Clustering incrémental et méthodes de détection de nouveauté : application à l'analyse intelligente d'informations évoluant au cours du temps

Pascal Cuxac, Jean-Charles Lamirel

Atelier CIDN'11 : Clustering incrémental et méthodes de détection de nouveauté : application à l'analyse intelligente d'informations évoluant au cours du temps

Pascal CUXAC [1], Jean-Charles LAMIREL [2]
1 INIST-CNRS, 2 allée du Parc de Brabois, CS 10310, 54519 Vandœuvre-lès-Nancy Cedex
2 LORIA-Equipe TALARIS, Campus Scientifique BP 239, 54506 Vandœuvre-lès-Nancy

1 Préface

Le développement de méthodes d'analyse dynamique de l'information, comme le clustering incrémental et les méthodes de détection de nouveauté, devient une préoccupation centrale dans un grand nombre d'applications dont le but principal est de traiter de larges volumes d'information variant au cours du temps. Ces applications se rapportent à des domaines très variés et hautement stratégiques, tels que l'exploration du Web et la recherche d'information, l'analyse du comportement des utilisateurs et les systèmes de recommandation, la veille technologique et scientifique, ou encore, l'analyse de l'information génomique en bioinformatique. La majorité des méthodes d'apprentissage ont été initialement définis d'une façon non incrémentale. Cependant, dans chacune des familles de méthodes, ont été développées des variantes incrémentales permettant de prendre en compte la composante temporelle d'un flux de données. D'une manière plus générale les algorithmes de clustering incrémental et les méthodes de détection de nouveauté sont soumis aux contraintes suivantes :

Possibilité d'être appliqués sans connaître au préalable toutes les données à analyser ;

Prise en compte d'une nouvelle donnée sans faire un usage intensif des données déjà prises en considération ;

Résultat disponible après l'insertion de toute nouvelle donnée ;

Les changements potentiels de l'espace de description des données doivent pouvoir être pris en considération.

L'objectif de cet atelier est de réunir des chercheurs confirmés, ainsi que des jeunes chercheurs, autour des problématiques et des applications du « clustering incrémental », et de la « détection de nouveautés ».

2 Principaux thèmes (liste non limitative)

- Les algorithmes et techniques de détection de nouveauté ;
- Les méthodes de classification incrémentales basées sur la densité ;
- Les méthodes adaptatives de type K-means ou centres mobile ;
- Les méthodes de classification hiérarchique adaptatives ;
- Les méthodes neuronales adaptatives ;
- Les approches probabilistes ;
- Les méthodes de partitionnement et de classification incrémentale basées sur les graphes attribués ;
- Les approches basées sur l'intelligence en essaim et les algorithmes génétiques ;
- Les méthodes de visualisation des résultats d'analyse de données évolutifs

3 Programme

3.1 Conférencier invité

- Hatem HAMZA ; société ITESOFT.
  Apprentissage incrémental dans l'industrie de l'analyse de documents : problèmes et perspectives

3.2 Articles acceptés :

- *Génération de résumés de mise à jour : Utilisation d'un algorithme de classification non supervisée pour détecter la nouveauté dans les articles de presse*
  Aurélien Bossard
  LIPN
- *AD-CLUST: Détection des Anomalies basée sur le Clustering*
  Imen Brahmi, Sadok Ben Yahia, Pascal Poncelet
  Faculté des Sciences de Tunis, LIRMM
- *Score de Fisher et détection de courbes anormales dans une séquence à des fins de diagnostic*
  Etienne Côme, Allou Samé, Patrice Aknin et Marc Antoni
  INRETS, SNCF
- *Comportement comparatif des méthodes de clustering incrémentales et non incrémentales sur les données textuelles hétérogénes*
  Raghvendra Mall, Mumtaz Ahmad, Jean-Charles Lamirel
  CDE IIIT Hyderabad, CASSIS - Loria - INRIA, TALARIS - Loria - INRIA
- *Impact de la taille de l'ensemble d'apprentissage : une étude empirique*
  Christophe Salperwyck, Vincent Lemaire
  Orange Labs, LIFL

4 Summary

The development of dynamic information analysis methods, like incremental clustering and novelty detection techniques, is becoming a central concern in a bunch of applications whose main goal is to deal with information which is varying over time. These applications relate themselves to very various and highly strategic domains, including web mining and adaptive information retrieval, user behaviour analysis and recommendation systems, technological and scientific survey, anomaly or intrusion detection, and even genomic information analysis, in bioinformatics.

The goal of this workshop is to bring together researchers whose main topic is to work on innovative and challenging incremental clustering and novelty detection approaches and on their application to analysis of time varying information of various natures.

Apprentissage incrémental dans l'industrie de l'analyse de documents : problèmes et perspectives

Hatem Hamza.
ITESOFT, Parc d'Andron. 30470 Aimargues

Résumé :

Les logiciels industriels d'analyse de documents permettent de traiter des millions de documents chaque jour. En plus du volume à traiter, la complexité des documents est une difficulté majeure à laquelle ces logiciels doivent répondre. Face à ces deux défis, les approches d'analyse de documents statiques ont prouvé leurs limites et montré plusieurs lacunes. En effet, la plupart des approches industrielles ou de la littérature scientifique ne proposent pas d'apprendre au fur et à mesure du traitement, et se retrouvent généralement bloquées dès qu'un nouveau cas de figure (classe de document jamais rencontrée, une nouvelle information à extraire) se présente. Ainsi, au moindre échec, un opérateur est obligé d'intervenir.
Afin de remédier à cela, les industriels tentent d'avoir des approches incrémentales et dynamiques. Ces approches résolvent plusieurs problèmes comme le fait de s'adapter à de nouvelles classes de documents ou à de nouvelles informations à extraire. Par contre, elles en soulèvent aussi beaucoup d'autres comme le choix des mesures de similarité ou le choix des bons exemples à apprendre en production…
Dans cet exposé, quelques exemples de l'expérience d'ITESOFT dans ce domaine sont présentés et discutés.

Génération de résumés de mise à jour : Utilisation d'un algorithme de classification non supervisée pour détecter la nouveauté dans les articles de presse

Aurélien Bossard

Laboratoire d'Informatique de Paris-Nord
UMR 7030, CNRS et Université Paris 13)
99, av. J.-B. Clment
93430 Villetaneuse
prenom.nom@lipn.univ-paris13.fr

Résumé. Dans cet article, nous présentons un système de résumé automatique multi-documents, dédié au résumé de mise à jour – ou de nouveauté. Dans une première partie, nous présentons la méthode sur laquelle notre système est fondé, CBSEAS, et son adaptation à la tâche de résumé de mise à jour. Générer des résumés de mise à jour est une tâche plus compliquée que de générer des résumés « standard », et nécessite une évaluation spécifique. Nous décrivons ensuite la tâche « Résumé de mise à jour » de TAC 2009, à laquelle nous avons participé afin d'évaluer notre système. Cette campagne d'évaluation internationale nous a permis de confronter notre système à d'autres systèmes de résumé automatique. Finalement, nous présentons et discutons les résultats intéressants obtenus par notre système.

1 Introduction

La recherche en résumé automatique, supportée par des campagnes d'évaluations et une communauté de chercheurs importantes, a connu ces dernières années des progrès rapides tant du point de vue des méthodes employées que des résultats qualitatifs. En effet, ce domaine répond à des besoins forts en recherche et extraction d'information, dûs notamment à l'augmentation des données électroniques consultables. Le domaine du résumé automatique s'étend, et s'intéresse désormais à différents médias comme supports de différents types de résumé. Comme champs d'application, on peut citer les fils d'e-mails, les blogs, les articles scientifiques ou encore les articles de presse ; et comme types de résumé les synthèses d'opinion, les résumés différentiels et les résumés de mise à jour. C'est précisément ce dernier type de résumé auquel nous nous intéressons dans cet article.

Un résumé de mise à jour sur un sujet donné est fondé sur l'hypothèse que le lecteur du résumé a déjà lu certains documents concernant ce sujet. Le résumé de mise à jour doit synthétiser les informations dont l'utilisateur n'a pas encore pris connaissance, donc les informations nouvelles qui apparaissent dans de nouveaux documents.

Dans cet article, nous présentons nos recherches en résumé automatique de mise à jour : nous avons développé un système de résumé automatique « standard », et l'avons adapté au résumé de mise à jour. Ce système est fondé sur une classification automatique de phrases à résumer, qui permet d'augmenter la diversité informationnelle des synthèses générées.

La tâche de génération de résumés de mise à jour a été proposée durant les campagnes d'évaluation DUC[1] 2007, et TAC[2] 2008 et 2009. Nous avons participé à la tâche « *Update Task* » de la campagne TAC 2009. Cette tâche permet d'évaluer deux types différents de résumé : les résumés « standard », guidés par une requête, et les résumés de mise à jour, également guidés par une requête.

Dans cet article, nous nous fondons sur un système de résumé automatique multi-documents, CBSEAS (Bossard, 2009), qui diffère des autres systèmes de résumé par l'identification de sous-thèmes et l'utilisation de la centralité locale à un sous-thème, en plus de la centralité globale, afin de produire de meilleurs résumés. Le regroupement de phrases véhiculant les mêmes informations, et donc leur classification en sous-thèmes, est crucial pour le résumé automatique multi-documents : savoir quelles phrases sont différentes les unes des autres, mais également parmi celles qui sont similaires, détecter la phrase centrale, peut aider à produire des résumer avec une diversité informationnelle importante. Au-delà de la question de la génération du résumé automatique, la question principale à laquelle nous tentons de répondre ici est : comment distinguer les informations nouvelles de celles déjà lues ?

Cet article montre comment les différents aspects de ces deux problématiques sont gérés. Nous présentons également l'évaluation de notre système sur la tâche « Résumé de mise à jour » de TAC 2008 et 2009.

2 État de l'art

Dans cette section, nous présentons un aperçu des méthodes existantes pour le résumé automatique et la gestion de la mise à jour. Ces domaines ont été largement étudiés ; nous limitons donc cet état de l'art aux travaux principaux et à ceux qui ont le plus inspiré notre approche.

2.1 Les approches de résumé automatique multi-documents

Le résumé automatique est étudié depuis le début du traitement des données textuelles. Très vite, les méthodes génératives ont montré leurs limites. Ces approches fortement dépendantes de la langue nécessitent en effet des ressources linguistiques complexes. Récemment, les recherches de Marcu (1998) ont tenté d'analyser la structure rhétorique pour sélectionner des phrases pertinentes, mais cette méthode est toujours limitée à des domaines applicatifs spécifiques.

Depuis les années 1950 (Luhn, 1958), la recherche en résumé automatique s'est concentrée sur l'extraction de phrases importantes – la création d'*extracts* – plutôt que sur la génération d'*abstracts*. Les phrases extraites doivent constituer un texte cohérent, fidèle aux idées/informations exprimées dans les documents d'origine. L'extraction de phrases est généralement réalisée en calculant un score pour chaque phrase des documents à résumer, et en

[1] Document Understanding Conference : http://www-nlpir.nist.gov/projects/duc/index.html
[2] Text Analysis Conference : http://www.nist.gov/tac

extrayant les mieux classées afin de produire un résumé. Le nombre de phrases ou de mots dans le résumé peut être déterminé à l'avance, mais peut également être calculé dynamiquement en utilisant un pourcentage de compression – par exemple 10% des documents d'origine.

Edmundson (1969) a défini des indices textuels qui peuvent être utilisés afin de déterminer l'importance d'une phrase. Il a notamment proposé une liste de mots-clés tels que « *hardly* », « *In conclusion* ». Les indices contiennent également la position des phrases et le nombre de mots qui co-occurrent dans le titre du document. Ces indices sont encore utilisés de nos jours dans la majorité des systèmes de résumés automatiques, comme dans celui de (Kupiec et al., 1995), qui les combine à un algorithme d'apprentissage. Cependant, ceux-ci sont limités puisqu'ils ne prennent pas en compte le contenu global du document.

D'autres systèmes se concentrent sur les fréquences des termes en corpus. Luhn (1958) a ouvert la voie aux systèmes statistiques de résumé par extraction. Il a proposé de construire une liste des termes importants des documents, en se fondant sur leur fréquence. Sont sélectionnés seulement ceux dont la fréquence appartient à un intervalle prédéfini. Plus une phrase présente de mots appartenant à cette liste, plus elle est pertinente. Radev et al. (2002) ont profité des avancées dans le domaine des statistiques textuelles en intégrant le *tf.idf* à la méthode de Luhn. La liste des termes importants, que Radev appelle « centroïde », est composée des *n* termes avec le plus grand *tf.idf*. Les phrases sont classées selon leur similarité au centroïde.

Les méthodes statistiques sont efficaces pour sélectionner les phrases qui reflètent le contenu global des documents à résumer. Une telle phrase est qualifiée de « centrale ». Cependant, ces méthodes ne sont pas conçues de manière à générer des résumés qui reflètent la diversité informationnelle des documents d'origine. La diversité informationnelle est aussi importante que la centralité lorsqu'on évalue la qualité d'un résumé. En effet, un résumé doit contenir toutes les informations importantes.

La méthode MMR – *Maximum Margin Relevance* – Carbonell et Goldstein (1998) cherche à résoudre le problème de la diversité. Les phrases qui maximisent la fonction de score présentée en Équation 1 sont sélectionnées incrémentalement. La fonction de score MMR prend en compte la diversité en soustrayant au score la centralité la similarité maximale de la phrase évaluée avec les phrases déjà sélectionnées. Cette méthode est utilisée largement et a été adaptée à différentes tâches de résumé automatique (Goldstein et al., 2000; Chowdary et Kumar, 2009; Ribeiro et de Matos, 2007; Wang et al., 2009).

$$MMR = \underset{P_i \in D \setminus S}{\operatorname{argmax}} \left[\lambda sim_1(P_i, Q) - (1 - \lambda) \underset{P_j \in S}{\operatorname{argmax}}\ sim_2(P_i, P_j) \right] \tag{1}$$

où Q est la requête utilisateur, D l'ensemble des phrases, S l'ensemble des phrases sélectionnées pour le résumé, et λ le facteur de nouveauté.

Dans le cas particulier du résumé multi-documents, la redondance est un bon indice de l'importance d'un élément d'information. MMR prend en compte la redondance, mais seulement dans le but d'éliminer les phrases redondantes, et non comme un critère d'extraction. Radev (Erkan et Radev, 2004) s'est appuyé sur les avancées récentes dans le domaine des réseaux sociaux afin d'utiliser la redondance comme le principal critère pour juger la pertinence d'une phrase. Il construit un graphe des documents à résumer, dans lequel les nœuds qui ont

le plus grand *prestige* sont ceux qui sont fortement liés à d'autres nœuds ayant eux-mêmes un *prestige* important.

Toutes les méthodes que nous avons présentées dans cet état de l'art considèrent le contenu global des documents à résumer pour évaluer la centralité des phrases. Cependant, nous considérons les documents non comme un tout, mais comme différents groupes de phrases formant des sous-thèmes. Dans chacun de ces sous-thèmes, des phrases centrales émergent, qui sont celles que nous voulons extraire.

2.2 Les approches de résumé de mise à jour

La tâche « Résumé de mise à jour » de DUC 2007 et TAC 2008 a révélé que générer un résumé de mise à jour est une tâche bien plus complexe que de générer des résumés « standard » Dang et Owczarzak (2008). Cette tâche pose en effet, au-delà du problème de la génération d'un résumé automatique, celui de la détection de la nouveauté. Nous présentons ici différentes stratégies visant à gérer le résumé de mise à jour.

Certaines auteurs, comme Galanis et Malakasiotis (2008), retirent des documents de mise à jour toutes les phrases dont la similarité à une phrase des documents initiaux est supérieure à un seuil défini empiriquement. D'autres préfèrent supprimer les phrases qui maximisent la similarité au jeu de documents initial jusqu'à ce que la similarité globale entre ce dernier et le jeu de documents de mise à jour soit en dessous d'un seuil prédéfini (He et al., 2008).

La méthode présentée par Boudin et Torres-Moreno (2008) sélectionne les phrases pour le résumé de mise à jour en utilisant la méthode MMR, décrite en Section 2.1. Le poids de la similarité aux phrases déjà sélectionnées est augmenté afin de réduire le risque d'extraire des phrases qui ne véhiculent pas d'information nouvelle.

Une autre méthode, introduite dans (Varma et al., 2009), vise à évaluer la nouveauté d'un mot. Le facteur de nouveauté (*fn*) d'un mot dans un document publié à une date *t* dépend de son nombre d'occurrences dans les documents antérieurs et dans les documents postérieurs :

$$fn(w) = \frac{|nd_t|}{|pd_t| + |D|} \tag{2}$$

$nd_t = d : w \in d \wedge t_d\, t$
$pd_t = d : w \in d \wedge t_d \leq t$
$D = d : t_d\, t$

Le facteur de nouveauté est utilisé pour mesurer la nouveauté d'une phrase. Cette méthode a prouvé son efficacité, tant sur les évaluations de TAC 2008 que sur celles de TAC 2009. Cependant, nous voulons évaluer une nouvelle méthode fondée sur la similarité entre phrases, qui ne nécessite pas, contrairement aux premières approches présentées, de fixer *a priori* un seuil de similarité.

3 CBSEAS, une approche générique pour le résumé automatique multi-documents

Nous voulons gérer spécifiquement l'aspect multi-documents en considérant la redondance comme le problème principal du résumé multi-documents. En effet, nous considérons les documents à résumer comme constitués de groupes de phrases qui véhiculent la même information. Dans chacun de ces groupes, une phrase peut être considérée centrale. Extraire une phrase dans chaque groupe de phrases peut mener à réduire le risque de voir apparaître de la redondance dans les résumés générés. De plus, extraire la phrase centrale permet de prendre en compte la centralité locale de chaque sous-thème. Enfin, cette modélisation nous autorise à prendre en compte un critère supplémentaire pour extraire les phrases : la centralité d'une phrase vis-à-vis du sous-thème dans lequel elle a été classée.

Notre système implémente cette méthode. La première étape consiste à regrouper les phrases similaires, puis d'extraire une phrase par classe.

3.1 Regroupement de phrases

Cette section décrit la première partie de notre système : le regroupement de phrases. Nous voulons un algorithme de classification flexible, dans lequel nous pouvons aisément adapter le critère de regroupement. *Fast global k-means* apparaît approprié à cet effet : cet algorithme prend en entrée une matrice de similarité ou de distance. Le modèle créé après avoir regroupé les phrases peut être utilisé non seulement afin d'extraire les phrases, mais également à des fins d'ordonnancement des phrases. Ceci sera l'objet de futures publications.

3.1.1 Pré-traitements

Les documents en entrée subissent des pré-traitements avant d'être traités par CBSEAS. Nous présentons ici les différents pré-traitements réalisés.

Étiquetage morpho-syntaxique Les documents sont analysés morpho-syntaxiquement : l'étiquetage morpho-syntaxique est assuré par tree-tagger[3] (Schmid, 1994). Cela permet de prendre en compte les différents types morpho-syntaxiques pendant le calcul de la similarité entre phrases.

Segmentation en phrases Certains auteurs choisissent de travailler sur des petites structures textuelles plutôt que sur des phrases complètes. Ils travaillent donc à l'extraction de groupes de mots syntaxiquement liés, et divisent les phrases en propositions (Marcu, 1998). Extraire des propositions plutôt que des phrases pose le problème de l'identification de telles propositions – bien que l'analyse syntaxique automatique ait récemment connu d'importants progrès – et de leur indépendance. D'autres auteurs extraient des paragraphes entiers dans le but d'augmenter la cohérence linguistique des résumés. Cependant, cela augmente le risque d'extraire des phrases non pertinentes.

Nous avons donc fait le choix d'extraire des phrases entières afin d'éviter de générer des résumés agrammaticaux et d'extraire des phrases non pertinentes.

[3] page web de tree-tagger : http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/

$$sim(s_1, s_2) = \frac{\sum_{mt} weight(mt) \times fsim(s_1, s_2)}{\frac{fsim(s_1, s_2) + gsim(s_1, s_2)}{\sum_{mt} weight(mt)}} \quad (3)$$

$$fsim(s_1, s_2) = \sum_{n_1 \in s_1} \sum_{n_2 \in s_2} tsim(n_1, n_2) \times \frac{tfidf(n_1) + tfidf(n_2)}{2} \quad (4)$$

$$gsim(s_1, s_2) = card\left((n_1 \in s_1, n_2 \in s_2) \mid tsim(n_1, n_2) < \delta\right) \quad (5)$$

where mt are the morphological types, s_1 and s_2 the sentences, tsim the similarity between two terms using WordNet and the JCn similarity measure Jiang et Conrath (1997) and δ a similarity threshold.

Étiquetage en entités nommées L'étiquetage en entités nommées permet de raffiner le calcul de similarité entre phrases. En effet, la reconnaissance de telles entités autorise la prise en compte de groupes lexicaux complexes tels que « le Président George W. Bush » comme une seule et même entité lexicale. Un tel groupe doit en effet être identifié comme une seule entité nommée. Dans le calcul de similarité entre phrases, il sera donc considéré comme un terme unique, et non comme quatre termes distincts. Les entités nommées sont étiquetées par le système ANNIE (Cunningham et al., 2002), développé pour l'architecture GATE.

3.1.2 Calcul de similarité entre phrases

Nous faisons l'hypothèse que la similarité entre phrases doit prendre en compte le type de documents que CBSEAS doit résumer, ainsi que le type de résumé demandé par l'utilisateur. Par exemple, les caractéristiques qui déterminent si deux phrases sont similaires diffèrent selon que l'on cherche à générer un résumé d'opinions ou un résumé d'analyse boursière. Dans le premier cas, les adjectifs, adverbes et verbes de sentiments sont discriminants ; dans le second cas, les catégories discriminantes seront les devises, montants, et noms de compagnie, soit majoritairement des entités nommées.

Nous voulons prendre en compte ce fait en utilisant une mesure de similarité paramétrable, qui peut être aisément adaptée aux différentes tâches auxquelles un système de résumé automatique peut être confronté. Nous voulons également prendre en compte les relations linguistiques entre termes – *e.g.* la synonymie, l'hyperonymie. Nous utilisons pour cela la mesure de similarité JCn (Jiang et Conrath, 1997) qui est fondée sur la distance entre *synsets* dans la taxonomie de WordNet (Fellbaum, 1998). Les Équations 3, 4, 5 présentent cette mesure.

3.1.3 Algorithme de classification

Une fois la matrice de similarité calculée, CBSEAS regroupe automatiquement les phrases similaires. Cette étape est réalisée en utilisant *fast global k-means* (Likas et al., 2001), une variante incrémentale de l'algorithme des k-moyennes (MacQueen, 1967). *Fast global k-means* résoud le problème du choix des *k* centres de classe initiaux posé par l'algorithme des k-moyennes. L'incrémentalité de *fast global k-means* le rend également intéressant dans le but de générer des résumés de mise à jour. Bien que des méthodes de classification ascendante

- p1 : Chess legend **Bobby Fischer**, who faces prison if **Bobby** returns to the **United States**, can only avoid **deportation** from **Japan** if **Iceland** upgrades its granting of **residency** to full citizenship, a **Japanese** lawmaker said Wednesday.
- p2 : **Fischer**, 62, has been detained in **Japan** awaiting **deportation** to the **United States**, where **Bobby** is wanted for violating economic sanctions against the former Yugoslavia by playing a highly publicized chess match there in 1992.
- p3 : **Fischer**, 62, has been detained in **Japan** awaiting **deportation** to the **United States**, where **Bobby** is wanted for violating economic sanctions against the former Yugoslavia by playing a highly publicized chess match there in 1992.
- p4 : But during Thursday's meeting the committee said it was awaiting more documents from **Japan** to be sure **Bobby** would be let go if given citizenship in **Iceland**.
- p5 : Palsson said **Fischer**, 62, would be informed of the vote Tuesday morning in **Japan**, where he has been detained awaiting deportation to the **United States** on a warrant for violating economic sanctions against the former Yugoslavia by playing a chess match there in 1992.
- p6 : **Iceland** recently issued **Fischer** a special passport and **residence** permit to help resolve the standoff over **Bobby**'s status.
- p7 : **Iceland** had previously sent **Fischer** a passport which showed **residency** status but not full nationality, a document not deemed sufficient to save him from **deportation** to the **United States**.
- p8 : **Iceland** had already agreed to issue a **resident**'s permit and special passport to **Fischer**, but that had not proved enough for the **Japanese** authorities.

Source : données de TAC 2009, Tâche « Résumé de mise à jour »

FIG. 1 – *Exemple d'une classe générée par CBSEAS*

hiérarchique aient été testées (et n'ont pas égalé les performances de *fast global k-means*), nous avons jusqu'à présent fourni une part plus importante de travail sur la question du critère de regroupement, donc de la similarité entre phrases (Bossard et Guimier De Neef, 2011). Celle-ci a en effet des conséquences non seulement sur la qualité du regroupement, mais également sur la sélection des phrases (*cf* Section 3.2.1).

Fast global k-means crée d'abord une classe qui contient tous les éléments à classifier. A chaque itération, l'algorithme ajoute une nouvelle classe dont le centre est l'élément le moins représentatif de sa classe. Chaque élément est alors placé dans la classe dont il est le plus proche du centre, et le centre de chaque classe est recalculé. L'algorithme s'arrête lorsque le nombre de classes demandé par l'utilisateur est atteint.

La Figure 1 présente une classe générée par CBSEAS. Les mots partagés par au moins la moitié des phrases de cette classe sont en gras afin d'identifier les raisons de ce regroupement.

3.2 Sélection de phrases

Après avoir regroupé les phrases, CBSEAS extrait une phrase par classe. Rappelons que les phrases ainsi extraites doivent minimiser la redondance, et ainsi produire un résumé avec une bonne diversité informationnelle. Le critère d'extraction est aussi important que le regroupement de phrases. En effet, la méthode utilisée pour déterminer les phrases à extraire influence la centralité du résumé. Nous présentons ici les trois critères utilisés par CBSEAS pour extraire les phrases : centralités globale et locale, et longueur des phrases. Le score final d'une phrase est la somme pondérée de ces trois scores. Les poids sont fixés à l'aide d'un algorithme génétique, détaillé dans (Bossard et Rodrigues, 2011).

3.2.1 Centralité locale

La centralité locale est la pertinence d'une phrase vis-à-vis du contenu de sa classe (ou sous-thème). Nous voulons que les phrases extraites reflètent au mieux les informations de

Liste des informations atomiques (AI) véhiculées par les phrases en Fig. 1 :

Information	**Poids**
Fischer faces deportation	5
Ficher chess player	4
Iceland issued Fischer a resident permit	4
Fischer violated economic sanctions against Yugoslavia	3
Iceland issued Fischer a passport	3
Fischer is 62	3
Fischer has been detained in Japan	3
Fischer faces prison	1
Fischer could avoid deportation	1
Iceland special passport can not avoid him deportation	1
Iceland wants to be sure Fischer would be let go if given citizenship	1

p2, p3 et p5 sont les phrases véhiculant le plus d'informations centrales :

Phrase	p1	p2	p3	p4	p5	p6	p7	p8
Somme des poids des AI	15	21	21	1	21	7	8	7

Similarités entre les phrases :

	p1	p2	p3	p4	p5	p6	p7	p8	somme
p1	1.0	.171	.171	.156	.150	.1	.133	.133	2.014
p2	.171	1.0	1.0	.091	.821	.031	.094	.064	3.272
p3	.171	1.0	1.0	.091	.821	.031	.094	.064	3.272
p4	.156	.091	.091	1.0	.083	.075	.069	.077	1.642
p5	.15	.821	.821	.083	1.0	.027	.108	.055	3.065
p6	.1	.031	.031	.075	.027	1.0	.167	.315	1.746
p7	.133	.094	.094	.069	.108	.167	1.0	.115	1.78
p8	.133	.064	.053	.077	.055	.315	.115	1.0	1.812

Scores de centralité locale tels que définis en Section 3.2.1 :

Phrase	p1	p2	p3	p4	p5	p6	p7	p8
Score	.171	1.0	1.0	.091	.821	.031	.094	.064

FIG. 2 – *Illustration du concept de centralité locale*

leur classe. L'idée derrière la centralité locale est la suivante : les phrases d'une classe C expriment un ensemble d'informations atomiques AI. La phrase la plus centrale de C est celle qui contient les informations les plus importantes de AI. Nous travaillons selon l'hypothèse que la redondance d'une information est corrélée à son importance. La phrase qui maximise la somme des similarités aux autres phrases, P_{max}, est la phrase la plus centrale. Elle reçoit un score de centralité égal à 1. Les autres phrases reçoivent un score égale à leur similarité à P_{max}. La Figure 2 illustre ce calcul.

3.2.2 Centralité globale

Le problème majeur d'une extraction de phrases fondée exclusivement sur la mesure de la centralité locale est la non prise en compte du contenu global des documents ou d'une éventuelle requête utilisateur. Pour générer des résumés précis qui répondent au besoin informationnel exprimé par un utilisateur, ou qui sont centrés sur les thèmes principaux des documents à résumer, nous devons ajouter la mesure de la centralité globale à celle de la centralité locale. Pour cela, nous identifions deux cas :

– l'utilisateur a une requête, le résumé doit donc être en rapport avec celle-ci ;
– l'utilisateur n'a pas de requête, le résumé doit donc être pertinent vis-à-vis du contenu global des documents.

