe à attribuer au nouvel objet. Contrairement aux autres méthodes, pour la construction d'un nouveau classifieur, les règles de classification ne sont pas triées. Plusieurs mesures d'évaluation de la qualité des règles extraites sont comparées (intérêt, surprise, Loevinger, score). Pour le classement d'un nouvel objet, on recherche toutes les règles qu'il vérifie. Si elles correspondent toutes à un même attribut de classe, ce dernier est assigné à l'objet, sinon, la règle qui obtient le meilleur indice de qualité attribue sa classe à l'objet.

Expérimentation D'après les résultats expérimentaux réalisés sur 12 jeux de données [BKM98], l'indice de qualité basé sur le score permet d'obtenir les meilleurs taux de classification. De plus, GARCm est plus efficace que les classifieurs traditionnels ID3 et C4.5 sur 6 jeux de données et que les approches de classification associative CBA et Harmony sur 9 jeux de données.

B - Comparaison des méthodes

Les tableaux 1.11 et 1.12 présentent une synthèse comparative des méthodes de classification basées sur les règles d'association détaillées précédemment. Ces tableaux proposent de comparer la génération des règles, la présence d'un élagage et / ou d'un tri des règles et le mode de classification.

Les méthodes CBA, CMAR, ARC-AC, ARC-BC, Barbu et al. et GARCm sont exhaustives car elles génèrent l'ensemble des règles vérifiant un certain support et sélectionnent ensuite parmi ces règles celles qui vérifient certaines propriétés. Les méthodes CPAR, et HARMONY utilisent des heuristiques pour ne générer que certaines règles. Elles ne peuvent donc pas garantir une génération des règles optimales.

Pour accélérer les traitements, des sélections sont appliquées sur les règles afin de ne conserver que les plus pertinentes : génération de règles vérifiant certaines heuristiques, support et confiance minimaux, taux d'erreur, corrélation positive, couverture de l'ensemble d'apprentissage. De plus, les règles sont souvent triées par précédence et par classe avant d'effectuer une classification par la ou les meilleure(s) règle(s) selon un critère donné.

TAB. 1.11 – Propriétés des méthodes de classification basées sur les règles d'association

Méthodes	Génération des règles	Elagage des règles
CBA	Basée sur Apriori Support minimal Exhaustive	Oui, par le taux d'erreur pessimiste
CMAR	Basée sur FP-growth Support et confiance minimaux Exhaustive	Oui, par précédence, par corrélation positive et par couverture de l'ensemble d'apprentissage
ARC-AC et ARC-BC	Basée sur Apriori Support minimal Exhaustive	Oui, par précédence des règles spécifiques et par couverture de l'ensemble d'apprentissage
CPAR	Basée sur FOIL et une heuristique de gain minimal Support minimal Non exhaustive	Non, sélection réalisée en amont sur les motifs
HARMONY	Tri des motifs fréquents, méthode diviser pour régner et recherche en profondeur Support minimal, élagage des motifs de support équivalents, élagage des attributs et des motifs non prometteurs Non exhaustive	Non, sélection réalisée en amont sur les motifs
Barbu et al.	Basée sur Apriori Support et confiance minimaux Exhaustive	Oui, suppression des règles et méta-règles non vérifiées sur les documents d'apprentissage
GARCm	Base générique $\mathcal{IGB}$ Support et confiance minimaux Exhaustive	Oui, par couverture de l'ensemble d'apprentissage

1.4 Conclusion

Dans cet état de l'art, nous avons souhaité présenter un certain nombre de notions concernant la reconnaissance d'images et l'analyse formelle des concepts.

TAB. 1.12 – Propriétés des méthodes de classification basées sur les règles d'association (suite)

Méthodes	Tri des règles	Classification
CBA	Oui, par précédence et par erreur de classification de l'ensemble d'apprentissage	La première règle vérifiée par l'objet
CMAR	Oui, par classe	Ensemble de règles vérifiées par classe obtenant le meilleur χ^2 pondéré
ARC-AC et ARC-BC	Oui, par classe et par précédence	Ensemble de règles vérifiées par classe obtenant la meilleure confiance moyenne
CPAR	Oui, par classe et par estimation de l'erreur de Laplace (pouvoir de prédiction)	Ensemble des k premières règles vérifiées par classe obtenant la meilleure estimation de Laplace
HARMONY	Oui, par classe et par précédence	Ensemble de règles vérifiées par classe obtenant la meilleure confiance moyenne
Barbu et al.	Non	Détection des symboles fréquents et des relations fréquentes entre symboles
GARCm	Non	La règle vérifiée qui obtient le meilleur indice de qualité

Parmi toutes les méthodes de calcul de descripteurs de formes existantes, nous avons décrit quelques approches statistiques et structurelles. Les signatures obtenues à partir des approches présentées fournissent une description relativement convenable pour des ensembles d'apprentissage suffisamment significatifs. L'augmentation du nombre de classes et d'échantillons en apprentissage semble mieux amortie par les approches statistiques, en comparaison aux approches structurelles. Cependant, même pour ces approches, la montée en échelle reste un problème compliqué et ouvert. Pour leur part, les approches structurelles paraissent plus facilement adaptables à la reconnaissance de formes en contexte. Il semble désormais que l'effort de recherche se situe plus sur

l'utilisation conjointe de plusieurs approches, que sur le développement de nouveaux descripteurs. Dans le chapitre suivant, nous décrirons plus en détail les signatures employées pour la reconnaissance de symboles ainsi que leur paramétrage.

Nous avons ensuite expliqué le fonctionnement des principaux classifieurs utilisés en reconnaissance de formes. Au regard de la littérature, il semble que les machines à vecteur de support obtiennent généralement les résultats les plus intéressants.

S'agissant de la partie relative à l'analyse formelle des concepts, nous avons détaillé les fondements mathématiques concernant le treillis de Galois, ainsi que ses propriétés et sa construction. Enfin, quelques applications à la classification, de ce graphe et des règles d'association, ont été énoncées et comparées. Avec l'augmentation de la capacité de traitement des ordinateurs, l'intérêt pour la structure de treillis n'a cessé de croître comme en témoignent la quantité importante d'applications le concernant. Cet intérêt provient sans doute de l'organisation très intuitive des données au sein de ce graphe.

Chapitre 2

Signatures utilisées et paramétrage pour la reconnaissance de symboles

2.1 Introduction

Pour tester l'efficacité d'un système de reconnaissance, il est d'usage d'utiliser des bases de données. Dans ce chapitre, nous décrivons tout d'abord les bases de symboles GREC (§ 2.2) qui vont nous servir à évaluer la qualité de notre classifieur. Dans un second temps, nous présentons les différentes signatures statistiques (§ 2.3) et structurelles (§ 2.4) utilisées dans notre classifieur basé sur le treillis de Galois. Nous décrivons également le paramétrage de ces signatures mis en place pour la reconnaissance de symboles.

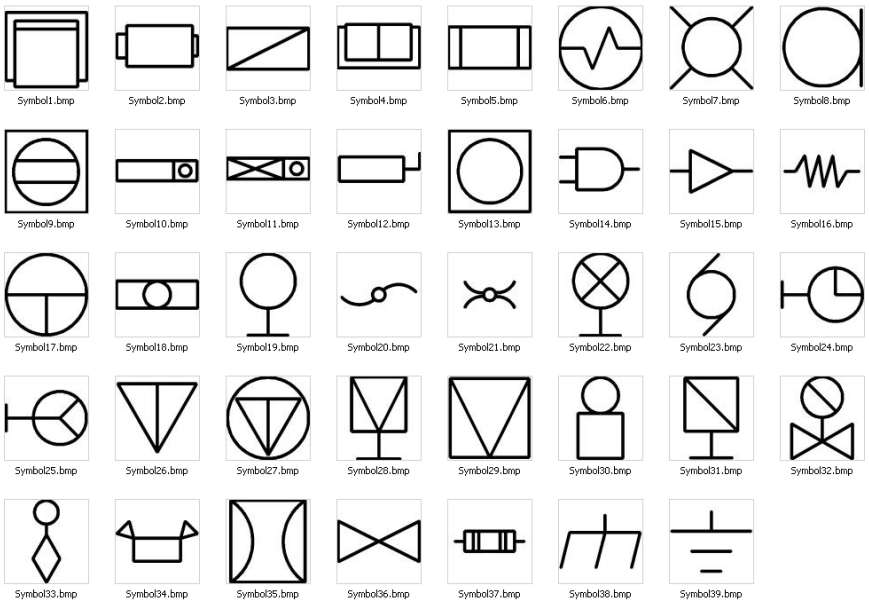
2.2 Description des bases GREC

2.2.1 La base GREC 2003

La base GREC 2003 [GRE03] (Graphics RECognition) est composée d'images de symboles segmentés, c'est-à-dire isolés. Elle a été créée pour un concours de reconnaissance de symboles organisé au cours de l'atelier Graphics Recognition qui a lieu tous les 2 ans. Il y a trois catégories de bruits proposées : les dégradations binaires (bruit impulsif), les distorsions vectorielles et les images réelles numérisées et photocopiées.

Concernant les dégradations binaires, la base contient 39 classes de symboles (voir Fig. 2.1) et 3510 symboles bruités. Chaque classe contient 1 symbole modèle (non bruité) et 90 symboles bruités. Les bruits appliqués sur les images reproduisent les déformations qui peuvent survenir lors de l'utilisation d'un photocopieur, d'un scanner ou encore d'une imprimante. Ils sont générés par la méthode Kanungo [Ka94] et sont composés de 9 types de dégradation (voir Fig. 2.2). Pour chaque type de dégradation, la base contient 10 symboles bruités par classe. Ces symboles sont également proposés en ayant subi une rotation ou un changement d'échelle.

FIG. 2.1 – Symboles modèles de la base GREC 2003 (dégradations binaires)



Pour les distorsions vectorielles, la base contient 15 classes de symboles architecturaux ou électriques (voir Fig. 2.3) et 600 symboles distordus. Les distorsions sont de trois types (voir Fig. 2.4). Ces symboles sont également proposés en ayant subi une rotation, un changement d'échelle, ou en étant doublement bruités par dégradations binaires et par distorsions vectorielles.

Enfin, les images réelles sont composées des 15 symboles architecturaux et électriques présentés précédemment et de 75 symboles bruités par numérisation et 75 symboles bruités par numérisation et photocopie.

FIG. 2.2 – Symboles bruités de la base GREC 2003 (dégradations binaires)

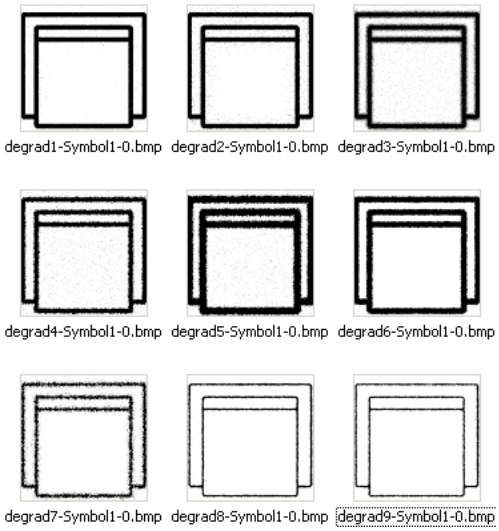
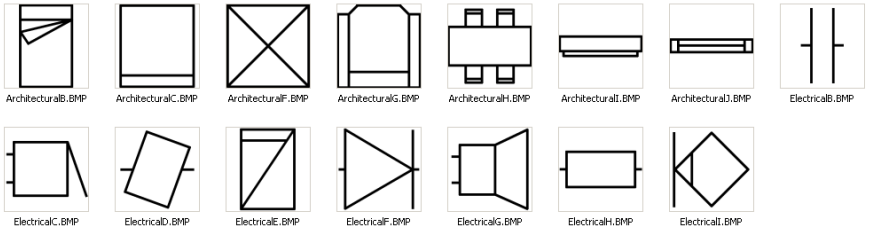


FIG. 2.3 – Symboles modèles de la base GREC 2003 (distorsions vectorielles)



2.2.2 La base GREC 2005

La base GREC 2005 [GRE05] est également issue du concours de reconnaissance de symboles (voir Fig. 2.5). Elle compte 150 symboles modèles au total et propose plusieurs tailles d'ensembles d'apprentissage et d'ensembles de test (25, 50, 100, 150 symboles à apprendre et respectivement 25, 50, 100, 100 symboles à tester). Les bruits appliqués sur les images sont également générés par la méthode Kanungo [Ka94] et sont composés de 6 types de dégradation (voir Fig. 2.6). Notons que les bruits appliqués sont nettement plus forts que dans la base GREC 2003, le dernier symbole correspondant au 6ème type de dégradation est quasiment effacé.

FIG. 2.4 – Symboles bruités de la base GREC 2003 (distorsions vectorielles)

FIG. 2.5 – Quelques symboles modèles de la base GREC 2005

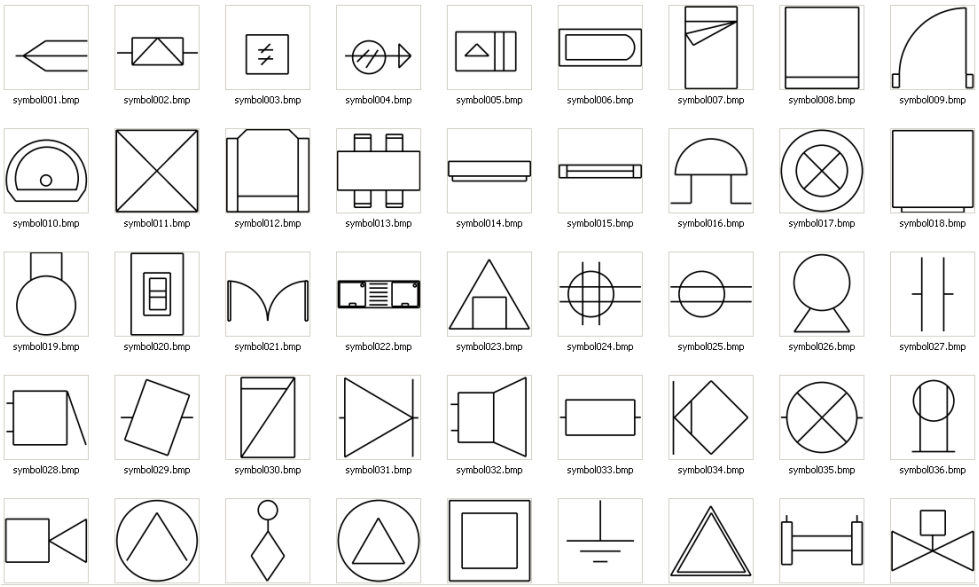
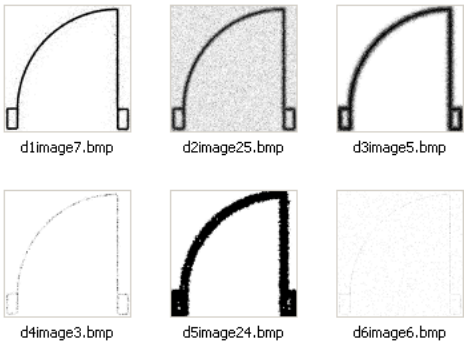


FIG. 2.6 – Symboles bruités de la base GREC 2005



2.3 Signatures statistiques

Nous avons choisi d'utiliser comme signature statistique, les trois signatures suivantes :

- la signature de Fourier-Mellin (voir § 1.2.1)
- la signature de Radon (voir § 1.2.1)
- la signature de Zernike (voir § 1.2.1)

Précisons que le choix de ces signatures n'engage à rien ; nous aurions très bien pu en utiliser d'autres. Toutefois, l'expérience des nombreuses utilisations de ces signatures permet une bonne appropriation de leur paramétrage (voir § 1.2.1). De plus, elles font référence en reconnaissance de formes et permettent d'obtenir de bons taux de classification en comparaison avec d'autres signatures.

Nous présentons les paramétrages utilisés pour chacune de ces trois signatures.

Signature de Fourier-Mellin

Rappelons que cette signature est paramétrable suivant deux termes p et q relatifs à l'invariance à l'orientation et à l'échelle. p est un coefficient radial et q un coefficient circulaire. Dans notre cas, nous avons fixé $p \in [0;2]$ si $q = 0$ et $p \in [-2;2]$ si $q \neq 0$ et $q \in [0;3]$, ce qui permet d'obtenir 33 valeurs numériques.

Signature de Radon

La R-signature de Radon comporte un seul paramètre : θ. Nous avons choisi de le faire varier pour obtenir 50 mesures de la R-signature, ce qui nous semblait suffisant pour caractériser les symboles. C'est-à-dire que l'on calcule la R-signature pour 50 orientations différentes, soit environ tous les 7°, et nous obtenons donc une signature comportant 50 valeurs numériques.

Signature de Zernike

Les moments de Zernike sont définis par deux paramètres : p et q qui fixent l'ordre du moment calculé. Nous calculons les moments de Zernike à l'ordre 7, ce qui nous permet d'obtenir une signature composée de 24 valeurs numériques.

2.4 Signature structurelle

Nous avons développé notre propre signature structurelle à partir de l'état de l'art présenté en partie 1.2.2. La mise en œuvre de cette signature a été effectuée en deux phases : l'extraction de primitives graphiques (segments) a été développée avec l'aide de Simon Bernard [Ber06] en 2006, puis la construction de la signature à partir de ces primitives a été implémentée par Mickaël Coustaty [Cou07] en 2007. Ces développements ont été réalisés au cours de leur stage de recherche de Master.

Dans le tableau de synthèse 1.2, nous avons vu que l'extraction basée sur les transformées de Hough/Radon semble offrir le meilleur compromis de par sa représentation sémantiquement riche et ses performances en terme de robustesse et d'invariance. Au regard des dégradations constatées sur les bases GREC, cette approche à base de transformée de Hough, semble capable d'absorber tous les types de bruit susceptibles d'être rencontrés. En effet, le caractère global de sa représentation lui permet de réduire l'effet du bruit. Nous avons donc choisi de baser notre extraction de primitives graphiques sur la transformée de Hough.

2.4.1 Extraction des segments par transformée de Hough

Dans cette partie, nous décrivons le principe de la transformée de Hough, les difficultés d'implémentation rencontrées et nous donnons une évaluation de la qualité de l'extraction des segments obtenus.

La transformée de Hough

En 1962, Paul Hough a défini la transformée qui porte son nom [Hou62] pour détecter d'éventuelles lignes droites dans des images en noir et blanc. Elle s'étend facilement à des formes génériques, et à des cadres de traitement d'images en niveaux de gris.

L'idée de base est que l'on peut transformer un ensemble de formes géométriques modèles d'un espace 2-dimensions, vers un espace paramétrique dans lequel elles peuvent être représentées de façon plus compacte. Le but étant de simplifier le problème de détection de formes remarquables, en le ramenant à un simple problème de détection de maxima locaux d'une fonction de cet espace de paramètres. Cette transformation s'appuie sur les représentations implicites des formes. L'ensemble des points x apparte-

nant à une même forme, identifiée par son vecteur de paramètres $p = (\alpha_1, \alpha_2, \ldots, \alpha_N)$, peuvent être définis par l'équation :

$$\mathcal{C}(x; (\alpha_1, \alpha_2, \ldots, \alpha_N)) = 0 \tag{2.1}$$

Pour un cercle de rayon r et de centre x_0, le vecteur de paramètres est : $p = (r, x_0)$ et l'ensemble des contraintes sera donné par : $\mathcal{C}(x; (r, x_0)) = \|x - x_0\| - r = 0$.

S'appuyant sur cette définition, Hough va construire pour chaque point $x = (x_1, x_2)$ d'une image, une hyper-surface dans l'espace des paramètres de dimensions N, correspondant à l'ensemble des formes pour lesquelles le point x respecte la définition 2.1. Autrement dit, l'hyper-surface est définie par l'ensemble des vecteurs p qui vérifient l'équation 2.1 pour un point x fixé. C'est ensuite les intersections de toutes les hypersurfaces correspondant à tous les points de l'espace image, qui vont permettre d'identifier les vecteurs p des formes effectivement présentes dans l'image.

Algorithmiquement le fonctionnement de la Transformée de Hough repose essentiellement sur l'utilisation d'un accumulateur, c'est-à-dire une discrétisation de l'intégralité de l'espace de projection. En pratique, cet accumulateur est un tableau à N dimensions où chaque élément correspond à un vecteur de coordonnées p, et donc à un membre de la famille des formes références. Plus précisément, chaque élément contient une valeur qui correspond au nombre d'occurrences de la forme concernée pour le vecteur de paramètres p dans l'image traitée.

Pour remplir l'accumulateur, il faut parcourir l'ensemble des points (pixels) de l'image et pour chacun d'eux, déterminer quels sont les vecteurs p pour lesquels le point appartient à la forme recherchée. Ainsi, pour chaque vecteur x nous devons déterminer quels sont les vecteurs p pour lesquels l'équation 2.1 est vérifiée. Dans le cas d'une image en noir et blanc, il s'agit simplement de traiter les pixels dits de forme, et d'incrémenter d'une unité chaque cellule de l'accumulateur vérifiant les contraintes. Dans le cas d'une image en niveau de gris, tous les pixels sont traités et la valeur des cellules de l'accumulateur est augmentée s'il y a lieu de la valeur du niveau de gris. De cette manière, si une forme de paramètres p est présente dans l'image, tous les pixels qui la composent participeront à son vote, formant ainsi un pic dans les valeurs de l'accumulateur.

Implémentation

Nous avons implémenté la transformée de Hough pour la détection de droites sur des images en noir et blanc. Cette implémentation a révélé un certain nombre de difficultés.

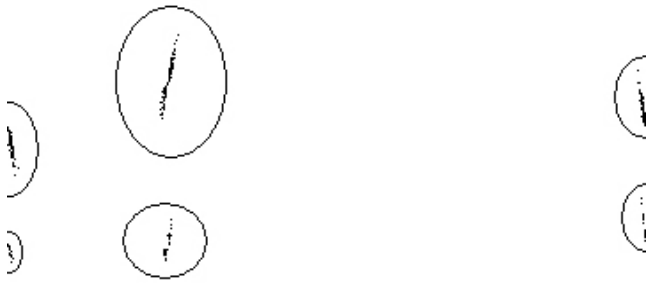
Tout d'abord, la complexité algorithmique qui dépend à la fois de la taille de l'image traitée et de la précision de la discrétisation de l'espace de Hough. Selon la précision de la discrétisation recherchée, les traitements peuvent être plus ou moins longs. Il s'agit d'obtenir le meilleur compromis entre performances et temps de traitement.

Nous avons également dû faire face à la complexité en espace mémoire pour stocker l'accumulateur. Cette complexité dépend aussi de la précision de la discrétisation de l'espace de Hough et du nombre de dimensions de l'espace. En effet, le nombre de paramètres de la forme recherchée (2 pour les droites, 3 pour les cercles, ...) correspond au nombre de dimensions de l'accumulateur.

Des pré-traitements des images ont été nécessaires pour réduire la complexité de la recherche : par exemple, une détection des contours ou une squelettisation permettent de réduire le nombre de pixels à considérer sur l'image. Dans notre traitement, nous avons utilisé une squelettisation.

Pour détecter les maxima locaux de l'accumulateur, un simple seuillage ne suffit pas, car la squelettisation n'étant pas parfaite, elle entraîne des déformations des droites. Au lieu d'obtenir une droite idéale, on obtient un petit groupe de droites au sein duquel une seule droite doit être sélectionnée. D'autre part, un groupe de droites peut être scindé en deux au sein de l'accumulateur (voir Fig. 2.7) car ce dernier est circulaire sur l'axe θ (angle 0° = angle 360°). Le développement d'une fonction d'étiquetage a permis de regrouper sous une même étiquette les droites appartenant à un même groupe. Puis, le calcul des centres de gravité de ces groupes de droites, a permis de sélectionner une unique droite représentante par groupe.

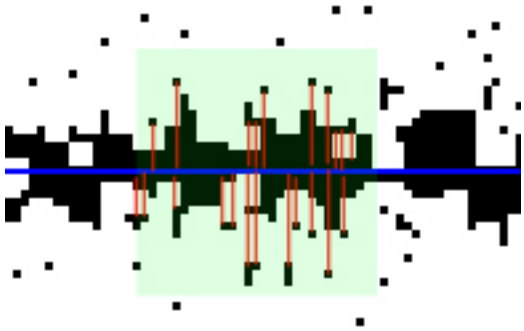
FIG. 2.7 – Représentation graphique d'un accumulateur contenant des groupes de droites (entourés d'un cercle). Certains groupes sont scindés en deux.



Enfin, l'exploitation des données obtenues nécessite des post-traitements. En effet, la

transformée de Hough permet de détecter des droites, or dans notre cas, nous souhaitons détecter des segments. Pour ce faire, à partir d'une droite obtenue, nous recherchons les pixels de l'image d'origine qui correspondent au trait, et plus particulièrement, nous essayons de les distinguer des pixels de bruit. Ce traitement se traduit par un parcours de la droite et une recherche dans son voisinage des pixels appartenant au trait. Sur l'image d'origine, les traits étants déformés, nous projetons les pixels du voisinage le long de la droite pour les aligner (voir Fig. 2.8), si la distance orthogonale des pixels à la droite reste faible (pas de bruit). On obtient alors des droites qui peuvent présenter des morcellements. Les extrémités de chaque morceau de trait sont ensuite détectées, et nous supprimons les morcellements si l'espacement entre les deux traits n'est pas trop important. Au final, nous obtenons donc les segments représentant le squelette du symbole.

FIG. 2.8 – Projection des pixels du voisinage le long de la droite (en bleu)



Evaluation expérimentale de l'extraction

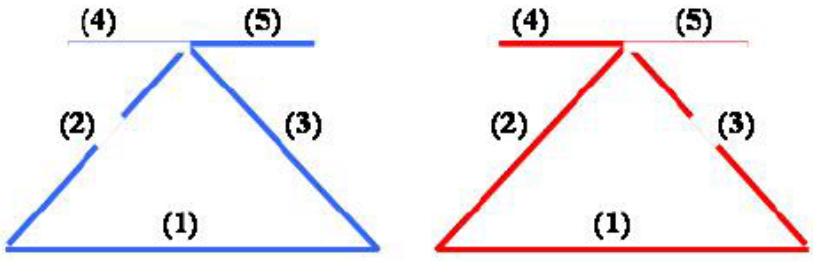
Dans les parties précédentes, nous avons expliqué la manière dont les segments du symbole ont été extraits à partir de la transformée de Hough. Nous passons maintenant à l'évaluation de la qualité de cette extraction.

L'évaluation est basée sur les *pourcentages d'associations* entre les symboles traités et les modèles auxquels ils correspondent effectivement. Ces associations sont effectuées à partir de distances de mise en correspondances (distances de matching) entre les segments. Le modèle que l'on attribue au symbole traité est celui pour lequel la distance en question est minimale.

La *distance de matching* permet d'évaluer la ressemblance entre l'ensemble des segments extraits du symbole et ceux extraits d'un modèle. De cette façon on peut dé-

terminer à quel modèle le symbole traité ressemble le plus. Plus précisément, il s'agit d'apparier les segments, en recherchant pour chaque segment extrait sur le symbole à reconnaître, son segment correspondant dans le symbole modèle (voir Fig. 2.9).

FIG. 2.9 – Distance de matching entre le symbole modèle (à gauche) et le symbole bruité avec les segments extraits (à droite). Plusieurs cas d'appariement sont représentés : (1) one-to-one : un segment modèle correspond à un segment extrait, (2) many-to-one : plusieurs segments modèles correspondent à un segment extrait, (3) one-to-many : un segment modèle correspond à plusieurs segments extraits, (4) fausse alarme : un segment extrait ne correspond à aucun segment modèle, (5) oubli : un segment modèle ne correspond à aucun segment.



Les tests ont été effectués sur les 15 symboles architecturaux (A1 à A7) et électriques (E1 à E8) de la base GREC 2003 (voir § 2.2.1). La figure 2.10 présente les pourcentages d'association (histogramme) et la distance de matching (courbe) pour chaque classe de symbole. Quant à elle, la figure 2.11 propose les mêmes informations mais en fonction des types de dégradation appliqués aux symboles. Nous constatons d'une part, que les pourcentages d'association sont très bons (97,9% en moyenne). Les segments extraits par la transformée de Hough dans des conditions de forte dégradation semblent donc suffisamment proches des segments idéaux pour leur être associés. D'autre part, les distances de matching ne sont pas totalement négligeables comme on aurait pu le penser (environ 0,09 pour le symbole A5) et elles varient d'une classe de symbole à l'autre. Une distance de matching élevée ne nuit pas toujours à une bonne association comme en atteste la classe de symbole A4 où le pourcentage d'association est de 100% alors que la distance de matching, supérieure à 0,07, fait partie des plus élevées. Néanmoins, on constate une certaine corrélation entre distance de matching élevée et baisse du pourcentage d'association sur la figure 2.11, notamment pour le type de dégradation 5.

Pour tenter d'expliquer la chute du pourcentage d'association pour le type de dégradation 5, nous avons observé l'épaisseur des traits des symboles selon le type de dégradation. Cette étude nous a permis de découvrir une corrélation entre l'épaisseur des traits et les pourcentages d'association. Le type de dégradation 5 correspond d'ailleurs à

FIG. 2.10 – Histogramme des pourcentages d'association et courbe de distance de matching en fonction des classes de symboles

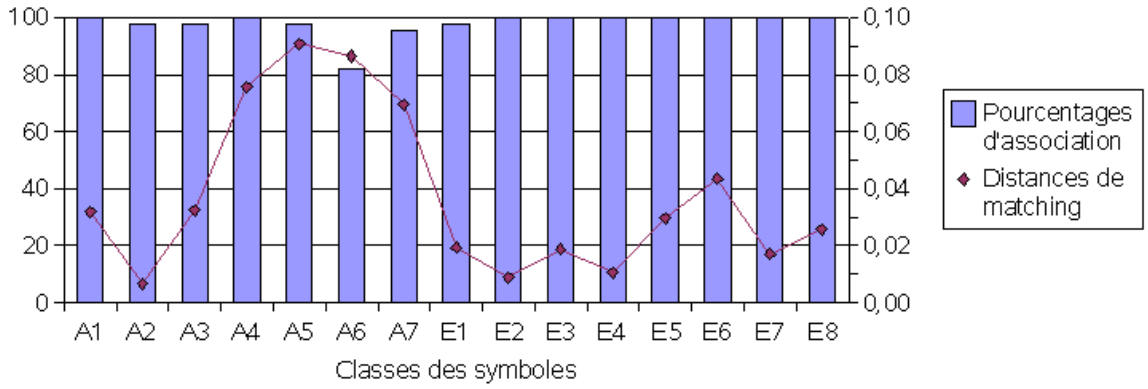
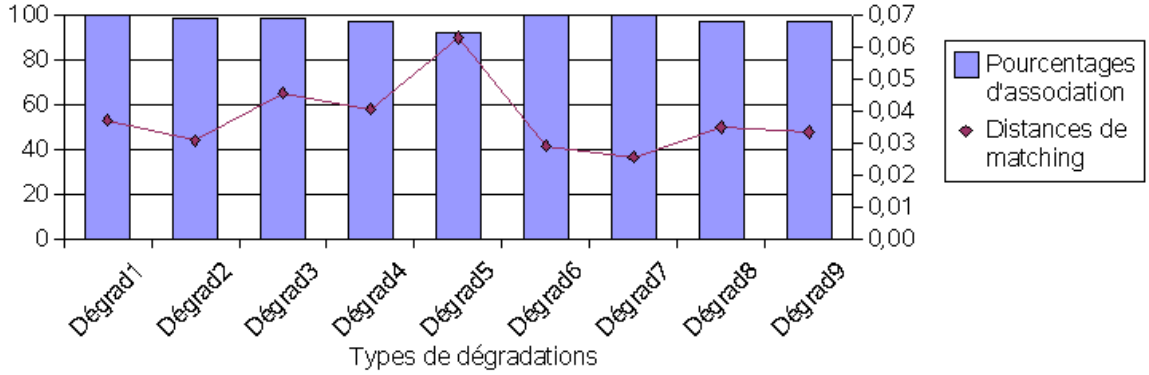


FIG. 2.11 – Histogramme des pourcentages d'association et courbe de distance de matching en fonction des types de dégradation appliqués aux symboles



une épaisseur maximale des traits (voir Fig. 2.2) et au taux d'association le plus faible. L'explication du faible taux provient sans doute des déformations générées lors de la squelettisation, qui sont d'autant plus importantes que les traits sont épais.

2.4.2 Construction de la signature structurelle à partir des segments extraits

A partir des segments extraits à l'aide de la transformée de Hough, nous souhaitions ensuite construire une signature tenant compte de l'organisation spatiale de ces segments. Cette signature devait être sous la forme d'une liste de valeurs numériques pour être ensuite introduite dans le classifieur basé sur le treillis de Galois. Parmi les

méthodes existantes de la littérature (voir § 1.2.2), nous avons choisi de nous inspirer de celle de Geibel et Wysotzki [GW96] qui allie deux avantages : elle est générique et elle est basée sur l'utilisation d'un graphe de vecteurs qui offre une description très haut niveau des symboles. Le développement s'est réalisé en deux étapes principales : tout d'abord la construction du graphe de vecteurs qui intégre les relations spatiales entre paire de segments, puis le calcul de la signature en elle-même. Après la description de ces deux étapes, nous présentons quelques éléments d'analyse statistique.

Construction du graphe de vecteurs

Nous avons dû adapter les descriptions des relations spatiales entre paires de segments au cas des symboles, en intégrant les contraintes d'invariance à la translation, la rotation et au changement d'échelle. En effet, les descriptions des symboles dans la méthode de Geibel et Wysotzki comportaient une orientation (comme par exemple avec la relation "au-dessus de"). Pour déterminer quelles relations spatiales utiliser pour décrire les paires de segments, nous avons réalisé un état de l'art, que nous rappelons ci-dessous.

Dans la littérature, on rencontre quatre catégories de relations spatiales :

Relations de contacts et angulaires Ce sont les plus couramment rencontrées dans la littérature : *Intersection*, *Colinéarité*, *Angle*, *Perpendicularité* et *Parallélisme*. Elles sont liées à l'utilisation de seuils de tolérance pour limiter les effets de la discrétisation des images.

Relations spatiales de projection Elles se composent :

- des *Jonctions de segments* : L, T, U, V, X, Y, λ, ... où la forme de la lettre décrit implicitement l'organisation spatiale de la paire de segments. Elles utilisent également des seuils de tolérance.
- du *Recouvrement* [DL04] : il est basé sur la projection des segments sur une ligne virtuelle.
- des *Projections* : les projetés obtenus selon un ou plusieurs axes peuvent être comparés suivant différents critères (pourcentage de recouvrement, inclusion, ...). Les projections ne sont pas invariantes à la rotation.

Relations métriques Pour les segments parallèles, la relation métrique associée est la *Distance relative* entre les segments ; et pour les segments non parallèles, il s'agit de la *Longueur relative*. Cette dernière est calculée soit en faisant le ratio entre les longueurs des deux segments, soit en faisant le ratio entre la longueur d'un segment et la taille de la fenêtre étudiée. Remarquons que ces relations métriques n'offrent pas une description suffisamment précise pour pouvoir distinguer un carré d'un rectangle.

Relations de sens Uniquement cité dans [WZY06], elles utilisent le sens et la direction d'un segment par rapport à un autre segment ou par rapport à un axe. Souvent obtenues par la "règle de la main droite" ou par la définition d'un axe de repère, ces relations donnent une information précise et complète, mais non invariante à la rotation.

Partant de cette étude de l'existant, nous avons construit une description des relations spatiales entre paire de segments à partir d'un triplet topologique :

$$< \textit{type de relation, valeur de la relation, longueur relative} >$$

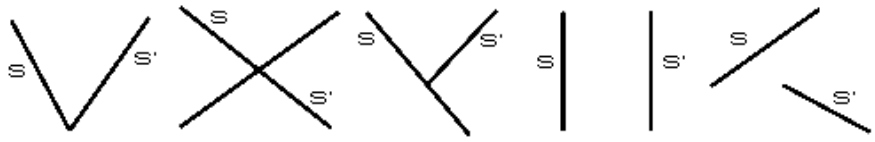
Plus précisément, chaque élément de ce triplet contribue à affiner la description de la relation entre les deux segments considérés :

Type de relation C'est un mixte entre les relations de contacts et angulaires et les jonctions de segments décrits précédemment. Nous avons sélectionné les relations P pour Parallélisme, V, X, et Y, auxquelles nous avons ajouté la relation O pour tous les autres cas (voir Fig. 2.12).

Valeur de la relation Il s'agit de la distance relative entre les deux segments, pour la relation P, et de l'angle formé entre les deux segments pour les relations V, X, Y et O.

Longueur relative entre les deux segments dans tous les cas.

FIG. 2.12 – Types de relation entre deux segments s et s' : V, X, Y, P et O (dans l'ordre de gauche à droite)

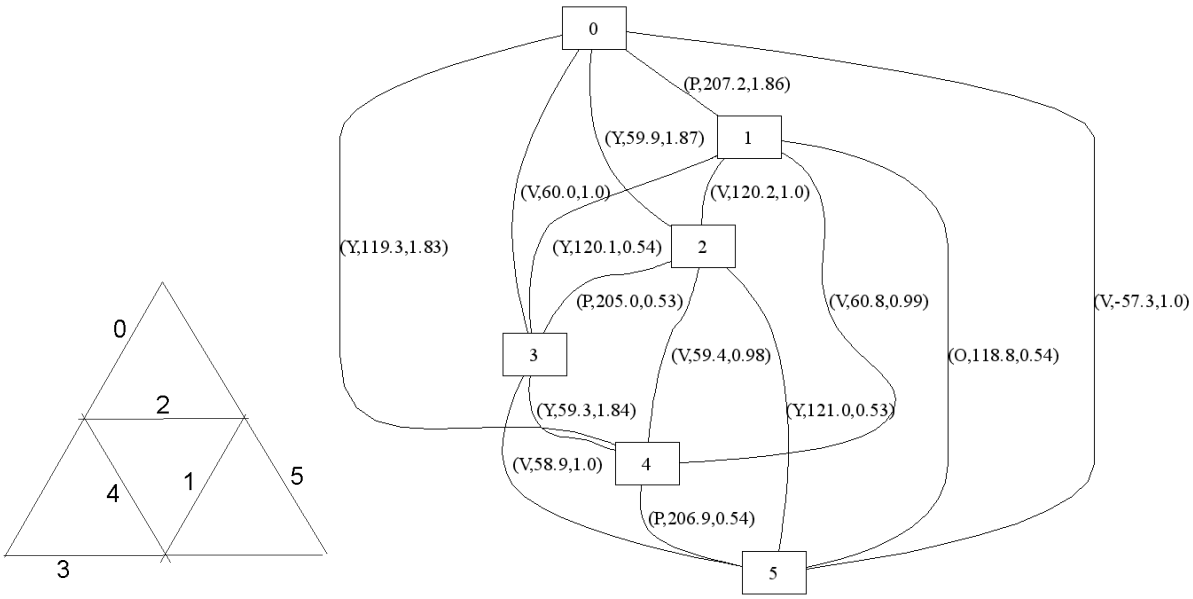


Les triplets topologiques sont calculés entre chaque paire de segments composant le symbole et sont stockés dans le graphe de vecteurs. Chaque nœud du graphe représente un segment du symbole et chaque arc du graphe est étiqueté par un triplet topologique (voir Fig. 2.13). La matrice d'adjacence du graphe de vecteur donne une représentation équivalente (voir Tab. 2.1). C'est une matrice symétrique.

Pour chaque symbole, on obtient un ensemble de triplets topologiques qui décrivent complètement l'organisation structurelle des segments. Cette description basée sur les triplets possède plusieurs avantages :

– Chaque paire de segments est décrite par un unique triplet topologique.

FIG. 2.13 – Segments extraits par transformée de Hough sur une image de symbole et le graphe de vecteurs associé



- Chaque symbole est caractérisé par un graphe unique et complet. La reconstruction des segments du symbole est possible à partir du graphe topologique.
- La description est invariante à la rotation et au changement d'échelle et aux distorsions vectorielles.

Cette représentation possède également quelques inconvénients :

- Elle ne prend pas en compte les arcs de cercle. Nous aurions pu au préalable essayer de détecter les cercles et arcs de cercle, mais cela générait une étape supplémentaire de décision lors de la reconnaissance de ce type de primitives. Cette étape aurait pu engendrer des erreurs (confusion) qui auraient pu porter un préjudice lourd à la reconnaissance. De ce fait, sans rejeter l'idée de la faire ultérieurement, nous avons opté pour l'exploitation de segments, afin d'expérimenter l'approche à base de treillis.
- Un grand nombre de triplets peuvent être nécessaires pour caractériser un symbole (au plus $(n-1)^2$ avec n le nombre de segments du symbole).

Pour restreindre le nombre de triplets calculés, il est possible d'effectuer une discrétisation des valeurs d'angle (par exemple, 30°, 45°, 60°, 90°), une discrétisation des distances entre segments parallèles (colinéaires, proches, un peu espacés, très éloignés), ou encore une discrétisation des distances relatives (égaux, légèrement différents, très différents).

TAB. 2.1 – Matrice d'adjacence associée au graphe de vecteurs

	0	1	2	3	4	5
0		P 207.2 1.86	Y 59.9 1.87	V 60 1	Y 119.3 1.83	V -57.3 1
1	P 207.2 1.86		V 120.2 1	Y 120.1 0.54	V 60.8 0.99	O 118.8 0.54
2	Y 59.9 1.87	V 120.2 1		P 205 0.53	V 59.4 0.98	Y 121 0.53
3	V 60 1	Y 120.1 0.54	P 205 0.53		Y 59.3 1.84	V 58.9 1
4	Y 119.3 1.83	V 60.8 0.99	V 59.4 0.98	Y 59.3 1.84		P 206.9 0.54
5	V -57.3 1	O 118.8 0.54	Y 121 0.53	V 58.9 1	P 206.9 0.54	

De plus, il est possible de réduire le triplet topologique en ne considérant que les types de relation, ou bien la paire $<$ ***type de relation, valeur de la relation*** $>$, ou encore seulement la paire $<$ ***type de relation, distance relative*** $>$.

Calcul de la signature

Le calcul de la signature mis en place est inspiré de la méthode de Geibel et Wysotzki [GW96]. Nous avons déjà précisé que cette méthode était générique et qu'elle offrait une représentation haut niveau des symboles. En effet, les agencements spatiaux entre les segments utilisés pour caractériser les symboles ne sont pas prédéfinis (généricité) et la taille des agencements est paramétrable (représentation haut niveau). Plus la taille des agencements considérés est importante, plus la description des symboles devient complexe et discriminante, mais dans le même temps, plus la robustesse aux distorsions est faible [RnL05]. Il est donc nécessaire de trouver le bon compromis entre niveau de description et robustesse.

Pour calculer les différents agencements possibles entre n segments, il faut simple-

ment élever la matrice d'adjacence décrite précédemment à la puissance n. En effet, en calculant la n^{ime} puissance d'une matrice d'adjacence, on calcule en fait les chemins de longueur n dans le graphe de vecteurs correspondant. Soit M une matrice d'adjacence, elle contient les chemins de longueur 1, M^2 contient les chemins de longueurs 2, et ainsi de suite ... Comme les matrices d'adjacence sur lesquelles nous travaillons ne sont pas booléennes 2.2, il est nécessaire d'étendre le produit de matrices booléennes à l'union de chaînes de caractères concaténées 2.3.

$$\forall(i,j) \in [0,L]^2, (A \times B)_{ij} = \sum_{k=1}^{L} (a_{ij} \times b_{ij}) \tag{2.2}$$

$$\forall(i,j) \in [0,L]^2, (A \times B)_{ij} = \bigcup_{k=1}^{L} (a_{ij} + b_{ij}) \tag{2.3}$$

où L est la taille de la matrice et $+$ est l'opérateur de concaténation de chaînes de caractères. Nous considérons ce produit seulement pour les chemins élémentaires, c'est-à-dire les chemins où tous les sommets sont distincts. De plus, les chemins symétriques sont regroupés. Par exemple, si on restreint le triplet topologique aux simples types de relation pour simplifier l'explication, les chemins POV et VOP étant symétriques, ils seront factorisés en 2*POV. Ainsi, en reprenant l'exemple de la matrice d'adjacence (voir Tab. 2.1) restreinte aux types de relation, on obtient au carré la matrice de la table 2.2.

TAB. 2.2 – Matrice d'adjacence élevée au carré

	0	1	2	3	4	5
0		1VO 3VY	2PV 2VY	2PY 1VV 1YY	2PV 2VY	1OP 1PY 1VV 1YY
1	1VO 3VY		1OY 2PY 1VV	1OV 2PV 1VY	1OP 1PY 1VV 1YY	2PV 2VY
2	2PV 2VY	1OY 2PY 1VV		4VY	2PY 1VV 1YY	1OV 2PV 1VY
3	2PY 1VV 1YY	1OV 2PV 1VY	4VY		2PV 2VY	1OY 2PY 1VV
4	2PV 2VY	1OP 1PY 1VV 1YY	2PY 1VV 1YY	2PV 2VY		1OV 3VY
5	1OP 1PY 1VV 1YY	2PV 2VY	1OV 2PV 1VY	1OY 2PY 1VV	1OV 3VY	

Nous calculons les puissances de matrices jusqu'à l'ordre 4 car pour les ordres supérieurs, les agencements étudiés peuvent être trop affectés par les bruits de distorsion.

