Livrable 3.2
Clustering collaboratif
supervisée, non supervisée et guidé

Projet ANR-10-COSI-012

FOSTER

FOuille de données Spatio-Temporelles

Application à la compréhension et à la surveillance de l'ERosion

Programme COSINUS 2010

Juillet 2013

Mots-clés :
Fouille de données, données spatio-temporelles, érosion, Nouvelle-Calédonie, Barcelonnette

Ce rapport constitue un livrable du projet FOSTER (ANR-10-COSI-012).

Il est construit principlement à partir de deux documents.

- Pour les chapitres 1 et 2 : Connaissances et clustering collaboratif d'objets complexes multi-sources, Thèse de Doctorat de l'Université de Strasbourg, Germain Forestier (2010)
- Pour le chapitre 3 : État de l'art sur les méthodes statistiques d'apprentissage actif par Alexis Bondu et Vincent Lemaire (France Telecom R&D)

Table des matières

1 État de l'art : Apprentissage collaboratif **7**
- 1.1 Introduction . . . 7
  - 1.1.1 Approches supervisées . . . 7
  - 1.1.2 Approches non supervisées . . . 7
- 1.2 Combinaison de plusieurs méthodes de clustering . . . 8
  - 1.2.1 Approches par ensemble . . . 8
  - 1.2.2 Approches multiobjectifs . . . 14
  - 1.2.3 Approches par combinaison de méthodes floues . . . 15
  - 1.2.4 Bilan . . . 15
- 1.3 Le clustering collaboratif . . . 16
  - 1.3.1 Problématique . . . 16
  - 1.3.2 Présentation de l'approche SAMARAH . . . 16
- 1.4 Résolution de conflits en clustering collaboratif . . . 24
  - 1.4.1 Problématique . . . 24
  - 1.4.2 Approche itérative pour la résolution de conflits . . . 25
  - 1.4.3 Approche évolutionnaire pour la résolution de conflits . . . 26

2 État de l'art : Clustering semi-supervisé **29**
- 2.1 Introduction . . . 29
- 2.2 Connaissances . . . 30
  - 2.2.1 Contraintes entre les objets . . . 30
  - 2.2.2 Ensemble d'objets étiquetés . . . 31
  - 2.2.3 Autres types de connaissances . . . 34
- 2.3 Intégration de connaissances en clustering collaboratif . . . 35
  - 2.3.1 Intégration des connaissances a posteriori . . . 36
  - 2.3.2 Guider le processus par les connaissances . . . 37
  - 2.3.3 Utilisation des connaissances dans les méthodes . . . 37

3 État de l'art : Apprentissage actif **39**
- 3.1 Introduction . . . 39
- 3.2 Méthodes d'apprentissage actif . . . 39
  - 3.2.1 Echantillonnage par incertitude . . . 40
  - 3.2.2 Echantillonnage par comité de modèles . . . 40
  - 3.2.3 Arbre sur l'espace des versions . . . 40
- 3.3 Conclusion . . . 40

4 Etudes en cours **43**
- 4.1 Classification collaborative multi-échelle . . . 43
- 4.2 Active learning . . . 43
- 4.3 Approches collaboratives multi-experts . . . 44

Introduction

Lorsque l'on différencie la classification supervisée et la classification non supervisée (clustering), l'élément clé réside dans la présence ou l'absence d'objets étiquetés permettant l'apprentissage d'une fonction de classification. Ces éléments étiquetés consistent en un ensemble d'objets dont on connait la classe ou l'étiquette.

Il existe de nombreuses méthodes de classification et de clustering. Celles-ci peuvent, à partir des mêmes données, fournir des résultats différents. Cette multitude de méthodes impose à l'expert de choisir une méthode ainsi que ses paramètres. Ce choix est lourd de conséquences et va conditionner le type de résultats qu'il pourra obtenir. Une connaissance importante du problème est nécessaire pour choisir "LA" méthode de classification/clustering de manière éclairée. Bien sûr, si l'expert n'a aucune connaissance sur ses données et sur le fonctionnement des algorithmes, il lui sera difficile de faire un choix sensé. Cependant, même avec une expertise sur les données et les algorithmes, il est bien souvent difficile de faire le *bon* choix.

Pour tenter de résoudre ce problème, la communauté scientifique s'est intéressée depuis plusieurs années à la combinaison de plusieurs méthodes de classification. L'objectif est d'utiliser l'avis de plusieurs méthodes dans le but d'en dégager un consensus ou une synthèse. Ces approches se basent sur l'intuition que combiner des informations fournies par différents acteurs, ou experts, peut améliorer la solution proposée à un problème. Le marquis de Condorcet, un philosophe, mathématicien et politologue français a été l'un des premiers à formaliser cette notion dans *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix*. Il décrit, ce qui sera par la suite appelé le théorème du jury de Condorcet. Dans ce théorème, il formalise la probabilité relative d'un groupe d'individus à arriver à une solution correcte. La version la plus simple du théorème stipule que le vote à la majorité d'un ensemble d'individus qui se prononcent indépendamment sur un problème à deux issues permet de réduire le risque d'erreur si ces individus font un choix correct avec une probabilité d'au moins $1/2$. Dans ce cas, plus le nombre d'individus votant augmente, plus la probabilité de faire une erreur de jugement en utilisant un vote diminue, jusqu'à devenir nulle pour une infinité de votants. Cependant, si la probabilité de se tromper parmi les votants est potentiellement supérieure à $1/2$, alors l'ajout d'individus votant ne fait qu'augmenter la probabilité d'erreur. Le tableau 1.1 montre un exemple de ce type de vote [Kuncheva, 2008].

Dans le cadre de Foster, nous sous sommes intéressés principalment aux méthodes non supervisées. De fait, le chapitre 1 présente les avancées en clusterings collaboratives.

Les éléments étiquetés sont généralement construits manuellement par l'expert, ce qui est très coûteux en temps. Ils sont vus ici comme l'expression de la connaissance de l'expert. Dans la majorité des problèmes rencontrés, on dispose généralement d'un grand nombre d'objets non étiquetés et de pas ou très peu d'objets étiquetés. Dans les approches de fouille dites classiques, les méthodes supervisées ne s'intéressent qu'aux objets étiquetés et les méthodes non supervisées qu'aux objets non étiquetés.

Depuis quelques années, de nouvelles méthodes de classification ont fait leur apparition.

– Apprentissage *semi-supervisé*, *partiellement supervisées* ou encore *hybrides*. Sous ces termes se regroupent un ensemble de méthodes et d'approches qui utilisent à la fois des données non étiquetées et des données étiquetées ainsi que d'autres types de connaissances. Ces nouvelles méthodes apportent une solution aux problèmes pour lesquels l'utilisation de méthodes totalement automatiques n'est pas satisfaisant. Il convient alors de différencier deux types d'approches : la classification semi-supervisée et le clustering semi-supervisé.

  - En clustering semi-supervisé, les objets étiquetés sont utilisés pour fournir des informations à l'algorithme de clustering qui va être guidé par ces connaissances dans sa recherche de clusters pertinents. Le clustering semi-supervisé est donc une opération transductive, car son objectif est de définir des clusters pour expliquer les données traitées, et éventuellement d'étiqueter les objets non étiquetés au départ.
  - En classification semi-supervisée (Chapelle et Zien, 2005)les données étiquetées sont utilisés pour construire un modèle généralisant ces données. Ensuite, les données non étiquetées sont utilisées pour affiner le modèle prédictif. On peut par exemple utiliser les données non étiquetées pour estimer les densités de probabilité de chacune des classes (Chappelle, 2005). La classification est une opération inductive, dont l'objectif est de créer un classifieur qui généralise le modèle des données disponibles, et qui peut être utilisé par la suite pour traiter d'autres données.
- l'Apprentissage actif. Il s'agit icci de construire l'ensemble d'apprentissage au cours de l'entraînement du modèle, en interaction avec un expert (humain). L'apprentissage débute avec peu de données étiquetées. Ensuite, le modèle sélectionne les exemples (non étiquetés) qu'il juge les plus "instructifs" et interroge l'expert à propos de leurs étiquettes. La particularité de l'apprentissage actif réside dans l'interaction du modèle avec son environnement. Contrairement à la stratégie "passive" où les exemples sont choisis avant l'apprentissage, de manière aléatoire, les stratégies "actives" permettent d'accélérer l'apprentissage en considérant d'abord les exemples les plus informatifs. Cette approche est particulièrement avantageuse lorsque les données sont coûteuses à acquérir et à étiqueter

Or, ceci est bien le cas dans Foster, où l'étiquettage des zones d'érosion s'est avéré beaucoup plus fastidieux que prévu. De fait nous sous sommes intéressés à ces approches. Ainis, le chapitre 2 présente un état de l'art sur les approches de clustering semi-supervisé alors que le chapitre 3 présente un état de l'art sur ces approches.

Chapitre 1

État de l'art : Apprentissage collaboratif

1.1 Introduction

Voici les notations utilisées dans les prochaines sections :
Notations :
- Soit $X = \{x_1, \ldots, x_n\}$ l'ensemble des n objets à classer ;
- Soit $\mathbb{C} = \{\mathcal{C}^{(1)}, \ldots, \mathcal{C}^{(N)}\}$ un ensemble de N résultats de clustering de X ;
- Soit $\mathcal{C}^{(i)} = \{C_1^{(i)}, \ldots, C_{K^{(i)}}^{(i)}\}$ un résultat de clustering de X en $K^{(i)}$ clusters ;
- Soit $n_k^{(i)} = |C_k^{(i)}|$ le nombre d'objets du cluster $C_k^{(i)}$;
- Soit $\alpha_{k,l}^{(i,j)} = |C_k^{(i)} \cap C_l^{(j)}|$ le nombre d'objets commun au cluster $C_k^{(i)}$ et au cluster $C_l^{(j)}$.

1.1.1 Approches supervisées

En classification supervisée, le théorème de Condorcet a pu facilement être utilisé et adapté, le domaine de définition de la variable cible, c'est-à-dire l'ensemble des classes, étant fixé, connu et commun à tous les classifieurs. Il est alors relativement simple de créer plusieurs modèles différents et de les combiner suivant leur prédiction. De nombreuses méthodes de combinaison ont été proposées [Kittler et al., 1998], ce domaine étant communément appelé classification par ensemble ou classification par méthode ensembliste. Ces méthodes de combinaison supervisées peuvent se contenter de fusionner les décisions de plusieurs classifieurs, mais certaines proposent également de modifier la manière dont l'algorithme de classification supervisée est appliqué (peu importe le choix de celui-ci). Ainsi, la méthode du STACKING [Wolpert, 1992] cherche à construire un méta-modèle en utilisant plusieurs méthodes de classification différentes. De cette manière, le STACKING tente de réduire le biais induit par la sélection d'une unique méthode. La méthode du BAGGING [Breiman, 1996] consiste à utiliser plusieurs échantillonnages des données et construire plusieurs classifieurs, dit faibles, c'est-à-dire que leur prédiction n'est que légèrement supérieure à l'aléatoire. Ces classifieurs faibles sont combinés par un méta-classifieur dit fort, qui va combiner les prédictions souvent en utilisant un vote [Kittler et al., 1998]. Enfin, la méthode de BOOSTING [Schapire, 1990] fonctionne de manière similaire mais pondère les classifieurs faibles lors de la combinaison suivant leur précision de classification ou un autre critère (par exemple une connaissance externe sur la qualité des sources). De plus, le choix des échantillons est dirigé afin de sélectionner les exemples difficiles à classer. De cette manière, l'algorithme de classification va se concentrer sur les exemples posant le plus de problèmes.

1.1.2 Approches non supervisées

Dans le cas non supervisé, ce type d'approches par combinaison est plus compliqué à mettre en place. En effet, lorsque l'on va générer plusieurs résultats de clustering à partir d'un même jeu de données, on disposera pour chaque objet uniquement de l'information de son appartenance aux

Classifieur	Prédiction	Précision
classifieur 1	□■□■■□□□□■□□□□■	10/15 = 0.667
classifieur 2	■■□□□□■□□□□□■□■	10/15 = 0.667
classifieur 3	□□■□□■■□□□□□■■□	10/15 = 0.667
vote	□■□□□□■□□□□□■□■	11/15 = 0.733

□ = correct, ■ = erreur.

TABLE 1.1 – Exemple d'amélioration d'un résultat par l'utilisation d'un vote à la majorité en classification supervisée.

clusters de chaque résultat. Comme il n'existe pas de lien évident entre les clusters d'un résultat et les clusters d'un autre résultat de clustering, il est difficile de comparer leur prédiction. Soient deux résultats de clustering $\{C_1^{(1)}, C_2^{(1)}, \ldots, C_{K^{(1)}}^{(1)}\}$ et $\{C_1^{(2)}, C_2^{(2)}, \ldots, C_{K^{(2)}}^{(2)}\}$, rien ne garantit que les deux clusters $C_k^{(.)}$ de ces deux résultats représentent sensiblement les mêmes groupes d'objets ($C_1^{(1)} \cap C_1^{(2)} = \varnothing$). De plus, les deux résultats peuvent tout à fait avoir un nombre de clusters différent ($K^{(1)} \neq K^{(2)}$).

Pour résoudre le fait que les clusters de différents résultats n'ont pas de lien direct, différentes méthodes permettant en partie de s'affranchir de cette correspondance ont été proposées dans la littérature. Les méthodes dites d'*ensemble clustering*, qui seront présentées en détail dans la section 1.2.1, tentent de trouver la partition *consensus*, résumant au mieux un ensemble de partitions donné. Seule la distribution des objets dans les clusters de chaque résultat est disponible. Les méthodes d'ensemble clustering s'intéressent uniquement à la construction de cette partition consensuelle, ne modifient pas les partitions données en entrée, et n'en créent pas de nouvelles. Par conséquent, pour que les méthodes soient réellement efficaces, les partitions fournies en entrée doivent être de qualité et de complémentarité suffisantes.

De manière sensiblement différente, les approches multiobjectifs présentées dans la section 1.2.2, considèrent le processus de clustering comme l'optimisation de plusieurs objectifs. Dans ces méthodes, un nombre important de résultats de clustering sont explorés, et ceux qui répondent le mieux aux différents objectifs sont conservés. Un algorithme génétique est généralement utilisé, et de nouveaux résultats de clustering sont créés en mélangeant les différents résultats disponibles. Cependant, les objectifs utilisés par les méthodes d'optimisation ne sont pas toujours ceux utilisés pour créer les partitions initiales, ce qui introduit un nouveau biais.

Dans la collaboration de méthodes floues, présentée en détail dans la section 1.2.3, l'algorithme FUZZY-C-MEANS [Dunn, 1974] est utilisé pour créer des clusters de différentes vues des données de manière collaborative. Ces travaux [Pedrycz, 2002] se concentrent uniquement sur l'algorithme FUZZY-C-MEANS ce qui limite son utilisation. Une autre méthode a récemment été proposée, basée sur l'algorithme SOM [Grozavu and Bennani, 2010].

Dans la section 1.3 nous étudions la méthode de clustering collaboratif développée par Wemmert et Gançarski [Wemmert, 2000]. Nous présentons le fonctionnement de la méthode et nous proposons des exemples pour illustrer son fonctionnement. Enfin, nous détaillons les solutions proposées lors du présent travail de thèse sous la forme d'une nouvelle méthode de clustering collaboratif. Des évaluations sont présentées pour illustrer la pertinence de cette nouvelle approche.

1.2 Combinaison de plusieurs méthodes de clustering

1.2.1 Approches par ensemble

Applications

Le but du clustering par ensemble est de produire un unique clustering à partir d'un ensemble de plusieurs clusterings donné. Les approches par ensemble ont été utilisées dans de nombreux

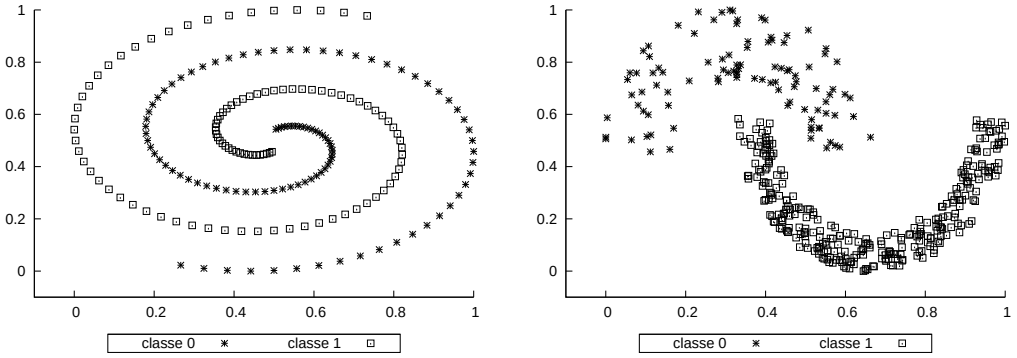


FIGURE 1.1 – Exemples de jeux de données avec des classes de formes spécifiques.

domaines, nous allons étudier les principales applications recensées dans la littérature [Punera and Ghosh, 2007].

Une des applications du clustering par ensemble est la réutilisation de connaissances acquises précédemment sous la forme de partitions existantes. Par exemple, lorsque des données sont reclassées, un clustering antérieur peut être utilisé comme une connaissance disponible. De plus, les informations utilisées pour créer cet ancien résultat ne sont peut-être plus disponibles, les données ayant évoluées, ou la partition ayant été produite par un expert humain [Strehl and Ghosh, 2002].

Une autre application s'intéresse à la classification multivue. Un objet peut être décrit par différents attributs qui ne sont pas forcément compatibles et il peut être impossible de les utiliser dans un processus de clustering unique. Ces objets sont classés en fonction des différents sous-ensembles d'attributs, et les résultats obtenus sont combinés grâce à une méthode par ensemble [Kreiger and Green, 1999]. Par exemple, dans des domaines comme le clustering d'images ou de documents web basé sur des annotations fournies par des experts, le fait que plusieurs sources d'informations possède chacune son propre jeu d'annotations rend souvent difficile l'application d'un processus de clustering unique (les étiquettes des classes pouvant être différentes pour chaque expert). Le chapitre 6 abordera ces concepts en détail.

Récemment, les méthodes de clustering par ensemble ont trouvé des applications dans le cadre de la préservation de la vie privée (*privacy-preserving*) [Aggarwal and Yu, 2008] et en fouille de données distribuées [Merugu and Ghosh, 2003]. En effet, il n'est pas toujours possible de centraliser toutes les données sur un site unique pour effectuer le processus de classification. On pourra par exemple citer le cas d'entreprises possédant des données commerciales sur leurs activités. Chacune de ces entreprises possèdent des données propres et celles-ci ne sont pas forcément prêtes à les partager avec les autres acteurs du marché. Il est cependant envisageable d'effectuer des clusterings indépendants sur chacune de ces données, et d'utiliser une approche par ensemble pour obtenir une solution finale (il faut évidement que celles-ci partagent une liste d'objets commune comme par exemple une liste de clients).

Les méthodes par ensemble ont pour objectif l'amélioration de la qualité des résultats en réduisant le biais induit par chaque algorithme, et donc d'améliorer la qualité du résultat final [Hadjitodorov and Kuncheva, 2007; Hadjitodorov et al., 2006]. Certaines méthodes permettent de trouver des clusters de forme spécifique comme des spirales (voir figure 1.1). En effet, utiliser plusieurs clusterings permet à certaines méthodes par ensemble de mieux saisir la distribution spatiale des données.

L'utilisation de plusieurs résultats de clustering augmente aussi la robustesse des solutions. De nombreux algorithmes souffrent de problèmes d'initialisation et tombent dans des minima locaux de leur fonction objective. Le clustering par ensemble apporte une solution à ces problèmes en utilisant des résultats issus d'un même algorithme mais initialisé à chaque fois différemment [Fred and Jain, 2005; Fred, 2001].

