Classification de documents XML à partir d'une représentation linéaire des arbres de ces documents

Anne-Marie Vercoustre*, Mounir Fegas*
Yves Lechevallier*, Thierry Despeyroux*

*INRIA Rocquencourt
B.P. 105 78153 Le Chesnay Cedex France
Prénom.Nom@inria.fr,
http://www-rocq.inria.fr

Résumé. Cet article présente un nouveau modèle de représentation pour la classification de documents XML. Notre approche permet de prendre en compte soit la structure seule, soit la structure et le contenu de ces documents. L'idée est de représenter un document par l'ensemble des sous-chemins de l'arbre XML de longueur comprise entre n et m, deux valeurs fixées a priori. Ces chemins sont ensuite considérés comme de simples mots sur lesquels on peut appliquer des méthodes standards de classification, par exemple K-means. Nous évaluons notre méthode sur deux collections: la collection INEX et les rapports d'activité de l'INRIA. Nous utilisons un ensemble de mesures bien connues dans le domaine de la recherche d'information lorsque les classes sont connues a priori. Lorsqu'elles ne sont pas connues, nous proposons une analyse qualitative des résultats qui s'appuie sur les mots (chemins) les plus caractéristiques des classes générées.

1 Introduction

XML est devenu un standard pour la représentation et l'échange de données. Le nombre de documents XML échangés augmente de plus en plus, et la quantité d'information accessible aujourd'hui est telle que les outils, même sophistiqués, utilisés pour rechercher l'information dans les documents ne suffisent plus. D'autres outils permettant de synthétiser ou classer de larges collections de documents sont devenus indispensables.

Dans ce contexte, de nombreux travaux proposent des méthodes de classification, supervisées ou non, pour organiser ou analyser de larges collections de documents XML. (Denoyer et al. (2003)) combinent plusieurs fonctions d'affectation (classifiers) pour classer des documents XML multimédia, (Despeyroux et al. (2005)) identifient, pour une collection homogène donnée, les types d'éléments XML les plus pertinents pour un objectif de classification. La similarité entre documents peut être définie en étendant le modèle vectoriel pour tenir compte de la structure (Doucet et Ahonen-Myka (2002), Yi et Sundaresan (2000)), ou seulement à partir de la structure d'arbre des documents, selon l'objectif visé ou l'hétérogénéité de la collection. Ainsi, la similarité structurelle peut être basée sur la distance entre arbres (Francesca et al. (2003), Nierman et Jagadish (2002), Dalamagas et al. (2004)), ou sur la détection de

```
<?xml version="1.0" ?>
<body>
  <sec sno="01">
    <st>Title of<it>Section 1</it>: </st>
    <p>This is a paragraph</p>
    <au>Name of the first author</au>
  </sec>
  <sec sno="02">
    <st>Title 2</st>
    <p1>Another type of paragraph</p1>
    <au> Name of the second author</au>
    <bib>References</bib>
  </sec>
  <sec sno="03">
    <st>Title 3</st>
    <p>One paragraph</p>
    <bib>References</bib>
  </sec>
</body>
```

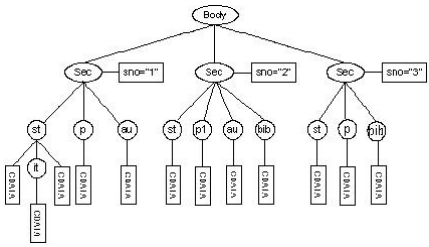

FIG. 1 – *Le document Test.xml et sa représentation sous forme d'arbre*

sous-arbres fréquents (Termier et al. (2002)), Costa et al. (2004)). Comme ces algorithmes ont une complexité assez élevée, les premiers travaillent sur des résumés d'arbres, tandis que les seconds sont basés sur les relations directes ou indirectes entre nœuds de l'arbre. Un inconvénient majeur est que la fréquence d'occurrence de certaines structures, comme les éléments de listes si fréquents dans les documents XML, est perdue pour certaines analyses.

Nous proposons ici une nouvelle représentation de documents XML en vue de la classification, qui permet de prendre en compte soit la structure seule, soit la structure et le contenu de ces documents. L'idée est de représenter un document par une linéarisation de son arbre XML en un ensemble de chemins générés selon certains paramètres. Les expressions de chemins vues comme de simples mots, et leur fréquence associée, permettent de se ramener au modèle vectoriel et d'appliquer une méthode simple et standard de classification. L'avantage est que l'on peut travailler sur de larges collections, tout en conservant certaines propriétés structurelles et textuelles des documents.