Dans le premier cas, nous utilisons la similarité à la requête comme score de centralité globale. Celui-ci est calculé de la même manière que la similarité entre phrases, présentée en Section 3.1.2. Dans le second cas, nous utilisons le score centroïde, tel que défini dans (Radev et al., 2002).

3.2.3 Longueur des phrases

La longueur des résumés est souvent limitée à un certain nombre de mots. Pour cette raison, nous avons choisi de donner un score à chaque phrase, en fonction de leur longueur, dans le but de pénaliser les phrases trop petites ou trop longues. La fonction de ce score est présentée en Équation 6.

$$score_{longueur} = \frac{1}{e^{(|longueur(phrase) - longueur_{demandee}|)}} \tag{6}$$

4 Générer des résumés de mise à jour

Avec le développement des sites de news en ligne, la détection et le résumé de la nouveauté est devenue une problématique de recherche importante. En effet, les utilisateurs qui suivent un sujet ne veulent pas avoir à lire tout nouvel article publié, mais uniquement les informations dont ils n'ont pas déjà pris connaissance. Le résumé de mise à jour répond à des besoins importants pour l'accès au contenu. De plus , les systèmes de résumé automatique « standard » génèrent des synthèses qui véhiculent un contenu informationnel qualitativement correct. La recherche peut donc se concentrer sur des tâches plus complexes, telles que celles récemment proposées par les campagnes d'évaluation DUC et TAC : résumé d'opinions, résumé de mise à jour (ou de nouveauté), ou résumé thématique. Dans cette section, nous présentons notre méthode pour gérer le résumé de mise à jour.

4.1 Intuitions

CBSEAS – Clustering-Based Sentence Extractor for Automatic Summarization – regroupe automatiquement les phrases les plus similaires. En d'autres termes, il crée différentes classes pour des phrases distantes sémantiquement. Notre méthode de classification peut aussi être utilisée pour classer les phrases en deux groupes :
– celles qui véhiculent des informations connues ;
– celles qui véhiculent des informations nouvelles.

Nous partons donc de l'hypothèse que les phrases des nouveaux documents qui véhiculent des informations anciennes partagent le même vocabulaire (étendu grâce à la similarité JCn) que les phrases que l'utilisateur a déjà lues.

La principale faiblesse d'une telle méthode est le manque de traitements linguistiques dédiés. Par exemple, la phrase *« After hemming and hawing and bobbing and weaving, the board of directors of Fanni Mae finally jettisoned Franklin D. Raines, the mortgage finance giant's former executive, and Timothy Howard, its former chief financial officer. »* issue du corpus

AQUAINT-2[4] est aisément identifiable comme une phrase porteuse de nouveauté du fait du temps employé et de l'emploi de « *finally* ». Dans le cas spécifique des dépêches de presse, cela signifie que l'information vient tout juste de paraître. Cependant, si l'utilisation d'indices linguistiques peut aider à détecter les phrases porteuses de nouveauté, cela limite également la méthode à un langage et un domaine uniques.

De plus, CBSEAS a démontré son efficacité à regrouper des phrases sémantiquement proches et à différencier les phrases éloignées. En effet, CBSEAS s'est classé à la troisième place tous systèmes confondus pour la gestion de la redondance dans les résumés sur la tâche « Résumé d'opinions » de TAC 2008 Bossard et al. (2008). C'est une raison supplémentaire pour utiliser notre méthode de regroupement afin de détecter les phrases porteuses de nouveauté.

4.2 Algorithme de mise à jour

Avant de tenter d'identifier les phrases porteuses de nouveauté, nous devons modéliser les informations que l'utilisateur a déjà lues. Nous pouvons alors confronter les nouveaux documents à ce modèle afin de déterminer si les phrases de ces documents véhiculent des informations nouvelles. La première étape de notre algorithme consiste donc à classifier les phrases issues des documents déjà lus – que nous appelons D_I – en k_I classes, comme décrit en Section 3.1.3.

Le modèle ainsi calculé – M_I – est alors utilisé pour la seconde étape de notre algorithme, qui consiste à déterminer si une phrase des nouveaux documents – D_U – doit être regroupée avec les phrases de D_I à compléter, ou créer une nouvelle classe qui ne contiendra que des phrases porteuses de nouveauté. *Fast global k-means*, moyennant quelques adaptations, peut être utilisé pour confronter des éléments à un modèle précédemment établi dans le but de déterminer si ces éléments peuvent intégrer ce modèle. Nous décrivons ici la partie de notre algorithme dédiée à la détection de nouveauté.

Premièrement, les similarités entre les phrases de D_U et les centres de classe de M_I ainsi qu'entre toutes les phrases de D_U sont calculées. Alors, les phrases de D_U à compléter sont ajoutées à M_I et *fast global k-means* est relancé à partir de la $k_I{}^{\text{ème}}$ itération avec les contraintes suivantes :

- Les phrases de D_I ne peuvent pas être déplacées vers un autre cluster, ceci afin de préserver le modèle M_I qui encode les anciennes informations. Cela évite également de perturber la portée sémantique des nouvelles classes, qui sont porteuses de nouveauté.
- Les centres de classe de M_I ne sont pas recalculés ; étant donné que la portée sémantique d'une classe dépend directement de son centre, cela évite de modifier la portée sémantique des classes de M_I par ajout de nouveaux éléments issus de D_U.

Le principal défaut de cet algorithme, qui est détaillé dans la Figure 3 est le choix de k_I – le nombre de classes de M_I – et celui de k_U – le nombre de classes de M_U. Nous avons décidé empiriquement de fixer k_U au nombre de phrases désiré pour le résumé de mise à jour, et k_I à $\frac{P_I}{P_U} \times k_U$, où P_I et P_U sont respectivement les phrases de D_I et D_U. Pour notre participation à la tâche « Résumé de mise à jour » de TAC 2009, nous avons utilisé un algorithme génétique entraîné sur les données de TAC 2008 afin de décider des meilleures

[4]Le corpus AQUAINT-2 est un sous-ensemble de la troisième édition du corpus anglais LDC Gigaword composé de nouveaux articles issus de différentes agences de presse.

//Classification pour M_I
pour tous les p de P_I
faire
  $cluster(p) \leftarrow C_1$
fin pour
pour i de 1 à k_I
faire
  pour n de 1 à i
  faire
    $centre(C_n) \leftarrow \underset{p_j \in C_n}{\text{argmax}}(\sum_{pm \in C_n} sim(p_j, p_m))$
  fin pour
  pour tous les p de P_I
  faire
    $cluster(p) \leftarrow \underset{C_m, 1<m<u}{\text{argmax}} (sim(centre(C_m), p)$
  fin pour
  si $i < k_I$ alors
    $cluster(\underset{p \in D_I}{\text{argmin}}(sim(p, centre(cluster(p)))) \leftarrow C_{i+1}$
  fin si
fin pour
//Détection de la nouveauté
pour tous les p de P_U
faire
  $cluster(p) \leftarrow argmax_{C_i, 1<i<k_I}(sim(centre(C_i), p))$
fin pour
pour i de k_I à $k_I + k_U$
faire
  pour n de $k_I + 1$ à i
  faire
    $centre(C_n) \leftarrow \text{argmax}_{p_j \in C_n}(\sum_{pm \in C_n} sim(p_j, p_m))$
  fin pour
  pour tous les p dans P_U
  faire
    $cluster(p) \leftarrow \underset{C_m, 1<m<i}{\text{argmax}} (sim(centre(C_m), p))$
  fin pour
  si $i < k_U$ alors
    $cluster(\text{argmin}_{p_m \in D_I}(sim(p_m, centre(cluster(p)))) \leftarrow C_{i+1}$
  fin si
  fin pour

FIG. 3 – *Algorithme de détection de la mise à jour*

valeurs pour ces variables. Aucune de ces solutions n'est idéale, puisqu'elles nécessitent qu'il existe au moins k_U nouvelles informations véhiculées par au moins k_U phrases différentes. D'autres solutions peuvent être envisagées, comme déterminer si ajouter des classes de mise à jour améliore ou détériore la qualité de la classification, en utilisant un indice de validité (Davies et Bouldin, 1979; Calinski et Harabasz, 1974; Beale, 1969).

Une fois les classes peuplées, le résumé de mise à jour est généré en extrayant une phrase par classe de mise à jour, comme décrit en Section 3.2.

5 Évaluation : Participation à TAC

Nous avons évalué notre travail sur la tâche de « Résumé de mise à jour » de la campagne d'évaluation TAC 2009, organisée par le NIST[5]. Nous présentons en détails la tâche, les différentes méthodes d'évaluation, et les résultats obtenus par notre système de résumé de mise à jour.

5.1 Description détaillée de la tâche

Les participants à la tâche de « Résumé de mise à jour » de la campagne d'évaluation de TAC 2009 devaient produire deux types différents de résumé : les résumés « standard » et les « résumés de mise à jour », tous deux guidés par une requête.

La tâche consiste en 44 sujets qui comportent chacun un titre (court), une requête, ainsi que deux jeux de documents : les documents initiaux et les documents de mise à jour. Les systèmes doivent générer deux résumés pour chacun des sujets : un résumé « standard » qui synthétise l'information des documents de mise à jour. Ce dernier résumé doit être produit en tenant compte du fait que son lecteur a déjà pris connaissance du contenu des documents initiaux. Les résumés sont limités à une longueur de 100 mots, quelle que soit la taille des documents d'origine.

Chaque jeu de documents comprend dix documents extraits du corpus ACQUAINT-2 (*cf* Section 4.1). Ces documents sont des dépêches de presse en anglais issues de différentes sources : AFP, NYT, APW, LTW, et Xinhua.

Les requêtes sont en langue anglaise et peuvent être complexes. Si la requête associée au Sujet D0848, présenté dans la Figure 4 est simple, ce n'est pas le cas de toutes, comme celle du sujet D0902 : « *Describe the debate over use of emergency contraceptives, also called the morning-after pill, and whether or not it should be available without a prescription* ».

5.2 Méthodes d'évaluation

Le NIST a utilisé trois méthodes différentes afin d'évaluer les résumés des participants. Les résumés sont évalués par le *package* d'évaluation ROUGE[6] Lin (2004). Les métriques ROUGE sont fondées sur les co-occurrences de n-grammes entre des résumés de référence et les résumés à évaluer. Leur principal avantage réside dans leur fonctionnement entièrement automatique. ROUGE peut donc être utilisé pour des expérimentations en dehors des campagnes d'évaluation. Cependant, l'évaluation de résumés ne peut pas être limitée à des comparaisons de séquences de n-grammes. NIST a donc choisi d'utiliser des méthodes d'évaluation plus précises mais non automatiques.

La seconde méthode utilisée par le NIST est la méthode Pyramide, décrite en détails dans (Nenkova et al., 2007). Les auteurs définissent la notion de SCU – *Summarization Content Unit* – une information qui apparaît dans les résumés. La méthode Pyramide consiste à extraire une liste de SCUs depuis les résumés de référence. Ces SCUs sont alors classés selon leur nombre d'occurrences. Selon les auteurs, ce classement peut être vu comme une pyramide où les informations les plus importantes sont au sommet et les moins importantes à la base. Les

[5]NIST : National Institute of Standards and Technology

[6]ROUGE : Recall-Oriented Understudy for Gisting Evaluation

Sujet D0848 : Airbus A380		Describe developments in the production and launch of the Airbus A380
Documents initiaux		
16/01/2005	AFP	The Airbus A380 : from drawing board to runway-ready in a decade
16/01/2005	AFP	A380 'superjumbo' will be profitable from 2008 : Airbus chief
16/01/2005	APW	Airbus prepares to unveil 1380 « superjumbo », world's biggest passenger jet
17/01/2005	LTW	Can Airports Accomodate the Giant Airbus A380 ?
19/01/2005	AFP	After fanfare, Airbus A380 now must prove it can fly
25/01/2005	AFP	Airbus mulls boosting A380 production capacity
10/04/2005	AFP	While US government moans, airports ready for Airbus giant
27/04/2005	AFP	Paris airport neighbors complain about noise from giant Airbus A380
27/04/2005	NYT	Giant Airbus 380 makes maiden flight
04/05/2005	AFP	Airbus A380 takes off on second test flight
Documents de mise à jour		
01/06/2005	AFP	Airbus announces delay in delivering new superjumbo A380
03/06/2005	AFP	German wing of Airbus denies superjumbo A380 parts delay
05/10/2005	AFP	US aviation officials to study A380 turbulence
15/10/2005	AFP	Airbus says it cannot meet demand for A380 superjumbo
18/10/2005	APW	Second Airbus A380 makes maiden flight
13/11/2005	APW	Airbus executive says company will pay millions in compensation for late A380deliveries
17/02/2006	APW	Airbus sees no delay to A380 after wing ruptured during test
22/02/2006	AFP	Airbus confident of A380 certification
26/03/2006	APW	33 people injured in evacuation frill for A380 super-jumbo
29/03/2006	APW	Airbus A380 superjumbo passes emergency evacuation test

FIG. 4 – *Exemple d'un sujet issu de la tâche « Résumé de mise à jour » de TAC 2009.*

résumés sont finalement évalués en extrayant les SCUs qu'ils contiennent et en les comparant à la pyramide.

L'évaluation Pyramide prend en compte la qualité linguistique des résumés : si une phrase est agrammaticale, elle ne véhicule aucun ou peu de SCUs. Cependant, cette méthode d'évaluation ne prend pas en compte la cohérence globale du résumé. C'est la raison pour laquelle le NIST a introduit des mesures d'évaluation entièrement manuelles. Elles sont décrites dans (Dang et Owczarzak, 2009). Ces mesures manuelles évaluent la « performance générale » (*overall responsiveness sic.*) et la lisibilité. La performance générale mesure la qualité linguistique, et à quel point un résumé répond aux besoins informationnels. Le score de lisibilité reflète la grammaticalité, la non-redondance, la clarté référentielle, le *focus*, la structure et la cohérence. La performance générale et la lisibilité ont été évaluées sur une grille à cinq points :

– 5 : très bon
– 4 : bon
– 3 : acceptable
– 2 : mauvais
– 1 : très mauvais.

Il aurait été intéressant de disposer d'une évaluation des différents critères sur lesquels le score de lisibilité est fondé, comme c'était le cas pour la tâche « Résumé d'opinion » de TAC 2008. Cela nous aurait permis de mieux analyser les résultats de notre système.

5.3 Baselines

Le NIST a fourni trois *baselines* pour la tâche « Résumé de mise à jour » de TAC 2009. La première (notée *Baseline* 1 dans les résultats) est le résultat de l'extraction des premières phrases dans le document le plus récent, jusqu'à ce que la limite de 100 mots soit atteinte. Cette *baseline* fournit une limite inférieure de la qualité que doit atteindre un système de résumé automatique plus probante que la sélection aléatoire de phrases.

La seconde *baseline* (Baseline 2) est générée en ré-ordonnant aléatoirement les phrases d'un résumé de référence. Cela donne un aperçu de l'impact d'un mauvais ordonnancement des phrases sur la qualité linguistique et la performance générale des résumés.

La troisième *baseline* (Baseline 3) est constituée de phrases complètes extraites manuellement. La méthode d'extraction est détaillée dans (Genest et al., 2009). L'idée sous-jacente à cette *baseline* est d'évaluer la limite suéprieure de ce que peut générer un système de résumé automatique, tant du point de vue du contenu que de la qualité linguistique.

5.4 Résultats et discussion

Dans cette section, nous présentons les résultats de notre système, les comparons à ceux des autres participants et les discutons.

La Figure 6 présente les résultats des évaluations Pyramide et Performance générale pour tous les participants. Notre système se classe parmi le premier tiers des participants pour les résumés initiaux, et dans les dix meilleurs systèmes pour les résumés de mise à jour. Le score Performance générale n'est pas aussi bon. Cela est dû à la mauvaise qualité linguistique des résumés générés par notre système. En effet, CBSEAS n'applique aucun des post-traitements communément appliqués, qui pourraient améliorer la cohérence des résumés. La Figure 8 présente les deux résumés générés par CBSEAS pour le Sujet D0911. La dernière phrase est coupée. Cela est dû à la limite de 100 mots imposée par la tâche. CBSEAS ne retire pas automatiquement la phrase qui dépasse cette limite. Cela a également un effet négatif sur le score de Qualité linguistique.

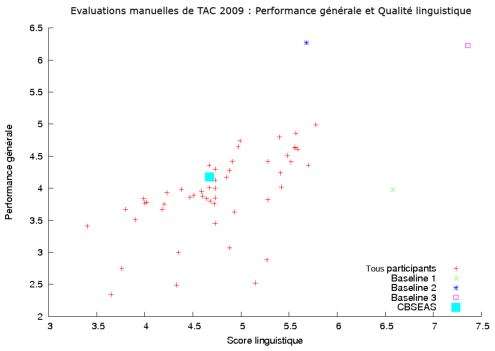


FIG. 5 – *Performance générale et score linguistique des systèmes de TAC 2009*

La Figure 7 présente les différents scores obtenus par CBSEAS, et sa position vis-à-vis des autres systèmes. La qualité linguistique apparaît comme le véritable point faible de notre

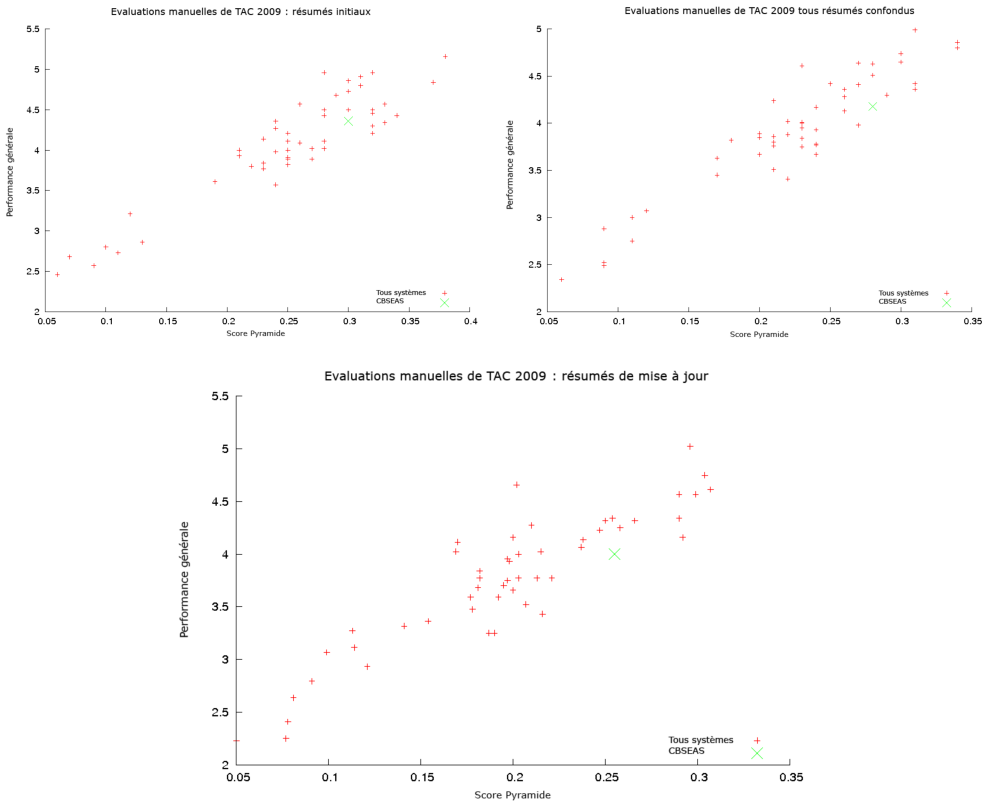


FIG. 6 – *Résultats de la tâche « Résumé et Mise à jour » de TAC 2009 : scores Pyramide et Performance générale*

système. Cependant, cette vue témoigne de l'efficacité de notre stratégie de gestion de la mise à jour. Cinq systèmes sur les quinze qui devancent CBSEAS pour le score Pyramide des résumés initiaux sont moins bien classés que notre système pour le score Pyramide des résumés de mise à jour. Les résumés de mise à jour obtiennent en moyenne des scores Pyramide 21.5% inférieurs à ceux des résumés initiaux. Notre système perd « seulement » 13.5% en score Pyramide – qui évalue le contenu informationnel. D'une manière générale, il a été noté que le résumé de mise à jour est une tâche délicate, et l'on peut constater que les systèmes produisent des résumés initiaux de meilleure qualité que les résumés de mise à jour.

L'évaluation proposée n'est pas complète : les résumés initiaux et de mise à jour sont évalués indépendamment les uns des autres. L'évaluation des résumés de mise à jour aurait pu être poussée plus avant, en évaluant notamment la présence de SCUs dans les résumés de mise à jour qui sont présents dans les documents initiaux. La redondance entre les résumés de mise à jour et le contenu déjà connu de l'utilisation n'est en effet pas évaluée.

Le score linguistique de la *Baseline 2*, présenté en Figure 5, est intéressant : égal à 5.68, il est dépassé par deux systèmes automatiques. Les meilleurs systèmes atteignent des scores

Scores moyens des résumés initiaux et de mise à jour

	ROUGE-2	ROUGE-SU4	Pyr.	Ling.	Perf. gén.
Class. de CBSEAS	9/53	10/53	11/53	31/53	18/53
Score de CBSEAS	0.0919	0.1305	0.28	4.67	4.18
Meilleur score	0.0273	0.0583	0.06	3.40	2.34
Moins bon score	0.1127	0.1452	0.34	5.78	4.99
Score moyen	0.0786	0.1168	0.226	4.751	3.922

Résumés initiaux

	ROUGE-2	ROUGE-SU4	Pyr.	Ling.	Perf. gén.
Class. de CBSEAS	8/53	8/53	15/53	35/53	19/53
Score de CBSEAS	0.1027	0.1338	0.3	4.91	4.3
Moins bon score	0.0282	0.0591	0.06	3.43	2.46
Meilleur score	0.1216	0.1510	0.38	5.93	5.16
Score moyen	0.0853	0.1214	0.252	4.762	4.075

Résumés de mise à jour

	ROUGE-2	ROUGE-SU4	Pyr.	Ling.	Perf. gén.
Class. de CBSEAS	8/53	15/53	10/53	24/53	22/53
Score de CBSEAS	0.0811	0.1223	0.26	4.75	3.98
Moins bon score	0.0264	0.0576	0.05	3.36	2.23
Meilleur score	0.1039	0.1395	0.31	5.89	5.02
Score moyen	0.0719	0.1122	0.198	4.742	3.769

FIG. 7 – *Résultats numériques détaillés de la tâche « Résumé et mise à jour » de TAC 2009*

équivalents à la *Baseline* 3 – constituée de phrases extraites manuellement – pour ce qui est de sélectionner les informations les plus importantes (score Pyramide). Cependant, ces systèmes sont toujours loin derrière cette *baseline* (*cf* Figure 5) pour la qualité linguistique et la performance générale. Cela prouve l'impact du réordonnancement de phrases sur la qualité linguistique, mais également sur la satisfaction d'un utilisateur vis-à-vis d'un résumé, exprimée par le score de performance générale.

La campagne d'évaluation TAC 2009 a montré que notre système est compétitif pour générer des résumés au contenu informationnel important, mais ne produit pas des résumés d'une qualité linguistique à la hauteur de leur contenu. CBSEAS arrive en effet à 83.3 % du score Pyramide de la *Baseline 3* – qui fournit le maximum de ce que pourrait réaliser un système de résumé automatique. La gestion de la mise à jour est très satisfaisante, puisque CBSEAS se classe encore mieux sur cette tâche que sur la tâche de résumé « standard ».

6 Conclusion

Dans cet article, nous avons présenté CBSEAS, un système générique de résumé automatique multi-documents, et un nouvel algorithme dédié à la gestion des résumés de mise à jour – ou de nouveauté. Notre système a obtenu des résultats compétitifs pendant la campagne d'évaluation TAC 2009. Les résultats comparés des résumés « standard » et des « résumés de mise à jour » montrent que notre stratégie de gestion de la mise à jour est efficace. Cependant, elle

pourrait être améliorée en filtrant en amont de notre méthode, les phrases des documents de mise à jour en utilisant une méthode telle que celle décrite dans Varma et al. (2009), fondée sur le facteur de nouveauté des mots. Les résultats mettent aussi en avant la qualité de la méthode de sélection des phrases par CBSEAS. Cependant, notre système manque de post-traitements, ce qui influe négativement sur la satisfaction générale des utilisateurs. Si le réordonnancement de phrases semble être particulièrement influent sur la compréhension du résumé par un utilisateur, l'impact d'autres post-traitements tels que la compression de phrases ou la résolution d'anaphores devraient être évalué dans des travaux à venir.

D0911 Bobby Fischer : résumé initial
Describe efforts to secure asylum in Iceland for chess legend Bobby Fischer.
Chess legend Bobby Fischer was on Monday granted citizenship by the parliament of Iceland, a move which could allow him to avoid deportation from Japan to the United States where he is wanted for violating sanctions against the former Yugoslavia. Chess legent Bobby Fischer, who faces prison if he returns to the United States, can only avoid deportation from Japan if Iceland upgrades its granting of residency to full citizenship. Iceland's parliament voted Monday to grant citizenship to fugitive U.S. chess star Bobby Fischer. Lawmakers in Iceland are likely to grant citizenship to mercurial chess genius Bobby Fischer, a
Score Pyramide : 0.622 Score linguistique : 6

D0911 Bobby Fischer : résumé de mise à jour
Describe efforts to secure asylum in Iceland for chess legend Bobby Fischer.
Iceland said Wednesday it hoped to give detained chess legend Bobby Fischer a passport before the weekend after granting him citizenship in a move that could allow him to avoid a US prison term. An Icelandic supporter of Bobby Fischer said Tuesday he had paid a registration fee that would allow the American chess legend to settle in Iceland. Chess legend Bobby Fischer could leave his Japanese detention cell by the weekend, his supporters said Tuesday, a day after Iceland's parliament voted to grant him citizenship. Japan said Tuesday it may let detained chess legend Bobby Fischer leave for Iceland,
Score Pyramide : 0.345 Score linguistique : 5

FIG. 8 – *Exemple d'un couple de résumés générés par CBSEAS.*

Références

Beale, E. M. L. (1969). Euclidean cluster analysis. *Bulletin of the International Statistical Institute 43*, 92–94.

Bossard, A. (2009). CBSEAS, a new approach to automatic summarization. In *Proceedings of the SIGIR 2009 Conference - Doctoral Consortium*, Boston, USA.

Bossard, A., M. Généreux, et T. Poibeau (2008). Description of the lipn systems at tac2008 : Summarizing information and opinions. In *Notebook papers and results of TAC 2008*, Gaithersburg, Maryland, USA.

Bossard, A. et E. Guimier De Neef (2011). étude de l'impact du regroupement automatique de phrases sur un système de résumé multi-documents. Technical report.

Bossard, A. et C. Rodrigues (2011). Combining a multi-document update summarization system – cbseas – with a genetic algorithm. Smart Innovation, Systems and Technologies. Springer.

Boudin, F. et E.-B. M. Torres-Moreno, Juan-Manuel (2008). A scalable MMR approach to sentence scoring for multi-document update summarization. In *Proceedings of the 2008 COLING Conference*, Manchester, UK, pp. 21–24.

Calinski, R. B. et J. Harabasz (1974). A dendrite method for cluster analysis. *Communications in Statistics 3*, 1–27.

Carbonell, J. et J. Goldstein (1998). The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR '98 : Proceedings of the 21st annual international ACM SIGIR conference*, New York, NY, USA, pp. 335–336. ACM.

Chowdary, C. R. et P. S. Kumar (2009). Esum : An efficient system for query-specific multi-document summarization. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, ECIR '09, Berlin, Heidelberg, pp. 724–728. Springer-Verlag.

Cunningham, H., D. Maynard, K. Bontcheva, et V. Tablan (2002). GATE : A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, Philadelphia, PA, USA.

Dang, H. T. et K. Owczarzak (2008). Overview of the TAC 2008 update summarization task. In *Notebook papers and results of TAC 2008*, Gaithersburg, Maryland, USA, pp. 10–23.

Dang, H. T. et K. Owczarzak (2009). Overview of the TAC 2009 update summarization task. In *Notebook papers and results of TAC 2009*, Gaithersburg, Maryland, USA.

Davies, D. L. et D. W. Bouldin (1979). A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-1*(2), 224–227.

Erkan, G. et D. R. Radev (2004). Lexrank : Graph-based centrality as salience in text summarization. *Journal of Artificial Intelligence Research (JAIR) 22*.

Fellbaum, C. (1998). *WordNet : An Electronic Lexical Database*.

Galanis, D. et P. Malakasiotis (2008). Aueb at tac 2008. In *Notebook papers and results of TAC 2008*, Gaithersburg, Maryland, USA.

Genest, P.-É., G. Lapalme, et M. Yousfi-Monod (2009). Hextac : the creation of a manual extractive run. In *Notebook papers and results of TAC 2009*, Gaithersburg, Maryland, USA.

Goldstein, J., V. Mittal, J. Carbonell, et M. Kantrowitz (2000). Multi-document summarization by sentence extraction. In *NAACL-ANLP 2000 Workshop on Automatic summarization - Volume 4*, Morristown, NJ, USA, pp. 40–48. Association for Computational Linguistics.