Problématique

La classification par ensemble peut être formellement définie comme la recherche d'un résultat de clustering $\mathcal{C}^{(*)}$ représentant au mieux la structure des données X, à partir de l'ensemble des informations disponibles dans les N résultats de $\mathbb{C}$. Pour cela, il est nécessaire de définir une fonction Φ prenant en entrée les N résultats de $\mathbb{C}$ et produisant un résultat de clustering $\mathcal{C}^{(*)}$ appelé communément *consensus* :

$$\Phi : \mathbb{C} \rightarrow \mathcal{C}^{(*)} \tag{1.1}$$

Le résultat $\mathcal{C}^{(*)}$ peut être vu comme une moyenne des résultats de l'ensemble, et donc comme étant le clustering le plus similaire (selon une fonction de similarité sim) à l'ensemble des N résultats de $\mathbb{C}$ parmi l'ensemble des résultats de clustering possibles $\breve{\mathbb{C}} = \{\breve{\mathcal{C}}^{(1)}, \breve{\mathcal{C}}^{(2)}, \ldots, \breve{\mathcal{C}}^{(m)}\}$ de X :

$$\mathcal{C}^{(*)} = \arg \max_{\breve{\mathcal{C}}^{(i)} \in \breve{\mathbb{C}}} \sum_{q=1}^{N} sim(\breve{\mathcal{C}}^{(i)}, \mathcal{C}^{(q)}) \tag{1.2}$$

Il est cependant impossible d'effectuer une recherche exhaustive parmi tous les résultats possibles $\breve{\mathbb{C}} = \{\breve{\mathcal{C}}^{(1)}, \breve{\mathcal{C}}^{(2)}, \ldots, \breve{\mathcal{C}}^{(m)}\}$ leur nombre étant dissuasif : $m = \frac{1}{K!} \sum_{k=1}^{K} \binom{K}{k} (-1)^{K-k} k^n$ (par exemple 171 798 901 possibilités de former 4 groupes à partir de 16 objets). De plus, quand ce problème est vu sous la forme d'un problème d'optimisation, la recherche d'un consensus devient un problème NP-complet [Megiddo and Supowit, 1984] car se rapportant à un problème de coloration de graphe.

Les travaux sur le clustering par ensemble s'intéressent essentiellement à deux aspects. Le premier est l'étude des méthodes permettant de générer les différents résultats composant $\mathbb{C}$. Le second est la définition de la fonction Φ permettant de générer le consensus. Dans les sections suivantes, nous verrons tout d'abord les principaux domaines d'application de la classification par ensemble. Nous verrons ensuite les principales méthodes pour générer l'ensemble $\mathbb{C}$ des résultats. Enfin les différentes approches abordées dans la littérature pour définir la fonction Φ permettant de trouver un consensus seront présentées.

Génération des résultats initiaux

Il existe différentes stratégies permettant de générer les résultats de clustering initiaux. Pour que le résultat final soit pertinent, il est nécessaire d'introduire une certaine diversité au sein des résultats initiaux. En effet, si tous les résultats de l'ensemble sont identiques, les combiner ne permettra pas d'améliorer le résultat final. Plusieurs heuristiques ont été proposées pour assurer cette diversité :

- utiliser différents algorithmes de clustering ;
- utiliser différentes initialisations d'un même algorithme ;
- utiliser différents sous-ensembles d'instances par échantillonnage ;
- utiliser différents sous-ensembles d'attributs ;
- utiliser des projections dans des sous-espaces différents.

Une description détaillée des différentes heuristiques permettant de générer ces résultats initiaux ainsi que leurs intérêts ont été donnés dans [Hadjitodorov et al., 2006] et [Hadjitodorov and Kuncheva, 2007]. Il a été montré qu'il est important d'obtenir une certaine diversité au sein de l'ensemble notamment en terme de nombre de clusters. En effet, combiner des résultats fortement corrélés réduit les possibilités d'une amélioration, le consensus ne pouvant être alors de qualité supérieure aux résultats de l'ensemble. Cependant ces approches nécessitent le calcul d'un nombre important de résultats pour être efficace, c'est-à-dire plusieurs dizaines voire plusieurs centaines. Si le jeu de données est volumineux, ces approches sont donc difficilement applicables.

Les techniques d'échantillonnage (*resampling*) sont souvent utilisées comme dans [Minaei-Bidgoli et al., 2004a,b]. L'intérêt de combiner des partitions créées par des clusterings *faibles* (*weak clustering*) a également été montré par Topchy et al. [2005]. Ces clusterings sont produits en utilisant des heuristiques simples sur les données comme par exemple une projection sur un axe aléatoire.

Méthodes de combinaison basées sur les graphes

L'une des premières approches pour le clustering par ensemble a été proposée par Strehl and Ghosh [2002] qui introduisent différents cadres d'application de ces méthodes et décrivent trois algorithmes permettant de calculer un consensus à partir d'un ensemble de résultats. La recherche d'un consensus est définie comme la recherche d'un résultat $\mathcal{C}^{(*)}$ qui partage le plus d'information avec les résultats de l'ensemble $\mathbb{C}$. Pour quantifier ce partage d'information, les auteurs utilisent la notion d'information mutuelle (*mutual information*), qui est une mesure symétrique pour quantifier statistiquement l'information partagée par deux distributions [Cover and Thomas, 2006]. L'objectif du clustering par ensemble tel que défini par Strehl and Ghosh [2002] est de trouver le consensus maximisant l'information mutuelle normalisée (NMI) avec les membres de l'ensemble :

$$\mathcal{C}^{(*)} = \arg\max_{\check{\mathcal{C}}^{(i)} \in \check{\mathbb{C}}} \sum_{q=1}^{N} NMI(\check{\mathcal{C}}^{(i)}, \mathcal{C}^{(q)}) \tag{1.3}$$

Pour trouver ce consensus, les trois algorithmes proposés utilisent la notion d'hypergraphe pour représenter l'ensemble des résultats de clustering. Un hypergraphe est un ensemble de sommets et d'hyper-arêtes, une hyper-arête étant une généralisation du concept d'arête pouvant être connectée à un ensemble de sommets. Pour chaque résultat $\mathcal{C}^{(i)} \in \mathbb{C}$, une matrice binaire d'appartenance $H^{(i)}$ est créée, composée d'une colonne pour chaque cluster du résultat (voir exemple tableau 1.2). La concaténation de l'ensemble des matrices $H = (H^{(1)} \dots H^{(N)})$ représente la matrice d'adjacence d'un hypergraphe à N sommets et $\sum_{i=1}^{N} K^{(i)}$ hyper-arêtes. Chaque colonne h_i définit une hyper-arête où 1 indique que cette hyper-arête contient le sommet et 0 qu'elle ne le contient pas. Les trois algorithmes proposés utilisent cette représentation.

La première méthode appelée CSPA (*Cluster-based Similarity Partioning Algorithm*), est basée sur la création d'une mesure de similarité entre les objets à partir des informations contenues dans les résultats de l'ensemble. Pour représenter cette similarité, une matrice S de taille $n \times n$ (n étant le nombre d'objets) est calculée à partir de la matrice H tel que $S = \frac{1}{N} HH^T$. Cette matrice est ensuite utilisée dans un algorithme de clustering classique pour générer le consensus.

Le seconde approche appelée HPGA (*HyperGraph Partitioning Algorithm*), utilise directement la méthode de partitionnement d'hypergraphe HMETIS [Karypis et al., 1997] pour partitionner l'hypergraphe H représentant l'ensemble des clusterings. L'objectif de cette méthode est de créer un clustering consensus qui coupe le moins d'hyper-arêtes.

Enfin, la troisième approche appelée MCLA (*Meta-CLustering Algorithm*) crée des méta-groupes au sein de l'hypergraphe des clusterings en fusionnant des clusters similaires. Ces méta-groupes sont ensuite utilisés pour déterminer le clustering final. La similarité entre deux clusters est calculée à partir du nombre d'instances qui sont regroupées ensemble dans ces deux clusters en utilisant l'indice de Jaccard (voir Annexe B). Le graphe où les clusters ont été fusionnés est ensuite partitionné en utilisant la méthode de partitionnement d'hypergraphe HMETIS. Un vecteur d'association entre les instances et les clusters est créé au sein de chaque méta-groupe. Les instances sont ensuite classées dans le meta-cluster ayant le plus fort degré d'association.

Dans une autre approche proposée par Fern and Brodley [2004], nommée HBGF (*Hybrid Bipartite Graph Formulation*), le problème est ramené à trouver une partition d'un graphe bipartite pour la formation du consensus. L'algorithme CSPA modélise l'ensemble comme un graphe dont les sommets représentent les instances, alors que l'algorithme MCLA modélise l'ensemble comme un graphe de clusters. L'algorithme HBGF combine ces deux approches et représente l'ensemble par un graphe bipartite où les instances ainsi que les clusters forment des sommets. Le graphe est bipartite car il n'y a pas d'arête entre les deux différents types de sommets (instance et cluster). Des règles sont définies pour affecter un poids $W(i,j)$ entre deux sommets (i,j), sur les arêtes du graphe :

- $W(i,j) = 0$ si i et j sont deux clusters ou deux instances
- $W(i,j) = 0$ si l'instance i n'appartient pas au cluster j
- $W(i,j) = 1$ si l'instance i appartient au cluster j

Le graphe bipartite est ensuite partitionné pour former le consensus. Cette classification est effectuée en utilisant l'algorithme HMETIS ou une méthode de clustering spectral [Dhillon, 2001].

	$\mathcal{C}^{(1)}$	$\mathcal{C}^{(2)}$	$\mathcal{C}^{(3)}$
x_1	1	3	1
x_2	1	3	1
x_3	2	1	1
x_4	2	1	1
x_5	3	2	2
x_6	3	2	2

Résultats de clustering

	$H^{(1)}$			$H^{(2)}$			$H^{(3)}$	
	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8
x_1	1	0	0	0	0	1	1	0
x_2	1	0	0	0	0	1	1	0
x_3	0	1	0	1	0	0	1	0
x_4	0	1	0	1	0	0	1	0
x_5	0	0	1	0	1	0	0	1
x_6	0	0	1	0	1	0	0	1

Matrice de transition du graphe.

TABLE 1.2 – Exemple de représentation par hypergraphe de trois résultats de clustering.

	$\mathcal{C}^{(1)}$	$\mathcal{C}^{(2)}$	$\mathcal{C}^{(3)}$
x_1	1	3	1
x_2	1	3	1
x_3	2	1	1
x_4	2	1	1
x_5	3	2	2
x_6	3	2	2

Résultats de clustering.

	x_1	x_2	x_3	x_4	x_5	x_6
x_1	-	3	1	1	0	0
x_2	3	-	1	1	0	0
x_3	1	1	-	3	0	0
x_4	1	1	3	-	0	0
x_5	0	0	0	0	-	3
x_6	0	0	0	0	3	-

Matrice de co-association.

TABLE 1.3 – Exemple de calcul de la matrice de co-association.

Méthodes de combinaison basées sur la matrice de co-association

Une autre approche dans le clustering par ensemble est présentée par Fred and Jain [2005] qui introduisent le concept d'accumulation de preuves (*evidence accumulation*). L'idée principale de cette approche est d'utiliser la *matrice de co-association* calculée à partir des résultats de l'ensemble. Cette matrice, de taille $n \times n$ (n étant le nombre d'objets), donne l'information du nombre de fois où deux objets ont été classés ensemble dans le même cluster dans les différents résultats de l'ensemble. Un exemple de calcul de cette matrice est donné dans le tableau 1.3. La matrice de co-association contient à l'indice (i, j) le nombre de fois où les objets (x_i, x_j) ont été classés ensemble dans les différents résultats :

$$\texttt{co-assoc}(i,j) = \frac{1}{K}\sum_{k=0}^{K} V(\mathcal{C}^{(k)}, x_i, x_j) \tag{1.4}$$

où $V(.)$ renvoie 1 si les instances (x_i, x_j) ont été classées ensemble dans le résultat $\mathcal{C}^{(k)}$ et 0 sinon. Une méthode de clustering hiérarchique est ensuite utilisée en prenant la matrice de co-association comme une matrice de distance avec un seuil t qui détermine le similarité minimale à partir duquel deux clusters seront fusionnés. Ce type d'approche permet notamment de trouver des clusters ayant des formes non conventionnelles comme des spirales (voir la figure 1.1, page 9). Il est cependant nécessaire que les clusters soient clairement séparables dans l'espace des données sous peine de ne former qu'un cluster unique lors du clustering final. De plus, la construction et l'utilisation de la matrice de co-association implique une complexité quadratique en temps et en mémoire en fonction du nombre d'observations $O(n^2)$, ce qui rend l'utilisation de ces méthodes peu attractive pour traiter de grands volumes de données.

Une méthode pour introduire une pondération dans la matrice de co-association a également été proposée par Duarte et al. [2005]. Elle permet de pondérer la décision de chaque résultat de l'ensemble en fonction d'un critère de qualité interne affecté à chaque résultat. Cet indice est ensuite utilisé pour mettre à jour la matrice de co-association pondérée :

$$\texttt{w-co-assoc}(i,j) = \frac{1}{K}\sum_{k=0}^{K} V(\mathcal{C}^{(k)}, x_i, x_j) \times Q(\mathcal{C}^{(k)}) \tag{1.5}$$

où $V(.)$ renvoie 1 si les instances (x_i, x_j) ont été classées ensemble dans le résultat $\mathcal{C}^{(k)}$ et 0 sinon, et $Q(.)$ renvoie l'indice de qualité du résultat $\mathcal{C}^{(k)}$.

Méthodes de combinaison basées sur la définition d'un nouvel espace de données

Une autre proposition [Topchy et al., 2003] consiste à construire un nouvel espace de données à partir de l'ensemble des clusterings. Le problème du clustering par ensemble est modélisé sous la forme d'un problème de clustering à attributs catégoriels. Chaque résultat de l'ensemble donne lieu a un attribut catégoriel représentant l'objet. Ce nouvel espace de données est ensuite utilisé pour effectuer le clustering final. Une fonction d'utilité est introduite qui permet d'évaluer la qualité du consensus en fonction des différents attributs. He et al. [2005] ont défini formellement le lien entre le clustering par ensemble et le clustering de données catégorielles. Dans des travaux similaires, Hu et al. [2006] utilisent un champ de Markov aléatoire ainsi qu'une estimation du maximum de vraisemblance pour définir une métrique entre les résultats de clustering. Ils présentent deux méthodes basées sur cette nouvelle similarité permettant de calculer un consensus.

Méthodes basées sur le ré-étiquetage et le vote

Une autre approche dans le clustering par ensemble consiste à effectuer un vote au sein des différents résultats pour produire le consensus. Ainsi, Ayad and Kamel [2008] présentent un algorithme de vote cumulatif qui permet de trouver un consensus parmi plusieurs partitions à nombre différent de clusters. Ils décrivent plusieurs approches de vote pondéré qui permettent de calculer une densité de probabilité résumant les clustering initiaux. Une étude empirique montre que leur approche semble donner de meilleurs résultats que les approches par graphe. D'autres approches par vote sont présentées par Nguyen and Caruana [2007] : IVC (*Iterative Voting Consensus*), IPVC (*Iterative Probabilistic Voting Consensus*) et IPC (*Iterative Pairwise Consensus*). Ces algorithmes utilisent une carte de caractéristiques calculée à partir de l'ensemble de résultats de clustering et utilisent une approche de type *Expectation-Maximisation* pour calculer le consensus. Un problème important dans ces approches par vote est la nécessité de mettre en correspondance les différents clusters des clusterings initiaux. La plupart des méthodes [Ayad and Kamel, 2008; Zhou and Tang, 2006] sélectionnent un membre de l'ensemble servant de *base* pour le ré-étiquetage des autres résultats. Une fois cette base choisie, les clusters des autres résultats sont mis en correspondance en observant le recouvrement des clusters. Cependant, le choix de l'heuristique de sélection de la base a d'importantes conséquences sur les résultats obtenus et détermine notamment le choix du nombre de clusters. Pour résoudre ce problème, Long et al. [2005] proposent une méthode basée sur un mécanisme de correspondance floue entre les différents clusters des résultats. L'algorithme proposé produit un consensus mais également une matrice de correspondance qui fournit les liens entre les clusters des différents résultats.

Une autre méthode proposée par Zhou and Tang [2006] ne s'intéressent qu'aux partitions ayant le même nombre de clusters (voir l'exemple tableau 1.4). Une fois les clusters ré-étiquetés, un vote à la majorité est effectué pour construire le consensus. Des pondérations sont appliquées à chacune des décisions en fonction de son taux d'accord avec les autres méthodes. Dimitriadou et al. [2002] présentent un algorithme basé sur la minimisation de l'erreur au carré entre les résultats de l'ensemble. L'algorithme de vote trouve une solution approchée, la recherche de la meilleure solution du problème de minimisation étant impossible, car il est nécessaire d'énumérer toutes les permutations possibles. L'algorithme effectue une recherche séquentielle, où pour chaque partition de l'ensemble, la meilleure permutation d'étiquette est effectuée. Une méthode multivue est proposée par Wemmert and Gancarski [2002a] pour construire un consensus parmi les résultats de l'ensemble. Dans cette méthode, chaque objet va voter pour le cluster auquel il appartient et pour un cluster dans chacun des autres résultats. Les clusters ayant obtenu le maximum de votes seront retenus pour construire le consensus. Cette approche est présentée en détails dans la section 1.3.2.

Les auteurs montrent à travers une formalisation probabiliste l'intérêt d'utiliser une méthode de clustering par ensemble. Pour cela, ils produisent les résultats de l'ensemble en effectuant des permutations aléatoires d'étiquettes à partir de la partition réelle connue et montrent qu'il est possible de retrouver cette vraie partition. Une étude est également présentée sur la distance entre les différentes partitions dans l'espace des partitions. Ainsi, ils montrent qu'une méthode de vote tend à converger vers une solution moyenne dans cet espace.

Une autre approche proposée par Oliveira and Pedrycz [2007] consiste à utiliser une correspondance douce entre les clusters pour refléter le fait qu'un cluster d'un résultat peut être lié avec

	$\mathcal{C}^{(1)}$	$\mathcal{C}^{(2)}$	$\mathcal{C}^{(3)}$
x_1	1	3	1
x_2	1	3	1
x_3	2	1	1
x_4	2	1	3
x_5	3	2	3
x_6	3	2	2

Résultats initiaux.

	$\mathcal{C}^{(1)}$	$\mathcal{C}^{(2)}$	$\mathcal{C}^{(3)}$
x_1	1	1	1
x_2	1	1	1
x_3	2	2	1
x_4	2	2	2
x_5	3	3	2
x_6	3	3	3

Résultats ré-étiquetés.

Objet	x_1	x_2	x_3	x_4	x_5	x_6
Vote	(**3**,0,0)	(**3**,0,0)	(1,**2**,0)	(0,**3**,0)	(0,1,**2**)	(0,0,**3**)
Résultat	1	1	2	2	3	3

Résultat du vote.

TABLE 1.4 – Exemple de ré-étiquetage suivi d'un vote.

plusieurs clusters d'un autre résultat. Une nouvelle fonction de consensus appelée ITK (*Information Theoretic* KMEANS) est proposée. Celle-ci prend en entrée le degré d'appartenance de chaque objet aux clusters des résultats et non pas une appartenance dure comme dans les autres travaux sur l'ensemble clustering. Le fait de posséder cette information d'appartenance permet de définir une fonction de consensus plus précise qui prend en compte l'information floue proposée par les algorithmes de clustering. Il est cependant nécessaire d'utiliser des algorithmes de clustering proposant ce type de résultat en sortie. L'algorithme EM est par exemple bien adapté car il est possible d'obtenir un degré d'appartenance à chaque loi composant le résultat.

Bilan

Les méthodes par ensemble consistent à créer un résultat de clustering appelé consensus à partir d'un ensemble de résultats de clustering. Ces méthodes s'intéressent principalement à deux aspects. Le premier est la création des résultats de l'ensemble (différents algorithmes, différentes initialisations, etc.). La seconde est la définition d'une fonction permettant de trouver la partition consensuelle finale. Ces méthodes nécessitent généralement beaucoup de résultats en entrée pour être efficaces et il est souvent difficile de faire un choix sur la fonction de consensus à utiliser.

1.2.2 Approches multiobjectifs

Le clustering multiobjectif a pour but d'optimiser simultanément plusieurs critères de clustering. L'idée est de mieux saisir la notion de cluster en définissant explicitement différentes fonctions objectives. Les algorithmes sont ainsi capables de produire un ensemble de solutions qui sont des compromis des différents objectifs utilisés.

Ainsi, la méthode MOCK (*Multi-Objective Clustering with automatic K-determination*) [Handl and Knowles, 2007] utilise deux objectifs : le premier est de maximiser la compacité des clusters, et le second leur connectivité. Un algorithme évolutionnaire multiobjectif est utilisé pour optimiser ces deux critères simultanément. La méthode utilise un front de Pareto [Konak et al., 2006] qui consiste à sélectionner les solutions non dominées, c'est-à-dire celles respectant le mieux les deux objectifs de manière simultanée. À la fin de l'évolution, les solutions présentes sur le front de Pareto forment l'ensemble des solutions fournies par l'algorithme. Une heuristique est ensuite utilisée pour sélectionner la meilleure solution potentielle en utilisant le nombre de clusters des solutions présentes sur le front. Dans [Handl and Knowles, 2006], les auteurs présentent un moyen d'intégrer des connaissances du domaine (voir chapitre 4) à travers un troisième objectif basé sur un ensemble d'objets étiquetés. Cette version semi-supervisée donne de meilleurs résultats que la version sans connaissance.