Ainsi, en calculant plusieurs puissances d'une même matrice, nous obtenons un ensemble de chemins de longueurs différentes qui caractérisent le symbole. La dernière étape consiste à collecter ces chemins et compter le nombre de fois où ils sont présents pour un symbole donné. En d'autres termes, nous recherchons le nombre d'occurrences d'agencements particuliers au sein du symbole de manière générique, c'est-à-dire sans avoir pré-défini les agencements recherchés. Les matrices d'adjacence étant symétriques, on ne compte que les chemins présents dans la partie supérieure des matrices. La table 2.3 présente la signature structurelle obtenue à partir de la matrice d'adjacence élevée au carré présentée par la table 2.2.

TAB. 2.3 – Exemple de signature structurelle obtenue

Chemins	1VO 3VY	2PV 2VY	2PY 1VV 1YY	1OP 1PY 1VV 1YY	1OY 2PY 1VV	1OV 2PV 1VY	4VY
Occurrences	2	4	2	2	2	2	1

Eléments d'analyse statistique

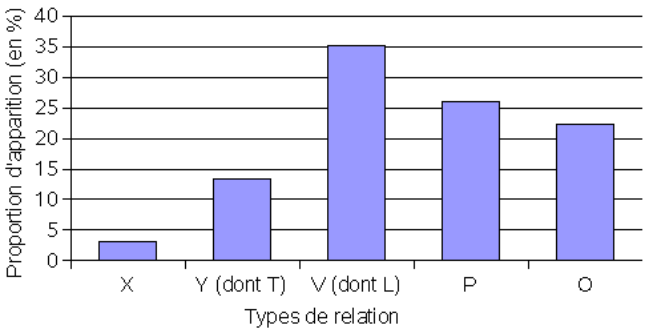
Nous avons réalisé quatre études sur les symboles de la base GREC 2003, présentée dans la partie 2.2.1. L'objectif des trois premières était de faire ressortir la fréquence d'apparition des types de relation, des angles et des différentes longueurs relatives. La quatrième étude concerne les chemins et notamment les tailles de signatures obtenues selon la longueur des chemins calculés et la fréquence d'apparition des chemins de longueur 1 et 2 dans les symboles.

A - Etude des types de relation

La fréquence d'apparition des différents types de relation n'est pas identique dans les symboles de la base GREC. La figure 2.14 donne leur répartition et nous renseigne sur la forte présence des relations V (jonction en forme de 'V'), P (parallèles) et O (autres). Les relations O englobent tous les segments sans intersection et non parallèles qui sont présents en très grand nombre dans les symboles. Elles permettent d'obtenir un graphe de vecteurs connexes même pour les symboles composés de plusieurs composantes connexes. Les signatures structurelles comportant des relations X et Y seront les plus discriminantes étant donné qu'on les trouve peu fréquemment dans les symboles.

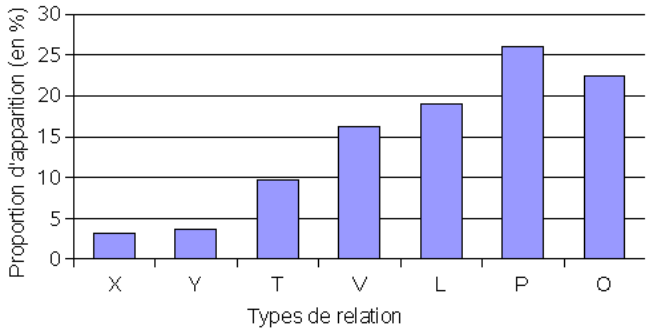
Dans la figure 2.15, nous avons ajouté des relations T et L, qui sont respectivement des relations Y à 90° et V à 90°, pour affiner la description des symboles. Ce graphique

FIG. 2.14 – Proportion d'apparition des types de relation X, Y, V, P et O



fait ressortir la présence conséquente des angles droits dans les symboles. Ce résultat ne semble pas aberrant puisque l'on rencontre un nombre important de carrés et rectangles dans les structures des symboles.

FIG. 2.15 – Proportion d'apparition des types de relation X, Y, T, V, L, P et O



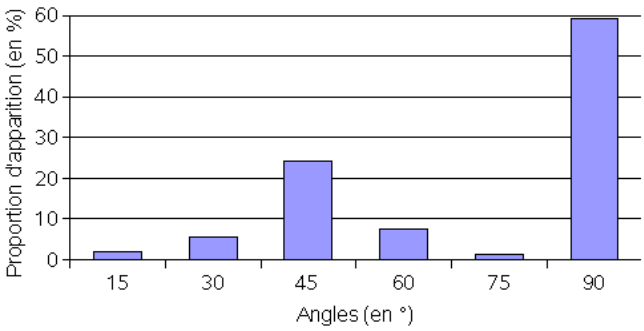
B - Etude des angles

De la même manière que pour l'étude de types de relation, nous avons recherché la fréquence d'apparition des angles. La figure 2.16 donne cette répartition et montre la présence prépondérante des angles droits. Les angles à 45° sont également assez présents. Pour restreindre le nombre de triplets topologiques possibles, nous avons appliqué une discrétisation des angles en fonction des résultats de cette étude. Nous avons ainsi choisi de conserver pour les types de relation V, X, Y et O les angles suivants : 30°, 45°, 60° et 90°.

C - Etude des longueurs relatives

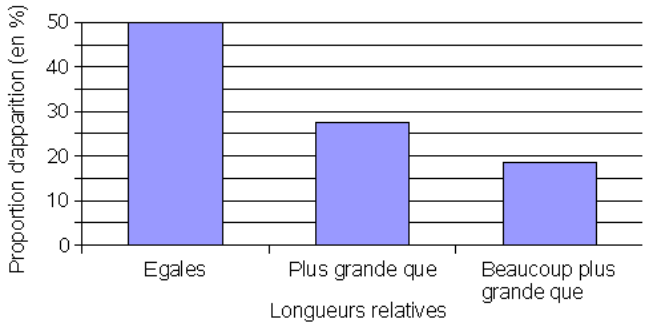
L'étude suivante porte sur les longueurs relatives. Comme le montre la figure 2.17, les longueurs relatives entre paire de segments ont été séparées en trois groupes : longueurs

FIG. 2.16 – Proportion d'apparition des angles



égales, une longueur est plus grande que l'autre ou enfin une longueur est beaucoup plus grande que l'autre. Ce graphique permet de voir qu'une grande majorité de segments sont de taille égale. La taille fixe et carrée des images de symboles (512*512) a peut-être une influence sur ce résultat. Nous conservons pour le calcul de la signature structurelle finale cette discrétisation des longueurs relatives en trois groupes.

FIG. 2.17 – Proportion d'apparition des longueurs relatives



D - Etude des chemins

La taille de la signature obtenue est déterminée par le nombre de chemins trouvés dans les symboles d'apprentissage. Le nombre de chemins calculés varie d'un symbole à l'autre. Lorsque l'on considère un ensemble de symboles d'apprentissage, nous calculons tout d'abord les chemins de chacun des symboles, puis nous collectons l'ensemble des chemins trouvés, tous symboles confondus. Si un chemin n'est pas présent pour un symbole donné, son nombre d'occurrences correspondant pour le symbole sera nul. La taille de la signature dépend de la longueur des chemins calculés, de la précision de la description des triplets topologiques et bien évidemment de la complexité des symboles de l'apprentissage. La table 2.4 présente la taille des signatures obtenues en fonction de la longueur des chemins calculés. La taille de la signature augmente largement en

fonction de la longueur des chemins. Les chiffres ont été obtenus avec le triplet topologique restreint aux types de relation. En prenant le triplet complet, pour les chemins de longueur 1, la taille de la signature passe à 63. Il sera donc sans doute nécessaire pour la suite du traitement de limiter la taille de la signature en sélectionnant les chemins les plus discriminants.

TAB. 2.4 – Taille des signatures obtenues pour différentes longueurs de chemins

Longueur des chemins [1]	1	2	3	4
Taille des signatures	7	28	161	759

[1] *triplet topologique restreint aux types de relation*

La table 2.5 nous montre la fréquence d'apparition des chemins de longueur 1 et 2 dans les symboles d'apprentissage. L'observation des fréquences d'apparition des chemins de longueur 1 confirme la présence majoritaire des types de relation L, O, P et V. Pour les chemins de longueur 2, comme l'on s'y attendait, les combinaisons LP, LV et VV qui correspondent à une succession d'angles à 45° ou 90° sont les plus représentées.

2.5 Conclusion

Ce chapitre a permis de décrire les signatures que nous allons utiliser pour caractériser les symboles à reconnaître, et également les principales bases de symboles sur lesquelles les tests expérimentaux ont été réalisés.

Le choix concernant les signatures statistiques de Fourier-Mellin, Radon et Zernike, se justifie par le fait qu'elles font référence et offrent de bons taux de classification. De plus, le paramétrage de ces signatures est facilité par leur utilisation fréquente dans les applications. Cependant, d'autres signatures faisant référence pourraient tout aussi bien être utilisées dans notre application.

Nous avons également présenté la mise en place d'une signature structurelle. Elle se compose d'une extraction de segments par transformée de Hough, et d'une description des structures élémentaires (rectangles, triangles, ...) composant les symboles, par un calcul de chemins dans le graphe topologique. Cette signature a l'avantage d'être modulable, puisqu'elle permet une description plus ou moins détaillée des symboles, et générique, car les structures élémentaires recherchées sont calculées automatiquement (pas de dictionnaire). Enfin, grâce à l'utilisation de la transformée de Hough, cette signature semble pouvoir offrir une bonne robustesse au bruit.

TAB. 2.5 – Fréquence d'apparition des chemins par symbole (chemins de longueur 1 et 2)

Chemins	L	LL	LO	LP	LT	LV	LX	LY
Apparition moyenne par symbole	2.2	2.05	2.47	3.75	0.68	3.45	0.63	0.26
Chemins	O	OO	OP	OT	OV	OX	OY	
Apparition moyenne par symbole	1.93	2.57	0.66	1.76	1.97	0.74	0.12	
Chemins	P	PP	PT	PV	PX	PY		
Apparition moyenne par symbole	2.25	1.27	1.6	2.72	0.4	0.56		
Chemins	T	TT	TV	TX	TY			
Apparition moyenne par symbole	0.79	0.31	0.82	0.19	0.25			
Chemins	V	VV	VX	VY				
Apparition moyenne par symbole	3.05	3.81	1.47	1.21				
Chemins	X	XX	XY					
Apparition moyenne par symbole	0.81	0.37	0.55					
Chemins	Y	YY						
Apparition moyenne par symbole	0.47	0.32						

Chapitre 3

Reconnaissance à l'aide du treillis de Galois

3.1 Introduction

Pour reconnaître les objets (ou plus précisément dans notre cas, les symboles), nous avons choisi d'utiliser comme classifieur un graphe particulier, le treillis de Galois. Egalement appelé treillis des concepts, il est issu du domaine de l'analyse formelle des concepts (AFC) [GW99]. Pour une description des treillis et de l'AFC, le lecteur devra se reporter à la partie 1.3 et pour plus de renseignements sur les classifieurs basés sur l'AFC, il est nécessaire de se référer à la partie 1.3.3.

Rappelons simplement que les *treillis de Galois* sont une représentation sous forme de graphe de toutes les correspondances possibles entre un ensemble d'*objets* (ou individus) O et un ensemble d'*attributs* (ou primitives, ou caractéristiques) I. Intuitivement, en partant du concept minimal $(O, f(O))$, et en progressant vers le concept maximal $(g(I), I)$, on observe une spécialisation de l'ensemble des objets et une généralisation de l'ensemble des attributs, ce qui se traduit pas une réduction du nombre d'objets au fur et à mesure que le nombre d'attributs augmente.

Cette partie concernant les treillis de Galois a pour but de décrire l'ensemble des étapes nécessaires à la reconnaissance des objets. Cette reconnaissance est classiquement réalisée en deux étapes : l'*apprentissage* (§ 3.2) et la *classification* (§ 3.3). L'apprentissage permet de stocker de manière organisée, sous la forme d'un treillis de Galois, des informations sur un ensemble d'objets connus. L'objectif de l'apprentissage est donc la construction du treillis de Galois. Ensuite, l'étape de classification (ou classement) doit

permettre d'attribuer à de nouveaux objets un label de classe, et ainsi de reconnaître un nouvel objet en lui associant l'objet le plus ressemblant parmi l'ensemble des objets connus.

3.2 Apprentissage

La phase d'apprentissage consiste à organiser l'information extraite d'un ensemble d'objets sous forme d'un treillis de Galois. Nous nous plaçons dans le cas où les objets sont des images. Chaque image est décrite par une signature donnée par un vecteur numérique (voir § 1.2.1 et 1.2.2) et un label de classe. Les signatures sont composées d'un certain nombre d'attributs extraits à partir des images. Comme l'illustre la figure 3.1, les deux phases principales composant l'apprentissage sont la *discrétisation* des données (ou quantification) et la *construction du treillis de Galois*. Il faut préciser que l'étape de discrétisation n'est pas toujours nécessaire. En effet, pour des données symboliques (ou qualitatives), les objets peuvent directement être distingués les uns des autres car ils possèdent telles ou telles caractéristiques. En revanche, pour des données numériques, la discrétisation est requise et consiste en la création d'intervalles disjoints. Les objets seront ainsi distinguables car ils possèdent des valeurs de leur signature appartenant à tels ou tels intervalles.

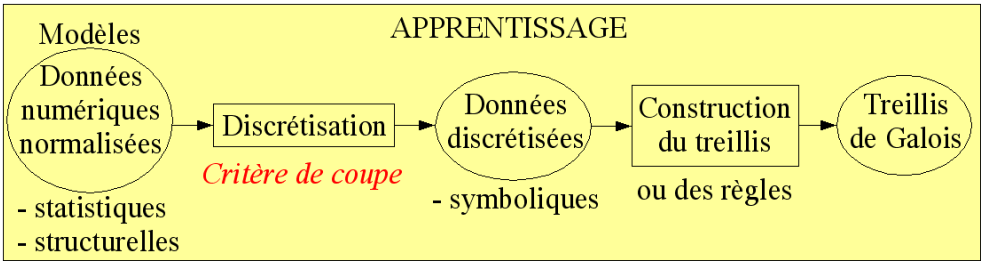
Nom: Apprentissage

Entrée: un *ensemble d'objets* O où chaque objet $p \in O$ est décrit par un vecteur numérique $p = (p_1, \dots p_n)$ où les valeurs sont normalisées et par une indication de classe $c(p)$.

Sortie: un *treillis de Galois* $(\beta(C), \leq)$ décrit par un ensemble de concepts $\beta(C)$ et une relation $\leq$ entre ces concepts.

Paramètre(s): Critère de coupe de la discrétisation

FIG. 3.1 – Description schématique de l'apprentissage



3.2.1 Normalisation des données

Il est nécessaire que les données soient préalablement normalisées pour que leur représentation soit équivalente. En effet, pour pouvoir comparer les données entre elles, nous devons leur appliquer une transformation pour qu'elles soient mises dans un même ordre de grandeur et qu'elles conservent leur rang.

Une première possibilité est d'appliquer une simple normalisation linéaire. Le principe est de calculer pour chaque attribut de la signature les valeurs min et max prises par les données et de recaler l'ensemble de ces valeurs dans un nouvel intervalle de données, par exemple $[a - b]$. Ainsi, chaque valeur p_i prise par un attribut $p \in I$ est transformée en p'_i selon la formule :

$$p'_i = \frac{|p_i - min|(b-a)}{max - min} + a$$

Une seconde possibilité est d'appliquer une normalisation suivant une loi normale à chaque attribut de notre signature. Cette loi est couramment employée car elle offre des propriétés intéressantes telles que la symétrie des données. Ainsi, chaque valeur p_i prise par un attribut i est transformée en p'_i selon la formule :

$$p'_i = \frac{p_i - \mu_p}{\sigma_p} \text{ avec } \mu_p = \frac{\sum_{i=1}^{n} p_i}{n} \text{ et } \sigma_p = \sqrt{\frac{(p_i - \mu_p)^2}{n}}$$

Une fois les données normalisées, nous pouvons réaliser l'apprentissage. Cette phase d'apprentissage se décompose en deux parties : une étape de discrétisation des données et une étape de construction du treillis de Galois. Lorsque les données sont de type numérique, il faut créer des intervalles de valeurs disjoints pour pouvoir distinguer

les données. Ce phénomène de découpage est appelé discrétisation. Par exemple, la température est une donnée de type numérique, et on peut former les intervalles de valeurs suivants : [5,0-10,0°C], [13,5-23,0°C] et [23,5-27,0°C].

3.2.2 Discrétisation des données

L'étape de discrétisation des données (ou quantification) consiste à partitionner les données pour obtenir des sous-ensembles contenant des objets qui possèdent des caractéristiques proches ou semblables. Autrement dit, les données numériques des différents objets sont organisées dans des intervalles disjoints de manière à obtenir une caractérisation spécifique de chaque classe d'objets.

Les méthodes de discrétisation [Rak97] peuvent être organisées selon trois axes :

- *supervisées / non-supervisées* : dans le cas de la discrétisation supervisée, on tient compte des classes des objets et de leurs similarités pour effectuer le découpage, alors que pour la discrétisation non-supervisée seules les similarités entre les objets sont considérées. Plus précisément, la discrétisation supervisée privilégie les découpes entre éléments de classes différentes ce qui permet de rapidement obtenir la séparation des classes. La discrétisation non-supervisée se base sur d'autres critères que la classe pour les découpes, comme par exemple la proximité des objets. La séparation peut donc concerner des objets d'une même classe, s'ils sont très dispersés. La discrétisation non-supervisée peut engendrer un très grand nombre de découpes avant d'obtenir la séparation des classes.
- *globales / locales* : pour la discrétisation globale, les intervalles sont contruits et fixés avant de réaliser la construction du classifieur. Au contraire, en discrétisation locale, le découpage est réalisé au fur et à mesure de sa construction. Ainsi à chaque étape de construction, on divise les ensembles d'objets concernés en sous-ensembles.
- *statiques / dynamiques* : la discrétisation statique est la stratégie la plus utilisée. Il s'agit de traiter les attributs indépendamment les uns des autres. A l'inverse, la discrétisation dynamique intègre l'information de tous les attributs simultanément pour construire les intervalles, ce qui permet la prise en compte d'une éventuelle corrélation entre les primitives.

Les études sur la discrétisation des données ont souvent été réalisées dans le cadre de l'utilisation d'un autre type de graphe mieux connu : l'arbre de décision (voir § 1.2.4). En effet, ce graphe est également utilisé comme classifieur et nécessite les même données d'entrée que le treillis de Galois. Nous préciserons ultérieurement que la ressemblance entre ces deux graphes ne s'arrête pas là (voir § 4.2.2).

Dans [DKS95], les auteurs réalisent une comparaison expérimentale entre les méthodes supervisées et non-supervisées de la discrétisation. Dans leurs tests d'évaluation, ils étudient les résultats sur 16 jeux de données de l'U.C. Irvine repository [MA94] pour trois méthodes : le découpage en intervalles égaux, l'algorithme 1RD (One-Rule Discretizer) de Holte [Hol93] et le découpage récursif minimisant l'entropie [FI93]. La première méthode est non-supervisée et les deux autres sont supervisées. La comparaison des méthodes est effectuée en appliquant deux classifieurs différents (C4.5 de [Qui93], et un classifieur bayésien : $\mathcal{MLC}++$ de [KJL+94]) à la suite de l'étape de discrétisation. Les classifieurs permettent de déterminer la classe d'appartenance de chaque objet d'un ensemble. En utilisant ces classifieurs, les auteurs peuvent donc évaluer la précision de la classification par rapport à celle obtenue directement par les classifieurs sans effectuer de discrétisation au préalable. Ils concluent que les trois méthodes donnent des résultats à peu près similaires, avec un léger avantage pour les méthodes supervisées. La meilleure performance moyenne est obtenue pour le découpage récursif minimisant l'entropie associé au classifieur bayésien.

La thèse de Ricco Rakotomalala [Rak97] comporte une partie sur la discrétisation des primitives continues. Dans ce chapitre, il explique notamment combien il est difficile de comparer les méthodes globales et locales, étant donné qu'elles possèdent toutes deux des avantages et des inconvénients. La discrétisation locale, avec un partitionnement en un nombre fixe d'intervalles, peut notamment conduire à un fractionnement de données, et la construction de l'arbre sera donc plus en profondeur. Cependant, ce procédé possède l'avantage de prendre en compte les interactions entre les primitives. S'agissant de la discrétisation globale, elle induit la construction d'un arbre plus en largeur. De plus, elle permet une réduction de la dimension à la fois au niveau des primitives (horizontalement) mais aussi au niveau des individus (verticalement) ce qui améliore la rapidité de traitement. Les tests effectués par [Qui96] et [DKS95] ne permettent pas de trancher entre les deux stratégies. Il est nécessaire de savoir ce que l'on souhaite obtenir pour utiliser la méthode la plus en adéquation avec les attentes.

En conclusion, il paraît intéressant de préférer l'utilisation d'une méthode de discrétisation supervisée, statique (les plus utilisées), basée sur un critère de minimisation de l'entropie (critère le plus efficace d'après [DKS95] et [Rak97]), et enfin de choisir entre stratégie globale et locale en fonction de l'objectif souhaité.

Revenons à notre cas d'étude, l'algorithme ci-dessous présente les éléments en entrée et en sortie lors d'un processus de discrétisation :

Nom: Discrétisation

Entrée: un *ensemble d'objets* O où chaque objet $p \in O$ est décrit par une signature qui est un *vecteur numérique* $p = (p_1, \ldots p_n)$ où chaque valeur est normalisée et par un *label de classe* $c(p)$.

Sortie:
- les *intervalles* regroupés par ensembles $I = I_1 \times I_2 \times \ldots \times I_m$ où les intervalles de chaque ensemble I_i sont disjoints, et couvrent l'ensemble des valeurs p_i prises par tous les objets $p \in O$.
- une *relation d'appartenance* R définie pour chaque objet $p \in O$ et chaque intervalle $x \in I$ par : $pRx \Leftrightarrow$ il existe $i = 1 \ldots m$ tel que $p_i \in x \in I_i$

Paramètre(s): Critère de coupe de la discrétisation

La discrétisation est réalisée sur les signatures des objets que l'on peut organiser sous la forme d'une table de données à double entrée (voir Tab. 3.1). Au commencement, on construit pour chaque attribut $i \leq n$ un intervalle $x \in I_i$ qui regroupe toutes les valeurs p_i prises par les différents objets $p \in O$. C'est ainsi que l'on initialise la relation d'appartenance R d'un objet à un intervalle. Après cette étape d'initialisation, chaque ensemble I_i contient un seul intervalle, et chaque objet $p \in O$ est en relation avec chaque intervalle $x \in I_i$.

Exemple 4 *Prenons l'exemple de la table 3.1. Elle présente les données normalisées de 10 objets répartis suivant 4 classes. La signature caractérisant chaque objet est composée de 3 attributs (a, b et c). Après initialisation de la relation d'appartenance R, on obtient alors la table 3.2.*

Il faut alors sélectionner un intervalle x à découper, et déterminer le point de coupe au sein de cet intervalle x. Pour cela, nous introduisons les notations suivantes :

- Pour un objet $p \in O$, on définit l'ensemble I_p des intervalles en relation d'appartenance avec p :
$$I_p = \{x \in I \text{ tel que } pRx\}.$$
- Pour chaque intervalle $x \in I$, on définit l'ensemble V_x des valeurs numériques prises par les objets en relation d'appartenance avec l'intervalle et rangées par

TAB. 3.1 – Signatures des 10 objets

Classe	*Ident.*	*Signature*		
		a [0-20]	*b* [0-20]	*c* [0-20]
1	1	1	4	15
	2	0	0	18
2	3	1	12	13
	4	0	16	15
	5	3	12	11
3	6	8	16	15
	7	6	20	20
	8	15	12	15
4	9	18	4	0
	10	20	12	2

ordre croissant :

$$\begin{aligned} V_x &= (p_i \text{ tel que } p_i \in x) \text{ rangés par ordre croissant} \\ &= v_1 \leq v_2 \leq \ldots \leq v_n \end{aligned}$$

Ainsi, il est nécessaire de sélectionner un intervalle $x \in I$, et une valeur $v_j \in V_x$, et de couper l'intervalle x en deux intervalles x' et x'' avec $V'_x = v_1 \leq \ldots \leq v_j$ et $V_{x''} = v_{j+1} \leq \ldots \leq v_n$. La sélection de l'intervalle ainsi que le point de coupe dépendent quant à eux d'un *critère de coupe* à définir.

Chaque objet sera alors en relation d'appartenance avec seulement l'un des deux intervalles créés, ce qui permettra de différencier les intervalles des deux sous-ensembles formés. Ce procédé de découpage des intervalles est réitéré jusqu'à ce que les classes soient séparées et distingables les unes des autres.

Lorsque la séparation des classes est vérifiée, chaque classe peut alors être caractérisée par son propre ensemble d'intervalles. On obtient ainsi une table discrétisée comprenant l'ensemble des objets $p \in O$ et l'ensemble des intervalles $I = I_1 \times I_2 \times \ldots \times I_m$ où I_i est l'ensemble des intervalles obtenus pour l'attribut $i = 1 \ldots m$. Remarquez que si l'attribut k n'a jamais été sélectionné pour être discrétisé, l'ensemble I_k contient un seul intervalle ($|I_k| = 1$) qui est en relation avec tous les objets. Cet attribut n'est pas discriminant, on peut donc le supprimer de la table discrétisée. Grâce à cette table, il est possible de retrouver la relation d'appartenance R, et donc, pour chaque objet $p = (p_1, p_2, \ldots, p_m) \in O$ où p_i est la valeur pour l'attribut $i = 1 \ldots m$, de connaître l'ensemble I_p des intervalles associés à p.

TAB. 3.2 – Intervalles obtenus après initialisation de la relation d'appartenance R

Classe	*Ident.*	*Intervalles*		
		a [0-20]	*b* [0-20]	*c* [0-20]
1	1	X	X	X
	2	X	X	X
2	3	X	X	X
	4	X	X	X
	5	X	X	X
3	6	X	X	X
	7	X	X	X
	8	X	X	X
4	9	X	X	X
	10	X	X	X

Il est nécessaire de remarquer que l'ensemble des attributs de la signature ne sont pas conservés pour la construction du treillis de Galois. En effet, tous les attributs qui ne sont pas choisis par le critère de coupe et restent donc en relation avec l'ensemble des objets, ne nous permettent pas de distinguer des classes d'objets. Nous les supprimons en effectuant une réduction du contexte. D'autre part, cette sélection d'attributs permet de réduire la complexité du problème en ne conservant que les attributs les plus pertinents. C'est une démarche importante lorsque l'on a peu d'objets en apprentissage car elle permet de faire face au problème de la malédiction de la dimensionnalité [Bel61]. Ce problème est le suivant : plus le nombre d'attributs pour décrire les données est grand, plus il est probable d'y trouver de l'information pertinente, mais également d'y trouver du bruit ou pire encore des attributs nuisibles à une bonne classification. Il est donc nécessaire de réduire le nombre de dimensions en supprimant ces attributs nuisibles ou inutiles, et en conservant les attributs pertinents.

Exemple 5 *Reprenons l'exemple de la table 3.1. Après discrétisation par la méthode de l'entropie, on obtient la table 3.3. On peut observer que chaque attribut a été sélectionné et découpé une fois, ils sont donc tous conservés.*

Beaucoup de *critères de coupe* permettent de sélectionner un intervalle afin de le diviser et de déterminer le point de coupe. Le choix de ce paramètre est décisif pour la phase d'apprentissage. Il est nécessaire de rechercher un intervalle $x \in I$ qui *maximise* un critère, pour une valeur de l'intervalle donnée v_j. Les valeurs de l'intervalle $V_x = (v_1 \dots v_n)$ sont bien naturellement incluses dans cet intervalle et rangées par ordre croissant. Cet intervalle sera alors coupé entre les valeurs v_j et v_{j+1}. Il est possible de

TAB. 3.3 – Table discrétisée selon un critère d'entropie

Classe	*Ident.*	*Intervalles*					
		a1 [0-3]	*a2* [6-20]	*b1* [0-4]	*b2* [12-20]	*c1* [0-2]	*c2* [11-20]
1	1	X		X			X
	2	X		X			X
2	3	X			X		X
	4	X			X		X
	5	X			X		X
3	6		X		X		X
	7		X		X		X
	8		X		X		X
4	9		X	X		X	
	10		X		X	X	

définir de nombreux critères de coupe dépendant ou non des données. Parmi ces critères, nous distinguons la distance maximale, l'entropie [FI93] et le coefficient de Hotelling [Hot36].

La méthode de la distance maximale est un critère de coupe non supervisé qui consiste à rechercher l'intervalle qui possède l'écart maximal entre deux valeurs consécutives, lorsque les valeurs de l'intervalle sont rangées par ordre croissant. Les deux autres critères de coupe proposés tiennent compte des labels de classes, ce sont donc des critères supervisés. La fonction d'entropie est une mesure caractérisant le degré de mélange des classes. Enfin, le coefficient de Hotelling tient compte à la fois de l'espace entre les éléments et des labels de classes. Ce critère permet de maximiser la distance entre les classes et également de minimiser la dispersion des classes.

– la distance maximale :

$$distance(v_j) = v_{j-1} - v_j$$

– l'entropie :

$$gain_E(v_j) = E(V_x) - (\frac{j}{n}E(v_1 \dots v_j) + \frac{n-j}{n}E(v_{j+1} \dots v_n))$$

avec

$$E(V) = -\sum_{k=1}^{|c(V)|} \frac{n_k}{n} log_2(\frac{n_k}{n})$$

la mesure d'entropie d'un intervalle V de n valeurs où n_k est le nombre d'objets de la classe k contenus dans l'intervalle.

– le coefficient de Hotelling :

$$gain_H(v_j) = H(V_x) - (\frac{j}{n}H(v_1 \dots v_j) + \frac{n-j}{n}H(v_{j+1} \dots v_n))$$

avec

$$H(V) = \frac{VarInter(V)}{VarIntra(V)}$$

la mesure de Hotelling d'un intervalle V de n valeurs, avec n_k le nombre d'objets de la classe k, g_k le centre de gravité de la classe k, g le centre de gravité de V, v_{k_i} le i-ème élément de la classe k,

$$VarInter(V) = \frac{1}{n} \sum_{k=1}^{|c(V)|} n_k (g_k - g)^2$$

la mesure de la variance inter-classes et

$$VarIntra(V) = \frac{1}{n} \sum_{k=1}^{|c(V)|} n_k (\sum_{i=1}^{n_k} (v_{k_i} - g_k)^2)$$

la mesure de la variance intra-classe.

Tout critère supervisé s'annule lorsque les classes sont séparées. Ainsi, lorsque la table discrétisée sépare les classes, le critère de Hotelling est nul, alors qu'un critère non supervisé peut encore être utilisé. En poursuivant la discrétisation, la table obtenue contiendra plus d'intervalles et une description plus fine des classes, mais risque de contenir des données plus corrélées. A l'inverse, un critère d'arrêt autre que la séparation entre classes pourrait être utilisé pour que la discrétisation s'arrête avant que les classes ne soient séparées.

Notons la possibilité d'intégrer des données symboliques aux données numériques. L'intégration de ces données consiste à calculer une extension de la relation d'appartenance R pour ensuite ajouter ces données au treillis. Cette intégration peut aussi être réalisée au cours de l'initialisation de la relation R, avant la discrétisation, et ainsi servir à affiner le critère de coupe.

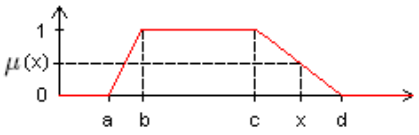
A la fin de la discrétisation, nous obtenons donc un ensemble d'intervalles disjoints pour chaque attribut de la signature qui a été choisi pour être découpé. La conjonction de plusieurs intervalles permet d'identifier telle ou telle classe d'objets. Dans notre cas expérimental, les objets à reconnaître sont des symboles bruités. La présence de ce bruit entraîne des modifications des valeurs de leur signature qui risquent de ne plus être incluses dans les intervalles correspondant à leur classe. Il est donc intéressant de rendre les bornes des intervalles plus souples et d'intégrer cette dérive au sein du processus de reconnaissance par l'utilisation non pas d'intervalles, mais de nombres flous [Zad65, Zad96].

Rappelons qu'un nombre flou A sur un univers U est défini par une *fonction d'appartenance* μ_A (ou μ) qui précise le *degré de vraisemblance* de l'assertion $x \in A$:

$$\begin{aligned} \mu : U &\rightarrow [0,1] \\ x &\mapsto \mu_A(x) \end{aligned}$$

Un nombre flou A est habituellement défini par un trapèze $[a,b,c,d]$ de *support* $[a,d]$ et de *noyau* $[b,c]$ (voir Fig. 3.2).

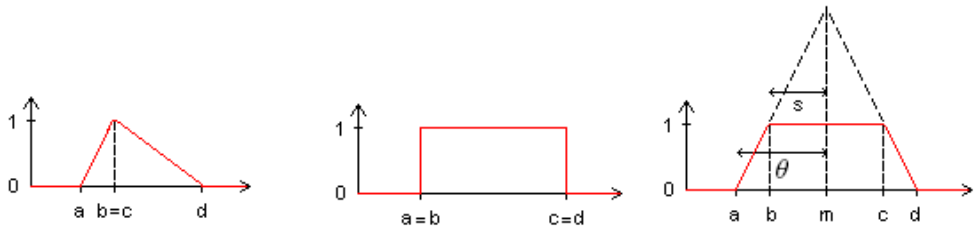
FIG. 3.2 – Exemple de nombre flou



Il existe quelques cas particuliers de nombres flous :
- *les triangulaires* : où $b = c$ (voir Fig. 3.4(a))
- *les rectangulaires* : où $a = b$ et $c = d$ (voir Fig. 3.4(b))
- *les symétriques* : de noyau $[m-t, m+t]$ et de support $[m-s*t, m+s*t]$, avec m le milieu du nombre flou, $t = |m-a| = |m-d|$ et $s = |m-b| = |m-c|$ (voir Fig. 3.4(c))

Une représentation des attributs par des intervalles correspond en logique floue à l'utilisation de nombres flous rectangulaires.

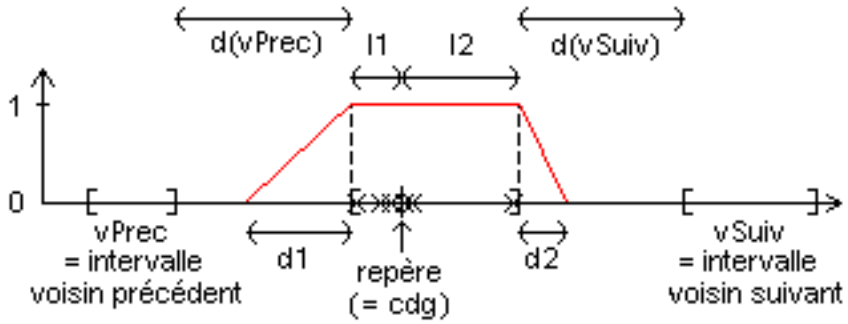
FIG. 3.3 – Cas particuliers de nombres flous



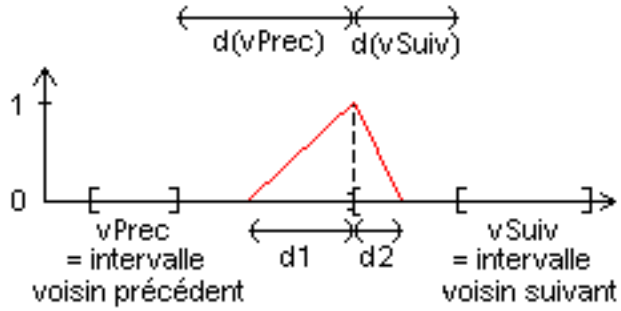
(a) triangulaire (b) rectangulaire (c) symétrique

Plusieurs extensions d'un intervalle (issu de la discrétisation) en un nombre flou sont envisageables. Nous en avons retenu une, illustrée par la figure 3.4, qui prend en compte la distribution des valeurs de l'intervalle ainsi que la proximité avec les intervalles consécutifs précédents et suivants. Ces derniers sont les intervalles disjoints voisins, issus du même attribut de la signature que l'intervalle courant, et triés par ordre croissant.

FIG. 3.4 – Exemple de nombre flou formé à partir d'un intervalle



(a) Intervalle non singleton



(b) Intervalle singleton

Dans les figures 3.4, un intervalle x est symbolisé par ses 2 bornes et par un ensemble de croix représentant la distribution de ses valeurs V_x. De plus, $vPrec$ et $vSuiv$ sont les intervalles voisins précédent et suivant l'intervalle x. Enfin, $d(vPrec)$ et $d(vSuiv)$ sont respectivement les distances entre les bornes des intervalles $vPrec$ et x, et entre les bornes des intervalles x et $vSuiv$. Il est nécessaire de préciser son repère à choisir parmi : le milieu de l'intervalle, le centre de gravité de la distribution des valeurs comprises dans l'intervalle, ou encore la valeur médiane de cette même distribution. Cet exemple de nombre flou est paramétrable selon un degré de flou θ tel que $0 <= \theta < 1$). Plus θ est petit, plus le nombre flou aura un support large ; et plus θ est grand, plus le nombre flou sera proche d'un nombre flou rectangulaire. La fonction f utilisée pour cet exemple est : $f(l, \theta) = l.\theta$. Les bornes floues inférieure et supérieure d'un intervalle non singleton 3.5(a) sont obtenues à partir des formules suivantes :

- $d_1 = min(f(l_2, \theta), d(vPrec))$
- $d_2 = min(f(l_1, \theta), d(vSuiv))$

Les bornes floues inférieure et supérieure d'un intervalle singleton 3.5(b) (intervalle où les bornes sont égales), sont obtenues à partir des formules suivantes :

– $d_1 = min(f(d(vPrec), \theta), d(vPrec))$
– $d_2 = min(f(d(vSuiv), \theta), d(vSuiv))$

Chaque intervalle possède au moins un intervalle voisin étant donné que l'on supprime les intervalles non découpés lors de la discrétisation. Si l'intervalle voisin précédent (respectivement suivant) n'existe pas, alors d_1 (respectivement d_2) reçoit la même valeur que celle calculée pour d_2 (respectivement d_1).

3.2.3 Construction du treillis de Galois

Après la phase de discrétisation vient la construction du treillis de Galois. Cette étape est totalement déterminée par la relation d'appartenance R obtenue. Il n'existe pas de critère ou de paramètre à prendre en compte pour la construction de ce graphe étant donné qu'il représente toutes les combinaisons possibles entre objets et intervalles qui sont en relation selon R. La représentation du treillis de Galois de la relation R est unique et les concepts correspondant aux relations objets-intervalles sont ordonnés par inclusion.

Nom: Construction du treillis

Entrée: une *relation d'appartenance* R entre un ensemble d'objets O et un ensemble d'intervalles I.

Sortie: un *treillis de Galois* $(\beta(C), \leq)$ décrit par un ensemble de concepts $\beta(C)$ et une relation $\leq$ entre ces concepts.

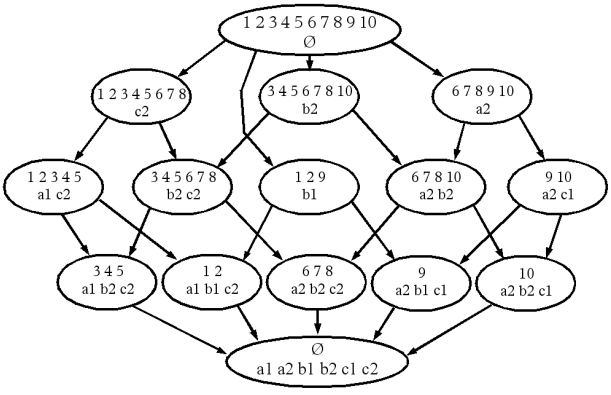
Paramètre(s): Aucun

Dans l'état de l'art, nous avons présenté les principaux algorithmes de construction du treillis de Galois (voir § 1.3.2). Rappelons que la principale limite de l'utilisation du treillis de Galois est due à son coût à la fois en temps et en espace. En effet, la taille du treillis est bornée par $2^{|O+I|}$ dans le pire des cas, et par $|O + I|$ dans le meilleur des cas. Des études de complexité en moyenne sont extrêmement difficiles à mener, car la taille du treillis dépend à la fois de la taille des données à classifier, mais aussi de leur organisation et de leur diversité. Notons cependant que sa taille reste raisonnable en pratique, comme l'illustrent les expérimentations qui en ont déjà été faites [MNN05].

Une manière de procéder pour construire le treillis de Galois est de suivre le principe de l'algorithme NextClosure (voir § 1.3.2), en établissant la liste de tous ses concepts selon un ordre (ordre lectique). La recherche des concepts du treillis consiste à trouver dans la table discrétisée les rectangles maximaux, c'est-à-dire les plus grands ensembles de relations d'appartenance entre objets et intervalles. Après avoir généré tous les concepts, il ne reste plus qu'à les ordonner par inclusion.

Exemple 6 *A partir de la table 3.3, on peut déterminer les concepts du treillis en recherchant les rectangles maximaux. Dans cet exemple, un ensemble* $X = \{x_1, x_2, \ldots, x_n\}$ *est écrit commme un mot* $x_1\, x_2 \ldots x_n$*. Il est possible d'effectuer les regroupements suivants :* $(1\,2,\ a_1\, b_1\, c_2)$*,* $(3\,4\,5,\ a_1\, b_2\, c_2)$*,* $(6\,7\,8,\ a_2\, b_2\, c_2)$*,* $(9,\ a_2\, b_1\, c_1)$ *et enfin* $(10,\ a_2\, b_2\, c_1)$*. On peut remarquer que chacun de ces concepts caractérise une classe. Par exemple, les objets 1 et 2 qui possèdent les mêmes intervalles sont regroupés au sein d'un même concept caractérisant la classe 1. De même, les objets 3, 4 et 5 d'une part, et 6, 7 et 8 d'autre part sont regroupés par concept caractérisant respectivement les classes 2 et 3. Enfin, la classe 4 est représentée par deux concepts différents, l'un contenant l'objet 9 et l'autre l'objet 10. Avec ces 5 premiers concepts créés, en réalisant les intersections 2 à 2 sur les intervalles, on peut alors générer tous les autres concepts. L'intersection sur les intervalles entre les concepts* $(1\,2,\ a_1\, b_1\, c_2)$ *et* $(3\,4\,5,\ a_1\, b_2\, c_2)$ *est* $a_1\, c_2$*. En recherchant dans la table discrétisée (voir Tab. 3.3), on a* $f(a_1\, c_2) = 1\,2\,3\,4\,5$*. Un nouveau concept est alors créé :* $(1\,2\,3\,4\,5,\ a_1\, c_2)$ *qui regroupe des objets de classes différentes, et ne caractérise donc plus une classe en particulier. En reproduisant ce procédé d'intersection sur les intervalles, on génère ainsi l'ensemble des concepts. Enfin, en les ordonnant par inclusion, on construit le treillis de Galois de la figure 3.5.*

FIG. 3.5 – Diagramme de Hasse du treillis de Galois



Il est possible de définir des *critères de validité* permettant d'invalider certains concepts pour ne pas les générer et ainsi réduire la taille du treillis. On retrouve ainsi

le même effet qu'avec le critère d'arrêt de la discrétisation, mais aussi avec la notion d'élagage introduite pour les arbres de décision, ou encore l'opérateur défini pour déterminer si un nœud est terminal ou non. Ces différents critères proposés peuvent donc s'appliquer ici. Ils permettent d'invalider des concepts jugés trop spécifiques et sources d'erreurs, et qui sont généralement proches (au sens de la relation $\prec$) de l'élément maximal du treillis. Ces heuristiques génèrent une structure issue du treillis dont des concepts "maximaux" ont été supprimés (il s'agit d'un inf-demi-treillis[1]) souvent appelé Iceberg [STB+02, NVRG05]. Un premier critère de validité simple et immédiat consiste à valider les concepts proposant une ou plusieurs classes candidates, et ainsi à invalider le seul concept maximal $(g(I), I)$ où $c(I)$ ne contient pas de classes candidates.

Afin de limiter la complexité exponentielle du treillis, notons la possibilité de générer non pas le treillis lui-même, mais une *représentation* du treillis. Parmi les nombreuses représentations proposées dans la littérature, citons la représentation par un *système de règles d'implication* (voir § 3.5.4) et plus précisément par la *base Guigues-Duquenne* [OD03] ou encore la *base canonique directe* [BN04] que l'on retrouve notamment en analyse de données [TB01]. Une telle représentation permet, outre sa propriété de représentativité condensée du treillis, d'éviter la génération complète du treillis en proposant une *génération à la demande* des concepts du treillis qui sont nécessaires au cours de la phase de classification. Pour plus d'informations sur la construction du treillis de Galois et ses liens avec les systèmes de règles, se référer à la partie algorithmique de [MB05].

Il est également possible d'envisager cette *génération à la demande* sans passer par un système de règles d'implication, mais directement en construisant les concepts nécessaires du treillis de Galois. L'algorithme de construction du treillis de Bordat [Bor86] (voir Alg. `Treillis de Galois` dans la partie 1.3.2) est le plus approprié pour une génération à la demande des concepts puisqu'il génère la relation de couverture entre les concepts. L'algorithme `Système de fermeture à la demande` est une extension de l'algorithme de Bordat, où nous ajoutons un critère de choix pour sélectionner un concept parmi $Couverture(A)$. Ce critère de choix est un paramètre de la classification qui sera présenté dans la partie suivante (voir § 3.3.4). Etant donné que l'algorithme de Bordat génère les concepts selon une stratégie en largeur d'abord, il a dû être adapté à une stratégie en profondeur d'abord de manière récursive, et être initialisé par $(O, f(O))$ le premier concept du treillis.