2 Représentation des documents XML

Les documents XML sont souvent représentés par des arbres étiquetés dont les étiquettes correspondent aux balises, et éventuellement au nom des attributs. Un exemple de document XML et de l'arbre correspondant est donné en figure 1. Intuitivement, un *chemin* est le plus court trajet entre la racine de l'arbre et un nœud et un sous-chemin est un segment de chemin. Nous proposons de représenter les documents XML par différents ensembles de sous-chemins. Cette représentation permet d'aplatir les documents tout en conservant une partie de l'information sur leur structure et donc de diminuer la complexité des traitements.

Notre motivation de prendre en compte les sous-chemins, et pas seulement les chemins, est que la ressemblance entre documents peut apparaître à différents niveaux. Pour des collections de documents très hétérogènes, les différences et ressemblances apparaitront dès les premiers nivaux (proche de la racine). Pour les collections plus homogènes, il peut être intéressant de ne conserver que les sous-chemins proches des feuilles, comme contexte au contenu textuel.

La notion de *chemin* que nous définissons maintenant couvre à la fois celles de chemin et de sous-chemin.

- Le **chemin** d'un nœud est la suite des nœuds visités à partir de la racine pour atteindre ce nœud, en parcourant l'arbre de fils en fils. Un chemin *terminal* est un chemin qui se termine par une feuille de l'arbre.
- L'ensemble des **sous-chemins** d'un nœud est l'ensemble des sous-séquences du chemin de ce noeud. Un sous-chemin est dit *terminal* s'il se termine par une feuille, *initial* s'il commence par le nœud racine.
- Un chemin ou un sous-chemin **textuel** est un chemin ou un sous-chemin prolongé par un mot du contenu textuel associé à un nœud. Lorsque le chemin est terminal, on considère le contenu textuel de la feuille. Si le chemin n'est pas terminal, on considère l'ensemble du contenu des feuilles qui sont des descendants du nœud considéré.
- La **longueur** d'un chemin ou d'un sous-chemin est égale au nombre de nœuds qui le constituent.

Dans la suite de cet article, nous utiliserons le terme **chemin** pour désigner indifféremment les **chemins** et les **sous-chemins**.

Un document XML sera représenté par un ensemble de chemins. L'ensemble des chemins représentant un document peut varier selon un certain nombre de paramètres définissant la génération de ces chemins et que nous pouvons choisir :

- `min, max` (entiers) : La longueur minimale et maximale des chemins (exprimée en nombre d'éléments). max = * signifie qu'il n'y a pas de contraintes sur la longueur maximale.
- `root` (booléen) : Selon que les chemins doivent être initiaux ou non.
- `leaf` (booléen) : Selon que les chemins doivent être terminaux ou non.
- `attrib` (booléen) : Selon que les chemins sont générés aussi pour les attributs ou non.
- `text` (booléen) : Selon que le contenu textuel est pris en considération ou non.

Comme nous verrons plus loin, ces différents paramètres permettent de couvrir des travaux menés par ailleurs.

Les chemins sont individuellement considérés comme des mots. Ce choix pose le problème d'une éventuelle dépendance entre eux. Par exemple "body.sec.st.it" et "sec.st.it" ne sont évidement pas totalement indépendants. Ceci est important lorsqu'on utilise des algorithmes de classification comme le K-means qui supposent que les mots sont indépendants, même si cela n'est pas complétement vérifié en pratique. Pour remédier à ce problème, nous avons séparé ces chemins en utilisant une variable par longueur (pour plus de détail, voir section 3).

Si l'option "`text`" est spécifiée, on récupère le contenu textuel du dernier nœud du chemin ainsi que celui de tous ses descendants, et chaque mot est attaché au chemin courant, créant autant de chemins textuels que de mots.

Les tables 1 et 2 présentent quelques ensembles de chemins, avec leur fréquence, pour l'exemple de la figure 1.