He, T., J. Chen, Z. Gui, et F. Li (2008). Ccnu at tac 2008 : Proceeding on using semantic method for automated summarization yield. In *Notebook papers and results of TAC 2008*, Gaithersburg, Maryland, USA.

Jiang, J. J. et D. W. Conrath (1997). Semantic similarity based on corpus statistics and lexical taxonomy. In *International Conference Research on Computational Linguistics (ROCLING X)*.

Kupiec, J., J. Pedersen, et F. Chen (1995). A trainable document summarizer. In *SIGIR '95 : Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, New York, NY, USA, pp. 68–73. ACM.

Likas, A., N. Vlassis, , et J. Verbeek (2001). The global k-means clustering algorithm. *Pattern Recognition 36*, 451–461.

Lin, C.-Y. (2004). Rouge : a package for automatic evaluation of summaries. In *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*, Barcelona, Spain.

Luhn, H. (1958). The automatic creation of literature abstracts. *IBM Journal 2*(2), 159–165.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In L. M. Le Cam et J. Neyman (Eds.), *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Volume 1, Statistics. University of California Press.

Marcu, D. (1998). Improving summarization through rhetorical parsing tuning.

Nenkova, A., R. J. Passonneau, et K. McKeown (2007). The pyramid method : Incorporating human content selection variation in summarization evaluation. *TSLP 4*(2).

Radev, D., A. Winkel, et M. Topper. (2002). Multi document centroid-based text summarization. In *Proceedings of the ACL 2002 Demo Session*, Philadelphia, PA, USA.

Ribeiro, R. et D. M. de Matos (2007). Extractive summarization of broadcast news : comparing strategies for european portuguese. In *Proceedings of the 10th international conference on Text, speech and dialogue*, TSD'07, Berlin, Heidelberg, pp. 115–122. Springer-Verlag.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the International Conference on New Methods in Language Processing*, Manchester, UK.

Varma, V., P. Bysani, K. R. V. Bharat, S. Kovelamudi, S. GSK, K. Kumar, et N. Maganti (2009). Iit hyderabad at tac 2009. In *Notebook papers and results of TAC 2009*, Gaithersburg, Maryland, USA.

Wang, B., B. Liu, C. Sun, X. Wang, et B. Li (2009). Adaptive maximum marginal relevance based multi-email summarization. In *Proceedings of the International Conference on Artificial Intelligence and Computational Intelligence*, AICI '09, Berlin, Heidelberg, pp. 417–424. Springer-Verlag.

Générer des résumés de mise à jour

Summary

In this article, we present a summarization system dedicated to update summarization. We first present the method on which this system is based, CBSEAS, and its adaptation to the update summarization task. Generating update summaries is a far more complicated task than generating "standard" summaries, and needs a specific evaluation. We describe TAC 2009 "Update Task", which we used in order to evaluate our system. This international evaluation campaign allowed us to confront CBSEAS to others automatic summarization systems. Finally, we show and discuss the interesting results obtained by our system.

$\mathcal{AD}$-$\mathcal{C}$LUST: Détection des Anomalies basée sur le Clustering

Imen Brahmi*, Sadok Ben Yahia *, Pascal Poncelet **

*Faculté des Sciences de Tunis.
Campus Universitaire, 1060 Tunis, Tunisie.
imen.brahmi@gmail.com, sadok.benyahia@fst.rnu.tn
** LIRMM UMR CNRS 5506, 161 Rue Ada, 34392 Montpellier Cedex 5, France
pascal.poncelet@lirmm.fr

Résumé. En réponse aux problèmes posés par la complexité croissante des réseaux et des attaques informatiques, la détection d'intrusions constitue un domaine motivant qui a attiré l'attention des chercheurs. En effet, la technique de clustering a été considérée comme une méthode très utile pour la détection des anomalies. Dans ce papier, nous introduisons un nouvel algorithme qui combine le clustering basé sur la densité et K-MEANS, visant à résoudre les lacunes de chacun d'eux. Les résultats des expériences montrent que notre algorithme offre une méthode efficace pour la détection d'intrusions avec un taux de détection élevés et un taux de faux positif faible.

1 Introduction

Le revers de la médaille de l'ouverture de l'entreprise sur Internet réside dans une éventuelle vulnérabilité face au nombre croissant d'intrusions. Par exemple, une étude récente du National Institute of Standards and Technology a montré que les dommages, pour les compagnies américaines, étaient estimés à plus de 59,6 millions de dollars par an. Pour pallier ce problème, la mise en place d'une politique de sécurité autour de ces systèmes est donc primordiale afin d'aboutir à un Système de Détection d'Intrusions (SDI)s efficace et robuste. En effet, la détection d'intrusions consiste à analyser les informations collectées afin de rechercher d'éventuelles attaques. Ainsi, les SDIs sont des outils qui automatisent les processus de surveillance des événements système et d'analyse du trafic réseau afin de détecter des comportements malveillants (Denning, 1987). Ils peuvent être classés en deux grandes catégories (Denning, 1987) : *les systèmes de détection d'anomalies* et *les systèmes de détection d'abus*.

Le principe des approches de détection d'abus consiste à appliquer des techniques d'apprentissage sur des attaques connues de manière à en définir leurs signatures [1]. Ensuite, à l'aide d'expressions régulières ou de correspondance de motifs ces dernières sont utilisées pour reconnaître les attaques. Si ces approches sont efficaces pour reconnaître les attaques connues, elles sont malheureusement mises en défaut lorsque de nouvelles attaques interviennent. D'un autre côté, les systèmes de détection d'anomalies s'intéressent à l'analyse des comportements

[1] La signature est définie comme une suite des étapes observables d'une attaque.

normaux et cherchent à les caractériser. Ils considèrent ainsi qu'une intrusion correspond à un comportement qui dévie de la norme.

Dans le cadre de la détection des anomalies, la technique basée sur le clustering a été considérée comme la plus importante technique d'apprentissage non supervisée consacrée à la découverte des nouvelles attaques (Portnoy et al., 2001; Bloedorn et al., 2001; Guan et al., 2003; Oh et Lee, 2003; Chan et al., 2005; Münz et al., 2007). En effet, le clustering (regroupement) est une méthode de classification non-supervisée des éléments selon leur ressemblance. Contrairement à la classification, il n'y a pas de connaissances a priori sur les classes dans lesquelles nous regroupons les éléments. La séparation des instances dans les différents groupes s'effectue en se basant sur le calcul de similarité entre les éléments (Yunling, 2004). Typiquement, plusieurs méthodes de clustering ont été mis en exergue, comme *le clustering par partitionnement*, *le clustering basé sur la densité*, etc (Yunling, 2004).

Particulièrement, K-MEANS (MacQueen, 1967) est la méthode par partitionnement la plus populaire utilisée dans les applications scientifiques et industrielles de clustering. Cependant, cette technique souffre de quelques lacunes, spécialement lors du traitement des données du trafic réseau. En effet, la performance de K-MEANS et son efficacité comme méthode de détection des anomalies dépend de la sélection aléatoire du nombre des groupes initiaux. Par conséquence, un "mauvais choix" de ce nombre diminuera la détection de vraies anomalies et augmentera la génération de fausses alarmes. Une partition optimale peut être obtenue à condition d'énumérer d'une façon exhaustive toutes les partitions possibles, ce qui est prohibitif du point de vue temps de calcul (Guan et al., 2003).

Par ailleurs, la méthode de clustering basé sur la densité résout le problème de la dépendance du choix aléatoire du nombre des groupes initiaux. Comme exemple, nous citons l'algorithme DBSCAN (Ester et al., 1996), qui permet le partitionnement des groupes en régions denses séparées par des régions qui le sont moins (*c.-à.-d.*, données isolées). Néanmoins, la méthode basée sur la densité a une grande influence sur la capacité de détection. Elle engendre des partitions sous-optimales et considère beaucoup d'instances comme des bruits [2]. Par conséquent, ces problèmes augmentent le taux des faux positifs et diminuent le taux de détection de vraies intrusions (Zhao et al., 2008).

Dans cet article, nous examinons une autre façon de détection des intrusions en se basant sur K-MEANS. Ainsi, nous introduisons un algorithme, nommé $\mathcal{AD}$-$\mathcal{C}$LUST (*Anomaly Detection-based Clustering*), visant la diminution de génération des faux positifs et l'augmentation de détection de vraies intrusions. $\mathcal{AD}$-$\mathcal{C}$LUST combine K-MEANS et le clustering basé sur la densité afin de profiter des avantages de chacun.

Le reste de l'article est organisé comme suit. Dans la section 2, nous passons en revue les travaux antérieurs. Nous introduisons l'idée générale de notre approche dans la section 3. Nous présentons l'algorithme $\mathcal{AD}$-$\mathcal{C}$LUST dans la section 4. Les résultats des expérimentations montrant l'utilité de l'approche proposée sont présentés dans la section 5. La conclusion et les travaux futurs font l'instance de la section 6.

[2] Une instance est bruitée si elle n'appartient à aucun groupe.

2 Détection d'intrusions basée sur le clustering

L'objectif des techniques du clustering est de grouper des éléments proches dans un même groupe de manière à ce que deux données d'un même groupe soient le plus similaires possible et que deux éléments de deux groupes différents soient le plus dissemblables possible (Yunling, 2004).

Pour détecter des intrusions de types connues et nouvelles, Portnoy et al. (2001) proposent un algorithme de clustering basé sur une version de "*single-linkage*". En effet, l'algorithme commence par la création d'un certain nombre de groupes vides. Ensuite, il calcule la distance entre les centres des groupes et chaque nouvelle instance. Le groupe ayant la plus courte distance sera sélectionné. Si cette plus courte distance est inférieure à un paramètre indiquant la largeur des groupes, alors l'instance est affectée au groupe sélectionné. Sinon, un nouveau groupe est créé avec cette instance comme un centre. Enfin, toutes les instances seront réaffectées aux centres des groupes mis à jour. Bien que l'algorithme de Portnoy et al. (2001) résout le problème de la dépendance du nombre des centres de groupe initiaux, il exige une dépendance de leurs largeur. De plus, avec une valeur incorrecte de la largeur des groupes, l'algorithme peut classifier quelques intrusions comme normaux et quelques normaux comme anormaux.

Dans (Sequeira et Zaki, 2002) les auteurs ont développé un SDI basé sur le clustering dynamique où le nombre de groupes est déterminé pendant le clustering. Le système développé, nommé ADMIT (*Anomaly-based Data Mining for InTrusions*) (Sequeira et Zaki, 2002), permet de distinguer les utilisateurs réels d'un système Unix et les masqueraders (qui essaient d'abuser le système en utilisant le compte d'un utilisateur réel). Un profil est créé pour chaque utilisateur du système selon les séquences de commandes utilisées sur la console. En faisant le clustering dynamique sur les séquences de commandes, on crée des groupes de commandes. Ces groupes sont raffinés à cause du bruit dans les données. Dans la phase d'évaluation, une comparaison de la similarité entre les séquences de commandes et les centres des clusters détermine si la séquence de commandes appartient à l'utilisateur réel ou à un masquerader. ADMIT permet de trouver les intrusions sur un hôte particulier, mais il ne détecte pas les intrusions au niveau du réseau.

De même, pour détecter les anomalies au niveau des hôtes, Oh et Lee (2003) proposent un algorithme de clustering basé sur l'idée du DBSCAN (*(Density-Based Spatial Clustering of Applications with Noise*) (Ester et al., 1996). L'algorithme permet d'analyser l'ensemble de transactions dans un jeu de données d'audit et de les séparer en régions fréquentes et autres infréquentes. Il permet de déterminer la valeur optimale des paramètres de clustering afin de regrouper correctement les transactions d'un utilisateur. Par conséquent, les groupes résultants représentent les régions fréquentes des transactions et ils permettent de capturer le comportement normal d'un utilisateur plus précisément. Une anomalie est détectée en comparant la nouvelle transaction au profil normal obtenu. Enfin, les aspects divers du comportement normal d'un utilisateur peuvent être analysés séparément.

Chan et al. (2005) cherchent des outliers [3]. Ils appliquent l'algorithme des *K-Plus Proches voisins* (*K-Nearest Neighbors* K-NN) sur des logs de connexion. Les outliers trouvés représentent des intrusions. Ils notent que les attaques sont retrouvées dans les groupes ayant une densité élevée ou basse statistiquement. C'est pourquoi ils ont orienté le clustering selon la densité et la distance des groupes en même temps.

[3] L'outlier est une instance très éloignée du reste des données.

Afin de détecter les nouvelles attaques, Chandola et al. (2006) appliquent un algorithme du facteur local d'éloignement (*Local Outlier Factor* LOF) (Breunig et al., 2000). L'algorithme LOF permet d'assigner à chaque instance un degré (un score) d'éloignement par apport à ses voisins en fonction de la densité locale de ceux-ci. Cependant, l'approche de Chandola et al. (2006) présente deux importantes lacunes : (*i*) premièrement, la sensibilité au choix du paramètre qui spécifie le nombre minimal d'instances dans le voisinage local ; et (*ii*) deuxièmement, leur approche n'est pas appropriée pour les données dimensionnelles avec un nombre élevé de dimensions comme les données du trafic réseau.

Münz et al. (2007) introduisent une approche comparable à celle de Chandola et al. (2006). Cependant, cette approche ne vise pas la classification des données comme normales ou anormales. Son objectif est d'identifier des intervalles de temps montrant les comportements anormaux dans le trafic réseau. Contrairement à Chandola et al. (2006) qui groupent les flux normaux et anormaux dans un groupe unique avec l'algorithme LOF, Münz et al. (2007) supposent que les flux normaux et anormaux forment des groupes différents. Ainsi, ils déploient l'algorithme K-MEANS pour séparer les groupes normaux des autres anormaux. Enfin, les auteurs utilisent une méthode basée sur la distance pour classifier les nouveaux comportements et détecter les outliers simultanément. Bien que l'algorithme de Münz et al. (2007) traite les flux des données de réseau en temps réel, il souffre du problème de dépendance du paramètre qui indique le nombre des centres des groupes initiaux.

En particulier, l'objectif principal des approches proposées, consacrées à la détection d'intrusions en se basant sur le clustering est la détection de nouvelles intrusions. Vue son importance, la détection des intrusions inconnues ne cesse de présenter une problématique faisant l'instance de plusieurs recherches. À cet égard, l'intérêt principal de cet article est de proposer un nouvel algorithme de détection des anomalies, appelé $\mathcal{AD}$-$\mathcal{C}$LUST. L'idée intuitive derrière notre approche est d'adapter la méthode K-MEANS et remédier à ses lacunes afin de l'appliquer efficacement dans le domaine de la détection d'intrusions.

3 $\mathcal{AD}$-$\mathcal{C}$LUST : Nouvel algorithme de détection des anomalies

L'algorithme de classification non-supervisée, nommé $\mathcal{AD}$-$\mathcal{C}$LUST profite de la simplicité conceptuelle, la rapidité et les faibles exigences en taille mémoire de l'algorithme K-MEANS. De même, il profite des avantages de clustering basé sur la densité (spécialement le partitionnement automatique des groupes). Ainsi, il combine les avantages de K-MEANS avec ceux de clustering basé sur la densité afin d'aboutir à un algorithme de détection des anomalies plus efficace. Cette combinaison permet la diminution de génération des faux positifs et l'augmentation de détection de vraies intrusions.

Les étapes de notre algorithme $\mathcal{AD}$-$\mathcal{C}$LUST peuvent être récapitulées comme suit :

1. Extraction des groupes basés sur la densité afin de déterminer les centres de groupe initiaux candidats ;
2. Calcul de la distance entre le centre d'un groupe candidat et l'instance qui sera affectée au groupe le plus proche avec une distance minimale. Ainsi, nous obtenons des nouveaux groupes dont les instances sont plus similaires ;

3. Si le nombre des groupes ainsi obtenus est supérieur au nombre des groupes attendu, alors nous devons fusionner les groupes chevauchés dans des nouveaux groupes. Autrement, si le nombre des groupes obtenus est inférieur au nombre des groupes attendu, alors nous devons partitionner les groupes avec élimination des groupes vides. De cette façon, la valeur des centres des groupes initiaux, nécessaire pour appliquer K-MEANS sera déterminée automatiquement en divisant ou en fusionnant les groupes ;
4. Afin de détecter les intrusions, pour chaque nouvelle instance I :
   (a) Calculer la distance Euclidienne et trouver le groupe ayant la distance la plus courte entre son centre et l'instance I. Pour une instance x_i et un centre de groupe z_i, la distance Euclidienne peut être calculée selon l'équation suivante :

$$distance(x_i, z_i) = \sqrt{\sum_{i=1}^{n} (x_i - z_i)^2} \quad (1)$$

   (b) Classifier I par la catégorie du groupe le plus proche.

Dans ce qui suit, nous allons détailler les étapes de notre approche permettant d'aboutir au nouvel algorithme $\mathcal{AD}$-$\mathcal{C}$LUST.

3.1 Recherche des centres initiaux

L'algorithme $\mathcal{AD}$-$\mathcal{C}$LUST combine le clustering basé sur la densité et K-MEANS afin de résoudre leurs inconvénients. L'idée de combinaison de l'algorithme basé sur la distance avec le clustering basé sur la densité a été proposé dans BRIDGE (Dash et al., 2001). En effet, BRIDGE utilise K-MEANS en une première étape. Contrairement à BRIDGE, $\mathcal{AD}$-$\mathcal{C}$LUST utilise le clustering basé sur la densité en première étape, afin de déterminer automatiquement le nombre des centres des groupes initiaux. Les groupes ainsi obtenus seront considérés comme des groupes initiaux candidats. Le nombre des groupes initiaux candidats est symbolisé par k'. k' spécifie le nombre minimum d'instances dans un voisinage [4] et contrôle la granularité des groupes finaux de clustering basé sur la densité. Si la valeur du paramètre k' est élevée, alors plusieurs groupes générés pourraient former potentiellement un seul groupe. Ces groupes générés sont de petite taille, avec des granularités fines et des instances plus semblable l'une à l'autre. Autrement, si la valeur du paramètre k' est faible, alors un nombre bas des groupes est formé et les instances non ressemblantes seront regroupées dans un seul groupe.

3.2 Fusionnement

Dans cette sous-section, nous discutons le cas où la valeur du paramètre k' est élevée. En effet, $\mathcal{AD}$-$\mathcal{C}$LUST peut fusionner deux groupes chevauchés en un plus grand groupe.

Soit {C'$_1$,.., C'$_i$,.., C'$'_k$} l'ensemble de k' groupes candidats basés sur la densité. Supposons que k' est supérieur au nombre attendu des groupes k. Chaque groupe C$_i$ contient un ensemble

[4] Une instance p est au voisinage d'une autre q, ssi la distance(p, q) est $\leq$ au rayon du voisinage ε.

$\mathcal{D}$	:	Base de données contenant n instances {p_1, ..., p_n}.
Z	:	Les centres des groupes.
C	:	Les groupes obtenus avant fusionnement.
C'	:	Les groupes obtenus après fusionnement.

TAB. 1 – *Liste des notations utilisées dans la procédure de fusionnement.*

d'instances $\{x_1, .., x_j, .., x_{n_i}\}$. Le rayon du groupe C_i est définie par :

$$r_i = max_{j=1}^{n_i} \sqrt{\sum_{i=1}^{n} (x_j - z_i)^2}, \quad x_j \in C_i \tag{2}$$

tel que z_i est le centre du groupe C_i et $\sqrt{\sum_{i=1}^{n} (x_j - z_i)^2}$ représente la distance entre l'instance x_j et z_i. La similarité entre deux groupes c_i et c_j est calculée par l'équation suivante :

$$d(c_i, c_j) = \frac{\sqrt{\sum_{i=1}^{n} (z_j - z_i)^2}}{r_i + r_j} \tag{3}$$

Afin de réduire le nombre des groupes candidats k' au nombre attendu k, nous pouvons fusionner deux groupes chevauchés selon l'équation 3. La procédure du fusionnement est illustrée par l'algorithme 1. Les notations utilisées sont récapitulées dans le tableau 1.

Algorithme 1 : La procédure de fusionnement.

```
Données : D, Z.
Résultats : C'.
1  début
2      pour chaque p_i ∈ D faire
3          pour chaque z_j ∈ Z faire
4              d_{i,j} = distance(p_i, z_j) ;
5          Affecter chaque instance p_i à son groupe le plus proche C_j avec la distance d_{i,j}
           minimale;
6          // Obtention des nouveaux groupes C
7      pour chaque C_i ∈ C faire
8          Recalculer le centre z_i du groupe C_i;
9          Calculer le rayon r_i du groupe selon l'équation 2;
10     pour i=1, |C| +1, i++ faire
11         pour j=1, |C| +1, j++ faire
12             Calculer la similarité d(c_i, c_j) selon l'équation 3;
13             Fusionner deux groupes avec la plus petite d(c_i, c_j) pour construire C' ;
14 fin
```

À chaque fois où la procédure du fusionnement est utilisée, deux groupes chevauchés sont fusionnés. Dans l'algorithme $\mathcal{AD}$-$\mathcal{C}$LUST (*cf.* Algorithme 2), la procédure se répète $(k' - k)$ fois jusqu'à achèvement du nombre final k. La procédure commence par l'allocation des instances dans les groupes initiaux (*cf.* lignes 2-6). Les lignes (7-9) recalculent les nouveaux

centres et les rayons des k' groupes. Enfin, les lignes (10-13) fusionnent les groupes chevauchés.

Dans la sous-section suivante, nous discutons le cas où la valeur du paramètre k' est faible.

3.3 Partitionnement

$\mathcal{AD}$-$\mathcal{C}$LUST partitionne les groupes en se basant sur la méthode K-MEANS. Cependant, K-MEANS est sensible aux outliers. Afin de résoudre ce problème, $\mathcal{AD}$-$\mathcal{C}$LUST calcule les distances Euclidienne entre les instances et les centres des groupes. Lorsque la distance d'une instance est supérieure à un seuil indiquant la distance maximale, cette instance est un outlier. Une fois qu'un outlier est trouvé, il sera assigné comme le centre d'un nouveau groupe. Par conséquent, toutes les données sont partitionnées en nouveaux groupes dans lesquels les instances sont plus similaires. En outre, l'application de K-MEANS peut générer des groupes vides. Ce fait engendre un problème de dégénérescence puisque les groupes vides sont inutiles pour la classification et la détection des attaques (Guan et al., 2003).

En effet, il existe deux méthodes permettant d'éliminer la dégénérescence : (1) la suppression des groupes vides, et (2) le remplacement des groupes vides par des nouveaux groupes qui sont non vides. La première solution réduit le nombre de groupes alors que la deuxième ne les change pas. Cette dernière cherche les instances appropriées pour former des groupes non vides et remplacer ainsi ceux qui sont vides. Ensuite, toutes les instances sont réaffectées à leurs groupes plus proches jusqu'à la disparition des groupes vides et la stabilité des centres.

Durant l'étape du partitionnement, $\mathcal{AD}$-$\mathcal{C}$LUST utilise la méthode de remplacement des groupes vides afin d'éliminer la dégénérescence.

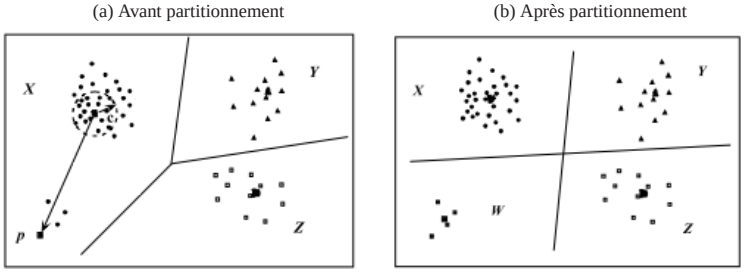


FIG. 1 – *Partitionnement.*

La figure 1 illustre l'étape du partitionnement. En effet, la figure 1(a) montre un jeu d'instances divisé en k groupes, $k = 3$. Chaque instance est assignée à son groupe le plus proche. L'instance c représente le centre du groupe X. L'instance p est assignée au groupe X puisqu'il est le groupe le plus proche auquel l'instance p peut être assignée. Cependant, l'instance p est éloignée de la majorité des instances dans le groupe. Ainsi, elle est un outlier local du groupe X. Soit d_{max} un seuil qui indique la distance maximale permettant de déterminer les outliers et r représente la distance entre l'instance p et le centre c. Un outlier est détecté si $r > d_{max}$. La figure 1(b) montre le jeu d'instances après partitionnement. L'outlier p forme un nouveau groupe, *c.-à.-d.* le groupe W.

4 L'algorithme $\mathcal{AD}$-$\mathcal{C}$LUST

Le pseudo-code de $\mathcal{AD}$-$\mathcal{C}$LUST est donné par l'algorithme 2. Les notations utilisées sont récapitulées dans le tableau 2.

$\mathcal{D}$	:	Base de données contenant n instances {p_1, ..., p_n}.
ε	:	Le rayon maximal d'un voisinage.
η	:	Le nombre minimum de points dans le ε.Voisinage.
N	:	Ensemble de point de ε.Voisinage de p_i
C	:	Les groupes classés finalement obtenus.
Z	:	Les centres des groupes finaux.
k	:	Le nombre des groupes attendu.
C'	:	Les groupes basés sur la densité initialement obtenus.
Z'	:	Les centres des groupes initiaux.
k'	:	Le nombre de groupes initialement obtenus en se basant sur la densité.

TAB. 2 – *Liste des notations utilisées dans l'algorithme $\mathcal{AD}$-$\mathcal{C}$LUST.*

En effet, l'algorithme commence par le partitionnement des groupes en se basant sur la densité (*cf.* lignes 2-14). Il détermine le voisinage des instances. Si le voisinage d'une instance est supérieur à η, alors un nouveau groupe est créé. Ensuite, cette instance est affectée au groupe courant. Sinon, l'instance est considérée comme bruit. Les groupes basés sur la densité obtenus sont les groupes candidats initiaux. L'algorithme calcule la distance entre chaque centre candidat et chaque instance. Ensuite, il affecte l'instance au groupe le plus proche (*cf.* lignes 15-17). Enfin, dans les (lignes 18-26), $\mathcal{AD}$-$\mathcal{C}$LUST applique les procédures du partitionnement et de fusionnement afin de déterminer automatiquement la valeur de k. Après la détermination de k, un nouvel ensemble des centres initiaux est obtenu et utilisé dans K-MEANS comme des groupes initiaux pour classer l'ensemble des données $\mathcal{D}$ (*cf.* lignes 27-28).

5 Évaluation expérimentale

Afin d'évaluer la performance d'$\mathcal{AD}$-$\mathcal{C}$LUST, nous avons mené une série d'expérimentations sur un PC équipé d'un Pentium 4 avec une fréquence d'horloge de 3 GHz et une mémoire RAM de 2 Go, utilisant la distribution de Linux Fedora Core 6 comme un système d'exploitation. Nous avons choisi de comparer notre algorithme à K-MEANS et à DBSCAN [5]. Durant les expérimentations effectuées, nous avons utilisé la base de données orientées détection d'intrusions, fréquemment utilisées pour évaluer les performances des algorithmes s'inscrivant dans cette tendance : KDD 1999 [6].

En effet, les données de KDD99 sont sous forme de lignes enregistrées. Chaque ligne référence une connexion. Chaque connexion est caractérisée par 41 attributs. À partir des valeurs de ces attributs, chaque connexion est étiquetée comme étant une connexion normale ou bien une attaque. La base de données KDD99 recense 22 attaques possibles qui peuvent être regroupées en quatre catégories :

– *Déni de Service (Denial-Of-Service (DOS))* ;

[5] Disponible à l'adresse suivante : *http ://codingplayground.blogspot.com/2009/11/dbscan-clustering-algorithm.html.*

[6] Disponible à l'adresse suivante : *http ://kdd.ics.uci.edu/databases/kddcup99/.*

Algorithme 2 : L'algorithme $\mathcal{AD}$-$\mathcal{C}$LUST

Données :
1. $\mathcal{D}$.

Résultats : C.

début
  C' :=0 ;
  k' :=0 ;
  pour chaque $p_i \in \mathcal{D}$ *et* p_i *n'est pas visité* **faire**
    Calculer le nombre de points N dans le voisinage de p_i défini par ε;
    si $N < \eta$ **alors**
      Marquer p_i comme bruit ;
    sinon
      Marquer p_i comme instance visitée ;
      Ajouter p_i au groupe C' ;
      C' := C' + 1 ;
      k' := k' + 1 ;
      Continuer la visite des points du voisinage ;
    // Nous obtenons C' groupes basés sur la densité.
  pour chaque *groupe C'* **faire**
    Calculer le centre z'_j de chaque groupe;
    Affecter chaque instance p_i à son groupe le plus proche tel que C'(p_i) = min_distance(p_i, z'_j);
  répéter
    si $k' > k$ **alors**
      // Répéter la procédure de fusionnement $k' - k$ itérations
      C' := Fusionnement($\mathcal{D}$, Z');
      Recalculer le centre z'_j de chaque groupe;
    sinon
      Partitionner les groupes en utilisant K-MEANS;
      Recalculer le centre z'_j de chaque groupe;
  jusqu'à *achever* k *groupes*;
  Z :=Z';
  C := K-MEANS(k, Z, $\mathcal{D}$);
fin

– *Les attaques de type "Remote to Local access" (R2L)* ;
– *Les attaques de type "User to Root attacks" (U2R)* ;
– *Les attaques de reconnaissance (Probing)*.