Dans le système de reconnaissance de symboles, nous utilisons principalement l'algorithme de génération à la demande des concepts pour une optimisation du temps. Toutefois, lorsque le besoin l'exige, il est possible de construire l'intégralité du treillis de Galois. Pour cela, nous utilisons l'algorithme de génération à la demande modifié,

[1]Un inf-demi-treillis est un treillis qui ne vérifie que la propriété de borne inférieure

Nom : `Couverture des objets`
Données: Un contexte K ; Un concept (A, B) de K
Résultat: Les sous-ensembles maximalement inclus de F_A
début

1. Calculer F_A : $F_A = \{g(b) \cap A \ : \ b \in I - B\}$;
2. Calculer F : les sous-ensembles maximalement inclus de F_A;

Retourner F;
fin

Nom : `Système de fermeture à la demande`
Données: Un contexte $K = (O, I, R)$; un sous-ordre $(\mathbb{C}'_I, \prec)$ de la relation de couverture $(\mathbb{C}, \prec)$; un ensemble fermé $B \in \mathbb{C}'_I$
début

si B *n'est pas marqué* **alors**
  F=`Couverture des attributs` (K, B);
  Sélectionner X dans F selon le critère de choix;
  $B' = B + X$;
  Ajouter la relation de couverture $B \prec B'$;
  si $B' \notin \mathbb{C}'_I$ **alors** Ajouter B' dans $\mathbb{C}'_I$;
  Marquer B;
fin
sinon
  Calculer $F = \{B' \setminus B \ : \ B' \in \mathbb{C}'_I$ et $B \prec B'\}$;
  Sélectionner X dans F selon le critère de choix;
fin
si $B' \neq I$ **alors** `Système de fermeture à la demande`$(K, \mathbb{C}'_I, \prec, B')$;
fin

comme nous venons de le préciser, en remplaçant l'instruction de sélection d'un concept par une boucle de parcours de tous les concepts. Dans la suite, pour des besoins de clarté, nous supposons que le treillis de Galois est intégralement généré. Remarquons que la stratégie en profondeur d'abord de génération de tous les concepts consisterait à remplacer `Sélectionner` X `dans` F par la boucle `Pour chaque` X `dans` F pour considérer tous les sous-ensembles de F de la même manière, ce qui rendrait inutile l'instruction `Sinon`.

3.3 Classification

Lorsque la phase d'apprentissage est achevée et que le treillis de Galois est généré, il est alors possible de passer à l'étape de classification (ou classement) illustrée par la figure 3.6. Le principe est de déterminer la classe de nouveaux objets, c'est-à-dire de reconnaître la classe d'objets qui peuvent être plus ou moins détériorés. Une telle détérioration prend tout son sens lorsque les objets sont des images, et les données numériques des signatures extraites de ces images.

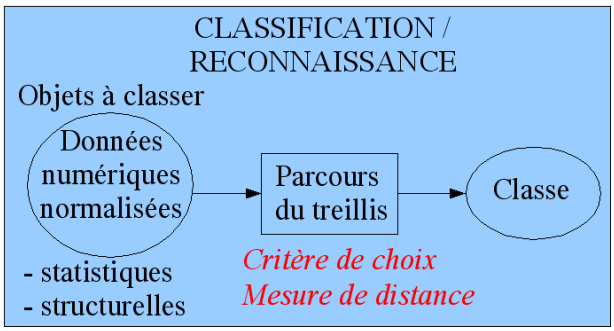
Nom: Classification

Entrée:
- le vecteur numérique $s = (s_1 \dots s_n)$ d'un objet s à classer
- le treillis de Galois $(\beta(C), \leq)$ issu de la phase d'apprentissage

Sortie: une classe $c(O)$ pour s

Paramètre(s): Une mesure de distance, un critère de choix

FIG. 3.6 – Description schématique de la classification

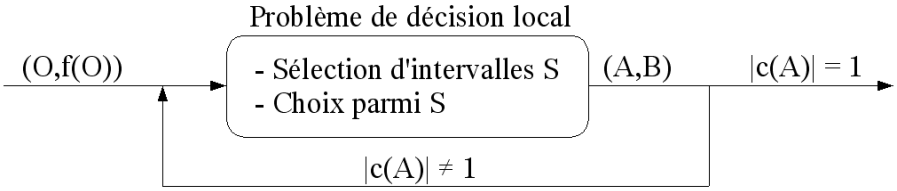


3.3.1 Principe de navigation

Le treillis de Galois peut être vu comme un espace de recherche dans lequel on évolue en fonction des attributs validés. Il s'agit à partir du *concept minimal* $(O, f(O))$ où toutes les classes des objets sont candidates à la reconnaissance et aucun intervalle

n'est validé, de progresser étape par étape au sein du treillis de Galois par validation de nouveaux intervalles et par conséquent réduction de l'ensemble d'objets, jusqu'à un *concept final* où les objets restants, qui sont en relation avec tous les intervalles validés durant le parcours du graphe, sont tous de la même classe. Plus formellement, à partir du concept minimal $(O, f(O))$, il s'agit d'itérer une prise de décision locale jusqu'à atteindre un concept final (A, B) où les objets de A sont de la même classe : $|c(A)| = 1$ (voir Fig. 3.7). A chaque prise de décision locale, on progresse dans le graphe d'un *concept courant* vers l'un de ses concepts successeurs et on valide ainsi un nouvel ensemble d'intervalles de taille toujours plus importante. Il faut alors définir un critère pour la sélection des intervalles à un niveau local. En pratique, la navigation est effectuée dans le diagramme de Hasse, c'est-à-dire la réduction transitive et réflexive du treillis de Galois (voir Fig. 1.4).

FIG. 3.7 – Navigation dans le treillis de Galois



3.3.2 Description d'une étape élémentaire de classification

La reconnaissance d'un objet avec le treillis de Galois repose donc sur l'itération d'une prise de décision locale, que nous appelons étape élémentaire de classification (voir Fig. 3.7). Elle consiste à partir d'un concept courant à sélectionner l'ensemble d'intervalles S issus de la structure et à faire un choix parmi ces intervalles. S est obtenu à partir des concepts successeurs du concept courant. Plus formellement, si on note (A, B) le concept courant, et $(A_1, B_1), \ldots, (A_n, B_n)$ ses n successeurs dans le treillis de Galois, alors S est un système d'intervalles (voir § 1.3.1) :

$$S = \bigcup_{i=1}^{n} B_i \backslash B = \{X_1, \ldots, X_n\}$$

Les ensembles de S vérifient les propriétés suivantes :
– Ils sont disjoints : $X_i \bigcap X_j = \emptyset,\ \forall i, j \leq n, i \neq j$

– X_i $(i \leq n)$ ne peut contenir 2 intervalles issus d'une même caractéristique j $(j \leq m)$: $|X_i \bigcap I_j| \leq 1$.

Le système S des intervalles sélectionnés à partir d'un concept courant s'obtient directement à partir de la structure, et ne nécessite pas de paramétrage spécifique. Une fois cette famille S calculée, il s'agit ensuite de choisir un ensemble d'intervalles X_i.

Ce choix est essentiel à toute étape élémentaire de classification, base de la navigation dans le treillis et de la classification de s. C'est la raison pour laquelle ce choix est dépendant des données, et non de la structure. Plus précisément, il est dépendant d'un *critère de choix* (voir § 3.3.4) défini à partir d'une *mesure de distance* (voir § 3.3.3) entre s et un intervalle.

Exemple 7 *Dans le treillis de Galois (voir Fig. 3.5), on mélange les intervalles provenant de différents attributs. Pour le concept minimal, le choix doit être fait parmi* $S = \{c_2, b_1, b_2, a_2\}$. *De plus, on peut être amené à sélectionner plusieurs intervalles en même temps. Par exemple, il y a 2 intervalles à tester pour les successeurs du concept* $(\{1, 2, 9\}, \{b_1\})$ *:* $S = \{a_1\, c_2, a_2\, c_1\}$.

Enfin, on peut constater qu'il existe dans le treillis de Galois plusieurs chemins (scénarii de reconnaissance) qui permettent d'aboutir à une même classe. C'est cette propriété qui nous a motivés à utiliser le treillis de Galois dans le cadre d'une classification d'objets détériorés.

3.3.3 Mesure de distance

Faire un choix parmi S nécessite l'utilisation d'une *mesure de distance* entre la valeur s_i de l'objet s à classer et un intervalle $x \in I_i$. Une première idée de distance d toute simple pourrait être :

$$d(s_i, x) = \begin{cases} 0 \text{ si } s_i \in x \\ 1 \text{ sinon} \end{cases}$$

Une amélioration permet d'intégrer l'éloignement entre la valeur s_i et l'intervalle x. Cette distance peut être définie entre s_i et la borne la plus proche de l'intervalle x, ou entre s_i et le milieu de l'intervalle x, entre s_i et le centre de gravité de l'intervalle x, ou encore entre s_i et la valeur médiane de l'intervalle x.

Une autre mesure plus sophistiquée consiste à intégrer l'éloignement entre la valeur s_i et l'intervalle x par rapport à la taille de l'intervalle. Par exemple, il est possible de calculer le ratio entre la distance euclidienne de s_i au centre de l'intervalle et la demi-longueur de celui-ci :

$$d(s_i, x) = \frac{\sqrt{(s_i - x_{milieu})^2}}{\sqrt{(x_{borneInf} - x_{milieu})^2}} = \frac{\sqrt{(s_i - x_{milieu})^2}}{\sqrt{(x_{borneSup} - x_{milieu})^2}}$$

Par abus de notation, notons $d(s, x)$ au lieu de $d(s_i, x)$ cette mesure de distance, afin de pouvoir l'étendre à un ensemble d'intervalles $X \subseteq I$:

$$d(X) = \frac{1}{|X|} \sum_{x \in X} d(s, x)$$

Dans la description de la discrétisation (voir § 3.2.2), nous avons présenté l'utilisation de nombres flous à la place des intervalles pour offrir plus de souplesse au traitement des valeurs issues de données bruitées. Dans ce cas, la mesure de distance peut correspondre à la *fonction d'appartenance* du nombre flou [Zad65, Zad96].

3.3.4 Critère de choix

Etant donnés $S = \{X_1, \ldots, X_n\}$ la famille des intervalles sélectionnés, et une *mesure de distance* $d(s, X)$ entre l'objet s à classer et un ensemble d'intervalles $X_i \subseteq I$, il s'agit de mettre en place un critère de choix afin de choisir X_i parmi S. Il est possible de définir un nombre important de critères de choix, dont une liste exhaustive n'est pas envisageable. Voici quelques exemples simples qui sont tous basés sur la distance d :

1. Choisir i tel que $d(s, X_i)$ est minimal.
2. Choisir i tel que $|X_i \bigcap I_k| = |\{x \in X_i \bigcap I_k\}|$ est maximal, avec I_k l'ensemble des k premiers intervalles de S rangés par ordre croissant suivant la distance $d(s, x)$.
3. Choisir i tel que $|\{x \in X_i$ tel que $d(s, x) < d_c\}|$ est maximal, avec d_c une constante que l'on peut assimiler à un degré de flou. En effet, si la mesure de distance employée est une fonction d'appartenance f associée à l'intervalle x, on peut remplacer l'assertion précédente par : Choisir i tel que $|\{x \in X_i$ tel que $f(s, x, \theta) > 0\}|$ est maximal, avec θ un degré de flou.

Le critère n°1, défini de manière globale sur chaque X_i, possède l'inconvénient de noyer le bruit. Le second critère de choix suit le principe des k plus proches voisins (voir § 1.2.4). Remarquez que le troisième critère est un cas particulier du second.

Ces critères étant locaux pour chaque $i = 1 \dots n$, on peut en imaginer des plus sophistiqués faisant apparaître :

4. Une combinaison de ces différents critères de choix, et surtout en cas de choix multiples, c'est-à-dire lorsque plusieurs X_i sont à égalité.
5. Les intervalles de $\bigcup_{i=1}^{n} X_i$ d'un même attribut j : $\bigcup_{i=1}^{n} X_i \bigcap I_j$ et $d(X_i \bigcap I_j)$ de manière à favoriser un attribut particulier j (par exemple un attribut supposé non détérioré).
6. La notion d'intervalles *discriminants* (lorsque l'union des intervalles ne se retrouve que dans le concept maximal $(g(I), I)$) et d'intervalles non discriminants (l'union des intervalles se retrouve dans un concept différent du concept maximal) afin de favoriser les intervalles non discriminants. Le choix dans ce cas n'exclurait pas le parcours de toute une partie du treillis de Galois. En effet, lorsque l'on doit choisir entre des intervalles discriminants, la prise de décision est délicate car exclusive, en revanche pour des intervalles non discriminants, la prise de risque est plus faible.
7. La proportion d'intervalles de X_i inférés par rapport à l'ensemble $|X_i|$, c'est-à-dire le pourcentage d'intervalles qui sont validés par le choix de i alors qu'ils ne sont pas en relation avec les valeurs de l'objet à reconnaître.

3.4 Reconnaissance itérative

Nous sommes partis du constat qu'une seule signature pour décrire les données et discriminer les classes ne suffisait pas toujours. De plus, même si la signature est assez discriminante, les intervalles disjoints formés lors de la discrétisation ne garantissent pas une reconnaissance parfaite surtout dans le cas de données bruitées. Il est possible que le bruit perturbe les valeurs des signatures et que ces dernières se retrouvent plus proches des intervalles correspondant à d'autres classes que des intervalles correspondant à leur propre classe. Ainsi, les mauvais intervalles seront validés au cours de la classification, ce qui engendrera des erreurs de classement. L'éloignement entre les classes est donc un critère important à prendre en compte. Plus l'écart entre les intervalles d'un attribut est grand, plus on a de chance de limiter les effets du bruit. Sur une même signature, l'éloignement des intervalles n'est pas identique pour tous les attributs. L'idéal serait donc d'arriver à sélectionner les attributs qui proposent un écart maximal entre les intervalles, comme le font les critères de coupe de la distance maximale et de Hotelling (voir § 3.2.2).

3.4.1 Principe

Nous souhaitons donc profiter de la complémentarité des différents types de signatures et utiliser les attributs les plus adéquats en présence de bruit. Dans le but d'utiliser conjointement une approche statistique et structurelle, nous avons développé un système de classification hiérarchique. Le principe est de reproduire le système de classification simple énoncé précédemment sur plusieurs itérations. A chaque itération, de nouveaux descripteurs (statistiques ou structuraux) des données sont insérés comme signatures en espérant ainsi améliorer la description des données traitées à l'itération précédente et poursuivre le processus de classification avec moins de risque d'erreurs de classement. Nous aurions pu procéder différemment en concaténant plusieurs signatures les unes à la suite des autres, mais le résultat obtenu serait réellement différent. En effet, par cette classification hiérarchique, nous cherchons d'une part, à effectuer une classification par affinage successif, et d'autre part à détecter les lacunes de certaines signatures pour mieux corriger les erreurs qu'elles peuvent générer.

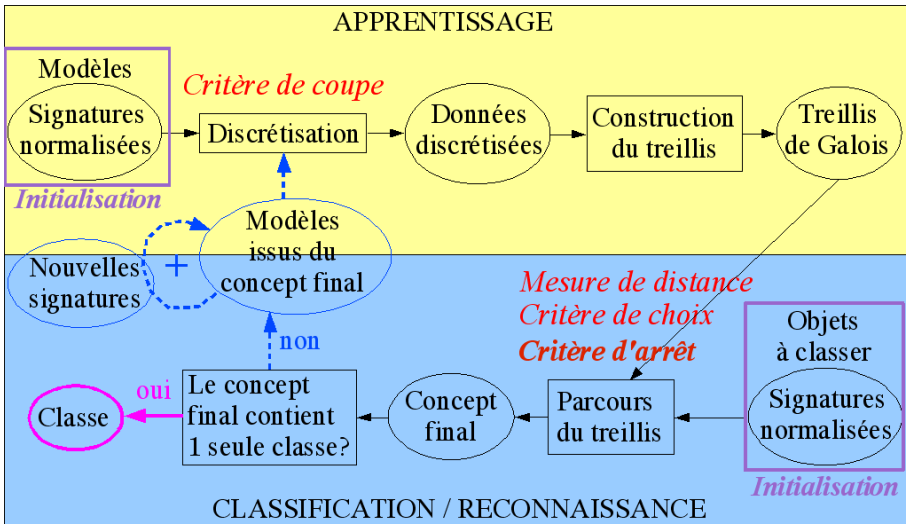
Avec la reconnaissance itérative, il est possible de stopper la discrétisation avant la séparation des classes. Dans ce cas, chacune des classes n'est plus caractérisable de manière indépendante. En revanche, on obtient des paquets de classes qui eux sont caractérisables et peuvent être distingués les uns des autres. On parle alors de reconnaissance "gros grains". Voici une liste non exhaustive de critères d'arrêt de la discrétisation pouvant être utilisés :

1. pouvoir distinguer et caractériser chacune des classes
2. avoir un taux de remplissage des intervalles inférieur à une valeur n
3. pouvoir distinguer au moins n classes

Le critère n°3 correspond à une classification "Top n", c'est à dire qu'au lieu de proposer une seule classe pour la reconnaissance d'un symbole, on propose n classes. Pour pouvoir atteindre l'objectif fixé par chacun de ces critères, il est nécessaire d'avoir une signature suffisamment discriminante.

Le déroulement de la reconnaissance itérative est présenté par la figure 3.8. Le passage d'une reconnaissance simple (en une seule itération) à une reconnaissance itérative, s'effectue en ajoutant un *critère d'arrêt de la classification*, qui permet de stopper la navigation dans le treillis de Galois (voir § 3.4.2). En cas d'arrêt de la classification, le concept courant sur lequel la navigation s'est arrêtée est appelé *concept final*. Deux possibilités s'offrent alors à nous : le concept final ne contient qu'une seule classe et la classification est donc terminée ; le concept final contient plusieurs classes et il faut alors poursuivre la classification, en utilisant une nouvelle signature, pour ce sous-ensemble de classes que l'on a tout de même réussi à distinguer d'autres classes. Le procédé de

FIG. 3.8 – Schéma de description du processus de reconnaissance itératif



reconnaissance est ainsi reproduit à chaque itération.

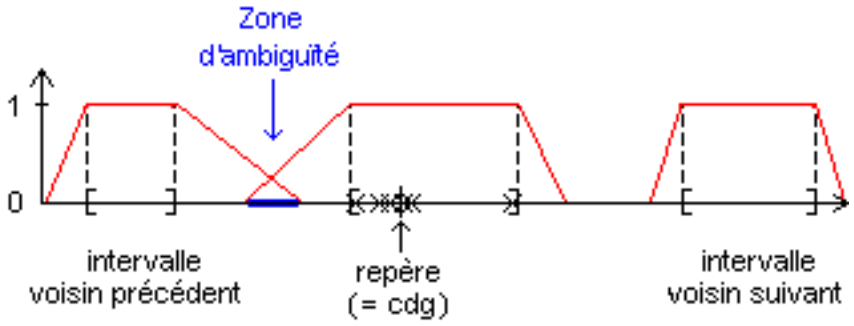
Pour poursuivre la classification, il est nécessaire de construire pour chaque concept final (contenant plusieurs classes) un nouveau treillis de Galois. Chacun des treillis de Galois est construit à partir des objets issus du concept final correspondant, mais ces objets sont décrits par une nouvelle signature. Ainsi, en repartant des objets issus du concept final et non de tous les objets, on prend en compte la classification effectuée à l'itération précédente. Pour constituer la signature de l'itération courante, il est possible d'utiliser des attributs issus d'une nouvelle signature ou ceux d'une signature ayant déjà servi à une itération précédente. Dans ce dernier cas, seuls les attributs de la signature n'ayant pas encore été utilisés sont introduits dans la signature courante. La signature courante contient toujours des attributs issus d'une même méthode de calcul, c'est-à-dire que nous ne mélangeons pas les attributs provenant par exemple de la signature de Radon et de la signature de Zernike.

3.4.2 Critère d'arrêt de la classification

Pour stopper la classification d'un objet, il est nécessaire de fixer un critère d'arrêt. Ce critère doit pouvoir évaluer la confiance en la décision prise à chaque étape élémentaire de classification. Lorsque la confiance est faible, cela veut dire que la décision prise par le système est risquée et qu'il vaut mieux arrêter la classification en l'état plutôt que de poursuivre et risquer de faire une erreur de classement.

Le critère d'arrêt que nous avons mis en place est basé sur la notion d'ambiguïté. Si les intervalles provenant d'un même attribut de la signature sont trop proches, leurs zones floues respectives peuvent se chevaucher, créant ainsi une zone d'ambiguïté (voir Fig. 3.9).

FIG. 3.9 – Zone d'ambiguïté d'un intervalle représenté par un nombre flou



Lors de la reconnaissance d'un objet, si une valeur tombe dans une zone d'ambiguïté, il est risqué de prendre une décision étant donné que cette valeur peut être rattachée à un intervalle comme à l'autre. Dans ce cas, la classification est interrompue.

3.5 Paramétrage pour la reconnaissance de symboles

Rappelons tout d'abord que cette approche de reconnaissance à base de treillis comporte les paramètres suivants :

Apprentissage – *Critère de coupe* de la discrétisation.

Reconnaissance – *Mesure de distance* (pour un attribut donné) entre la valeur prise par le symbole à reconnaître et l'intervalle correspondant.
- *Critère de choix* d'un concept successeur parmi l'ensemble des concepts successeurs candidats.
- (*Critère d'arrêt* de la navigation dans le treillis de Galois seulement dans le cas de la reconnaissance itérative.)

Dans cette partie, nous présentons un ensemble d'expérimentations qui nous ont permis de faire des choix au sein des paramètres de la méthode. Nous nous plaçons dans le cadre de la reconnaissance de symboles et testons les différents éléments paramétrables sur les bases GREC (voir § 2.2).

3.5.1 Signatures

Nous avons souhaité comparer l'efficacité des signatures statistiques et structurelle présentées précédemment (voir chapitre 2). En réalité, on ne peut évaluer que leur adéquation au système de reconnaissance, étant donné qu'elles sont insérées au sein d'une chaîne de traitements, où il peut y avoir des corrélations entre les différents éléments utilisés. Nous avons effectué ces tests sur un ensemble d'apprentissage composé de 8 classes (10 symboles appris par classe). Les taux de reconnaissance obtenus sont présentés dans la figure 3.10, et les tailles de treillis correspondantes dans la figure 3.11. Il faut préciser que la signature structurelle n'a pas été assez discriminante pour obtenir la séparation totale des classes. Le taux de classification indiqué a été obtenu en ne comptabilisant que les symboles reconnus dans un concept final ne contenant qu'une seule classe.

FIG. 3.10 – Taux de reconnaissance obtenus pour les différentes signatures

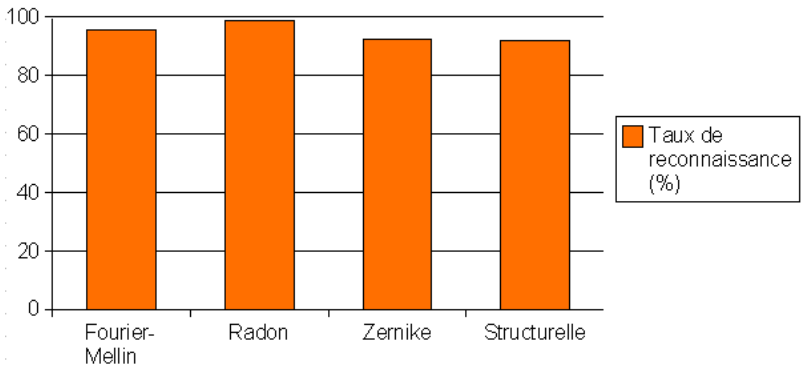
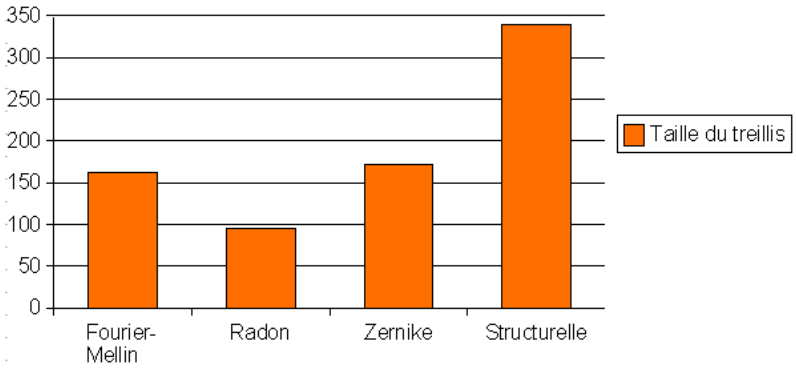


FIG. 3.11 – Tailles des treillis obtenus pour les différentes signatures



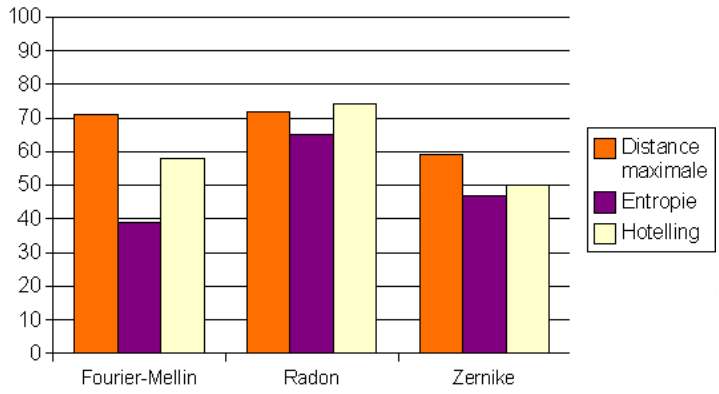
Si l'on compare les treillis obtenus sur le même ensemble de symboles d'apprentissage mais décrits par différentes signatures, on constate que la taille du treillis est variable.

Intuitivement, une signature efficace, propose un treillis de petite taille, car les objets d'une même classe sont regroupés dans les mêmes concepts (voir dans le même concept). En effet, la signature permet de différencier les objets de classes différentes tout en permettant un regroupement des objets d'une même classe. Un treillis de petite taille signifie qu'il y a peu d'objets isolés (sinon, il y aurait plus de concepts). Ainsi, la signature de Radon offre des taux de reconnaissance intéressants et une taille de treillis relativement faible.

3.5.2 Critère de coupe de la discrétisation

Dans cette expérimentation, nous avons choisi de tester l'adéquation des critères de coupe à notre système de reconnaissance. Les critères de coupe testés sont ceux présentés dans la partie discrétisation (voir § 3.2.2) : la distance maximale, l'entropie et le coefficient de Hotelling. Pour évaluer ces trois critères, nous avons utilisé deux jeux de données issus de la base GREC 2003 (voir § 2.2.1), appelés "classes 1 à 10" et "classes 11 à 20". L'apprentissage est réalisé avec un symbole modèle par classe (soit 10 symboles pour chaque jeu de données) et la classification sur 90 symboles bruités par classe (soit 900 symboles bruités pour chaque jeu de données). Les figures 3.12 et 3.13 présentent les taux de reconnaissance obtenus par le treillis de Galois et les figures 3.14 et 3.15, la taille du treillis, pour chacun des trois critères de coupe, respectivement pour les jeux de données "classes 1 à 10" et "classes 11 à 20" et en utilisant les signatures de Fourier-Mellin, Radon et Zernike.

FIG. 3.12 – Taux de reconnaissance obtenus selon le critère de coupe (classes 1 à 10)



Ces résultats nous poussent à sélectionner, pour la reconnaissance de symboles, la distance maximale comme critère non supervisé et le coefficient de Hotelling comme critère supervisé. En effet, ces deux critères de coupe obtiennent les meilleurs taux de

FIG. 3.13 – Taux de reconnaissance obtenus selon le critère de coupe (classes 11 à 20)

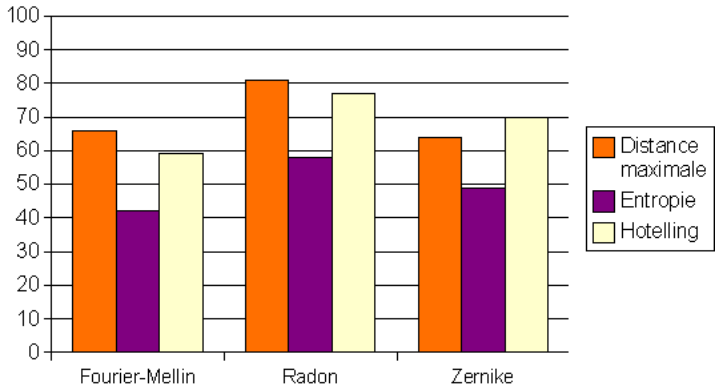
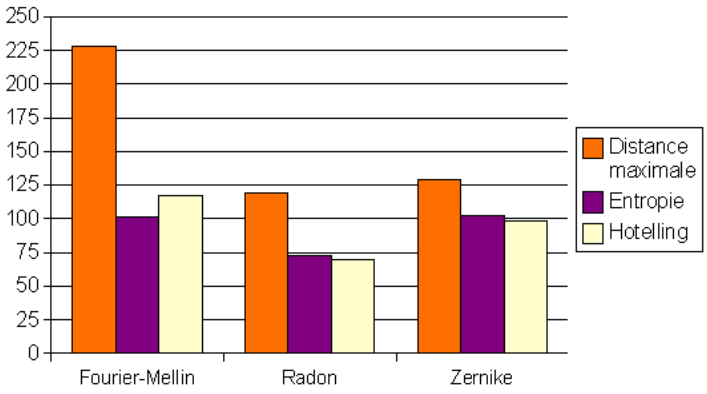


FIG. 3.14 – Taille du treillis selon le critère de coupe (classes 1 à 10)

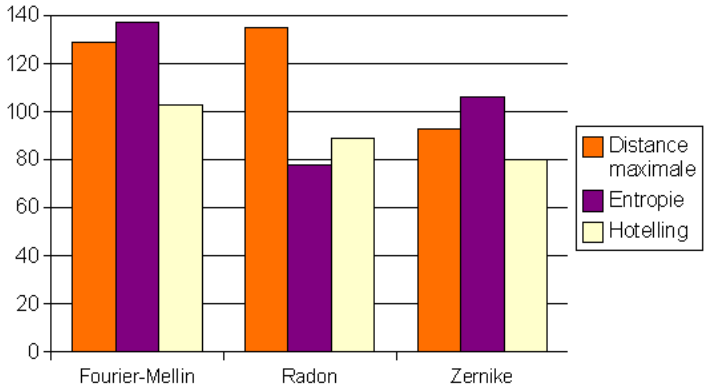


reconnaissance quelle que soit la signature utilisée. D'autre part, le treillis généré à partir du critère de Hotelling contient en général un moins grand nombre de concepts qu'avec les autres critères de coupe. Cette propriété nous intéresse étant donné que plus la structure du treillis est réduite, plus la classification est rapide.

3.5.3 Visualisation des symboles de l'apprentissage

Nous avons souhaité représenter les données de l'apprentissage pour évaluer visuellement la sélection des attributs réalisée par le critère de choix. Le nombre d'attributs de la signature détermine le nombre de dimensions pour la représentation graphique. Lorsque le nombre d'attributs est supérieur à deux, plusieurs possibilités de représentation sont envisageables. Par exemple, pour un faible nombre d'attributs, il est possible

FIG. 3.15 – Taille du treillis selon le critère de coupe (classes 11 à 20)



de représenter les objets en considérant toutes les paires d'attributs les unes après les autres. Mais lorsque le nombre d'attributs est trop élevé, ce procédé devient fastidieux, c'est pourquoi nous avons opté pour une projection des objets dans un espace à deux dimensions. Pour que l'effet des combinaisons des attributs sur les objets restent observables facilement, nous avons utilisé une fonction de projection d'un espace à n dimensions en un espace à 2 dimensions appelée Sammon. Cette fonction est disponible sur le logiciel libre *R* [R]. Cette représentation est intéressante du fait qu'elle offre sur un seul graphique une idée de l'organisation des données au sein d'un concept. Cependant, cette représentation n'étant pas linéaire, il est possible que les distances entre les objets ne soient pas respectées.

Nous souhaitons visualiser l'organisation de 10 classes de symboles de la base GREC 2003 (voir la légende Fig. 3.19) d'un ensemble d'apprentissage décrits par la signature de Radon. Les graphiques suivants représentent les symboles décrits par l'ensemble des 50 attributs de la signature (Fig. 3.17) ou seulement 7 attributs sélectionnés par le critère de Hotelling (Fig. 3.18). En comparant les deux graphiques, on constate que les classes 2, 4 et 5 restent autant mélangées, les classes 3, 6, 7, 8 et 10 restent autant séparées, et les classes 1 et 9 sont mieux séparées avec les 7 attributs sélectionnés par le critère de Hotelling. Globalement, la distinction des classes semble donc meilleure avec les 7 attributs de la signature de Radon qu'avec la totalité.

FIG. 3.16 – Légende

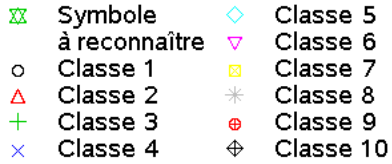


FIG. 3.17 – Symboles de l'apprentissage décrits par les 50 attributs de la signature de Radon

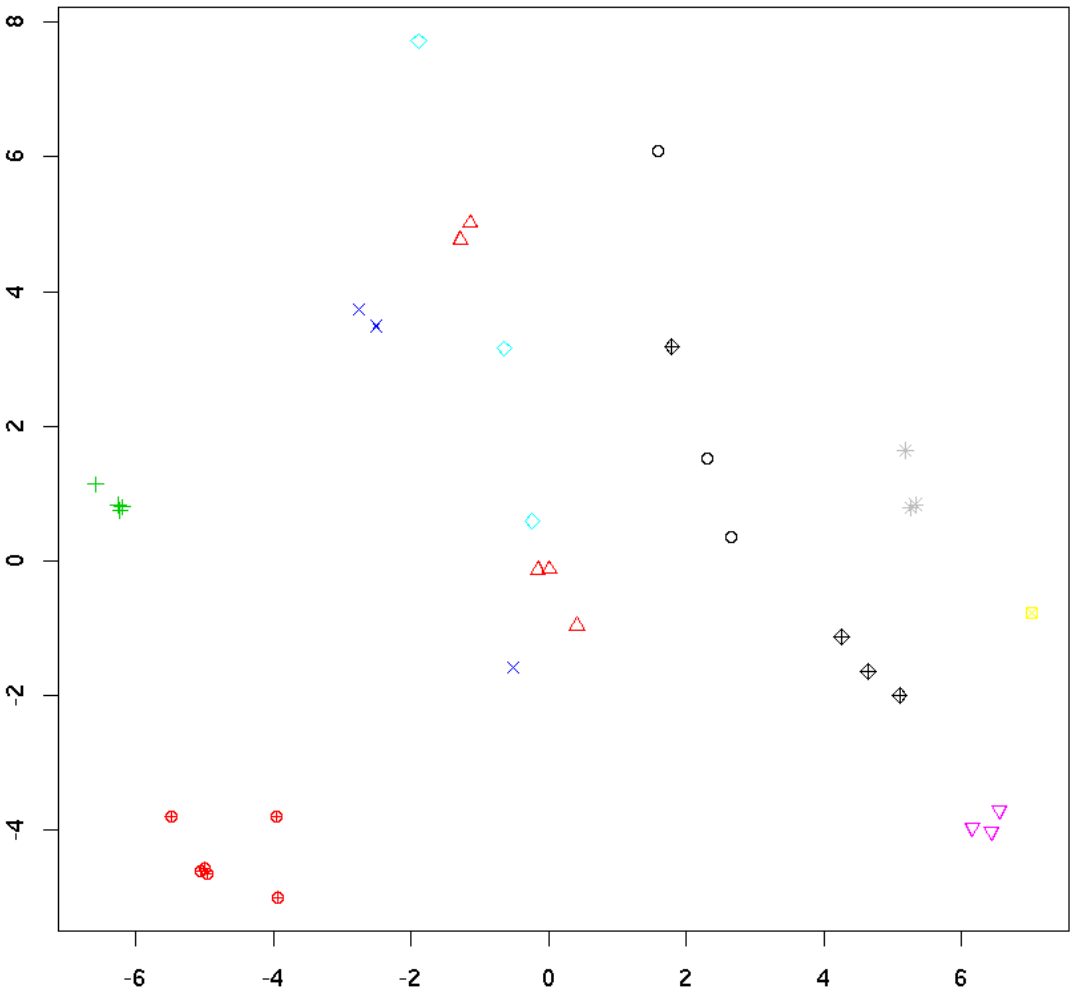
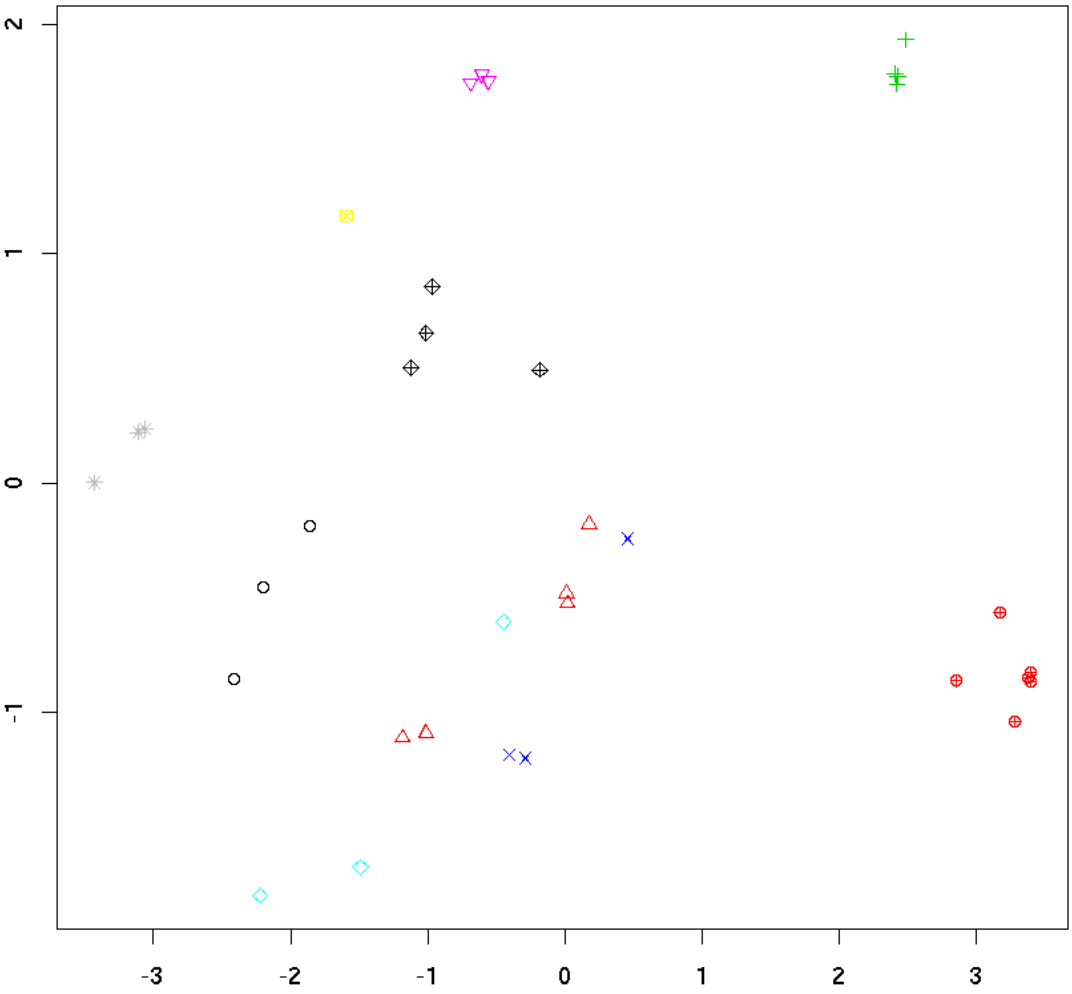


FIG. 3.18 – Symboles de l'apprentissage décrits par 7 attributs de la signature de Radon sélectionnés par le critère de Hotelling



3.5.4 Comparaison du treillis et des systèmes de règles

Dans cette expérimentation, nous avons choisi de comparer les dimensions de certaines structures obtenues (treillis de Galois, règles d'association de différentes bases) en utilisant les symboles de la base GREC 2003 décrits par la signature de Radon.

Rappelons qu'à partir de la table discrétisée, il est possible de construire un treillis de Galois ou des règles d'association. Nous avons choisi de comparer les dimensions et taux de reconnaissance de ces structures dans le cas d'une validation croisée à 5 blocs. La validation croisée [KF82] est une technique d'évaluation de la reconnaissance couramment employée. Il s'agit de partager un ensemble de données en n blocs de taille équivalente. Chacun des blocs sert à tour de rôle d'ensemble de test, et les $n-1$ blocs restants servent d'ensemble d'apprentissage. L'évaluation comporte donc au total n tests qui fournissent chacun un taux de classification. Le résultat final de la validation croisée est donnée par la moyenne des taux de classification obtenus à ces n tests. Dans cette expérimentation, chacun des cinq blocs de la validation croisée contient 91 symboles provenant de 5 classes. La table 3.4 présente le nombre d'attributs $|S|$ de la table discrétisée, le nombre de concepts du treillis, le nombre de règles dans la base canonique Σ_{can} et le nombre de règles dans la base canonique directe Σ_{cd} pour les cinq blocs de la validation croisée. Nous donnons également les taux de reconnaissance obtenus en utilisant le treillis de Galois à titre indicatif.

TAB. 3.4 – Dimensions et taux de reconnaissance obtenus en validation croisée (5 blocs)

	Bloc 0	Bloc 1	Bloc 2	Bloc 3	Bloc 4
$\|S\|$	7	8	8	6	7
Nombre de concepts	20	42	24	25	23
$\|\Sigma_{can}\|$	33	62	32	31	32
$\|\Sigma_{cd}\|$	280	779	724	103	293
Taux de reconnaissance (%)	98,6	99,2	99,2	98,9	99,2

Dans cette expérience, nous souhaitons comparer les même structures que précédemment mais cette fois en utilisant une base aléatoirement générée à la place des symboles de la base GREC 2003. Nous testons l'évolution des dimensions en fonction du nombre d'attributs $|S|$ et du nombre de règles de la base aléatoire Σ. La table 3.5 présente l'évolution de la taille des structures avec un nombre fixe de règles dans la base aléatoire ($|\Sigma| = 15$) en fonction de la taille de S. Quant à elle, la table 3.6 montre l'évolution de la taille des structures avec un nombre fixe d'attributs ($|S| = 7$) en fonction de la taille de Σ.

TAB. 3.5 – Evolution de la taille des structures en fonction de la taille de S ($|\Sigma| = 15$)

$\|S\|$	5	5	5	6	6	6	7	7	7	8	8	8	9	9	9
Nombre de concepts	4	5	3	8	10	7	14	24	10	14	28	37	89	102	60
$\|\Sigma_{can}\|$	4	4	3	6	5	4	11	10	11	10	7	10	12	11	12
$\|\Sigma_{cd}\|$	5	6	3	17	11	6	24	18	39	32	13	31	54	42	47

TAB. 3.6 – Evolution de la taille des structures en fonction de la taille de Σ ($|S| = 7$)

$\|\Sigma\|$	5	5	5	10	10	10	15	15	15	20	20	20	25	25	25
Nombre de concepts	48	51	48	11	21	34	13	8	12	6	11	5	5	3	5
$\|\Sigma_{can}\|$	5	5	4	7	7	6	10	7	7	9	8	6	5	5	6
$\|\Sigma_{cd}\|$	11	8	7	17	13	9	25	19	20	26	14	10	7	5	12

Dans la littérature, les systèmes de règles sont réputés pour être une représentation condensée du treillis de Galois. Or dans la table 3.4, on constate que le nombre de concepts du treillis est moins important que le nombre de règles de la base canonique Σ_{can}, et beaucoup moins important que le nombre de règles de la base canonique directe Σ_{cd}. En revanche, dans les tableaux 3.5 et 3.6 où les données sont générées aléatoirement, le calcul du système de règles reste, comme l'on s'y attendait, plus intéressant. Il est possible d'expliquer cette différence par le fait que l'on oblige les treillis à être co-atomistiques[2] pour la reconnaissance de symboles alors que cette propriété n'est pas vérifiée dans l'expérimentation sur la base aléatoire. D'autre part, les données étudiées sont très différentes. En effet, les données de la base GREC sont particulières car les objets sont décrits par un très grand nombre d'attributs (de très nombreuses signatures peuvent être calculées sur les images), on parle de contexte génomique. En revanche, dans les données générées aléatoirement, le nombre d'attributs est bien moindre, comme c'est le cas habituellement sur les données issues de la fouille de données. Cette différence a certainement une influence sur les résultats observés.

Notons que lorsque le nombre de concepts augmente, la taille des bases diminue, et inversement. Dans le pire des cas, lorsque le nombre de règles est nul ($|\Sigma| = 0$), le nombre de concepts du treillis est maximal : $2^{|S|}$. Ainsi, on observe que pour un nombre de règles de la base aléatoire fixé et une augmentation du nombre d'attributs, le nombre

[2]Dans un treillis co-atomistique, les concepts "finaux" contenant une seule classe, sont toujours couverts par le concept max.

de règles de la base canonique augmente mais reste néanmoins exploitable, tandis que le nombre de concepts explose.

Remarquons également que plus le nombre de règles est élevé, plus les règles risquent d'être redondantes entre elles, et plus la taille de la base diminue entraînant une augmentation du nombre de concepts. On observe dans cette expérimentation que lorsque le nombre d'attributs de la base aléatoire est fixé et que le nombre de règles augmente, le nombre de règles de la base canonique diminue et tout comme le nombre de concepts du treillis de Galois où la diminution est encore plus nette.