3 Algorithme de classification automatique

SClust est un algorithme, dit de *nuées dynamiques* (Celeux et al. (1989)), permettant de partitionner un ensemble de données en un nombre k (prédéfini) de classes homogènes (Verde et al. (2001)). Cette méthode est assez proche de celle de K-means, où la distance entre deux classes est basée sur la fréquence des mots du vocabulaire choisi (modalités). La principale différence réside dans la façon de recalculer les centres de gravité : une fois par itération pour

Mots	Tf
body.sec.bib	2
body.sec.st	2
body.sec.p	3
body.sec.au	2
body.sec.sno@	3

TAB. 1 – *Chemins non textuels de longueur 3 ; le caractère @ indique un attribut.*

Mots	Tf
body.sec.au."name"	2
body.sec.p."paragraph"	3
body.sec.bib."refer"	2
body.sec.sno@."01"	1
body.sec.st."tit"	3
body.sec.sno@."03"	1
body.sec.p."type"	1
body.sec.au."author"	2
body.sec.sno@."02"	1

TAB. 2 – *Chemins textuels de longueur 4*

les *nuées dynamiques*, à chaque affectation d'objet dans la classe pour le *K-means*. Si l'on utilise la distance euclidienne, la distance entre deux objets x et y est calculées par la formule : d(x,y) = $\sqrt{\sum_{j=1}^{m}(x_j - y_j)^2}$, où m est le nombre de modalités.

De plus, dans SClust, les objets peuvent être décrits par plusieurs variables, et chaque variable contient un ensemble de modalités caractérisant l'objet.

Par exemple, l'objet *voiture* possède les modalités *poids*, *longueur*, *largeur*, *hauteur* et *couleur*. Il existe clairement une dépendance entre le poids et les dimensions de la voiture, ce qui nous amène à créer deux variables, la première, *dimensions*, contenant les modalités *longueur*, *largeur* et *hauteur*, la seconde, *caractéristiques*, contenant *poids* et *couleur*.

Dans notre cas (partition des documents), les individus sont les documents XML et les modalités sont représentées par l'ensemble des chemins regroupés par longueur où chaque longueur représente une variable. La distance entre deux documents x et y est alors remplacée par la formule : d(x,y) = $\sqrt{\sum_{k=1}^{p}\sum_{j=1}^{m_k}(x_j^k - y_j^k)^2}$, où p est le nombre de variables et m_k est le nombre de modalités pour la variable p.

La complexité d'un algorithme de classification automatique est difficile à calculer, car elle dépend de la phase d'initialisation qui est faite de manière aléatoire. Généralement, on peut dire qu'elle est linéairement dépendante de la taille des données, du nombre d'exécutions (dû à l'initialisation aléatoire) et du nombre d'itérations (fixés par l'utilisateur).

4 Évaluation et mesures

Pour évaluer de notre algorithme, quatre métriques sont employées. Elles permettent de comparer le résultat de la classification avec une classification existante, lorsque celle-ci existe :

- **Entropie** : mesure la façon dont les classes prédéfinies sont distribuées ou réparties dans chaque classe calculée. Une valeur faible de l'entropie correspond à un meilleur paritionnement (Zhao et Karypis (2001)).
- **Pureté** : mesure la proportion de documents de la classe calculée appartenant à la classe à priori la plus fréquente, c'est à dire le pourcentage de documents de la classe calculée appartenant à la classe majoritaire (Zhao et Karypis (2001)). La pureté d'une classe calculée est égale à 1 si tous les documents sont issus de la même classe à priori.

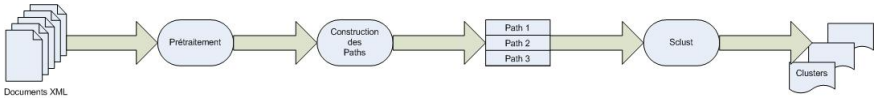

FIG. 2 – *Chaîne de Traitement*

– **F-measure** : proposée par Larsen et Aone (1999), combine les mesures de précision et de rappel pour essayer d'assigner chaque classe calculée à une classe prédéfinie de telle sorte que deux classes calculées ne peuvent pas être assignés à la même classe prédéfine. La F-measure mesure un équilibre entre la précision et le rappel. C'est la moyenne harmonique des deux valeurs précédentes.
– **Rand corrigé** : (ou *Corrected Rand*) proposé par Hubert et Arabie (1985) pour comparer deux partitions. On peut utiliser cette métrique pour comparer le résultat de la classification automatique avec une classification existante, ou bien de comparer deux partitions obtenues à partir de l'algorithme de classification automatique.