Nous utilisons cinq sous-ensembles de l'ensemble de données KDD99 avec des tailles différentes variant de 40,000 à 200,000 connexions. Les caractéristiques de ces bases sont résumées par le tableau 3. La première colonne du tableau indique la taille de chaque sous-ensemble de données. La deuxième colonne présente le nombre des instances normaux, alors que la dernière illustre la distribution des attaques.

À travers des expériences d'estimation de paramètres, nous décidons finalement de choisir 5 comme la valeur de ε, 5 comme la valeur du seuil permettant de détecter les outliers et 8

Taille de sous-ensemble de la base de données	Normal	Intrusions
S_1=40,000	30,000	10,000
S_2=80,000	60,000	20,000
S_3=120,000	90,000	30,000
S_4=160,000	120,000	40,000
S_5=200,000	100,000	100,000

TAB. 3 – *Descriptif des jeux de données.*

comme la valeur de η.

Dans le contexte de la détection d'intrusions, deux mesures importantes peuvent être mis en exergue pour évaluer les algorithmes :

1. Le Taux de Détection (TD) mesure combien d'attaques sont détectées correctement ;
2. Le Taux de Faux positifs (TF) mesure combien d'alertes sont fausses.

TD est équivalent au taux de rappel des algorithmes d'apprentissage, alors que TF est relatif au taux de la précision.

De plus, pour évaluer l'efficacité de la technique de classification, nous nous basons sur le *pourcentage de classification correcte* (PCC) des instances défini par :

$$PCC = \frac{nombre\ des\ instances\ correctement\ classees}{nombre\ total\ des\ instances\ classees} \quad (4)$$

5.1 Analyse de la performance d'$\mathcal{AD}$-$\mathcal{C}$LUST

Dans cette sous-section, nous analysons la performance d'$\mathcal{AD}$-$\mathcal{C}$LUST en utilisant les cinq sous-ensembles de données. Le tableau 4 évalue la performance de $\mathcal{AD}$-$\mathcal{C}$LUST en terme des deux taux TD et TF. Ainsi, nous pouvons conclure que $\mathcal{AD}$-$\mathcal{C}$LUST génère un taux élevé de détection (plus de 98%). De plus, il engendre peu de faux positifs, soit 2%.

Sous-ensemble de données	k	TD	TF
S_1	167	98.6741%	2.6230%
S_2	453	99.2843%	1.8159%
S_3	697	99.2074%	1.6259%
S_4	700	99.3282%	1.4206%
S_5	745	99.3189%	2.1575%

TAB. 4 – *Les taux de détection (TD) et de faux positifs (TF) d'*$\mathcal{AD}$-$\mathcal{C}$LUST.

Par ailleurs, le tableau 5 présente les PCC d'$\mathcal{AD}$-$\mathcal{C}$LUST relativement aux quatre catégories d'intrusions, *i.e.*, *DoS*, *U2R*, *Probing*, *R2L*. À cet égard, nous remarquons que le PCC de la catégorie *U2R* est 40%. Nous pouvons expliquer ce TD faible par la pauvreté du nombre des instances de *U2R* dans l'ensemble de données d'apprentissage, soit 52. Par conséquent, il est difficile de regrouper des *U2R*. Par contre, si le nombre des instances de *U2R* est plus élevé, alors le TD sera également élevé. De plus, d'après les expériences nous constatons également que le nombre de groupes vides générés par $\mathcal{AD}$-$\mathcal{C}$LUST est faible de sorte que nous n'avons

pas besoin de les traiter. Par exemple, le nombre des groupes vides pour le sous-ensemble de données de taille 200.000 est d'environ 6. La bonne performance d'$\mathcal{AD}$-$\mathcal{C}$LUST s'explique par la détermination automatique du nombre des groupes et par le bon partitionnement des centres des groupes initiaux.

Sous-ensemble de données	k	*DoS*	*U2R*	*Probing*	*R2L*
S_1	167	99.3782%	34.6154%	95.0085%	86.4121%
S_2	453	99.5097%	38.4615%	95.8364%	89.7869%
S_3	697	99.3845%	38.4615%	94.0833%	88.7211%
S_4	700	99.4450%	38.4615%	95.2764%	89.0764%
S_5	745	99.4389%	42.3077%	94.6433%	85.6128%

TAB. 5 – *Les PCC d'*$\mathcal{AD}$*-*$\mathcal{C}$LUST *relatifs aux différentes catégories d'intrusions.*

5.2 Comparaison des capacités de détection

Pour montrer que $\mathcal{AD}$-$\mathcal{C}$LUST est plus performant par rapport aux autres algorithmes de clustering, nous exécutons séparément K-MEANS, DBSCAN et $\mathcal{AD}$-$\mathcal{C}$LUST avec le même sous-ensemble de données de taille 100,000.

Le tableau 6 présente les résultats obtenus en termes de taux de détection et de fausses alarmes. L'analyse du tableau 6 montre que $\mathcal{AD}$-$\mathcal{C}$LUST possède la meilleure capacité de détection en comparaison avec K-MEANS et DBSCAN.

Algorithme	k	TD	TF
K-MEANS	5	26.2579%	0.3256
	23	57.5292	3.0951
	300	87.9820	8.9001
	600	97.1953	12.5054
	700	99.1166	15.9213
DBSCAN	621	94.6456	24.5815
AD-CLUST	621	**99.1246%**	**1.4107%**

TAB. 6 – *Les taux TD et TF d'*$\mathcal{AD}$*-*$\mathcal{C}$LUST *vs.* K-MEANS *et* DBSCAN.

En effet, K-MEANS est expérimenté avec des valeurs différentes du nombre des groupes k. D'après le tableau 6, nous remarquons que les deux taux TD et TF de K-MEANS augmentent avec l'augmentation de k. Cependant, cette relation n'est pas appropriée puisque pour un algorithme de détection d'anomalies efficace le TF doit décroître avec l'augmentation de TD (Guan et al., 2003). De plus, nous observons que lorsque les valeurs de k sont égaux à 5, 23 et 300 les taux TD sont faibles. Lorsque la valeur de k est égale à 600, le taux TD augmente brusquement pour arriver à 97%. Ce changement est dû à l'obtention de la partition optimale. Ainsi, avec K-MEANS, nous avons essayé différentes valeurs du paramètre k afin de trouver l'optimal [7].

[7] La valeur optimale du paramètre k est la valeur appropriée qui donne une partition efficace.

Cependant, $\mathcal{AD}$-$\mathcal{C}$LUST réduit le temps de calcul et partitionne automatiquement le même ensemble de données en 621 groupes. En revanche, nous remarquons que l'algorithme $\mathcal{AD}$-$\mathcal{C}$LUST permet la diminution des taux TF durant l'augmentation du TD, contrairement à K-MEANS.

De plus, le tableau 6 montre que l'algorithme $\mathcal{AD}$-$\mathcal{C}$LUST donne un TD (99.1246%) plus élevé par rapport à DBSCAN (94.6456%). L'algorithme génère un TF faible (1.4107%) alors que celui de DBSCAN s'élève à 24.5815%. Les mauvaises résultats de DBSCAN s'explique par les partitions sous-optimales de l'ensemble de données et l'élimination de plusieurs instances considérées comme des données bruitées.

Dans la suite des expériences, nous fixons le paramètre k de K-MEANS à l'optimal trouvé au cours des expérimentations précédentes, *c.-à.-d.* à 621.

Les deux taux TD et TF sont liés. Par conséquent, ces deux taux sont considérés comme les deux paramètres permettant l'élaboration d'une courbe ROC (*Receiver Operating Characteristic*) (Maxion et Roberts, 2004). La courbe ROC met en relation dans un graphique les taux de fausses alarmes (TF en abscisse) et les taux de détection (TD en ordonnée). ROC est une "*courbe de caractéristiques d'efficacité*" permettant d'étudier les variations de la spécificité et de la sensibilité (Maxion et Roberts, 2004). La figure 2 compare la courbe ROC d'$\mathcal{AD}$-$\mathcal{C}$LUST *vs.* celles de K-MEANS et DBSCAN. Plus l'algorithme de détection d'intrusions est performant, plus il va générer des fausses alarmes, plus le taux de détection est élevé. Ainsi, d'après la figure 2, nous remarquons que l'algorithme $\mathcal{AD}$-$\mathcal{C}$LUST est plus efficace que K-MEANS et DBSCAN.

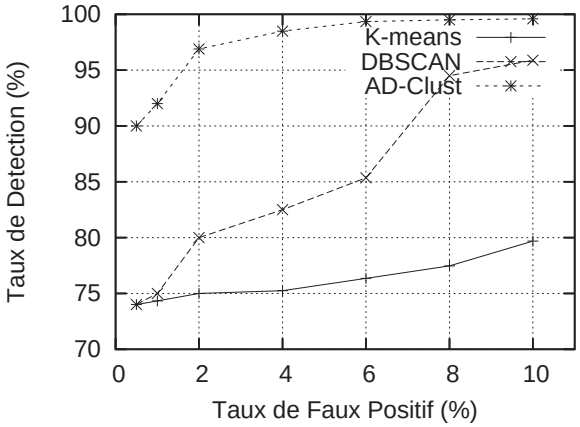


FIG. 2 – *Efficacité*

Certains algorithmes perdent beaucoup de leur efficacité face à des données volumineuses. Nous parlons ainsi de *la scalabilité*. La figure 3 met en exergue *le compromis qualité-rapidité* d'$\mathcal{AD}$-$\mathcal{C}$LUST. D'après la figure 3, nous remarquons que le temps d'exécution des trois algorithmes augmente en fonction de l'augmentation du volume des données d'entrée. En outre, il est claire que l'algorithme $\mathcal{AD}$-$\mathcal{C}$LUST est plus rapide que DBSCAN et K-MEANS.

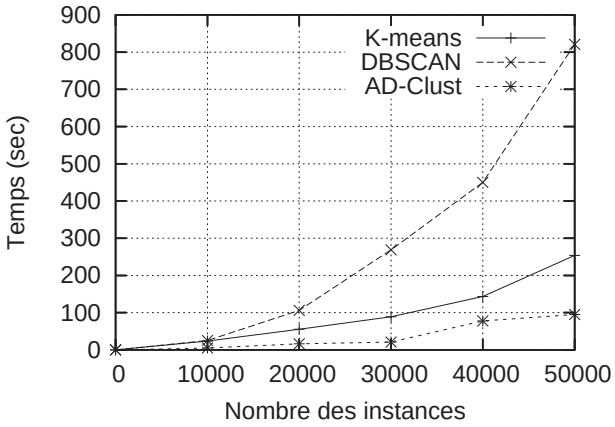


FIG. 3 – *Scalabilité*

Généralement, la plupart des chercheurs qui appliquent les algorithmes d'apprentissage sur la base de données KDD99 trouvent que les PCC obtenus par les catégories *DoS* et *Probe* sont élevés. Cependant, les catégories *R2L* et *U2R* possèdent des PCC faibles (Sabhnani et Serpen, 2004). La figure 4 compare les PCC obtenus avec notre algorithme par rapport à ceux trouvés avec K-MEANS et DBSCAN pour les quatre catégories d'attaques. Les résultats expérimentales montrent que l'algorithme $\mathcal{AD}$-$\mathcal{C}$LUST obtient un meilleur PCC *vs.* ceux obtenus avec K-MEANS et DBSCAN.

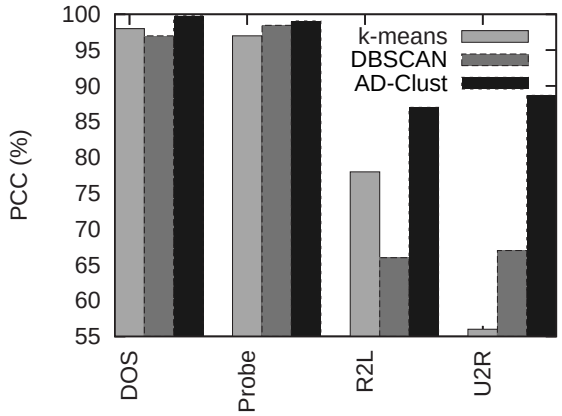


FIG. 4 – *Les PCC relatives aux quatre catégories d'attaques.*

6 Conclusion et perspectives

Les approches de détection d'intrusions actuelles, basée sur K-MEANS, souffrent de la dépendance du nombre des groupes initiaux, la sensibilité aux outliers, et la dégénérescence. Ceci engendre la génération excessive de fausses alarmes et la diminution de détection de vraies intrusions. Pour remédier ces insuffisances, nous avons proposé un nouvel algorithme, appelé $\mathcal{AD}$-$\mathcal{C}$LUST, de détection des anomalies qui se base sur la technique de clustering. Le nouvel algorithme combine le clustering basé sur la densité et K-MEANS. Son objectif est l'adaptation de l'algorithme K-MEANS et la résolution de ses limites afin de l'appliquer efficacement dans le domaine de la détection d'intrusions. Les expérimentations réalisées ont montré que $\mathcal{AD}$-$\mathcal{C}$LUST est un algorithme efficace, puisqu'il génère peu de fausses alarmes et détecte plus de vraies intrusions. Comme perspectives de prolongement de ce travail, nous proposons d'examiner la détection distribuée d'anomalies en se basant sur le système multi-agents et l'algorithme $\mathcal{AD}$-$\mathcal{C}$LUST.

Références

Bloedorn, E., A. D. Christiansen, W. Hill, C. Skorupka, L. M. Talbot, et J. Tivel. (2001). Data Mining for Network Intrusion Detection : How to Get Started. Technical report, The MITRE Corporation.

Breunig, M. M., H.-P. Kriegel, R. T. Ng, et J. Sander (2000). LOF : Identifying Density-Based Local Outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, Texas, USA*, pp. 93–104.

Chan, P. K., M. V. Mahoney, et M. H. Arshad (2005). Learning Rules and Clusters for Anomaly Detection in Network Traffic. In V. Kumar, J. Srivastava, et A. Lazarevic (Eds.), *Managing Cyber Threats : Issues, Approaches and Challenges*, pp. 81–99. Springer.

Chandola, V., E. Eilertson, L. Ertoz, G. Simon, et V. Kumar (2006). Data Mining for Cyber Security. In A. Singhal (Ed.), D*ata* W*arehousing and Data* M*ining* T*echniques for* C*omputer Security*, pp. 83–103. Springer.

Dash, M., H. Liu, et X. Xu (2001). '1 + 1 > 2' : Merging distance and density based clustering. In *Proceedings of the 7th International Conference on Database Systems for Advanced Applications (DASFAA 2001), Hong Kong, China*, pp. 32–39.

Denning, D. E. (1987). An Intrusion Detection Model. *IEEE Transactions on Software Engineering 13*(2), 222–232.

Ester, M., H.-P. Kriegel, J. Sander, et X. Xu (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In E. Simoudis, J. Han, et U. M. Fayyad (Eds.), *Second International Conference on Knowledge Discovery and Data Mining, Portland, Oregon*, pp. 226–231.

Guan, Y., A. Ghorbani, et N. Belacel (2003). Y-MEANS : A Clustering Method for Intrusion Detection. In *Proceedings of Canadian Conference on Electrical and Computer Engineering ; Montréal, Québec, Canada*, pp. 1083–1086.

MacQueen, J. (1967). Some Methods for Classification and Analysis of Multivariate Observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics*

and Probability, Berkeley, pp. 281–297.

Maxion, R. A. et R. R. Roberts (2004). Proper Use of ROC Curves in Intrusion/Anomaly Detection. Technical report series cs-tr-871, School of Computing Science, University of Newcastle upon Tyne.

Münz, G., S. Li, et G. Carle (2007). Traffic Anomaly Detection Using K-Means Clustering. In *Proceedings of GI/ITG-Workshop MMBnet 2007, Hamburg, Germany*.

Oh, S. H. et W. S. Lee (2003). Optimized Clustering for Anomaly Intrusion Detection. In *Proceedings of the 7th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, PAKDD 2003, Seoul, Korea*, pp. 576–581.

Portnoy, L., E. Eskin, et W. S. J. Stolfo (2001). Intrusion Detection with Unlabeled Data using Clustering. In *Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001), Philadelphia, PA*.

Sabhnani, M. et G. Serpen (2004). Why Machine Learning Algorithms Fail in Misuse Detection on KDD Intrusion Detection Data Set. *Journal of Intelligent Data Analysis 6*, 1–13.

Sequeira, K. et M. J. Zaki (2002). ADMIT : Anomaly-based Data Mining for Intrusions. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada*, pp. 386–395.

Yunling, W. (2004). A comparative study of classification algorithms for network intrusion detection. Ms-thesis, Florida Atlantic University.

Zhao, Z., S. Guo, Q. Xu, et T. Ban (2008). G-MEANS : A Clustering Algorithm for Intrusion Detection. In *ICONIP 2008*, pp. 563–570.

Summary

Coupled with the explosion of the number of the network-oriented applications, intrusion detection as an increasingly popular the area is attracting more and more research efforts. Int this respect, clustering technique has been recognized as a very useful method for the anomaly intrusion detection. In this paper, a novel density-based K-MEANS clustering algorithm has been introduced to overcome the drawbacks of clustering-based density and K-MEANS algorithms. Carried out experiments show that our proposed algorithm is an effective method for the intrusion detection with a high detection rate and a low false positive rate.

Score de Fisher et détection de courbes anormales dans une séquence à des fins de diagnostic

Etienne Côme*, Allou Samé*, Patrice Aknin* et Marc Antoni**

* INRETS
2, Rue de la Butte Verte F-93166 Noisy le Grand Cedex,
come@inrets.fr,
http://www.inrets.fr
** SNCF, Direction Contrats et Services Clients ,
Pôle Technique Ingénierie de Maintenance
Cellule émergence et prospectives
17 Rue d'Amsterdam, 75379, Paris Cedex 08

Résumé. Cet article concerne la détection de courbes anormales dans une séquence de courbes à des fins de diagnostic. La méthodologie proposée se base sur l'utilisation conjointe d'un modèle génératif permettant d'estimer un modèle pour les courbes correspondants à un fonctionnement nominal du système et sur un test du score permettant de déterminer si une nouvelle courbe est anormale ou non par rapport à ce modèle de fonctionnement nominal. La méthodologie sera détaillée ainsi qu'un exemple d'utilisation concrète de celle-ci dans le contexte du diagnostic du mécanisme d'aiguillage. Dans cette application, les courbes correspondent à l'évolution de la puissance consommée par le moteur de l'aiguille durant la manœuvre d'aiguillage.

1 Introduction

Le score de Fisher, c'est-à-dire le gradient de la log-vraisemblance, est une quantité centrale en statistique qui a déjà démontrée son intérêt dans de nombreux problèmes tel que la classification supervisé Jaakkola et Haussler (1998); Jaakkola et al. (1999) ou la détection de ruptures dans les paramètres d'un modèle statistique, Horvarth et Parzen (1994). Dans cet article, l'intérêt de celui-ci pour analyser un flux de courbes sera étudié. Nous verrons tout d'abord comment le score de Fisher peut être calculé dans le contexte d'un modèle de régression à processus logistique latent, Chamroukhi et al. (2010). Le processus latent évoqué permet de gérer la succession des différentes phases de la manœuvre d'aiguillage au sein d'une même courbe. Ce modèle génératif de type modèle de mélange d'expert, cf. Carvalho et Tanner (2006), est en effet particulièrement adapté pour modéliser des courbes présentant des changements de régimes, telles que celles utilisées pour le diagnostic des aiguillages (cf. figure 1). L'intérêt du score de Fisher pour exploiter ce modèle sera ensuite doublement illustré. Tout d'abord au travers de la définition d'un algorithme de type gradient stochastique

pour mettre à jour les paramètres de ce modèle génératif à chaque nouvelle acquisition de courbe. Ensuite, grâce à un test du score Rao (1948) permettant de détecter des courbes atypiques. Finalement, ces deux ingrédients seront combinés pour construire un algorithme de détection en ligne de courbes anormales. Cet algorithme sera ensuite utilisé dans le cadre du diagnostic et du suivi de point de fonctionnement d'un organe clé de l'infrastructure ferroviaire, le mécanisme d'aiguillage. Dans cette application l'un des objectifs visés est l'identification en ligne des défauts précoces de ce système afin de permettre la mobilisation rapide des services de maintenance concernés. Ce suivi d'état du mécanisme d'aiguillage est réalisé à partir de plusieurs réalisations de courbes organisées en séquence, chaque réalisation représentant la puissance électrique consommée par le moteur d'aiguillage durant une manœuvre (cf. figure 1). Ce travail a donc pour objectif applicatif de développer une méthode de détection en ligne d'anomalie sur les courbes de manœuvres d'aiguillage. Ces travaux font suite à ceux proposés dans Samé et al. (2008) qui s'appuyaient sur des tests séquentiels d'hypothèses multiples.

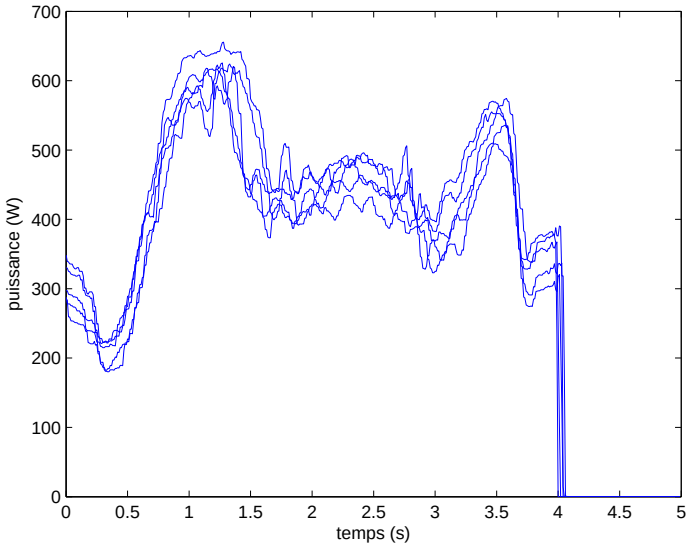


FIG. 1 – *Exemple d'une séquence de 5 signaux acquis durant des manœuvres d'aiguillage*

L'article est organisé de la manière suivante. La section 2 présentera le modèle de régression à processus logistique latent utilisé pour modéliser les courbes. La méthode d'estimation classique des paramètres de ce modèle, c'est à dire à l'aide d'un algorithme EM, sera rappelée brièvement dans cette même section. La section suivante présentera le calcul du score de Fisher dans le contexte de ce modèle ainsi que son utilisation dans le contexte d'un gradient stochastique. Le test du score utilisé pour détecter les courbes anormales ainsi que sa mise en œuvre pratique sera décrit en détail dans la section 4. Enfin, la section 5 détaillera les résultats obtenus à l'aide de cette méthodologie sur les données d'aiguillage.

2 Modèle génératif de bon fonctionnement

Dans cette section nous rappelons très brièvement le modèle de régression à processus logistique latent, Chamroukhi et al. (2010), utilisé pour modéliser les courbes de bon fonctionnement. On désigne par $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n$ une base d'apprentissage de n courbes, où chaque réalisation $\boldsymbol{y}_i$ est une courbe échantillonnée en m points. Pour simplifier les notations, la valeur d'une courbe $\boldsymbol{y}_i$ au point x_j sera notée y_{ij}. Chaque courbe $\boldsymbol{y}_i$ est ainsi représentée par l'ensemble de points $(y_{i1}, \ldots, y_{im})$ observés aux points $(x_1, \ldots, x_m)$.

2.1 Modèle de régressions à processus logistique latent

Le modèle de régression à processus logistique latent suppose que chaque signal $\boldsymbol{y}_i$ est défini par :

$$\forall j = 1, \ldots, m, \quad \left\{ \begin{array}{l} y_{ij} = \boldsymbol{\beta}_{z_{ij}}^T \boldsymbol{x}_j + \epsilon_{ij} \\ \epsilon_{ij} \sim \mathcal{N}(0, \sigma_{z_{ij}}^2), \end{array} \right. \tag{1}$$

où les variables z_{ij} sont des variables latentes, à valeurs dans $z_{ij} \in \{1, \ldots, K\}$, permettent de basculer d'un modèle de régression à l'autre parmi K modèles. Le paramètre vectoriel $\boldsymbol{\beta}_{z_{ij}} \in I\!R^{p+1}$ désigne le vecteur des coefficients d'un polynôme de degré p organisé par ordre de degré croissant, $\boldsymbol{x}_j = (1, x_j, \ldots, x_j^p)^T \in I\!R^{p+1}$ correspond au vecteur des monômes associés au paramètre $\boldsymbol{\beta}_{z_{ij}}$ et ϵ_{ij} est un bruit gaussien centré de variance $\sigma_{z_{ij}}^2$. Partant du modèle qui vient d'être défini, on peut montrer que, conditionnellement à x_j, la variable y_{ij} est distribuée suivant le mélange de loi :

$$p(y_{ij}|x_j; \boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k(x_j) \phi\big(y_{ij}; \boldsymbol{\beta}_k^T \boldsymbol{x}_j, \sigma_k^2\big), \tag{2}$$

où $\boldsymbol{\theta} = (\{\pi_k(x_j)\}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K, \sigma_1, \ldots, \sigma_K)$ est le vecteur des paramètres du modèle, $\phi(\cdot; \mu, \sigma^2)$ est la densité d'une variable normale centrée réduite unidimensionnelle d'espérance μ et de variance σ^2 ; les proportions du mélange $\pi_k(x_j) = P(z_{ij} = k|x_j)$ sont des fonctions de la variable x_j qui présentent donc une forte dimensionnalité. Afin de réduire leur dimension, nous utilisons le modèle logistique de paramètre $\boldsymbol{w} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K)$ suivant Jordan et Jacob (1994) :

$$\pi_k(x_j; \boldsymbol{w}) = \frac{\exp\left(\boldsymbol{w}_{k0} + \boldsymbol{w}_{k1} x_j\right)}{\sum_{\ell=1}^{K} \exp\left(\boldsymbol{w}_{\ell 0} + \boldsymbol{w}_{\ell 1} x_j\right)}. \tag{3}$$

Le vecteur des paramètres à estimer s'écrit donc finalement :

$$\boldsymbol{\theta} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K, \sigma_1, \ldots, \sigma_K).$$

Contrairement aux modèles de régression classiques, qui supposent que les paramètres de régression sont constants sur tout le signal, le modèle proposé autorise leur variation au cours du temps, en basculant d'un sous modèle de régression à l'autre. Ce qui permet notamment de mieux modéliser les non linéarités et les changements brusques qui apparaissent clairement dans les signaux étudiés (voir figure 2).

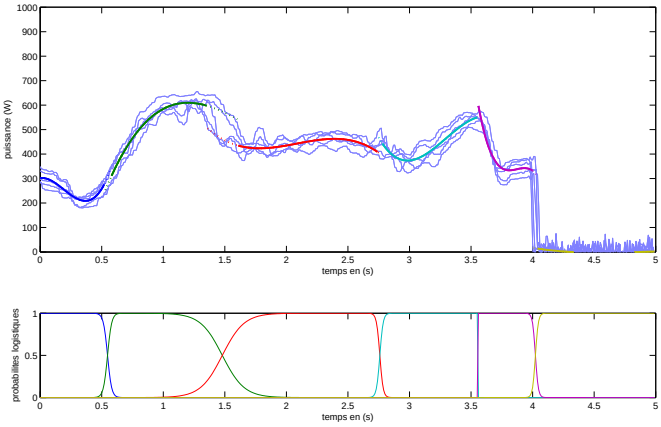


FIG. 2 – *Signal de mesure pour 5 mesures et modèle estimé. Chacun des sous modèle de régressions est représenté par une couleur et les proportions de chacun d'entre eux sont données sur la figure du bas. Le modèle comprend 6 sous modèles de régression et chaque sous modèle est cubique* $\boldsymbol{x}_j = (1, x_j, x_j^2, x_j^3)^T$. *Pour la bonne estimation de la dernière phase du signal, où la puissance est uniformément nulle, un bruit gaussien à été rajouté sur cette phase afin d'éviter l'explosion de la vraisemblance.*

2.2 Estimation des paramètres du modèle

L'estimation du paramètre $\boldsymbol{\theta}$ caractérisant un ensemble de n courbes $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n$ s'effectue par la méthode du maximum de vraisemblance qui est mise en œuvre par un algorithme EM, Dempster et al. (1977), particulièrement adapté à ce contexte de régression à variable latente.