En conclusion, le treillis de Galois semble plus approprié que les bases de règles sur les jeux de données GREC. En effet, dans ce cadre, le treillis conserve une taille inférieure à celle des bases de règles. Ce résultat peut s'expliquer par la propriété de co-atomicité vérifiée par les treillis de Galois que nous générons.

3.5.5 Comparaison de la construction du treillis avec / sans génération à la demande

Il s'agit, dans cette expérimentation, de tester l'efficacité de l'algorithme de génération à la demande par rapport à la génération du treillis entier. Comme nous l'avons décrit précédemment (voir § 3.2.3), nous avons la possibilité d'utiliser une extension de l'algorithme de Bordat pour ne construire que les concepts nécessaires pour la classification des objets. Ainsi, il nous est permis de passer outre le problème de l'exponentialité du treillis de Galois dans le pire des cas.

Dans la table 3.7, nous présentons les durées obtenues en apprentissage et en classification ainsi que le nombre de concepts construits dans le treillis avec et sans génération à la demande. L'ensemble d'apprentissage contient 25 symboles modèles (donc 25 classes) et l'ensemble de test contient 10 symboles bruités. Précisons tout d'abord que les taux de reconnaissance obtenus avec et sans génération à la demande sont bien identiques puisque le procédé de classification reste inchangé. D'après ces résultats, on observe qu'en apprentissage, l'algorithme de génération à la demande est très rapide étant donné que seul le concept minimal du treillis est généré. En revanche, la classification est plus longue que dans la génération du treillis entier car les concepts nécessaires sont construits au fur et à mesure de la navigation dans le treillis. Enfin, le nombre de concepts obtenus est bien moins important pour la génération à la demande, du moins pour cette expérimentation.

TAB. 3.7 – Apprentissage avec 25 symboles modèles - Reconnaissance de 10 symboles bruités

	Apprentissage	Classification	Nombre de concepts
Treillis entier	430,2 sec	2 sec	3185
Génération à la demande	0,5 sec	9,8 sec	282

Toutefois, ces résultats invitent à une discussion plus approfondie. Pour une génération à la demande, le nombre de symboles de l'ensemble de test et leur ressemblance vont influer sur la durée du traitement de la classification. Remarquons que ce ne serait pas le cas sans génération à la demande. En effet, pour le premier symbole à reconnaître, les concepts se trouvant sur son chemin de reconnaissance durant la navigation seront construits et stockés. Ensuite, pour la reconnaissance du second symbole, si ce dernier appartient à la même classe que le premier, il y a de fortes chances pour que le chemin de reconnaissance soit identique, et dans ce cas la navigation sera beaucoup plus rapide que pour le premier symbole (aussi rapide que lorsque le treillis entier est généré). En revanche, si le second symbole est d'une autre classe que le premier, il possède certainement des attributs différents, et il faudra construire les concepts supplémentaires, ce qui prendra un peu plus de temps. En résumé, si les symboles à reconnaître sont très diversifiés et utilisent tous les scénarii de classification possibles, le treillis généré à la demande ne sera pas plus rapide, car presque tous les concepts du treillis seront construits. A l'inverse, si les symboles à reconnaître ne correspondent qu'à un seul scénario de classification, le gain de temps offert par la génération à la demande sera très important.

3.5.6 Mesure de distance et critère de choix

Comme mesure de distance, nous avons choisi d'utiliser la fonction d'appartenance des nombres flous présentée précédemment dans la partie 3.2.2 par la figure 3.4. Nous souhaitions principalement que la mesure soit paramétrable par un degré de flou afin de pouvoir modifier la souplesse du système.

Parmi la liste des critères de choix énoncée en partie 3.3.4 rappelons les quatre premiers :

1. Choisir i tel que $d(s, X_i)$ est minimal.
2. Choisir i tel que $|X_i \bigcap I_k| = |\{x \in X_i \bigcap I_k\}|$ est maximal, avec I_k l'ensemble des k premiers intervalles de S rangés par ordre croissant suivant la distance $d(s, x)$.
3. Choisir i tel que $|\{x \in X_i$ tel que $d(s, x) < d_c\}|$ est maximal, avec d_c une constante

que l'on peut assimiler à un degré de flou. En effet, si la mesure de distance employée est une fonction d'appartenance f associée à l'intervalle x, on peut remplacer l'assertion précédente par : Choisir i tel que $|\{x \in X_i$ tel que $f(s, x, \theta) > 0\}|$ est maximal, avec θ un degré de flou.

4. Une combinaison de ces différents critères de choix, et surtout en cas de choix multiples, c'est-à-dire si plusieurs X_i donnent le même résultat.

Nous avons comparé expérimentalement sur les symboles de la base GREC les critères de choix suivants :

Distance minimale Appliquer le critère n°1

K-PPV Appliquer le critère n°2 (avec k = 1).

Combinaison Appliquer le critère n°3 avec $\theta = 0$, ce qui équivaut à un nombre flou rectangulaire dont le support est égal aux bornes de l'intervalle. Puis en cas de choix multiples, appliquer le critère n°3 avec $0 < \theta < 1$. Le support du nombre flou rectangulaire est alors élargi aux bornes floues de l'intervalle de manière proportionnelle à sa taille. Puis en cas de choix multiples, appliquer le critère n°1, ce qui équivaut à un nombre flou symétrique dont le centre est le repère (milieu, centre de gravité ou médiane) associé à l'intervalle.

Nous avons retenu le dernier critère de choix (Combinaison) qui possède l'avantage de ne pas traiter directement les données de manière globale contrairement au critère de "distance minimale"; et de procéder par palier pour une validation progressive des données. En effet, il permet tout d'abord de valider les données qui tombent exactement dans les intervalles, en cas d'égalité, on assouplit un peu le traitement en validant les données proches des intervalles (dans les bornes floues), enfin en cas d'égalité on choisit la plus petite distance moyenne.

3.5.7 Visualisation des étapes de classification

Pour observer le déroulement de la classification, nous avons représenté sur des graphiques en deux dimensions, l'organisation des différents symboles de la base GREC 2003 appris lors de l'apprentissage, ainsi que le symbole en cours de reconnaissance (voir la légende Fig. 3.19). Dans nos exemples, le symbole à reconnaître est toujours représenté par une étoile verte. Chaque graphique donne une représentation d'un concept du treillis avec l'ensemble des objets qu'il contient et les attributs correspondants. Le nombre d'attributs contenus dans le concept détermine le nombre de dimensions sur la représentation graphique. Lorsque le nombre d'attributs est supérieur à 2, nous appliquons la fonction Sammon du logiciel R pour obtenir une projection des données en deux dimensions.

Exemple 1

Dans l'exemple suivant, nous proposons une classification aboutissant à une bonne reconnaissance d'un symbole appartenant à la classe 3 (symbole '+' vert). A l'étape 1 (Fig. 3.20), le symbole à reconnaître est proche des classes 3 et 6, et éloigné des classes 2, 4, 7, 9 et 10. Dès l'ajout d'un second attribut (étape 2 - Fig. 3.21), on observe que le symbole à reconnaître est proche des objets de la classe 3 et que les classes 2 et 7 ne sont plus candidates à la reconnaissance. A l'étape 3 (Fig. 3.22), les nouveaux attributs ajoutés confirment la proximité entre le symbole à reconnaître et les éléments de la classe 3. La classe 9 est éliminée des classes candidates. Ensuite, à l'étape 4 (Fig. 3.23) il ne reste plus que les classes 3 et 10, et il n'y a pas d'ambiguïté concernant l'affectation de la classe au symbole à reconnaître étant donné sa position à la fois proche de la classe 3 et éloignée de la classe 10.

FIG. 3.19 – Légende

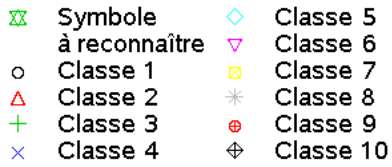
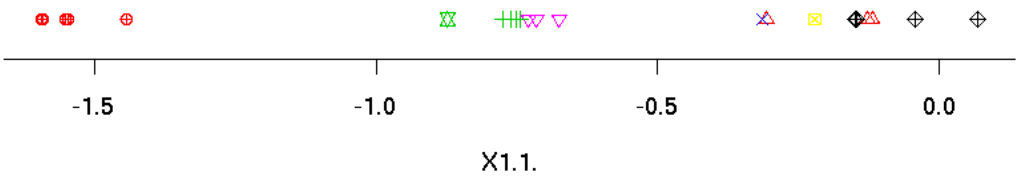


FIG. 3.20 – Exemple de classification d'un symbole de la classe 3 : étape 1



Exemple 2

Dans ce nouvel exemple, nous proposons une classification aboutissant à une bonne reconnaissance d'un symbole appartenant à la classe 2 (triangle rouge). A l'étape 1 (Fig. 3.26), le symbole à reconnaître est proche des classes 2, 9 et dans une moindre mesure de la classe 6. A l'étape 2 (Fig. 3.27), on observe que les objets de la classe 2 sont regroupés en 2 ensembles éloignés. Le symbole à reconnaître est proche d'un ensemble de 3 objets de la classe 2. Les classes 6 et 9 ne sont plus candidates à la reconnaissance. A l'étape 3 (Fig. 3.28), les nouveaux attributs ajoutés confirment la proximité entre le

FIG. 3.21 – Exemple de classification d'un symbole de la classe 3 : étape 2

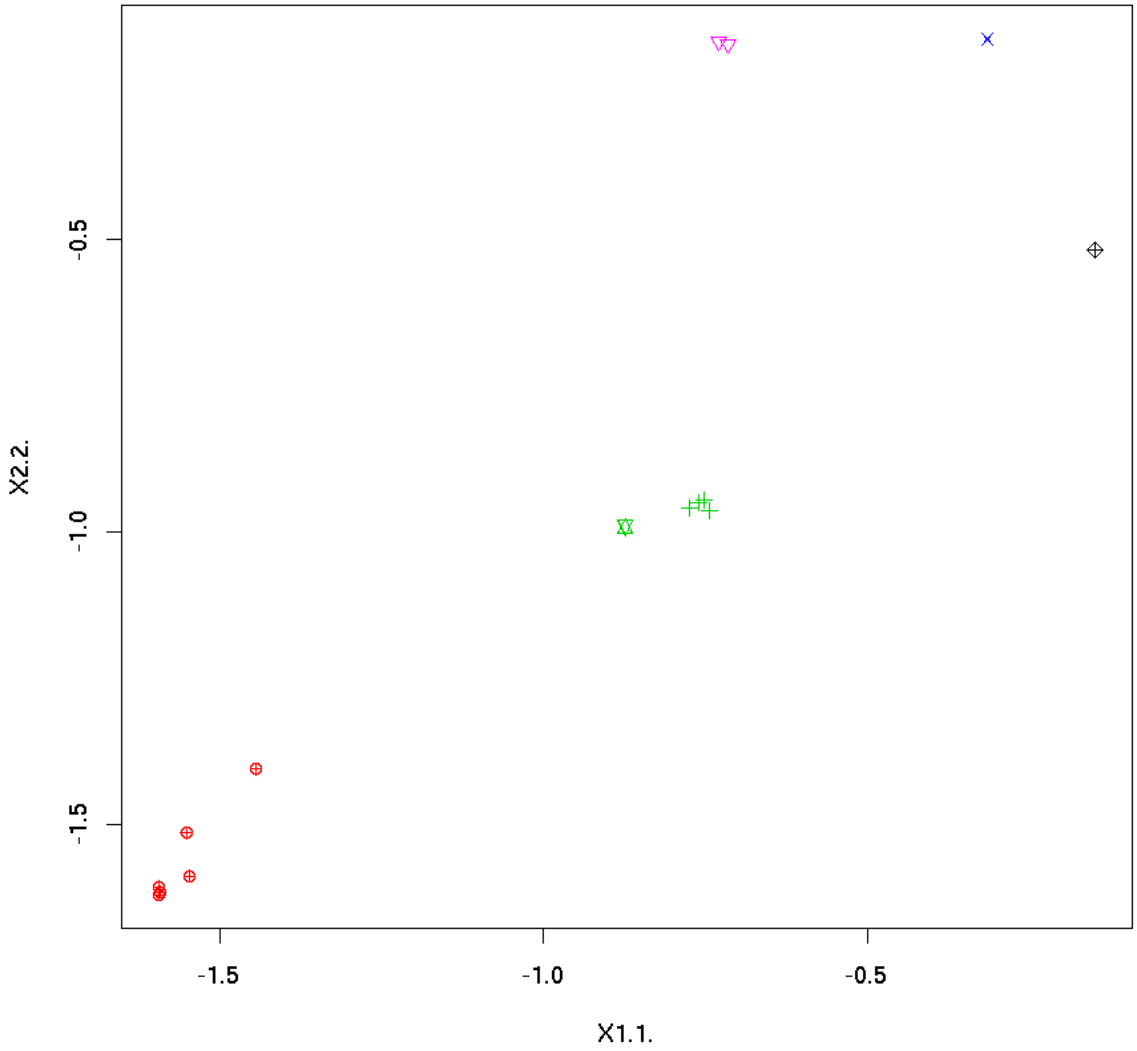


FIG. 3.22 – Exemple de classification d'un symbole de la classe 3 : étape 3

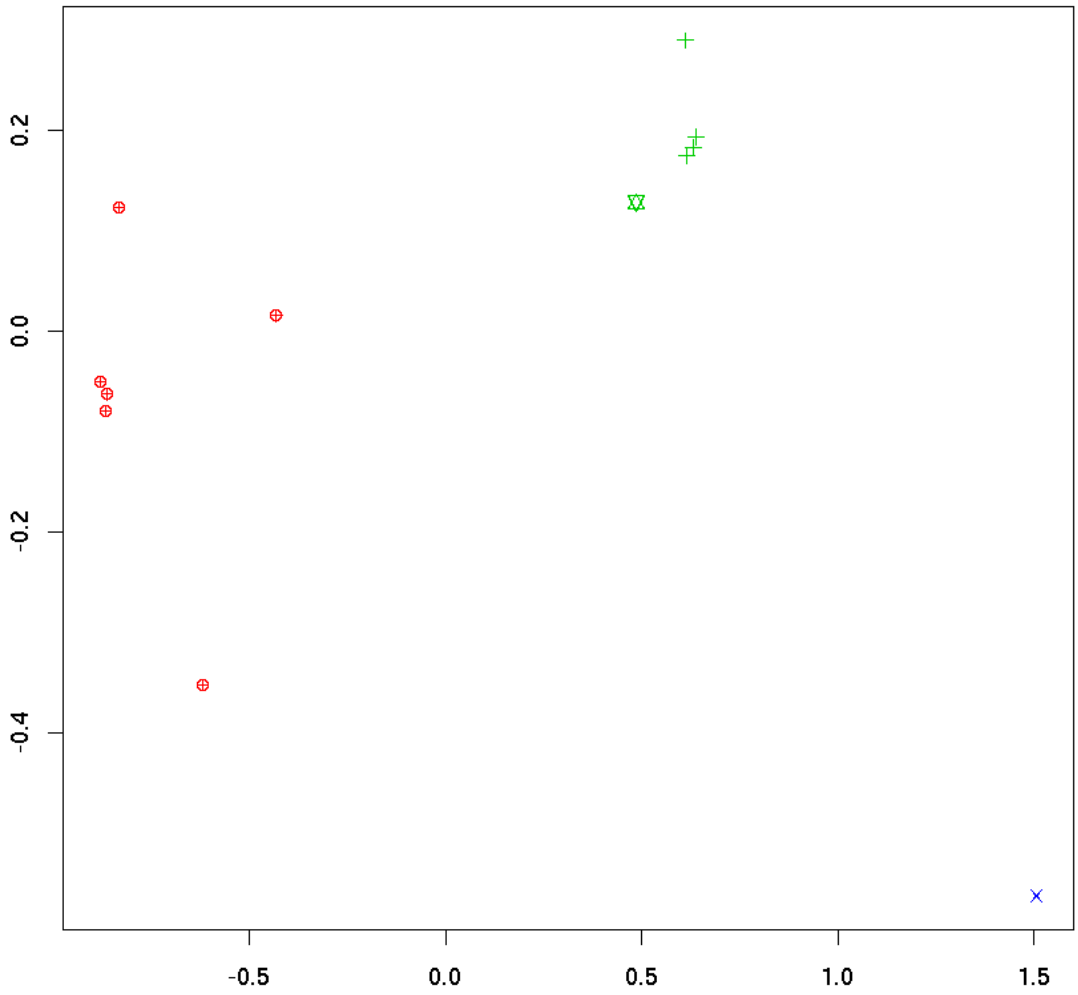


FIG. 3.23 – Exemple de classification d'un symbole de la classe 3 : étape 4

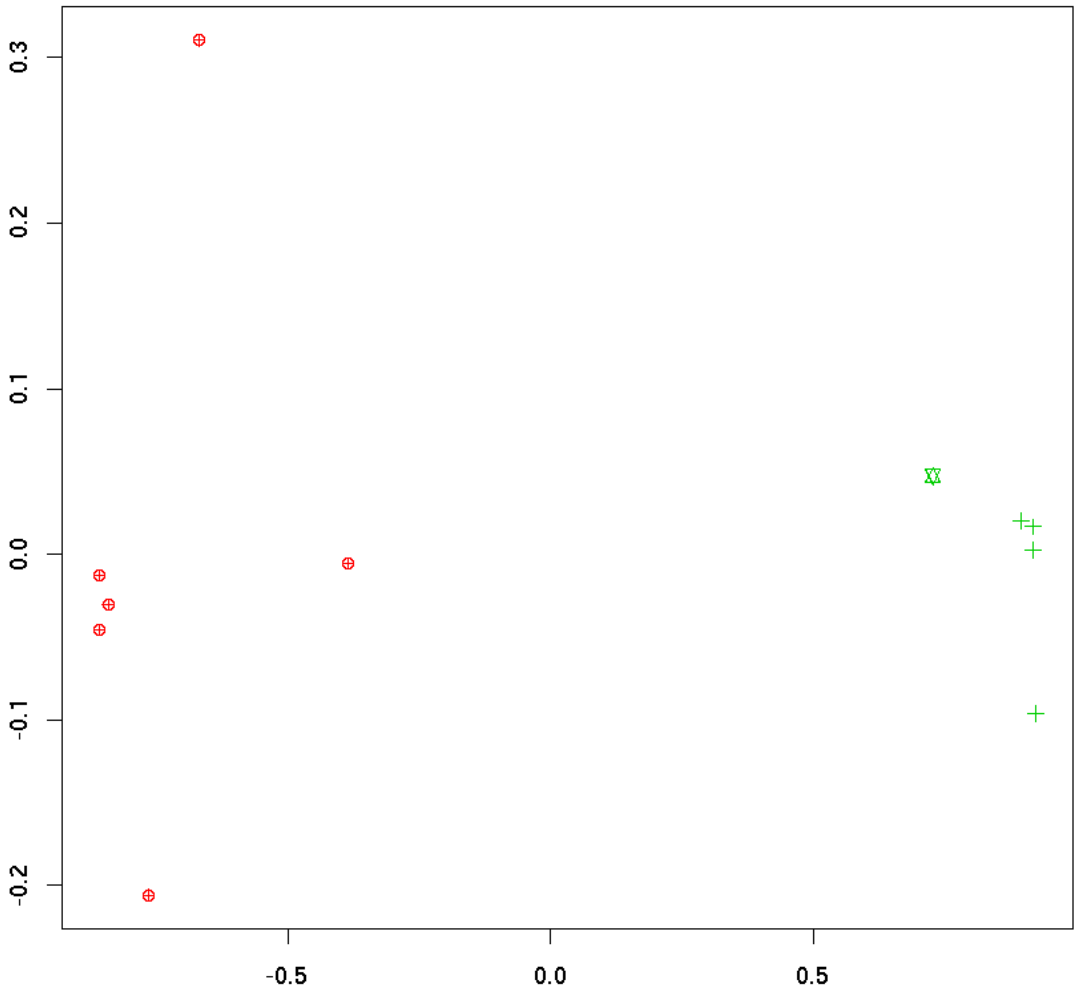
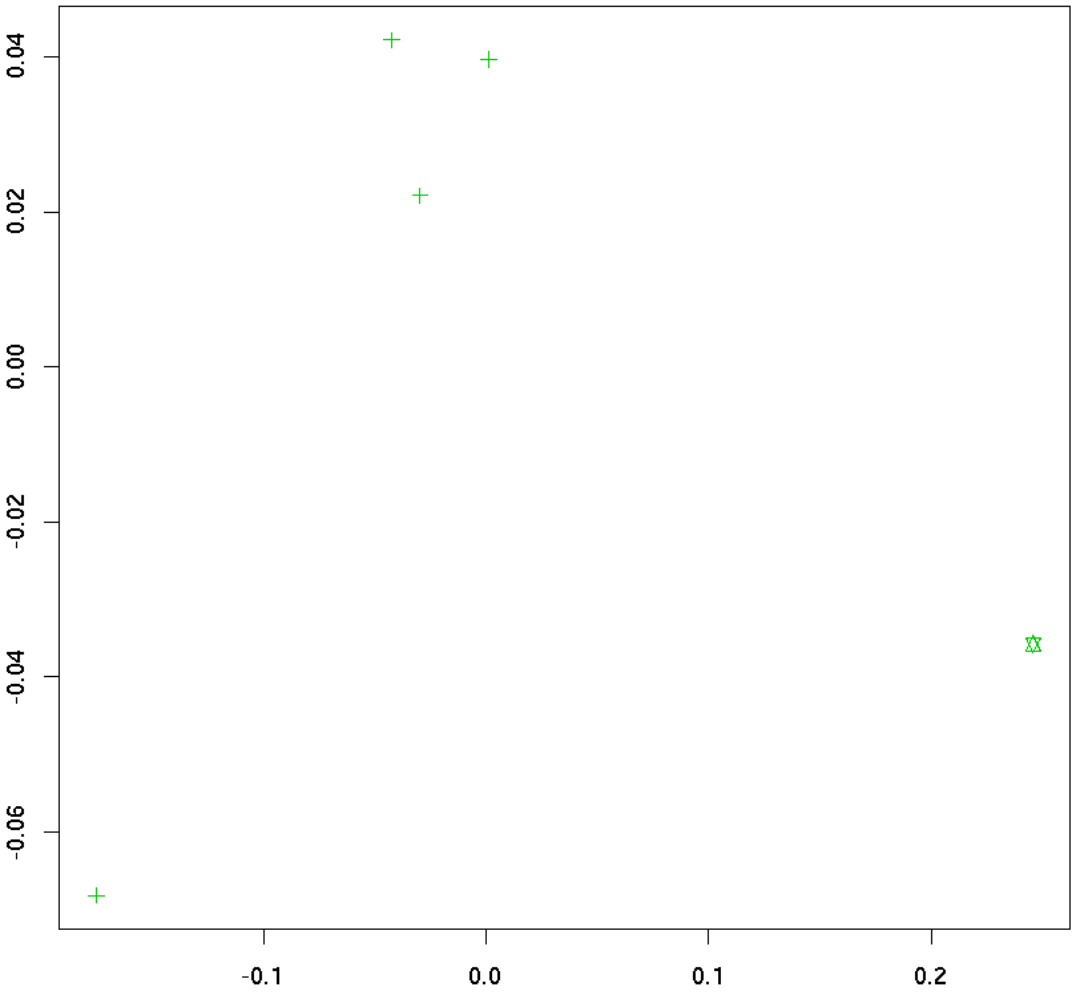


FIG. 3.24 – Exemple de classification d'un symbole de la classe 3 : étape 5



symbole à reconnaître et un élément de la classe 2. La classe 3 est éliminée des classes candidates. Ensuite, à l'étape 4 (Fig. 3.29) il ne reste plus que les classes 2, 4 et 7. Deux éléments de la classe 2, sont éloignés du symbole à reconnaître, le troisième élément de la classe 2 reste très proche et c'est lui qui permet de classer le symbole correctement. L'objet de la classe 7 est bien éloigné, en revanche, celui de la classe 4 est le second objet le plus proche du symbole, il peut donc porter à confusion.

FIG. 3.25 – Légende

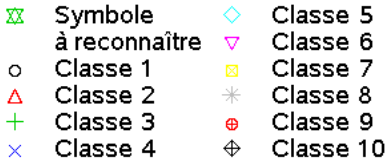


FIG. 3.26 – Exemple de classification d'un symbole de la classe 2 : étape 1

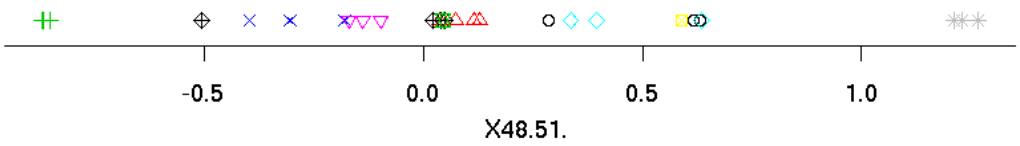


FIG. 3.27 – Exemple de classification d'un symbole de la classe 2 : étape 2

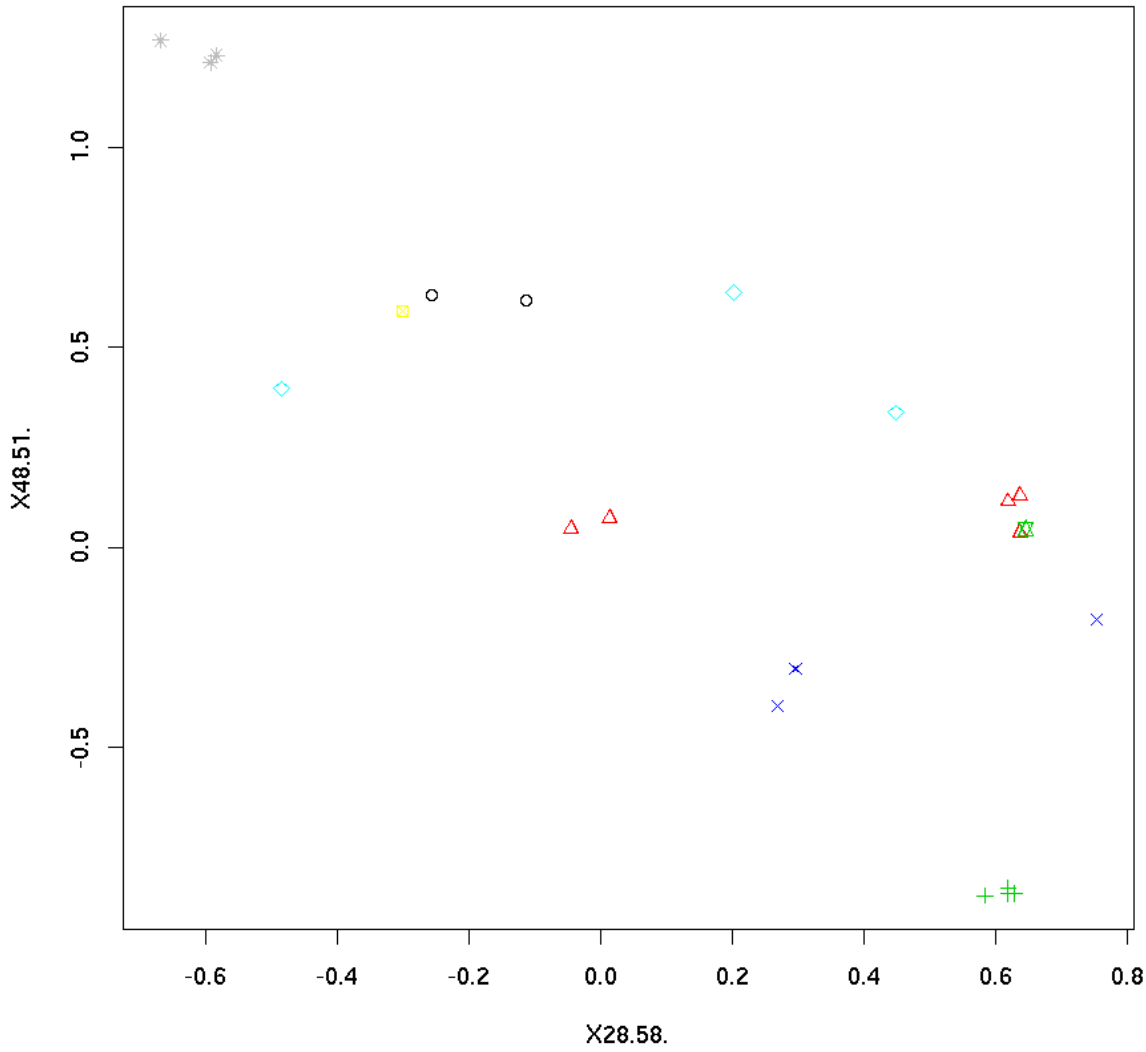


FIG. 3.28 – Exemple de classification d'un symbole de la classe 2 : étape 3

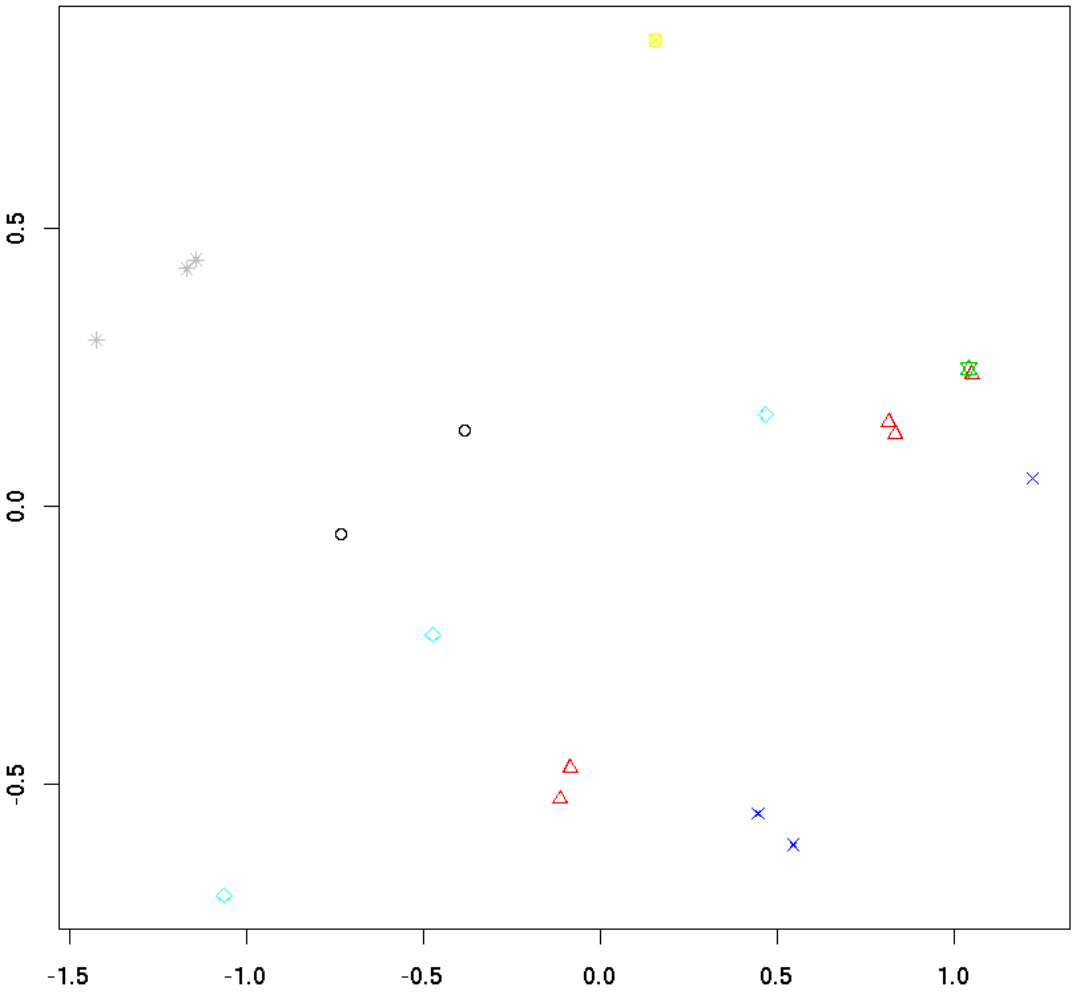


FIG. 3.29 – Exemple de classification d'un symbole de la classe 2 : étape 4

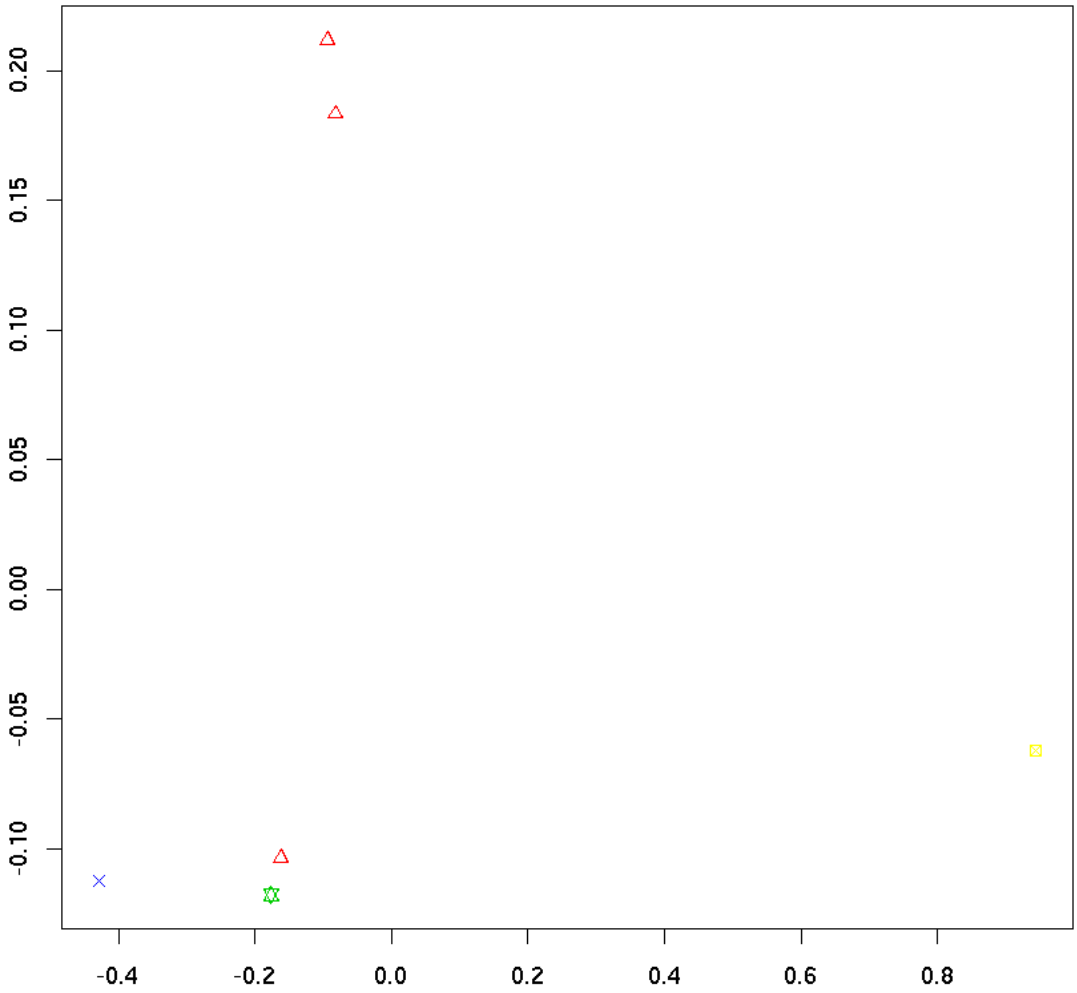


FIG. 3.30 – Exemple de classification d'un symbole de la classe 2 : étape 5

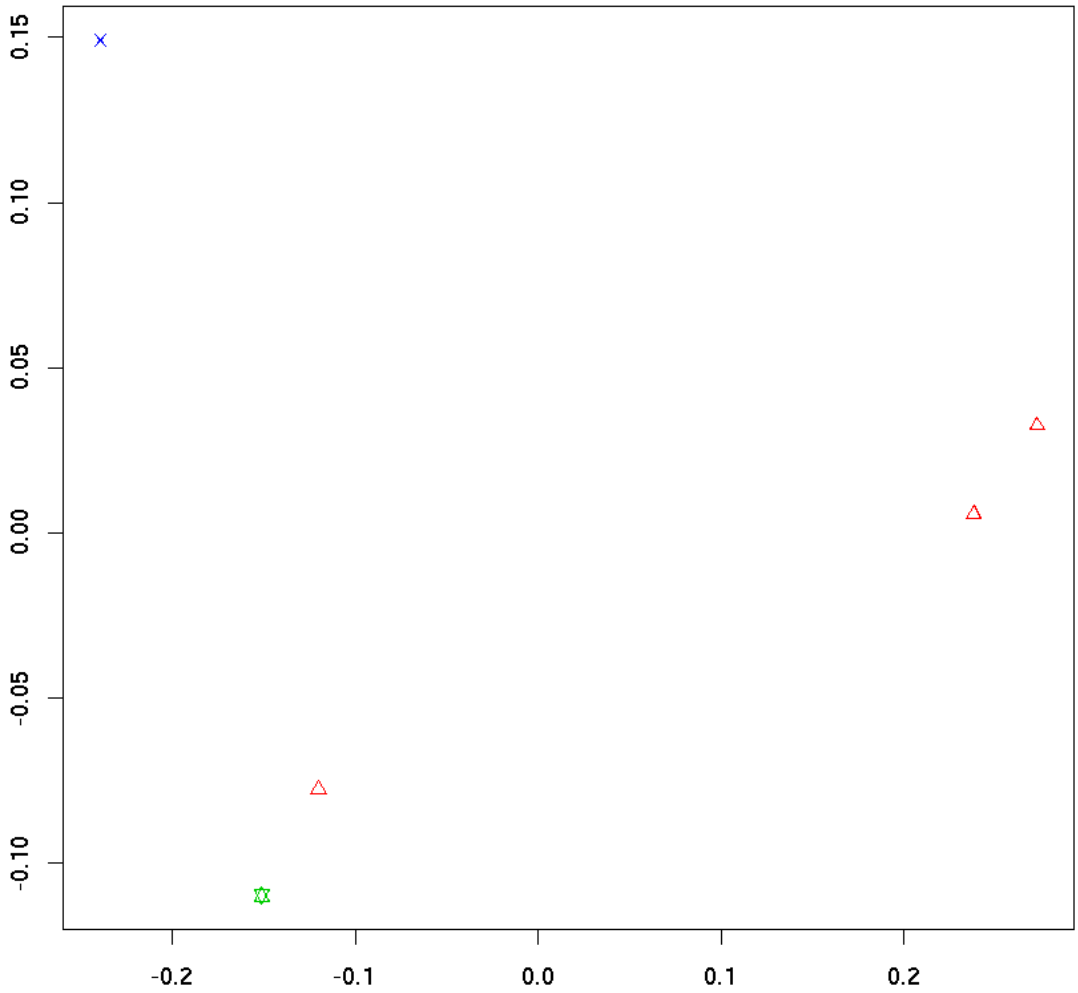
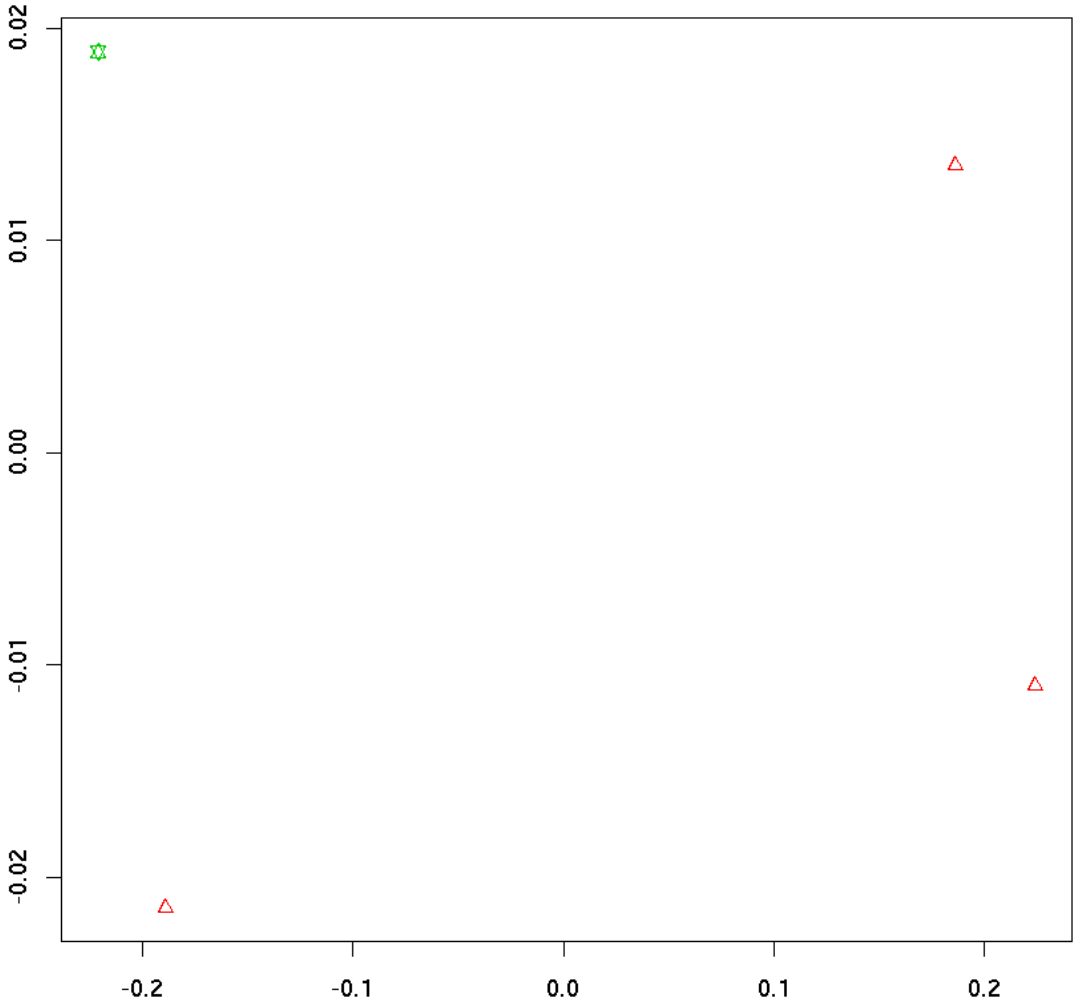


FIG. 3.31 – Exemple de classification d'un symbole de la classe 2 : étape 6



Exemple 3

Dans l'exemple suivant, nous proposons une mauvaise classification d'un symbole appartenant à la classe 4 (symbole '×' bleu foncé). L'erreur de classification est due au fait que les attributs de la signature utilisée n'offrent pas un regroupement intra-classe un espacement inter-classes assez importants entre les classes 2, 4 et 5. De ce fait, les objets se mélangent et le risque de confusion est élevé (voir Fig. 3.34). Jusqu'à l'étape 4, un des objets de la classe 4 reste le plus proche du symbole à reconnaître, mais l'ajout d'attributs supplémentaires à l'étape 5, entraine un rapprochement vers la classe 2.