5 Traitement des collections

Le traitement pour partitionner une collection de documents XML se compose de trois principales étapes qui sont montrées dans la Figure 2.

Le prétraitement consiste principalement à réduire le nombre de chemins générés, en essayant de diminuer successivement le nombre de balises, de mots et de chemins différents. Le filtrage sur les balises utilise une connaissance sémantique de ces balises. On peut, optionnellement :

– Remplacer les balises synonymes par un seul représentant.
Par exemple, remplacer $ss1$, $ss2$, $ss3$ par sec pour les sections ; $numeric\text{-}list$, $bullet\text{-}list$ par $list$ pour les listes ; $h1$, $h2$, $h3$, $h4$, $h1a$, $h2a$ par h pour les headers.
– Ignorer la descendance de certaines balises. Par exemple $math$ et $formula$ pour les formules mathématiques dont les descendants ont une granularité trop fine pour la classification, et $table$ pour les tables.
– Supprimer les balises de mise en forme du texte (bold, italic etc.).

Pour le texte, nous utilisons les méthodes classiques de réduction du vocabulaire :

– Utiliser une *Stopword List* qui est un ensemble de mots dits vides qui n'ont aucun rôle à jouer dans la classification automatique et donc ne doivent pas être inclus dans les chemins.
– Supprimer les mots de moins de 4 caractères.
– Stemming : pour le processus de normalisation des termes, nous avons utilisé l'algorithme de Porter (Porter's Stemmer) (Porter (1997)). Cet algorithme supprime les suffixes des mots en anglais. La suppression des suffixes permet de réduire la taille et la complexité des données pour améliorer les performances. Le principale avantage de cet algorithme est sa rapidité de traitement de grand volume de données.

Enfin nous réduisons le nombre final de chemins en utilisant les fréquences relatives TfIdf de ces chemins : les chemins trop fréquents ou trop rares sont éliminés.

6 Expériences et résultats

Pour valider notre approche, nous utilisons deux collections de documents XML. La première est celle d'INEX (INitiative for the Evaluation of XML Retrieval[1]) et la deuxième est la collection XML des rapports d'activité des 160 équipes de recherche de l'INRIA (Institut National de Recherche en Informatique et en Automatique) pour l'année 2004. Lorsqu'il existe des classes connues a priori, l'évaluation consiste à comparer les partitions obtenues avec les classes existantes, à l'aide des mesures décrites précédemment. Dans le cas contraire, nous nous contentons d'une analyse qualitative des classes générées.

6.1 Corpus INEX

Ce corpus permet aux participants d'INEX d'évaluer différentes méthodes pour la recherche d'information dans des documents XML. La collection comporte 12107 articles issus des publications de 12 magazines et 6 *transactions* de *IEEE Computer Society* entre la période de 1995 à 2002.

Dans Denoyer (2004), la collection a été utilisée pour évaluer des méthodes de classification en essayant de classer automatiquement les documents dans un des 18 journaux de la collection. Devant la mauvaise qualité des résultats, l'auteur a étiqueté la collection en 6 thèmes (Ordinateur, Graphisme, Hardware, Intelligence Artificielle, Internet et Architecture Parallèle) et 2 classes structurelles (les *transactions* et les autres articles). Nous avons repris ces classifications pour notre évaluation.

Différentes expériences sont lancées sur ce corpus, en utilisant soit la structure seule, soit la structure et le contenu. Avec la structure seulement, nous cherchons à séparer les 2 classes structurelles (*transactions* vs articles). En utilisant la structure et le contenu, nous cherchons à retrouver soit les 6 thèmes prédéfinis, soit les 18 journaux initiaux.

6.1.1 Utilisation de la structure seule : *transactions* vs articles

Dans cette expérience, nous avons représenté les documents par leurs chemins de longueur comprise entre 3 et 5 (groupés dans trois variables), commençant à la racine. La première variable essaie donc de partitionner à partir des structures de haut niveau, tandis que la deux autres variables pourront éventuellement capter des similarités dans des parties plus détaillées du document.

La figure 3 montre, pour différents nombres de classes, les valeurs des différentes mesure de similarité avec les 2 classes existantes.