On suppose que les courbes sont indépendantes entre elles et que, conditionnellement aux x_j $(j = 1, \ldots, m)$, les variables y_{ij} $(j = 1, \ldots, m)$ sont indépendantes également. Sous ces hypothèses, la log-vraisemblance à maximiser s'écrit :

$$\begin{aligned} L(\boldsymbol{\theta}) &= \log p(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n | \mathbf{x}; \boldsymbol{\theta}) \\ &= \sum_{i=1}^{n} \log p(\boldsymbol{y}_i | \mathbf{x}; \boldsymbol{\theta}) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{m} \log \sum_{k=1}^{K} \pi_k(x_j; \boldsymbol{w}) \phi\big(y_{ij}; \boldsymbol{\beta}_k^T \boldsymbol{x}_j, \sigma_k^2\big), \end{aligned}$$

où $\mathbf{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m)^T$. En notant $\mathbf{z}_i = (z_{ij})_{1 \leq j \leq m}$ le vecteur des labels identifiant les sous modèles de régression associé à $\boldsymbol{y}_i$, la vraisemblance dite "complétée" Dempster

et al. (1977) s'écrit quant à elle :

$$\begin{aligned} L_c(\boldsymbol{\theta}) &= \log p(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n, \mathbf{z}_1, \ldots, \mathbf{z}_n | \mathbf{x}; \boldsymbol{\theta}) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{K} z_{ijk} \log \pi_k(x_j; \boldsymbol{w}) \phi\big(y_{ij}; \boldsymbol{\beta}_k^T \boldsymbol{x}_j, \sigma_k^2\big), \end{aligned}$$

où la variable binaire z_{ijk} vaut 1 si $z_{ij} = k$ et vaut 0 sinon. La maximisation de la vraisemblance $L(\boldsymbol{\theta})$ est effectuée par l'algorithme EM (Expextation-Maximization) Dempster et al. (1977) qui consiste à partir d'un paramètre initial $\boldsymbol{\theta}^{(0)}$ et à répéter les deux étapes suivantes jusqu'à la convergence :

Etape E (Expectation) Calcul de l'espérance de la vraisemblance complétée, conditionnellement aux données observées et aux paramètre courant $\boldsymbol{\theta}^{(q)}$ (q étant l'itération courante) :

$$\begin{aligned} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(q)}) &= E[L_c(\boldsymbol{\theta}) | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_n; \boldsymbol{\theta}^{(q)}] \\ &= \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{K} t_{ijk}^{(q)} \log \pi_k(x_j; \boldsymbol{w}) \phi(y_{ij}; \boldsymbol{\beta}_k^T \boldsymbol{x}_j, \sigma_k^2), \end{aligned}$$

où

$$t_{ijk}^{(q)} = \frac{\pi_k(x_j; \boldsymbol{w}) \phi\big(y_{ij}; \boldsymbol{\beta}_k^T \boldsymbol{x}_j, \sigma_k^2\big)}{\sum_{\ell=1}^{K} \pi_\ell(x_j; \boldsymbol{w}) \phi\big(y_{ij}; \boldsymbol{\beta}_\ell^T \boldsymbol{x}_j, \sigma_\ell^2\big)}. \tag{4}$$

Comme le montre l'expression de Q, cette étape nécessite simplement le calcul des probabilités a posteriori $t_{ijk}^{(q)}$.

Etape M (Maximization) Calcul du paramètre $\boldsymbol{\theta}^{(q+1)}$ maximisant, par rapport à $\boldsymbol{\theta}$, la quantité Q qui peut aussi s'écrire :

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(q)}) = Q_1(\boldsymbol{w}) + Q_2(\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K, \sigma_1^2, \ldots, \sigma_K^2), \tag{5}$$

avec

$$\begin{aligned} Q_1 &= \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{K} t_{ijk}^{(q)} \log \pi_k(x_j; \boldsymbol{w}) \quad \text{et} \\ Q_2 &= \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{K} t_{ijk}^{(q)} \log \phi\big(y_{ij}; \boldsymbol{\beta}_k^T \boldsymbol{x}_j, \sigma_k^2\big). \end{aligned}$$

La maximisation de Q par rapport à $\boldsymbol{\theta}$ peut être effectuée en maximisant séparément Q_1 par rapport à $\boldsymbol{w}$ et Q_2 par rapport à $(\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K, \sigma_1^2, \ldots, \sigma_K^2)$. La maximisation de Q_1 par rapport à $\boldsymbol{w}$ est un problème de régression logistique multinomial pondéré par les $t_{ijk}^{(q)}$, qui peut être résolu par un algorithme itératif de type IRLS (Iterative Reweighted Least Squares) Green (2003). La maximisation de Q_2 par rapport aux $\boldsymbol{\beta}_k$ revient à résoudre séparément K problèmes de moindres carrés pondérés par les

$t_{ijk}^{(q)}$. Ces résolutions permettent d'obtenir des formules analytiques de mises à jour des paramètres fournies en annexe.

En ce qui concerne le choix du nombre de composantes régressives et de la dimensionalité du vecteur de covariables (c'est à dire le degré des polynômes), il est possible d'utiliser le critère BIC (Bayesian Information Criterion), Schwarz (1978) permettant de comparer différents modèles en compétition.

La section suivante présente le calcul du score de Fisher dans le contexte du modèle qui vient d'être présenté ainsi que son utilisation pour mettre à jour les paramètres du modèle à chaque nouvelle acquisition de courbe à l'aide d'un algorithme de type gradient stochastique.

3 Score de Fisher d'une courbe et gradient stochastique

Le vecteur score d'un modèle correspond au gradient de la log-vraisemblance. Dans cet article celui-ci sera considéré relativement à une nouvelle courbe $\boldsymbol{y}_i$ et nous définissons pour cela :

$$S^{(i)}(\boldsymbol{\theta}) = [s_1^{(i)}(\boldsymbol{\theta}), \ldots, s_R^{(i)}(\boldsymbol{\theta})]^T,$$

où R est le nombre de paramètres du modèle et

$$s_r^{(i)}(\boldsymbol{\theta}) = \frac{\partial L(\boldsymbol{\theta}; \boldsymbol{y}_i)}{\partial \boldsymbol{\theta}_r},$$

est la dérivée de la log-vraisemblance d'une nouvelle courbe par rapport au paramètre $\boldsymbol{\theta}_r$ du modèle.

Dans le contexte du modèle utilisé ici, le vecteur score peut être calculé en utilisant une formulation classique du gradient pour les modèles à variables latentes, cf. Salakhutdinov et al. (2003). Cette formulation utilise, comme l'algorithme EM, les relations existantes entre la vraisemblance et la vraisemblance complétée. Plus particulièrement, on a l'égalité suivante :

$$\begin{aligned} s_r^{(i)}(\boldsymbol{\theta}) &= \frac{\partial L(\boldsymbol{\theta}; \boldsymbol{y}_i)}{\partial \boldsymbol{\theta}_r} && (6) \\ &= E\left[\frac{\partial L_c(\boldsymbol{\theta}; Y_i, Z_i)}{\partial \boldsymbol{\theta}_r} | \boldsymbol{y}_i; \boldsymbol{\theta}\right]. && (7) \end{aligned}$$

Ce qui permet, par exemple, d'obtenir pour la composante du vecteur score associée au paramètre $\boldsymbol{\beta}_k$ la formule :

$$\begin{aligned} \frac{\partial L(\boldsymbol{\theta}; \boldsymbol{y}_i)}{\partial \boldsymbol{\beta}_k} &= E\left[\frac{\partial L_c(\boldsymbol{\theta}, Y_i, Z_i)}{\partial \boldsymbol{\beta}_k} | \boldsymbol{y}_i, \boldsymbol{\theta}\right] \\ &= \frac{1}{\sigma_k^2} \sum_{j=1}^{m} t_{ijk} \left(y_{ij} - \boldsymbol{\beta}_k^T \boldsymbol{x}_j\right) \boldsymbol{x}_j, && (8) \end{aligned}$$

avec t_{ijk} les probabilités a posteriori de la composante k au point x_j, calculée de la même manière que dans le contexte de l'algorithme EM (équation 4). Les autres composantes

du vecteur score pour le modèle de régression à processus logistique latent sont fournies en annexe.

En ce qui concerne l'estimation des paramètres du modèle pour une séquence de courbes, il serait théoriquement possible d'utiliser l'algorithme EM présenté dans la section précédente pour ré-estimer à chaque nouvelle courbe les paramètres associés à l'ensemble des courbes déjà observées. Cependant cette stratégie est irréaliste d'un point de vue applicatif, car elle entraine une augmentation du temps de calcul à chaque nouvelle acquisition de donnée ainsi qu'une augmentation des besoins de stockage, les courbes précédentes devant être conservées.

Nous proposons donc d'utiliser une stratégie moins onéreuse en terme de stockage et de traitement. Celle-ci se base sur l'utilisation de l'algorithme EM défini précédemment pour fournir un premier estimé des paramètres du modèle (en utilisant un petit ensemble initial de courbes) puis sur l'utilisation d'un algorithme de gradient stochastique, cf. Benveniste et al. (1987), pour mettre à jour les paramètres après l'acquisition de chaque nouvelle courbe. Cette stratégie ne requiert pas le stockage des courbes précédentes et conduit à un coût de traitement constant. Les paramètres du modèle sont donc mis à jour en utilisant la règle suivante :

$$\hat{\boldsymbol{\theta}}^{(i)} = \hat{\boldsymbol{\theta}}^{(i-1)} + \lambda_i \, S^{(i)}(\hat{\boldsymbol{\theta}}^{(i-1)}), \tag{9}$$

avec λ_i le pas du gradient stochastique qui peut être gardé fixe pour permettre une adaptation du modèle à des changements lents et progressifs des paramètres, ou bien décroitre au cours du temps si l'on considère que le système est stationnaire. Dans ce dernier cas, on pourra utiliser par exemple :

$$\lambda_i = \frac{1}{1+i} \tag{10}$$

Les désavantages d'une telle approche sont principalement liés à la difficulté de définir un pas de gradient adapté au problème à traiter. La section suivante présente une solution pour tester si une nouvelle courbe de mesure est anormale par rapport au modèle courant, utilisant elle aussi le vecteur score associé à une nouvelle courbe.

4 Test du score pour la détection d'anomalie

Les tests de type "score" sont particulièrement utiles pour résoudre des problèmes de tests lorsque l'on ne dispose d'information que sur l'hypothèse nulle, Rao (1948). Ils s'appuient sur la distribution asymptotique du vecteur score d'un modèle génératif. Dans le cadre de notre problème, ce test va permettre de quantifier à chaque nouvelle acquisition de courbe $\boldsymbol{y}_i$, les deux hypothèses suivantes :

$$\begin{aligned} H_0 &: \boldsymbol{\theta}_1 = \boldsymbol{\theta}_2 = \ldots = \boldsymbol{\theta}_{i-1} = \boldsymbol{\theta}_i \\ H_1 &: \boldsymbol{\theta}_1 = \boldsymbol{\theta}_2 = \ldots = \boldsymbol{\theta}_{i-1} \neq \boldsymbol{\theta}_i, \end{aligned}$$

ou i est l'indice de la courbe courante.

En effet, sous H_0 le vecteur score associé à une nouvelle courbe $\boldsymbol{y}_i$ suit asymptotiquement en $\hat{\boldsymbol{\theta}}$ (Jiang et Tanner (1999))) une loi normale donnée par :

$$S^{(i)}(\hat{\boldsymbol{\theta}}) \sim \mathcal{N}(0, I_f(\hat{\boldsymbol{\theta}})), \tag{11}$$

avec $I_f(\hat{\boldsymbol{\theta}})$ la matrice d'information de Fisher associée à l'observation d'un signal et $\hat{\boldsymbol{\theta}}$ l'estimateur du maximum de vraisemblance de $\boldsymbol{\theta}$:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta}; \boldsymbol{y}_1, \ldots, \boldsymbol{y}_i) \tag{12}$$

Cette approximation asymptotique n'est pas irréaliste à partir du moment où les courbes étudiées ont une dimensionalité suffisante. Dans notre application, si les courbes sont par exemple échantillonnées en $m = 500$ points cette approximation est tout à fait pertinente.

Grâce à cette distribution asymptotique, il est possible de définir à partir du vecteur de score associé à une courbe une statistique de test $S_i(\hat{\boldsymbol{\theta}})^T I_f(\hat{\boldsymbol{\theta}})^{-1} S_i(\hat{\boldsymbol{\theta}})$ scalaire qui suivra approximativement une loi du Chi deux à R degrés de liberté.

$$S^{(i)}(\hat{\boldsymbol{\theta}})^T \left(I_f(\hat{\boldsymbol{\theta}})\right)^{-1} S^{(i)}(\hat{\boldsymbol{\theta}}) \sim \chi^2(R), \tag{13}$$

Ainsi, la région critique approchée du test, pour une erreur de première espèce α, est donnée par :

$$W = \left\{ S^{(i)}(\hat{\boldsymbol{\theta}}) \,:\, S^{(i)}(\hat{\boldsymbol{\theta}})^T \left(I_f(\hat{\boldsymbol{\theta}})\right)^{-1} S^{(i)}(\hat{\boldsymbol{\theta}}) > h(R, 1-\alpha) \right\} \tag{14}$$

avec $h(R, 1-\alpha)$ le quantile d'ordre $1-\alpha$ de la loi du Chi deux à R degrés de liberté.

4.1 Mise en œuvre pratique

Pour mettre en œuvre ce test sur le problème de détection d'anomalie dans une séquence de courbes, il est nécessaire de disposer d'un estimé des paramètres du modèle courant $\hat{\boldsymbol{\theta}}$ et de la matrice de Fisher I_f associée à une courbe ou d'une estimation de celle-ci. Lorsque ces deux quantités sont estimées, il suffit de calculer la statistique de test et de la comparer au quantile désiré de la loi du Chi deux, pour déterminer si une nouvelle courbe est anormale. En ce qui concerne l'estimation de $\hat{\boldsymbol{\theta}}$ nous proposons d'utiliser l'algorithme de type gradient stochastique présenté précédemment. Et, pour ce qui est du problème plus complexe d'estimation de la matrice d'information de Fisher, nous proposons d'utiliser une solution assez simple basée sur l'estimation de la matrice d'information de Fisher observée. En effet, le calcul de la matrice d'information de Fisher fait intervenir l'espérance de la matrice Hessienne H de la log-vraisemblance du modèle considéré ;

$$I_f(\hat{\boldsymbol{\theta}}) = -E[H(\hat{\boldsymbol{\theta}})].$$

Cette quantité pose problème dans les modèles à variable latentes et de plus, des arguments théoriques et pratiques soutiennent l'utilisation de l'information de Fisher

observée $I_o(\hat{\boldsymbol{\theta}})$, cf. Guddattu et Rao (2009); Bradley et Hinkey (1978), qui correspond simplement à :

$$I_o(\hat{\boldsymbol{\theta}}) = -H(\hat{\boldsymbol{\theta}}).$$

Lorsque l'algorithme EM est utilisé pour estimer les paramètres d'un modèle, différentes solutions sont envisageables pour calculer ou estimer la matrice d'information de Fisher observée, cf. Mc Lachlan et Krishnan (1996) chap. 4. Cependant, lorsque l'on dispose de plusieurs réalisations i.i.d, ce qui est le cas ici puisque les courbes sont supposées indépendantes, une méthode assez simple consiste à estimer la matrice d'information de Fisher associée à une courbe par la covariance empirique du vecteur score. Ce qui donne, lorsque l'on se place en $\hat{\boldsymbol{\theta}}$ l'expression suivante :

$$\hat{I_o}(\hat{\boldsymbol{\theta}}) = \frac{1}{n} \sum_{i=1}^{n} S^i(\hat{\boldsymbol{\theta}}) S^i(\hat{\boldsymbol{\theta}})^T \tag{15}$$

Cet estimateur est un estimateur convergent de la matrice d'information de Fisher (Mc Lachlan et Krishnan (1996)).

Les différents éléments que nous venons de présenter permettent de proposer un algorithme de détection en ligne de signaux anormaux qui tire partie de chacun d'eux.

4.2 Algorithme de détection en ligne de signaux anormaux

L'algorithme de détection procède de la manière suivante. Le score de Fisher associé à la courbe est calculé en utilisant l'équation 6 ainsi que la statistique de test (cf. équation 13). Si le test est négatif le modèle ainsi que l'estimé de la matrice d'information de Fisher sont mis à jour, sinon la courbe est déclarée comme anormale. Ces différentes étapes sont résumées dans l'algorithme 1. Cet algorithme suppose que l'initialisation est pertinente c'est à dire que l'on dispose d'une première estimation du modèle de bon fonctionnement qui est suffisamment proche de la véritable valeur du paramètre ; le test du score suppose en effet que l'on calcule le vecteur score associé à une nouvelle courbe en $\hat{\boldsymbol{\theta}}$. Nous proposons donc d'utiliser un petit nombre n_{app} de courbes pour fournir, grâce à l'algorithme EM décrit en section 2, cette première estimation. En ce qui concerne l'initialisation de l'estimé de la matrice d'information de Fisher, il est possible d'utiliser l'équation 15 si n_{app} est supérieur à la dimension du modèle. Dans le cas contraire, il est possible de régulariser la matrice estimée en ajoutant sur la diagonale de celle-ci une quantité positive afin d'obtenir une matrice non singulière.

Algorithm 1 Pseudo-code de l'algorithme de détection de signaux anormaux.

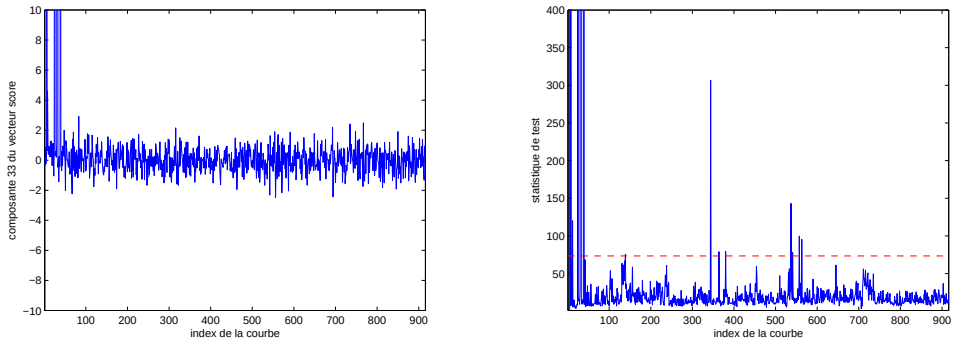
Entrées : $\lambda, \alpha, I_o^{(0)}, \boldsymbol{\theta}^{(0)}, n_{app}$
pour chaque nouvelle courbe **faire**
  Calcul du vecteur score de la courbe et de la statistique de test
  $S^{(i)}(\hat{\theta}) = \frac{\partial L(\hat{\boldsymbol{\theta}};\boldsymbol{y}_i)}{\partial \boldsymbol{\theta}}$
  $b^{(i)} = S^{(i)}(\hat{\theta})^T \hat{I}_o{}^{-1} S^{(i)}(\hat{\theta})$
  si $b^{(i)} < h(r, 1-\alpha)$ **alors**
    Mise à jour des paramètres et de l'estimée de la matrice d'information de Fisher
    $n_{app} \leftarrow n_{app} + 1$
    $\hat{I}_o \leftarrow \left(\hat{I}_o(n_{app}-1) + S^{(i)}(\hat{\theta})S^{(i)}(\hat{\theta})^T\right)/n_{app}$
    $\hat{\boldsymbol{\theta}} \leftarrow \hat{\boldsymbol{\theta}} + \lambda S^{(i)}(\hat{\boldsymbol{\theta}})$
  sinon
    Déclarer la courbe comme anormale
  fin si
fin pour

Les réglages de l'algorithme proposé sont le pas du gradient stochastique, ainsi que l'erreur de première espèce désirée.

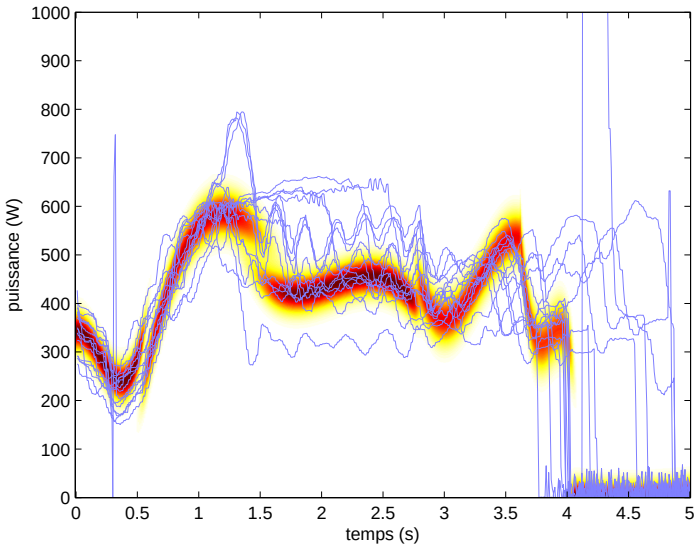
5 Détection d'anomalie sur les aiguillages

La méthodologie proposée a été appliquée à une séquence de 916 courbes issues de la surveillance d'un appareil de voie particulier de la SNCF. Cette séquence couvre la période allant de Juillet 2005 à Octobre 2007 avec une fréquence de mesure de deux à quatre courbes par jours. Les résultats de l'estimation initiale sont présentés en figure 2 avec un modèle à 6 composantes régressives cubiques, faisant donc intervenir 40 paramètres. Ce modèle à été sélectionné à l'aide du critère BIC. La démarche précédente a été mise en œuvre en utilisant 5 courbes pour fournir le premier estimé des paramètres du modèle à l'aide de l'algorithme EM. Le pas du gradient stochastique à été fixé a priori à 0.01 et la probabilité d'erreur de première espèce à également à 0.001. La figure 3 présente les résultats obtenus. Enfin, la matrice de Fisher utilisée pour l'initialisation a été régularisée car elle était singulière.

Nous pouvons observer sur la figure 3 (a) que le score à un comportement relativement stable, excepté pour quelques courbes présentant un score de Fisher s'écartant largement de 0. Ces mêmes indices de courbes sont repérable sur le graphique présentant la statistique de test, figure 3 (b), qui agrège l'information en provenances de toutes les composantes du vecteur score. Les 15 courbes qui dépassent le seuil de détection fixé (6,8,12,25,31,32,39,139,344,364,380,537,541,557,563) présentent effectivement des profils différents des courbes associés au modèle nominal comme le montre la figure 3 (c) qui permet de comparer celle-ci au modèle nominale estimé durant la séquence.



(a) (b)



(c)

FIG. 3 – *Evolution d'une composante du vecteur score le long de la séquence, en l'occurrence le coefficient associé au monôme d'ordre de 2 du dernier sous modèle de régression (a). Évolution de la statistique de test le long de la séquence ; le seuil de détection est en pointillé rouge (b). Courbes détectées comme anormales (c) en bleu et modèle de référence à la fin de la séquence en dégradé de couleur (pour chaque abscisse x_j la densité du modèle est calculée et représentée à l'aide d'une échelle de couleur, du blanc pour les probabilités faibles au rouge pour les fortes probabilités).*

6 Conclusion

Cet article a présenté une méthodologie pour détecter dans une séquence de courbes, des courbes atypique s'écartant de la courbe moyenne. Cette méthodologie a été ap-

pliquée au diagnostic et au suivi de point de fonctionnement des aiguillages qui sont des organes importants et sensibles de l'infrastructure ferroviaire. Elle a permis de mettre en évidence des courbes atypiques dans la séquence étudiée. Des travaux complémentaires restent à être menés sur l'influence des différentes approximations faites (estimation de la matrice de Fisher, utilisation d'un test asymptotique) sur l'erreur de première espèce ainsi que sur la question du choix d'un modèle optimal. Enfin, des expériences complémentaires sur données réelles et simulées sont en projet.

Références

Benveniste, A., M. Metivier, et P. Priouret (1987). *Algorithme adaptatif et approximations stochastiques*. Masson.

Bradley, E. et D. Hinkey (1978). Assessing the accuracy of the maximum likelihood estimator : Observed versus expected fisher information. *Biometrika 65*(3), 457–483.

Carvalho, A. X. et M. A. Tanner (2006). Modelling nonlinearities with mixtures of experts of time series models. *International Journal of Mathematics and Mathematical Sciences 9*, 1–22.

Chamroukhi, F., A. Samé, G. Govaert, et P. Aknin (2010). A hidden process regression model for functional data description. application to curve discrimination. *Neurocomputing 73*(7-9), 1210–1221.

Dempster, A. P., N. M. Laird, et D. B. Rubin (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society B 39*, 1–38.

Green, P. (2003). Iteratively reweighted least squares for maximum likelihood estimation,and some robust and resistant alternatives. *Journal of the Royal Statistical Society B 46*, 149–192.

Guddattu, V. et A. Rao (2009). On the use of observed fisher information in wald and score test.

Horvarth, L. et E. Parzen (1994). *Limit theorem for fisher score change processes*, Volume 23 of *lecture notes*, pp. 157–169. IMS.

Jaakkola, T., M. Diekhans, et D. Haussler (1999). Using the fisher kernel method to detect remote protein homologies. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, pp. 149–158. AAAI Press.

Jaakkola, T. et D. Haussler (1998). Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems 11*, pp. 487–493.

Jiang, W. et M. A. Tanner (1999). On the asymptotic normality of hierarchical mixtures-of-experts for generalized linear models. *IEEE Trans. on Information Theory 46*, 1005–1013.

Jordan, M. I. et R. A. Jacob (1994). Hierarchical mixtures of experts and the EM algorithm. *Neural Computation 6*, 181–214.

Mc Lachlan, G. J. et T. Krishnan (1996). *The EM algorithm and Extension*. Wiley.

Rao, C. R. (1948). Large sample tests of statistical hypotheses concerning several parameters with applications to problems of estimation. *Mathematical Proceedings of the Cambridge Philosophical Society 44*, 50–57.

Salakhutdinov, R., T. T. Roweis, et Z. Ghahramani (2003). Optimization with EM and expectation-conjugate-gradient. In *International Conference on Machine Learning (ICML)*, pp. 672–679.

Samé, A., F. Chamroukhi, et P. Aknin (2008). Détection séquentielle de défauts sur des signaux de manoeuvres d'aiguillage. In *Workshop Surveillance, Sûreté et Sécurité des Grands Systèmes*.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics 6*, 461–464.

Annexe 1

Les équations de mise à jour des paramètres du modèle de régression à processus logistique latent utilisés lors de l'étape M de l'algorithme EM sont les suivantes :

$$\boldsymbol{\beta}_k^{(q+1)} = (\mathcal{K}^T W_k^{(q)} \mathcal{K})^{-1} \mathcal{K}^T W_k^{(q)} \mathcal{Y}, \tag{16}$$

où $W_k^{(q)}$ est la matrice diagonale ayant pour éléments diagonaux

$$(t_{11k}^{(q)}, \ldots, t_{1mk}^{(q)}, \ldots, t_{n1k}^{(q)}, \ldots, t_{nmk}^{(q)}),$$

et

$$\mathcal{K} = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix} \text{ et } \mathcal{Y} = \begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_n \end{bmatrix}.$$

Notons que les matrices $W_k^{(q)}$, $\mathcal{K}$ et $\mathcal{Y}$ sont de dimensions respectives $nm \times nm$, $nm \times (p+1)$ et $nm \times 1$. La maximisation de Q_2 par rapport à σ_k^2 fournit quant à elle les estimations :

$$(\sigma_k^2)^{(q+1)} = \frac{\sum_{i=1}^n \sum_{j=1}^m t_{ijk}^{(q)} (y_{ij} - \boldsymbol{\beta}_k^T \boldsymbol{x}_j)^2}{\sum_{i=1}^n \sum_{j=1}^m t_{ijk}^{(q)}}. \tag{17}$$

Annexe 2

Le vecteur score du modèle correspondant à une nouvelle courbe $\boldsymbol{y}_i$ est donnée par les équations suivantes :

$$\begin{aligned}
\frac{\partial L(\boldsymbol{\theta};\boldsymbol{y}_i)}{\partial \boldsymbol{w}_{k0}} &= E\left[\frac{\partial L_c(\boldsymbol{\theta};Y_i,Z_i)}{\partial \boldsymbol{w}_{k0}}|\boldsymbol{y}_i,\boldsymbol{\theta}\right] \\
&= \sum_{j=1}^{m}\big(t_{ijk}-\pi_k(x_j;\boldsymbol{w})\big) \qquad (18)\\
\frac{\partial L(\boldsymbol{\theta};\boldsymbol{y}_i)}{\partial \boldsymbol{w}_{k1}} &= E\left[\frac{\partial L_c(\boldsymbol{\theta};Y_i,Z_i)}{\partial \boldsymbol{w}_{k1}}|\boldsymbol{y}_i,\boldsymbol{\theta}\right] \\
&= \sum_{j=1}^{m}\big(t_{ijk}-\pi_k(x_j;\boldsymbol{w})\big)x_j \qquad (19)\\
\frac{\partial L(\boldsymbol{\theta};\boldsymbol{y}_i)}{\partial \boldsymbol{\beta}_k} &= E\left[\frac{\partial L_c(\boldsymbol{\theta},Y_i,Z_i)}{\partial \boldsymbol{\beta}_k}|\boldsymbol{y}_i,\boldsymbol{\theta}\right] \\
&= \frac{1}{\sigma_k^2}\sum_{j=1}^{m} t_{ijk}\left(y_{ij}-\boldsymbol{\beta}_k^T\boldsymbol{x}_j\right)\boldsymbol{x}_j \qquad (20)\\
\frac{\partial L(\boldsymbol{\beta};\boldsymbol{y}_i)}{\partial \sigma_k} &= E\left[\frac{\partial L_c(\boldsymbol{\theta},Y_i,Z_i)}{\partial \sigma_k}|\boldsymbol{y}_i,\boldsymbol{\theta}\right] \\
&= \sum_{j=1}^{m} t_{ijk}\frac{\left((y_{ij}-\boldsymbol{\beta}_k^T\boldsymbol{x}_j)^2-\sigma_k^2\right)}{\sigma_k^3} \qquad (21)
\end{aligned}$$

avec $t_{ijk}=\frac{\pi_k(\boldsymbol{x}_j;\boldsymbol{w})\phi\left(y_{ij};\boldsymbol{\beta}_k^T\boldsymbol{x}_j,\sigma_k^2\right)}{\sum_{\ell=1}^{K}\pi_\ell(\boldsymbol{x}_j;\boldsymbol{w})\phi\left(y_{ij};\boldsymbol{\beta}_\ell^T\boldsymbol{x}_j,\sigma_\ell^2\right)}$

Summary

This paper deals with the automatic recognition of abnormal curves in curves stream. The proposed methodology use jointly a generative model which enable the estimation of the nominal behavior and a score test which is used to decide if a new curve is abnormal or not. An example of this methodology will be detailed in the context of rail switch monitoring. In this application curves correspond to electrical power consumed during the switch process and are recorded on a daily basis.