FIG. 3.32 – Légende

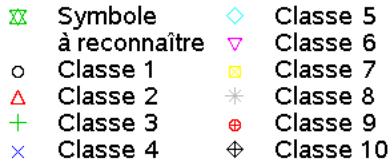


FIG. 3.33 – Exemple d'erreur de classification d'un symbole de la classe 4 : étape 1

FIG. 3.34 – Exemple d'erreur de classification d'un symbole de la classe 4 : étape 2

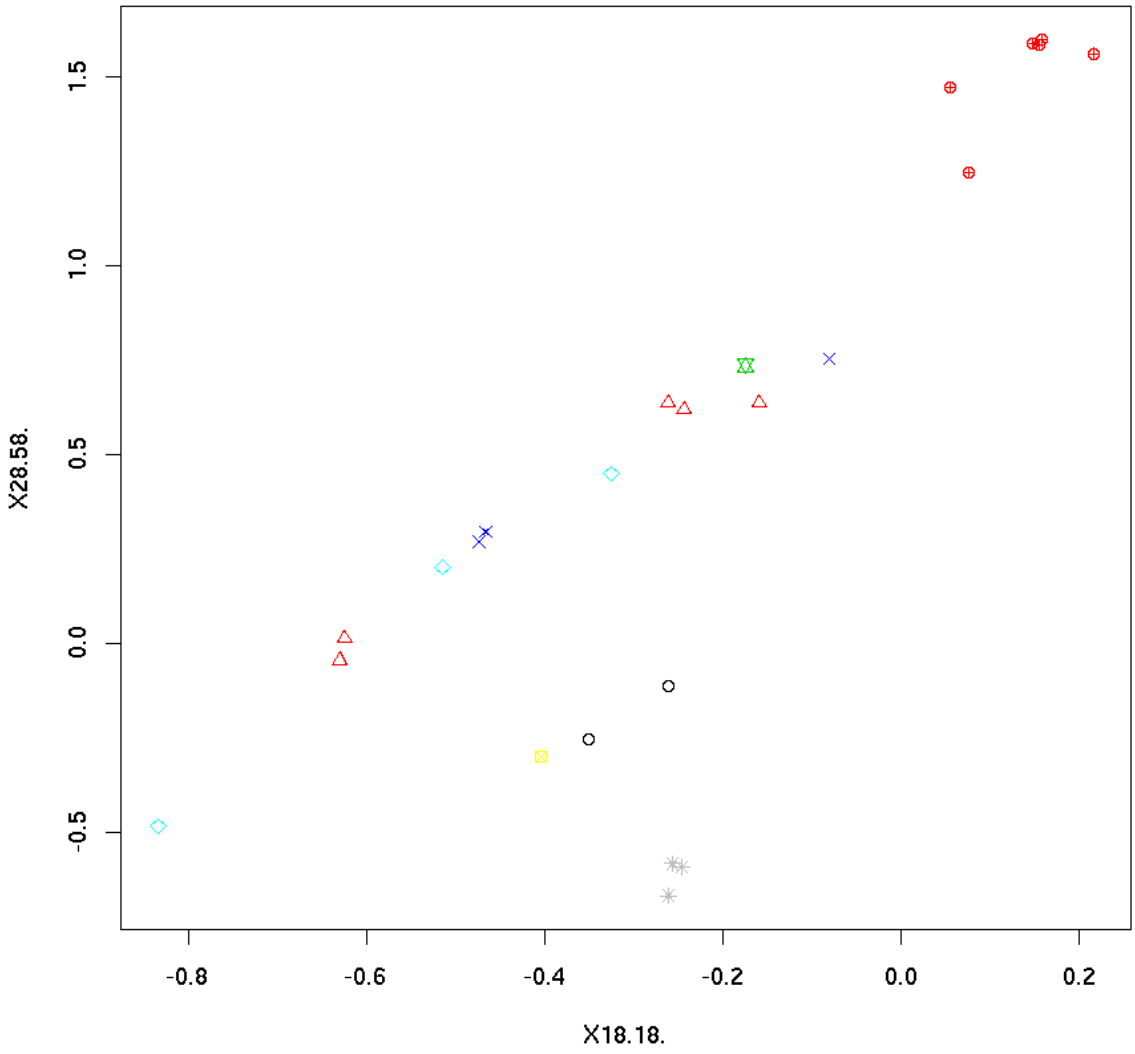


FIG. 3.35 – Exemple d'erreur de classification d'un symbole de la classe 4 : étape 3

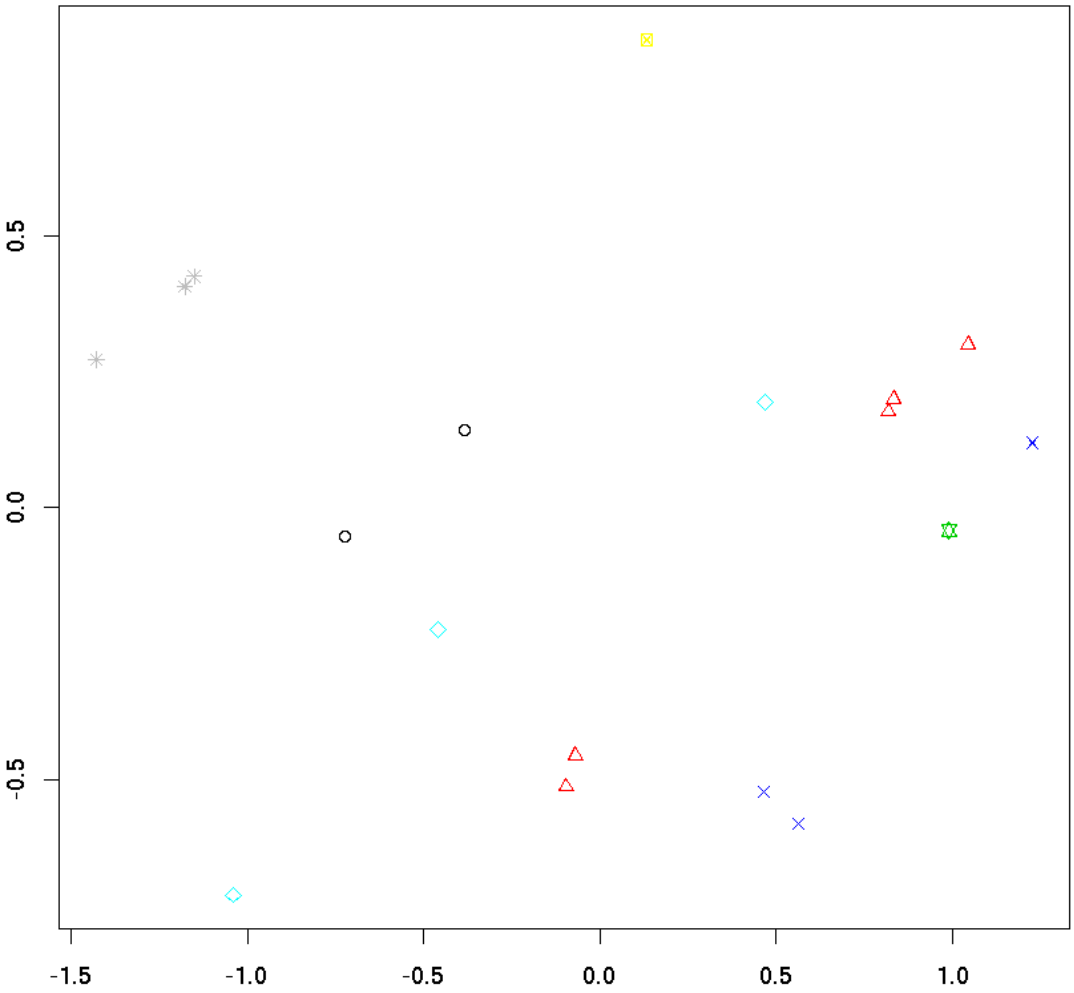


FIG. 3.36 – Exemple d'erreur de classification d'un symbole de la classe 4 : étape 4

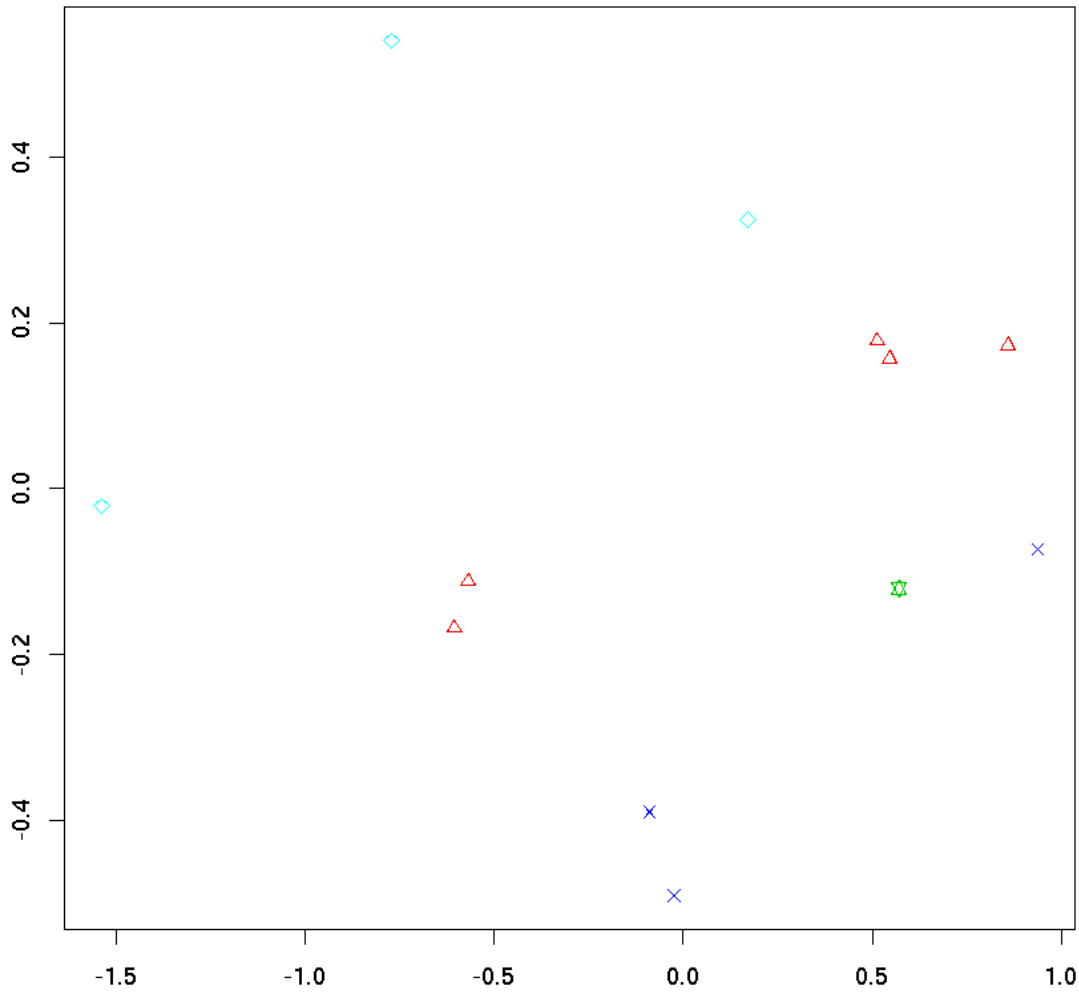


FIG. 3.37 – Exemple d'erreur de classification d'un symbole de la classe 4 : étape 5

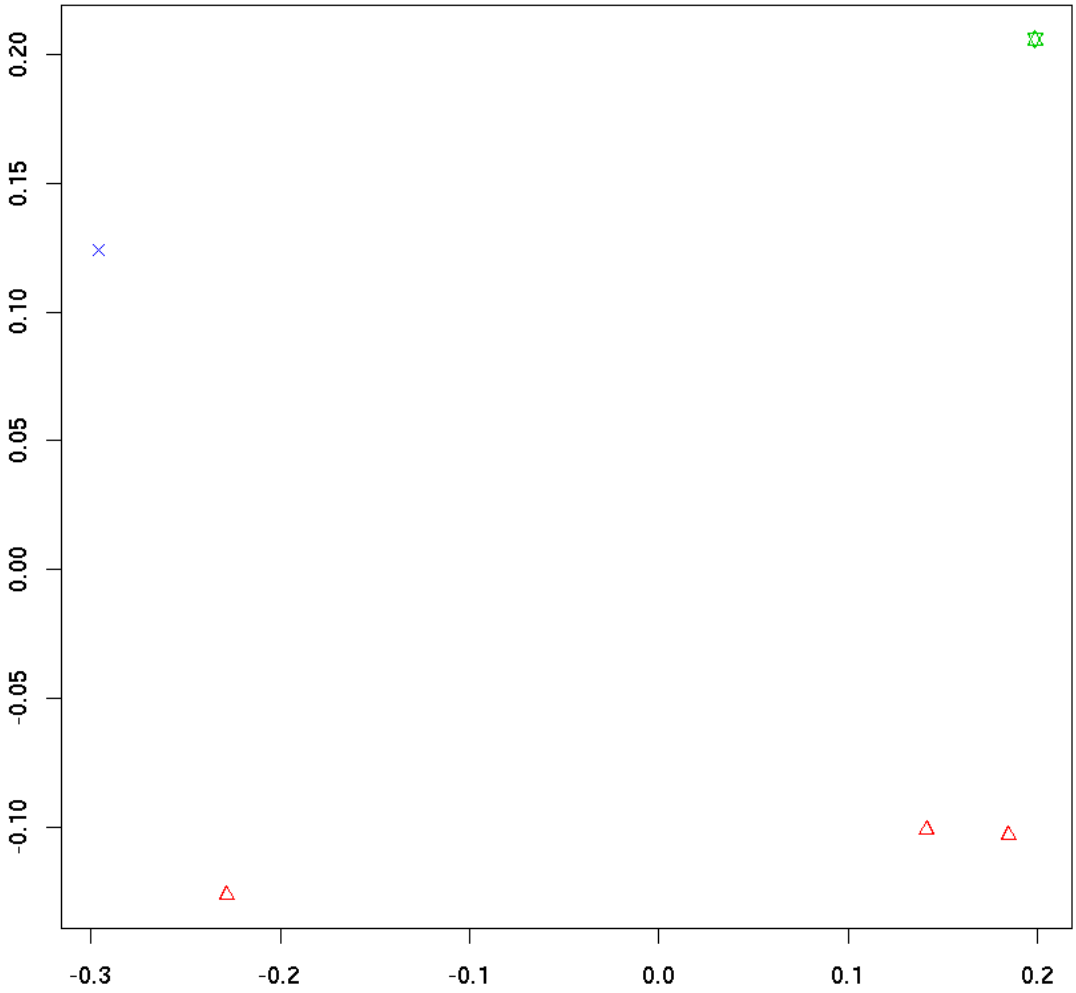
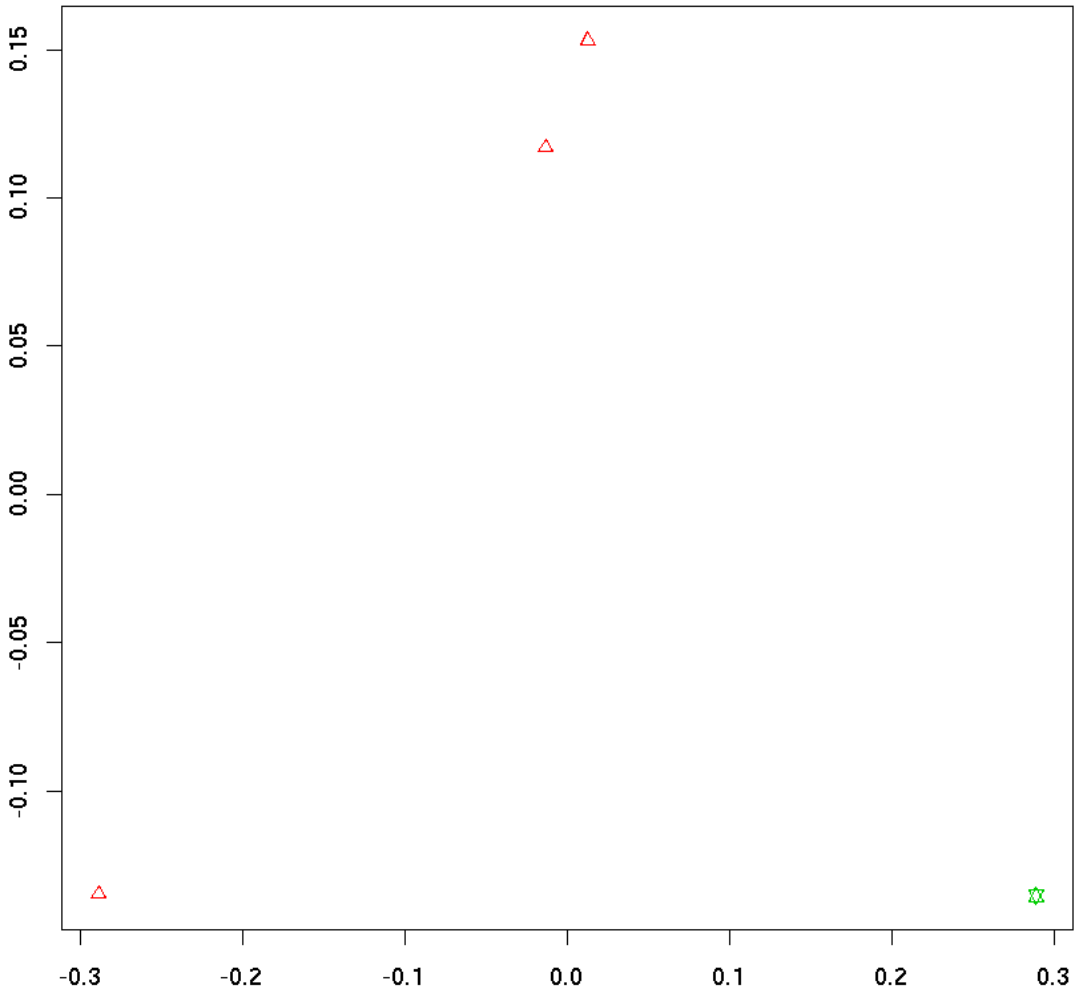


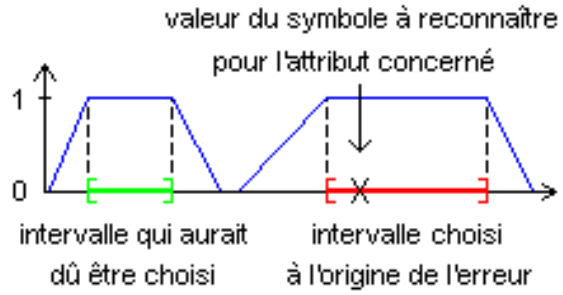
FIG. 3.38 – Exemple d'erreur de classification d'un symbole de la classe 4 : étape 6



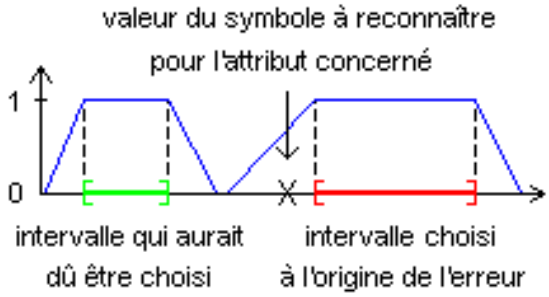
3.5.8 Etude des types d'erreurs de classification

Nous avons recensé les différents types d'erreurs qui pouvaient se produire au cours de la classification (voir Fig. 3.39). Dans ces schémas, l'intervalle vert et l'intervalle rouge font chacun partie d'un concept successeur différent. L'intervalle vert appartient au concept qui doit être choisi car il contient la classe du symbole à reconnaître et l'intervalle rouge appartient au concept qui est en réalité choisi et entraine l'erreur de classification. Lorsque les intervalles sont représentés sur un même axe, ils sont issus du même attribut de la signature (type 1, 2 et 4). En revanche, dans le type 3, les intervalles sont issus de deux attributs différents.

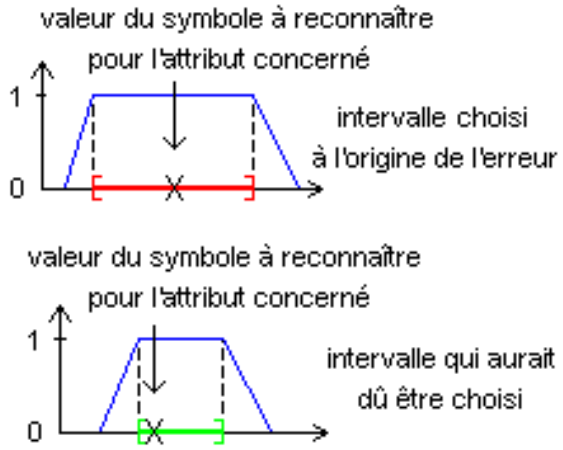
FIG. 3.39 – Types d'erreurs de classification



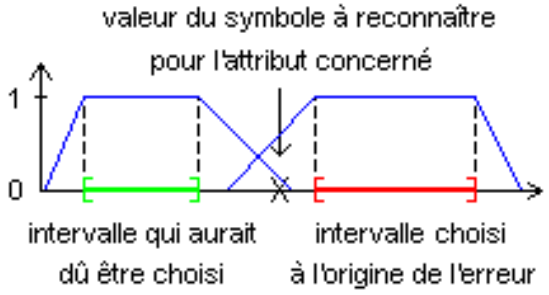
(a) type 1



(b) type 2



(c) type 3



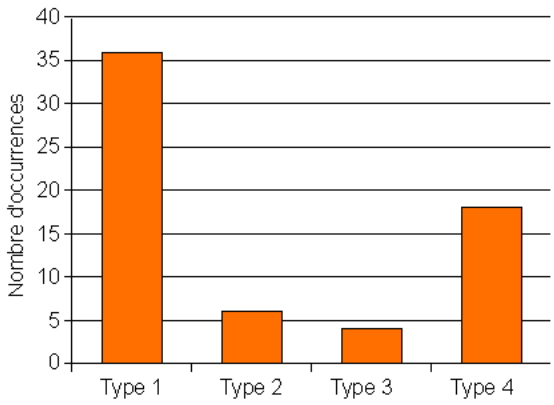
(d) type 4

Les deux premiers types d'erreurs correspondent à une valeur qui tombe dans le mauvais intervalle (type 1) ou à proximité du mauvais intervalle (type 2) à cause du bruit. Les types d'erreurs 3 et 4 sont issus d'un problème d'égalité entre les intervalles. Dans le cas 3, pour le premier intervalle, la valeur du symbole à reconnaître est décalée

à cause du bruit et tombe dans l'intervalle, alors qu'il ne caractérise pas la classe de ce symbole. Pour le second intervalle, la valeur du symbole à reconnaître est correcte et tombe également dans l'intervalle. Nous sommes donc dans un cas d'égalité. Le critère de choix préconise donc d'observer les valeurs qui tombent à proximité des intervalles : sur l'exemple, il n'y a pas de valeurs dans ce cas. Il faut alors calculer la distance minimale entre les valeurs et le milieu de l'intervalle (on peut aussi choisir le centre de gravité ou la médiane comme repère pour l'intervalle). Et c'est sur ce critère de distance minimale que l'on va finalement choisir l'intervalle rouge car la valeur est ici plus proche du milieu de l'intervalle que pour l'intervalle vert. Dans le cas 4, il s'agit également d'un problème d'égalité entre les intervalles. Aucune valeur ne tombe dans un intervalle, donc on passe directement à la seconde étape du critère de choix qui consiste à observer les valeurs à proximité des intervalles. Il y a égalité car la valeur tombe dans les bornes floues des deux intervalles. Il faut alors calculer la distance minimale et c'est l'intervalle rouge qui l'emporte.

Nous avons réalisé une étude sur un ensemble de 64 erreurs de classification recensées lors de la reconnaissance d'un ensemble de symboles de la base GREC 2003. Nous présentons tout d'abord le nombre d'occurrences de chacun des quatre types d'erreurs présentés précédemment (voir Fig. 3.40). Le type d'erreur 1 est de loin le plus fréquent, mais on observe que le type d'erreur 4 est également relativement fréquent.

FIG. 3.40 – Nombre d'occurrences des différents types d'erreurs



Parmi ces erreurs, nous avons voulu observer plus en détail les intervalles mis en jeu dans ces erreurs de classement. Les figures 3.41 et 3.42 présentent respectivement les tailles des intervalles ayant généré des erreurs, et les écarts entre ces mêmes intervalles. Remarquons qu'il n'est pas possible d'évaluer l'écart entre les intervalles pour le type d'erreur 3-1 puisque les intervalles ne sont pas issus du même attribut de la signature.

Les types d'erreurs 1, 2 et 3 semblent concerner des intervalles de toute taille, alors

que le type d'erreur 4 est plutôt réservé aux intervalles très petits ou très grands. Concernant l'écart entre les intervalles ayant généré des erreurs de classification, les types d'erreurs 1 et 2 se produisent plutôt sur des écarts faibles alors que le type d'erreurs 4 se produit plutôt sur des écarts moyens.

FIG. 3.41 – Nombre d'occurrences des tailles des intervalles ayant généré des erreurs

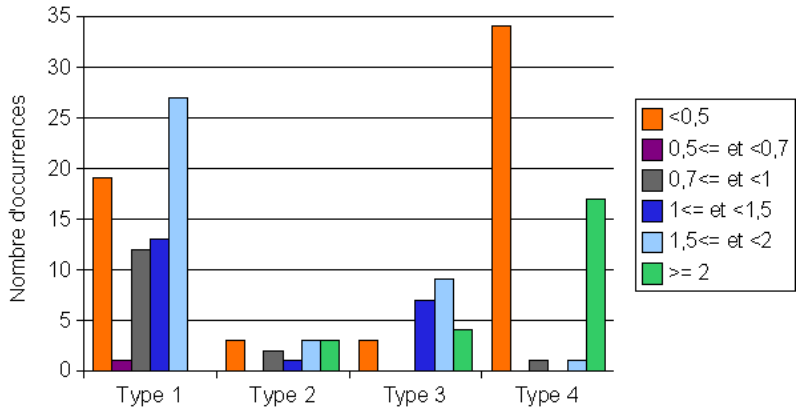
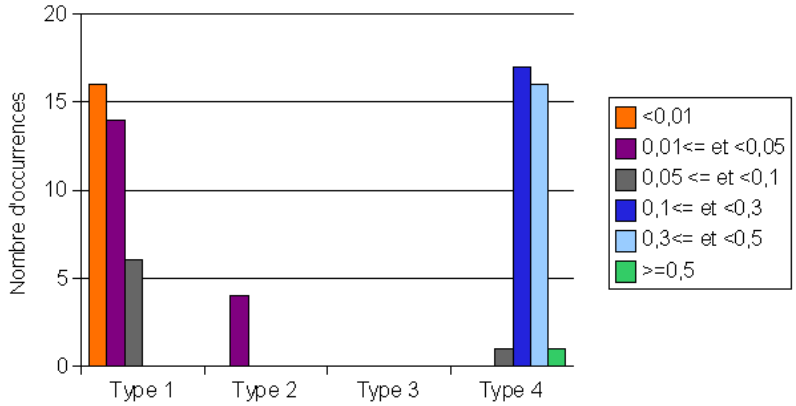


FIG. 3.42 – Nombre d'occurrences des écarts entre les intervalles ayant généré des erreurs



Dans le cadre de la reconnaissance itérative le type d'erreur 4 devrait pouvoir être supprimé étant donné que la valeur est située dans une zone d'ambiguïté (voir § 3.4.2). En revanche, pour les autres types d'erreurs, il y a deux possibilités quant à leur origine : ils peuvent provenir d'un apprentissage pas assez représentatif ou bien d'une faible discrimination de la signature où les descriptions des objets sont trop proches et n'offrent pas assez d'espace entre les intervalles pour assurer une certaine robustesse au bruit. Ces types d'erreurs ne semblent pas pouvoir être supprimés en phase de classification. Ils sont déterminés lors de l'apprentissage et c'est seulement au cours de cette phase que l'on peut agir sur eux.

Cette obervation a également montré que les erreurs étaient en majorité situées assez bas dans le treillis de Galois, c'est-à-dire dans les dernières étapes de la navigation. Le nombre d'objets diminue au fur et à mesure que l'on progresse dans le graphe. Cette faible représentativité des classes dans les dernières étapes du parcours du treillis, peut expliquer les erreurs de classification.

C'est pourquoi, l'idée de mettre en place un système de reconnaissance itératif, lié à ces différentes observations, prend tout son intérêt. En cas d'ambiguïté, le parcours de classification serait stoppé avant d'atteindre une classe, et recommencerait ensuite avec de nouveaux descripteurs.

3.6 Conclusion

Ce chapitre, principale contribution au travail de recherche de cette thèse, décrit la mise en place d'un classifieur basé sur l'utilisation d'un treillis de Galois, que nous avons baptisé NAVIGALA (NAVIgation into GAlois LAttice). Il reprend les grandes étapes usuelles d'un processus de reconnaissance, à savoir l'apprentissage et la classification. Lors de l'apprentissage, les données sont d'abord discrétisées selon un *critère de coupe*, pour ensuite être intégrées dans la construction d'un treillis de Galois. La classification d'un symbole à reconnaître s'effectue alors par une navigation dans le graphe généré, au moyen d'une *mesure de distance* et d'un *critère de choix*.

L'utilisation du classifieur nécessite donc la prise en compte de trois paramètres. Cependant, pour la mise en œuvre d'une reconnaissance itérative, un quatrième paramètre permettant l'*arrêt de la navigation* dans le treillis de Galois doit être ajouté. Cette reconnaissance itérative a pour objectif de stopper la classification, lorsque la prise de décision entre deux ou plusieurs concepts est trop délicate et risque d'engendrer une erreur. Dans ce cas, la poursuite de la classification est réalisée à partir de nouveaux descripteurs des données, donnant lieu à la construction d'un nouveau treillis de Galois.

Enfin, plusieurs expérimentations sont proposées pour préciser le calibrage de ces différents paramètres, et pour mieux comprendre le fonctionnement du treillis de Galois en reconnaissance. Par exemple, l'étude des erreurs de classification nous a montré que tous les types d'erreur ne sont pas forcément corrigeables lors de la classification. En effet, pour certains types d'erreur, seule une amélioration de la discrétisation peut éviter ou limiter leur apparition.

Chapitre 4

Résultats, discussion et perspectives

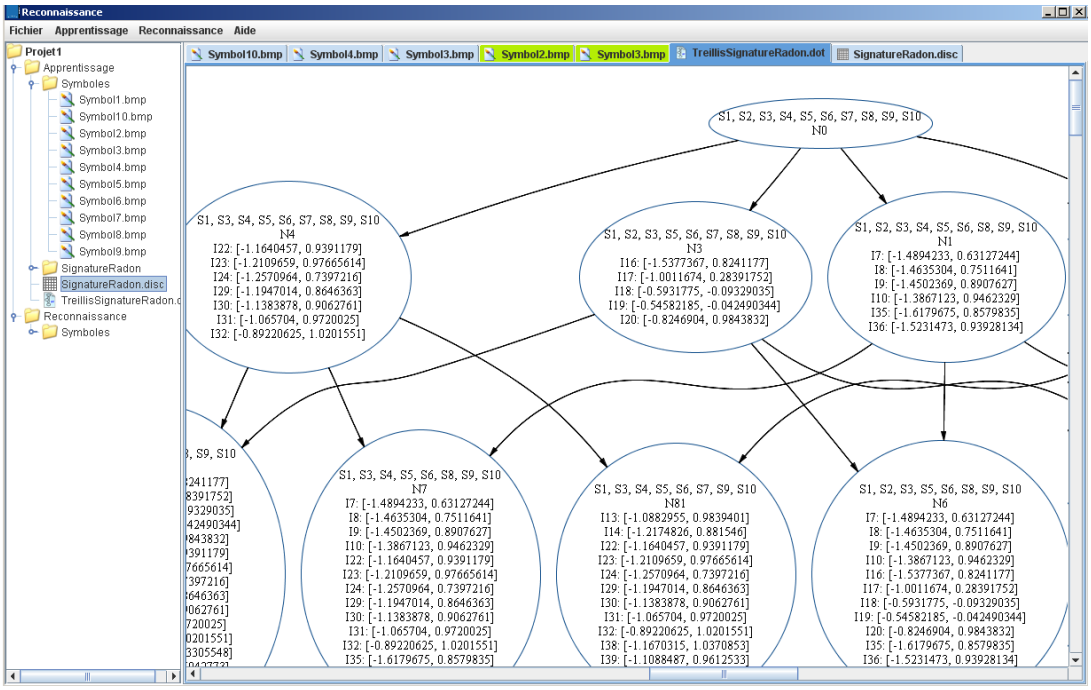
4.1 Introduction

Ce chapitre a pour objectif premier de présenter plusieurs résultats expérimentaux (§ 4.2) et notamment des études comparatives impliquant le classifieur basé sur le treillis de Galois. Nous avons regrouper toutes les fonctionnalités de notre méthode de reconnaissance, NAVIGALA, dans un logiciel du même nom (voir Annexe A). La figure A.4 donne un aperçu de l'interface de ce logiciel. C'est à partir de cette application que tous les tests décrits dans cette étude expérimentale ont été réalisés. Pour terminer ce chapitre, différents éléments de discussion seront abordés concernant les éventuelles pistes de recherche envisageables (§ 4.3).

4.2 Etude expérimentale

Cette étude comporte plusieurs résultats expérimentaux concernant l'utilisation du treillis de Galois comme classifieur (§ 4.2.1. Parmi ces résultats, nous proposons notamment une étude de l'influence de l'augmentation de la taille de l'apprentissage sur la classification et une comparaison entre reconnaissance simple et itérative. Ensuite, cette étude présente également une comparaison théorique assez approfondie avec l'arbre de décision, et une comparaison expérimentale avec des classifieurs usuels en reconnaissance de formes (§ 4.2.2). Enfin, une dernière étude comparative permet d'évaluer le système NAVIGALA par rapport aux autres classifieurs issus de l'analyse formelle des concepts sur des bases de données non dédiées à la reconnaissance de symboles (§ 4.2.3).

FIG. 4.1 – Vue d'ensemble de l'interface et visualisation d'un treillis de Galois



4.2.1 Résultats expérimentaux concernant NAVIGALA

Tout d'abord, nous avons testé l'efficacité du classifieur NAVIGALA pour la reconnaissance les symboles de l'apprentissage. Il s'est avéré que l'expérimentation a confirmé ce que nous supposions. En effet, nous obtenons une reconnaissance parfaite de ces symboles.

A - Augmentation de la taille de l'apprentissage

Cette expérimentation a pour but d'évaluer l'efficacité de notre système de reconnaissance en fonction de la taille de l'ensemble d'apprentissage. En effet, nous avons fait varier le nombre de symboles appris par type de bruit et par classe sur un groupe de 10 classes. Rappelons que la base de symboles GREC 2003 [GRE03] contient 10 symboles de chaque classe pour chacun des 9 niveaux de dégradation. Plus précisément, cette expérimentation a consisté pour chaque type de bruit, à sélectionner aléatoirement n symboles (n allant de 1 à 9) de chaque classe pour les placer en apprentissage, les symboles restants ont quant à eux été placés dans l'ensemble de test. Pour chaque quantité de symboles appris par type de bruit, nous avons effectué 5 tirages (v0 à v4).

Nous présentons dans la figure 4.2 les taux de reconnaissance obtenus à ces 5 tirages, le taux minimum, le premier quartile, le taux médian, le troisième quartile, le taux maximum et le taux moyen. On observe que logiquement les taux s'améliorent en fonction de l'augmentation du nombre de symboles appris par type de bruit. Au-delà d'une certaine taille, les taux ont tendance à stagner voire à régresser légèrement. Quant à elle, la figure 4.3 donne les tailles des treillis de Galois obtenus lors de ces 5 tirages, c'est-à-dire le nombre de concepts générés. La taille des treillis augmente de manière assez régulière au fur et à mesure de l'augmentation de la taille de l'ensemble d'apprentissage.

FIG. 4.2 – Evolution du taux de reconnaissance (en %) en fonction de la taille de l'ensemble d'apprentissage

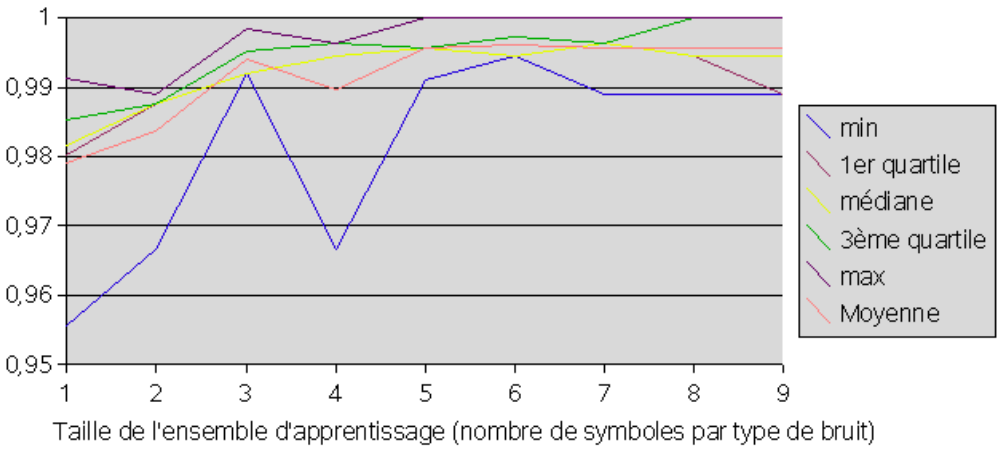
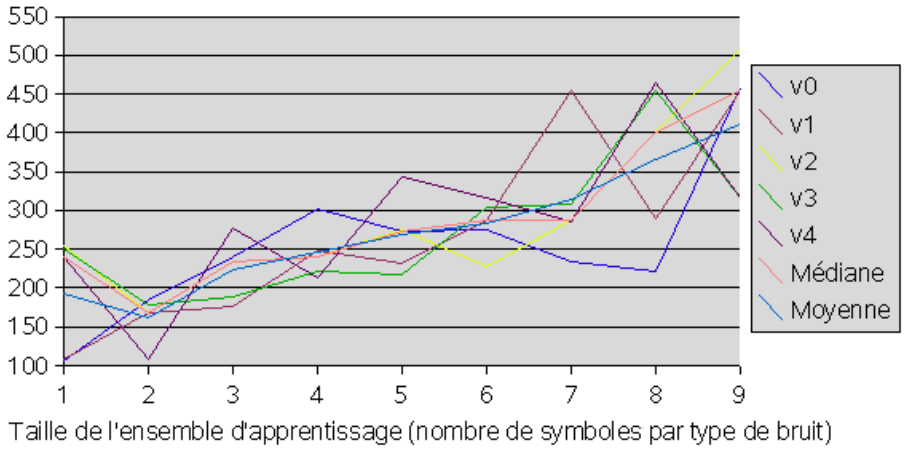


FIG. 4.3 – Evolution de la taille du treillis de Galois (nombre de concepts) en fonction de la taille de l'ensemble d'apprentissage



En conclusion, pour un nombre fixé de classes, l'augmentation de la taille de l'ensemble d'apprentissage améliore la reconnaissance jusqu'à un certain niveau. Au-delà,

les taux de classification restent stables ou régressent du fait du sur-apprentissage. Concernant la taille du treillis de Galois, il est préférable qu'elle reste faible afin de ne pas alourdir le traitement.

B - Comparaison de la reconnaissance avec / sans les symboles modèles dans l'ensemble d'apprentissage

Nous avons vu précédemment que les symboles appris par NAVIGALA étaient ensuite parfaitement reconnus. Dans cette nouvelle expérience, nous avons voulu observer si la présence des symboles modèles, c'est-à-dire non bruités, dans l'ensemble d'apprentissage composé de 10 classes avait une quelconque influence sur la reconnaissance.

Les taux de reconnaissance avec / sans les symboles modèles en apprentissage sont présentés par la table 4.1 et l'évolution de la taille des treillis de Galois avec / sans les symboles modèles en apprentissage est donnée dans la table 4.2. La taille de l'apprentissage correspond au nombre de symboles appris par type de bruit. On constate que les tailles des treillis et les taux de reconnaissance restent la plupart du temps identiques. Cependant quelques différences apparaissent (en rouge) dans cette comparaison.

TAB. 4.1 – Evolution des taux de reconnaissance (en %) obtenus aux 5 tirages (v0 à v4) en fonction de la taille de l'ensemble d'apprentissage sans /avec les symboles modèles

	Taille de l'apprentissage	v0	v1	v2	v3	v4
Sans modèles	3	99,52	99,21	99,84	99,21	99,21
	4	99,63	99,44	99,63	99,44	96,67
	5	99,56	100,00	99,56	99,56	99,11
	6	99,44	99,44	99,72	99,44	100,00
Avec modèles	3	99,52	99,21	99,84	**99,37**	99,21
	4	99,63	99,44	99,63	99,44	**96,85**
	5	99,56	100,00	99,56	99,56	99,11
	6	99,44	99,44	99,72	99,44	100,00

D'une part, on observe par deux fois (en rouge) que le taux de reconnaissance est légèrement amélioré. Dans ces deux cas, nous avons comparé les intervalles générés au cours de la discrétisation et les structures de treillis construites. Il s'avère que les intervalles construits sont presque tous identiques, seules les bornes de deux ou trois intervalles sont très légèrement décalées par les nouvelles valeurs ajoutées des symboles

TAB. 4.2 – Evolution de la taille des treillis (nombre de concepts) obtenus aux 5 tirages (v0 à v4) en fonction de la taille de l'ensemble d'apprentissage avec / sans les symboles modèles

	Taille de l'apprentissage	v0	v1	v2	v3	v4
Sans modèles	3	240	176	233	188	278
	4	302	248	240	222	214
	5	272	232	276	217	344
	6	276	288	228	303	317
Avec modèles	3	240	176	233	188	**211**
	4	302	248	240	222	214
	5	272	232	276	217	344
	6	276	288	228	**298**	**311**

modèles. Les treillis sont pour leur part parfaitement identiques. Ce décalage des bornes des intervalles permet alors de reconnaître plus de symboles.

D'autre part, on observe une réduction de la taille du treillis par trois fois (en rouge). Pour comprendre ce phénomène, nous avons comparé les discrétisations réalisées avec et sans les symboles modèles en apprentissage. Intéressons-nous plus particulièrement au cas v4 (du tableau 4.2) concernant une taille d'apprentissage de 3 symboles par type de bruit qui présente une diminution de 67 concepts avec l'ajout des symboles modèles. Nous avons constaté que les intervalles construits ne sont pas les mêmes. Dès la seconde étape de discrétisation, au lieu de découper l'attribut 24 dans la version sans les symboles modèles, c'est l'attribut 26 qui est découpé une seconde fois. Ce changement va être déterminant pour toutes les découpes suivantes de la discrétisation, car les intervalles générés à partir de l'attribut 26 (I26(26) et I51(26)) sont plus discriminants que ceux obtenus à partir de l'attribut 24 (I24(24) et I51(24)). Dans le treillis issu de l'apprentissage sans les symboles modèles, nous avons observé que l'intervalle I24(24) complété par les intervalles I60(3), I26(26) et I52(48) caractérisent les classes C6, C7 et C8, l'intervalle I51(24) caractérise les classes C1, C2, C3, C4, C5, C9 et C10 et l'intervalle I26(26) complété de l'intervalle I52(48) caractérisent les classes C1, C2, C4, C6, C7, C8 et C9. En revanche, dans le treillis issu de l'apprentissage avec les symboles modèles, l'intervalle I26(26) complété par les intervalles I60(3) et I52(48) caractérisent les classes C6, C7 et C8 et l'intervalle I51(26) complété par l'intervalle I52(48) caractérisent les classes C1, C2, C4 et C9. Les classes sont donc mieux discriminées par les intervalles issus de l'attribut 26 de la signature et le treillis généré est d'autant plus concis.

Avec ces tests, on peut conclure qu'une légère modification de l'ensemble d'apprentissage peut parfois bouleverser la structure du treillis de Galois. On comprend donc que le graphe construit est assez instable. Le fait d'ajouter un symbole pour une classe donnée peut entraîner un agrandissement des intervalles correspondants à ce symbole, et ainsi, modifier la discrétisation et donc la forme du graphe. D'autre part, l'ajout des symboles modèles en apprentissage améliore légèrement la reconnaissance et réduit le nombre de concepts du treillis de Galois.

C - Comparaison de la taille du treillis et des taux de reconnaissance avec / sans appliquer une analyse discriminante

Dans cette expérimentation, nous avons voulu évaluer l'effet d'une transformation des attributs de la signature par une analyse discriminante sur la reconnaissance. Le principe de l'analyse discriminante est de trouver un changement de repère qui offre le maximum de discrimination entre les classes. Les attributs d'origines sont combinés pour fournir un nouvel espace de représentation.

La table 4.3 présente les taux de reconnaissance obtenus avec / sans analyse discriminante appliquée sur les signatures des symboles, et la table 4.4 donne les tailles des treillis correspondant. Nous avons utilisé 10 classes de symboles dans l'ensemble d'apprentissage. On constate que sur l'ensemble des tests effectués, l'utilisation de l'analyse discriminante ne permet d'améliorer les taux de reconnaissance qu'une seule fois. En revanche, elle permet la construction d'un treillis de Galois de taille plus faible dans 8 cas sur 20. En observant les tailles de treillis obtenus sans analyse discriminante, on peut constater qu'elles sont assez régulières quelle que soit la taille de l'apprentissage ; elles sont comprises entre 176 et 344 concepts. Par contre, l'utilisation de l'analyse discriminante apporte une certaine instabilité dans les tailles des treillis qui sont comprises entre 115 et 775. Il semble donc que pour ce jeu de données, l'utilisation d'une analyse discriminante n'est pas intéressante.

En conlusion, contrairement à ce qu'on pouvait supposer, l'ajout d'un pré-traitement basé sur une analyse discriminante des données n'apporte pas toujours une amélioration de la reconnaissance. La chute des taux de classification peut s'expliquer par le fait que l'analyse discriminante ne permet pas d'obtenir une discrimination équivalente pour toutes les classes considérées. Ainsi, le changement de repère peut être plus favorable à la discrimination de certaines classes alors que les autres restent plus difficilement séparables.

TAB. 4.3 – Evolution des taux de reconnaissance (en %) obtenus aux 5 tirages (v0 à v4) en fonction de la taille de l'ensemble d'apprentissage avec / sans analyse discriminante

	Taille de l'apprentissage	v0	v1	v2	v3	v4
Sans analyse discriminante	3	99,52	99,21	99,84	99,21	99,21
	4	99,63	99,44	99,63	99,44	96,67
	5	99,56	100,00	99,56	99,56	99,11
	6	99,44	99,44	99,72	99,44	100,00
Avec analyse discriminante	3	98,10	98,89	97,78	97,46	97,94
	4	96,85	98,33	98,89	98,15	**98,52**
	5	99,33	98,44	99,33	99,11	98,67
	6	98,61	98,89	99,17	98,89	97,22

TAB. 4.4 – Evolution de la taille des treillis (nombre de concepts) obtenus aux 5 tirages (v0 à v4) en fonction de la taille de l'ensemble d'apprentissage avec / sans analyse discriminante

	Taille de l'apprentissage	v0	v1	v2	v3	v4
Sans analyse discriminante	3	240	176	233	188	278
	4	302	248	240	222	214
	5	272	232	276	217	344
	6	276	288	228	303	317
Avec analyse discriminante	3	377	203	**177**	**121**	368
	4	**115**	**212**	**159**	734	399
	5	315	386	665	**210**	775
	6	**241**	359	382	**231**	503

D - Comparaison entre reconnaissance simple et reconnaissance itérative

Nous avons souhaité effectuer une comparaison en terme de taux de reconnaissance et de taille du treillis généré entre la reconnaissance simple et la reconnaissance itérative. Dans ce cadre, nous avons mis en place une étude comparative pour différentes tailles d'apprentissage en faisant varier la quantité de classes observées et le nombre de symboles appris par classe. Pour caractériser les symboles de la base GREC 2003, nous avons calculé la signature de Radon. La reconnaissance simple consiste à utiliser

cette signature une seule fois, alors que la reconnaissance itérative consiste à l'utiliser au maximum trois fois. En effet, pour observer les effets de l'utilisation du mode itératif, il nous a paru intéressant d'observer les résultats avec une même signature utilisée à chaque itération. Le tableau 4.5 propose les taux de reconnaissance et taille de treillis obtenus pour 10, 20 ou 39 classes et pour un ensemble d'apprentissage comportant 1 symbole modèle, 1 symbole modèle + 3 symboles bruités, ou 1 symbole modèle + 9 symboles bruités, pour chaque classe.