Nous remarquons que les métriques sont cohérentes entre elles et atteignent leurs valeurs optimales lorsque le nombre de classes calculées est égal à 4. Dans ce cas, la répartition des deux classes (*transaction* et article) à l'intérieur de chaque classe calculée est montrée dans la figure 4.

Nous constatons que les articles se répartissent sur les classes 1, 2 et 3. Les *transactions* sont bien regroupées dans la classe 4. Dans la classe 1, nous constatons la présence de quelques *transactions* dont la structure est probablement assez proche de celle des articles.

Nous avons constaté que l'utilisation des attributs dans les chemins n'améliore pas les performances du système.

[1] http ://inex.is.informatik.uni-duisburg.de

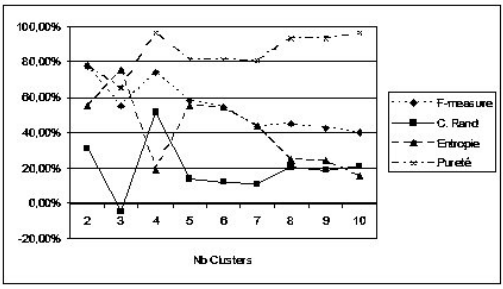

FIG. 3 – *Mesures d'évaluation (classification en articles et transactions) en fonction du nombre de classes (structure seule).*

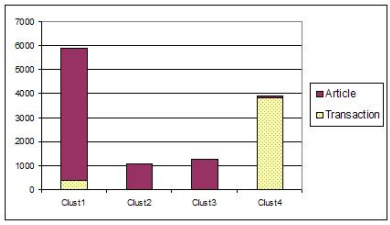

FIG. 4 – *Répartition des classes à priori Articles et Transactions sur quatre classes calculées.*

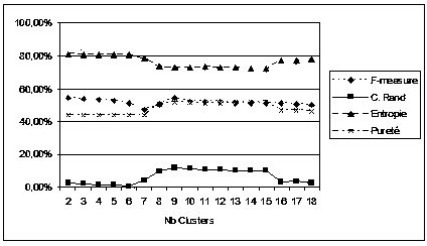

FIG. 5 – *Mesures d'évaluation (classification thématique) en fonction du nb de classes (structure + contenu).*

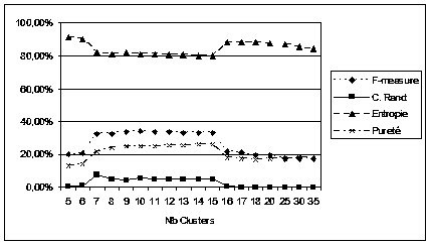

FIG. 6 – *Mesures d'évaluation (classification en journaux) en fonction du nombre de classes (structure + contenu).*

6.1.2 Utilisation de la structure et du contenu

Dans ces expériences, nous n'avons utilisé que 5% de documents de la collection d'INEX (tirés au hasard), pour limiter la taille des données et donc les temps de calculs. Nous présentons ici les résultats en utilisant des chemins textuels de longueur 3, commençant à la racine, faute de place pour présenter plus de résultats. Nous considérons les deux classifications : les 6 thèmes et les 18 journaux.

La figure 5 montre la valeur des différentes mesures pour la comparaison avec les 6 thèmes, en faisant varier le nombre de classes calculées.

On peut noter que les quatre métriques sont cohérentes entre elles et ceci lorsque le nombre de classes calculées varie entre 8 et 15. Le meilleur regroupement est celui en 9 classes, qui permet d'avoir des valeurs optimales pour l'ensemble des métriques.

Une deuxième façon de prendre en compte le contenu des documents XML consiste à attacher le texte aux chemins terminaux. Nous avons constaté que les résultats diffèrent peu, selon que l'on utilise des chemins textuels commençants à partir de la racine ou des chemins textuels terminaux. Cela vient sans doute de la faible profondeur des arbres.