Comportement comparatif des méthodes de clustering incrémentales et non incrémentales sur les données textuelles hétérogenes

Raghvendra Mall *, Mumtaz Ahmad **
Jean-Charles Lamirel***

*CDE, IIIT Hyderabad, Hyderabad, India,
raghvendra.mall@research.iiit.ac.in,
http://www.iiit.net
**CASSIS, Loria, INRIA, Nancy, France,
mumtaz.ahmad@loria.fr,
http://www.loria.fr
***TALARIS, Loria, INRIA, Nancy, France,
lamirel@loria.fr,
http://www.loria.fr

Résumé. Les algorithmes de clustering neuronaux affichent de très bonnes performances pour l'analyse de distributions-test, aussi bien que pour celle de données textuelles homogènes. Cela vaut plus particulièrement pour les versions adaptatives récentes de ces algorithmes, tel que l'algorithme de gaz neuronal croissant incrémental standard (IGNG), ou celui basé sur la maximisation d'étiquetage (IGNG-F). Nous montrons cependant qu'une diminution drastique des performances de tous ces algorithmes, aussi bien que de celles des algorithmes plus classiques, peut être observée quand des corpus de données hétérogènes, comme des corpus textuels polythématiques, sont considérés en entrée. Nous proposons ensuite différentes adaptations de l'algorithme de gaz neuronal croissant incrémental combinant l'exploitation incrémentale de la connaissance fournie par l'étiquetage des clusters avec des mesures de distance classiques entre clusters et données. Nous montrons que ces adaptations apportent une amélioration très significative des performances sur tous les corpus, et en particulier sur les corpus hétérogènes.

1 Introduction

Les méthodes de clustering neuronales sont basées sur le principe de l'exploitation de relations de voisinage entre les clusters, qu'elles soient prédéfinies (topologie fixe), comme les "cartes auto-organisatrices" (SOM) (Kohonen (1982)), ou dynamiques (topologie libre), comme les gaz de neurones statiques (NG) (Martinetz et Schulten (1991)) ou les gaz de neurones croissants (GNG) (Fritzke (1995)). Comparativement au méthodes de clustering usuelles, comme K-means, cette stratégie les rend moins sensibles aux conditions initiales, ce qui repré-

sente un avantage déterminant dans le cadre de l'analyse des données fortement multidimensionnelles et éparses, comme les données textuelles.

Récemment Prudent et al. (Prudent et Ennaji (2005)) ont proposé un algorithme de gaz neuronal croissant incrémental (IGNG) exploitable pour des tâches générales de clustering. Il représente une adaptation de l'algorithme GNG qui relaxe la contrainte de l'évolution périodique du réseau inhérente à ce dernier algorithme. Les auteurs de la méthode ont montré qu'IGNG produisait de meilleurs résultats que les méthodes neuronales existantes, et également que les méthodes de clustering non neuronales usuelles, sur des distributions-test. Cependant, en utilisant des mesures génériques de qualité pour évaluer les clusters produits par IGNG, Lamirel et al. (Lamirel et al. (2010)) ont également montré que la méthode produisait des résultats dégénérés sur des corpus de données hétérogènes et éparses, comme des corpus de données textuelles polythématiques, lesquelles s'avèrent particulièrement difficiles à clusteriser. Pour faire face à ce problème, ces derniers auteurs ont proposé un nouvel algorithme de gaz neuronal croissant incrémental basé sur la maximisation de l'étiquetage des clusters (IGNG-F) (Lamirel et al. (2010)). Cette stratégie permet de s'affranchir de l'utilisation d'une mesure de distance usuelle pour déterminer un gagnant pendant l'apprentissage, en considérant l'approche de maximisation d'étiquetage comme l'unique processus de sélection du gagnant. Cependant, comme nous le prouvons également dans la suite de ce papier, bien que cette méthode se comporte mieux qu'IGNG sur les corpus de données assez complexes, et qu'elle présente l'avantage de fonctionner sans aucun paramètre, elle ne s'avère pas aussi efficace que prévu quand la complexité de clustering d'un corpus continue à croitre, avec notamment l'apparition de propriétés de jointures hétérogènes entre les groupes potentiels de données. Aussi, nous présentons également ci-après plusieurs adaptations de l'approche IGNG-F basées sur la combinaison de critères de maximisation d'étiquetage des clusters avec des critères basés sur la distance. Les résultats obtenus sur tous les corpus exploités, et particulièrement sur les plus complexes, reflètent clairement les avantages de nos nouvelles adaptations. Des mesures génériques de qualité décrites ci-après sont employées pour évaluer ces résultats.

2 Evaluation de la qualité du clustering

Les méthode classiques d'évaluation de la qualité du clustering sont basées sur l'inertie inter-clusters et l'inertie intra-cluster, présentés dans (Davies et Bouldin (1979)). Mais il a été montré par Lamirel et al. (Lamirel et al. (2004)) que les mesures d'inertie, ou leur dérivées, qui sont basées sur les profils des clusters, sont à la fois fortement biaisées et très dépendantes de la méthode de clustering utilisée. Par ailleurs, ces mesures ne permettent pas de détecter des résultats de clustering dégénérés, comprenant des "clusters poubelles" de très grande taille, conjointement à des "clusters poussières" de très petite taille (Lamirel et al. (2010)). Pour faire face à ce problème, les auteurs précédents ont proposé une nouvelle approche basée sur le $Rappel$ et la $Précision$ non supervisées. Le calcul de ces indices de qualité s'opère à partir des propriétés ou des étiquettes propres des clusters, dont les poids sont maximisés par des données associées à ces derniers. La $Précision$ des propriétés propres d'un cluster agit en tant que mesure d'homogénéité de contenu des clusters (mesure intra), alors que le $Rappel$ desdites propriétés agit en tant que mesure de discrimination entre les contenus des différents clusters (mesure inter). Ces mesures ont été prouvées indépendantes de la méthode de clustering. Les

mesures de $Macro\text{-}Rappel$ et $Macro\text{-}Précision$ sont des mesures orientées par les clusters, étant donné qu'elles fournissent les valeurs moyennes du $Rappel$ et de la $Précision$ pour chaque cluster. Les mesures complémentaires de $Micro\text{-}Rappel$ et de $Micro\text{-}Précision$, orientées par les propriétés, sont obtenues en faisant la moyenne des valeurs de $Rappel$ et de $Précision$ des propriétés propres indépendamment de la structure des clusters. La différence entre $Macro$ et $Micro$ mesures de permet de confirmer l'apparition des résultats de clustering dégénérés (Lamirel et al. (2010)). Une mesure additionnelle de $Micro\text{-}Précision\ Cumulée$ (CMP) se concentre sur l'hétérogénéité des plus gros clusters. L'algorithme IGNG-F est un algorithme de maximisation de vraisemblance qui exploite une combinaison harmonique des mesures de qualité précédemment décrites (c.-à-d. une F-mesure d'étiquetage) comme un substitut à la distance classique dans le processus de clustering (Lamirel et al. (2010)). Les auteurs ont montré qu'il fournissait les meilleurs résultats pour le clustering de corpus de données homogènes.

3 Adaptations de l'algorithme IGNG-F

Le problème majeur que rencontre l'algorithme IGNG-F dans le cas du traitement de corpus de données très hétérogènes est qu'il peut associer à un cluster des données dont les étiquettes sont complètement différentes de celles qui existent dans les données préalablement associées à ce dernier. Ceci mène à une forte diminution des performances de l'algorithme, étant donné que des étiquettes (et des données) susceptibles d'appartenir à différents clusters sont ainsi agglomérées ensembles. Pour faire face à ce problème, nous proposons trois adaptations différentes de l'approche d'IGNG-F que nous décrivons ci-après.

3.1 IGNG-Fv1

Nous employons des critères basés sur la distance pour limiter le nombre de prototypes (c.-à-d de clusters) considérés comme candidats pour chaque nouvelle donnée entrante (cf. Annexe). Ceci revient à définir un seuil de voisinage pour chaque nouvelle donnée, stratégie qui fait défaut dans l'approche IGNG-F. Cette stratégie se rapproche de celle de l'exploitation du paramètre σ de variance du nuage des données implantée par l'algorithme IGNG, la seule différence étant que le nouveau critère proposé est orienté-utilisateur et peut ainsi être modifié en conséquence. Nous avons observé expérimentalement qu'il existait des valeurs singulières de ce critère, dépendantes des corpus, qui pouvaient être employées comme seuil, valeurs au delà desquelles tous les prototypes avaient tendance à être choisis dans le voisinage.

3.2 IGNG-Fv2 and IGNG-Fv3

Nous employons les critères basés par distance pour avoir un seuil de voisinage pour chaque nouvelle donnée entrante. Une autre adaptation importante est que le $F\text{-}mesure$ d'étiquetage qui doit être calculée en tenant compte de cette donnée ne prend plus en compte les nouvelles étiquettes spécifiques à cette donnée. Les nouvelles étiquettes sont celles qui ne participaient pas à la liste d'étiquettes d'un prototype avant l'"augmentation" de ce prototype avec la donnée entrante. Cette stratégie évitera notamment d'associer à un prototype des données

dont les étiquettes sont complètement différentes de celles existants préalablement dans ce dernier, et empêchera ainsi la formation de clusters hétérogènes. Nous modifions l'étape du calcul de l'influence d'étiquetage des données d'entrée sur les prototypes existants, comme proposée dans l'algorithme IGNG-F (Lamirel et al. (2010)). Dans l'étape de maximisation d'étiquetage de cet algorithme une étiquette est attachée au prototype qui a la $F\text{-}mesure$ maximale pour cette étiquette. Cependant, dans notre cas, s'il y a plus d'un maximiseurs (c.-à-d. plus d'un prototype candidat) nous attachons l'étiquette à tous les maximiseurs, exploitant ainsi une stratégie de clustering partiellement recouvrante. Considérant maintenant le comportement de général de l'algorithme, quand une augmentation de $F\text{-}mesure$ est induite sur plusieurs prototypes par la donnée entrante, nous choisissons le prototype gagnant selon la formule :

$$Kappa1 = \Delta_{L-F_x(P_1)} \times SL(P_1) - \frac{EucDist(P_1, x)}{criteria \times 10} \quad (1)$$

$$Kappa2 = \Delta_{L-F_x(P_1)} \times SL(P_1) + \frac{CosineSim(P_1, x)}{10} \quad (2)$$

La mesure $Kappa1$ est employée pour IGNG-Fv2 et $Kappa2$ est employée pour IGNG-Fv3. En procédant de la sorte, nous cherchons à maximiser l'augmentation de $F\text{-}mesure$ (Δ) tout en maximisant conjointement le nombre d'étiquettes partagées (par exemple entre le prototype P_1 et la donnée x). Nous nous servons également de la distance euclidienne ($EucDist()$) ou de la similarité cosinus ($CosineSim()$) entre les profils de prototypes et le profil de la données entrante. Plus petite est la distance euclidienne, ou plus grande est la similarité cosinus, plus grande est la probabilité de la donnée entrante d'être associée au prototype. La variable $critére$ (pour $Kappa1$) est identique à celle utilisée pour limiter la taille du voisinage dans l'algorithme IGNG-Fv1. Elle est également employée afin de réduire l'influence de la mesure de distance, de manière à ce que la valeur de $Kappa1$ demeure non négative. De manière similaire, pour obtenir des contributions égales, nous divisons la valeur de $CosineSim()$ par 10 (pour $Kappa2$).

4 Evaluation

4.1 Données de test

Pour nos expériences, nous employons 3 corpus de données différents, de complexité de clustering croissante, à savoir les corpus Total-Utilisation, PROMTECH et Lorraine. Le corpus Total-Utilisation est constitué de 220 notices de brevet liés à la technologie des huiles moteurs et il couvre uniquement le champ élémentaire "Utilisation" desdites notices. Les 2 autres corpus sont issus de la base de données bibliographique PASCAL de l'INIST. Le corpus PROMTECH est constitué de 1794 notices bibliographiques liées à la recherche en optoélectronique au cours de la période 1996-1999. Le corpus Lorraine est un ensemble de 1920 notices bibliographiques couvrant tous les domaines de recherche impliquant au moins un laboratoire de recherche Lorrain sur une année. Le tableau 1 montre que, pour le corpus homogène Total-Utilisation, le nombre moyen d'étiquettes, ou mots-clés, par document est faible (3.268), et que, parmi toutes les étiquettes, seules 62 sur 234 participent au recouvrement, ce qui facilite le processus de clustering. Par contre, pour les corpus PROMTECH et Lorraine, les étiquettes

couvrent des sujets très variés, fournissant une grande hétérogénéité de matières sur les périodes considérées. Le nombre moyen d'étiquettes recouvrantes par document est extrêmement élevé pour le corpus Lorraine (1.852). Par ailleurs, pour ce dernier corpus, le tableau 1 indique également que le nombre de termes-pivots distincts est très élevé, même dans les motifs, ou groupes d'étiquettes recouvrantes communes, de taille importante. Il est bien plus élevé que pour les corpus PROMTECH ou Total-Utilisation, indiquant son hétérogénéité plus importante. Généralement plus importante sera l'hétérogénéité du recouvrement, et parallèlement, plus élevé sera le recouvrement moyen entre les documents, plus grande sera la complexité de clustering, étant donné que dans ce dernier cas une méthode de clustering aura tendance à agglomérer dans un même groupe l'information appartenant à différents clusters.

Pour chaque méthode neuronale, nous avons opéré diverses expériences en faisant varier le nombre de clusters pour les méthodes statiques, et les paramètres de voisinage pour les méthodes incrémentales. Nous avons finalement gardé les meilleurs résultats de clustering de chaque méthode, en considérant les valeurs de $Rappel$-$Précision$, F-$mesure$ et $Micro$-$Précision\ cumulée$ (CMP).

4.2 Résultats

Le tableau 2 (a) fournit les valeurs de $Macro$-F-$Mesure$, de $Micro\ F$-$Mesure$ et de CMP pour le corpus homogène Total-Utilisation. Le nombre de clusters représente le nombre réel de clusters non vides, parmi l'ensemble des clusters effectivement exploités par les méthodes pour grouper les données. C'est les valeurs de CMP qui illustrent le mieux la différence entre les diverses approches. Les algorithmes SOM, K-Means, NG et GNG obtiennent de bonnes valeurs de $Macro$ et de $Micro\ F$-$Mesure$, mais néanmoins des valeurs de CMP inférieures à celles des méthodes de la famille IGNG, à l'exception de la méthode IGNG-Fv1. En effet, bien que nous ayons des valeurs élevées de CMP pour IGNG, IGNG-F, IGNG-Fv2 et IGNG-Fv3, la valeur de CMP de IGNG-Fv1 est extrêmement faible, ce qui signifie que les propriétés associées aux clusters produits par cette méthode ne permettent pas de distinguer efficacement entre ces derniers. Le nombre moyen d'étiquettes par document étant peu élevé pour ce corpus (voir le tableau 1), cela résulte en des vecteurs normalisés d'étiquette associés aux données qui sont distants les uns des autres, comparativement à ceux des autres corpus. Ceci signifie également une distance plus élevée entre les données semblables, et peut représenter une des causes probables des mauvais résultats de la méthode IGNG-Fv1. En effet, corrélativement, nous voyons également que les meilleurs résultats sont obtenus par IGNG-Fv2 qui exploite spécifiquement des critères de voisinage et une procédure de sélection basée sur la mesure $Kappa1$ pour choisir les neurones gagnants pour chaque donnée.

Le tableau 2 (b) reflète que le fait que les valeurs $Macro$ et de $Micro\ F$-$Mesure$ sont faibles pour les méthodes SOM et K-Means, et presque semblables pour les algorithmes basés sur IGNG, en ce qui concerne le corpus PROMTECH. La mesure de CMP aide à nouveau à mieux différencier les méthodes. Nous observons que les valeurs de CMP sont élevées pour les méthodes SOM, K-Means, IGNG-Fv1 et IGNG-Fv2, mais que la valeur maximum est atteinte par l'algorithme GNG. Une des raisons principales est que cet algorithme exploite un processus exhaustif et itératif de minimisation des erreurs en traversant plusieurs fois l'en-

semble complet des données. La méthode K-Means obtient la deuxième meilleure valeur de CMP (0.35). Cependant, sa valeur de $Micro\text{-}Précision$ est la plus faible de toutes, illustrant un résultat moyen inférieur à celui de toutes les autres méthodes. Parmi les algorithmes incrémentaux de la famille IGNG, c'est IGNG-Fv2 qui obtient la meilleure valeur de CMP (0.318). L'algorithme IGNG-Fv2 obtient de plus une valeur $Micro\text{-}Précision$ élevée, illustrant le fait que sa qualité moyenne de groupement est tout à fait correcte, bien qu'il puisse subsister des clusters qui agglomèrent indument les données, et donc les étiquettes

Le tableau 2 (c) illustre les difficultés énormes que rencontrent les diverses méthodes pour clusteriser le corpus Lorraine, le plus hétérogène. Les méthodes K-means, IGNG, IGNG-Fv1 et à un moindre degré, la méthode GNG, obtiennent des valeurs très basses de CMP, indiquant des résultats dégénérés (matérialisant la présence de clusters poubelles attirant la majeure partie des données, parallèlement à celle de clusters poussières représentant des groupes marginaux ou non complètement formés). Les seuls comportements conformes aux attentes sont ceux des méthodes SOM et IGNG-Fv2. Les meilleurs résultats sur cet ensemble complexe de données hétérogènes sont néanmoins obtenus avec la méthode IGNG-Fv2 (valeur de CMP de 0.166). Ces bons résultats sont principalement dus au choix du neurone gagnant basé sur la mesure $Kappa1$ qui met en lumière l'importance de prendre en considération de manière coordonnée le nombre maximum d'étiquettes partagées entre un cluster et une donnée entrante, l'augmentation maximale de la $F\text{-}mesure$ d'étiquetage générée par cette donnée, et, initialement, la distance maximale entre la donnée et le cluster gagnant.

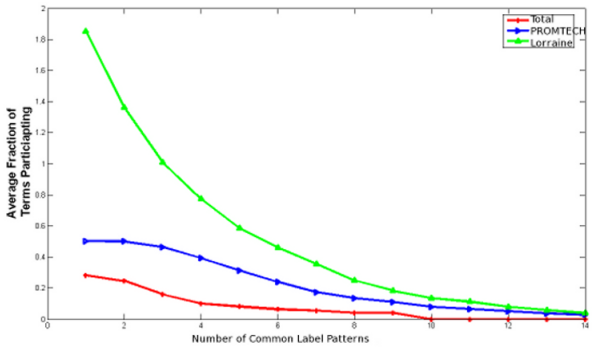


FIG. 1 – *Proportion moyenne d'étiquettes distinctes apparaisant dans les motifs d'étiquettes recouvrants pour les 3 corpus expérimentaux.*

5 Conclusion

L'utilisation de l'algorithme IGNG en combinaison avec la connaissance fournie par l'étiquetage courant des clusters, obtenue de manière incrémentale, et avec celle de mesures de similarité basées sur la distance (IGNG-Fv2) permet d'obtenir un accroissement significatif des performances de clustering, ainsi que des résultats stables, pour tous types de corpus textuels. L'approche proposée fournit par ailleurs les meilleurs résultats pour le corpus textuel

TAB. 1 – *Synthèse des informations sur les 3 corpus expérimentaux.*

Corpus	Mot-clés	Documents	Fréquence Seuil	Nb. moyen étiq. par document	Recouvr. total Nb. étiquettes	Recouvr. moyen Etiquettes par document
Total	234	220	2	3.268	62	0.28
PROMTECH	903	1794	3	8.12	903	0.503
Lorraine	3556	1920	2	8.99	3556	1.852

Clustering Method	No of clusters	F-Measure Macro	F-Meausre Micro	Cumulative Micro-Precision
SOM	90 (196)	0.766719	0.692499	0.530187
KMeans	53 (55)	0.758615	0.594195	0.291856
NG	82(96)	0.815711	0.714835	0.6
GNG	86 (93)	0.784056	0.755068	0.670653
IGNG	86 (87)	0.837917	0.796471	0.729932
IGNG-F	68 (81)	0.83054	0.745341	0.690571
IGNG-Fv1	70 (72)	0.823281	0.690168	0.184432
IGNG-Fv2	93 (96)	0.855971	0.798538	0.731729
IGNG-Fv3	91 (96)	0.851049	0.793669	0.723774

(a) Clustering Results on Homogeneous Total Dataset

Clustering Method	No of clusters	F-Measure Macro	F-Meausre Micro	Cumulative Micro-Precision
SOM	311 (361)	0.368056	0.353343	0.309484
Kmeans	186(185)	0.41429	0.375741	0.356678
GNG	215 (215)	0.440858	0.407637	0.48383
IGNG	296 (300)	0.465731	0.416523	0.242226
IGNG-F	300 (300)	0.490176	0.430829	0.249885
IGNG-Fv1	294 (295)	0.500796	0.430363	0.308201
IGNG-Fv2	255 (255)	0.476337	0.418685	0.318215
IGNG-Fv3	269 (270)	0.491436	0.422971	0.275672

(b) Clustering Results on PROMTECH DATASET

Clustering Method	No of clusters	F-Measure Macro	F-Meausre Micro	Cumulative Micro-Precision
SOM	216 (225)	0.35142	0.295278	0.136492
K-Means	198 (400)	0.392339	0.288297	0.027891
GNG	136 (136)	0.327476	0.256538	0.07864
IGNG	198 (198)	0.339541	0.279246	0.028195
IGNG-F	198 (198)	0.351322	0.280931	0.030203
IGNG-Fv1	198 (198)	0.351322	0.280931	0.030203
IGNG-Fv2	198 (198)	0.34936	0.278785	0.166262
IGNG-Fv3	198 (198)	0.352215	0.280295	0.028852

(c) Clustering Results on Cmp-Kma Heterogeneous Dataset

FIG. 2 – *Qualité de clustering des différents algorithmes sur les 3 corpus expérimentaux.*

hétérogène Lorraine, ce qui prouve sa pertinence dans un contexte incrémental, étant donné que ce corpus représente une approximation statique des variations de sujets qui pourraient se produire dans les scénarios variables dans le temps.

Références

Davies, D. L. et D. W. Bouldin (1979). A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-1*(2), 224–227.

Fritzke, B. (1995). A growing neural gas network learns topologies. In *Advances in Neural Information Processing Systems 7*, pp. 625–632. MIT Press.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics 43*(1), 59–69.
Anglais

Lamirel, J.-C., Z. Boulila, M. Ghribi, P. Cuxac, et C. François (2010). A new incremental growing neural gas algorithm based on clusters labeling maximization : application to clustering of heterogeneous textual data. In *23rd International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems (IEA-AIE 2010)*, Cordoba Espagne.

Lamirel, J.-C., C. François, S. A. Shehabi, et M. Hoffmann (2004). New classification quality estimators for analysis of documentary information : Application to patent analysis and web mapping. *Scientometrics 60*(3), 445–562.

Lamirel, J.-C., M. Ghribi, et P. Cuxac (2010). Exploitation d'une mesure de micro-précision cumulée non supervisée pour l'évaluation fiable de la qualité des résultats de clustering. In *42èmes Journées de Statistique*, Marseille France. Société Française de Statistique (SFdS).

Martinetz, T. et K. Schulten (1991). A "neural gas" network learns topologies. *Teuvo Kohonen, Kai MÃd'kisara, Olli Simula, and Jari Kangas, editors, Artificial Neural Networks, Elsevier, Amsterdam,*, 397–402.

Prudent, Y. et A. Ennaji (2005). An incremental growing neural gas learns topologies. In *Neural Networks, 2005. IJCNN '05. Proceedings. 2005 IEEE International Joint Conference on*, Volume 2, pp. 1211–1216 vol. 2.

Annexe

IGNG-Fv1 Algorithm

Notations
A_n represents the set of data associated to the neuron n.
L_n represents the list of labels whose $L\text{-}F\text{-}measure$ is maximized by the neuron n

Parameters
$\delta = 0$ (labeling quality evolution criteria)
$a_L =$ maximum age of a connection
$a_M =$ age of maturity of a neuron
$\epsilon_a =$ learning coefficient for the winner

ϵ_b = learning coefficient for the other neurons
k = neighbourhood range criteria
Initialization
C = set of the connections between the neurons = ϕ
N = set of initial mature neurons = ϕ
E = set of initial embryo neurons = ϕ

While(!Stopping Criteria) do

1. **Random choice of an input data**
   A new input data point x is chosen with probability $p(x) = \frac{1}{D}$
2. **Neighbourhood selection criteria**
   Select all the existing neurons which satisfy the neighbourhood criteria - $Distance(n, x) > k$, where $Distance()$ represents the euclidean distance between a neuron (here n) and the data point (here x).
3. **Calculation of labeling influence on selected neurons**
   $\forall n \in (N \cup E)$, calculate
   $\Delta_{L-F_x(P_1)} = \frac{1}{|L_{n^{+x}}|} \sum_{l \in L_{n^{+x}}} L - F_{n^{+x}}(l) - \frac{1}{|L_n|} \sum_{l' \in L_n} L - F_n(l')$
   where n^{+x} represents the neuron n extended with the data x such as : $A_{n^{+x}} = A_n \cup \{x\}$
   $/*$ This $\Delta_{L-F_x(P_1)}$ represents the $Kappa$ measure for IGNG-Fv1 $*/$
4. **Selection of winner neurons**
   $\forall n \in (N \cup E)$, calculate
   $\Delta_{L-F_x(P_1)} = \frac{1}{|L_{n^{+x}}|} \sum_{l \in L_{n^{+x}}} L - F_{n^{+x}}(l) - \frac{1}{|L_n|} \sum_{l' \in L_n} L - F_n(l')$
   where n^{+x} represents the neuron n extended with the data x such as : $A_{n^{+x}} = A_n \cup \{x\}$
5. **Selection of winner neurons**
   P_1 and P_2 are the winner and second winner neurons such as :
   $\forall n \in (N \cup E) - \{P_1, P_2\}, \Delta_{L-F_x(P_1)} > \Delta_{L-F_x(P_2)} > \Delta_{L-F_x(n)}$
6. **Evolution Phase**
   $/*$ No winner case $*/$
   If $(\Delta_{L-F_x(P_1)} < \delta)$ then
   – **Creation of a new embryo neuron n :**
     $E = E \cup \{n\}, A_n = \{x\}$
     $/*$ The new neuron gets the same reference vector and the same labels as that of data x $*/$
   EndIf
   $/*$ One single winner case $*/$
   If $(\Delta_{L-F_x(P_1)} > \delta > \Delta_{L-F_x(P_2)})$ then
   – **Creation of a new embryo neuron n :**
     $E = E \cup \{n\}, A_n = \{x\}$
     $/*$ The new neuron gets the same reference vector and the same labels as that of data x $*/$
   – **Creation of new connection :**
     $C = C \cup \{P_1, n^*\}$
     $/*$ A connection between the new neuron n and the winning neuron P_1 is created $*/$

EndIf
$/*$ More than one winning neuron $*/$
If $(\Delta_{L-F_x(P_2)} > \delta)$ then
- **Adding x to list of data associated to P_1 :**
  $A_{P_1} = A_{P_1} \cup \{x\}$
- **Updating age of connections with the winner**
  $\forall (x, y) \in C$, if $((x = P_1) \vee (y = P_1)$ then $Increment(Age(x, y))$
- **Adjusting reference vector of neurons**
  $w_{P_1}(t+1) = w_{P_1}(t+1) + \epsilon_a(x - w_{P_1}(t))$
  $w_i(t+1) = w_i(t+1) + \epsilon_b(x - w_i(t))$
- **For each neighbours of P_1**
  (a) **Updating connection between P_1 and P_2**
  If $(((x, y) \in C | x = P_1 \vee y = P_2) = \phi)$ then $C = C \cup \{P_1, P_2\}$
  else $Age((P_1, P_2)) = 0$
  $/*$ If a connection exists between P_1 and P_2 its age is set to 0 otherwise it is created $*/$
  (b) **Removing connections**
  $\forall (x, y) \in C$, if $Age((x, y)) > a_L)$ then $C = C - \{(x, y)\}$
  $/*$ Every connection that reaches maximal age a_L is suppressed $*/$
  (c) **Removing neurons**
  $\forall n \in N, \forall (x, y) \in C$,
  if $(((x, y) \in C | x = n \vee y = n) = \phi)$ then $N = N - \{n\}$
  $/*$ Every mature neuron without connections is suppressed $*/$

7. **Change of status of embryos**
   $\forall n \in E$, if $(Age(n) > a_m)$,
   then $E = E - n$, $N = N \cup \{n\}$

EndWhile

Summary

Neural clustering algorithms show high performance in the general context of the analysis of standard test distribution, as well in the one of the analysis of homogeneous textual dataset. This is especially true for the recent adaptive versions of these algorithms, like the incremental growing neural gas algorithm (IGNG) and incremental growing neural gas based on label maximization algorithm (IGNG-F). In this paper we highlight that there is a drastic decrease of performance of all these algorithms, as well as the one of more classical algorithms, when a heterogeneous datasets, like polythematic textual datasets are considered as an input. We provide variations to incremental growing neural gas algorithm exploiting knowledge from clusters about the current labeling of clusters in an incremental way along with the cluster data distance measures which leads to significant increase in performance.