Faceli et al. [2006] ont proposé une autre méthode appelée MOCLE (*Multi-Objective Clustering Ensemble*) qui intègre les deux mêmes fonctions objectives (maximisation de la compacité et la connectivité des clusters) que MOCK. Cependant, un opérateur de croisement spécial est ajouté qui

utilise des techniques de clustering par ensemble. Le but de la méthode MOCLE est de produire un ensemble de solutions représentant des compromis entre les deux objectifs alors que MOCK cherchait à trouver une unique solution. Une extension semi-supervisée de MOCLE a également été proposée par Faceli et al. [2007]. La connaissance à propos de la structure des données est également intégrée à l'aide d'un objectif additionnel. Enfin, une autre approche proposée par Law et al. [2004] utilise également plusieurs objectifs. Cependant, la solution finale est produite en sélectionnant les meilleurs clusters dans les résultats produits. Une méthode d'échantillonnage est utilisée pour estimer la qualité des clusters en se basant sur la stabilité de leur présence sur plusieurs exécutions.

1.2.3 Approches par combinaison de méthodes floues

Une architecture de clustering flou a été introduite par Pedrycz [2002] dans laquelle plusieurs sous-ensembles d'objets d'un jeu de données initial sont traités dans le but de trouver une structure commune. Les différents sous-ensembles sont tout d'abord traités indépendamment, puis, chaque partition est modifiée en accord avec les autres partitions : chaque résultat produit à partir d'un sous-ensemble est modifié par rapport aux informations obtenues sur les autres sous-ensembles. Des expériences ainsi que plus de détails sur la méthode sont fournis dans [Pedrycz and Rai, 2008]. Une application de cette approche pour l'analyse de contenu web est proposée par Loia et al. [2007]. Les auteurs présentent une méthode collaborative basée sur la proximité des objets et montrent comment cette information peut être utilisée pour découvrir la structure d'informations sur le web dans des espaces sémantiques différents.

Une autre plateforme de clustering collaboratif flou est proposée par Mitra et al. [2006] où les ensembles bruts (*rough sets*) sont utilisés pour créer un paradigme collaboratif. Un algorithme de clustering est développé en intégrant les avantages des ensembles flous et des ensembles bruts. Une analyse quantitative et des résultats expérimentaux sont aussi présentés sur des données artificielles et réelles. Ces approches collaboratives floues apportent des fondements théoriques intéressants mais sont limitées sur de nombreux aspects : chaque partition doit avoir le même nombre de clusters ; la question difficile de la correspondance entre les clusters est supposée résolue ; la distance entre chaque point et le centre des clusters dans chaque solution doit être connue. Malgré ces contraintes, il a été montré, au moins sur des exemples simples à deux ou trois clusters que la collaboration a un effet positif sur la qualité des clusters.

Des travaux récents [Grozavu and Bennani, 2010] utilisent l'algorithme SOM pour effectuer cette tâche de clustering distribué de plusieurs sous-ensembles d'objets. La méthode est décomposée en deux phases. La première consiste à appliquer l'algorithme SOM sur les différentes données indépendamment. La seconde phase consiste à faire collaborer ces différentes cartes pour les enrichir. Des travaux similaires sont proposés par [Cleuziou et al., 2009] avec une approche permettant de traiter des données multi-représentées, c'est-à-dire un même ensemble d'individus décrits par plusieurs représentations.

1.2.4 Bilan

Dans cette section nous avons présenté les principales approches de la littérature concernant l'utilisation de plusieurs algorithmes ou résultats de clustering. Les méthodes de clustering par ensemble (section 1.2.1) ne s'intéressent généralement pas à la génération des résultats initiaux et ne s'intéressent qu'à la création du résultat consensuel final. Ainsi les algorithmes utilisés pour créer les résultats initiaux ne sont pas ou peu étudiés. Ces méthodes introduisent un nouveau biais, relatif à la fonction objective choisie pour fusionner les différents résultats. De plus, il est souvent nécessaire de générer de nombreuses partitions (de quelques dizaines à plusieurs centaines) pour obtenir des résultats pertinents. Ces problèmes sont partagés avec les méthodes multiobjectifs (section 1.2.2) qui utilisent des objectifs qui ne sont pas forcément ceux utilisés pour la création des solutions initiales. De plus, les méthodes multiobjectifs génèrent de très nombreuses solutions pour s'assurer de trouver des résultats pertinents. Enfin, les méthodes multiobjectifs proposent en général en sortie un ensemble de résultats hétérogènes et le choix de la solution finale est laissé à l'expert. Les méthodes par combinaison de méthodes floues (section 1.2.3) proposent des fondements théoriques solides sur la collaboration entre méthodes, mais ces méthodes ne proposent pas de solution à de nombreux problèmes (nombre de clusters différents, correspondance entre les

clusters, passage à l'échelle, etc.). Dans la section suivante nous allons aborder une autre approche du clustering collaboratif, plus générique, et indépendante de la méthode de clustering utilisée.

1.3 Le clustering collaboratif

1.3.1 Problématique

Dans le cadre du travail collectif, trois approches existent et sont utilisées notamment en management et en science de l'éducation. La première, la *coordination* est une activité collective qui consiste à construire un tout à partir de parties plus petites fournies par différents individus. La seconde, la *coopération* est une action conjointe où les acteurs se mettent d'accord sur un ensemble défini de règles à suivre qui permettent d'agréger le travail de chaque individu de manière efficace. Et enfin, la *collaboration*, qui est un processus de création collective et de partage de connaissances entre plusieurs individus. L'utilisation du terme *collaboratif* pour qualifier les méthodes présentées dans cette section, se justifie par la volonté de créer un système dans lequel les individus partagent activement leurs informations locales dans le but d'une recherche de solution globale. L'objectif est de trouver une solution que les méthodes n'aurait pas pu, ou auraient éprouvé des difficultés à trouver seules.

Dans cette section nous allons étudier la méthode de clustering collaboratif proposée par Wemmert [2000] appelée SAMARAH. Cette méthode consiste en la collaboration de différentes méthodes de clustering pour tenter de trouver un consensus sur le clustering d'un jeu de données. L'objectif de cette collaboration est de réduire l'impact du choix d'une méthode et de ses paramètres sur le résultat. Étant donné un ensemble de N résultats de clustering, l'idée est de modifier de façon itérative et collaborative les résultats initiaux afin d'en améliorer la similarité et la qualité, autorisant ainsi un consensus final plus pertinent et de meilleure qualité.

1.3.2 Présentation de l'approche samarah

La méthode SAMARAH est basée sur le principe d'un raffinement mutuel et itératif de plusieurs résultats de clustering. Le système peut être décomposé en trois grandes étapes :

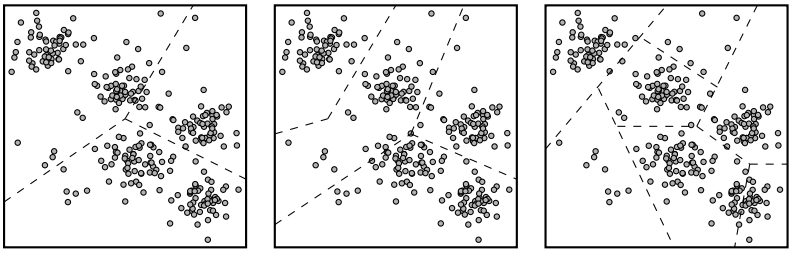
1. La génération des résultats initiaux ;
2. Le raffinement des différents résultats ;
3. La combinaison des résultats raffinés.

La première étape de la méthode consiste à générer les résultats initiaux qui seront ensuite utilisés par celle-ci. Dans cette étape, des algorithmes de clustering sont appliqués sur les données. Différents algorithmes ou paramétrages du même algorithme peuvent être utilisés. La figure 1.2 donne un exemple de trois clusterings différents d'un même jeu de données.

Lors de la deuxième étape, chacun des résultats va être comparé avec l'ensemble des résultats proposés par les autres méthodes. Le but est d'évaluer la similarité entre les différents résultats pour observer les différences dans les regroupements des objets. Une fois ces différences (appelées par la suite *conflits*) identifiées, l'objectif est de modifier les résultats pour tenter de réduire ces différences, c'est-à-dire résoudre les conflits. En fonction des informations obtenues sur les différences entre les résultats, des modifications (fusion de clusters, scission de clusters, reclustering de clusters) sont appliquées de manière itérative aux résultats. Cette étape peut être vue comme une remise en cause en fonction des informations fournies par les autres acteurs de la collaboration. Après plusieurs itérations de raffinement, il est attendu des résultats qu'ils soient plus similaires qu'avant cette étape collaborative.

Lors de la troisième étape, les différents résultats raffinés sont combinés si nécessaire pour proposer un résultat unique. Ce calcul d'un résultat consensuel est simplifié par la forte similarité des résultats.

La figure 1.3 présente un schéma des différentes étapes du clustering collaboratif. Dans les sections suivantes nous allons étudier en détail les différentes étapes de la méthode.



(a) 3 clusters (b) 4 clusters (c) 7 clusters

FIGURE 1.2 – Exemple de trois clusterings différents du même jeu de données.

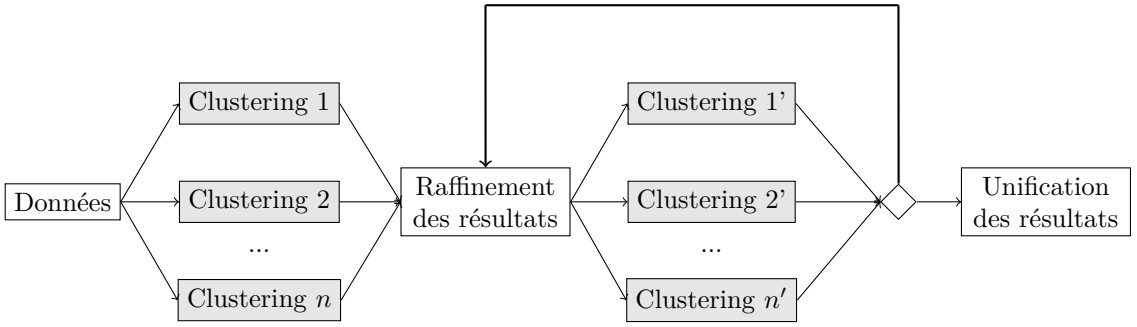


FIGURE 1.3 – Schéma illustrant les différentes étapes du clustering collaboratif.

Génération des résultats initiaux

La génération des résultats initiaux consiste à appliquer plusieurs méthodes de clustering aux données. Il est nécessaire lors de cette étape de faire un choix sur le nombre de méthodes impliquées dans la collaboration ainsi que sur leur type. Nous avons vu dans le chapitre 2 qu'il existe un grand nombre de méthodes de clustering. La plupart des méthodes de clustering les plus courantes peuvent être utilisées dans la méthode SAMARAH. Nous verrons toutefois qu'il est nécessaire de modifier légèrement les méthodes pour pouvoir les utiliser de manière efficace dans le système. On obtient donc à l'issue de cette étape, un ensemble de clusterings différents $\mathbb{C} = \{\mathcal{C}^{(1)}, \ldots, \mathcal{C}^{(N)}\}$, chaque clustering $\mathcal{C}^{(i)}$ ayant été généré à partir d'une méthode et/ou d'un paramétrage spécifique.

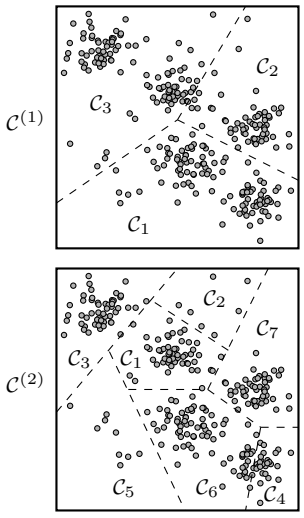
Raffinement des résultats

Le raffinement des résultats consiste à comparer les différents résultats et à observer la répartition des objets dans les différents clusters des différents résultats. L'objectif est d'identifier sur quelle partie du regroupement des données les méthodes sont en désaccord (et respectivement en accord). Pour se faire il est nécessaire de comparer les résultats entre eux. Pour pouvoir observer les similitudes et les différences de chaque résultat par rapport à tous les autres résultats, ceux-ci sont comparés deux à deux. Pour effectuer cette comparaison, la matrice de confusion (Ω) est calculée pour chaque couple de clusterings $(\mathcal{C}^{(i)}, \mathcal{C}^{(j)})$:

$$\Omega^{(i,j)} = \begin{pmatrix} \alpha_{1,1}^{(i,j)} & \cdots & \alpha_{1,K^{(j)}}^{(i,j)} \\ \vdots & \ddots & \vdots \\ \alpha_{K^{(i)},1}^{(i,j)} & \cdots & \alpha_{K^{(i)},K^{(j)}}^{(i,j)} \end{pmatrix} \text{ où } \alpha_{k,l}^{(i,j)} = \frac{\left| C_k^{(i)} \bigcap C_l^{(j)} \right|}{\left| C_k^{(i)} \right|} \tag{1.6}$$

La matrice de confusion $\Omega^{(i,j)}$ contient la répartition des objets dans les clusters de deux résultats de clustering. Cette matrice permet d'observer si les objets d'un cluster d'un résultat ont été regroupés de manière similaire dans l'autre résultat ou au contraire ont été répartis dans plusieurs clusters et dans quelles proportions.

Cette matrice de confusion est le cœur de la comparaison des résultats en clustering collaboratif. À partir de cette information, il est possible d'évaluer la similarité des résultats, et plus



	$C_1^{(1)}$	$C_2^{(1)}$	$C_3^{(1)}$
$C_1^{(2)}$		0.11	0.42
$C_2^{(2)}$		0.05	0.03
$C_3^{(2)}$			0.50
$C_4^{(2)}$	0,45		
$C_5^{(2)}$	0,05		0,05
$C_6^{(2)}$	0,50	0,02	
$C_7^{(2)}$		0,82	

$\Omega^{(1,2)}$

	$C_1^{(2)}$	$C_2^{(2)}$	$C_3^{(2)}$	$C_4^{(2)}$	$C_5^{(2)}$	$C_6^{(2)}$	$C_7^{(2)}$
$C_1^{(1)}$				1	0,55	0,98	0,02
$C_2^{(1)}$	0,13	0,43				0,02	0,98
$C_3^{(1)}$	0,87	0,57	1		0,45		

$\Omega^{(2,1)}$

FIGURE 1.4 – Exemple de calcul de matrice de confusion à partir de deux résultats de clustering.

encore, d'identifier avec précision leurs désaccords. Un autre intérêt de cette matrice est qu'elle est totalement indépendante de la méthode utilisée pour créer les clusters, car celle-ci ne nécessite que l'information de l'affectation des objets aux clusters pour être calculée. La figure 1.4 présente un exemple de calcul de matrice de confusion pour deux résultats.

Nous allons voir dans la suite de cette section un ensemble de critères statistiques utilisant cette matrice de confusion et qui permettent d'obtenir des informations sur la similarité de deux résultats de clustering. Le premier critère est un critère de similarité inter-cluster qui permet la comparaison de deux clusters de deux résultats différents. Pour se faire deux matrices de confusion sont calculées. En effet, comme les résultats n'ont, d'une part, pas forcément le même nombre de clusters, et que d'autre part, les coefficients de la matrice sont normalisés par la taille des clusters d'un des deux résultats, on obtient $\Omega^{i,j} \neq \Omega^{j,i}$. Il est donc nécessaire de calculer cette matrice dans les deux *sens* ($\Omega^{i,j}$ et $\Omega^{j,i}$) pour pouvoir comparer les deux résultats. Pour calculer la similarité S entre les deux clusters, on peut ensuite utiliser ces deux matrices, en observant les deux coefficients correspondants aux deux clusters à comparer :

$$S\left(C_k^{(i)}, C_l^{(j)}\right) = \alpha_{k,l}^{(i,j)} \alpha_{l,k}^{(j,i)} \tag{1.7}$$

Cependant, ce critère ne prend en compte que l'intersection entre les deux clusters considérés sans prendre en compte la répartition des objets dans les autres clusters. Si par exemple les objets d'un cluster d'un résultat se retrouve à 50% dans le cluster d'un autre résultat, la répartition des 50% d'objets restants n'est pas prise en compte. Que ces objets soient répartis dans un autre cluster ou dans plusieurs autres clusters, la valeur du coefficient est identique. Pour prendre en compte cette distribution, la similarité entre deux clusters est définie telle que :

$$S\left(C_k^{(i)}, C_l^{(j)}\right) = \rho_k^{(i,j)} \alpha_{l,k}^{(j,i)} \tag{1.8}$$

où

$$\rho_k^{(i,j)} = \sum_{r=1}^{K^{(j)}} (\alpha_{k,r}^{(i,j)})^2 \tag{1.9}$$

Une fois cette similarité définie entre les clusters des différents résultats, il est possible de définir une fonction qui va mettre en correspondance un cluster $C_k^{(i)}$ d'un résultat $\mathcal{C}^{(i)}$ et son cluster le plus similaire dans un autre résultat $\mathcal{C}^{(j)}$:

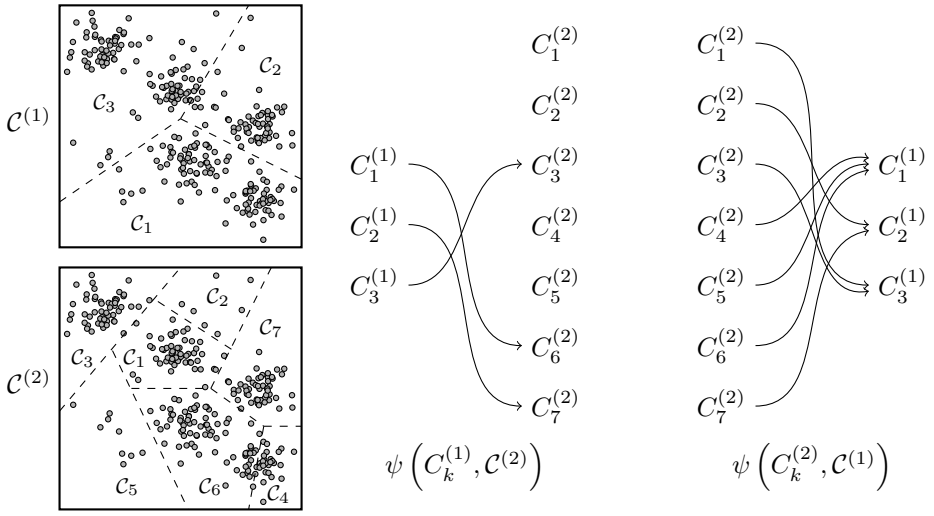
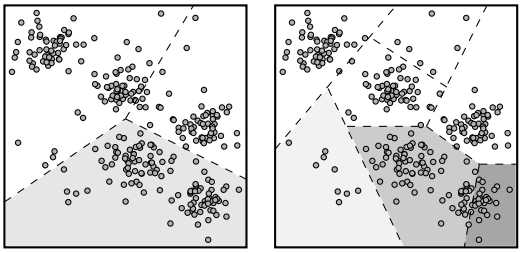


FIGURE 1.5 – Exemple de fonction de correspondance entre les clusters de deux résultats.



(a) $\mathcal{C}^{(1)}$, 3 clusters (b) $\mathcal{C}^{(2)}$, 7 clusters

Importance du conflit : $CI\left(\mathcal{K}_2^{(1,2)}\right) = 0.573$

FIGURE 1.6 – Exemple de conflit entre deux résultats de clustering.

$$\psi\left(C_k^{(i)}, \mathcal{C}^{(j)}\right) = \arg \max_{C_l^{(j)} \in \mathcal{C}^{(j)}} S\left(C_k^{(i)}, C_l^{(j)}\right) \tag{1.10}$$

La figure 1.5 montre un exemple du calcul de cette fonction de correspondance entre deux résultats. Grâce à cette fonction de correspondance, il est possible d'identifier si les différents résultats sont en accord sur le regroupement des données. Nous définissons le concept de *conflit* entre un cluster d'un résultat et un autre résultat, si ce cluster ne se retrouve pas de manière parfaite dans l'autre résultat.