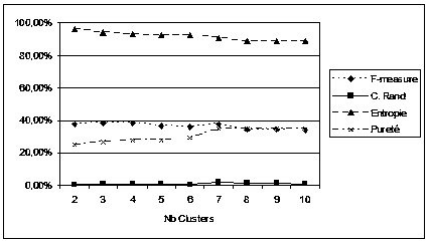

FIG. 7 – *Mesure d'évaluation (classification en Thèmes du RA) en fonction du nombre de classes (structure seule).*

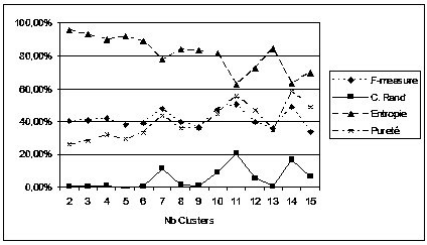

FIG. 8 – *Mesure d'évaluation (classification en Thèmes du RA) en fonction du nombre de classes (structure + contenu).*

La figure 6 montre la valeur des différentes mesures pour la comparaison avec les 18 journeaux, en faisant varier le nombre de classes.

Les résultats obtenus sont moins bons que ceux de la classification thématique. Ceci est due à l'hétérogénéité de la structure et du contenu des documents dans chaque revue, et confirme les résultats obtenus dans Denoyer (2004).

6.2 Rapports d'activités INRIA (RA)

L'INRIA publie un rapport d'activité annuel dont l'annexe scientifique est accessible sur le web. C'est un ensemble de rapports en anglais produits par chaque équipe de recherche. Pour plus de détails sur la structure de ces rapports, se reporter à (Despeyroux et al. (2005)).

Les équipes de recherches sont regroupées en 5 thèmes : Systèmes Biologiques, Systèmes Cognitifs, Systèmes de Communication, Systèmes Numériques et Systèmes Symboliques. On peut supposer que les rapports d'activité des équipes refléteront plus ou moins ces thèmes dans la présentation de leur activité de recherche et donc que l'on puisse retrouver ces thèmes en classant les documents à partir de leur contenu. Par contre, l'hypothèse est qu'en utilisant seulement la structure des rapports, on a aucune chance d'identifier ces thèmes. En effet la structure des rapports est basée sur un modèle commun (DTD), avec peu de parties optionnelles.

6.2.1 Utilisation de la structure seule

La figure 7 montre que l'hypothèse précédente est vérifiée, puisqu'en utilisant la structure seule (chemins non textuels de longueur comprise entre 3 et 5), la valeur des différentes mesures est uniformément mauvaise, quelque soit le nombre de classes (en particulier le Rand Corrigé est proche de zero). Ceci est tout à fait logique car, comme indiqué en section 4, le Rand Corrigé permet de comparer deux partitions. Or, les deux partitions sont construites de manière totalement différente (l'une est basée sur la structure et l'autre sur le contenu).

Par contre, nous pouvons essayer d'analyser quantitativement le résultat d'une partition en n classes.

Le tableau 3 présente les chemins les plus fréquents pour 7 classes (par ordre d'importance). Nous nous sommes limités à ne présenter que les chemins de longueur 2 pour plus de clarté.

Classe 1	Classe 2	Classe 3	Classe 4	Classe 5	Classe 6	Classe 7
raweb.logiciels	raweb.topic	raweb.logiciels	raweb.international	raweb.domaine	raweb.contrats	raweb.logiciels
raweb.domaine	raweb.resultats	raweb.contrats	raweb.biblio	raweb.resultats		raweb.domaine
raweb.resultats			raweb.contrats			
raweb.biblio						
raweb.international						

TAB. 3 – *Les chemins importants dans les classes*

On peut interpréter ces résultats en disant, par exemple, que les équipes ayant plus de logiciels et de contrats de collaboration se trouvent dans la classe 3, et que les équipes de la classe 4 se caractérisent par la diversité des activités internationales, le nombre important des publications et des contrats de collaboration. Il faut comprendre qu'un chemin comme raweb.logiciels synthétise (dans le tableau de résultats) en fait beaucoup de chemins de longueur 3 (raweb.logiciels,subsection) correspondant à la descriptions des différents logiciels. On voit ici l'importance pour certaines analyses de conserver tous les éléments de liste et pas seulement leur existence ou non existence.

6.2.2 Utilisation de la structure et du contenu

En utilisant des chemins textuels, nous constatons (figure 8) une amélioration des différentes mesures de comparaisons avec les Themes. Globalement, Le meilleur regroupement est celui en 11 classes, où l'on constate une décroissance de l'entropie, ce qui signifie que les documents ne sont pas trop éparpillés sur les classes, et une augmentation du Rand corrigé, signifiant que ce regroupement se rapproche un peu de la classification en thèmes.

7 État de l'art

Un aspect important et caractérisant des travaux sur la classification de documents XML est la manière de représenter les documents.