Impact de la taille de l'ensemble d'apprentissage : une étude empirique

Christophe Salperwyck[*,**], Vincent Lemaire[*]

[*]Orange Labs
2, Avenue Pierre Marzin 22300 Lannion
prenom.nom@orange-ftgroup.com
[**] LIFL (UMR CNRS 8022) - Université de Lille 3
Domaine Universitaire du Pont de Bois
59653 Villeneuve d'Ascq Cedex

Résumé. Les techniques d'apprentissage ont montré leurs capacités à traiter des volumétries importantes de données. La plupart des approches d'apprentissage statistique utilisent des bases d'apprentissage de taille finie et produisent des modèles statiques. Cependant dans certaines situations : apprentissage actif ou incrémental, la tâche d'apprentissage commence avec très peu de données. Il est nécessaire dans ces cas de s'intéresser à des algorithmes capables de générer des modèles avec peu de données. Les classifieurs de la littérature sont souvent classés à l'aide de critères tels que leurs performances en classification, leurs capacités à trier les données (ranking)... Mais cette taxonomie des classifieurs peut singulièrement évoluer si l'on s'intéresse à leurs capacités en présence d'une faible quantité d'exemples. A notre connaissance seulement quelques études ont été réalisées sur cette problématique. Cette étude cherche à étudier un plus large panel à la fois d'algorithmes d'apprentissage (9 types d'algorithmes différents) mais aussi de jeux de données (17 bases de l'UCI).

1 Introduction

Les techniques d'apprentissage ont montré leurs capacités à traiter des volumétries importantes de données et ce sur des problèmes réels (Guyon et al., 2009; Féraud et al., 2010). Néanmoins le plus gros effort a été réalisé pour des analyses de données homogènes et stationnaires. La plupart des approches d'apprentissage statistique (machine learning) utilisent des bases d'apprentissage de taille finie et produisent des modèles statiques. Cette étude se positionne dans le cadre de l'apprentissage supervisé.

Cependant dans certaines situations, la tâche d'apprentissage commence avec très peu de données. Il est nécessaire dans ces cas là de s'intéresser à des algorithmes capables de générer des modèles avec peu de données et si possible avec une faible variance. L'apprentissage actif et l'apprentissage incrémental sont les deux principales catégories d'apprentissage pour lesquelles un algorithme capable d'apprendre avec peu de données est nécessaire.

L'**apprentissage actif** (Settles, 2010) est utilisé dans les cas où de nombreuses données sont disponibles mais leurs étiquetages coûtent cher. Dans ce cas on cherche à sélectionner le

plus petit nombre de données permettant de réaliser le meilleur apprentissage. On se retrouve donc avec peu de données que l'on espère être très pertinentes. Un algorithme capable de réaliser de bonnes prédictions avec peu de données est donc nécessaire afin d'avoir une procédure d'étiquetage la moins coûteuse possible.

L'**apprentissage incrémental** (Michalski et al., 1986) commence par nature par apprendre avec peu de données car théoriquement il doit apprendre dès qu'on lui fournit les premiers exemples. Le modèle est raffiné au fur et à mesure de l'arrivée des nouveaux exemples. La qualité du modèle fourni au départ dépend donc là aussi de la capacité de l'algorithme à apprendre avec peu d'exemples. L'apprentissage incrémental existe depuis de nombreuses années mais est récemment revenu au goût du jour avec les flux de données. En effet de nombreuses applications génèrent des flux de données : réseaux de capteurs, millions de logs d'accès à des pages Web... Les données arrivent rapidement et ne sont visibles qu'une fois, il est donc nécessaire d'apprendre celles-ci au fur et à mesure de leurs arrivées. L'apprentissage incrémental apparaît dès lors comme une solution naturelle à ces problèmes de flux.

De leur coté **les classifieurs** de la littérature tels que les arbres de décisions, réseaux de neurones, support vector machine... sont souvent classés à l'aide de critères tels que leurs performances en classification, leurs capacités à trier les données (ranking)... Mais cette taxonomie des classifieurs peut singulièrement évoluer si l'on s'intéresse à leurs capacités en présence d'une faible quantité d'exemples. On peut citer par exemple les arbres d'Hoeffding (Domingos et Hulten, 2000) qui sont très largement utilisés comme méthode d'apprentissage incrémental sur les flux. La construction de l'arbre est incrémentale et les feuilles se transforment en nœud au fur et à mesure de l'arrivée des exemples. Or il apparait (Gama et al., 2003) qu'il est intéressant de mettre des classifieurs dans les feuilles de l'arbre avant que celles-ci ne soient transformées. Un classifieur pertinent avec peu de données permet d'avoir des modèles locaux de qualité dans les feuilles.

A notre connaissance seulement quelques études ont été réalisées sur cette problématique des performances des classifieurs de la littérature versus la taille de l'ensemble d'apprentissage. Forman et Cohen ont étudié trois algorithmes (Bayésien naïf, SVM et régression logistique) sur des jeux de petites tailles et déséquilibrés. Dans (Lim et al., 2000), les auteurs s'intéressent à la vitesse d'apprentissage en temps absolu contrairement à cette étude qui s'intéresse à la qualité d'apprentissage par rapport aux nombres d'exemples appris.

L'étude présentée dans cet article cherche à étudier un plus large panel à la fois d'algorithmes d'apprentissage (9 types d'algorithmes différents) mais aussi de jeux de données (17 bases de l'UCI). Dans un premier temps (section 2) les classifieurs qui seront utilisés dans cette étude ainsi que les valeurs de leurs paramètres testés seront présentés. Le protocole expérimental sera présenté section 3 : les jeux de données utilisés, le découpage des jeux de données en ensemble d'apprentissage et de test, les métriques d'évaluation de nos résultats. La section 4 présentera les résultats obtenus et les analysera par rapport à la typologie des différents classifieurs étudiés. Dans une dernière partie, nous conclurons et nous proposerons une suite possible à cette étude.

2 Vitesse d'apprentissage et Typologie des algorithmes testés

2.1 Vitesse d'apprentissage

Il est important de noter que cette étude ne se concentre pas sur les bornes ou temps de convergence d'un classifieur étant donné un ensemble d'apprentissage comportant n exemples. Ceci correspondrait à déterminer en quelque sorte le temps CPU qu'il faudrait à ce classifieur pour apprendre les n exemples. L'analyse de la vitesse de convergence ou la complexité de l'algorithme d'apprentissage n'est pas le but de cette étude.

On entend par vitesse d'apprentissage dans cette étude, la capacité d'un classifieur à obtenir une solution "intéressante" à un problème de classification à l'aide d'un minimum d'exemple présent en apprentissage. Un exemple de comparaison est donné dans la figure 1 : cette figure présente les résultats que pourraient obtenir deux classifieurs sur un même problème de classification. L'axe des abscisses indique le nombre d'exemples (n) utilisé pour entraîner le classifieur et l'axe des ordonnées l'AUC obtenue (AUC=$f(n)$).

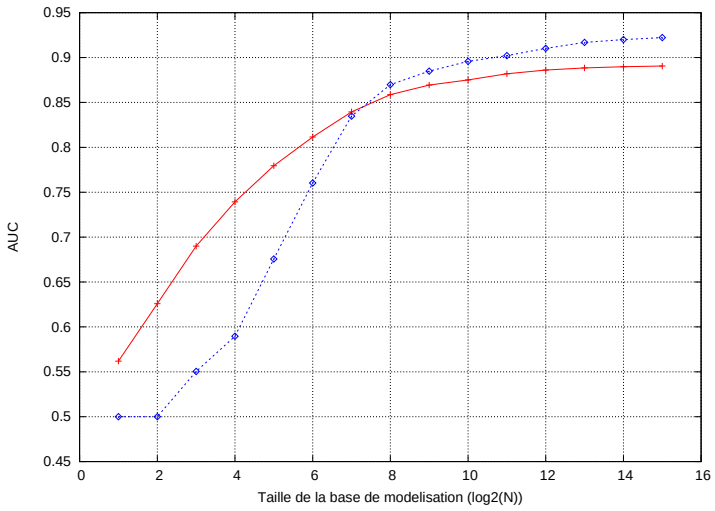


FIG. 1 – *Illustration de la vitesse d'apprentissage de deux classifieurs (AUC en test)*

Cette figure illustre deux comportements différents : l'algorithme "rouge" (+) atteint une meilleure performance dans un premier temps mais ensuite lorsque n devient grand l'algorithme "bleu" (○) est finalement plus optimal. Si l'on souhaite avoir toujours le meilleur modèle à disposition, il est évident que l'on doit prendre en compte les deux modèles et que le modèle rouge (+) devra être utilisé en premier puis le modèle bleu (○) par la suite.

Ceci illustre parfaitement le but de cette étude : dans le cas de l'apprentissage incrémental ou de l'apprentissage actif et pour des ensembles d'apprentissage de petites tailles existe-t-il des types de classifieurs plus adaptés que d'autres ?

2.2 Typologie des algorithmes testés

Il existe diverses familles de modèles de classification basées sur l'apprentissage de paramètres. Ces familles peuvent être :

- les classifieurs linéaires pour lesquels il s'agit d'apprentissage de surface séparatrice linéaire. Ce type de classifieur est basé sur une combinaison linéaire des variables explicatives du phénomène tel que $y = \sum_j w_j X_j$ où X_j désigne l'une des variables explicatives et w_j son poids.
- les classifieurs non linéaires tel que les perceptrons multicouches, les k plus proches voisins ou les arbres de décision.

Sachant que :

- ces familles peuvent se recouvrir partiellement ;
- le placement n'est pas toujours aisé (un arbre de décision peut être vu comme une surface de décision linéaire par morceaux) ;
- un classifieur linéaire peut traiter des problèmes de classification non linéaire sous peine d'une projection préalable (l'"astuce" du noyau pour les séparateurs à vaste marge (SVM))
- des sous découpes sont présentes. Par exemple les classifieurs linéaires peuvent être rangés en 2 grandes sous familles de méthodes pour estimer les paramètres du vecteur w d'un classifieur linéaire :
  - modèle génératif : il s'agit d'abord de modéliser les probabilités conditionnelles $P(X|C)$. L'estimation de $P(C|X)$ viendra ensuite sous la condition de connaître $P(C)$. On citera à titre d'exemple : l'analyse discriminante linéaire (LDA - (Fisher, 1936)) et le classifieur Bayésien naïf (Langley et al., 1992).
  - modèle discriminant : il s'agit d'abord de maximiser la qualité de la classification sur un jeu de test en estimant directement $P(C|X)$ ou un score apparenté à l'étiquette des données de modélisation. On citera à titre d'exemple : la régression linéaire, la régression logistique, le perceptron et les machines à vecteurs de support.

On peut encore en trouver d'autres sous appellations comme :

- les classifieurs probabilistes qui estiment les probabilités conditionnelles des classes ($P(C|X)$) étant donné un vecteur d'entrée constitué par les combinaisons des variables explicatives du phénomène. Ce n'est pas le cas de nombreux classifieurs comme les scores retournés par les séparateurs à vaste marge (qui sont pourtant souvent utilisés comme mesures de confiance pour la classification de nouveaux exemples).
- les classifieurs paramétriques qui présupposent que les variables explicatives suivent une loi de probabilité définie au préalable (a priori). On citera par exemple le classifieur naïf de Bayes dans le cas où on présuppose que les $P(X_j)$ suivent des lois gaussiennes (Gama et al., 2005).

L'étude proposée par cet article ne peut couvrir l'ensemble des familles des classifieurs. Elle s'est néanmoins attachée à paver au mieux cette espace. Le tableau 1 résume les types de classifieurs qui ont été testés. Ces différents classifieurs et leurs paramètres sont présentés dans la sous-section suivante.

	Classifieur linéaire	Classifieur non linéaire
Modèle génératif	Naïve Bayes, Bayésien naïf sélectif	Réseaux Bayésien
Modèle discriminant	Régression logistique	Arbre de décision, Forêt d'arbres de décision

TAB. 1 – *Typologie versus classifieurs testés.*

2.3 Classifieurs testés

Pour tester l'ensemble des classifieurs cités dans la section précédente (table 1) deux logiciels accessibles au 'public' [1] ont été utilisés : WEKA (Witten et Frank, 2005) (version 3.7.1) et Khiops (Boullé, 2004) (version 7.0). Pour chacun de ces logiciels se sont les valeurs par défaut qui ont été utilisées et dont on présente les résultats en section 4. A titre d'exemples les paramètres par défaut pour la régression logistique sont fixés à -1 (∞) pour le nombre d'itération maximale et à 10^8 pour le ridge.

- Pour le logiciel Weka :
  - Bayes
    - Non supervisé (John et Langley, 1995) : Bayésien naïf standard. L'estimateur numérique est basé sur l'analyse des données d'apprentissage.
    - Supervisé : même chose que précédemment sauf qu'une discrétisation supervisée (MDL) est utilisée pour convertir les attributs numériques en attributs nominaux.
    - BayesNet (Bouckaert, 2004) : réseau Bayésien utilisant divers algorithmes de recherche et de mesures de qualité. Utilisé avec les paramètres par défaut : "SimpleEstimator" et algorithme de recherche "K2 search".
  - Régression :
    - Régression logistique : régression logistique polynomial avec un estimateur ridge qui est basé sur (Cessie et Houwelingen, 1992). Paramètres par défaut sauf pour le nombre d'itération maximale : 100.
  - Arbres :
    - ADTree (Freund et Mason, 1999) : arbre de décision avec plusieurs sous arbres combinés avec du boosting.
    - J48 (Quinlan, 1993) : arbre de décision C4.5 proposé par Quinlan.
    - SimpleCart (Breiman et al., 1984) : arbre binaire basé sur le coefficient de Gini.
    - RandomForest (Breiman, 2001) : forêt d'arbres par sélection aléatoire d'un sous ensemble des attributs ; 10 arbres par défaut dans la forêt ; 40 arbres ont aussi été testés.
  - Vote :
    - VFI (Demiröz et Güvenir, 1997) : classification par vote sur les intervalles des attributs. Les intervalles sont construits autour des classes pour chaque attribut (en fait une sorte de discrétisation). Le compte des classes est enregistré pour chaque intervalle pour chaque attribut et la classification se fait par vote. Le paramètre par défaut utilise l'option "weight feature intervals by confidence". Nous avons aussi essayé sans.

1. Toutes les expériences de cette étude sont reproductibles.

– Pour le logiciel Khiops [2] : outils développé par Orange Labs. Il implémente, pour la classification supervisée, un Bayésien naïf et Bayésien naïf sélectif.
Le Bayésien naïf et le Bayésien naïf sélectif (Boullé, 2006b) ont été testé avec différents prétraitements sur les attributs numériques et nominaux. Les deux prétraitements testés sur les attributs nominaux sont :
– Basic Grouping : un groupe par valeur observée
– Discrétisation MODL (Boullé, 2006a).
Sur les attributs numériques trois prétraitements différents ont été testés :
– Equal Frequency
– Equal Width
– Méthode de groupement MODL (Boullé, 2005).
Les deux méthodes non supervisées Equal Frequency et Equal Width sont utilisées avec un nombre fixé à 10 intervalles. Si le nombre de valeurs observées est en dessous de 10, le nombre d'intervalles est réduit à ce nombre de valeurs observées.

3 Protocole expérimental

A travers la mise en place du protocole expérimental, on cherche à étudier le comportement des algorithmes en fonction de la taille de l'ensemble d'apprentissage. Le premier point consiste à utiliser des jeux de données à la fois variés et reconnus par la communauté de la fouille de données. Le second concerne le découpage des jeux de données en un ensemble d'apprentissage et de test pertinent par rapport à la problématique. Le troisième s'intéresse à la méthode d'évaluation mettant en avant les algorithmes qui apprennent bien avec peu de données.

3.1 Jeux de données

Afin que les résultats de cette étude puissent être généralisés des jeux de données variés du domaine de la classification ont été utilisés. Dans ce but, un large panel de jeux de données a été sélectionné à partir du dépôt de l'UCI (Blake et Merz, 1998). Des jeux avec seulement des variables catégorielles, ou seulement des variables continues, ou un mix des deux ont été utilisés. La taille de ceux-ci va d'une centaine à plusieurs dizaine de milliers d'exemples. La table 2 présente les caractéristiques en termes de nombre d'exemples, nombre d'attributs numériques et nominaux, précision du classifieur prédisant la classe majoritaire.

Cette étude ne s'intéresse qu'à des problèmes de classification binaire. Pour des problèmes à plus de deux classes le protocole expérimental devrait être différent [3].

3.2 Découpage des jeux de données en ensemble d'apprentissage et de test

Afin de générer nos jeux de données de petites tailles tout en gardant un protocole semblable à ceux utilisées dans les publications du domaine, nous avons choisi une 10-validation

2. www.khiops.com

3. En effet, la question se pose si l'on doit comparer, par exemple, l'apprentissage de peu d'exemples sur un problème binaire versus un problème à 20 classes.

	Jeu de données	#Var	#Var Cont.	#Var Cat.	# Exemples (n)	Précision maj.
1	Adult	15	7	8	48842	0.7607
2	Australian	14	6	8	690	0.5550
3	Breast	10	10	0	699	0.6552
4	Bupa	6	6	0	345	0.5797
5	Crx	15	6	9	690	0.5550
6	German	24	24	0	1000	0.7
7	Heart	13	10	3	270	0.5555
8	Hepatitis	19	6	13	155	0.7935
9	Horsecolic	27	7	20	368	0.6304
10	Hypothyroid	25	7	18	3163	0.9522
11	Ionosphere	34	34	0	351	0.6410
12	Mushroom	22	0	22	8416	0.5332
13	Pima	8	8	0	768	0.6510
14	SickEuthyroid	25	7	18	3163	0.9073
15	Sonar	60	60	0	208	0.5336
16	Spam	57	57	0	4307	0.6473
17	Tictactoe	9	0	9	958	0.6534

TAB. 2 – *Caractéristiques des jeux de données.*

croisée. Sur l'ensemble d'apprentissage (90%) de cette validation croisée, les exemples ont été tirés aléatoirement parmi les tailles suivantes :

$$S = \{S_1, S_2, ..., S_{max}\} = \left\{2, 2^2, 2^3, 2^4, \ldots, 2^{(\lfloor log_2(0.9n-1)\rfloor)}\right\}$$

où n est la taille maximale du jeu de données [4]. Le tirage a été réalisé 10 fois pour chaque jeu de données afin d'obtenir une moyenne et une variance sur le résultat. La performance des algorithmes est évaluée sur les 10% restant (des n exemples, voir table 2) de la validation croisée. La figure 3.2 présente ce protocole.

3.3 Le critère d'évaluation : ALC

Les expérimentations menées dans cet article donnent une courbe d'AUC (Fawcett, 2004) sur l'ensemble de test versus le nombre d'exemples utilisés pour réaliser l'apprentissage. Chaque courbe est constituée de $|S|$ points pour tous les algorithmes d'apprentissage utilisés.

Les performances en prédiction ont été évaluées avec l'ALC (Area under the Learning Curve - (Guyon et al., 2010)). L'AUC (Area Under the ROC Curve) est calculée pour tous les points $(2, 4, 8...)$ définis dans le jeu de données. Le score obtenu correspond à l'ALC normalisé calculé de la manière suivante :

$$score = \frac{(ALC - Arand)}{(Amax - Arand)}$$

où $Amax$ est la meilleure aire atteignable (soit 1) et $Arand$ est l'aire d'une solution basée sur des prédictions aléatoires (soit 0.5).

4. .9n correspond au fait que l'on réalise une 10-validation croisée

Impact de la taille de l'ensemble d'apprentissage

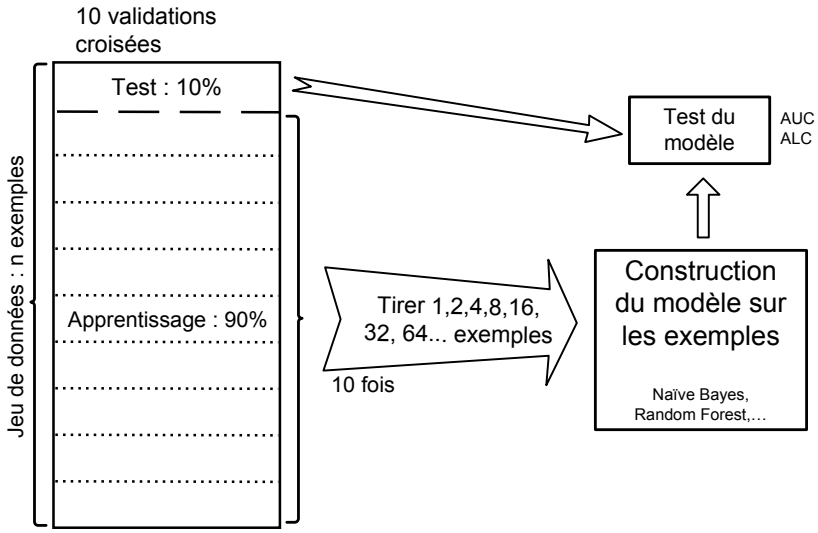


FIG. 2 – *Construction des jeux de données*

Ce critère possède de bonnes propriétés par rapport à l'objectif de cette étude. En effet, on cherche à évaluer la capacité d'un algorithme à apprendre avec une faible quantité de données. Ce critère met particulièrement en avant les plus petits jeux de données. L'abscisse de notre courbe d'ALC est logarithmique ce qui signifie que l'AUC obtenue pour 2 exemples contribue autant à la valeur de l'ALC que l'AUC obtenue pour 1024 exemples.

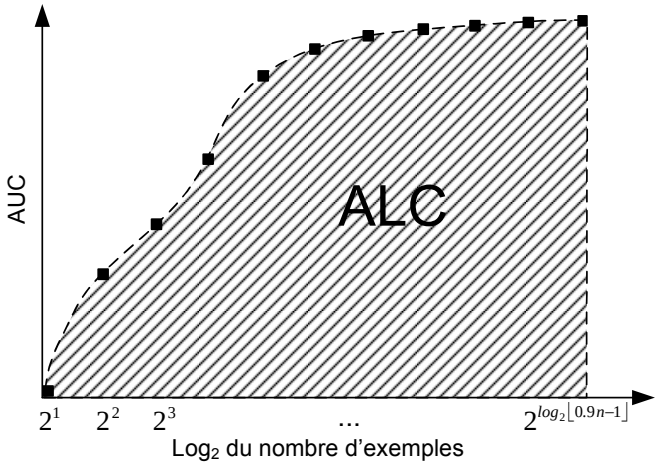


FIG. 3 – *Calcul de l'ALC : aire sous la courbe d'AUC*

4 Résultats

Cette section présente les résultats de l'étude. Les notations utilisées sont tout d'abord expliquées, ensuite les différentes tables et courbes de résultats sont présentées pour finalement les analyser et en proposer une interprétation.

4.1 Notations

Afin de pouvoir obtenir des tableaux de résultats complets et concis, nous avons dû utiliser des abréviations quant aux noms des algorithmes utilisés. L'environnement d'exécution constitue la première abréviation. Les algorithmes utilisés provenant de Weka sont préfixés par "W" et ceux provenant de Khiops n'ont pas de préfixes. Les algorithmes testés constituent la seconde abréviation :

- Weka
  - W-ADT : ADTree
  - W-BN : BayesNet - Réseau Bayésien
  - W-Log100 : Régression logistique
  - W-NB-NS : Bayésien naïf non supervisé
  - W-NB-S : Bayésien naïf supervisé
  - W-RF10/40 : Random Forest avec 10/40 arbres
  - W-VFI / W-VFIn : VFI avec/sans l'option "weight feature intervals by confidence"
  - W-SCart : SimpleCart
- Khiops Le format est ALGO-VAR_CONT-VAR_CAT
  - ALGO - NB : Bayésien naïf ou SNB : Bayésien naïf sélectif
  - VAR_CONT pour les variables continues - EF : EqualFrequency / EW : EqualWidth / M : MODL
  - VAR_CAT pour les variables catégorielles - BG : Basic Grouping / M : MODL

4.2 Tableaux et courbes présentés

Tableaux

Les performances en ALC obtenues par les différents algorithmes sur les différents jeux de données sont présentés dans la table 3. La dernière ligne donne la moyenne de l'ALC sur tous les jeux de tests par algorithme. Afin d'avoir une vue plus synthétique, la table 5 donne le rang moyen de chaque algorithme ainsi que son ALC moyenne et son AUC finale (moyenne des AUC du plus grand ensemble d'apprentissage : $2^{(\lfloor log_2(0.9n-1) \rfloor)}$).

Etant donné que cette étude se focalise particulièrement sur l'apprentissage des jeux de données de petites tailles, il est intéressant regarder le début des courbes d'AUC. Dans cette optique, des tableaux identiques aux tableaux 3 sont proposés dans les tableaux 4 mais ici l'ALC est calculée entre 2 et 2^6 exemples en apprentissage. Ce tableau permet d'observer plus particulièrement les algorithmes sur le début de la phase d'apprentissage. De façon identique, on retrouve pour les rangs par algorithme la table 6 qui se limite aussi à 2^6 exemples.