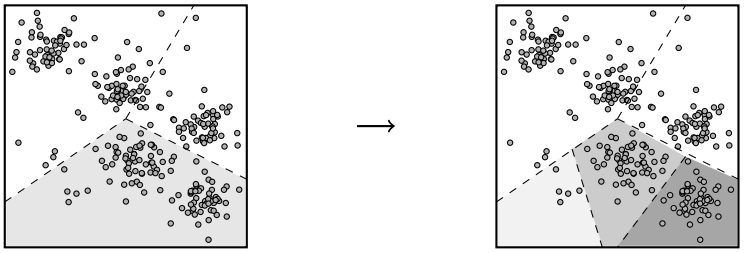
L'étape de détection des conflits consiste à chercher dans $\mathbb{C}$ tous les couples $\left(C_k^{(i)}, \mathcal{C}^{(j)}\right)$, $i \neq j$, tel que $S\left(\mathcal{C}_k^{(i)}, \psi\left(C_k^{(i)}, \mathcal{C}^{(j)}\right)\right) < 1$, ce qui signifie que le cluster $C_k^{(i)}$ ne peut pas être trouvé exactement dans le résultat $\mathcal{C}^{(j)}$. La liste des conflits de $\mathbb{C}$ peut se définir comme :

$$\texttt{conflits}(\mathbb{C}) = \left\{(C_k^{(i)}, \mathcal{C}^{(j)}) : i \neq j, S\left(C_k^{(i)}, \psi\left(C_k^{(i)}, \mathcal{C}^{(j)}\right)\right) < 1\right\} \tag{1.11}$$

Chaque conflit $\mathcal{K}_k^{(i,j)}$ est lié a un cluster $C_k^{(i)}$ et un résultat $\mathcal{C}^{(j)}$. Son importance CI, est évaluée grâce à la similarité entre le cluster du premier résultat et les clusters du deuxième résultat (équation (1.8)).

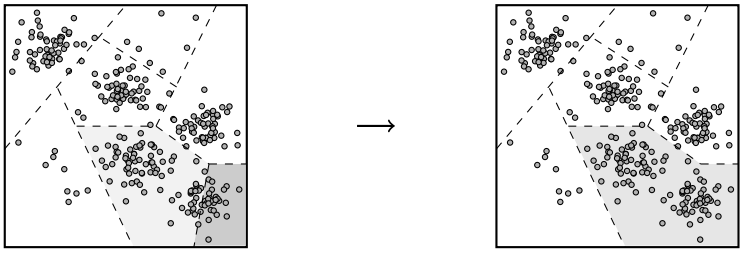
$$CI\left(\mathcal{K}_k^{(i,j)}\right) = 1 - S\left(C_k^{(i)}, \psi\left(C_k^{(i)}, \mathcal{C}^{(j)}\right)\right) \tag{1.12}$$

La figure 1.6 donne un exemple de conflit entre deux résultats. Plus un cluster est dispersé dans plusieurs clusters d'un autre résultat, plus il est en conflit avec ce résultat et plus l'importance



$\mathcal{C}^{(1)}$ avant l'application de l'opérateur. $\mathcal{C}^{(1)}$ après l'application de l'opérateur.

FIGURE 1.7 – Exemple d'application de l'opérateur de scission.



$\mathcal{C}^{(1)}$ avant l'application de l'opérateur. $\mathcal{C}^{(1)}$ après l'application de l'opérateur.

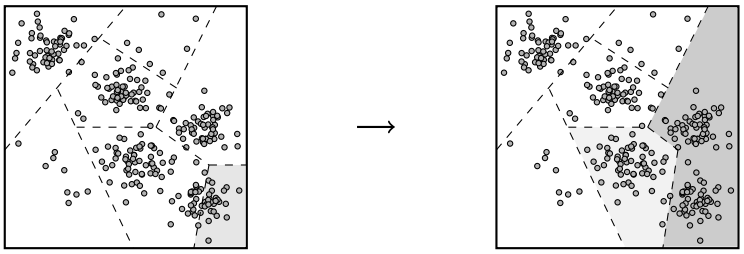
FIGURE 1.8 – Exemple d'application de l'opérateur de fusion.

du conflit est élevée. La suite du processus collaboratif consiste à essayer de résoudre ces conflits. Dans sa version initiale, la méthode SAMARAH cherche toujours à résoudre le conflit le plus important. C'est cette approche que nous allons étudier dans un premier temps. Plus loin dans ce chapitre (section 1.4) nous présentons différentes stratégies pour résoudre ces conflits, en étudiant notamment leur ordre de résolution.

Le conflit le plus important est sélectionné parmi tous les conflits détectés entre tous les couples de résultats. La résolution d'un conflit $\mathcal{K}_k^{(i,j)}$ consiste à appliquer un opérateur sur chacun des résultats impliqués dans le conflit : $\mathcal{C}^{(i)}$ et $\mathcal{C}^{(j)}$. Les opérateurs qui peuvent être appliqués aux résultats sont :

- la **fusion** de clusters : plusieurs clusters sont fusionnés ensemble. La figure 1.7 donne un exemple de cet opérateur ;
- la **scission** de cluster en sous-clusters : un clustering est appliqué aux objets d'un cluster pour créer des sous-clusters. La figure 1.8 donne un exemple de cet opérateur ;
- la **reclustering** d'un cluster : un cluster est retiré et ses objets sont distribués dans les autres clusters restants. La figure 1.9 donne un exemple de cet opérateur.

L'opérateur à appliquer est choisi grâce au nombre de clusters impliqués dans le conflit, c'est-à-dire tel que $S(C_k^{(i)}, C_l^{(j)}) > p_{cr}$, où $0 \leq p_{cr} \leq 1$ est un paramètre choisi. Par exemple si $p_{cr} = 0.2$ cela signifie que si $C_k^{(i)} \cap C_l^{(j)}$ représente moins de 20% des objets de $C_k^{(i)}$, $C_l^{(j)}$ n'est pas considéré comme un représentant de $C_k^{(i)}$.



$\mathcal{C}^{(1)}$ avant l'application de l'opérateur. $\mathcal{C}^{(1)}$ après l'application de l'opérateur.

FIGURE 1.9 – Exemple d'application de l'opérateur de reclustering.

Les détails du processus de sélection de l'opérateur à appliquer sont présentés dans l'algorithme 1. Un opérateur est appliqué à chacun des deux résultats impliqués dans le conflit. Cependant, l'application de ces deux opérateurs (un sur chaque résultat) n'est pas toujours pertinente. En effet, cela n'entraine pas toujours une augmentation de la similarité des résultats impliqués dans le conflit. De plus, une itération de cette résolution de conflit peut mener à des solutions triviales et non voulues. Par exemple, tous les résultats ayant un unique cluster, ou un cluster pour chaque objet. Ces solutions non pertinentes doivent être évitées.

Algorithm 1: Résolution d'un conflit

Entrées: $\mathbb{C}$ les différents résultats de clustering, $\mathcal{K}_k^{i,j}$ le conflit à résoudre
Sorties: $\mathbb{C}^\star = \texttt{conflictResolution}\left(\mathbb{C}, \mathcal{K}_k^{i,j}\right)$ le nouvel ensemble après la résolution
soit $\kappa = \{C_l^{(j)}, \forall 1 \leq l \leq K^{(j)} : S(C_k^{(i)}, C_l^{(j)}) > p_{cr}\}$
si $|\kappa| > 1$ **alors**
  $\mathcal{C}^{(i')} = \mathcal{C}^{(i)} \setminus \{C_k^{(i)}\} \cup \texttt{scission}(C_k^{(i)}, |\kappa|)$
  $\mathcal{C}^{(j')} = \mathcal{C}^{(j)} \setminus \kappa \cup \texttt{fusion}(\kappa, \mathcal{C}^{(j)})$
sinon
  $\mathcal{C}^{(i')} = \texttt{reclustering}(\mathcal{C}^{(i)} \setminus \{C_k^{(i)}\})$
$\{\mathcal{C}^{(i\star)}, \mathcal{C}^{(j\star)}\} = \arg\max \gamma^{(I,J)}$ avec $I \in \{i, i'\}, J \in \{j, j'\}$
$\mathbb{C}^\star = \mathbb{C} \setminus \{\mathcal{C}^{(i)}, \mathcal{C}^{(j)}\} \cup \{\mathcal{C}^{(i\star)}, \mathcal{C}^{(j\star)}\}$

Pour cela, la méthode SAMARAH utilise un *coefficient local d'évaluation*, qui permet de contrôler la convergence de la collaboration entre deux clusterings. Ce coefficient est basé sur la similarité entre tous les clusters de deux résultats, et prend en compte un coefficient de qualité des clusters eux-mêmes (δ), sélectionné par l'expert. Ce coefficient de qualité évalue la qualité du clustering pour éviter d'obtenir une solution triviale :

$$\gamma(\mathcal{C}^{(i)}, \mathcal{C}^{(j)}) = \frac{1}{2}\left(p_s.\left(\frac{1}{K^{(i)}}\sum_{k=1}^{K^{(i)}}\omega_k^{(i,j)} + \frac{1}{K^{(j)}}\sum_{k=1}^{K^{(j)}}\omega_k^{(j,i)}\right) + p_q.\left(\delta^{(i)} + \delta^{(j)}\right)\right) \tag{1.13}$$

où

$$\omega_k^{(i,j)} = S\left(C_k^{(i)}, \psi\left(C_k^{(i)}, \mathcal{C}^{(j)}\right)\right) \tag{1.14}$$

et, p_q et p_s sont des paramètres du système ($p_q + p_s = 1$). Nous verrons en détail la définition de l'indice de qualité δ qui nous permettra notamment d'intégrer des connaissances dans la méthode dans le chapitre suivant (Chapitre 4).

Soit $\mathcal{C}^{(i')}$ (respectivement $\mathcal{C}^{(j')}$) le résultat de l'application d'un opérateur sur $\mathcal{C}^{(i)}$ (respectivement $\mathcal{C}^{(j)}$). Le coefficient local d'évaluation (équation (1.13)) est calculé pour chacun des couples : $(\mathcal{C}^{(i)}, \mathcal{C}^{(j)})$, $(\mathcal{C}^{(i')}, \mathcal{C}^{(j')})$, $(\mathcal{C}^{(i')}, \mathcal{C}^{(j)})$, $(\mathcal{C}^{(i)}, \mathcal{C}^{(j')})$. Le couple ayant la meilleure évaluation est conservée comme solution à la résolution du conflit.

Après cette étape de résolution locale d'un conflit (locale car effectuée pour un couple de résultats alors que plus de deux clusterings peuvent être impliqués dans la collaboration), une étape d'évaluation globale est engagée. Cette étape globale d'évaluation a pour but de valider ou non les décisions prises au niveau local. Le *coefficient global d'évaluation* est calculé en fonction de tous les coefficients locaux d'évaluation pour chaque couple de résultats :

$$\mathbf{\Gamma}(\mathbb{C}) = \frac{1}{N}\sum_{i=1}^{N}\Gamma(\mathcal{C}^{(i)}) \tag{1.15}$$

où

$$\Gamma(\mathcal{C}^{(i)}) = \frac{1}{N-1}\sum_{\substack{j=1 \\ j\neq i}}^{N}\gamma(\mathcal{C}^{(i)}, \mathcal{C}^{(j)}) \tag{1.16}$$

Trois cas peuvent survenir concernant l'évolution de ce critère pendant la phase de collaboration :

1. Cette étape de résolution de conflit permet d'obtenir une meilleure solution au niveau global. Dans ce cas, la solution courante est remplacée par cette nouvelle solution. La liste des conflits est recalculée, et une nouvelle itération est engagée ;
2. L'étape de résolution propose la même solution qu'avant l'application des opérateurs, ce qui signifie que la résolution de ce conflit n'est pas pertinente. Ce conflit est retiré de la liste, et une nouvelle itération est engagée ;
3. Si la solution proposée par la résolution du conflit donne un résultat globalement moins pertinent, ce résultat est tout de même considéré pour éviter de tomber dans un extremum local. Cependant, si dans la suite des itérations, aucune résolution de conflit ne permet d'obtenir une meilleure solution (après avoir épuisé la moitié des conflits à résoudre), tous les résultats sont réinitialisés à la meilleure solution courante et le conflit est retiré.

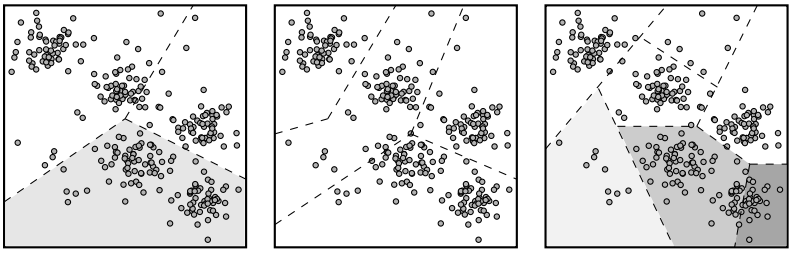
Ce processus itère jusqu'à épuisement de la liste des conflits, c'est-à-dire que tous les conflits ont été résolus, ou que la résolution des conflits restants ne permet pas d'améliorer les résultats. L'algorithme 2 présente ce processus en détail. La figure 1.10 présente un ensemble de résultats avant et après une itération où un conflit est résolu.

Algorithm 2: Clustering collaboratif

soit $\breve{\mathbb{C}} = \{\mathbb{C}^i\}_{1 \leq i \leq m}$ l'ensemble initial de résultats de clustering
soit $\breve{K} = \texttt{conflicts}(\breve{\mathbb{C}})$ l'ensemble des conflits sur $\breve{\mathbb{C}}$ (équation (1.11))
soit $\breve{\mathbb{C}}^{\texttt{best}} = \breve{\mathbb{C}}$ la meilleure solution temporaire
soit $\breve{K}^{\texttt{best}} = \breve{K}$ les conflits de la meilleure solution temporaire
tant que $|\breve{K}| \geq 0$ **faire**
  $\mathcal{K}_k^{i,j} = \arg\max_{\mathcal{K}_l^{r,s} \in \breve{K}} CI\left(\mathcal{K}_l^{r,s}\right)$
  $\breve{\mathbb{C}} = \texttt{conflictResolution}\left(\breve{\mathbb{C}}, \mathcal{K}_k^{i,j}\right)$ (Algorithme 1)
  si $\mathbf{\Gamma}(\breve{\mathbb{C}}) > \mathbf{\Gamma}(\breve{\mathbb{C}}^{best})$ **alors**
    $\breve{\mathbb{C}}^{\texttt{best}} = \breve{\mathbb{C}}$
    $\breve{K}^{\texttt{best}} = \breve{K} = \texttt{conflicts}(\breve{\mathbb{C}})$
    $bt = 0$
  sinon si $\breve{\mathbb{C}}^{t+1} = \breve{\mathbb{C}}^t$ **alors**
    $\breve{K} = \breve{K} \setminus \mathcal{K}_k^{i,j}$
  sinon
    $bt := bt + 1$
    $\breve{K} = \breve{K} \setminus \mathcal{K}_k^{i,j}$
    si $bt > |\breve{K}|$ **alors**
      $\breve{\mathbb{C}} = \breve{\mathbb{C}}^{\texttt{best}}$
      $\breve{K} = \breve{K}^{\texttt{best}} \setminus \mathcal{K}_k^{i,j}$
calcul du consensus

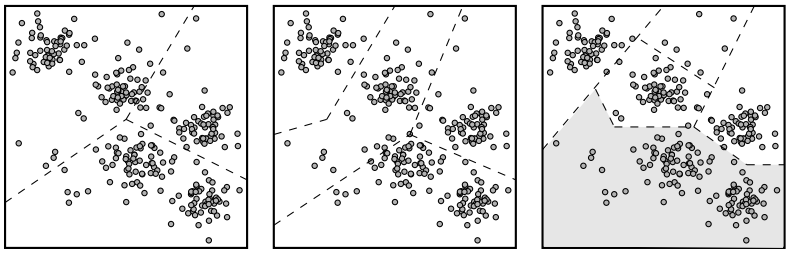
Combinaison des résultats raffinés

Après l'étape de raffinement, tous les résultats sont relativement similaires, avec un nombre de clusters proche. Chaque résultat peut être individuellement intéressant pour l'expert, comme chaque résultat a été créé d'une part grâce à son algorithme propre, et d'autre part par les informations obtenues des autres méthodes pendant la collaboration. Cependant, si l'expert est intéressé par un résultat unique, représentant le mieux possible l'ensemble des résultats, il est possible d'appliquer des algorithmes d'ensemble clustering vue en section 1.2.1. Nous présentons ici l'une de



(a) $\mathcal{C}^{(1)}$ (b) $\mathcal{C}^{(2)}$ (c) $\mathcal{C}^{(3)}$

Coefficient global d'évaluation avant résolution d'un conflit : $\mathbf{\Gamma} = 0.285$



(d) $\mathcal{C}^{(1)}$ (e) $\mathcal{C}^{(2)}$ (f) $\mathcal{C}^{(3')}$

Coefficient global d'évaluation après résolution d'un conflit : $\mathbf{\Gamma} = 0.318$

FIGURE 1.10 – Trois résultats avant (première ligne) et après (seconde ligne) la résolution d'un conflit entre $\mathcal{C}^{(1)}$ et $\mathcal{C}^{(3)}$.

ces méthodes développée par Wemmert and Gancarski [2002b] et qui utilise les concepts et les méthodes mis en place en clustering collaboratif.

Dans cet algorithme de vote, chaque résultat $\mathcal{C}^{(i)}$ effectue un vote pour chaque objet x. Il va tout d'abord voter pour le cluster $C_k^{(i)}$ auquel est associé x dans son résultat, ainsi que pour tous les clusters correspondants $\psi(C_k^{(i)}, \mathcal{C}^{(j)})$ dans les autres résultats de clustering impliqués dans la collaboration. Une fois tous les votes effectués pour chaque résultat et chaque objet, la valeur maximale indique le meilleur cluster pour l'objet x, par exemple $C_l^{(j)}$. Cela signifie que l'objet x doit être dans le cluster $C_l^{(j)}$ d'après l'opinion majoritaire des résultats.

Pour chaque objet x, un ensemble de vecteurs de vote est calculé :

$$\mathcal{V}(x) = \left\{ (v_1^{(i)}(x), \ldots, v_{K^{(i)}}^{(i)}(x)), 1 \leq i \leq N \right\} \tag{1.17}$$

où

$$v_k^{(i)}(x) = \sum_{j=1}^{N} vote(x, C_k^{(i)}, \mathcal{C}^{(j)}) \tag{1.18}$$

et

$$vote(x, C_k^{(i)}, \mathcal{C}^{(m)}) = \begin{cases} 1 & \text{si } (i = m \text{ et } x \in \mathcal{C}_k^i) \\ & \text{ou } x \in \psi(C_k^{(i)}, \mathcal{C}^{(m)}) \\ 0 & \text{sinon} \end{cases} \tag{1.19}$$

L'objet x est affecté au cluster $\breve{\mathcal{V}}$, défini comme :

$$\breve{\mathcal{V}}(x) = \arg\max_{C_k^{(i)}} v_k^{(i)}(x) \tag{1.20}$$

Le tableau 1.5 présente un exemple de l'application de cet algorithme de vote. Une fois les différents vecteurs de $\mathcal{V}$ calculés, le maximum est cherché pour assigner un cluster d'un résultat à chaque objet (voir équation (1.20)). En cas d'égalité, le premier cluster trouvé pour un objet est conservé. Ces égalités rendent possible le fait que plusieurs clusters soient créés dans le résultat final mais qu'ils représentent en fait des clusters très similaires dans deux résultats différents. Il est possible d'utiliser des heuristiques pour réduire le nombre de clusters dans le résultat final,

	$\mathcal{C}^{(1)}$	$\mathcal{C}^{(2)}$	$\mathcal{C}^{(3)}$
x_1	1	3	1
x_2	1	3	1
x_3	2	1	1
x_4	2	1	1
x_5	3	2	2
x_6	3	2	2

Résultats initiaux.

	$\mathcal{C}^{(1)}$	$\mathcal{C}^{(2)}$	$\mathcal{C}^{(3)}$
$\mathcal{V}(x_1)$	$\{(3,1,0),$	$(1,0,3),$	$(1,2,0)\}$
$\mathcal{V}(x_2)$	$\{(3,1,0),$	$(1,0,3),$	$(1,2,0)\}$
$\mathcal{V}(x_3)$	$\{(1,3,0),$	$(3,0,1),$	$(1,2,0)\}$
$\mathcal{V}(x_4)$	$\{(1,3,0),$	$(3,0,1),$	$(1,2,0)\}$
$\mathcal{V}(x_5)$	$\{(0,0,3),$	$(0,3,0),$	$(0,0,3)\}$
$\mathcal{V}(x_6)$	$\{(0,0,3),$	$(0,3,0),$	$(0,0,3)\}$

Votes pour les différents objets.

Objet	x_1	x_2	x_3	x_4	x_5	x_6
Résultat	1	1	2	2	3	3

Résultat du vote.