On constate, d'une part, que les taux de reconnaissance obtenus en mode itératif sont dans la plupart des cas légèrement supérieurs à ceux obtenus en mode simple. D'autre part, comme l'on pouvait s'y attendre, les treillis grossissent de manière assez conséquente lorsque le nombre de classes en apprentissage augmente. Cependant, cet accroissement du nombre de classes n'implique pas toujours une chute des taux de reconnaissance. Ces résultats peuvent s'expliquer par le fait que les classes ajoutées dans l'ensemble d'apprentissage sont peut-être plus simples à discriminer pour la signature de Radon que celles contenues dans le paquet de 10 classes.

Pour conclure, la détection des erreurs dues aux zones d'ambiguïté lors de l'utilisation du mode de reconnaissance itératif semble fonctionner correctement. Il est vrai, que ce type d'erreur (voir Expérimentation 3.5.8) n'est pas très fréquent, c'est pourquoi l'amélioration des taux de reconnaissance n'est pas très importante. Toutefois, on constate une augmentation de la reconnaissance en mode itératif, qui prouve que l'ambiguïté était réelle et que le fait d'utiliser de nouveaux attributs de la signature de Radon a permis de lever cette ambiguïté.

4.2.2 Comparaison avec quelques classifieurs usuels sur la reconnaissance de symboles

Dans le domaine de la recherche, il est essentiel de créer des ponts entre les différents outils qui peuvent être issus de divers domaines. Ces liens entre les domaines permettent de ne pas redécouvrir les mêmes résultats sous des formalismes différents. Concernant le treillis de Galois, un auteur a choisi en particulier de comparer théoriquement et expérimentalement ce graphe à plusieurs autres classifieurs. En effet, dès 1987, Oosthuizen a étudié le lien existant entre les treillis et les algorithmes génétiques [Oos87]. Il a également observé les points communs avec les réseaux de neurones [Oos89], et plus récemment avec les rough sets [Oos94]. Plus récemment, Mephu Nguifo a montré qu'un sup-demi treillis pouvait être assimilable à l'architecture d'un réseau de neurones multi-couches [MN93]. Ces liens sont importants car ils permettent de mieux situer les

TAB. 4.5 – Comparaison des taux de reconnaissance et taille des treillis entre reconnaissance simple et itérative

Reconnaissance		10 classes	20 classes	39 classes
Simple	Apprentissage : 1 modèle / classe	56,99% 513 symboles reconnus sur 900 (387 erreurs) 88 concepts	56,38% 1015 symboles reconnus sur 1800 (785 erreurs) 647 concepts	61,68% 2165 symboles reconnus sur 3510 (1345 erreurs) 4002 concepts
	Apprentissage : 1 modèle + 3 bruités / classe	84,94% 739 symboles reconnus sur 870 (131 erreurs) 73 concepts	81,66% 1421 symboles reconnus sur 1740 (319 erreurs) 680 concepts	80,63% 2736 symboles reconnus sur 3393 (657 erreurs) 7825 concepts
	Apprentissage : 1 modèle + 9 bruités / classe	97,28% 788 symboles reconnus sur 810 (22 erreurs) 82 concepts	96,97% 1571 symboles reconnus sur 1620 (49 erreurs) 1080 concepts	97,75% 3088 symboles reconnus sur 3159 (71 erreurs) 17460 concepts
Itérative	Apprentissage : 1 modèle / classe	58,22% 524 symboles reconnus sur 900 (376 erreurs) 88 concepts	55,72% 1003 symboles reconnus sur 1800 (797 erreurs) 647 concepts	62,84% 2206 symboles reconnus sur 3510 (1304 erreurs) 4000 concepts
	Apprentissage : 1 modèle + 3 bruités / classe	86,78% 755 symboles reconnus sur 870 (115 erreurs) 73 concepts	81,14% 1412 symboles reconnus sur 1740 (328 erreurs) 680 concepts	81,07% 2751 symboles reconnus sur 3393 (642 erreurs) 7800 concepts
	Apprentissage : 1 modèle + 9 bruités / classe	97,41% 789 symboles reconnus sur 810 (21 erreurs) 82 concepts	97,28% 1576 symboles reconnus sur 1620 (44 erreurs) 1080 concepts	97,81% 3090 symboles reconnus sur 3159 (69 erreurs) 17505 concepts

outils les uns par rapport aux autres et d'engager une réflexion plus poussée sur leurs forces et leurs faiblesses.

Partant de ce constat, nous avons mis en place plusieurs comparaisons expérimentales avec des classifieurs usuels de la reconnaissance de formes, et nous avons plus particulièrement détaillé l'analyse comparative entre l'arbre de décision et le treillis de Galois.

A - Comparaison avec l'arbre de décision

Pour faciliter la compréhension du treillis de Galois et pour réaliser une comparaison avec une autre structure, nous avons choisi un autre type de graphe mieux connu, l'arbre de décision. Ce dernier se définit pour des données discrétisées, et intègre dans sa construction, tout comme le treillis de Galois, les étapes de sélection des intervalles et de classification. La comparaison débute par une description de la phase d'apprentissage pour l'arbre de décision, qui comprend une discrétisation des données, l'expansion de l'arbre et son élagage. Ensuite, vient la phase de classification assez similaire à celle du treillis de Galois. Le formalisme mis en place permet de confronter relativement simplement les deux approches. Pour finir, nous donnons une preuve de l'inclusion de l'arbre de décision dans la structure du treillis de Galois.

A.1 - Description de l'apprentissage

La phase de discrétisation est commune aux deux méthodes. En revanche, la phase de construction diffère.

Pour construire un arbre de décision, il est nécessaire de passer par deux phases successives : l'expansion de l'arbre, puis son élagage. L'étape d'expansion consiste à sélectionner les tests à associer aux différents nœuds de décision et ainsi fixer l'ordre dans lequel ils seront effectués. Il est également important de déterminer la position des feuilles et d'attribuer une classe à chacune d'elles. Cette étape est réalisée sur un ensemble d'apprentissage. Ensuite vient la phase d'élagage, qui est faite à partir d'un ensemble test. Elle permet de réduire la taille de l'arbre. Le principe d'élagage est le suivant : en partant des feuilles de l'arbre, on remonte vers la racine en remplaçant les nœuds de décision par des feuilles. On construit ainsi tous les arbres élagués candidats et pour chacun d'eux, on calcule l'erreur de classification sur l'ensemble test. L'arbre retenu est celui qui minimise l'erreur.

Expansion Il n'est pas envisageable de construire l'ensemble des arbres de décision pour ensuite sélectionner le meilleur, étant donné l'important nombre de combinaisons possibles. Il faut savoir que pour n primitives prenant en moyenne chacune m valeurs, il existe $\prod_{i=1}^{n} i^{m^{n-i}}$ arbres possibles. Par exemple, dans le cas de 4 primitives prenant chacune 3 valeurs, on peut obtenir 55296 arbres différents. On cherche donc à construire l'arbre intelligemment en suivant une méthode descendante. En observant l'algorithme 2 de construction d'un arbre binaire, on peut constater que la phase d'expansion requiert la définition de trois opérateurs :

1. Sélectionner une primitive discriminante et donc un test à associer au nœud
2. Décider si un nœud est terminal
3. Affecter une classe à un nœud

Le premier opérateur nécessite l'utilisation d'une fonction caractérisant le degré de mélange des classes. En effet, pour savoir si une primitive est discriminante, il faut vérifier si elle sépare au mieux les objets de classes différentes, et donc il est obligatoire de pouvoir estimer dans quelle proportion les objets sont mélangés. Généralement, on utilise la fonction de Gini ou bien la fonction entropie. Toutes deux ont la particularité d'atteindre leur maximum lorsque les objets de classes différentes sont mélangés de manière homogène et valent zéro s'il n'y a aucun mélange. On cherche donc à trouver la primitive qui possède le plus grand pouvoir discriminant, c'est-à-dire celle qui minimise la fonction de Gini ou d'entropie.

A partir de ces fonctions, on peut introduire la notion de gain, qui correspond à la réduction d'entropie attendue suite à la partition des objets, suivant une primitive qui peut prendre plusieurs valeurs. Ce calcul du gain est réalisé pour l'ensemble des primitives. On choisira d'affecter au nœud de décision la primitive qui possède un gain maximal.

S'agissant du second opérateur : "décider si un nœud est terminal", il est possible d'utiliser différents critères. Le plus simple est de considérer que l'on crée une feuille lorsqu'il ne reste que des objets appartenant à une même classe. Cependant, ce critère entraîne un partitionnement relativement conséquent des objets de l'ensemble d'apprentissage. Il existe d'autres critères permettant d'arrêter plus rapidement la construction de l'arbre tel que un seuil de profondeur maximal à ne pas dépasser, un seuil minimal d'objets, un faible gain d'information, ou encore un seuil de pureté du nœud en terme de classes au-delà duquel il faut interrompre le partitionnement des objets.

Enfin, pour le troisième opérateur, on affecte la classe majoritaire au nœud.

Nom : `ContruireArbre(E, A)`
Données: E = un ensemble d'apprentissage, A = un arbre (vide au début)
début
  si *le critère de terminaison de l'arbre est vérifié* **alors**
    Créer une feuille du nom de la classe dominante dans E;
    Ajouter la feuille à A;
  sinon
    Sélectionner la primitive la plus discriminante;
    Créer un nœud comportant un test sur la primitive afin de diviser E en 2 sous-ensembles Egauche qui vérifie la primitive et Edroit qui ne la vérifie pas;
    Ajouter le nœud à A;
    ConstruireArbre(Egauche, A);
    ConstruireArbre(Edroit, A);
fin

Algorithm 2: Construction d'un arbre binaire récursivement

Elagage Il est réalisé sur l'arbre obtenu par l'algorithme de construction. Lors de la phase d'expansion, l'arbre est généralement développé jusqu'à obtenir des feuilles pures ; il contient donc un nombre conséquent de nœuds de décision. Sa complexité doit nécessairement être limitée. La phase d'élagage a pour objectif la simplification de l'arbre qui se traduit par la réduction du nombre de nœuds de décision. L'élagage est effectué progressivement en remontant des feuilles vers la racine. A chaque itération, on sélectionne le nœud qui minimise un critère dans le but de le transformer en feuille. Le critère doit être choisi de manière à ce que le bénéfice de l'élagage soit un bon compromis entre la complexité de l'arbre et l'erreur de classification obtenue sur un ensemble test.

Parmi les critères existants, on peut citer :

le Minimal Cost-Complexity Pruning (MCCP) [BFOS84] Il s'agit de sélectionner selon certaines heuristiques une famille d'arbres de décision, et de choisir parmi cette famille, l'arbre de décision T qui minimise la valeur $R_\alpha(T) = R(T) + \alpha C(T)$ où $R(T)$ est le coût d'une mauvaise classification pour T, α est un coefficient de complexité et $C(T)$ est le nombre de feuilles de T.

le Minimum Error Pruning (MEP) [NB86] Pour chaque nœud de l'arbre de décision, on calcule le taux d'erreur maximal acceptable, ainsi que le taux d'erreur engendré par le sous-arbre issu du nœud. Si le taux du sous-arbre issu du nœud est supérieur au taux maximal acceptable par le nœud, alors le nœud est élagé.

le Pessimistic Error Pruning (PEP) [Qui87] Le principe consiste à examiner chaque sous-arbre en considérant le taux d'erreur observé et le nombre de feuilles du sous-arbre. Si le sous-arbre obtient trop d'erreurs de classification, alors il est élagué et

remplacé par l'une de ses feuilles. Il n'y a pas besoin d'un ensemble de validation de l'élagage, mais seulement d'un ensemble d'apprentissage.

le Error-Based Pruning (EBP) [Qui93] Il s'agit d'une amélioration de la méthode précédente (PEP) pour permettre en plus de l'élagage, la greffe de certaines branches de l'arbre. C'est la méthode implémentée dans l'algorithme C4.5.

Lorsqu'un nœud est sélectionné par le critère, il est transformé en feuille. On obtient alors un nouvel arbre qui est conservé pour être comparé par la suite à tous les autres arbres ainsi créés. L'arbre retenu est celui qui possède la plus petite erreur de classification.

Nom : `ElaguerArbre(`A_{max}`)`
Données: A_{max} = l'arbre binaire obtenu par l'algorithme d'expansion
Résultat: l'un des arbres binaires de l'ensemble (A_{max}, A_1, ..., A_n)
début
  $k := 0$;
  $A_k := A_{max}$;
  tant que A_k *possède plus d'un nœud* **faire**
    pour *chaque nœud* n *de* A_k **faire**
      Calculer le critère $c(A_k, n)$ sur l'ensemble d'apprentissage;
    Choisir le nœud n_m pour lequel le critère est minimum;
    A_{k+1} se déduit de A_k en y remplaçant n_m par une feuille;
    k := k+1;
  Choisir dans l'ensemble des arbres obtenus (A_{max}, A_1, ..., A_n) celui qui possède la plus petite erreur de classification sur l'ensemble test;
fin

Algorithm 3: Elagage d'un arbre binaire

A.2 - Description de la classification

Dans l'arbre de décision, un nœud de décision correspond à une étape élémentaire de classification qui se caractérise complètement par :

- l'ensemble D des *primitives sélectionnées* pour le test, et associées au nœud,
- l'ensemble V des *primitives validées*, depuis l'étape initiale, la racine,
- l'ensemble C des *classes candidates* à la classification, ensemble qui se déduit de l'ensemble N des objets testés au nœud courant. Si l'indication de la classe d'un objet x est donnée par $c(x)$, on peut étendre cette indication de classe à un ensemble d'objets N, où $c(N)$ est l'ensemble des classes des objets de N

A l'étape initiale de classification, aucune primitive validée n'est associée au nœud ($V = \emptyset$) et toutes les classes sont candidates ($|C|$ est maximal). A l'étape finale, toutes les primitives nécessaires à la classification de l'objet sont associées au nœud, ainsi que la classe obtenue ($|C| = 1$).

Rappelons qu'une étape élémentaire de classification (voir § 3.3.2) dans le treillis de Galois consiste, à partir d'un concept courant, à sélectionner l'ensemble d'intervalles S issus de la structure, et à faire un choix parmi ces intervalles. S est obtenu à partir des concepts successeurs du concept courant. Plus formellement, si on note (A, B) le concept courant, et $(A_1, B_1), \ldots, (A_n, B_n)$ ses n successeurs dans le treillis de Galois, alors S est une famille d'intervalles :

$$S = \bigcup_{i=1}^{n} B_i \backslash B = \{X_1, \ldots, X_n\}$$

Les ensembles de S vérifient les propriétés suivantes :
- Ils sont disjoints : $X_i \bigcap X_j = \emptyset,\ \forall i, j \leq n, i \neq j$
- X_i $(i \leq n)$ ne peut contenir 2 intervalles issus d'une même caractéristique j $(j \leq m)$: $|X_i \bigcap I_j| \leq 1$.

On peut ici remarquer que l'arbre de décision est un cas particulier où :
- $|X_i| = 1$ chaque ensemble d'intervalles X_i de S est un singleton
- $\bigcup_{i \leq n} X_i = I_k$ l'union de ces singletons correspond exactement aux intervalles d'une caractéristique I_k.

Ce principe de navigation dans le treillis s'étend donc à l'arbre de décision.

Que ce soit pour un arbre ou pour un treillis, la famille S des intervalles sélectionnés à partir d'un concept courant s'obtient directement à partir de la structure, et ne nécessite pas de paramétrage spécifique.

L'exemple suivant va nous permettre d'illustrer la comparaison entre les deux structures (graphe et arbre) construites à partir d'un même ensemble de données.

Exemple 8 *Le jeu de données présenté par la table 4.6 (à gauche), nous a permis d'obtenir la table discrétisée 4.6 (à droite). Cette dernière nous a ensuite servi de base pour construire un arbre de décision (voir Fig. 4.5) et un treillis de Galois (voir Fig. 4.4). Sur l'arbre de décision, à partir de la racine de l'arbre, il faut choisir parmi $S = \{a_1, a_2\}$ les intervalles créés à partir de la même caractéristique a. En revanche, dans le treillis de Galois, on mélange les intervalles provenant de différents attributs. De plus, on peut être amené à sélectionner plusieurs intervalles en même temps alors qu'on teste un seul intervalle à la fois avec l'arbre. Enfin, on peut constater qu'il existe dans le treillis de Galois plusieurs chemins qui permettent d'aboutir à une même classe alors qu'il n'y a qu'une seule possibilité dans l'arbre de décision.*

A.3 - Inclusion de l'arbre de décision dans le treillis de Galois

TAB. 4.6 – Signatures des 10 objets (à gauche) et table discrétisée selon un critère d'entropie (à droite)

Classe	*Ident.*	*Signature*		
		a [0-20]	*b* [0-20]	*c* [0-20]
1	1	1	4	15
	2	0	0	18
2	3	1	12	13
	4	0	16	15
	5	3	12	11
3	6	8	16	15
	7	6	20	20
	8	15	12	15
4	9	18	4	0
	10	20	12	2

Classe	*Ident.*	*Intervalles*					
		a1 [0-3]	*a2* [6-20]	*b1* [0-4]	*b2* [12-20]	*c1* [0-2]	*c2* [11-20]
1	1	X		X			X
	2	X		X			X
2	3	X			X		X
	4	X			X		X
	5	X			X		X
3	6		X		X		X
	7		X		X		X
	8		X		X		X
4	9		X	X		X	
	10		X		X	X	

La figure 4.5 représente l'arbre de décision associé aux données de l'exemple construit selon une mesure d'entropie. Remarquons que la taille de l'arbre de décision est plus condensée que celle du treillis de Galois. En effet, la taille d'un arbre est polynomiale en la taille des données alors que celle d'un treillis est exponentielle dans le pire des cas. Une autre remarque importante porte sur le fait que tous les nœuds de l'arbre de décision sont présents dans le treillis de Galois, et ce, quel que soit le critère de construction de l'arbre. De plus, l'organisation de la structure de l'arbre se retrouve dans le treillis comme le montre la figure 4.6, où l'arbre (en gras) est inclus dans le treillis.

Théorème 2 *Soit une relation R entre un ensemble d'objets et un ensemble d'intervalles.*
Alors tout arbre de décision issu de R est inclus dans le treillis de Galois associé à R.

Preuve
Dans le treillis de Galois, à tout concept (A, B), on associe un *concept de classification* $(c(A), B)$, où $c(A)$ représente l'ensemble des classes des objets de A. Dans l'arbre de décision, à tout nœud de décision (V, D), où V est l'ensemble des intervalles validés et D l'ensemble des intervalles sélectionnés pour le test $(D = I_i)$, on associe un *nœud de classification* $(C, \varphi(V))$, où C est l'ensemble des classes issues du nœud. Il s'agit de montrer que :

Tout "nœud de classification" est un "concept de classification". C'est-à-dire qu'on peut associer à tout nœud de classification $(C, \varphi(V))$ un unique concept de classification

FIG. 4.4 – Treillis de Galois

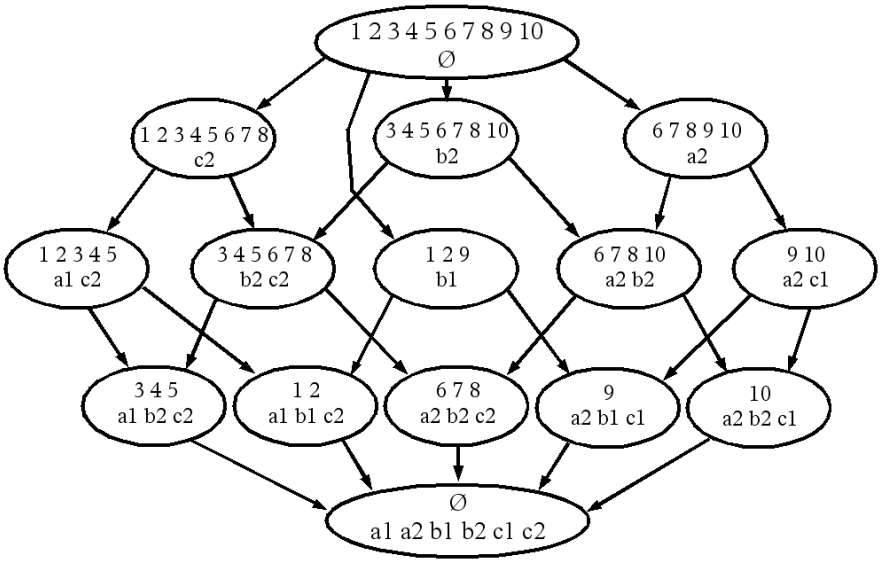
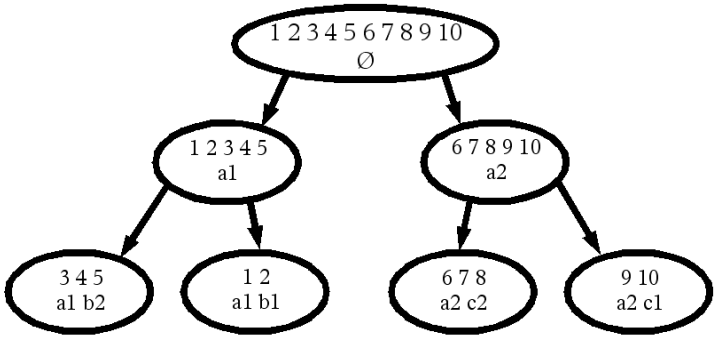


FIG. 4.5 – Arbre de décision

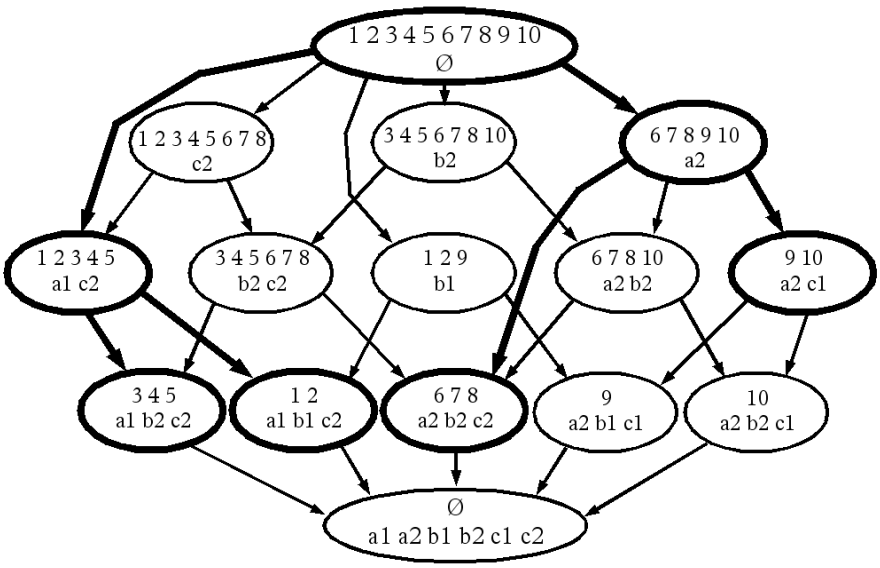


$(c(A), B)$.

Soit $(C, \varphi(V))$ un nœud de classification. Par définition d'un concept et de l'opérateur φ, il existe un concept (A, B) tel que $B = \varphi(V)$ et $A = g(\varphi(V))$. Il s'agit alors de montrer que $C = c(A)$, sachant que C est l'ensemble des classes issues du nœud :

$C = c(p \in O : \varphi(V) \subseteq I_p)$	par construction de l'arbre
$C = c(g(\varphi(V)))$	par définition de g
$C = c(A)$	car (A,B) est un concept

FIG. 4.6 – Inclusion de l'arbre de décision (en gras) dans le treillis de Galois



Les deux égalités sont vérifiées donc tout "nœud de classification" est un "concept de classification". De plus, l'arbre possède, tout comme le treillis, la propriété d'inclusion entre deux nœuds de décision. On peut donc en déduire que l'arbre de décision est inclus dans le treillis de Galois. □

A.4 - Comparaison structurelle

Dans cette étude comparative, nous avons cherché à illustrer la reconnaissance du treillis de Galois et de l'arbre de décision par des exemples. A partir de la table discrétisée donnée (voir Tab. 4.7), nous avons construit le treillis de Galois de la figure 4.10, ainsi que six arbres de décision représentés sur les figures 4.7, 4.8 et 4.9. Ces six arbres représentent toutes les combinaisons possibles de construction à partir des données de la table discrétisée.

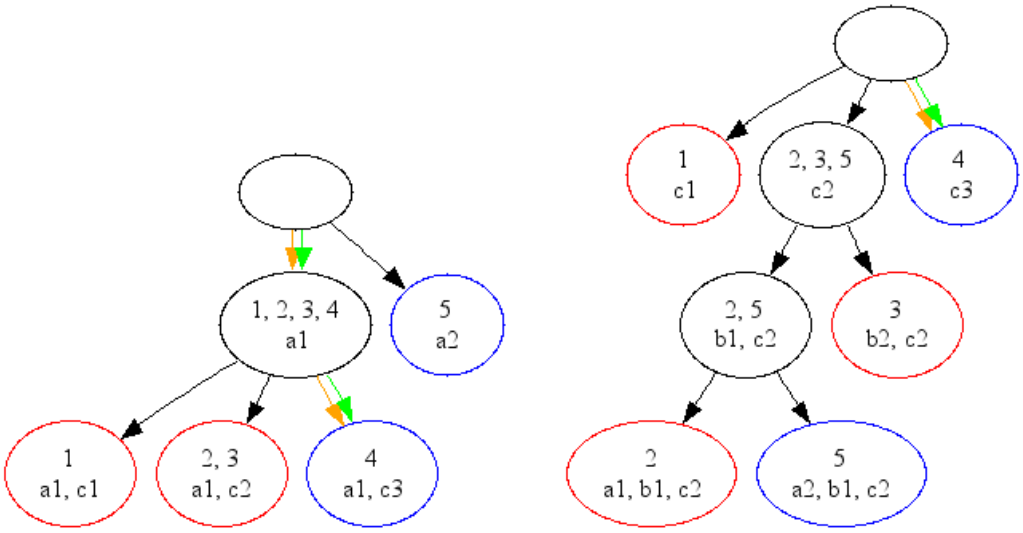
Prenons l'exemple d'un symbole bruité de la classe 2 à reconnaître, dont la signature est la suivante : $a = 1$, $b = 4$ et $c = 4$. L'attribut b, qui semble moins robuste, a subi l'influence du bruit, et la valeur de la signature se retrouve dans le mauvais intervalle b_2, qui ne caractérise jamais la classe 2 dans la table discrétisée. Sur l'ensemble des graphes, le chemin de reconnaissance est représenté par des flèches oranges.

Voyons maintenant comment se comportent les graphes lorsqu'il faut reconnaître un symbole bruité de la classe 1, dont la signature est la suivante : $a = 1$, $b = 7$ et $c = 4$. On

TAB. 4.7 – Exemple de table discrétisée

Classe	*Ident.*	*Intervalles*						
		a1 **[0-2]**	***a2*** **[3-5]**	***b1*** **[0-2]**	***b2*** **[3-5]**	***c1*** **[0-1]**	***c2*** **[2-3]**	***c3*** **[4-5]**
1	1	X		X		X		
	2	X		X			X	
	3	X			X		X	
2	4	X		X				X
	5		X	X			X	

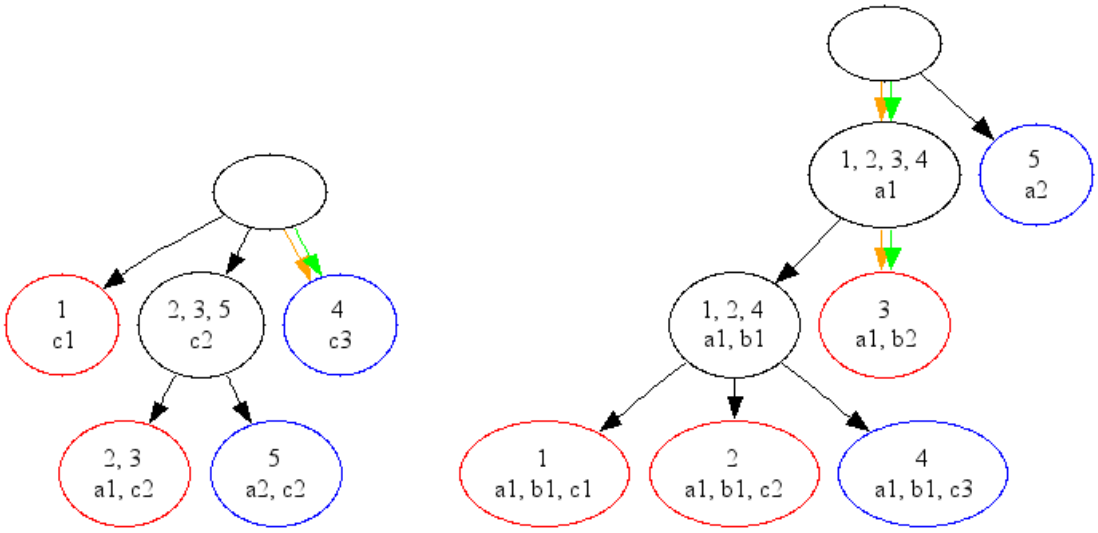
FIG. 4.7 – Reconnaissance avec les arbres de décision 1 et 2



constate qu'un bruit fort est porté sur l'attribut b qui expulse la valeur de la signature hors des intervalles b_1 et b_2, tout en restant plus proche de l'intervalle b_2. De plus, du bruit est également porté sur l'attribut c, puisque la valeur de la signature tombe dans le mauvais intervalle c_3, qui ne caractérise jamais la classe 1 dans la table discrétisée. Cette fois, les chemins de reconnaissance sont représentés par des flèches vertes.

La table 4.8 présente différents résultats pour des exemples simples et représentatifs de graphes. Selon l'arbre de décision construit, les résultats de reconnaissance peuvent varier. La classification dépend largement de l'ordre dans lequel les attributs sont proposés. Dès que l'attribut c est testé, il est déterminant pour la reconnaissance ; il conditionne le résultat pour aboutir aussi bien à une classification correcte, qu'à une

FIG. 4.8 – Reconnaissance avec les arbres de décision 3 et 4



erreur.

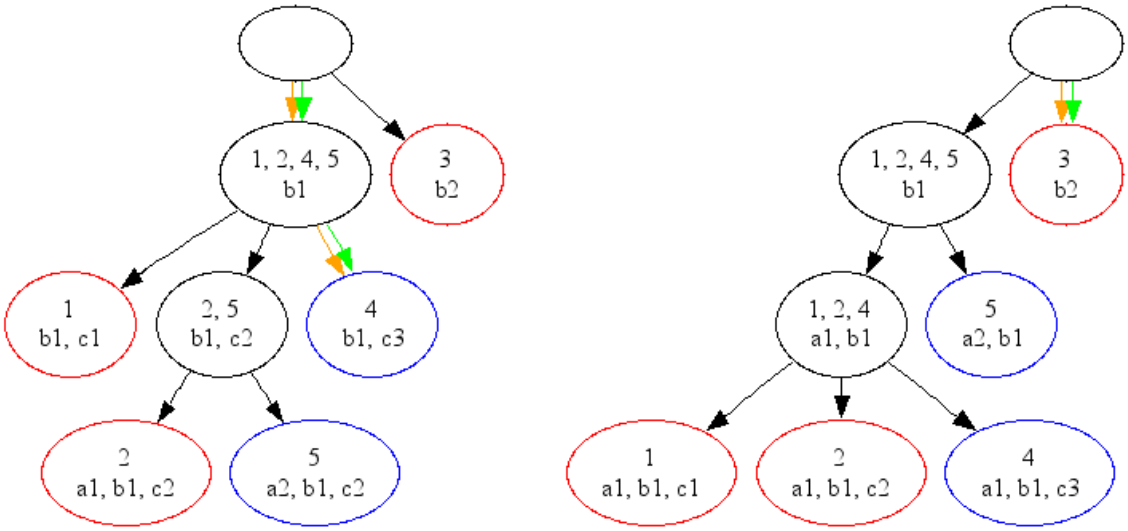
TAB. 4.8 – Reconnaissance des différents graphes

Graphe	Classification du symbole $a = 1, b = 4, c = 4$	Classification du symbole $a = 1, b = 7, c = 4$
Arbre 1	Correcte	Erreur
Arbre 2	Correcte	Erreur
Arbre 3	Correcte	Erreur
Arbre 4	Erreur	Correcte
Arbre 5	Correcte	Erreur
Arbre 6	Erreur	Correcte
Treillis	Erreur	Correcte

A.5 - Comparaison expérimentale

Dans une seconde expérimentation, nous avons comparé les taux de reconnaissance obtenus par les deux graphes sur les symboles de la base GREC 2003. La phase de discrétisation a été réalisée de manière commune pour que l'arbre de décision et le treillis de Galois reçoivent les mêmes données en entrée. Pour la construction de l'arbre de décision, nous avons utilisé la méthode CART. La figure 4.11 présente les résultats obtenus pour deux groupes de symboles : les classes 1 à 10 et les classes 11 à 20. Sur ces données, le treillis de Galois propose de meilleurs taux de reconnaissance que l'arbre de

FIG. 4.9 – Reconnaissance avec les arbres de décision 5 et 6



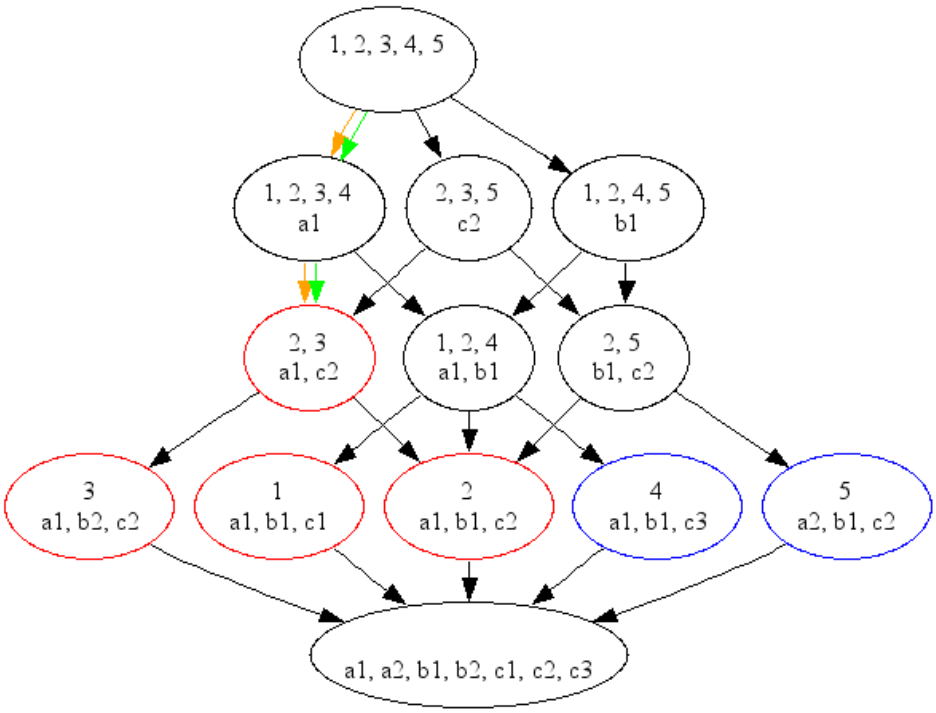
décision, et ce pour les trois signatures utilisées.

En conclusion, nous avons tout d'abord apporté la preuve que quelle que soit la structure d'arbre de décision obtenue, elle sera incluse dans celle du treillis de Galois. Avec le formalisme mis en place, nous avons donc pu associer plus facilement les liens de parenté entre ces deux structures. Toutefois, une distinction s'impose entre elles : l'arbre de décision doit être considéré comme un modèle de raisonnement, alors que le treillis de Galois est plutôt un modèle de représentation. Enfin, lors de l'expérimentation sur les symboles de la base GREC 2003, nous avons observé que le treillis obtenait une meilleure reconnaissance que l'arbre de décision.

B - Comparaison avec le k-ppv et le bayésien naïf

Nous avons également effectué une comparaison expérimentale entre NAVIGALA, le classifieur bayésien naïf et les k-plus proches voisins (k-ppv). Pour le k-ppv, nous avons utilisé la distance euclidienne comme mesure de similarité. La base de test comprend deux ensembles de 10 classes de symboles (à savoir cl1-10 et cl11-20) issus de la base GREC 2003 (voir § 2.2) et un ensemble de 25 classes (à savoir cl1-25) de la base GREC 2005 où le bruit appliqué est beaucoup plus important. Dans cette expérience, les symboles sont décrits par la signature statistique de Radon composée de 50 valeurs (voir § 1.2.1).

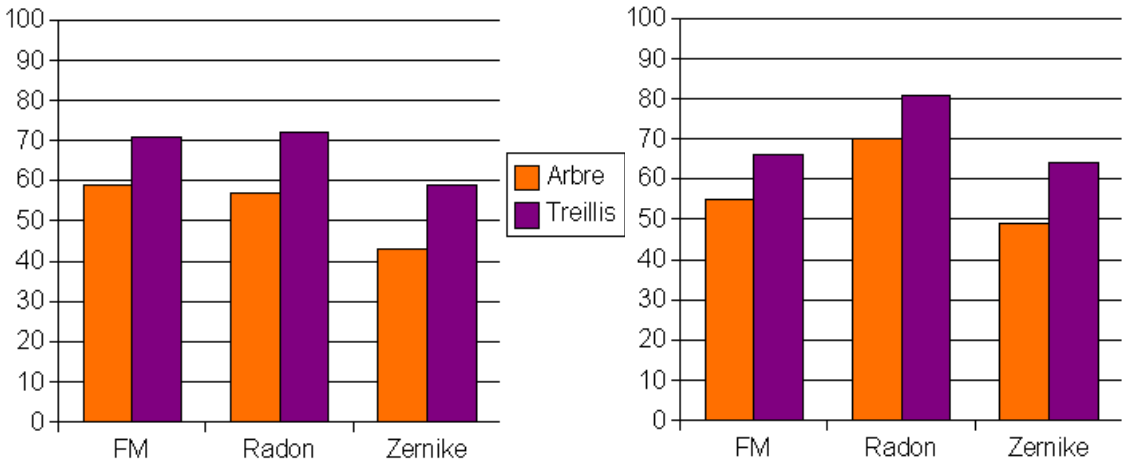
FIG. 4.10 – Reconnaissance avec le treillis de Galois



Les classifieurs sont évalués en validation croisée pour différentes tailles d'apprentissage des symboles de GREC 2003 : 5 blocs de 182 symboles (Exp1), 10 blocs de 91 symboles (Exp2), 26 blocs de 35 symboles (Exp3), et pour GREC 2005 : 5 blocs de 35 symboles (Exp4). Les taux de reconnaissance moyens sont donnés en figure 4.12. Pour Exp4, les mauvais taux sont dus au bruit très important des images GREC 2005. On constate que le k-ppv offre les meilleurs taux, et que NAVIGALA offre de meilleurs résultats que Bayes. Cependant, Bayes est meilleur que NAVIGALA pour de grands ensembles d'apprentissage (Exp1 et Exp2). Toutefois, NAVIGALA n'utilise que 6 à 15 des 50 attributs de la signature alors que les autres classifieurs en utilisent la totalité.

En conclusion, les résultats obtenus par NAVIGALA sont plutôt encourageants ; le treillis de Galois semble compétitif sur des ensembles d'apprentissage de taille réduite et n'utilise qu'un faible nombre d'attributs pour décrire les symboles.

FIG. 4.11 – Taux de reconnaissance obtenus par l'arbre de décision et le treillis de Galois sur plusieurs signatures pour les classes de symboles 1 à 10 (graphique de gauche) et 11 à 20 (graphique de droite)



4.2.3 Comparaison avec quelques classifieurs issus de l'AFC sur d'autres bases

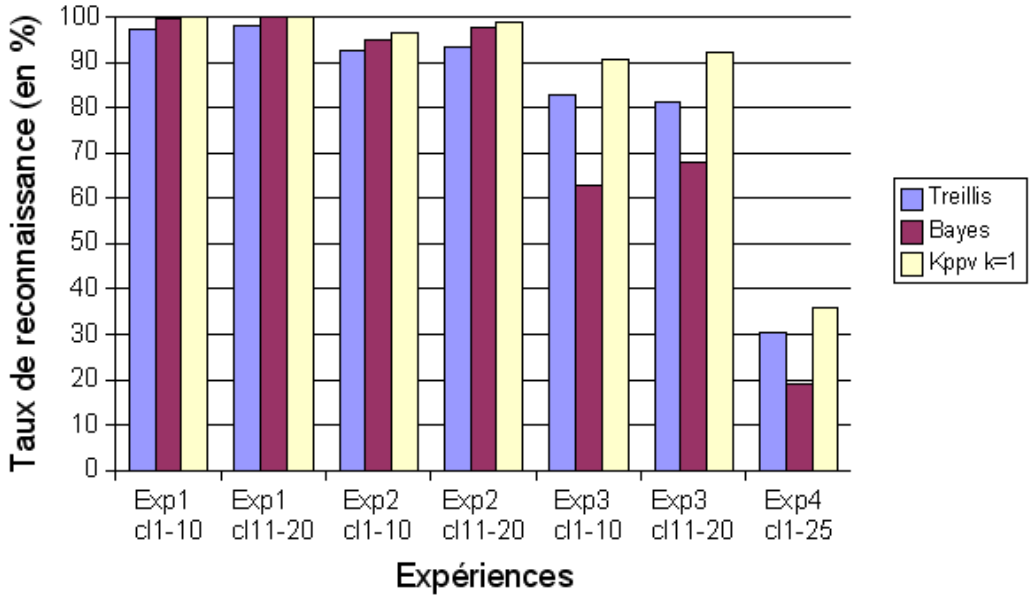
Dans le chapitre présentant l'état de l'art, nous avons décrit sept méthodes de classification basées sur le treillis de Galois (voir § 1.3.3). Nous souhaitons à présent proposer des éléments comparatifs entre ces sept méthodes et la méthode NAVIGALA, et présenter les résultats d'une étude expérimentale réalisée sur plusieurs bases de données issues de l'UCI Repository [BKM98].

A - Comparaison théorique

La figure 4.13 permet de situer toutes ces méthodes de classification, en décrivant les grandes étapes de leur fonctionnement. En bleu, les méthodes de sélection peuvent être regroupées selon qu'elles utilisent seulement les concepts, les concepts et les règles de classification, une double sélection, ou simplement des règles contextuelles. En rouge, notre approche se distingue par une utilisation de l'intégralité du graphe pour une classification des objets par navigation.

De même, les tableaux 4.9 et 4.10 proposent une synthèse détaillée des méthodes (S=Supervisé, NS=Non Supervisé), les constructions obtenues, le procédé de classification, la complexité et les résultats expérimentaux obtenus par les auteurs des méthodes.

FIG. 4.12 – Taux de reconnaissance obtenus en validation croisée pour les 3 classifieurs

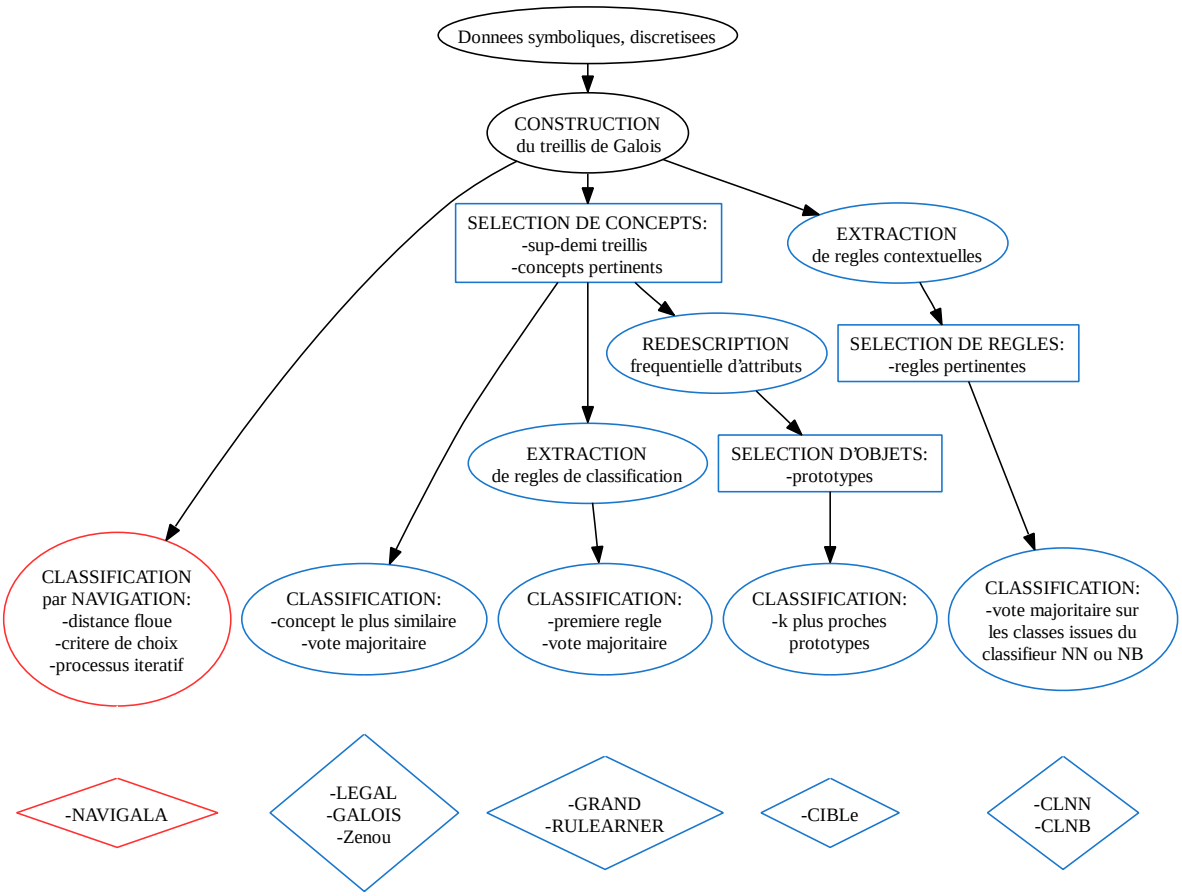


Dans ce qui suit, nous discutons le comportement des huit méthodes dans le cas où les classes sont faiblement représentées, le cas bruité et le cas où le nombre de classes est important. Dans cette discussion, nous opposons les méthodes *orientées sélection*, et la méthode NAVIGALA, *orientée navigation*.