Courbes

Nous avons choisi quatre courbes mettant en avant différents types d'algorithmes. L'idée est de comparer le comportement de cet algorithme par rapport aux autres algorithmes sur un jeu

	W-ADT	W-BN	W-Log100	NB-EF-BG	NB-EF-M	NB-EW-BG	W-NB-NS	W-NB-S	W-RF10	W-RF40	W-VFI	W-VFI-N
Adult	58.05	**64.28**	54.19	62.31	58.98	62.64	59.88	63.70	54.73	58.44	58.02	63.80
Australian	59.40	71.55	55.17	65.75	64.07	64.11	56.43	**72.15**	59.88	63.23	59.03	69.06
Breast	74.97	89.12	87.27	**92.31**	**92.31**	92.18	88.26	89.04	88.25	89.89	88.40	92.30
Bupa	20.17	2.02	**24.06**	14.81	14.81	13.20	12.52	1.98	21.74	23.99	9.05	9.32
Crx	60.16	70.12	54.60	63.91	62.79	62.32	56.30	**70.63**	59.02	62.47	57.33	66.57
German	25.98	13.10	26.69	27.61	27.61	26.13	30.53	13.12	27.29	**31.69**	24.84	26.89
Heart	47.19	62.19	51.25	61.31	58.26	60.21	54.56	62.84	53.64	57.15	49.75	**64.22**
Hepatitis	39.92	60.38	37.27	49.64	46.22	47.78	37.92	**63.16**	46.93	50.36	46.71	58.88
Horsecolic	52.73	59.81	48.31	51.87	44.01	52.06	40.61	61.58	57.34	**62.27**	54.20	58.31
Hypothyroid	59.21	50.78	59.27	65.87	**66.75**	64.06	52.81	48.77	63.05	66.13	52.94	60.41
Ionosphere	57.42	59.18	43.17	65.91	65.91	64.68	56.58	59.06	65.86	**70.29**	52.06	61.54
Mushroom	82.20	**92.23**	87.20	88.99	86.26	88.99	92.07	92.07	88.76	89.97	89.25	91.01
Pima	37.70	30.76	41.99	41.50	41.50	40.16	38.35	30.84	40.68	**44.43**	26.77	30.06
SickEuthyroid	60.46	52.57	57.49	59.44	59.29	59.50	50.28	49.25	61.87	**66.85**	45.41	47.93
Sonar	32.64	36.54	36.04	38.10	38.10	37.32	31.72	36.51	41.80	**48.32**	23.12	34.37
Spam	66.53	72.86	68.86	79.93	79.93	77.29	71.07	72.76	75.92	**80.08**	67.72	78.29
Tictactoe	32.40	28.40	**53.24**	26.91	20.52	26.91	28.73	28.73	37.77	41.83	26.00	27.61
Moyenne ALC	51.01	53.88	52.12	56.25	54.55	55.27	50.51	53.89	55.56	**59.26**	48.86	55.33

	W-J48	NB-M-BG	NB-M-M	NB-EW-M	SNB-EF-BG	SNB-EF-M	SNB-EW-BG	SNB-EW-M	SNB-M-BG	SNB-M-M	W-SCart
Adult	41.68	64.06	56.63	59.25	55.25	55.13	55.73	55.52	57.92	56.02	40.97
Australian	51.60	65.60	63.22	62.32	53.20	54.13	52.04	53.04	54.85	57.18	42.70
Breast	66.84	76.24	76.24	92.18	79.29	79.29	79.40	79.40	76.30	76.30	57.85
Bupa	10.54	1.91	1.91	13.20	13.58	13.58	12.00	12.00	1.91	1.91	8.53
Crx	51.68	63.73	62.90	61.01	52.92	54.38	51.99	53.15	54.50	57.31	44.03
German	14.39	14.14	14.14	26.13	22.27	22.27	22.11	22.11	14.15	14.15	10.06
Heart	29.21	57.34	45.95	56.90	40.91	40.75	38.92	38.76	42.81	41.28	22.88
Hepatitis	21.05	46.83	40.49	44.66	30.99	32.43	31.32	32.50	30.52	31.34	10.62
Horsecolic	40.18	55.08	49.01	44.06	45.12	40.67	45.36	40.90	47.42	43.50	2.04
Hypothyroid	43.79	54.46	52.09	65.06	56.56	56.62	60.77	61.03	53.04	52.56	39.54
Ionosphere	42.89	53.04	53.04	64.68	55.15	55.15	55.48	55.48	53.42	53.42	36.27
Mushroom	77.71	88.99	86.26	86.26	82.56	81.77	82.56	81.77	82.56	81.77	74.62
Pima	25.34	29.71	29.71	40.16	37.65	37.65	37.56	37.56	29.89	29.89	20.08
SickEuthyroid	47.80	52.14	45.32	59.64	56.16	55.95	55.57	55.68	46.20	45.49	45.21
Sonar	19.40	31.97	31.97	37.32	25.26	25.26	25.21	25.21	27.12	27.12	16.20
Spam	52.67	66.63	66.63	77.29	65.93	65.93	60.53	60.53	66.33	66.33	46.11
Tictactoe	21.79	26.91	20.52	20.52	25.50	20.29	25.50	20.29	25.50	20.29	25.47
Moyenne ALC	38.74	49.93	46.83	53.57	46.96	46.54	46.59	46.17	44.97	44.46	31.95

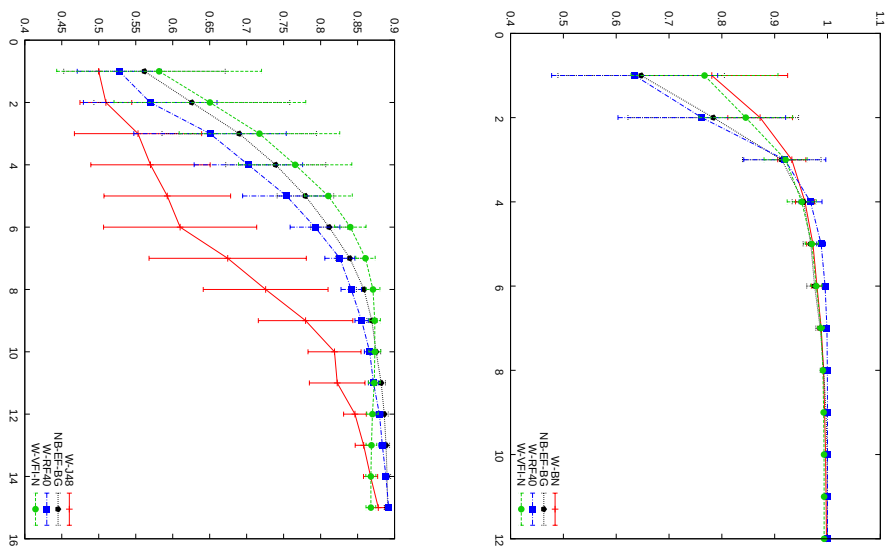
TAB. 3 – *ALC par algorithme et jeu de données*

	W-ADT	W-BN	W-Log100	NB-EF-BG	NB-EF-M	NB-EW-BG	W-NB-NS	W-NB-S	W-RF10	W-RF40	W-VFI	W-VFI-N
Adult	28.64	39.37	30.01	40.85	30.41	40.73	29.73	38.23	30.58	33.46	37.73	**46.18**
Australian	45.86	65.32	45.53	57.02	53.94	55.99	44.12	**66.17**	48.43	52.13	51.11	62.37
Breast	61.52	83.70	82.08	88.89	88.89	**88.92**	82.76	83.63	82.88	85.20	84.21	88.86
Bupa	14.19	2.02	**17.75**	10.48	10.48	9.76	8.26	1.96	14.90	16.28	8.30	8.66
Crx	46.73	62.86	44.37	54.27	52.00	53.40	43.71	**63.59**	46.86	50.82	48.45	58.91
German	17.49	4.54	15.51	17.20	17.20	16.92	18.51	4.59	18.64	**21.74**	15.29	18.44
Heart	42.48	59.44	47.95	58.07	54.24	57.68	49.47	60.14	50.22	53.59	46.64	**61.78**
Hepatitis	36.16	57.85	35.04	46.30	41.94	45.26	32.16	**60.91**	43.53	46.90	43.34	55.85
Horsecolic	44.41	54.46	41.07	44.70	33.88	45.11	29.08	**57.08**	49.18	54.92	49.17	52.52
Hypothyroid	26.39	9.91	33.30	38.32	**38.81**	37.85	16.01	7.44	35.21	37.81	35.21	34.46
Ionosphere	45.63	47.00	35.35	58.20	58.20	56.82	44.75	46.89	55.36	**60.61**	37.83	50.35
Mushroom	61.75	**84.61**	73.47	77.27	71.14	77.27	84.41	84.41	75.56	78.09	77.66	82.42
Pima	28.26	19.00	30.58	31.50	31.50	31.17	26.76	19.12	32.82	**35.95**	26.51	31.45
SickEuthyroid	31.74	17.08	32.90	37.90	36.64	39.65	20.62	12.17	35.28	**42.33**	33.59	34.56
Sonar	26.88	31.71	33.79	33.12	33.12	33.15	26.72	31.65	36.10	**42.42**	22.51	33.67
Spam	44.03	54.31	57.14	68.25	68.25	67.19	55.39	54.33	60.64	66.77	55.83	**68.26**
Tictactoe	17.09	19.82	**27.38**	17.54	10.61	17.54	20.27	20.27	19.21	21.63	16.04	18.71
Moyenne ALC	36.43	41.94	40.19	45.88	43.01	45.55	37.22	41.92	43.26	47.10	40.55	**47.50**

	W-J48	NB-M-BG	NB-M-M	NB-EW-M	SNB-EF-BG	SNB-EF-M	SNB-EW-BG	SNB-EW-M	SNB-M-BG	SNB-M-M	W-SCart
Adult	11.21	30.34	40.53	19.72	23.43	20.65	23.48	21.34	24.63	17.83	4.35
Australian	38.85	52.99	57.47	52.79	36.58	37.57	36.15	37.43	39.30	42.05	24.83
Breast	54.86	88.92	63.19	63.19	67.84	67.84	68.03	68.03	63.14	63.14	40.97
Bupa	6.78	9.76	1.31	1.31	8.87	8.87	8.14	8.14	1.30	1.30	4.23
Crx	39.50	51.15	54.52	52.17	36.12	37.66	36.06	37.56	38.72	42.01	26.97
German	8.43	16.92	3.82	3.82	9.83	9.83	10.95	10.95	3.81	3.81	2.88
Heart	25.93	53.66	53.81	39.91	34.19	33.94	32.90	32.73	36.44	34.35	17.85
Hepatitis	18.19	41.07	43.88	35.83	26.66	27.93	26.98	28.10	25.99	26.20	8.54
Horsecolic	31.30	34.56	48.65	39.77	34.61	27.70	35.41	28.48	37.81	30.81	1.67
Hypothyroid	11.47	38.19	18.51	14.28	19.32	19.50	29.47	29.63	14.87	14.49	7.11
Ionosphere	31.15	56.82	39.30	39.30	41.67	41.67	42.30	42.30	38.53	38.53	21.60
Mushroom	52.99	71.14	77.27	71.14	62.30	60.64	62.30	60.64	62.30	60.64	46.68
Pima	18.38	31.17	15.81	15.81	24.77	24.77	26.04	26.04	15.86	15.86	9.92
SickEuthyroid	16.31	38.65	22.80	9.66	25.84	25.53	27.55	27.60	9.53	8.90	10.63
Sonar	15.39	33.15	26.04	26.04	18.60	18.60	19.24	19.24	21.24	21.24	11.52
Spam	32.00	67.19	41.92	41.92	38.96	38.96	35.31	35.31	40.13	40.13	21.05
Tictactoe	8.92	10.61	17.54	10.61	13.53	9.83	13.53	9.83	13.53	9.83	5.50
Moyenne ALC	24.80	42.72	36.85	31.60	30.77	30.09	31.40	30.79	28.65	27.71	15.66

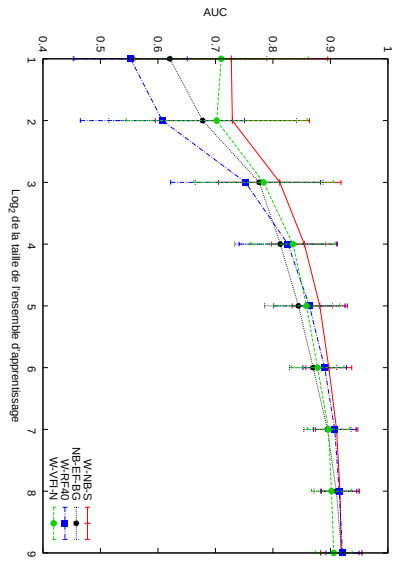
TAB. 4 – *ALC par algorithme et jeu de données mais avec l'aire entre* 2 *et* $2^6 = 64$ *exemples en apprentissage*

Impact de la taille de l'ensemble d'apprentissage

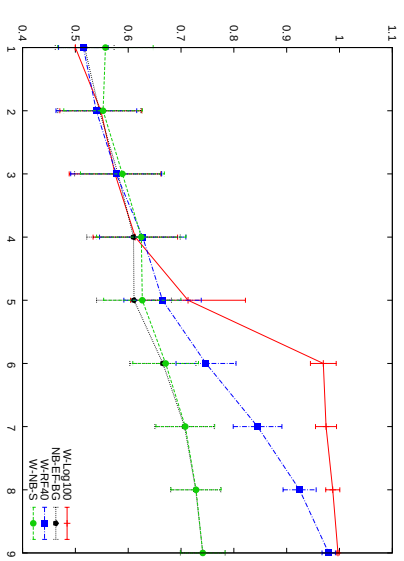


(a) Jeu de données Adult

(b) Jeu de données Mushroom



(c) Jeu de données Crx



(d) Jeu de données TicTacToe

FIG. 4 – *AUC en fonction du log_2 du nombre d'exemples appris*

Algorithme	Rang moyen	ALC moyen	AUC finale moyenne
W-RF40	**3.65**	**59.26**	**91.30**
NB-EF-BG	4.53	56.25	88.04
NB-EW-BG	6.47	55.27	86.82
W-VFI-N	6.82	55.33	83.13
NB-EF-M	7.06	54.55	88.04
W-RF10	7.29	55.56	89.70
W-BN	8.18	53.88	87.36
W-NB-S	8.53	53.89	87.35
NB-EW-M	9.00	53.57	86.80
W-ADT	10.88	51.01	88.66
W-Log100	10.88	52.12	88.55
W-NB-NS	11.00	50.51	87.09
NB-M-BG	11.94	49.93	86.81
W-VFI	13.65	48.86	82.04
SNB-EF-BG	14.88	46.96	88.15
SNB-EW-BG	15.47	46.59	87.06
NB-M-M	15.94	46.83	86.84
SNB-EF-M	16.06	46.54	88.20
SNB-EW-M	16.53	46.17	86.99
SNB-M-BG	16.53	44.97	87.07
SNB-M-M	17.82	44.46	87.19
W-J48	20.65	38.74	82.84
W-SCart	22.24	31.95	81.70

TAB. 5 – *Rang moyen. ALC moyen et AUC finale moyenne*

de données qui ne leur est pas favorable. L'un des meilleurs exemples est la figure 4(d) qui montre un jeu de données qui fonctionne très bien avec une régression logistique mais qui est très mauvais avec un Bayésien naïf.

4.3 Analyse

Analyse globale

Cette étude a été réalisée sans optimisation des paramètres des algorithmes mais principalement en utilisant les paramètres par défaut de ceux-ci. C4.5 (J48) et CART (SimpleCart) sont des références en apprentissage mais cette étude montre qu'ils ne sont pas adaptés pour bien travailler avec des petits jeux de données. En effet ils se retrouvent derniers que ce soit au niveau du rang moyen ou de l'ALC moyenne.

Au contraire, le Bayésien naïf est connu pour bien fonctionner sur de petits jeux de données (Domingos et Pazzani, 1997). Les figures 4(a) et 4(b) confirment ce résultat. Très rapidement les classifieurs générés sont capables de donner de bons résultats, on observe assez souvent qu'il faut entre 2^5 et 2^8 exemples en apprentissage pour être très proche de l'AUC maximale.

Les algorithmes basés sur les arbres ne fonctionnent pas bien directement (C4.5, Cart) mais une fois combinés avec des techniques de bagging/boosting ils sont parmi les meilleurs de cette étude : RandomForest. La figure 4 montre le bon comportement général des forêts d'arbres sur 4 jeux de données. Il faut noter que les sous-figures 4(c) et 4(d) mettent en avant des classifieurs (W-NB-S et W-Log100) qui sont particulièrement bien adaptés à ces deux jeux de données (Crx et TicTacToe). Même dans ces cas, RandomForest affiche de bons résultats ;

Impact de la taille de l'ensemble d'apprentissage

Algorithme	Rang moyen	ALC moyen	AUC finale moyenne
W-RF40	**4.35**	47.10	**86.21**
W-VFI-N	4.59	**47.50**	82.43
NB-EF-BG	4.65	45.88	83.45
NB-EW-BG	5.41	45.55	82.34
W-RF10	6.88	43.26	84.09
NB-EF-M	7.47	43.01	83.29
NB-EW-M	8.29	42.72	82.12
W-BN	8.88	41.94	81.58
W-NB-S	8.88	41.92	81.28
W-Log100	10.00	40.19	79.99
W-VFI	10.24	40.55	77.91
W-NB-NS	11.53	37.22	82.60
W-ADT	11.76	36.43	83.02
NB-M-BG	11.88	36.85	80.92
SNB-EW-BG	15.65	31.40	81.93
SNB-EF-BG	15.88	30.77	82.44
NB-M-M	16.18	31.60	80.29
SNB-EW-M	16.76	30.79	81.76
SNB-EF-M	17.12	30.09	82.20
SNB-M-BG	17.76	28.65	80.18
SNB-M-M	19.41	27.71	80.12
W-J48	19.94	24.80	74.14

TAB. 6 – *Rang moyen, ALC moyen et AUC finale moyenne mais avec l'aire entre* 2 *et* $2^6 = 64$ *exemples en apprentissage*

de manière générale il se situe mieux que les autres classifieurs sur la grande majorité des jeux de données.

Il est étonnant de voir apparaitre dans le haut du tableau de cette étude un algorithme assez peu connu : VFI. Cet algorithme se comporte particulièrement bien avec le paramètre qui ne pondère pas les intervalles des variables en fonction de la confiance ("weight feature intervals by confidence" : W-VFI-N). Si on s'arrête aux 64 premiers exemples en apprentissage (Tableau 6), il se retrouve premier en ALC et deuxième en termes de rang avec un très faible écart par rapport au premier.

Analyse pour les modèles discriminants

Les algorithmes discriminants sont représentés dans cette étude par les forêts d'arbre (RF et RF40) et la régression logistique (Log100). L'algorithme RandomForest paramétré avec une taille de forêt de 40 arbres est le meilleur de cette étude sur tous les points de vue, premier pour le rang, l'ALC moyenne et l'AUC finale moyenne. La possibilité de cet algorithme à essayer de nombreux arbres avec seulement une partie des variables explicatives lui permet d'être à la fois bon en début d'apprentissage ainsi qu'en fin.

Bien qu'elle se positionne après RandomForest, le Bayésien naïf et VFI, la régression logistique n'est pas si éloignée au niveau de son score d'ALC. Dans certains cas particuliers elle fonctionne même bien mieux que les autres méthodes comme le montre la figure 4(d) sur le jeu de données TicTacToe.

Analyse pour les modèles génératifs

Parmi les modèles génératifs testés dans cette étude, le Bayésien naïf est le plus évalué. On constate que la discrétisation des variables continues et le groupement des variables catégorielles influencent grandement les résultats. Le meilleur choix est constitué par la paire ("EqualFrequency", "Basic Grouping") suivi de ("EqualWidth", "Basic Grouping"). La discrétisation régularisée (Boullé, 2006b) et la méthode de groupement MODL (Boullé, 2005) sont trop robustes pour être capables de générer un modèle avec peu d'exemples. Dans le cadre de cette étude, les méthodes "les plus simples" ont l'avantage de s'exprimer plus rapidement et donc de donner de meilleurs résultats.

Les méthodes régularisées (approche MODL et Bayésien naïf sélectif) sont connues pour avoir un faible biais-variance (Cucker et Smale, 2008). Par conséquent afin de maintenir ce faible biais-variance, leur variance et donc aussi leur AUC va être plus faible en début d'apprentissage. Cette plus grande robustesse les amène à ne pas prédire un résultat très variable et donc à rester "silencieuses". Ces méthodes se retrouvent donc avec des scores faibles dans le cadre de cette étude et se positionnent en dernières positions bien qu'elles soient quand même meilleures que C4.5 et Cart.

Les modèles génératifs générant un classifieur non linéaire (voir table 1) sont représentés dans cette étude par les réseaux Bayésiens. Ils se comportent assez bien mais se situent juste après les RandomForest, VFI et le Bayésien naïf. Sur certains jeux de données : Adult et Mushrooms (figure 4(b)), ils sont les meilleurs classifieurs en termes d'ALC.

Comparaisons avec des analyses existantes

Les résultats de la littérature [5] sont confirmés dans cette étude : les méthodes par vote et ensemble de classifieurs ont de très bons résultats Bauer et Kohavi (1999), les modèles génératifs sont meilleurs que les modèles discriminants quand le nombre d'exemples est faible (Bouchard et Triggs, 2004), les méthodes régularisées sont robustes (Cucker et Smale, 2008).

5 Conclusion et travaux futurs

5.1 Vers un nouveau critère ?

Nous avons vu précédemment les résultats présentés dans deux tableaux : un contenant les résultats de l'aire sous la courbe d'AUC (ALC) s'arrêtant à 2^6 exemples (tableau 6) et un autre donnant l'AUC finale (tableau 5). Le premier tableau indique le comportement de l'algorithme au début de l'apprentissage et le second sa performance en fin d'apprentissage. Les deux axes de la figure 5 reprennent ces deux critères d'évaluation. En moyenne, seul "RandomForest" et "NB-EF-BG" arrivent à être bien positionnés au niveau de ces deux critères. La régression logistique et le boosting d'arbres : ADTree ont de bonnes performances en AUC finale mais leurs ALC sont relativement faibles. Au contraire les algorithmes de type "VFI" ont une bonne ALC mais une mauvaise AUC finale.

5. La plupart des études de la littérature ne sont pas réalisée sur de si petits jeux de données, par conséquence pour les comparer il faut se placer après avoir appris sur au moins $2^7(128)$ à $2^{10}(1024)$ exemples.

Impact de la taille de l'ensemble d'apprentissage

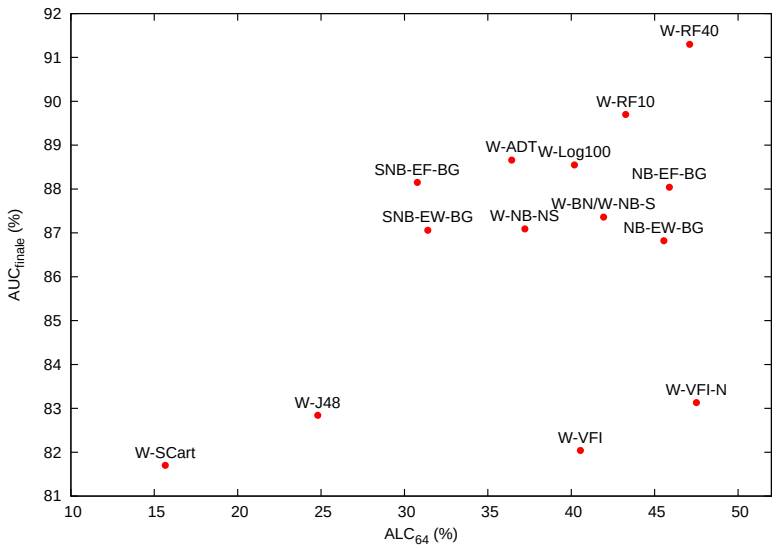


FIG. 5 – *Positionnement des différentes méthodes en termes d'*ALC_{2^6} *et d'*AUC_{finale}

Cette analyse pose la question de faire évoluer le critère d'ALC vers un nouveau critère qui prendrait en compte ces deux aspects. En effet, il serait intéressant d'avoir un critère synthétique nous permettant d'évaluer la qualité d'un algorithme selon ces deux axes. Ce critère pourrait par exemple s'écrire sous la forme suivante :

$$Critère = \frac{ALC_{2^6} + AUC_{finale}}{2}$$

5.2 Combinaison d'algorithmes

Dans cette étude, peu de cas se présentent où un algorithme particulier est très bon sur le début de l'apprentissage et un autre algorithme est très bon sur la fin de l'apprentissage. Cependant si nous étions dans une telle situation, c'est à dire comme celle de la figure 6, il serait intéressant d'utiliser deux algorithmes distincts pour réaliser l'apprentissage : W-VFI-N pour les 2^8 premiers exemples puis une Bayésien naïf sélectif pour la fin de l'apprentissage.

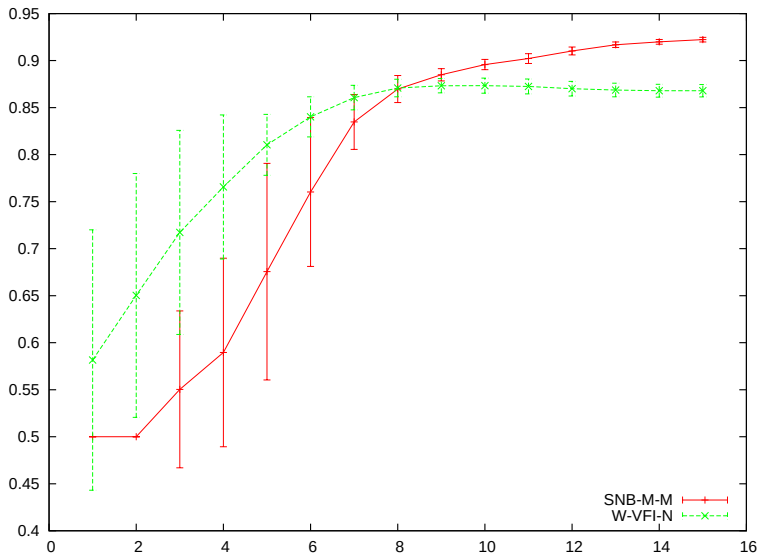


FIG. 6 – *Un jeu de données (Adult) où une combinaison d'algorithmes est souhaitable*

5.3 Recommandations

Il ressort de cette étude que plusieurs algorithmes peuvent être utilisés pour construire un modèle sur un ensemble d'apprentissage de faible taille : le Bayésien naïf qui était attendu pour ces qualités sur de faible quantité de données et les forêts d'arbres. Ces deux algorithmes nécessitent d'être paramétrés pour obtenir leurs meilleurs performances : les couples ("Equal-Frequency", "Basic Grouping") et ("EqualWidth", "Basic Grouping") fonctionnent le mieux pour le Bayésien naïf et une forêt de 40 arbres pour "RandomForest". Si l'on s'intéresse plus particulièrement au tout début de l'apprentissage, une méthode par vote sur des intervalles (VFI) se positionne au même niveau que les meilleures méthodes de cette étude.

Références

Bauer, E. et R. Kohavi (1999). An empirical comparison of voting classification algorithms : Bagging, boosting, and variants. *Machine learning 36*(1), 105–139.

Blake, C. L. et C. J. Merz (1998). UCI Repository of machine learning databases. `http://archive.ics.uci.edu/ml/` visité pour la dernière fois : 15/09/2010.

Bouchard, G. et B. Triggs (2004). The tradeoff between generative and discriminative classifiers. In *IASC International Symposium on Computational Statistics (COMPSTAT)*, pp.

721–728.

Bouckaert, R. (2004). Bayesian Network Classifiers in Weka. `http://weka.sourceforge.net/manuals/weka.bn.pdf`.

Boullé, M. (2004). Khiops : A Statistical Discretization Method of Continuous Attributes. *Machine Learning 55*(1), 53–69.

Boullé, M. (2005). A grouping method for categorical attributes having very large number of values. *Machine Learning and Data Mining in Pattern Recognition*, 228–242.

Boullé, M. (2006a). MODL : A Bayes optimal discretization method for continuous attributes. *Machine Learning 65*(1), 131–165.

Boullé, M. (2006b). Regularization and Averaging of the Selective Naive Bayes classifier. *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, 1680–1688.

Breiman, L. (2001). Random forests. *Machine learning 25*(2), 5–32.

Breiman, L., J. Friedman, R. Olshen, et C. Stone (1984). *Classification and regression trees*. Chapman and Hall/CRC.

Cessie, S. L. et J. V. Houwelingen (1992). Ridge estimators in logistic regression. *Applied Statistics*.

Cucker, F. et S. Smale (2008). Best Choices for Regularization Parameters in Learning Theory : On the Bias-Variance Problem. *Foundations of Computational Mathematics 2*(4), 413–428.

Demiröz, G. et H. Güvenir (1997). Classification by voting feature intervals. *Machine Learning : ECML-97*, 85–92.

Domingos, P. et G. Hulten (2000). Mining high-speed data streams. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 71–80. ACM New York, NY, USA.

Domingos, P. et M. Pazzani (1997). On the optimality of the simple Bayesian classifier under zero-one loss. *Machine learning 130*, 103–130.

Fawcett, T. (2004). ROC graphs : Notes and practical considerations for researchers. *Machine Learning 31*, 1–38.

Féraud, R., M. Boullé, F. Clérot, F. Fessant, et V. Lemaire (2010). The orange customer analysis platform. In *Industrial Conference on Data Mining (ICDM)*, pp. 584–594.

Fisher, R. (1936). The use of multiple measurements in taxonomic problems. *Annals of eugenics 7*, 179–188.

Freund, Y. et L. Mason (1999). The alternating decision tree learning algorithm. In *Machine learning*, pp. 124–133.

Gama, J., P. Medas, et P. Rodrigues (2005). Learning Decision Trees from Dynamic Data Streams. In *Proceedings of the ACM Symposium on Applied ComputingLearning Decision Trees from Dynamic Data Streams*.

Gama, J., R. Rocha, et P. Medas (2003). Accurate decision trees for mining high-speed data streams. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 523–528. ACM New York, NY, USA.

Guyon, I., G. Cawley, G. Dror, et V. Lemaire (2010). Results of the Active Learning Challenge. In *JMLR W&CP, Workshop on Active Learning and Experimental Design, collocated with*

AISTATS, Sardinia, Italy, Volume 10, pp. 1–26.

Guyon, I., V. Lemaire, G. Dror, et D. Vogel (2009). Analysis of the kdd cup 2009 : Fast scoring on a large orange customer database. *JMLR : Workshop and Conference Proceedings 7*, 1–22.

John, G. et P. Langley (1995). Estimating continuous distributions in Bayesian classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, pp. 338–345. Morgan Kaufmann.

Langley, P., W. Iba, et K. Thompson (1992). An analysis of Bayesian classifiers. In *Proceedings of the National Conference on Artificial Intelligence*, Number 415, pp. 223–223.

Lim, T., W. Loh, et Y. Shih (2000). A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine learning 40*(3), 203–228.

Michalski, R. S., I. Mozetic, J. Hong, et N. Lavrac (1986). The Multi-Purpose incremental Learning System AQ15 and its Testing Application to Three Medical Domains. *Proceedings of the Fifth National Conference on Artificial Intelligence*, 1041–1045.

Quinlan, J. R. (1993). *C4.5 : programs for machine learning*. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc.

Settles, B. (2010). Active learning literature survey. `http://pages.cs.wisc.edu/~bsettles/pub/settles.activelearning.pdf`.

Witten, I. H. et E. Frank (2005). *Data mining : practical machine learning tools and techniques*. Morgan Kaufmann Series in Data Management Systems, Morgan Kaufmann, second edition.

Summary

Learning algorithms proved their ability to deal with large amount of data. Most of the statistical approaches use defined size learning sets and produce static models. However in specific situations: active or incremental learning, the learning task starts with only very few data. In that case, looking for algorithms able to produce models with only few examples becomes necessary. The literature's classifiers are generally evaluated with criteria such as: accuracy, ability to order data (ranking)... But this classifiers' taxonomy can really change if the focus is on the ability to learn with just few examples. To our knowledge, just few studies were performed on this problem. This study aims to study a larger panel of both algorithms (9 different kinds) and data sets (17 UCI bases).