TABLE 1.5 – Exemple de l'application de l'algorithme de vote à la fin du processus collaboratif.

comme par exemple établir une correspondance entre les clusters très similaires et effectuer un ré-étiquetage en post-traitement.

1.4 Résolution de conflits en clustering collaboratif

Dans cette section, nous allons présenter nos propositions concernant la résolution de conflits en clustering collaboratif. De nouvelles stratégies de résolution de conflits développées lors de cette thèse vont être présentées.

1.4.1 Problématique

Comme nous l'avons vu dans la section précédente, le clustering collaboratif consiste à résoudre des conflits entre différents résultats de clustering en vue d'améliorer leur qualité et leur similarité. Pour résoudre ces conflits, la méthode SAMARAH propose de tenter de résoudre à chaque étape le conflit le plus important (voir section 1.3.2). Cette stratégie, bien que pouvant apparaître comme la plus efficace, peut mener à des solutions non optimales. En effet, en résolvant à chaque itération le conflit le plus important, il est possible que le système converge très rapidement vers un extremum local. La stratégie proposée, qui consiste à autoriser une perte de qualité pendant un certain nombre d'itérations permet de réduire ce risque, mais n'assure toujours pas un parcours optimal de l'espace de recherche. De plus, il peut être difficile pour l'expert de choisir le nombre d'itérations pendant lesquelles la qualité peut diminuer. Rappelons que la détection de conflits s'effectue au niveau local, c'est-à-dire entre couples de résultats de clustering. L'information locale (la répartition des objets dans les clusters d'un couple de résultats) permet de choisir le couple de résultats qui est le plus en désaccord, et par conséquent celui qui tirera le mieux parti de la résolution d'un conflit. Cependant, la validation au niveau global est sujette à une amélioration globale de la solution, c'est-à-dire une amélioration de l'ensemble des couples, ou au moins d'une partie importante. Il est donc difficile de s'assurer que les modifications faites au niveau local vont améliorer la solution globalement.

Il pourrait être envisagé de calculer l'évaluation globale lors de la résolution de chaque conflit. Ceci permettrait de s'assurer que la résolution du conflit considéré au niveau local permet la meilleure amélioration au niveau globale. Cependant, cette évaluation globale nécessite un nombre de calculs important car le coefficient local d'évaluation (voir équation (1.13)) est calculé pour chaque couple de résultats et est ensuite moyenné. On obtient donc $4(N(N-1))$ (N étant le nombre de résultats) calculs à effectuer pour choisir parmi les quatre couples résultant de la tentative de résolution d'un conflit. Il ne parait donc pas raisonnable de calculer ce coefficient pour chaque possibilité de résolution de conflit, surtout si le nombre de résultats impliqués dans le processus collaboratif est important.

En étudiant plus en détails l'espace des solutions possibles en clustering collaboratif, on se rend rapidement compte de l'importance du nombre potentiel de solutions. Soit $\breve{\mathbb{C}}$ l'ensemble de toutes

les combinaisons de résultats de clustering possibles de N résultats de clustering des données X. L'objectif est de trouver à partir d'un ensemble $\mathbb{C} \in \breve{\mathbb{C}}$, l'ensemble $\mathbb{C}\star$ tel que :

$$\mathbb{C}\star = \arg\max_{\mathbb{C}\in\breve{\mathbb{C}}}(\mathbf{\Gamma}(\mathbb{C})) \tag{1.21}$$

En fonction du nombre de résultats de clustering impliqués dans la collaboration, l'espace de recherche, c'est-à-dire le nombre de solutions possibles, peut croître très rapidement. En effet, le nombre de clusterings possibles pour un jeu de données étant déjà très élevé (voir section 1.2.1), le fait de faire collaborer plusieurs méthodes fait encore augmenter cette complexité. Soient p le nombre de clusterings possibles et N le nombre de clusterings prenant part dans la collaboration, on obtient p^N solutions potentielles pour $\mathbb{C}\star$.

Dans cette section, nous allons étudier le problème de la résolution de conflit en clustering collaboratif. Différentes stratégies de résolution de conflits sont présentées. Nous étudions tout d'abord un premier groupe de stratégies, appelées stratégies itératives qui consistent toujours à sélectionner un conflit candidat et son éventuelle résolution dans un processus itératif. Chaque stratégie propose une manière différente d'effectuer le choix du conflit candidat. Ces stratégies conservent en grande partie les principes de la méthode SAMARAH. Seul l'ordre dans lequel les conflits sont résolus est modifié. Par contre, dans la dernière stratégie développée lors de cette thèse, le clustering collaboratif est étudié comme un processus d'optimisation. Dans cette stratégie, une adaptation d'un algorithme génétique est utilisée pour optimiser la fonction objective $\Gamma(\mathbb{C})$. Cette approche modifie plus profondément la méthode collaborative existante et propose un formalisme et une approche totalement nouveaux.

1.4.2 Approche itérative pour la résolution de conflits

Lors de la description de la méthode collaborative SAMARAH, nous avons vu qu'il était nécessaire de définir l'ordre dans lesquels les conflits sont considérés. Nous allons maintenant présenter nos propositions pour explorer d'autres stratégies que la sélection du conflit le plus important. De nombreuses stratégies ont été explorées, nous présentons ici les plus intéressantes. Rappelons que les conflits sont évalués en fonction de la similarité des clusters des résultats. Un conflit implique le cluster d'un résultat et des clusters d'un autre résultat. Á noter que quelque soit la stratégie adoptée, le conflit sélectionné sera résolu de la même manière et son résultat ne sera pris en compte que s'il améliore le résultat global.

Conflit le plus important (WCC)

Cette stratégie (appelée WCC) consiste à sélectionner le conflit le plus important, c'est-à-dire celui qui implique le couple de résultats ayant le conflit avec un coefficient de conflit le plus élevé (voir équation (1.12)). C'est la stratégie utilisée dans la version originelle de SAMARAH.

Dans ce cas, la sélection du conflit $\mathcal{K}$ à résoudre peut s'écrire :

$$\mathcal{K} := \arg\max_{\mathcal{K}_{(i)}\in\breve{\mathcal{K}}} CI\left(\mathcal{K}_{(i)}\right) \tag{1.22}$$

La figure 1.11 illustre la mise en œuvre de ce type de sélection sur un exemple de liste de conflits. La probabilité de sélection des conflits est de 1 pour le premier conflit et de 0 pour les autres conflits de la liste. Si on considère la liste des conflits triés par leur importance associé croissante, seul le premier conflit peut être sélectionné comme candidat à la résolution.

Conflit aléatoire (RCC)

Cette stratégie (appelée RCC) consiste à considérer que tous les conflits ont le même potentiel d'amélioration au niveau global. Elle ne tient pas compte de l'importance du conflit dans la sélection du conflit à résoudre et choisit un conflit aléatoirement dans la liste des conflits.

Dans ce cas, la sélection de conflit peut s'écrire :

$$\mathcal{K} := \texttt{aléatoire}\left(\breve{\mathcal{K}}\right) \tag{1.23}$$

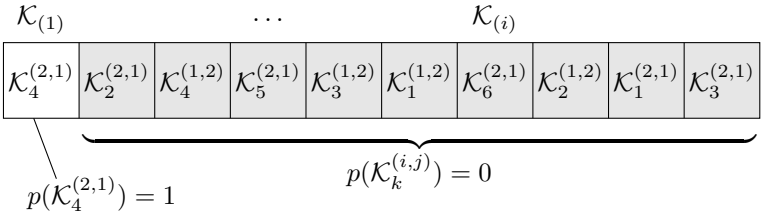


FIGURE 1.11 – Sélection du conflit le plus important (WCC).

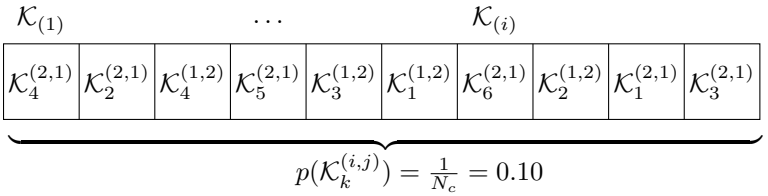


FIGURE 1.12 – Sélection aléatoire du conflit à résoudre (RCC).

La fonction `aléatoire` sélectionne un conflit aléatoirement dans la liste des conflits. À la manière de la stratégie mise en place lors de la sélection du conflit le plus important, ce conflit aléatoirement choisi n'est appliqué que si il améliore la similarité globale de la solution. Par conséquent, la probabilité $p(\mathcal{K}_{(i)})$ que le i-ème conflit soit sélectionné est de :

$$p(\mathcal{K}_{(i)}) = \frac{1}{N_c} \tag{1.24}$$

avec N_c le nombre de conflits. La figure 1.12 illustre la mise en œuvre de ce type de sélection sur un exemple.

Pondération des conflits (R-WCC)

Cette approche (appelée R-WCC) utilise le principe de la roue de la roulette qui consiste à effectuer une sélection proportionnelle à l'importance du conflit. Cette approche est inspirée des méthodes de sélection développées pour la sélection des solutions dans les algorithmes génétiques [Goldberg, 1989] (voir Annexe C). L'importance du conflit est utilisée pour associer une probabilité de sélection à chaque conflit. Dans ce cas, la probabilité de sélection d'un conflit peut s'écrire :

$$p(\mathcal{K}_{(i)}) = \frac{CI\left(\mathcal{K}_{(i)}\right)}{\sum_{j=1}^{N_c} CI\left(\mathcal{K}_{(j)}\right)} \tag{1.25}$$

avec N_c le nombre de conflits. Bien que les conflits ayant une importance élevée aient plus de chance d'être sélectionnées, il y a toujours une possibilité que des conflits moins importants soient sélectionnés.

Dans ce cas, la sélection de conflit peut s'écrire :

$$\mathcal{K} := \mathcal{K}_{(i)} | \left(\sum_{j=0}^{i-1} p(\mathcal{K}_{(j)}) \leq \nu \wedge \sum_{k=0}^{i+1} p(\mathcal{K}_{(k)}) > \nu \right) \tag{1.26}$$

où ν est un réel aléatoire dans $[0; 1]$. La figure 1.13 illustre la mise en œuvre de ce type de sélection sur un exemple de liste de conflits.

1.4.3 Approche évolutionnaire pour la résolution de conflits

Problématique

La classification collaborative peut être vue comme la recherche, à partir d'un ensemble de résultats de clustering, d'un autre ensemble de résultats de clustering qui maximise la similarité

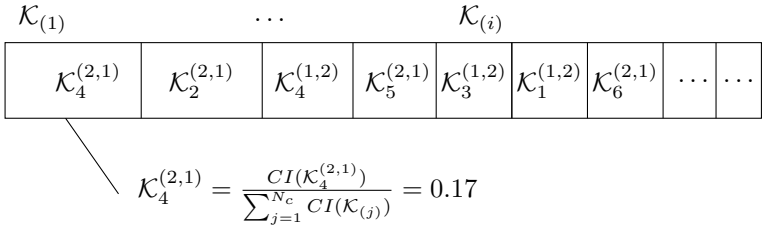


FIGURE 1.13 – Sélection pondérée du conflit à résoudre (R-WCC).

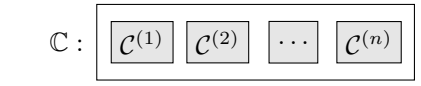


FIGURE 1.14 – Représentation d'une solution dans le cadre de l'algorithme génétique composée de plusieurs résultats de clustering.

ainsi que la qualité des résultats (voir équation 1.21). Pour trouver ce nouvel ensemble de résultats nous allons voir dans cette section comment nous avons utilisé et adapté un algorithme génétique. Cette stratégie (appelée GR) implique de redéfinir complètement l'approche collaborative et bien que les concepts et les critères d'évaluation soient conservés, la stratégie mise en place pour la recherche d'une solution maximisant la similarité et la qualité des résultats est totalement différente et innovante.

Les algorithmes génétiques appartiennent à la famille des algorithmes évolutionnaires. Leur but est d'obtenir une solution approchée à un problème d'optimisation, lorsqu'il n'existe pas (ou qu'on ne connaît pas) de méthode exacte pour le résoudre en un temps raisonnable. Les algorithmes génétiques utilisent la notion de sélection naturelle développée au $20^{\text{ème}}$ siècle par le scientifique Charles Darwin et l'appliquent à une population de solutions potentielles au problème donné. Cette population évolue au cours des générations, ce qui va simuler l'évolution de la population au fil du temps. Au cours de ces générations, les solutions qui répondent le mieux au problème sont conservées et sont utilisées pour en construire de nouvelles.

Les algorithmes génétiques sont connus pour être efficaces même si $\mathbb{F}(g)$ définit un espace complexe de recherche (c'est à dire avec de nombreux optima locaux). Les paramètres de l'individu à optimiser sont appelé génotype de l'individu.

Formalisation

Dans le cadre du clustering collaboratif, une solution dans notre algorithme génétique sera un ensemble de résultats de clustering $\mathbb{C}$ (voir figure 1.14). Contrairement aux approches itératives, nous allons manipuler un ensemble d'ensembles de résultats. Au lieu de faire évoluer un ensemble de résultats qui est modifié itérativement en résolvant un conflit à la fois, nous allons utiliser plusieurs ensembles de résultats qui vont évoluer conjointement et vont pouvoir mieux explorer l'espace des solutions. Ces différentes solutions vont évoluer au cours de générations de l'algorithme génétique. Elles vont être utilisées pour créer de nouvelles solutions qui répondent mieux au problème, c'est-à-dire dont la similarité et la qualité sont supérieures. Il est donc important de noter qu'une solution est un ensemble de résultats de clustering et non pas un unique clustering.

Pour initialiser l'algorithme génétique, il est nécessaire de générer une population initiale qui va servir de base pour la création de nouvelles solutions au cours des générations. Cette population est composée d'un ensemble de solutions au problème considéré. Pour créer cette population initiale, nous avons fait le choix de ne partir que de la solution disponible initialement dans les approches itératives, et de modifier celle-ci par l'application d'opérateurs paramétrés aléatoirement sur les résultats la composant. À partir de la solution initialement disponible, un ensemble d'opérateurs (scission, fusion et reclustering) est appliqué aux différents résultats pour créer des solutions alternatives. Ce choix a été fait pour pouvoir se comparer de manière équitable avec les approches itératives qui ne considèrent qu'une solution initiale.

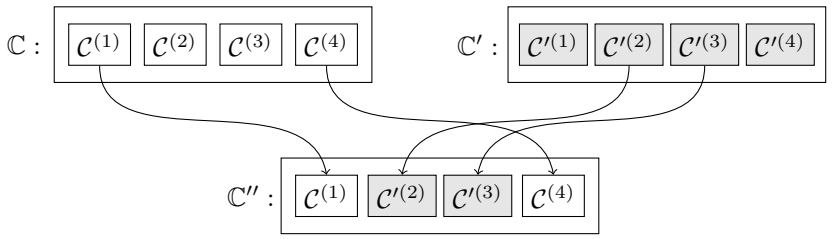


FIGURE 1.15 – Illustration de l'opérateur de croisement.

Une fois cette population créée, l'algorithme génétique repose sur les étapes suivantes qui représentent la transition entre les générations :

1. *évaluation des solutions de la population :* chaque solution est évaluée grâce à son coefficient global d'évaluation défini dans SAMARAH (voir équation (1.15)). Cette mesure reflète la similarité et la qualité des résultats et est utilisée pour juger de la qualité d'une solution. Plus cette valeur est élevée, meilleure est la solution.
2. *sélection des solutions par rapport à leur évaluation* : une solution avec un fort coefficient global d'évaluation a plus de chance d'être sélectionnée qu'une solution avec une valeur plus faible. La probabilité de sélection d'une solution est donc proportionnelle à sa qualité.
3. *croisement :* l'opérateur de croisement est utilisé pour créer de nouvelles solutions à partir des solutions existantes dans la population. Deux solutions (appelées *parents*) sont sélectionnées et donnent naissance à une nouvelle solution en mélangeant les résultats de clustering présents dans les deux solutions (*slicing cross-over*). La figure 1.15 illustre ce processus de croisement. Il est important de noter que l'ordre des résultats dans chaque solution est important et doit être conservé. En effet, chaque résultat est lié à la méthode utilisée pour générer ce résultat particulier, qui est ensuite utilisé pour résoudre les conflits. Lors de la création de la nouvelle solution, à chaque rang, il est équiprobable de sélectionner un résultat du premier parent ou du deuxième parent.
4. *mutation :* à chaque génération, un conflit sélectionné aléatoirement dans la liste des conflits associé à un résultat selon une probabilité fixée a priori par l'expert est résolu. Ainsi, nous cherchons à éviter à l'algorithme d'être bloqué dans un optimum local et nous créons de nouvelles solutions en maintenant de la diversité au sein de la population. La meilleure solution de la population n'est pas affectée par l'opérateur de mutation car il est possible que l'application de cet opérateur diminue la qualité du résultat.

L'utilisation d'un algorithme génétique permet d'effectuer une recherche plus complète dans l'espace des solutions. En effet, beaucoup plus de solutions vont être explorées ce qui augmente les chances de trouver une solution proposant une meilleure similarité ainsi qu'une meilleure qualité. Cependant, comme beaucoup plus de solutions vont être manipulées, cette approche est plus gourmande en mémoire et en temps que les approches itératives.

Pour améliorer la rapidité de la convergence de l'algorithme génétique, nous avons utilisé le principe dit de l'optimisation *Lamarckienne* [Blansché, 2006]. Cette optimisation consiste, à chaque génération, à appliquer à chaque solution une optimisation locale qui a pour but d'améliorer légèrement la solution. Dans notre cas, l'opération à appliquer à chaque solution est la résolution du conflit le plus important. Si la résolution de conflit améliore la solution, alors la modification est conservée et elle est appliquée à la solution. En utilisant cette approche l'ensemble des solutions s'améliore légèrement à chaque génération. Ce processus permet d'améliorer la rapidité de convergence de l'algorithme génétique. Néanmoins, il est toujours difficile de trouver un bon compromis entre rapidité de convergence et maintien de la diversité dans la population. Cependant, d'après nos expériences, l'utilisation de l'approche Lamarckienne est bénéfique et permet d'améliorer les performances de l'algorithme sans perdre de qualité au niveau des solutions.

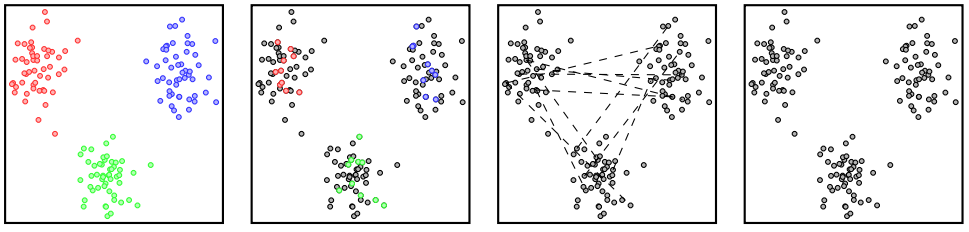
Chapitre 2

État de l'art : Clustering semi-supervisé

2.1 Introduction

Les résultats parfois décevants des algorithmes de clustering face aux données de plus en plus complexes ont poussé la communauté scientifique à proposer de nouvelles approches permettant d'obtenir de meilleurs résultats sur ce type de données. Comme présenté dans le Chapitre 2, le processus de clustering est par définition une approche non supervisée, c'est-à-dire qu'il se base uniquement sur les données et n'utilise pas de connaissances. Cependant, sans aucune supervision, les algorithmes de clustering peuvent aboutir à des solutions non pertinentes. Les recherches se sont donc concentrées sur des approches permettant de guider le processus par des connaissances du domaine ou fournies par l'expert. L'objectif est de permettre à l'expert humain d'incorporer des connaissances du domaine dans le processus de fouille de données, et de le guider ainsi vers de meilleurs résultats. Plusieurs études [Anand et al., 1995; Kopanas et al., 2002] ont montré le rôle important joué par ces connaissances du domaine ainsi que par l'expert dans le processus de fouille de données. Ces études expliquent que le processus d'extraction de connaissances dans les données ne peut pas être totalement automatique et il est nécessaire d'étudier les mécanismes permettant les interactions entre d'une part les traitements automatiques et d'autre part la supervision de l'expert qui guide ces traitements. L'idéal est de pouvoir représenter la connaissance de l'expert en vue d'automatiser l'utilisation de celle-ci dans le processus de fouille. Cependant, en fonction des domaines, la représentation et le type des connaissances peuvent être très hétérogènes.

Deux principales représentations des connaissances ont été proposées jusqu'alors dans le cadre du clustering. La première concerne l'utilisation de contraintes entre objets et la seconde l'utilisation d'objets étiquetés. Ces deux types de connaissances sont présentés dans les sections suivantes et sont illustrés sur la figure 2.1 qui les situent par rapport aux approches supervisées et non supervisées.