Doucet et Ahonen-Myka (2002) représentent un document XML en utilisant le modèle vectoriel dans lequel les modalités sont soit les balises (balises XML), soit les mots du texte (contenu), soit une combinaison des deux (balises et texte). La technique TfIdf est utilisée pour normaliser les éléments du vecteur et l'algorithme de classification utilisé est le K-means. Ceci correspond à notre approche, dans le cas où l'on fixe respectivement la longueur des chemins non textuels à 1, textuels à 1, ou textuels et non textuels à 1.

Yi et Sundaresan (2000), proposent un modèle vectoriel structuré afin de tenir compte de la structure des documents XML. Les éléments du vecteur sont des termes (mots) ou un autre vecteur structuré (récursivité). Ceci revient a représenté chaque élément par son chemin à partir de la racine (nœud et mots du texte). Ceci est donc équivalent à notre représentation à partir de chemins commencant à la racine, avec les chemins textuels allant jusqu'aux feuilles. La méthode de classification probabiliste basée sur le modèle de génération de documents de Bernoulli est ensuite appliquée sur ce vecteur.

Une autre amélioration (Jianwu et Xiaoou (2002)) consiste à modéliser des éventuels liens entre les documents dans le SVM (Structured Link Vector Model), où les éléments du vecteur sont les termes, la structure et le voisinage des documents. La méthode K-means est utilisée

comme algorithme de classification. Notre approche ne tient pas compte des liens entre les documents.

Dans Yoon et al. (2001), deux représentations de documents sont proposées. La première consiste à utiliser une matrice binaire (*bitmap*) à 2 dimensions (2D) : les documents et les *ePath* (le chemin d'un élément feuille à partir de la racine). Cette représentation a été étendue à 3D : les documents, les ePath, et le contenu. Les ePaths correspondent au cas où nous utilisons les chemins de longueur quelconque allant de la racines au feuilles. Toutefois la methode de classification automatique est très différente.

Liu et al. (2004) traite des documents XML homogènes (même DTD). Cette approche travaille sur des arbres XML ordonnés pour extraire des informations selon trois méthodes : chemins, paires des nœuds (père-fils, frères, etc.) ou bien une combinaison des deux (hybride). Ces informations sont, ensuite, transformées en un vecteur de la forme (information, nombre d'occurrence). Pour réduire la taille du vecteur, la technique d'analyse en composante principale (ACP) est utilisée. La première représentation des documents est celle qui est la plus proche de celle que nous utilisons (chemins de longueur inférieure à une certaine valeur, commençant à un niveau donné dans l'arbre). L'idée d'utiliser des chemins commençant à un certain niveau mériterait d'être explorée, car elle peut faciliter l'interprétation des résultats.

D'autres travaux porte sur des arbres étiquetés, en utilisant plusieurs techniques, comme la détection de sous-arbres fréquents (Termier et al. (2002)), le *tree-matching* (Costa et al. (2004)) et les résumés d'arbres (Dalamagas et al. (2004)). Ces travaux utilisent une représentation des documents à partir des relations binaires entre nœuds qui s'éloignent de notre représentation. Les méthodes de classification utilisées sont également très différentes et plus appropriées au traitement de collections hétérogènes.

8 Conclusion

Nous avons proposé dans ce travail un nouveau modèle de représentation des documents XML qui prend en considération la structure et le contenu de ces documents. Ce modèle est basé sur la notion de linéarisation des chemins qui nous a permis d'aplatir les documents. Avec cette représentation, nous avons pu ensuite appliquer un algorithme de classification simple et performant, en l'occurrence SClust, et montré nos résultats obtenus sur les deux collections, INEX et les rapports d'activité.

Bien que les deux collections utilisées ne soient pas très adaptées aux problèmes de classification automatique thématique à partir de la structure, notre modèle nous a permis tout de même de séparer les documents de structure différente (le cas 'articles vs *transactions*' dans INEX), et d'exhiber des propriétés inconnus des documents en analysant les parties des DTD les plus utilisées. Ce modèle nous offre aussi la possibilité de reconstruire les DTDs, et ceci en analysant l'ensemble des chemins représentant chaque classe.