Classes faiblement représentées Dans les méthodes orientées sélection, le fait de limiter l'apprentissage aux objets les plus représentés, implique que certaines classes d'objets faiblement représentées peuvent être exclues de l'apprentissage. Ainsi avec la méthode LEGAL-E [MN93], un sous-ensemble d'objets d'apprentissage ayant des attributs très similaires, ou des objets d'apprentissage isolés, vérifieront très peu de concepts pertinents et ne pourront donc pas être reconnus. A l'inverse, dans NAVIGALA nous conservons l'ensemble des objets sans favoriser les plus représentés, ce qui nous permet d'être exhaustif. En l'état actuel des choses, la détection de données d'apprentissage aberrantes n'est pas possible. Elles sont intégrées dans le treillis, alors qu'en sélection, les données aberrantes à faible occurrence sont bien supprimées.

Robustesse au bruit La navigation permet d'éviter dans une certaine mesure, l'influence d'un bruit porté sur plusieurs attributs. En effet, les attributs sont validés les uns à la suite des autres et non par une moyenne, comme dans les méthodes orientées sélection. De plus, l'ordre de validation des attributs est modifiable en fonction de leur robustesse au bruit. Les attributs les plus représentés sur les objets sont proposés au début du parcours du treillis et la fréquence diminue lors de la progression dans le

FIG. 4.13 – Points communs / différences des méthodes



graphe. Enfin, la mesure de distance floue permet d'assouplir les bornes des intervalles caractérisant les attributs et d'absorber les perturbations des valeurs engendrées par le bruit. Certaines méthodes de sélection font face à ce problème : "LEGAL-E résiste bien au bruit à cause de l'utilisation des critères de validité et de quasi-cohérence" [MN93], mais, "le choix des seuils de validité et de quasi-cohérence peut nécessiter un temps de travail considérable." [MN93]

Nombre important de classes Certains algorithmes de sélection ne sont pas prévus pour gérer un grand nombre de classes. "CIBLe a du mal à caractériser des données contenant un grand nombre de classes surtout lorsqu'elles sont complexes" [NMN99]. En navigation, il est possible d'effectuer la classification en plusieurs phases et en utilisant différents types de signatures. Il s'agit du mode de reconnaissance itératif.

TAB. 4.9 – Propriétés des méthodes de classification basées sur les treillis de Galois (S=Supervisé, NS=Non Supervisé)

Méthodes	Apprentissage	Construction	Classification
GRAND	S et NS	Treillis Sélection des concepts les plus généraux de chaque classe Extraction des règles de classification	Vote majoritaire parmi les règles les plus spécifiques pour l'objet
LEGAL	S	Sup-demi treillis contenant les concepts pertinents	Vote majoritaire parmi les concepts pertinents
GALOIS	S et NS	Treillis Sélection des concepts pertinents	Concept le plus similaire pour l'objet ou vote majoritaire
RULE-ARNER	S	Treillis Sélection des concepts pertinents Extraction des règles de classification	Application de la première règle vérifiée par l'objet à classer ou vote majoritaire parmi les règles vérifiées
CIBLe	S	Sup-demi treillis Sélection des concepts pertinents Redescription du contexte Sélection de prototypes	Vote majoritaire parmi les plus proches voisins
CLNN et CLNB	S	Treillis Extraction des règles contextuelles Sélection des règles pertinentes	Vote majoritaire parmi les classes retournées par le classifieur NN ou NB des règles contextuelles vérifiées par l'objet
Zenou et Samuelides	S	Treillis Concepts pertinents d'où sont extraits des amers visuels (intentions des concepts)	Classement si tous les amers visuels vérifiés appartiennent à une même pièce
NAVIGALA	S	(Treillis) optionnel Possibilité de générer les concepts à la demande	Navigation dans le treillis jusqu'à un concept représentant une classe

B - Comparaison expérimentale

Nous avons effectué des comparaisons expérimentales sur les systèmes présentés précédemment. Les bases de données utilisées sont issues de l'UCI Repository [BKM98] :

TAB. 4.10 – Propriétés des méthodes de classification basées sur les treillis de Galois (suite)

Méthodes	Complexité	Comparaison expérimentale
GRAND	$O(2^n n^4)$ avec n minimum entre le nombre d'exemples et le nombre d'attributs	Taux comparables aux autres méthodes : Assistant, AQ15, AQR, Bayes et CN2
LEGAL	$O((\|L\|n(1-\alpha))$ avec $\|L\|$ le nombre de concepts du treillis	Taux comparables à GLUE et supérieurs C4.5
GALOIS	$O(3^m 2^m n) < O(3^{2m} n)$ avec m le nombre d'attributs et n le nombre d'exemples	Taux comparables aux méthodes de la littérature
RULE-ARNER	$O(2^n n^4)$ avec n minimum entre le nombre d'exemples et le nombre d'attributs	Taux comparables à C4.5 et CN2, voire légèrement supérieurs
CIBLe	$O(min(n-1, m-1)^{h+1} m^3)$ construction du sup-demi treillis + $O(min(n-1, m-1)^{h+1})$ recherche du seuil	Taux supérieurs aux autres méthodes :IBi, K^* et Pebls
CLNN et CLNB	$O(\|L\|\|E\|^3 + \|L\|m + \|L'\|) + O(\|L'\|m).O(NN/NB)$ avec $\|E\|$ le nombre de fils d'un nœud $\|L'\| < \|L\|$ et $O(NN/NB)$ les complexités des classifieurs NN ou NB	Taux en moyenne supérieurs à ceux de NBTree, CBA et C4.5Rules
Zenou et Samuelides	$O(3^m 2^m n + \|L\|^2 m + \|L\|m)$ avec n le nombre d'exemples et m le nombre d'attributs	Taux très encourageants (18 photographies reconnues sur 20)
NAVIGALA	$(O(\|L\|n^3)$ treillis optionnel $+)$ $O(nm^2)$ classification avec n le nombre d'exemples et m le nombre d'attributs	Taux proches du k-ppv, du bayésien et supérieurs à CART

Breast-cancer (BC), Iris (IR), Monks1, Monks2 et Monks3 (M1, M2, M3), Soybean-small (SS) et Zoo (ZO) (voir Tab. 4.11). Nous avons repris les résultats expérimentaux disponibles dans les papiers décrivant les méthodes : RULEARNER [Sah95], CIBLe [NMN99], CLNB - CLNN [XHLL02], et C4.5Rules [XHLL02]. De ce fait, nous ne sommes pas en mesure de présenter pour ces méthodes, les taux de reconnaissance de l'ensemble

des bases de données. Le tableau 4.13 présente les taux d'erreur de classification obtenus en validation croisée.

TAB. 4.11 – Bases de données du ML Repository

Bases de données	Nombre d'objets	Nombre d'attributs		Nombre de classes
		Numériques	Symboliques	
BC	699	9	0	2
IR	150	4	0	3
M2	432	0	6	2
M3	432	0	6	2
SS	47	0	35	4
ZO	101	1	15	7

TAB. 4.12 – Nombre d'attributs et nombre de concepts utilisés en moyenne par NAVIGALA sur chaque base

Bases de données	Validation croisée	Nb attributs utilisés	Nb concepts du treillis
BC	10 blocs	32	6498
	5 blocs	34	6512
IR	10 blocs	13	94
	5 blocs	11	75
M2	1 bloc	16	1407
M3	1 bloc	16	1064
SS	10 blocs	6	11
	5 blocs	6	11
ZO	10 blocs	19	722
	5 blocs	18	538

En général, NAVIGALA obtient des taux d'erreurs de classification relativement proches de ceux obtenus pour les autres classifieurs sauf pour les bases MONKS2 et MONKS3. Ces deux bases ne correspondent pas au type de données que l'on souhaite traiter avec NAVIGALA. En effet, ces bases contiennent seulement 2 classes, cas très rare en classification de symboles. D'autre part, sur la base MONKS3, 5% de bruit est appliqué sur les classes (permutation des étiquettes de classe des signatures). Ce type de bruit est différent de celui pour lequel le système NAVIGALA a été développé, car habituellement le bruit s'applique sur les images et se répercute sur les attributs

de la signature et non sur les classes. Ces résultats pourraient s'expliquer par un sur-partitionnement des données comme en témoigne le nombre important de concepts dans le treillis (voir Tab. 4.12) et la présence de données aberrantes produites par ce type de bruit. NAVIGALA obtient de meilleurs résultats que CIBLe pour les bases Soybean-Small et Zoo où le nombre de classes est plus important.

TAB. 4.13 – Résultats obtenus en validation croisée sur quelques bases de données du ML Repository

Bases	Valid. croisée	Taux d'erreur de classification					
		NAVI.	RULE. [Sah95]	CIBLe [NMN99]	CLNB [XHLL02]	CLNN [XHLL02]	C4.5R [XHLL02]
BC	10 blocs	5.4			3.1	3.4	5.0
	5 blocs	5.5		4.6			
IR	10 blocs	7.4			5.3	5.3	4.7
	5 blocs	4.1					
M2	1 bloc	24	25.2	10.2			
M3	1 bloc	19	4.9	1.4			
SS	10 blocs	2.5					
	5 blocs	2.3		8			
ZO	10 blocs	4.0			3.9	3.9	7.8
	5 blocs	4.9		6.1			

En conclusion, cette étude comparative présente huit méthodes de classification basées sur l'utilisation du treillis de Galois. Les méthodes orientées sélection, dédiées à la fouille de données, nécessitent un traitement rapide, sur des données en général non bruitées. L'objectif de ces sélections est de réduire l'espace d'apprentissage selon différents critères de pertinence, tel que le nombre d'occurrence des attributs. La méthode NAVIGALA, quant à elle, est orientée navigation, et développée pour traiter efficacement des images de symboles dans un contexte bruité. Le nombre de classes traitées est potentiellement important. Nous cherchons à conserver toute l'information de l'apprentissage pour proposer de nombreux scénarii de classification et ainsi limiter l'effet du bruit sur la reconnaissance. Cette exhaustivité n'est cependant pas néfaste pour le traitement, étant donné que la complexité est largement réduite par une génération à la demande des concepts. Les données étudiées en fouille de données sont habituellement décrites par un faible nombre d'attributs. En revanche, en traitement d'images, de nombreuses signatures peuvent être extraites des images pour caractériser les objets, ainsi il en résulte un très grand nombre d'attributs. Ce type de contexte avec de nombreux

attributs est appelé contexte génomique. Il possède entre autres la particularité d'être représenté par un nombre plus petit de concepts dans le treillis par rapport au nombre de règles d'implication dans une base équivalente.

4.2.4 Conclusion

A partir de ces différents résultats proposés, nous avons d'abord pu déterminer l'influence de l'augmentation de la taille de l'ensemble d'apprentissage sur la reconnaissance. Cette augmentation améliore la reconnaissance jusqu'à un certain seuil au-delà duquel elle stagne ou régresse. Ensuite, une certaine instabilité de la structure du treillis de Galois a été observée, avec l'expérimentation concernant l'ajout des symboles modèles dans l'ensemble d'apprentissage. D'autre part, nous avons vu que l'utilisation d'une analyse discriminante en pré-traitement ne permettait pas d'améliorer la reconnaissance sur la base des symboles GREC. Nous avons également pu observer l'intérêt du mode de reconnaissance itératif par rapport au mode simple. En effet, ce mode itératif permet de supprimer les erreurs relatives aux zones d'ambiguïté.

Un résultat structurel important concerne l'inclusion de l'arbre de décision dans le treillis de Galois. Cette inclusion permet notamment au treillis d'obtenir des taux de reconnaissance comparables ou supérieurs à ceux de l'arbre de décision. De plus, il limite davantage le partitionnement des données que l'arbre, puisque sa structure regroupe les "feuilles" qui partagent les mêmes attributs.

Nous avons également constaté que le treillis de Galois obtenait des résultats de reconnaissance proches de ceux des classifieurs usuels en reconnaissance de formes. Il améliore notamment la reconnaissance dans des conditions difficiles (faible taille d'apprentissage, bruit important), alors qu'il n'utilise qu'une quantité très restreinte d'attributs de la signature.

Enfin, en comparaison avec les autres classifieurs basés sur le treillis de Galois, l'approche NAVIGALA se distingue par son principe de classification par navigation. En effet, dans le cadre de la reconnaissance d'objets bruités, il est important de conserver l'exhaustivité de description du treillis, afin de minimiser les perturbations induites par le bruit.

4.3 Discussion et perspectives

Dans cette partie, nous souhaitons présenter certaines pistes de recherche concernant l'utilisation du treillis de Galois comme classifieur. En effet, ce travail de thèse permet d'envisager plusieurs éléments qui pourraient éventuellement permettre d'améliorer la reconnaissance. Ces éléments ont été organisés selon qu'ils appartiennent à la phase d'apprentissage, de validation ou de reconnaissance.

4.3.1 Phase d'apprentissage

L'efficacité du système de reconnaissance basé sur le treillis de Galois dépend essentiellement de la robustesse des descripteurs des données et de la qualité de la discrétisation obtenue en phase d'apprentissage. En effet, malgré la prise en compte des bornes floues des intervalles, visant à apporter de la souplesse dans le traitement de reconnaissance, on ne parvient pas totalement à éviter les erreurs dues à une trop grande proximité des intervalles.

Changer la signature. Le calcul d'une signature plus robuste au bruit serait une première solution pour un perfectionnement de la reconnaissance.

Modifier la discrétisation. Une seconde possibilité serait d'améliorer la discrétisation. Dans ce but, il est possible d'imaginer l'utilisation d'un critère de coupe qui évaluerait la robustesse des attributs, et refuserait les découpes entre intervalles trop proches, de manière à limiter les risques d'erreurs de reconnaissance. Toute la difficulté consiste à évaluer le risque d'erreur. A priori, plus l'espace entre les intervalles est important, plus le risque d'erreur est faible. Cependant, pour les attributs très robustes au bruit, l'espace entre les valeurs peut sembler faible, tout en n'engendrant aucune erreur de classement. Il est difficile de mettre en place un tel critère de coupe sans utiliser de seuils, ce qui ne constitue pas une solution très satisfaisante. De plus, les seuils sont difficiles à fixer car l'espacement entre les données ne converge pas au fur et à mesure des découpes de la discrétisation. Ainsi, une découpe "risquée" à un instant t entre deux intervalles proches peut être à l'origine d'une découpe à un instant $t + 1$ très discriminante. Le critère de coupe ne doit donc pas stopper la discrétisation trop tôt afin de garantir la séparation des classes et à l'inverse, ne doit pas non plus poursuivre les découpes trop longtemps et être à l'origine d'un sur-partitionnement des données. Pour améliorer la discrétisation, il serait également envisageable d'utiliser un critère de coupe multi-dimension, qui tiendrait compte des corrélations entre les attributs

de la signature. Une meilleure connaissance de l'organisation des données pourrait ainsi permettre des découpes plus avantageuses entre les classes. Une autre piste pourrait être envisagée pour perfectionner le critère de coupe, qui consisterait à reprendre le principe de fonctionnement des machines à vecteur de support (SVM). En effet, pour limiter les erreurs de classification, il est judicieux d'effectuer les découpes entre les données les plus espacées possibles, ce qui équivaut à rechercher les marges les plus vastes dans le cadre de l'utilisation des SVM.

Ajouter un post-traitement. Enfin, il est possible d'envisager un post-traitement à la discrétisation, pour regrouper des intervalles issus d'un même attribut dans le treillis de Galois, afin de former des intervalles plus grands. Cette manière d'utiliser les intervalles est appelée "scaling". On constate parfois dans le treillis de Galois que pour certaines classes de symboles, un attribut en particulier ne participe pas à la discrimination de la classe. Cependant, lors de la reconnaissance des symboles de cette classe, le choix concernant cet attribut en particulier peut engendrer une erreur de classement. Plus précisément, avec la mise en place du scaling, les concepts du treillis de Galois contiendraient par exemple les intervalles I_1, I_2 ou encore $I_1 + I_2$.

Mettre en place un ajout incrémental. Une amélioration intéressante consisterait à proposer une phase d'apprentissage incrémentale, où l'ajout de nouveaux objets serait obtenu sans reprendre toute la discrétisation et la construction du treillis de Galois. Pour la discrétisation et pour la construction du treillis, des algorithmes incrémentaux existent et pourraient être implémentés. D'autre part, il serait judicieux de pouvoir identifier les modifications obtenues sur la structure du treillis entre deux étapes successives de discrétisation. Dans cette optique, nous avons réalisé une étude pour observer l'influence d'une découpe d'intervalle sur les concepts du treillis. Nous avons recherché la présence de modifications particulières appelées "duplications" qui correspondent à des propriétés fortes d'évolution de la structure de graphe. Cependant, d'après ce que nous avons observé, il ne semble pas que l'évolution du treillis de Galois en fonction de la discrétisation respecte des propriétés particulières. De plus, lors de l'ajout de nouveaux objets dans l'ensemble d'apprentissage, la structure du treillis de Galois semble évoluer de manière assez instable.

4.3.2 Phase de validation

Nous utilisons un ensemble de symboles pour l'apprentissage et un second ensemble de symboles pour la reconnaissance. Afin d'optimiser le classifieur, il est envisageable de considérer un troisième ensemble de symboles dédié à la phase de validation. Cette

phase de validation est effectuée juste après l'apprentissage et permet d'intervenir sur le classifieur et d'optimiser son comportement pour la phase de reconnaissance. Avec cette phase de validation, trois possibilités d'intervention sur le treillis de Galois construit à l'apprentissage s'offrent à nous :

Modification locale du contenu du treillis. La structure du treillis serait conservée.

- Déplacer le repère considéré pour les intervalles (centre de gravité, médiane, ou milieu) pour décaler les bornes floues.
- Modifier les bornes des intervalles, avec le risque de modifier la structure du graphe et ne plus vérifier la propriété de treillis. En effet, si on rétrécit les bornes des intervalles, le risque de bouleverser la structure du graphe est assez important car certaines valeurs n'appartiendront plus à aucun intervalle. A l'inverse, si on élargit les bornes des intervalles, on conserve la propriété de treillis tant que les intervalles issus d'un même attribut ne se chevauchent pas. Dès qu'ils se chevauchent, certaines valeurs peuvent appartenir à plusieurs intervalles issus d'un même attribut. Dans l'expérimentation 4.2.1, nous avons présenté les effets d'une légère modification de l'ensemble d'apprentissage. L'ajout de quelques symboles modèles entraînait des changements sur les bornes des intervalles qui se répercutaient sur la discrétisation et ainsi bouleversaient la forme du treillis de Galois.
- Ajouter une pondération aux arcs du treillis, pour favoriser les chemins les plus empruntés. Cette manière de procéder peut créer des inégalités entre les classes, car les classes très représentées seraient favorisées. On retrouve ce problème de sur-apprentissage dans les réseaux de neurones.
- Mettre en place le scaling à l'aide de l'ensemble de validation. En effet, on pourrait détecter les intervalles à l'origine des erreurs et rechercher s'ils ne sont pas essentiels dans la discrimination des classes concernées. Dans ce cas, il serait possible de transformer les intervalles I_1 ou I_2 en un intervalle $I_1 + I_2$ plus grand qui limiterait les erreurs de classification.

Reprise globale de la discrétisation. Il faudrait alors reconstruire un nouveau treillis de Galois.

- Les techniques de bagging et de boosting peuvent être envisagées. Il s'agit de créer plusieurs classifieurs à partir d'un même ensemble de données partagé en plusieurs sous-ensembles et de les faire coopérer. Chaque sous-ensemble est à l'origine d'un classifieur lui correspondant. Le résultat final est obtenu par un vote. Dans ce cadre, l'ajout de nouveaux objets pourrait faire évoluer la structure du treillis de Galois.
- La robustesse d'un attribut peut être évaluée par une phase de validation. Les attributs non robustes pourraient ainsi être supprimés, et la discrétisation serait effectuée à nouveau sans ces attributs.

Utilisation d'un modèle hiérarchique et itératif du treillis pour un sous-ensemble d'objets. Le principe serait de détecter à la phase de validation les concepts sur lesquels les erreurs de classification se produisent afin de stopper la reconnaissance en l'état. Dans les concepts concernés par ces erreurs, il serait possible de récupérer les objets afin de les caractériser par une nouvelle signature qui donnerait lieu à une nouvelle discrétisation et un nouveau treillis. La reconnaissance pourrait alors être effectuée par le parcours habituel du treillis et lors du passage sur un concept qui pose problème, le parcours basculerait sur un autre treillis et se poursuivrait jusqu'à atteindre une classe.

4.3.3 Phase de reconnaissance

Comme nous l'avons précisé, l'efficacité de cette étape dépend largement de la structure de treillis obtenu à l'apprentissage. Toutefois, même si la marge de progression est restreinte, il est concevable d'améliorer la reconnaissance en apportant des modifications à la navigation.

Navigation par sauts. Au lieu de se déplacer pas à pas dans le treillis, il est possible de faire des déplacements plus importants en effectuant des sauts. Tout le problème réside dans le choix de considérer les attributs indépendamment les uns des autres ou bien au contraire de manière globale. Lorsque l'on choisit la première option, un seul attribut peut faire basculer la reconnaissance vers une mauvaise classe. En revanche, en considérant les attributs de manière globale, c'est l'influence des attributs en moyenne qui peut avoir tendance à fausser la reconnaissance. Plusieurs navigations par sauts peuvent être envisagées :

- La navigation peut débuter non pas par le concept min[1], mais par le concept x qui contient tous les intervalles validés[2] par l'objet à reconnaître. Puis, la navigation s'effectue pas à pas.
- Lorsque plusieurs intervalles issus de concepts successeurs différents sont validés en même temps, un saut vers le concept x qui contient tous ces intervalles validés peut être effectué. Le reste du temps, la navigation s'effectue pas à pas.

Dans ces deux cas, il est nécessaire de modifier le critère de choix en intégrant l'utilisation de l'opérateur de fermeture pour retrouver dans l'ensemble des concepts du treillis celui qui contient tous les intervalles validés. Il faut remarquer que certains intervalles non validés peuvent ainsi être englobés par ce choix du concept. Le danger de ce type de navigation par sauts est de se retrouver directement dans

[1]Concept min : premier concept de la navigation qui contient tous les objets
[2]Les valeurs tombent exactement dans les bornes des intervalles.

le concept max[3], ce qui pourrait se produire si une combinaison d'intervalles est validée alors qu'elle n'apparaît jamais dans la table discrétisée.

Navigation avec backtracking. Au lieu de toujours progresser du haut (concept min) vers le bas du treillis (concepts finaux), il pourrait être envisagé lorsque la prise de décision est trop risquée, de remonter dans le treillis de Galois afin de choisir un autre chemin de classification.

Aide à la navigation par intégration d'informations globale. Il est possible de tempérer / moduler les décisions prises au niveau local lors de la navigation pas à pas dans le treillis de Galois, par une / des confirmation(s) globale(s).

Classification avec rejet. Pour améliorer la classification, un système de rejet pourrait être ajouté à la méthode Navigala. Ainsi, une classe d'objet qui n'a pas été apprise lors de l'apprentissage, ne se verrait pas obligatoirement attribuer un label de classe lors de la navigation dans le graphe, car elle pourrait être détectée et identifiée comme classe inconnue.

Pour comparer la navigation pas à pas (locale) et la navigation par saut du concept min vers les concepts finaux correspondant aux classes (globale), nous avons cherché à illustrer leur fonctionnement par des exemples. A partir de la table discrétisée (voir Tab. 4.14), nous avons construit le treillis de Galois de la figure 4.14.

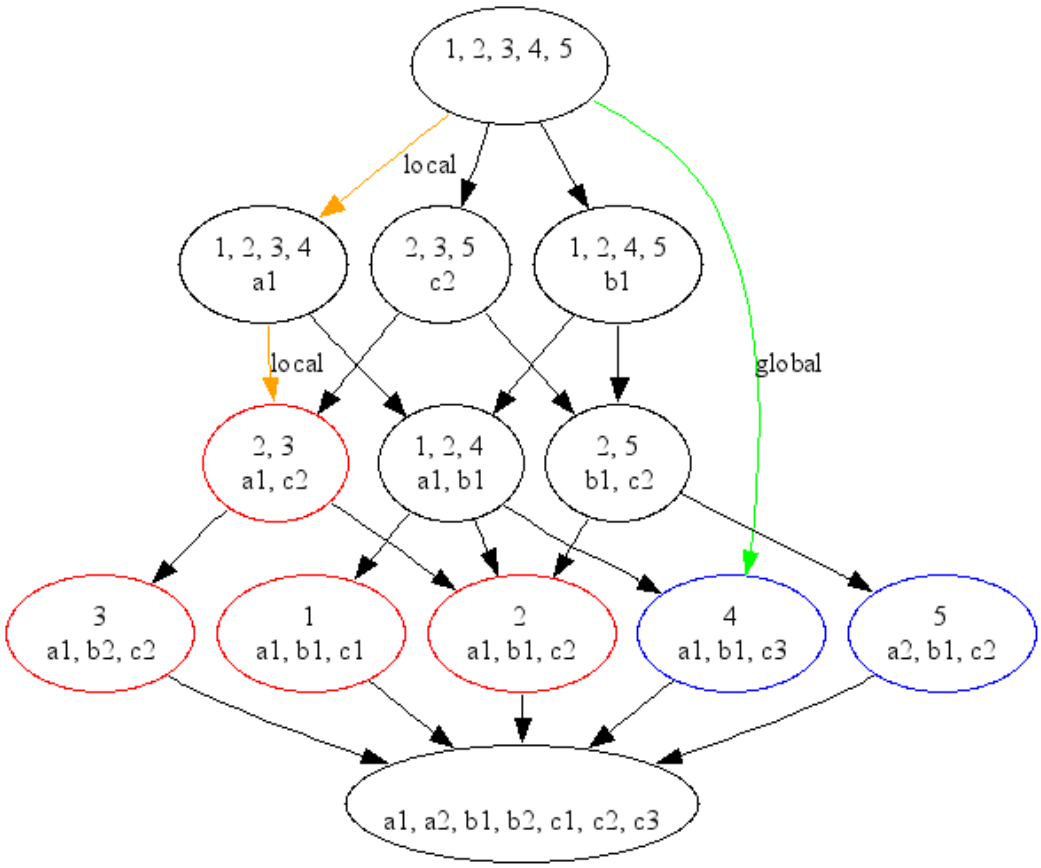
TAB. 4.14 – Exemple de table discrétisée

Classe	*Ident.*	*Intervalles*						
		a1 [0-2]	*a2* [3-5]	*b1* [0-2]	*b2* [3-5]	*c1* [0-1]	*c2* [2-3]	*c3* [4-5]
1	1	X		X		X		
	2	X		X			X	
	3	X			X		X	
2	4	X		X				X
	5		X	X			X	

Imaginons que nous ayons à reconnaître un symbole bruité de la classe 2 dont la signature est la suivante : $a = 1$, $b = 4$ et $c = 4$. L'attribut b, qui semble moins robuste, a subi l'influence du bruit, et la valeur de la signature se retrouve dans le mauvais intervalle b_2, qui ne caractérise jamais la classe 2 dans la table discrétisée. Dans ce cas, la navigation globale permet de bien reconnaître la classe du symbole malgré le bruit.

[3]Concept max : concept du treillis qui contient tous les intervalles et aucun objet correspondant. Ce concept ne caractérise aucune classe.

FIG. 4.14 – Reconnaissance locale vs globale dans le treillis de Galois



En revanche, en navigation locale, la validation successive des attributs a_1, puis c_2 mène à une erreur de classification.

Prenons maintenant un nouvel exemple, avec un symbole bruité de la classe 1 à reconnaître, dont la signature est la suivante : $a = 1$, $b = 7$ et $c = 4$. On constate qu'un bruit fort est porté sur l'attribut b qui expulse la valeur de la signature hors des intervalles b_1 et b_2, tout en restant plus proche de l'intervalle b_2. De plus, du bruit est également porté sur l'attribut c, puisque la valeur de la signature tombe dans le mauvais intervalle c_3, qui ne caractérise jamais la classe 1 dans la table discrétisée. Pour ce nouveau symbole à reconnaître, les parcours dans le treillis de Galois des navigations globale et locale sont identiques à ceux obtenus pour le symbole précédent, mais cette fois, c'est la navigation locale qui permet de bien reconnaître le symbole.

D'après ces exemples, il semble que la navigation locale permet d'absorber un bruit plus fort que la navigation globale. Cependant, ces résultats sont à nuancer car ils dépendent de l'ordre dans lequel sont proposés les intervalles et également des attributs

sur lesquels le bruit se porte.

Dans le treillis de Galois, lors de la reconnaissance, les intervalles qui sont proposés en premier dans la navigation, sont les intervalles partagés par le plus grand nombre d'objets. Cette organisation peut être contraignante si les intervalles les plus fréquents sont peu robustes au bruit. Quelques expérimentations sur la base de symboles GREC 2003 ont été réalisées pour déterminer à quel niveau de profondeur dans le treillis les erreurs pouvaient avoir lieu. Le but était d'évaluer l'intérêt de mettre en place le processus de reconnaissance itérative. Il s'est avéré que sur cette base, les erreurs étaient situées très tard dans la navigation, le plus souvent à l'avant-dernier concept. Les erreurs correspondaient donc à des choix à effectuer entre deux ou trois classes représentées par très peu d'objets de l'ensemble d'apprentissage. Suite à ce résultat, la reconnaissance itérative a été mise en place.

La discrétisation que nous avons mise en place conduit à la construction d'un treillis de Galois particulier. En effet, puisque l'on force chaque objet à avoir une valeur qui tombe dans un intervalle pour chaque attribut de la signature, le treillis obtenu est co-atomistique. Les co-atomes sont les concepts du treillis qui précèdent le concept max ; il s'agit des concepts représentant les classes. Cette propriété se caractérise par le fait que chaque objet est décrit par le même nombre d'intervalles. Cela crée un nombre important de combinaisons entre les attributs et le treillis de Galois offre de ce fait autant de possibilités de scénarii de classification. Cependant, il peut arriver parfois qu'une classe soit très simple à séparer des autres classes à l'aide de quelques attributs, et dans ce cas, le fait de vouloir absolument décrire les objets de cette classe par tous les autres attributs, peut alourdir le processus de reconnaissance. Sans co-atomicité, le concept représentant cette classe serait alors situé beaucoup plus haut dans le treillis. D'autre part, cette contrainte ne permet pas la gestion des données manquantes. En résumé, la co-atomicité n'est pas une propriété obligatoire pour obtenir un treillis de Galois. Elle est intéressante en terme de reconnaissance lorsque la description des données est complète. Cependant, il est possible de s'en passer lorsque l'objectif est de manipuler des données manquantes.

Enfin, il serait intéressant d'envisager une comparaison théorique et expérimentale entre des classifieurs basés sur des approches structurelles et la méthode Navigala combinée avec l'utilisation de notre signature structurelle.

4.4 Conclusion

Dans ce dernier chapitre, plusieurs études comparatives ont été énumérées, confrontant l'approche à base de treillis aux classifieurs usuels de la reconnaissance de formes, puis aux classifieurs issus de l'analyse formelle des concepts. Dans ce cadre, NAVIGALA a fourni des résultats relativement compétitifs.

Dans un second temps, différentes pistes de recherche ont été proposées pour améliorer la reconnaissance obtenue par NAVIGALA. Il semble notamment, que l'ajout d'une étape de validation au processus d'apprentissage permette la suppression de certaines erreurs de classification, offrant ainsi des perspectives d'amélioration encourageantes.

Conclusion générale

Les travaux de recherche menés lors de cette thèse, concernent un sujet à la frontière entre deux domaines, la reconnaissance de symboles d'une part, et l'analyse formelle des concepts (AFC) d'autre part.

Dans l'état de l'art, nous avons donné différentes références concernant ces deux domaines. Parmi les signatures utilisées dans la littérature en reconnaissance de formes, nous avons plus particulièrement détaillé les approches structurelles et statistiques. Nous avons également décrit plusieurs classifieurs communément employés en reconnaissance de formes. Enfin, nous avons achevé cet état de l'art par des fondements mathématiques relatifs à l'AFC et au treillis de Galois, ainsi que quelques contributions concernant la classification à base de treillis et de règles d'association.

Dans un second temps, nous avons présenté les descripteurs de formes utilisés pour nos expérimentations et leurs paramétrages. Il s'agit des signatures de Fourier-Mellin, Radon, Zernike et d'une signature structurelle. Cette dernière est fondée sur une extraction de segments par une transformée de Hough et une description des symboles constituée de chemins dans un graphe topologique.

Ensuite, dans le troisième chapitre, la méthode de classification à base de treillis de Galois est décrite de manière détaillée. Elle suit les étapes classiques de la reconnaissance (apprentissage et classification) et dépend de trois paramètres : un critère de coupe, une mesure de distance et un critère de choix. Une étude expérimentale relative à ces paramètres a également été mise en place. Ce chapitre détaille d'autre part, un mode de reconnaissance itérative. Il s'agit de traiter les données à classer de manière hiérarchique. On affine petit à petit la reconnaissance des symboles en isolant un certain nombre de classes parmi toutes les classes candidates. A chaque étape, il est possible d'utiliser une nouvelle signature pour décrire les symboles. Cette reconnaissance itérative nécessite la mise en place d'un quatrième paramètre afin d'arrêter la progression dans le graphe lorsque la prise de décision est jugée trop risquée.

Le dernier chapitre, quant à lui, recense plusieurs études comparatives entre la méthode NAVIGALA et des classifieurs usuels de reconnaissance de formes d'une part, et entre cette même méthode et des classifieurs issus de l'AFC d'autre part. Ce chapitre offre également quelques pistes de recherche en perspective de ce travail.

Incontestablement, l'essentiel de la contribution à cette thèse concerne le développement d'une nouvelle méthode de classification basée sur l'utilisation d'un treillis de Galois. L'originalité de cette méthode tient tout d'abord au fait que ce graphe n'a jamais été employé comme classifieur dans le domaine de la reconnaissance de symboles. De plus, cette approche est basée sur une navigation dans le graphe, alors que les autres classifieurs à base de treillis sont plutôt utilisés comme outil de sélection des éléments pertinents pour la classification.

Ce travail de thèse nous a également permis de réaliser un système de reconnaissance de symboles NAVIGALA. Cet outil, implémenté en langage Java, permet une visualisation et une évaluation aisée de l'ensemble du processus de classification. Plus particulièrement, il est possible d'observer les symboles, les signatures générées, la table discrétisée, le treillis de Galois et les résultats expérimentaux. NAVIGALA contient également les deux modes de reconnaissance simple et itérative.

Un autre développement important porte sur la signature structurelle à base de transformée de Hough. Comme en atteste l'état de l'art de ce manuscrit, il existe de nombreuses références concernant ce type de signatures. La singularité de cette approche, provient de l'intégration de composantes structurelles de haut niveau dans la représentation des symboles, et ce, de manière générique. Par ailleurs, elle autorise un ajustement de la complexité de la description.

Nous sommes conscients que cette méthode possède certaines limitations. Premièrement, elle n'est pas totalement automatique puisqu'elle dépend de trois paramètres. Même si un paramétrage par défaut peut sembler convenir, nous ne pouvons garantir les résultats sur l'ensemble des bases de données existantes. Une autre difficulté à surmonter concerne la complexité du treillis de Galois. Même si dans la pratique la taille du graphe reste la plupart du temps raisonnable, elle peut, dans le pire des cas, être exponentielle. La génération à la demande du treillis est toutefois un moyen ingénieux pour limiter les effets de cette exponentialité. Enfin, le système de reconnaissance est largement dépendant de la qualité des signatures décrivant les symboles et de celle de la discrétisation.

Cependant, nous avons montré que le treillis de Galois bénéficiait également de nombreuses propriétés avantageuses pour un classifieur. Comme tout graphe, il offre

une bonne lisibilité et les résultats expérimentaux sont facilement interprétables. A l'instar de l'arbre de décision, il nécessite une phase de discrétisation comportant un certain nombre de paramètres pour lui permettre de traiter des données numériques. Mais, à l'inverse de l'arbre de décision, sa construction est unique et ne dépend d'aucun paramètre. De plus, le treillis intègre naturellement dans sa structure le regroupement des symboles qui partagent les mêmes attributs. De ce fait, on obtient un moindre partitionnement des données que dans l'arbre de décision. Par ailleurs, le treillis de Galois autorise une utilisation des données continues et discrètes, ce qui est rarement le cas pour les autres classifieurs. Les nombreux scénarii de classification qu'il propose lui confèrent une certaine robustesse au bruit puisque l'ordre de traitement des attributs de la signature n'est pas totalement imposé. Pour finir, la mise en place d'une reconnaissance hiérarchique et itérative permet le traitement conjoint de descripteurs structurels et statistiques.

Enfin, comme perspectives de travail, il peut être envisagé de rechercher des signatures plus performantes en terme de robustesse. Une autre possibilité d'amélioration de la classification pourrait provenir d'une modification de la phase de discrétisation. Pour cela, il faudrait réfléchir au moyen d'intégrer une évaluation de la résistance au bruit dans le calcul du critère de coupe, ou bien permettre à ce critère de prendre en compte plusieurs dimensions à la fois, ou encore de trouver un critère qui s'inspire du principe des machines à vecteur de support. L'ajout d'un ensemble de validation serait un autre moyen pour perfectionner plus aisément l'apprentissage et notamment la discrétisation. Pour terminer, grâce au développement de la signature structurelle, une piste de recherche captivante s'ouvre à nous, qui consiste en l'adaptation du système pour une reconnaissance des symboles en contexte.

Bibliographie

Publications liées à cette thèse

[BGO07a] K. Bertet, S. Guillas, and J.M. Ogier. Extensions of bordat's algorithm for attributes. In *Fifth International Conference on Concept Lattices and their Applications (CLA'2007)*, pages 38–49, Montpellier, France, October 24-26 2007.

[BGO07b] K. Bertet, S. Guillas, and J.M. Ogier. On the joint use of two implicational bases. In *Fifth International Conference on Formal Concept Analysis (ICFCA'07)*, pages 23–39, Clermont-Ferrand, France, February 12-16 2007.

[CGV+07] M. Coustaty, S. Guillas, M. Visani, K. Bertet, and J.M. Ogier. Flexible structural signature for symbol recognition using a concept lattice classifier. In *Seventh IAPR International Workshop on Graphics Recognition (GREC'07)*, Curitiba, Brazil, September 20-21 2007.

[GBO05a] S. Guillas, K. Bertet, and J.M. Ogier. Concept Lattices : a Tool for Supervised Classification ? In *Sixth IAPR Workshop on Graphics Recognition (GREC'05)*, pages 56–67, Hong Kong, China, August 2005.

[GBO05b] S. Guillas, K. Bertet, and J.M. Ogier. Les treillis de Galois : un outil pour la sélection de primitives ? *Traitement du Signal (numéro spécial)*, 22(3) :273–291, 2005.

[GBO06a] S. Guillas, K. Bertet, and J.M. Ogier. Comment utiliser le treillis de Galois en reconnaissance d'images ? In Nicole Vincent and Nicolas Loménie, editors, *Atelier Extraction de COnnaissance à partir d'Images (ECOI) - EGC 2006*, pages 31–36, Lille, France, 2006.

[GBO06b] S. Guillas, K. Bertet, and J.M. Ogier. Concept lattice classifier : a first step towards an iterative process of recognition of noised graphic objects. In S. Ben-Yahia and E. Mephu-Nguifo, editors, *Fourth International Conference on Concept Lattices and their Applications (CLA'2006)*,

pages 257–263, Yasmine Hammamet, Tunisia, October 30 - November 1 2006.

[GBO06c] S. Guillas, K. Bertet, and J.M. Ogier. A generic description of the concept lattices' classifier : Application to symbol recognition. *Graphics Recognition : Ten Years Review and Future Perspectives (Revised Selected Papers)*, 3926 :47–60, 2006. LNCS, Springer Berlin / Heidelberg. Revised and extended version of paper first presented at Sixth IAPR International Workshop on Graphics Recognition (GREC'05).

[GBO06d] S. Guillas, K. Bertet, and J.M. Ogier. Reconnaissance de symboles bruités à l'aide d'un treillis de Galois. In L. Likforman-Sulem, editor, *Neuvième Colloque International Francophone sur l'Ecrit et le Document (CIFED'06), dans le cadre de la SDN'06*, pages 85–90, Fribourg, Suisse, 2006. isbn 2-9522067-1-6.

Références bibliographiques du mémoire

[Ada01] S. Adam. *Interprétation de Documents Techniques : des Outils à leur Intégration dans un Système à Base de Connaissances.* PhD thesis, Université de Rouen, 11 décembre 2001.

[Ait08] M.J. Aitkenhead. A co-evolving decision tree classification method. *Expert Systems with Applications*, 34 :18–25, 2008.

[AKA91] D. Aha, D. Kibler, and M. Albert. Instance-based learning algorithms. *Machine Learning*, 6 :37–66, 1991.

[AOC+00] S. Adam, J.M. Ogier, C. Cariou, R. Mullot, J. Gardes, and Y. Lecourtier. Fourier-Mellin based invariants for the recognition of multi-oriented and multi-scaled shapes : Application to engineering drawings analysis, in invariants for pattern recognition and classification. *Series in Machine Perception and Artificial Intelligence*, 42 :132–147, 2000.

[AOC+01] S. Adam, J.M. Ogier, C. Cariou, R. Mullot, J. Gardes, and Y. Lecourtier. Utilisation de la transformée de Fourier-Mellin pour la reconnaissance de forme multi-orienté et multi-échelle : application à l'analyse automatique de documents techniques. *Traitement du Signal*, 18(1) :17–33, 2001.

[AS94] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In J.B. Bocca, M. Jarke, and C. Zaniolo, editors, *Proc. 20th International Conference on Very Large Data Bases (VLBD)*, pages 487–499. Morgan Kaufmann, 1994.

[ATP05] E. Avci, I. Turkoglu, and M. Poyraz. Intelligent target recognition based on wavelet packet neural network. *Expert Systems with Applications*, 29 :175–182, 2005.

[AZ02a] M. Antonie and O. Zaiane. Classifying text documents by associating terms with text categories. In *Proceedings of the Thirteenth Austral-Asian Database Conference (ADC'02)*, Melbourne, Australia, 2002.

[AZ02b] M. Antonie and O. Zaiane. Text document categorization by term association. In *Proceedings of IEEE International Conference on Data Mining (ICDM'02)*, pages 19–26, Maebashi City, Japan, 2002. IEEE Computer Society.

[AZD07] K. Amolins, Y. Zhang, and P. Dare. Wavelet based image fusion techniques - an introduction, review and comparison. *ISPRS Journal of Photogrammetry and Remote Sensing*, 62(4) :249–263, 2007.

[BEBY06] I. Bouzouita, S. Elloumi, and S. Ben-Yahia. GARCm : Generic Association Rules based Classifier Multi-parameterizable. In S. Ben-Yahia and E. Mephu-Nguifo, editors, *Proceedings of the Fourth International Conference on Concept Lattices and their Applications (CLA'06)*, pages 115–125, Yasmine Hammamet, Tunisia, 2006.

[Bel61] R. Bellman. *Adaptive control processes : a guided tour*. Princeton Univ. Press, 1961.

[Ber06] S. Bernard. Extraction de primitives structurelles pour la reconnaissance de symboles : Une approche robuste par Transformée de Hough. Master's thesis, Université de La Rochelle, La Rochelle, France, 2006.

[BFOS84] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and regression trees*. Wadsworth Inc., Belmont, California, 1984.

[BGV92] B.E. Boser, I.M. Guyon, and V.N. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *5th Annual ACM Workshop on COLT*, pages 144–152, Pittsburgh, PA, 1992. ACM Press.

[BHAT05] E. Barbu, P. Héroux, S. Adam, and E. Trupin. Fouille de graphes et découverte de règles d'association : application à l'analyse d'images de document. *Revue Nouvelles Technologies de l'Information*, pages 463–468, 2005.

[BHO07] S. Belkasim, E. Hassan, and T. Obeidi. Explicit invariance of Cartesian Zernike moments. *Pattern Recognition Letters*, 2007.