(a) Supervisé. (b) Partiellement étiqueté. (c) Partiellement contraint. (d) Non supervisé.

FIGURE 2.1 – Exemple des différents types de connaissances selon Jain [2009].

2.2 Connaissances

2.2.1 Contraintes entre les objets

En clustering contraint, les connaissances sont exprimées sous la forme de relations *must-link* et *cannot-link* et sont utilisées pour guider le processus de clustering. Un lien must-link indique que deux objets devraient être dans le même cluster à la fin du processus alors qu'un lien cannot-link signifie le contraire. Ce type de connaissance est parfois plus simple à obtenir qu'un classique ensemble d'objets étiquetés. En effet, il n'est pas nécessaire de connaitre le nombre de classes, leurs étiquettes, leurs caractéristiques, etc. Seule l'information entre couple d'objets est modélisée.

Formellement, l'ensemble des contraintes est noté C avec $(C_=)$ l'ensemble des must-link et $(C_\neq)$ l'ensemble des cannot-link : $C = (C_=) \cup (C_\neq)$. Si deux objets x_i et x_j doivent être dans le même cluster, ils seront liés par un lien must-link exprimé par $c_=(i, j)$. À l'inverse, si deux objets ne doivent pas être placés dans le même cluster, ils seront liés par une contrainte cannot-link $c_\neq(i, j)$. On notera qu'il existe une inférence transitive concernant les must-link. Par exemple si $c_=(i, j)$ et $c_=(j, k)$ alors $c_=(i, k)$. Ce n'est pas le cas concernant les cannot-link comme $c_\neq(i, j)$ et $c_\neq(j, k)$ n'implique pas $c_\neq(i, k)$. Cependant on notera que si $c_=(i, j)$ et $c_\neq(i, k)$ alors $c_\neq(j, k)$. Ces propriétés permettent d'inférer de nouvelles contraintes à partir d'un ensemble de contraintes disponibles. Il est courant de représenter l'ensemble des must-link par un graphe et d'en étudier les composantes connexes. Il faut être vigilant car plus on augmente le nombre de contraintes, plus il peut être difficile de trouver une solution ne violant aucune de celles-ci. De plus, certaines contraintes peuvent être contradictoires et peuvent alors rapidement rendre inefficace le système tentant de les utiliser.

Wagstaff et al. [2001] ont présenté une version contrainte de l'algorithme KMEANS (COP-KMEANS) qui utilise ce type de contrainte pour biaiser l'affectation des objets aux clusters. À chaque étape, l'algorithme essaie de satisfaire les contraintes spécifiées par l'expert. Ces travaux contiennent également une version contrainte de l'algorithme COBWEB (COP-COBWEB). Ces algorithmes ne renvoient un résultat que s'ils trouvent une solution qui satisfait toutes les contraintes.

Ces algorithmes qui tentent de faire respecter toutes les contraintes cherchent généralement une solution optimale. Ceci n'est pas toujours la meilleure stratégie car il n'y a aucune garantie que l'algorithme pourra converger et trouver cette solution. Davidson and Ravi [2005] ont montré que la satisfaction de toutes les contraintes must-link et cannot-link par un algorithme de type KMEANS est un problème NP-complet, et que les contraintes cannot-link peuvent empêcher l'algorithme de converger. Le problème de convergence peut être évité en étudiant le respect des contraintes au niveau local et non au niveau d'une solution globale. Davidson and Ravi [2005] ont proposé un algorithme de clustering hiérarchique contraint et ont montré que le problème de satisfaction des contraintes devient alors un problème P-complet si l'on considère un respect local des contraintes.

Ces contraintes peuvent également servir à apprendre une fonction de distance biaisée par la connaissance des liens entre les objets [Bilenko et al., 2004]. La distance entre deux objets est réduite pour un lien must-link et augmentée pour un cannot-link. Soit X les données, l'ensemble des must-link $C_=$ et l'ensemble des cannot-link $C_\neq$, le but est de trouver une distance métrique D entre les objets de X qui minimise :

$$\sum_{c_=(x,y)} D(x, y) \tag{2.1}$$

et maximise :

$$\sum_{c_\neq(x,y)} D(x, y) \tag{2.2}$$

Dans [Basu et al., 2004], les auteurs expliquent qu'il existe deux manières d'utiliser les contraintes. La première est de modifier un algorithme de clustering pour qu'il respecte les contraintes imposées. La seconde est de modifier la fonction de distance entre les objets. Il propose d'utiliser ces deux approches conjointement via un modèle probabiliste basé sur les champs de Markov aléatoires. Un algorithme basé sur EM est utilisé pour trouver un minimum local à la fonction objective combinant respect des contraintes et modification de la métrique.

Klein et al. [2002] ont utilisé les contraintes entre paire d'objets ainsi que la position et le voisinage de ces objets pour créer de nouvelles contraintes. Les contraintes sont propagées aux objets avoisinant les objets impliqués dans des contraintes. Par exemple si les objets (x, y) sont liés par un must-link ($c_=(x, y)$) et que l'objet z est *proche* de x alors il existe sûrement un lien must-link entre z et y ($c_=(z, y)$). Cette approche permet de générer de nombreuses contraintes à partir d'un nombre limité de contraintes initiales. Cette stratégie a amélioré les résultats des précédentes versions contraintes de l'algorithme KMEANS et requiert généralement moins de contraintes initiales pour obtenir les mêmes performances. Il est cependant nécessaire de définir la notion de voisinage pour considérer qu'un objet est assez proche d'un autre pour bénéficier de ses contraintes.

Dans [Cohn et al., 2003], la divergence asymétrique de Kullback–Leibler a été utilisée pour le clustering de documents avec interaction de l'expert. Ainsi, la fonction de distance apprise (c'est-à-dire biaisée par les contraintes) reflète les différentes perspectives des experts sur les documents. Dans [Bilenko et al., 2004], les auteurs proposent un algorithme appelé MPCK-KMEANS, qui procède à l'apprentissage d'une distance sur les objets affectés par des contraintes à chaque itération de l'algorithme de clustering. Ainsi, l'algorithme apprend différentes fonctions de distance pour chacun des clusters.

L'intégration de connaissances sous la forme de contraintes a également été étudiée dans le clustering basé sur la densité (voir Chapitre 2). Ruiz et al. [2009] ont proposé un système basé sur l'algorithme DBSCAN qui tire parti des must-link et des cannot-link. L'algorithme proposé utilise DBSCAN pour créer des clusters temporaires qui sont ensuite coupés et fusionnés en fonction des contraintes disponibles. Les expériences présentées montrent que la version contrainte de DBSCAN donne de meilleurs résultats que la version sans contrainte.

Les travaux récents en clustering contraint se concentrent sur l'évaluation de l'utilité (c'est-à-dire l'intérêt potentiel) d'un ensemble de contraintes [Davidson et al., 2006; Wagstaff, 2007]. Le but est de pouvoir évaluer la qualité d'un ensemble de contraintes, c'est-à-dire de vérifier, par exemple, si les contraintes sont cohérentes par rapport à l'espace des données, s'il y a des contraintes contradictoires, etc.

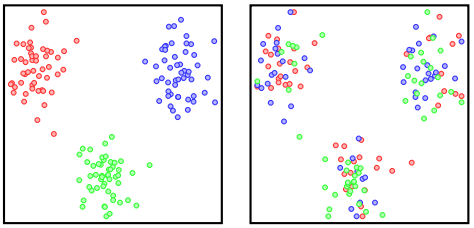
2.2.2 Ensemble d'objets étiquetés

Une deuxième approche utilisée pour l'intégration de connaissances en clustering fait appel à un ensemble d'objets étiquetés. L'ensemble des données disponibles $X = \{x_1, \ldots, x_n\}$ est généralement séparé en deux sous-ensembles contenant d'une part les objets dont on connait l'étiquette, et d'autre part les objets dont l'étiquette est inconnue. Formellement, on définit $\mathcal{Y} = \{y_1, \ldots y_t\}$ comme l'ensemble des étiquettes possibles (c'est-à-dire l'ensemble des labels des classes), $X_l = \{(x_i, y_i)|i = 1, \ldots, m\}$ l'ensemble des données dont l'étiquette est connue et $X_u = \{x_i|i = m+1, \ldots, l\}$ celui des objets dont l'étiquette est inconnue. Ce type d'approche partage des similarités avec les approches supervisées. En effet dans le cas supervisé, l'ensemble des objets étiquetés X_l est utilisé pour apprendre une fonction de classification qui est ensuite appliquée pour prédire la classe des objets de X_u. Ici, l'approche est sensiblement différente car l'objectif est d'étudier la structure de X_u en appliquant un algorithme de clustering, mais en utilisant les informations disponibles dans X_l pour améliorer le processus. De plus, les contraintes applicables sur l'ensemble X_l en classification supervisée sont généralement moins importantes en clustering semi-supervisé. Par exemple, il n'est pas toujours nécessaire d'avoir des exemples pour chaque classe, le nombre d'objets étiquetés peut être très faible ou encore le nombre d'objets étiquetés peut être très différent suivant les classes.

Différentes approches existent concernant l'utilisation de ce type de connaissances. Cependant, la majorité d'entre-elles font un certain nombre d'hypothèses sur ces objets étiquetés. Les deux plus importantes sont :

- les objets *proches* dans l'espace des données ont probablement la même étiquette ;
- deux objets que l'on peut connecter en passant par des régions de forte densité ont probablement la même étiquette.

Ces deux hypothèses sont assez intuitives. Elles stipulent que l'information des classes, ici utilisée comme connaissance, suit globalement la distribution des données. En effet, si la distribution des étiquettes n'a aucune corrélation avec la distribution des clusters, alors la connaissance de ces étiquettes n'apportera aucune information au clustering (voir figure 2.2). Pire, elle risque de fausser et de guider notre algorithme vers une solution non voulue. Bien sûr, il peut être intéressant



(a) (b)

FIGURE 2.2 – Deux cas où la dis3tribution des classes suit la distribution des données (a) et où elle ne la suit pas (b). Dans le cas (b) la connaissance des classes n'est pas une information pertinente.

dans certains cas que l'information des classes ne suive pas totalement la distribution des données. Par exemple si une classe est composée de plusieurs clusters dans l'espace des données, les deux règles précédentes sont alors violées. Cependant, cette connaissance peut être utilisée pour identifier des informations que les données seules ne pouvaient pas fournir. Comme, par exemple, que deux clusters mal séparés dans l'espace des données appartiennent à des classes différentes. Les connaissances peuvent également servir à lever des ambiguïtés quand deux clusterings des données semblent identiques et que rien par rapport aux données ne nous permet de faire un choix. Dans ce cas, les connaissances peuvent servir à sélectionner le résultat le plus en accord avec les connaissances de l'expert.

Utilisation des connaissances pour l'initialisation

Dans [Basu et al., 2002], l'ensemble des objets étiquetés est utilisé pour initialiser les centroïdes dans l'algorithme KMEANS. Deux algorithmes ont été proposés, Seeded-KMEANS et Constrained-KMEANS. Dans Seeded-KMEANS, les objets étiquetés sont utilisés pour initialiser les clusters et les clusters sont mis à jour durant le processus de clustering comme dans la version classique de KMEANS. Les objets étiquetés ne sont utilisés que lors de l'étape initiale d'initialisation des centroïdes. Dans Constrained-KMEANS, les objets étiquetés sont utilisés pour l'initialisation et sont affectés à un cluster et ne peuvent plus en changer par la suite. Seuls les objets non étiquetés peuvent changer de cluster durant l'étape d'affectation aux clusters lors des itérations de l'algorithme. Le choix entre ces deux approches est généralement fait en fonction de la connaissance sur le bruit dans les données. En effet, si les données sont bruitées il peut être dangereux de ne pas autoriser de changement de clusters.

Ce type d'approche par initialisation a l'avantage de pouvoir laisser un espace de liberté à l'expert pour la découverte de clusters pour lesquels aucune connaissance n'est disponible. Par exemple, si l'expert dispose d'objets étiquetés pour deux classes, il peut utiliser ces objets pour initialiser deux centroïdes, mais peut également rajouter des centroïdes initialisés aléatoirement dans l'espoir de découvrir de nouveaux clusters. Cette approche contraint cependant l'expert à paramétrer avec précision le nombre de clusters attendus.

Dans [Wang et al., 2007], les auteurs ont introduit deux stratégies pour une initialisation de ces clusters sans utilisation de connaissance. La première stratégie intitulée Farthest-Seeded-KMEANS, consiste à initialiser les clusters sans connaissance en cherchant les objets les plus éloignés des centroïdes déjà créés. À l'initialisation de l'algorithme, un centroïde est créé pour chaque ensemble d'objets appartenant à la même classe dans l'ensemble des objets étiquetés. Puis, pour chaque centroïde que l'expert souhaite ajouter, l'algorithme va chercher l'objet des données le plus éloigné de tous les centroïdes déjà existants. Il est important de noter que cette stratégie est très sensible aux objets atypiques [1]. En effet, ceux-ci ont une grande chance d'être sélectionnés pour l'initialisation des clusters sans connaissance. La deuxième stratégie proposée est intitulée Splitting-Seeded-KMEANS. Dans cette approche, autant de clusters que de classes sont créés et les clusters sont initialisés avec les connaissances disponibles. Cette étape revient à appliquer l'algorithme Seeded-KMEANS sur les données. Puis, le cluster le plus dispersé est coupé en deux (les auteurs utilisent la

1. Objets très différents par rapport aux autres objets composant le jeu de données.

distance moyenne des objets d'un cluster au centroïde pour effectuer le choix). Cette opération de scission est appliquée jusqu'à obtenir le nombre de clusters demandé par l'expert. Cette stratégie suppose que les clusters les plus dispersés sont en fait composés de plusieurs *sous-clusters* qu'il est intéressant d'identifier. Les auteurs ont montré sur quelques expérimentations que ces stratégies donnent de meilleurs résultats que l'initialisation aléatoire.

Bohm and Plant [2008] ont également proposé un algorithme de clustering hiérarchique basé sur la densité qui utilise des données étiquetées. À chaque étape, l'algorithme affecte l'étiquette la plus consistante à chaque cluster. Cette information est ensuite utilisée pour guider la construction de la hiérarchie.

Pour introduire des connaissances dans un algorithme de clustering quand les données étiquetées ne partagent qu'un sous-ensemble des attributs avec les objets non étiquetés, Gao et al. [2006] ont formulé une nouvelle approche comme un problème d'optimisation. Les auteurs ont introduit deux algorithmes d'apprentissage qui sont basés sur des méthodes de clustering dures et floues. Une étude empirique montre que les algorithmes proposés améliorent la qualité des résultats de clustering malgré un nombre limité d'objets étiquetés.

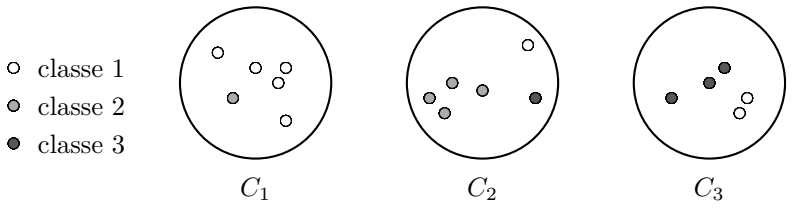
Utilisation des connaissances pour l'évaluation de la pureté des clusters

Une autre approche pour tirer parti de ces objets étiquetés est d'évaluer la pureté des clusters. Rappelons qu'une des hypothèses concernant l'utilisation d'objets étiquetés comme connaissance est que deux objets de la même étiquette sont probablement proches dans l'espace des données. Ceci peut être traduit par le fait que les objets de la même étiquette appartiennent au même cluster ou qu'un cluster ne contient que des objets ayant la même étiquette. Cette propriété peut être utilisée dans un algorithme de clustering pour guider le processus de clustering pour créer des clusters ayant une forte pureté, c'est-à-dire contenant des objets d'une unique étiquette.

Il existe de très nombreuses manières d'étudier la pureté des clusters d'un résultat de clustering. À titre d'exemple, la figure 2.3 illustre une façon simple de calculer la pureté. Ce calcul compte, pour chaque cluster, le nombre d'objets de l'étiquette la plus représentée dans le cluster. Cette stratégie considère que tous les objets du cluster doivent appartenir à l'étiquette majoritaire (c'est-à-dire la plus représentée au sein du cluster). Les objets des étiquettes minoritaires (c'est-à-dire n'étant pas de l'étiquette majoritaire) sont considérés comme des erreurs. Ce calcul est effectué pour chaque cluster et est ensuite moyenné pour donner une évaluation à l'ensemble du clustering.

Ce type d'évaluation peut être intégré dans des algorithmes de clustering comme un moyen d'utiliser des connaissances. Par exemple, Demiriz et al. [1999] ont utilisé ce type d'approche pour créer un algorithme de clustering semi-supervisé. Cet algorithme repose sur une version de l'algorithme KMEANS utilisant un algorithme génétique pour calculer les centroïdes. La fonction objective de cet algorithme est une moyenne géométrique entre la fonction objective classique de KMEANS (la dispersion des clusters) et un critère de pureté calculé à l'aide de la connaissance de l'étiquette d'un certain nombre d'objets. Les expériences menées montrent que l'algorithme utilisant les connaissances fournit de meilleurs résultats que l'algorithme utilisant uniquement les données non étiquetées.

Une approche faisant également appel à l'évaluation de la pureté des clusters est proposée par Eick et al. [2004]. Cette approche appelée *clustering supervisé*, consiste à utiliser l'étiquette des objets, afin de créer des clusters purs en terme d'étiquette. Dans ce cadre, on considère que l'on connait l'étiquette de tous les objets à classer. Le but est de former les groupes les plus homogènes possible en terme de classe. Deux principaux algorithmes sont proposés. Le premier, SRIDHCR (*Single Representative Insertion/Deletion Steepest Descent Hill Climbing with Randomized Restart*), consiste à chercher un ensemble de médoides qui maximise la pureté des clusters. L'algorithme démarre en sélectionnant aléatoirement un certain nombre d'objets comme médoides initiaux. Puis, dans un processus itératif, des médoides sont retirés et d'autres sont ajoutés à l'ensemble. À chaque itération, la solution maximisant la pureté des clusters en terme d'étiquette est conservée. Comme l'algorithme est très dépendant des médoides sélectionnés initialement, celui-ci est relancé plusieurs fois avec différents ensembles initiaux. La meilleure solution parmi les différentes exécutions est conservée. Le deuxième algorithme SCEC (*Supervised Clustering using Evolutionary Computing*), utilise un algorithme génétique pour trouver le meilleur ensemble de médoides. Une solution est codée comme un ensemble de taille variable de médoides. Une population de ces solutions est générée



$$\mathbf{\Pi}_{\text{simple}} = (5 + 4 + 3)/17 \approx 0.71$$

FIGURE 2.3 – Exemple de calcul de pureté simple.

aléatoirement au début de l'algorithme, puis évolue au cours des générations. L'avantage de l'algorithme génétique (SCEC) par rapport à la solution itérative (SRIDHCR) est que l'espace des solutions est mieux parcouru et l'algorithme a donc plus de chance de trouver une meilleure solution, bien qu'il ne soit jamais garanti d'atteindre le maximum global.

Il est également à noter que, d'une façon générale, l'ensemble d'objets étiquetés peut être utilisé pour générer un ensemble de contraintes (*must-link* et *cannot-link*). Il suffit de créer des must-link entre les objets partageant la même étiquette et des cannot-link entre les objets d'étiquettes différentes. On peut alors envisager d'utiliser tous les algorithmes vus dans la section précédente. Ainsi, les contraintes sont généralement vues comme une connaissance plus faible que des objets étiquetés.

2.2.3 Autres types de connaissances

Dans les deux sections précédentes nous avons vu l'utilisation de connaissance en clustering sous forme de contraintes (section 2.2.1) et sous forme d'objets étiquetés (section 2.2.2). Cependant, d'autres types de connaissance existent. Par exemple, on peut considérer que le nombre de clusters qui est souvent un paramètre de l'algorithme de clustering est une connaissance du domaine. En effet, il est nécessaire de disposer d'une expertise importante sur ses données pour pouvoir fournir le nombre de clusters adéquat. Un autre type de connaissance peut être la taille des clusters, c'est-à-dire une taille minimale ou maximale d'objets par clusters. Bradley et al. [2000] ont défini la τ_n-contrainte qui spécifie une taille n maximale pour les clusters. La ϵ-contrainte définit que deux objets à l'intérieur d'un même cluster doivent être situés plus près qu'un seuil de distance ϵ. La δ-contrainte définit quant à elle que deux objets de deux clusters différents doivent être au moins séparés d'une distance égale à δ [Davidson and Ravi, 2005]. À noter que ces approches ne précisent pas comment paramétrer ces seuils qui sont laissées au choix de l'expert.

Kumar and Kummamuru [2008] ont également introduit un autre type de connaissance dans un algorithme de clustering basé sur la comparaison relative entre les objets. Par exemple *"l'objet x est plus proche de l'objet y que de l'objet z"*. Une étude expérimentale sur des données à forte dimension a montré que l'algorithme proposé obtenait de meilleures performances et était plus robuste qu'un algorithme similaire utilisant des contraintes entre les objets (*must-link* et *cannot-link*).

Un autre type de connaissances peut être défini sous la forme d'un résultat de clustering déjà existant. En effet, il existe souvent plusieurs clusterings possibles d'un même jeu de données. L'expert peut avoir obtenu un résultat de clustering, en être satisfait, mais vouloir continuer le processus pour identifier d'autres regroupements parmi ses données. Le premier clustering sert alors de contrainte pour éviter de retrouver cette solution. Le but est de trouver une autre solution de bonne qualité mais différente de la solution déjà trouvée lors de la première exécution. Bae and Bailey [2006] ont proposé un système appelé COALA pour trouver un clustering alternatif à partir d'un clustering existant. Le système est basé sur la recherche d'une solution de bonne qualité mais la plus dissimilaire possible d'un clustering donné. Qi and Davidson [2009] ont continué ce travail de recherche de clustering alternatif en posant le problème comme un problème d'optimisation contrainte ce qui permet son analyse en détail. L'algorithme proposé est flexible car il permet à l'expert de donner des retours positifs et négatifs sur les clusters de la solution existante. L'expert pourra indiquer quels clusters il souhaite conserver et s'il souhaite privilégier une solution très

différente ou au contraire légèrement différente mais en préservant la qualité.

Une autre formalisation des connaissances a été proposée par Pedrycz [2004] qui les définit de trois façons différentes comme l'incertitude de l'appartenance aux classes, l'information basée sur la proximité de paires d'objets et enfin l'étiquetage d'une partie des données. L'auteur discute des différentes manières d'intégrer ces connaissances dans l'algorithme FUZZY-C-MEANS. Bouchachia and Pedrycz [2006] ont étendu ces travaux et présenté un algorithme de clustering flou qui prend en compte des connaissances du domaine à travers des objets étiquetés. Un des avantages de la méthode est de prendre en compte les classes séparées en plusieurs clusters. Pendant l'étape de collaboration, la méthode identifie les classes correspondantes à plusieurs clusters et ajoute ou retire des clusters à partir de cette information. L'algorithme Fuzzy C-means a été utilisé pour expérimenté les concepts théoriques présentés. (A REVOIR)

Une autre alternative pour représenter et utiliser des connaissances est de faire appel à une ontologie. Une ontologie [Gruber, 1995] est un réseau sémantique qui regroupe un ensemble de concepts tentant de décrire un domaine. Ces concepts sont liés les uns aux autres par des relations taxonomiques (hiérarchisation des concepts) d'une part, et sémantiques d'autre part. Les ontologies fournissent un cadre formel pour la représentation de connaissances. Ce type de représentation a également été étudié pour intégrer des connaissances dans les algorithmes de clustering. Adryan and Schuh [2004] ont utilisé *Gene Ontology* (GO) qui est une ontologie contenant des annotations sur les fonctions des gènes sous forme hiérarchique. La structure de GO représente les liens entre les termes biologiques associés aux gènes. Les auteurs proposent une application qui incorpore la structure de GO comme une plateforme pour sélectionner des sous-ensembles de données d'expression de gène qui sont ensuite utilisés dans un processus de clustering. Breaux and Reed [2005] ont présenté un système utilisant un algorithme de clustering hiérarchique modifié pour tenir compte de la hiérarchie d'une ontologie. L'algorithme est utilisé pour le clustering de documents car l'ontologie utilisée fournit des relations entre les mots qui peuvent être utilisées comme une distance dans l'algorithme de clustering. Une autre proposition [Hotho et al., 2001] pour le clustering de documents utilise les connaissances d'une ontologie lors d'une pré-étape au clustering ainsi que pour la sélection des résultats. Différents jeux de données sont produits en utilisant différents attributs sélectionnés grâce aux informations issues des concepts de l'ontologie. Plusieurs résultats de clustering sont ensuite calculés à l'aide de l'algorithme KMEANS. Les résultats sont ensuite distingués et expliqués grâce aux différents concepts correspondants dans l'ontologie. Une ontologie peut également servir à fournir des connaissances pour personaliser la recherche d'information dans des bases de données [Brut et al., 2008].

Les ontologies sont un moyen puissant de structurer et d'organiser des connaissances sur un domaine. Cependant, les connaissances ainsi représentées sont très spécifiques à un domaine et bien souvent à une application donnée. Les informations que l'on peut retirer des ontologies sont très hétérogènes et peuvent affecter les algorithmes différemment selon les domaines d'application.

2.3 Intégration de connaissances en clustering collaboratif

Il existe différents niveaux d'intégration des connaissances dans la méthode SAMARAH. La figure 2.4 présente les trois principaux niveaux d'intégration des connaissances.

- Le premier niveau (1) consiste à utiliser des connaissances a posteriori, c'est-à-dire à la fin du processus. En effet, nous verrons que les informations disponibles vont nous permettre d'améliorer le résultat final. Le processus mis en œuvre est un mécanisme d'étiquetage qui consiste à affecter une étiquette aux clusters et non plus aux objets. Nous verrons que ce niveau est le plus simple à mettre en œuvre car il ne nécessite pas de modifier la méthode directement, mais uniquement d'ajouter une étape en fin du processus.

- Le second niveau (2) consiste à utiliser les connaissances directement dans la méthode de clustering collaboratif afin de guider le processus de collaboration. Rappelons que le système doit prendre de nombreuses décisions quant aux modifications à apporter aux résultats pour les rendre plus similaires et de bonne qualité. Les connaissances sont utilisées à ce niveau pour évaluer la qualité des résultats et guider le choix des modifications à effectuer.

- Le dernier niveau (3) consiste à modifier les méthodes qui vont prendre part à la collaboration pour qu'elles tiennent compte des connaissances directement au niveau de leur clustering des don-

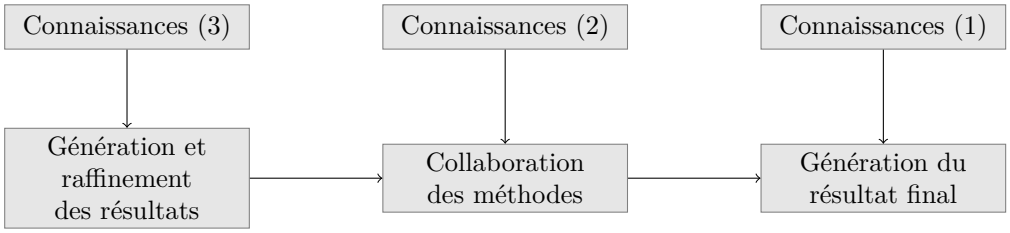


FIGURE 2.4 – Les différents niveaux d'intégration des connaissances.

nées. Ce niveau est le plus coûteux à mettre en œuvre car il nécessite de modifier en profondeur chacune des méthodes que l'on souhaite voir collaborer. En effet, il est nécessaire d'adapter les opérateurs spécifiques à chaque méthode (scission, fusion, reclustering) pour qu'ils utilisent les connaissances disponibles.

2.3.1 Intégration des connaissances a posteriori

L'intégration de la connaissance a posteriori consiste à utiliser la connaissance des objets étiquetés pour étiqueter les clusters d'un résultat de clustering et à améliorer le résultat produit. La plupart des algorithmes de clustering semi-supervisé opèrent sous deux contraintes fortes :

1. À chaque étiquette (ou classe) ne correspond qu'un seul cluster ;
2. On dispose d'au moins un exemple de chaque classe.

Or, ces deux hypothèses sont souvent mises à défaut dans les problèmes réels. En effet, il est souvent possible de rencontrer des classes correspondant en fait à plusieurs clusters. De plus, le choix des objets étiquetés est souvent effectué de manière aléatoire ce qui explique pourquoi il n'est pas garanti d'avoir des exemples pour chacune des classes réelles. Nous nous proposons ici de relâcher ces deux contraintes. Pour cela, nous utilisons les étiquettes des objets pour affecter une étiquette à chacun des clusters quand cela est possible. Pour chaque cluster, la distribution des étiquettes présentes dans celui-ci est étudiée. Un vote à la majorité est ensuite effectué pour connaître l'étiquette la plus probable pour le cluster. Une approche sensiblement similaire a déjà été explorée par Jia and Richards [2005] pour réduire la taille d'un jeu d'apprentissage en classification supervisée. Nous allons nous en inspirer ici pour proposer un mécanisme d'étiquetage des clusters.

La première étape consiste à appliquer un clustering collaboratif sur les données. À l'issue du processus collaboratif, nous allons travailler sur le résultat unifié, c'est-à-dire le résultat final issu de la fusion des différents résultats impliqués dans la collaboration (voir section 1.3.2). Il aurait cependant été possible d'appliquer la même approche à chaque résultat raffiné.

Soit $\mathcal{C}^{(*)} = \{C_1^{(*)}, \ldots, C_k^{(*)}\}$ le résultat unifié obtenu à la fin du processus collaboratif. L'objectif est d'utiliser les connaissances pour affecter une étiquette de $\mathcal{Y} = \{Y_1, \ldots, Y_L\}$ à chacun des clusters de $\mathcal{C}^{(*)}$. Un vote est effectué au sein de chaque cluster pour identifier l'étiquette la plus présence. Pour cela, l'intersection entre les objets appartenant au cluster et les objets étiquetés est calculée. L'intersection maximale est conservée pour affecter une étiquette au cluster. La règle de vote suivante est appliquée :

$$\texttt{classe-majoritaire}(C_k^{(*)}) = \arg \max_{\mathcal{C}_i^{(*)} \in \mathcal{C}^{(*)}} |\mathcal{C}_i^{(*)} \cap Y_l| \tag{2.3}$$

Si aucun objet étiqueté n'est disponible dans un cluster, c'est-à-dire $\forall k, \mathcal{C}_i^{(*)} \cap Y_k = \emptyset$, une nouvelle étiquette est donnée au cluster. Les deux principaux avantages de cette approche sont qu'elle permet :

1. Qu'une classe ne corresponde pas nécessairement qu'à un seul cluster car plusieurs clusters peuvent obtenir la même étiquette à l'issue de l'étape de vote ;
2. Que certains clusters ne soient pas étiquetés.

Le premier avantage permet de découvrir des classes composées de plusieurs clusters dans l'espace des données. Ceci est de plus en plus fréquent avec l'augmentation de la masse de données et de la granularité des classes. Le second avantage permet quant à lui de laisser un degré de liberté à la méthode pour découvrir de nouvelles classes [Forestier et al., 2012]. Une étape manuelle est nécessaire où les clusters non étiquetés sont présentés à l'expert.

2.3.2 Guider le processus par les connaissances

Dans la méthode collaborative, un certain nombre de critères sont utilisés pour comparer les résultats et évaluer leur qualité,otamment utilisés lors de la phase de résolution de conflits entre deux résultats. Le coefficient local d'évaluation γ permet comparer la similarité et la qualité d'un couple de résultats de clustering : le coefficient local d'évaluation $\gamma^{(i,j)}$ évalue la similarité et la qualité de deux résultats de clustering $\mathcal{C}^{(i)}$ et $\mathcal{C}^{(j)}$. Ce coefficient inclut un critère de qualité $\delta^{(i)}$ qui évalue la qualité du résultat $\mathcal{C}^{(i)}$. On peutremplacer ce coefficient par un coefficient générique $Q(\mathcal{C}^{(i)})$ permettant d'intégrer différents types de connaissances dans la méthode. Celui-ci peut prendre en compte deux aspects du résultat, sa qualité interne et sa qualité externe :

- L'évaluation interne consiste à évaluer la qualité du clustering à travers une mesure de qualité non-supervisée .
- L'évaluation externe consiste à évaluer la qualité du résultat par rapport à des informations externes fournies par l'expert (par exemple le nombre de clusters, des objets étiquetés, des contraintes, etc.).

2.3.3 Utilisation des connaissances dans les méthodes

Le dernier niveau de connaissance se situe directement au cœur des méthodes utilisées pendant la collaboration. En effet, il est possible de faire collaborer directement des méthodes de clustering semi-supervisées. Il est cependant nécessaire de modifier ces méthodes pour qu'il soit possible d'appliquer les opérateurs de scission, fusion et reclustering nécessaire à la modification des résultats pendant la collaboration. Une étude préliminaire a été effectuée pour étudier comment modifier la méthode KMEANS méthode et ses opérateurs. Cette approche a donné des résultats encourageants mais nécessite une modification trop importante des différentes méthodes et rend donc difficile son application.

Chapitre 3

État de l'art : Apprentissage actif

3.1 Introduction

Les méthodes d'apprentissage actif permettent au modèle d'intéragir avec son environnement en sélectionnant les situations les plus "informatives". Le but est d'entraîner un modèle en utilisant le moins d'exemples possible. La construction de l'ensemble d'apprentissage est réalisée en interaction avec un expert humain de manière à maximiser les progrès du modèle. Le modèle doit être capable de détecter les exemples les plus utiles pour son apprentissage et de demander à l'oracle : "Que faut-il faire dans ces situations ?". L'apprentissage actif a pour but de détecter les exemples les plus "instructifs" pour les étiqueter, puis de les incorporer à l'ensemble d'apprentissage. Castro et al. [2005] distingue deux scénarii possibles : l'échantillonnage adaptatif et l'échantillonnage sélectif (voir figure 3.1). Il s'agit de deux manières différentes de poser le problème de l'apprentissage actif.

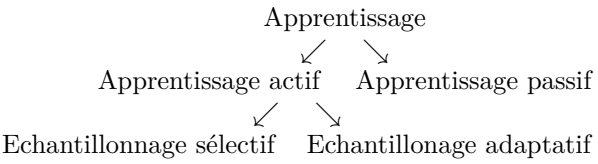


FIGURE 3.1 – Echantillonnage

- Dans le cas de l'échantillonnage adaptatif [Singh et al., 2006], le modèle demande à l'oracle des étiquettes correspondant à des vecteurs de descripteurs. Le modèle n'est pas restreint et peut explorer tout l'espace de variation des descripteurs, à la recherche de zones à échantillonner plus finement. Dans certains cas l'échantillonnage adaptatif peut poser problème lors de sa mise en oeuvre. En effet, il est difficile de savoir si les vecteurs de descripteurs générés par le modèle ont toujours un sens vis à vis du problème initial.
- Dans le cas de l'échantillonnage sélectif [Roy and McCallum, 2001], le modèle n'observe qu'une partie restreinte de l'univers matérialisée par des exemples d'apprentissage démunis d'étiquette. Par conséquent, les vecteurs d'entrées sélectionnés par le modèle correspondent toujours à une donnée brute. On emploie généralement l'image d'un "sac" d'instances pour lesquelles le modèle peut demander les labels associés.

3.2 Méthodes d'apprentissage actif

Le problème de l'échantillonnage sélectif a été posé formellement par Muslea [2002]. Celui-ci met en jeu une fonction d'utilité, $Utile(u, M)$, qui estime l'intérêt d'une instance u pour l'apprentissage du modèle M. Grâce à cette fonction, le modèle présente à l'oracle les instances pour lesquelles il espère la plus grande amélioration de ses performances. Cette fonction $Utile(u, M)$ doit être définie pour exprimer une stratégie d'apprentissage actif particulière. Le problème est donc de

savoir comment "préjuger" efficacement de l'intérêt d'un exemple pour l'apprentissage d'un modèle, avant même que celui-ci n'ait été étiqueté

3.2.1 Echantillonnage par incertitude

L'incertitude d'une prédiction peut être définie par exemple, rapport à un seuil de décision. Prenons l'exemple d'un réseau de neurones qui ne possède qu'un seul neurone de sortie et qui est utilisé dans le cadre d'une classification binaire. Supposons que la sortie du réseau de neurones soit une valeur continue comprise entre 0 et 1. Un seuil de décision est défini pour faire correspondre une sortie à une des deux classes. Plus la sortie du réseau de neurones est proche du seuil de décision, plus la décision sera considérée comme incertaine. La séparatrice correspondant au seuil de décision du modèle est tracée ainsi que des lignes de niveaux relatifs aux valeurs de sortie du modèle. Les données non-étiquetées qui se situent à proximité de la séparatrice (au sens des lignes de niveaux) sont considérées comme étant les plus incertaines, elles seront donc choisies pour être étiquetées par l'oracle.

Cette première approche a l'avantage d'être intuitive, facile à mettre en oeuvre et rapide. L'échantilonnage par incertitude montre cependant ses limites lorsque le problème à résoudre n'est pas séparable par le modèle. En effet, cette stratégie aura tendance à sélectionner les exemples à étiqueter dans des zones de mélange, où il n'y a peut être plus rien à apprendre.

3.2.2 Echantillonnage par comité de modèles

L'échantillonnage par comité de modèles (ou "Query-by-Committee") est une stratégie qui vise à réduire l'espace des versions3 d'un problème d'apprentissage [Dima and Hebert, 2005]. L'espace des versions $S \in H$ est l'ensemble des hypothèses consistantes avec les données d'apprentissage étiquetées, cet espace représente les informations contenues dans l'ensemble d'apprentissage.
L'échantillonnage par comité génère une multitude de modèles d'apprentissage qui sont entraînés en parallèle sur les mêmes données. En considérant que chacun de ces modèles trouvent une hypothèse consistante $h \in S$, on dispose alors d'un "échantillon" d'hypothèses qu'on espère représentatif de l'espace des versions. On cherche à mesurer le désaccord au sein du comité de modèles lors de la prédiction du label des exemples non étiquetés. Les exemples qui suscitent le plus grand désaccord sont ceux qui ont la plus forte probabilité de réduire S une fois étiquetés.

Néanmoins, dans la pratique, on n'a pas la garantie que le comité de modèles échantillonne correctement l'espace de versions. En effet, il peut être impossible pour les modèles de trouver des hypothèses consistantes avec les données d'apprentissage

3.2.3 Arbre sur l'espace des versions

Contrairement à l'échantillonnage par comité , certains modèles d'apprentissage permettent de manipuler directement l'espace des versions (sans avoir recourt à un comité de modèles). On citera à titre d'exemple la stratégie qui consiste à choisir les exemples à étiqueter dans la marge d'un SVM [Tong and Koller, 2000] ou celle des SG-net [Cohn et al., 1994] qui est basée sur les réseaux de neurones. Il existe de nombreuses autres approches d'apprentissage actif basées sur la réduction de l'espace des versions, notamment celles qui sont dépendantes du modèle.

L'apprentissage actif peut être vu ici comme le parcours d'un arbre de décision construit sur l'espace des versions du modèle. Le noeud principal de l'arbre correspond au premier exemple qu'on soumet à l'oracle. Selon l'étiquette attribuée à cet exemple, on se place sur une des branches issues de ce noeud. Cela à pour effet de désigner le prochain noeud à parcourir et donc le prochain exemple à étiqueter. Dans le cadre de cette approche, toutes les décisions sont prises lors de la construction de l'arbre de décision.

3.3 Conclusion

Plusieurs questions peuvent être soulevées et devront être traitées en partie dans le projet

- Évaluation des stratégies d'apprentissage actif : La qualité d'une stratégie active est généralement représentée par une courbe mesurant la performance du modèle en fonction du nombre d'exemples étiquetés. Le critère de performance utilisé peut prendre plusieurs formes selon le problème traité. Ce type de courbe permet uniquement de comparer les stratégies entre elles de manière ponctuelle, c'est-à-dire pour un nombre d'exemples fixé. Si on observe des courbes qui se croisent, il est impossible de déterminer si une stratégie est meilleure qu'une autre (sur la totalité du jeu de données). Une piste pour résoudre ce problème serait de mesurer l'apport d'une stratégie active par rapport à la stratégie alléatoire et d'intégrer ce nouveau critère sur la totalité du jeu de données.
- Ensemble de test : les méthodes d'apprentissage actif sont généralement utilisées dans le cas où l'acquisition des données est coûteuse. Dans la pratique, on ne dispose pas d'ensemble de test et il est difficile d'évaluer la qualité du modèle au cours de son apprentissage.
- Critères d'arrêt d'un apprentissage actif : On peut soit définir un nombremaximal d'exemples à étiqueter, soit chercher un critère plus fin qui n'ajoute des exemples que si les progrès du modèle sont avérés. Des travaux futurs seront réalisés dans ce sens.