[Bir67] G. Birkhoff. *Lattice theory*, volume 25. American Mathematical Society, 3rd edition, 1967.

[BK99] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms : Bagging, boosting, and variants. *Machine Learning*, 36((1-2)) :105–139, 1999.

[BKM98] C. Blake, E. Keogh, and C. Merz. Uci repository of machine learning databases, 1998.

[BM70] M. Barbut and B. Monjardet. *Ordres et classifications : Algèbre et combinatoire*. Hachette, Paris, 1970. 2 tomes.

[BN04] K. Bertet and M. Nebut. Efficient algorithms on the family associated to an implicationnal system. *DMTCS*, 6(2) :315–338, 2004.

[Bor86] J.P. Bordat. Calcul pratique du treillis de Galois d'une correspondance. *Math. Sci. Hum.*, 96 :31–47, 1986.

[BS07] A. Broumandnia and J. Shanbehzadeh. Fast Zernike wavelet moments for Farsi character recognition. *Image and Vision Computing*, 25 :717–726, 2007.

[BSA91] S.O. Belkasim, M. Shridar, and M. Ahmadi. Pattern recognition with moment invariants : a comparative study and new results. *Pattern Recognition*, 24 :1117–1138, 1991.

[BV07] V. Bruni and D. Vitulano. Combined image compression and denoising using wavelets. *Signal Processing : Image Communication*, 22 :86–101, 2007.

[CBK03] G.Y. Chen, T.D. Bui, and A. Krzyzak. Contour-based handwritten numeral recognition using multiwavelets and neural networks. *Pattern Recognition*, 36 :1597–1604, 2003.

[CDB86] R.L. Cannon, J.V. Dave, and J.C. Bezdek. Efficient implementation of the fuzzy c-means clustering algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence archive*, 8(2) :248–255, March 1986.

[CF04] D.R. Carvalho and A.A. Freitas. A hybrid decision tree/genetic algorithm method for data mining. *Information Sciences*, 163(1-3) :13–35, June 2004.

[CH67] T. Cover and P. Hart. Nearest Neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1) :21–27, 1967.

[Che69] M. Chein. Algorithme de recherche des sous-matrices premières d'une matrice. *Bulletin Mathématique de la Sociologie Scientifique de la R.S. de Roumanie*, 1969.

[CKB87] B. Cestnik, I. Kononenko, and I. Bratko. ASSISTANT 86 : a knowledge-elicitation tool for sophisticated users. In *Progress in Machine Learning*, pages 31–45. Sigma Press, 1987.

[CMKM02] A. Cornuejols, L. Miclet, Y. Kodratoff, and T. Mitchell. *Apprentissage artificiel : Concepts et algorithmes*. Eyrolles, 2002. Collection : Technique scientifique télécom.

[CN89] P. Clark and T. Niblett. The cn2 induction algorithm. *Machine Learning*, 3 :261–283, 1989.

[Cou07] M. Coustaty. Mise en place d'une signature structurelle dédiée aux symboles. Master's thesis, Université de La Rochelle, La Rochelle, France, 2007.

[CP98] A. Chhabra and I. Phillips. The second international graphics recognition contest - raster to vector conversion : A report. In *Graphics recognition, algorithms and systems - Selected papers from GREC'97*, volume 1389, pages 390–410. Lecture Notes in Computer Science (LNCS), 1998.

[CR93] C. Carpineto and G. Romano. Galois : An order-theoretic approach to conceptual clustering. In *Proceedings of ICML'93*, pages 33–40, Amherst, July 1993.

[CS93] S. Cost and S. Salzberg. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, 10 :57–78, 1993.

[CT95] J. Cleary and L. Trigg. K* : an instance-based learner using an entropic distance measure. In *Proceedings of Machine Learning Conference*, pages 108–114, Tahoe City, CA, USA, 1995.

[CV00] L.P. Cordella and M. Vento. Symbol recognition in documents : a collection of techniques ? *International Journal on Document Analysis and Recognition (IJDAR)*, 3(2) :73–88, 2000.

[Das91] B. Dasarathy. Nearest neighbor (NN) norms : NN pattern classification techniques. *IEEE Computer Society Press*, 1991.

[Dau92] I. Daubechies. *Ten lectures on wavelets*. Capital City Press, 1992.

[DDG99] S. Derrode, M. Daoudi, and F. Ghorbel. Invariant content-based image retrieval using a complete set of Fourier-Mellin descriptors. *Int. Conf. on Multimedia Computing and Systems (ICMCS'99)*, pages 877–881, june 1999.

[DE05] J. Ducrou and P. Eklund. Combining spatial and lattice-based information landscapes. In *Formal Concept Analysis*, volume 3403, pages 64–78. Springer Berlin, 2005.

[Del05] M. Delalandre. *Analyse de documents graphiques : une approche par reconstruction d'objets*. PhD thesis, Université de Rouen, Rouen, France, 2005.

[DHA+02] M. Delalandre, P. Heroux, S. Adam, E. Trupin, and J.M. Ogier. A statistical and structural approach for symbol recognition using XML modelling. *Structural and Syntactical Pattern Recognition (SSPR)*, 2396 :281–290, 2002. LNCS Springer.

[DHJ06] A. Deruyver, Y. Hodé, and J.M. Jolion. Pyramide adaptative et graphe sémantique : Un processus de segmentation dirigé par la connaissance. In *15ème édition du congrès Reconnaissance des Formes et Intelligence Artificielle (RFIA'06)*, 2006.

[DKS95] J. Dougherty, R. Kohavi, and M. Sahami. *Supervised and unsupervised discretization of continuous features.* Morgan Kaufman, 1995.

[DL04] P. Dosch and J. Lladós. Vectorial signatures for symbol discrimination. In J. Lladós and Y.B. Kwon, editors, *Graphics Recognition : Recent Advances and Perspectives - Selected papers from GREC'03*, volume 3088, pages 154–165. Lecture Notes in Computer Science, 2004.

[DP91] B.A. Davey and H.A. Priestley. *Introduction to lattices and orders.* Cambridge University Press, 2nd edition, 1991.

[DTOL05] M. Delalandre, E. Trupin, J.M. Ogier, and J. Labiche. Contextual system of symbol structural recognition based on an object-process methodology. *Electronic Letters on Computer Vision and Image Analysis*, 5(2) :16–29, 2005.

[Duq07] V. Duquenne. What can lattices do for teaching math. ? In *Fifth International Conference on Concept Lattices and Their Applications*, 2007. To appear.

[EBM07] H.M. El-Bakry and N. Mastorakis. New fast normalized neural networks for pattern detection. *Image and Vision Computing*, 25(11) :1767–1784, November 2007.

[Ekl06] P. Eklund. An intelligent user interface for browsing and search MPEG-7 images using concept lattices. In S. Ben-Yahia and E. Mephu-Nguifo, editors, *Proceedings of Fourth International Conference on Concept Lattices and Their Applications (CLA'06)*, pages 3–24, Yasmine Hammamet (Tunisia), October 30 - November 1st 2006.

[FFJ05] M.J. Fonseca, A. Ferreira, and J.A. Jorge. Generic shape classification for retrieval. In Liu Wenyin and Josep Lladós, editors, *Graphics Recognition : Ten years Review and Future Perspectives - Selected papers from GREC'05*, volume 3926, pages 291–299. Lecture Notes in Computer Science, 2005.

[FI93] U.M. Fayyad and K.B. Irani. *Multi-interval discretization of continuous-valued attributes for classification learning.* Morgan Kaufman, 1993.

[Fis87] D.H. Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2 :139–172, 1987.

[FM71] M.J. Fisher and A.R. Meyer. Boolean matrix multiplication and transitive closure. In *12th Annual Sympsosium on Switching and Automata Theory*, pages 129–131, 1971.

[FMN04] H. Fu and E. Mephu-Nguifo. A parallel algorithm to generate formal concepts for large data. In *2nd Intl. Conference on Formal Concept Analysis (ICFCA)*, volume 2961, pages 394–401, Sydney, Australia, February 23-26 2004. Springer-Verlag, LNAI.

[Fre61] H. Freeman. On the encoding of arbitrary geometric configurations. *Transactions on Electronic and Computer*, 10 :260–268, 1961.

[Gan84] B. Ganter. Two basic algorithms in concept analysis. *Technische Hochschule Darmstadt (Preprint 831)*, 1984.

[GBO] S. Guillas, K. Bertet, and J.M. Ogier. Towards an iterative classification based on concept lattice.

[GBYMNS06] G. Gasmi, S. Ben-Yahia, E. Mephu-Nguifo, and Y. Slimani. IGB : une nouvelle base générique informative des règles d'association. *Revue I3 (Information, Interaction, Intelligence) en Sciences du Traitement de l'Information*, 6(1) :37 pages, octobre 2006.

[GD86] J.L. Guigues and V. Duquenne. Familles minimales d'implications informatives résultant d'un tableau de données binaires. *Mathématiques et Sciences Humaines*, 95 :5–18, 1986.

[Gel05] A. Gely. A generic algorithm for generating closed sets of binary relation. *Formal Concept Analysis, Third Int. Conf., ICFCA 2005*, pages 223–234, February 2005.

[GMA91] R. Godin, R. Missaoui, and H. Alaoui. Learning algorithms using a Galois lattice structure. *Third International Conference on Tools for Artificial Intelligence, San Jose, Calif.*, pages 22–29, 1991.

[GMLPR04] V. Gunes, M. Menard, P. Loonis, and S. Petit-Renaud. Systems of classifiers : state of the art and trends. *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI), World Scientific*, 17(8) :1303–1324, 2004.

[GMM95] R. Godin, G. Mineau, and R. Missaoui. Incremental concept formation algorithms based on galois (concept) lattices. *Computational Intelligence*, 11(8) :246–267, 1995.

[GRC03] V. Guigue, A. Rakotomamonjy, and S. Canu. SVM et k-ppv pour la reconnaissance d'émotions. In *19ème Colloque GRETSI sur le traitement du signal et des images*, page 4 pages, Paris, 2003.

[GRE03] GREC. www.cvc.uab.es/grec2003/symreccontest/index.htm, 2003. Base d'images GREC 2003 (Graphics RECognition), date visite : 08-01-2008.

[GRE05] GREC. www.cs.cityu.edu.hk/grec2005, 2005. Base d'images GREC 2005 (Graphics RECognition), date visite : 08-01-2008.

[GSW02] P. Geibel, K. Schädler, and F. Wysotzki. Learning of class descriptions from class discriminations : A hybrid approach for relational objects. In M. Jarke, J. Koehler, and G. Lakemeyer, editors, *KI 2002 : Advances in Artificial Intelligence*, volume 2479, pages 186–204, Berlin, 2002. Springer, LNAI.

[GW96] P. Geibel and F. Wysotzki. Learning relational concepts with decision trees. In Lorenza Saitta, editor, *Machine Learning : Proceedings of the Thirteenth International Conference*, pages 166–174, San Fransisco, 1996. Morgan Kaufmann Publishers.

[GW99] B. Ganter and R. Wille. *Formal concept analysis, Mathematical foundations*. Springer Verlag, Berlin, 1999.

[HCT07] P.Y. Hao, J.H. Chiang, and Y.K. Tu. Hierarchically SVM classification based on support vector clustering method and its application to document categorization. *Expert Systems with Applications*, 33 :627–635, 2007.

[Hew97] P.C. Hew. Recognition of printed digits affected by rotation noise using zernike moments and gaussian models. Technical report, March 1997.

[Hol93] R.C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11 :63–90, 1993.

[Hot36] H. Hotelling. Relations between two sets of variates. *Biometrika*, XXVIII(2) :321–377, 1936.

[Hou62] P.V.C. Hough. Method and means for recognizing complex patterns. U.S. Patent 3069654, December 1962.

[HPM+98] L. Heutte, T. Paquet, J.V. Moreau, Y. Lecourtier, and C. Olivier. A structural/statistical feature based vector for handwritten character recognition. *Pattern Recognition Letters*, 19 :629–641, 1998.

[HTL03] W. Huang, C.L. Tan, and W.K. Leow. Model-based chart image recognition. In J. Lladós and Y.B. Kwon, editors, *Graphics Recognition : Recent Advances and Perspectives - Selected papers from GREC'03*, volume 3088, pages 87–99. Lecture Notes in Computer Science, 2003.

[Hua97] P.W. Huang. Indexing pictures by key objects for large-scale image databases. *Pattern Recognition*, 30(7) :1229–1237, 1997.

[I.71] Munro I. Efficient determination of the transitive closure of a directed graph. *Information Processing Letter*, pages 56–58, 1971.

[Ing07] J. Inglada. Automatic recognition of man-made objects in high resolution optical remote sensing images by SVM classification of geometric image features. *ISPRS Journal of Photogrammetry and Remote Sensing*, 62 :236–248, 2007.

[JLS04] A.T.B. Jin, D.N.C. Ling, and O.T. Song. An efficient fingerprint verification system using integrated wavelet and fourier-mellin invariant transform. *Image and Vision Computing*, 22 :503–513, 2004.

[Ka94] T. Kanungo and al. Document degradation models : parameter estimation and model validation. In *IAPR Workshop on machine vision applications, Kawasaki (Japan)*, pages 552–557, 1994.

[KF82] D.J. Krus and E.A. Fuller. Computer assisted multicrossvalidation in regression analysis. *Educational and Psychological Measurement*, 42 :187–193, 1982.

[KJL+94] R. Kohavi, G. John, R. Long, D. Manley, and K. Pfleger. *MLC++ : A machine learning library in C++*. IEEE Computer Society Press, 1994.

[KO01] S. Kuznetsov and S. Obiedkov. Comparing performance of algorithms for generating concept lattices. In *Proceedings of ICCS'01 workshop on CLKDD*, volume 42, pages 35–47, July 2001.

[KO02] S. Kuznetsov and S. Obiedkov. Comparing performance of algorithms for generating concept lattices. *Journal of Experimental and Theorical Artificial Intelligence*, 14(2-3) :189–216, 2002.

[Koh96] R. Kohavi. Scaling up the accuracy of naïve-Bayes classifiers : A decision-tree hybrid. In *Proceedings of the second International Conference on Knowledge Discovery and Data Mining*, pages 202–207, Menlo Park, CA, 1996. The AAAI Press.

[KP03] A. Kadyrov and M. Petrou. Object signatures invariant to affine distortions derived from the trace transform. *Image and Vision Computing*, 21 :1135–1143, 2003.

[KP05] K.C. Kwak and W. Pedrycz. Face recognition : a study in information fusion using fuzzy integral. *Pattern Recognition Letters*, 26 :719–733, 2005.

[KS02] C. Kan and M.D. Srinath. Invariant character recognition with Zernike and orthogonal Fourier-Mellin moments. *Pattern Recognition*, 35 :143–154, 2002.

[KTK08] I. Kurt, M. Ture, and T. Kurum. Comparing performances of logistic regression, classification and regression tree, and neural networks for predicting coronary artery disease. *Expert Systems with Applications*, 34 :18–25, 2008.

[LAT+05] H. Locteau, S. Adam, E. Trupin, J. Labiche, and P. Héroux. Symbol recognition combining vectorial and statistical features. In Liu Wenyin and Josep Lladós, editors, *Graphics Recognition : Ten years Review and Future Perspectives - Selected papers from GREC'05*, volume 3926, pages 76–87. Lecture Notes in Computer Science, 2005.

[Lea92] VF. Leavers. Use of the radon transform as a method of extractiong information about shape in two dimensions. *Image and Vision Computing*, 10(2) :99–107, 1992.

[LHM98] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *Proceedings of the fourth International Conference on*

Knowledge Discovery and Data Mining, pages 80–86, New York, USA, 1998.

[LHP01] W. Li, J. Han, and J. Pei. CMAR : Accurate and efficient classification based on multiple class-association rules. In *Proceedings of IEEE International Conference on Data Mining (ICDM'01)*, pages 369–376, San Jose, CA, 2001. IEEE Computer Society.

[LI87] Reuters Ltd. and Carnegie Group Inc. The reuters-21578 text categorization test collection, 1987.

[LL01] B. Lerner and N.D. Lawrence. A comparison of state-of-the-art classification techniques with application to cytogenetics. *Neural Comput.*, 10 :39–47, 2001.

[Lla06] J. Lladós. Perspectives on the analysis of graphical documents. In L. Likforman-Sulem, editor, *Actes du 9ème Colloque International Francophone sur l'Ecrit et le Document (CIFED)*, pages 25–30, 2006.

[LMN90] M. Liquière and E. Mephu-Nguifo. LEGAL : LEarning with GAlois Lattice. In *Actes des Journées Françaises sur l'Apprentissage (JFA)*, pages 93–113, Lannion, France, avril 1990.

[LVS03] J. Lladós, E. Valveny, and G. Sanchez. A case study of pattern recognition : Symbol recognition in graphic documents. In Jean-Marc Ogier and Eric Trupin, editors, *Pattern Recognition in Information Systems*, pages 1–13. Proceedings of the 3rd International Worshop on Pattern Recognition in Information Systems, ICEIS Press, 2003.

[LVSM01] J. Lladós, E. Valveny, G. Sanchez, and E. Marti. Symbol recognition, current advances and perspectives. *Les actes de IAPR International Workshop on Graphics RECgnition (GREC), Kingston, Canada*, pages 109–129, 2001.

[LVSM02] J. Lladós, E. Valveny, G. Sanchez, and E. Marti. Symbol recognition : Current advances and perspectives. In *Graphics recognition, algorithms and applications - Selected papers from GREC'01*, volume 2390, pages 104–127. Lecture Notes in Computer Science (LNCS), 2002.

[MA94] P.M. Murphy and D.W. Aha. Uci repository of machine learning databases, 1994.

[Mal89] S.G. Mallat. Multiresolution approximations and wavelet orthonormal bases of $l^2(r)$. *Trans Am Math Soc*, 315(1) :69–87, 1989.

[MB05] B. Monjardet and K. Bertet. The multiple facets of the canonical direct implicational basis. Technical report, CAMS, October 2005.

[MCP98] R. Milanese, M. Cherbuliez, and T. Pun. Invariant content-based image retrieval using the Fourier-Mellin transform. In *Intl. Conference on Ad-*

vances in Pattern Recognition (ICAPR'98), Forte House, Plymouth, UK, 23-25 November 1998.

[Mer09] J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society*, 209 :415–446, 1909.

[MM96] C.J. Merz and P. Murphy. Uci repository of machine learning databases, 1996.

[MMHL86] R. Michalski, I. Mozetic, J. Hong, and N. Lavrac. The multi-purpose incremental learning system AQ15 and its testing application to three medical domains. In *National Conference on Artificial Intelligence*, pages 1041–1045, 1986.

[MN93] E. Mephu-Nguifo. Une nouvelle approche basée sur le treillis de galois, pour l'apprentissage de concepts. *Mathématiques et Sciences Humaines*, 124 :19–38, 1993.

[MNN05] E. Mephu-Nguifo and P. Njiwoua. Treillis des concepts et classification supervisée. *Technique et Science Informatiques, RSTI*, 24(4) :449–488, 2005. Hermès - Lavoisier, Paris, France.

[MOLP99] E. Magli, G. Olmo, and L. Lo Presti. Pattern recognition by means of the radon transform and the continuous wavelet transform. *Signal Processing*, 73 :277–289, 1999.

[NB86] T. Niblett and I. Bratko. *Learning decision rules in noisy domains.* Cambridge University Press, 1986.

[NK04] M. Novotni and R. Klein. Shape retrieval using 3D Zernike descriptors. *Computer-Aided Design*, 36 :1047–1062, 2004.

[NMN99] P. Njiwoua and E. Mephu-Nguifo. Améliorer l'apprentissage à partir d'instances grâce à l'induction de concepts : le système CIBLe. *Revue d'intelligence Artificielle (RIA)*, 13(2) :413–440, 1999. Hermès Science.

[Nor78] E. Norris. An algorithm for computing the maximal rectangles in a binary relation. *Revue Roumaine de Mathématiques Pures et Appliquées*, 23(2), 1978.

[NR99] L. Nourine and O. Raynaud. A fast algorithm for building lattices. In *Third International Conference on Orders, Algorithms and Applications*, Montpellier, France, august 1999.

[NVRG05] K. Nehme, P. Valtchev, M.H. Rouane, and R. Godin. On computing the minimal generator family for concept lattices and icebergs. In B. Ganter and R. Godin, editors, *Proceedings of the 3rd Intl. Conference on Formal Concept Analysis*, volume 3403, pages 192–207, Lens, France, 14-18 February 2005. Springer Verlag, Lecture Notes in Computer Science.

[OD03] S. Obiedkov and V. Duquenne. Incremental construction of the canonical implication basis. In *Fourth International Conference Journée de l'Informatique messine*, pages 15–23, 2003. submitted to Discrete Applied Mathematics.

[Oos87] G. Deon Oosthuizen. Supergran : A connectionist approach to learning, integrating genetic algorithms and graph induction. In *International Conference on Genetic Algorithms (ICGA)*, pages 132–139, 1987.

[Oos88] G. Oosthuizen. *The use of a Lattice in Knowledge Processing.* PhD thesis, University of Strathclyde, Glasgow, 1988.

[Oos89] G. Deon Oosthuizen. Machine learning : a mathematical framework for neural network, symbolic and genetics-based learning. In *Proceedings of the third international conference on Genetic algorithms*, pages 385–390, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.

[Oos94] G. Deon Oosthuizen. Rough sets and concept lattices. In *RSKD '93 : Proceedings of the International Workshop on Rough Sets and Knowledge Discovery*, pages 24–31, London, UK, 1994. Springer-Verlag.

[PBKM07] G.A. Papakostas, Y.S. Boutalis, D.A. Karras, and B.G. Mertzios. A new class of zernike moments for computer vision applications. *Information Sciences*, 177 :2802–2819, 2007.

[PP00] J.R. Parker and J. Pivovarov. Recognizing symbols by drawing them. *International Journal of Imaging and Graphics*, 1(4) :1–14, 2000.

[QCJ93] J.R. Quinlan and R.M. Cameron-Jones. Foil : A midterm report. In *European Conf. Machine Learning*, pages 3–20, Vienna, Austria, 1993.

[QJH06] R.J. Qureshi, Ramel J.Y., and Cardot H. De l'appariement de graphes symboliques à l'appariement de graphes numériques : Application à la reconnaissance de symboles. In Laurence Likforman-Sulem, editor, *Neuvième Colloque International Francophone sur l'Ecrit et le Document (CIFED'06), dans le cadre de la SDN'06*, pages 31–36, Fribourg, Suisse, Septembre 2006.

[Qui86] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1, 1986.

[Qui87] J.R. Quinlan. Simplifying decision trees. *International Journal of Man-Machine Studies*, 27 :221–234, 1987.

[Qui93] J.R. Quinlan. *C4.5 : Programs for Machine Learning.* Morgan Kaufman, Los Altos, California, 1993.

[Qui96] J.R. Quinlan. *Bagging, boosting and C4.5.* AAAI Press, Menlo Park, CA, 1996.

[R] R. Logiciel libre R, environnement pour le calcul statistique et les représentations graphiques, www.r-project.org.

[Rad17] J. Radon. Über die Bestimmung von Funktionen durch ihre Integralwerte längs gewisser Mannigfaltigkeiten. *Berichte Saechsische Akademie der Wissenschaften*, pages 262–277, 1917.

[Rak97] R. Rakotomalala. *Graphes d'induction*. PhD thesis, Université Claude Bernard, Lyon I, Décembre 1997.

[Rak05] R. Rakotomalala. Arbres de décision. *Revue MODULAD*, 33, 2005.

[RHW86] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. *Learning internal Representations by error propagation*, chapter 8, pages 318–362. MIT Press, parallel distributed processing : explorations in the microstructure of cognition edition, 1986.

[RM07] A. Rodtook and S.S. Makhanov. A filter band method to construct rotationally invariant moments for pattern recognition. *Pattern Recognition Letters*, 28 :1492–1500, 2007.

[RnL05] M. Rusiñol and J. Lladós. Symbol spotting in technical drawings using vectorial signatures. In Liu Wenyin and Josep Lladós, editors, *Graphics Recognition : Ten years Review and Future Perspectives - Selected papers from GREC'05*, volume 3926, pages 35–45. Lecture Notes in Computer Science, 2005.

[Rod00] M.A. Rodrigues. *Invariants for pattern recognition and classification*, volume 42 of *Machine Perception Artificial Intelligence*. World Scientific, 2000.

[Ros58] F. Rosenblatt. The perceptron : a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6) :386–408, 1958.

[Sah95] M. Sahami. Learning classification rules using lattices. In Nada Lavrac and Stefan Wrobel, editors, *Proceedings of ECML'95*, pages 343–346, Heraclion, Crete, Greece, April 1995.

[SHG+05] U. Surapong, M. Hammoud, C. Garrido, P. Franco, and J.M. Ogier. Ancient graphic documents characterization. In *Sixth IAPR Workshop on Graphics Recognition (GREC'05)*, pages 97–105, Hong Kong, China, 2005.

[SHKY04] J.S. Seo, J. Haitsma, T. Kalker, and C.D. Yoo. A robust image fingerprinting system using the radon transform. *Signal Processing : Image Communication*, 19 :325–339, 2004.

[SI99] D. Shen and H.H.S. Ip. Discriminative wavelet shape descriptors for recognition of 2-d patterns. *Pattern Recognition*, 32 :151–165, 1999.

[SSB+97] B. Schölkopf, K. Sung, C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik. Comparing support vector machines with Gaussian kernels to

radial basis function classifiers. *IEEE Transaction on Signal Processing*, 45 :2758–2765, 1997.

[STB+02] G. Stumme, R. Taouil, Y. Bastide, N. Pasquier, and L. Lakhal. Computing iceberg concept lattices with TITANIC. *Data and Knowledge Engineering*, 42(2) :189–222, August 2002.

[SZ04] M. Samuelides and E. Zenou. Learning-based visual localization using formal concept lattices. In *2004 IEEE Workshop on Machine Learning for Signal Processing*, page 10, Sao Luis (Brésil), September 29 - October 1st 2004.

[TB01] R. Taouil and Y. Bastide. Computing proper implications. *Proceedings of ICCS-2001 Internationnal Workshop on Concept Lattices-Based Theory, Methods and tools for Knowledge Discovery in Databases*, pages 290–303, 2001.

[Tea03] M. Teague. Image analysis via the general theory of moments. *Journal of Optical Society of America (JOSA)*, 70 :920–930, 2003.

[TJT96] O.D. Trier, A.K. Jain, and T. Taxt. Features extraction methods for character recognition - a survey. *Pattern Recognition*, 29 :641–662, 1996.

[TL03] K. Tombre and B. Lamiroy. Graphics recognition - from re-engineering to retrieval. *Proceedings of 7th ICDAR, Edinburgh (Scotland, UK)*, pages 148–155, August 2003.

[TS92] G.G. Towell and J.W. Shavlik. Interpretation of artificial neural networks : mapping knowledge-based neural networks into rules. *Advances in Neural Information Processing Systems*, 4, 1992.

[TV06] O.R. Terrades and E. Valveny. A new use of the ridgelets transform for describing linear singularities in images. *Pattern Recognition Letters*, 27 :587–596, 2006.

[TW03a] S. Tabbone and L. Wendling. Adaptation de la transformée de radon pour la recherche d'objets à niveaux de gris et de couleurs. *Technique et Science Informatique*, 22(9) :1041–1068, 2003.

[TW03b] S. Tabbone and L. Wendling. Adaptation de la transformée de Radon pour la recherche d'objets à niveaux de gris et de couleurs. *Technique et Science Informatiques, RSTI*, 22(9) :1139–1166, 2003. Hermès - Lavoisier, Paris, France.

[TWT03] S. Tabbone, L. Wendling, and K. Tombre. Matching of graphical symbols in line-drawing images using angular signature information. *International Journal on Document Analysis and Recognition*, 6(2) :115–125, October 2003.

[Vap98] V.N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[VD04] E. Valveny and P. Dosch. Symbol recognition contest : A synthesis. In J. Lladós and Y.B. Kwon, editors, *Graphics Recognition : Recent Advances and Perspectives - Selected papers from GREC'03*, volume 3088, pages 368–385. Lecture Notes in Computer Science, 2004.

[VML02] P. Valtchev, R. Missaoui, and P. Lebrun. A partition-based approach towards constructing galois (concept) lattices. *Discrete Mathematics*, 3(256) :801–829, 2002.

[VOR97] F.J. Venter, G.D. Oosthuizen, and J.D. Roos. Knowledge discovery in databases using lattices. *Expert Systems With Applications*, 13(4) :259–264, 1997.

[Wil82] R. Wille. Restructuring lattice theory : an approach based on hierarchies of concepts. *Ordered sets*, pages 445–470, 1982. I. Rival (ed.), Dordrecht-Boston, Reidel.

[WK05] J. Wang and G. Karypis. HARMONY : Efficiently mining the best rules for classification. In *Proceedings of the International Conference of Data Mining (SDM'05)*, Newport Beach, CA, 2005.

[WMLW00] J. Waldemark, M. Millberg, T. Lindblad, and K. Waldemark. Image analysis for airbone recognition and missile applications. *Pattern Recognition Letters*, 21 :239–251, 2000.

[WZY06] L. Wenyin, W. Zhang, and L. Yan. An interactive example-driven approach to graphics recognition in engineering drawings. In *International Journal on Document Analysis and Recognition (IJDAR)*, 2006.

[XHLL02] Z. Xie, W. Hsu, Z. Liu, and M. Lee. Concept lattice based composite classifiers for high predictability. *Journal of Experimental and Theoretical Artificial Intelligence*, 14(2/3) :143–156, 2002. Taylor and Francis Ltd.

[XY03] J.H. Xiaoxin Yin. CPAR : Classification based on predictive association rules. In *Proceedings of the International Conference on Data Mining (SDM'03)*, pages 369–376, San Francisco, CA, 2003.

[YW03] L. Yan and L. Wenyin. Interactive recognition of graphic objects in engineering drawings. In J. Lladós and Y.B. Kwon, editors, *Graphics Recognition : Recent Advances and Perspectives - Selected papers from GREC'03*, volume 3088, pages 128–141. Lecture Notes in Computer Science, 2003.

[Zad65] L.A. Zadeh. Fuzzy sets. *Inf. Control*, 8 :338–353, 1965.

[Zad96] L.A. Zadeh. Fuzzy logic and approximate reasoning. pages 238–259, 1996.

[ZS04] E. Zenou and M. Samuelides. Utilisation des treillis de Galois pour la caractérisation d'ensembles d'images. In *14ème Congrès Francophone AFRIF-AFIA de Reconnaissance des Formes et Intelligence Artificielle (RFIA'2004)*, volume 1, pages 395–404, Toulouse (France), Janvier 2004.

[ZT06] D. Zuwala and S. Tabbone. Une méthode de localisation et de reconnaissance de symboles sans connaissance a priori. In Laurence Likforman-Sulem, editor, *Neuvième Colloque International Francophone sur l'Ecrit et le Document (CIFED'06), dans le cadre de la SDN'06*, pages 127–131, Fribourg, Suisse, Septembre 2006.

[Zuw06] Daniel Zuwala. *Reconnaissance de symboles sans connaissance a priori.* PhD thesis, Institut National Polytechnique de Lorraine, 6 novembre 2006.

Annexe A

NAVIGALA

Avec l'aide de deux étudiants de Master, Philippe Sachot et Antoine Mercier, nous avons créé une application pour le classifieur basé sur le treillis de Galois décrit précédemment dans le chapitre 3. Ce logiciel a été appelé NAVIGALA (NAVIgation into GAlois LAttice), porte le même nom que la méthode de reconnaissance qu'il intègre. Nous avons fait le choix au départ de favoriser la portabilité du système, c'est pourquoi l'implémentation a été réalisée en langage Java.

A.1 Description du prototype

L'application NAVIGALA suit exactement le fonctionnement décrit dans le chapitre 3. Il comporte les deux étapes principales de la classification : apprentissage et reconnaissance. L'apprentissage intègre la description des symboles par une signature, la construction de la table discrétisée et puis la génération du treillis de Galois. La reconnaissance s'effectue ensuite par un parcours du graphe. Voici les différentes fonctions proposées dans NAVIGALA ainsi que l'ordre dans lequel elles sont employées :

1. Importation et visualisation d'images de symboles.
2. Paramétrage et calcul des signatures suivantes : Fourier-Mellin, R-signature (Radon), moments de Zernike ; Importation de signatures au format XML. Visualisation des fichiers signatures (rappel visuel du symbole et du paramétrage).
3. Normalisation des signatures (linéaire ou normale).
4. Construction de la table discrétisée et visualisation des intervalles obtenus.

5. Génération du treillis de Galois entier / à la demande et visualisation du treillis obtenu.
6. Reconnaissance simple / itérative. La reconnaissance itérative contient au plus trois étapes. A chaque étape, il est possible d'utiliser la signature de notre choix avec le paramétrage désiré. La même signature peut servir plusieurs fois. Dans ce cas, les descripteurs utilisés lors d'une étape précédente sont supprimés de la signature pour l'étape courante.
7. Visualisation des résultats de la reconnaissance : matrice de confusion et taux de classification.

Les résultats et structures obtenus (signatures, table discrétisée, treillis de Galois, résultats de reconnaissance) sont conservés dans un répertoire propre à l'expérimentation courante.

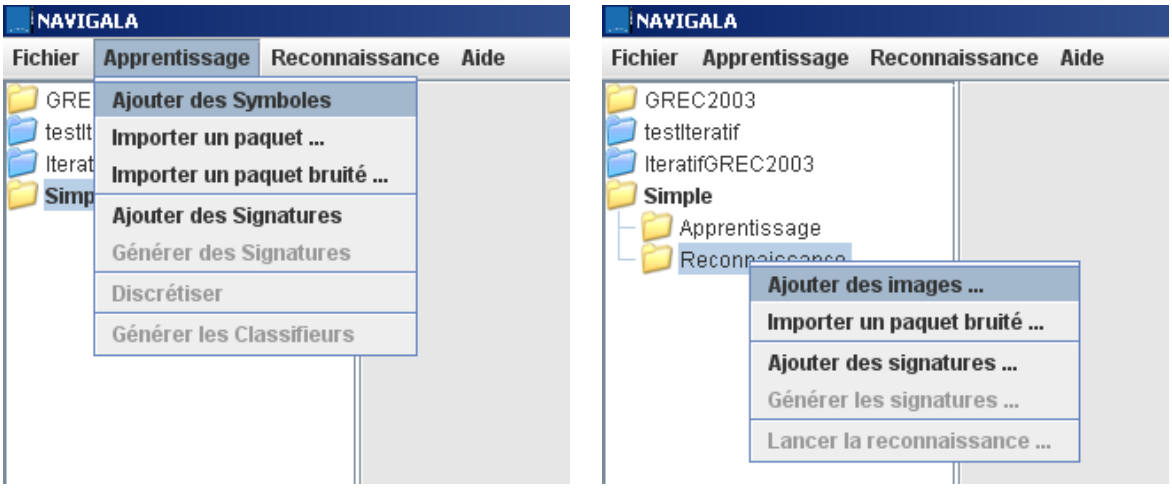
A.2 Interface graphique

Le développement de l'interface a été pensé pour obtenir un résultat visuel agréable et pour une utilisation la plus simple possible. L'ordre des traitements est contraint afin de guider l'utilisateur et d'éviter les erreurs de manipulation. L'interface comprend trois parties : en haut, le menu décline les fonctions présentées dans la description du prototype ; à gauche, une arborescence permet de naviguer dans les différentes expérimentations ; et au centre, la fenêtre principale sert à visualiser les différents résultats obtenus lors des traitements.

Le menu propose l'ensemble des fonctionnalités que nous venons de présenter. L'ordre de lancement des actions est contraint, c'est-à-dire que les boutons sont grisés tant qu'il n'est pas possible de les exécuter. Un menu contextuel équivalent est disponible par un clic droit, à partir des éléments situés dans l'arborescence. La figure A.1 présente le menu principal du bouton Apprentissage et le menu contextuel du bouton Reconnaissance accessible par l'arborescence.

Dans le bandeau de gauche, les différentes expérimentations apparaissent sous forme de projets (ou dossiers). Chaque dossier à la racine représente une expérimentation. Le dossier correspondant à l'expérimentation courante est en gras. Un dossier jaune indique qu'il s'agit d'une reconnaissance simple, et un dossier bleu représente une reconnaissance itérative. Dans ces dossiers d'expérimentation, la structure choisie pour organiser les données est toujours la même. Elle se décline en deux dossiers : Apprentissage et Reconnaissance. Comme son nom l'indique, le dossier Apprentissage renferme

FIG. A.1 – Menu principal (à gauche) et contextuel (à droite)



les données de l'apprentissage. Il contient un dossier Symboles, un/plusieurs dossier(s) Signatures, un/plusieurs fichier(s) d'extension .disc représentant la/les table(s) discrétisée(s) et un fichier d'extension .dot correspondant au treillis de Galois. Le dossier Reconnaissance contient la même hiérarchie que le dossier Apprentissage mais pour les données à reconnaître, avec en plus un fichier d'extension .reco qui décrit les résultats expérimentaux. Un exemple d'arborescence est donné dans la figure A.2. Notons qu'il est possible d'importer directement des fichiers signatures sans avoir à les calculer à partir des images.

La fenêtre principale sert à visualiser les données. Un double-clic dans le bandeau de gauche sélectionne un élément qui apparaît dans cette fenêtre principale. A chaque élément correspond un onglet qui apparaît en tête de cette fenêtre et facilite la navigation. Les onglets sont de couleur différente pour différencier les données issues de l'apprentissage (bleu) de celles issues de la reconnaissance (vert). Pour les signatures calculées par NAVIGALA, les onglets contiennent bien évidemment les valeurs des invariants obtenus, mais également un rappel visuel du symbole correspondant, ainsi que les paramètres sélectionnés pour l'obtention de la signature. La figure A.3 donne un aperçu de la fenêtre principale lorsque différents onglets bleus et verts sont ouverts. L'onglet principal en cours de visualisation correspond à la signature de Radon d'un symbole. Pour finir, la figure A.4 donne une vision d'ensemble de l'interface avec l'affichage d'un treillis de Galois.

FIG. A.2 – Arborescence contenant différentes expérimentations

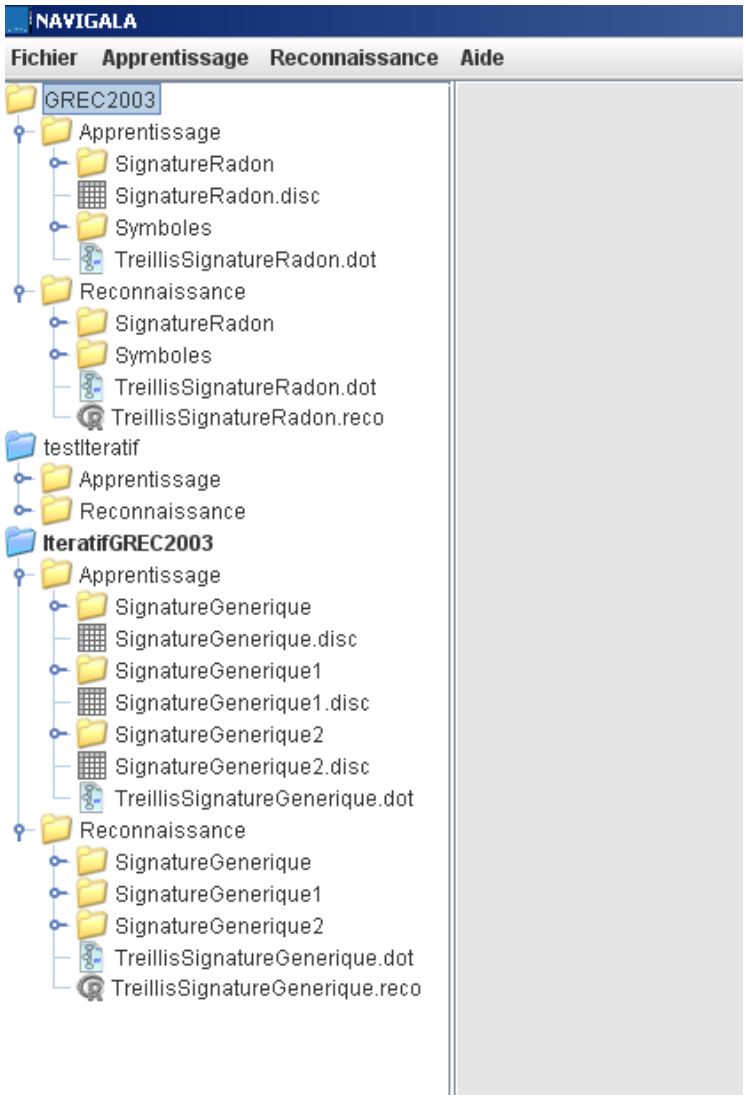


FIG. A.3 – Onglets de couleur de la fenêtre principale et visualisation d'une signature

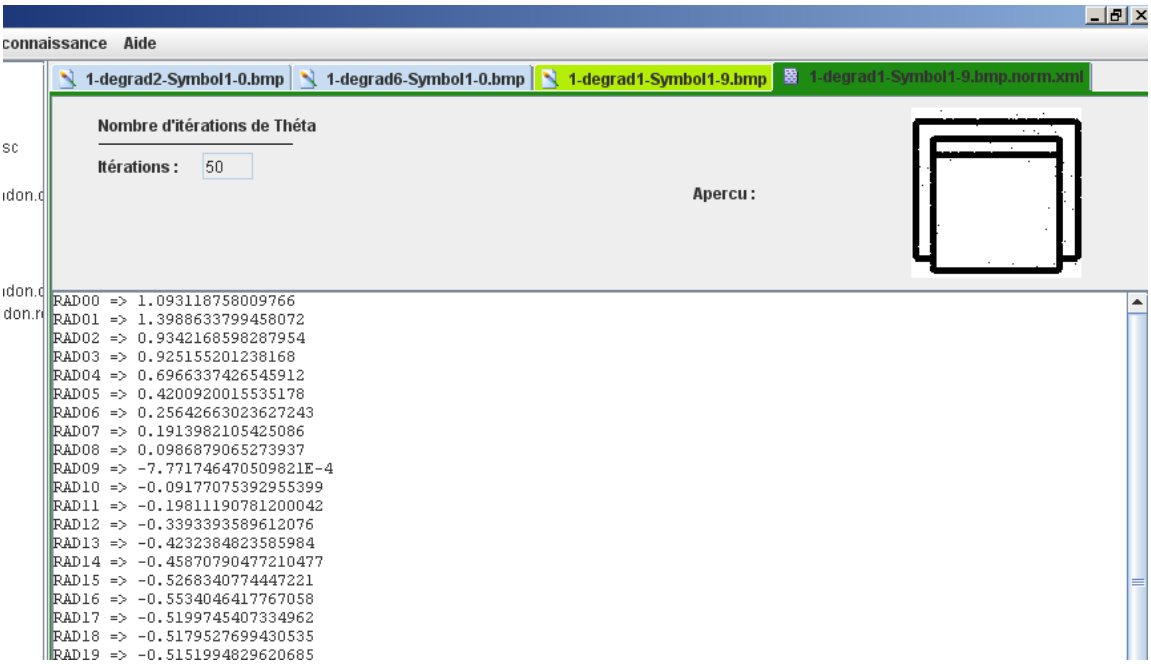
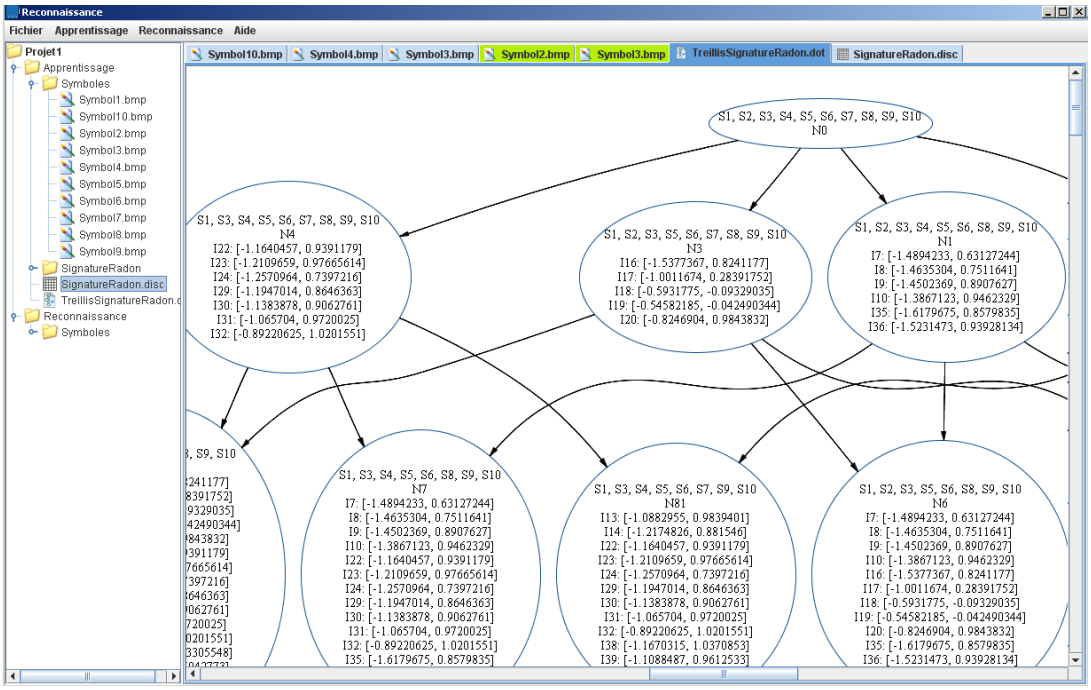



FIG. A.4 – Vue d'ensemble de l'interface et visualisation d'un treillis de Galois