D'une manière générale, les stratégies d'apprentissage actif permettent d'estimer l'utilité des exemples d'apprentissage. Ces mêmes critères pourraient être utilisés dans le cadre d'un apprentissage en ligne. L'ensemble d'apprentissage serait constitué des N exemples les plus "utiles" vus jusqu'à présent (avec N fixé). Cette approche permettrait de considérer des problèmes d'apprentissage non stationnaires et donc d'effectuer un apprentissage qui s'adapte aux variations du système observé.

Chapitre 4

Etudes en cours

À suite à la rédaction de ces états de l'art, de nombreux travaux ont été effectués.

4.1 Classification collaborative multi-échelle

Le processus de classification collaboratif SAMARAHa été étendu à l'ensemble du processus d'extraction, donnant naissance à une collaboration "multi-étapes" permettant d'utiliser toutes les images d'une scène à un moment donnée quelque soient leurs résolutions. Cela consiste à créer, remettre en cause et modifier des résultats d'une segmentation en fonction à la fois des résultats de classification et de segmentation à une résolution différente. Pour cela, un processus itératif dont chaque étape est composée d'une segmentation suivie d'une classification a été défini. La classification porte sur les segments proposés par la segmentation et sert de base à la segmentation de l'étape suivante. L'originalité forte est que contrairement aux méthodes existantes de segmentation multirésolution, la méthode proposée part de la résolution la plus grossière pour finir par la plus fine : approche descendante.

Ces travaux ont donné lieu entre autres à une publication dans une revue internationale portant sur la méthode elle-même

- C. Kurtz, N. Passat, P. Gançarski, A. Puissant : "Extraction of complex patterns from multiresolution remote sensing images : A hierarchical top-down methodology", Pattern Recognition

Une deuxième publication portant sur son application directe à la segmentation multi-echelle d'images d'érosion est en cours de soumission.

4.2 Active learning

Les glissements de terrains sont un risque naturel très répandu et très dynamique à l'échelle temporelle et spatiale. Les études ont permis le développement et application des algorithmes apprentissage automatique et vision par ordinateur pour assister la cartographie et surveillance des glissements de terrain aux différentes échelles temporelles et spatiales :

1. Développement d'une méthode supervisée et itérative ("Active Learning") pour la cartographie des glissements à partir des images satellite de très haute résolution.
2. Développement et évaluation des méthodes d'extraction des caractéristiques pour la détection des fissures superficielles à partir des images aériennes multi-temporelles.
3. Evaluation des diffèrents méthodes "Structure From Motion (SFM)" pour la reconstruction 3D d'un glissement active à partir de photographie terrestre
4. Analyse des incertitudes des cartographies manuelles et de leur influence sur l'évaluation des approches automatiques

Ces travaux ont été productifs et ont donné lieu à de nombreuses publications dont :

- Stumpf, A., Malet, J.-P., Kerle, N., Niethammer, U., Rothmund, S., Image-based mapping of surface fissures for the investigation of landslide dynamics. Geomorphology 186, 12-27.

- Stumpf, A., Lachiche, N., Malet, J.P., Kerle, N, Puissant, A. (in press) : Active learning in the spatial domain for remote sensing image classification. IEEE Transactions on Geoscience and Remote Sensing
- Lampert, T. A. Stumpf, A., and P. Gancarski, An Empirical Study into Annotator Agreement, Ground Truth Estimation, and Algorithm Evaluation, International Journal of Computer Vision, under review

4.3 Approches collaboratives multi-experts

L'évaluation des algorithmes de détection est une étape de recherche qui est bien trop souvent négligée. Les évaluations classiques utilisent une seule vérité-terrain, et des mesures qui sont biaisées par des hypothèses erronées sur les données. Par exemple, les courbes ROC sont normalisées dans chaque classe (dans ce cas, les classes positives et négatives) et font donc l'hypothèse que ces classes seront produites également par le processus de génération de données (formation de fissures et occurrence d'un glissement de terrain par exemple). La précision suppose prend en compte un ratio des exemples positifs et négatifs qui dépend de l'ensemble de test des données, ce qui peut souvent être mal pris en compte : cas de sur ou sous-échantillonnage, par exemple. Nous avons montré que le rang d'un algorithme par rapport à un autre peut changer lorsque ce ratio change et que de plus, le biais en résultant n'est pas aussi déterminable aussi facilement comme on le suppose souvent. C'est pourquoi nous avons proposé plusieurs mesures et d'outils pour faire face à ceci. Par exemple, pour :

- jouer sur le calcul de la précision elle-même
- trouver le point dans lequel le rang de deux algorithmes s'inverse ;
- mesurer la précision en tenant compte de l'évolution temporelle des données
- etc.

Ces mesures ont été validées en comptant le nombre de vrais/faux négatifs et de vrai/faux positifs par rapport à plusieurs verités-terrain. Ainsi, nous avons mené une étude sur le calcul de ces mesures lorsque l'on dispose de plusieurs vérités de terrain et mis en évidence plusieurs faits intéressants tels que : le rang d'un algorithme dépend de l'annotation utilisée ou de la méthode de combinaison l'accord entre plusieurs annotateurs est très faible. De fait, les méthodes couramment utilisées pour combiner les annotations (vote par exemple) ne font que souligner les caractéristiques les plus évidentes dans les données (cela a aussi des conséquences sur les algorithmes d'apprentissage supervisés) ; la variance de la performance de l'algorithme est directement proportionnelle à la variance des annotations. Nous avons montré ceci pour quatre ensemble de données : deux jeux géographiques (glissements de terrain et l'imagerie de fissures), un jeu de données médicales et enfin, l'ensemble des données de segmentation Berkeley afin de vérifier la généralité de la méthode proposée. Nous avons également proposé une méthodologie pour donner une limite sur les performances au lieu d'un indice de performance brute, qui reflète mieux le caractère incertain de vérités terrain dans les tâches de détection. Ces travaux, en cours de finalisation sont très prometteurs et ont donné lieu à une publication en conférence et deux soumisisons dans une revue internationale :

1. T. Lampert, P. Gançarski, The Bane of Skew : Uncertain Ranks and Unrepresentative Precision, Machine Learning, submitted, 2013.
2. T. Lampert, A. Stumpf, P. Gançarski, An Empirical Study into Annotator Agreement, Ground Truth Estimation, and Algorithm Evaluation', International Journal of Computer Vision, submitted, 2013.
3. A. Stumpf, T. Lampert, J.-P. Malet and N. Kerle, 'Multi-scale Line Detection for Landslide Fissure Mapping', In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium, pp. 5450–5453, Munich, Germany, July 22–27, 2012.

Bibliographie

B. Adryan and R. Schuh. Gene-ontology-based clustering of gene expression data. *Bioinformatics*, 20(16) :2851–2852, 2004. ISSN 1367-4803.

C. Aggarwal and P. S. Yu. *Privacy-Preserving Data Mining Models and Algorithms*, volume 34 of *Advances in Database Systems*. Springer, 2008.

S. Anand, D. A. Bell, and J. G. Hughes. The role of domain knowledge in data mining. In *CIKM '95 Proceedings of the fourth international conference on Information and knowledge management*, pages 37–43, New York, NY, USA, 1995. ACM. ISBN 0-89791-812-6.

H. Ayad and M. S. Kamel. Cumulative voting consensus method for partitions with variable number of clusters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(1) :160–173, 2008.

E. Bae and J. Bailey. Coala a novel approach for the extraction of an alternate clustering of high quality and high dissimilarity. *Data Mining, IEEE International Conference on*, 0 :53–62, 2006. ISSN 1550-4786.

S. Basu, A. Banerjee, and R. J. Mooney. Semi-supervised clustering by seeding. In *International Conference on Machine Learning*, pages 19–26, 2002.

S. Basu, M. Bilenko, and R. J. Mooney. A probabilistic framework for semi-supervised clustering. In *International Conference on Knowledge Discovery and Data Mining*, pages 59–68, 2004.

M. Bilenko, S. Basu, and R. J. Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *International Conference on Machine Learning*, pages 81–88, 2004.

A. Blansché. *Sélection automatique d'attributs et combinaison de classifieurs pour des données complexes*. PhD thesis, Université Louis Pasteur, Strasbourg, 2006.

C. Bohm and C. Plant. Hissclu a hierarchical density-based method for semi-supervised clustering. 2008.

A. Bouchachia and W. Pedrycz. Data clustering with partial supervision. *Data Mining and Knowledge Discovery*, 12(1) :47–78, 2006.

P. S. Bradley, K. P. Bennett, and A. Demiriz. Constrained k-means clustering. Technical report, MSR-TR-2000-65, Microsoft Research, 2000.

T. D. Breaux and J. W. Reed. Using ontology in hierarchical information clustering. *Hawaii International Conference on System Sciences*, 4 :111b, 2005. ISSN 1530-1605.

L. Breiman. Bagging predictors. *Mach. Learn.*, 24(2) :123–140, 1996. ISSN 0885-6125.

M. Brut, F. Sedes, R. Grigoras, and V. Charvillat. An ontology based approach for providing multimedia personalised recommendations. *International Journal Web Grid Services*, 4(3) :314–329, 2008. ISSN 1741-1106.

R. Castro, R. Willett, and R. Nowak. Faster rate in regression via active learning. In *NIPS (Neural Information Processing Systems), Vancouver.*, 2005.

G. Cleuziou, M. Exbrayat, L. Martin, and S. J.-H. Cofkm a centralized method for multiple-view clustering. In *IEEE International Conference on Data Mining*, pages 752–757, 2009.

D. Cohn, R. Caruana, and A. Mccallum. Semi-supervised clustering with user feedback. Technical report, Cornell University, 2003.

D. A. Cohn, L. Atlas, and R. E. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2) :201–221, 1994.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., Hoboken, NJ, USA, 2 edition, 2006.

I. Davidson and S. Ravi. Clustering under constraints feasibility issues and the k-means algorithm. In *SIAM Data Mining Conference*, 2005.

I. Davidson, K. L. Wagstaff, and S. Basu. Measuring constraint-set utility for partitional clustering algorithms. In *European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 115–126, 2006.

A. Demiriz, K. Bennett, and M. Embrechts. Semi-supervised clustering using genetic algorithms. pages 809–814, 1999.

I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 269–274, 2001.

C. Dima and M. Hebert. Active learning for outdoor obstacle detection. In *Proceedings of Robotics : Science and Systems*. Cambridge, 2005.

E. Dimitriadou, A. Weingessel, and K. Hornik. A combination scheme for fuzzy clustering. *International Journal of Pattern Recognition and Artificial Intelligence*, 16(7) :901–912, 2002.

F. Duarte, A. Fred, A. Lourenco, and M. Rodrigues. Weighting cluster ensembles in evidence accumulation clustering. *Portuguese Conference on Artificial Intelligence*, pages 159–167, 2005.

J. C. Dunn. Well separated clusters and optimal fuzzy-partitions. *Journal of Cybernetics*, 4 : 95–104, 1974.

C. F. Eick, N. Zeidat, , and Z. Zhao. Supervised clustering - algorithms and benefits. In *International Conference on Tools with Artificial Intelligence*, pages 774–776, 2004.

K. Faceli, A. de Carvalho, and M. de Souto. Multi-objective clustering ensemble. *International Conference on Hybrid Intelligent Systems*, pages 51–51, 2006.

K. Faceli, A. C. P. F. de Carvalho, and M. C. P. de Souto. Multi-objective clustering ensemble with prior knowledge. volume 4643, pages 34–45. Springer, 2007.

X. Z. Fern and C. E. Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *ICML '04 Proceedings of the twenty-first international conference on Machine learning*, page 36, New York, NY, USA, 2004. ACM. ISBN 1-58113-828-5.

G. Forestier, A. Puissant, C. Wemmert, and P. Gançarski. Knowledge-based region labeling for remote sensing image interpretation. *Computers, Environment and Urban Systems*, 36(5) :470–480, Dec 2012. doi : 10.1016/j.compenvurbsys.2012.01.003. URL `http://icube-intranet.unistra.fr/papr/2012/2-FPWG12`.

A. Fred and A. Jain. Combining multiple clusterings using evidence accumulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6) :835–850, 2005.

A. L. N. Fred. Finding consistent clusters in data partitions. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems*, Lecture Notes in Computer Science, pages 309–318. Springer, 2001.

J. Gao, P. Tan, and H. Cheng. Semi-supervised clustering with partial background information. In *SIAM International Conference on Data Mining*, 2006.

D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

N. Grozavu and Y. Bennani. Classification collaborative non supervisée. In *Conférence francophone sur l'apprentissage automatique (CAP)*, 2010.

T. Gruber. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human Computer Studies*, 43(5/6) :907–928, 1995.

S. T. Hadjitodorov and L. I. Kuncheva. Selecting diversifying heuristics for cluster ensembles. In M. Haindl, J. Kittler, and F. Roli, editors, *Multiple Classifier Systems*, volume 4472 of *Lecture Notes in Computer Science*, pages 200–209. Springer, 2007.

S. T. Hadjitodorov, L. I. Kuncheva, and L. P. Todorova. Moderate diversity for better cluster ensembles. *Information Fusion*, 7(3) :264–275, 2006.

J. Handl and J. Knowles. An evolutionary approach to multiobjective clustering. *IEEE Transactions on Evolutionary Computation*, 11(1) :56–76, 2007.

J. Handl and J. D. Knowles. On semi-supervised clustering via multiobjective optimization. In *Genetic and Evolutionary Computation Conference*, pages 1465–1472, 2006.

Z. He, X. Xu, and S. Deng. A cluster ensemble method for clustering categorical data. *Information Fusion*, 6 :143–151, 2005.

A. Hotho, A. Maedche, , and S. Staab. Ontology-based text clustering. 2001.

T. Hu, Y. Yu, J. Xiong, and S. Y. Sung. Maximum likelihood combination of multiple clusterings. *Pattern Recognition Letters*, 27(13) :1457–1464, 2006.

A. K. Jain. Data clustering 50 years beyond k-means. *Pattern Recognition Letters*, 31(8) :651–666, 2009.

X. Jia and J. Richards. Fast k-nn classification using the cluster-space approach. *Geoscience and Remote Sensing Letters, IEEE*, 2(2) :225–228, 2005.

G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar. Multilevel hypergraph partitioning application in vlsi domain. In *ACM IEEE Design Automation Conference*, pages 526–529, 1997.

J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas. On combining classifiers. *IEEE Transaction Pattern Analysis Machine Intelligence*, 20(3) :226–239, 1998.

D. Klein, S. Kamvar, and C. Manning. From instance-level constraints to space-level constraints making the most of prior knowledge in data clustering. In *In The Nineteenth International Conference on Machine Learning*, pages 307–314, 2002.

A. Konak, D. Coit, and A. Smith. Multi-objective optimization using genetic algorithms a tutorial. *Reliability Engineering & System Safety*, 91(9) :992–1007, 2006.

I. Kopanas, N. M. Avouris, and S. Daskalaki. The role of domain knowledge in a large scale data mining project. In *Methods and Applications of Artificial Intelligence Lecture Notes in Artificial Intelligence*, pages 288–299. Springer-Verlag, 2002.

A. Kreiger and P. Green. A generalized rand-index method for consensus clustering of separate partitions of the same data base. *Journal of Classification*, 16 :63–89, 1999.

N. Kumar and K. Kummamuru. Semisupervised clustering with metric learning using relative comparisons. *IEEE Transactions on Knowledge and Data Engineering*, 20(4) :496–503, 2008. ISSN 1041-4347.

L. I. Kuncheva. Classifier ensembles facts, fiction, faults and future, 2008.

M. Law, A. Topchy, and A. Jain. Multiobjective data clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 424–430, 2004.

V. Loia, W. Pedrycz, and S. Senatore. Semantic web content analysis a study in proximity-based collaborative clustering. *Fuzzy Systems, IEEE Transactions on*, 15(6) :1294–1312, 2007.

B. Long, Z. M. Zhang, and P. S. Yu. Combining multiple clusterings by soft correspondence. In *International Conference on Data Mining*, pages 282–289. IEEE Computer Society, 2005.

N. Megiddo and K. J. Supowit. On the complexity of some common geometric location problems. *SIAM Journal on Computing*, 13(1) :182–196, 1984.

S. Merugu and J. Ghosh. Privacy-preserving distributed clustering using generative models. In *IEEE International Conference on Data Mining*, pages 211–218. IEEE Computer Society, 2003.

B. Minaei-Bidgoli, A. Topchy, and W. Punch. Ensembles of partitions via data resampling. *International Conference on Information Technology Coding and Computing*, 2 :188–192, April 2004a.

B. Minaei-Bidgoli, A. P. Topchy, and W. F. Punch. A comparison of resampling methods for clustering ensembles. In *International Conference on Artificial Intelligence*, pages 939–945, 2004b.

H. Mitra, Banka, and W. Pedrycz. Rough-fuzzy collaborative clustering. *IEEE Transactions on Systems, Man, and Cybernetics*, 36 :795–805, 2006.

I. Muslea. *Active Learning With Multiple View*. PhD thesis, University of southern california, 2002.

N. Nguyen and R. Caruana. Consensus clusterings. In *International Conference on Data Mining*, pages 607–612. IEEE Computer Society, 2007.

J. V. d. Oliveira and W. Pedrycz. *Advances in Fuzzy Clustering and its Applications*. John Wiley & Sons, Inc., New York, NY, USA, 2007.

W. Pedrycz. Collaborative fuzzy clustering. *Pattern Recognition Letters*, 23 :1675–1686, 2002.

W. Pedrycz. Fuzzy clustering with a knowledge-based guidance. *Pattern Recognition Letters*, 25 (4) :469–480, 2004.

W. Pedrycz and P. Rai. A multifaceted perspective at data analysis a study in collaborative intelligent agents. *IEEE Transactions on Systems, Man, and Cybernetics*, 38(4) :1062–1072, 2008.

K. Punera and J. Ghosh. *Advances in Fuzzy Clustering and its Applications*. Wiley-Interscience, 2007.

Z. Qi and I. Davidson. A principled and flexible framework for finding alternative clusterings. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 717–726, 2009.

N. Roy and A. McCallum. Toward optimal active learning through sampling estimation of error reduction. In C. Morgan Kaufmann, San Francisco, editor, *Conf. on Machine Learning*, pages 441–448, 2001.

C. Ruiz, M. Spiliopoulou, and E. Menasalvas. Density-based semi-supervised clustering. *Data Mining and Knowledge Discovery*, 2009.

R. E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2) :197–227, 1990. ISSN 0885-6125.

A. Singh, R. Nowak, and P. Ramanathan. Active learning for adaptive mobile sensing networks. In A. Press, editor, *Fifth international conference on Information processing in sensor networks (IPSN '06)*, pages 60–68, 2006.

A. Strehl and J. Ghosh. *Journal on Machine Learning Research*, 3 :583–617, 2002.

S. Tong and D. Koller. Support vector machine active learning with applications to text classification. In *17th International Conference on Machine Learning (ICML'00)*, pages 999–1006. P. Langley (Ed.), Morgan Kaufmann Publishers, San Francisco, US, 2000.

A. Topchy, A. Jain, and W. Punch. Clustering ensembles models of consensus and weak partitions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12) :1866–1881, 2005.

A. P. Topchy, A. K. Jain, and W. F. Punch. Combining multiple weak clusterings. In *IEEE International Conference on Data Mining*, pages 331–338, 2003.

K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl. Constrained k-means clustering with background knowledge. In *International Conference on Machine Learning*, pages 557–584, 2001.

K. L. Wagstaff. Value, cost, and sharing open issues in constrained clustering. In *International Workshop on Knowledge Discovery in Inductive Databases*, pages 1–10, 2007.

C. Wang, W. Chen, P. Yin, and J. Wang. Semi-supervised clustering using incomplete prior knowledge. In *Computational Science – ICCS 2007*, pages 192–195, 2007.

C. Wemmert. *Classification hybride distribuée par collaboration de méthodes non supervisées*. Thèse de doctorat, Université de Strasbourg, 2000.

C. Wemmert and P. Gancarski. A multi-view voting method to combine unsupervised classifications. In *Artificial Intelligence and Applications*, pages 447–452, Malaga, Spain, 2002a.

C. Wemmert and P. Gancarski. A multi-view voting method to combine unsupervised classifications. In *Artificial Intelligence and Applications*, pages 447–452, Malaga, Spain, september 2002b.

D. H. Wolpert. Stacked generalization. *Neural Network*, 5(2) :241–259, 1992. ISSN 0893-6080.

Z.-H. Zhou and W. Tang. Clusterer ensemble. *Knowledge-Based Systems*, 19(1) :77–83, 2006.