Comme travaux futurs, nous envisageons les points suivants :

- Prendre en compte l'ordre des nœuds dans l'arbre XML (modèle d'arbres ordonné).
- Appliquer notre approche sur d'autres collections de documents XML, comme MovieDB et WIPO, et surtout sur des collections dédiées au XML Mining comme les collections du track INEX 'XML Mining', collections synthétiques éventuellement plus hétérogènes .

– Utiliser des métriques qui ne nécessite pas d'avoir une classification *a priori* comme : *la variance inter-classes* et *intra-classe*.

Références

Celeux, G., E. Diday, G. Govaert, Y. Lechevallier, et H. Ralambondrainy (1989). *Classification Automatique des Données, Environnement statistique et informatique*. Dunod informatique, Bordas, Paris.

Costa, G., G. Manco, R. Ortale, et A. Tagarelli (2004). A Tree-Based Approach to Clustering XML Documents by Structure. In *PKDD*, pp. 137–148.

Dalamagas, T., T. Cheng, K.-J. Winkel, et T. K. Sellis (2004). Clustering XML Documents Using Structural Summaries. In *EDBT Workshops*, pp. 547–556.

Denoyer, L. (2004). *Apprentissage et inférence statistique dans les bases de documents structurés : Application aux corpus de documents textuels*. Ph. D. thesis, L'UNIVERSITÉ DE PARIS 6.

Denoyer, L., J.-N. Vittaut, P. Gallinari, S. Brunesseaux, et S. Brunesseaux (2003). Structured Multimedia Document Classification. In *ACM Document Engeneering*, Grenoble.

Despeyroux, T., Y. Lechevallier, B. Trousse, et A.-M. Vercoustre (2005). Experiments in clustering homogeneous xml documents to validate an existing typology. In *Proceedings of the 5th International Conference on Knowledge Management (I-Know)*, Vienne, Autriche.

Doucet, A. et H. Ahonen-Myka (2002). Naïve Clustering of a large XML Document Collection. In *INEX Workshop*, pp. 81–87.

Francesca, F. D., G. Gordano, R. Ortale, et A. Tagarelli (2003). Distance-based Clustering of XML Documents. In L. De Raedt et T. Washio (Eds.), *MGTS-2003 : Proceedings of the First International Workshop on Mining Graphs, Trees and Sequences*, pp. 75–78. ECML/PKDD'03 workshop proceedings.

Hubert, L. et P. Arabie (1985). Comparing partitions. *Journal of Classification 2*, 193–218.

Jianwu, Y. et C. Xiaoou (2002). A semi-structured document model for text mining. *J. Comput. Sci. Technol. 17*(5), 603–610.

Larsen, B. et C. Aone (1999). Fast and effective text mining using linear-time document clustering. In *KDD '99 : Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA, pp. 16–22. ACM Press.

Liu, J., J. T. L. Wang, W. Hsu, et K. G. Herbert (2004). XML Clustering by Principal Component Analysis. In *ICTAI*, pp. 658–662.

Nierman, A. et H. V. Jagadish (2002). Evaluating Structural Similarity in XML Documents. In *Proceedings of the Fifth International Workshop on the Web and Databases (WebDB 2002)*, Madison, Wisconsin, USA.

Porter, M. F. (1997). An algorithm for suffix stripping. In *Readings in information retrieval*, San Francisco, CA, USA, pp. 313–316. Morgan Kaufmann Publishers Inc.

Termier, A., M.-C. Rousset, et M. Sebag (2002). TreeFinder : a First Step towards XML Data Mining. In *ICDM '02 : Proceedings of the 2002 IEEE International Conference on Data*

Mining (ICDM'02), Washington, DC, USA, pp. 450. IEEE Computer Society.

Verde, R., F. A. T. De Carvalho, et Y. Lechevallier (2001). A Dynamical Clustering Algorithm for Symbolic Objects, Tutorial on Symbolic Data Analysis. In *GfK1 Conference*, Munich, Allemagne.

Yi, J. et N. Sundaresan (2000). A classifier for semi-structured documents. In *KDD '00 : Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA, pp. 340–344. ACM Press.

Yoon, J. P., V. Raghavan, V. Chakilam, et L. Kerschberg (2001). BitCube : A Three-Dimensional Bitmap Indexing for XML Documents. *Journal of Intelligent Information Systems 17*(2-3), 241–254.

Zhao, Y. et G. Karypis (2001). Criterion functions for document clustering : Experiments and analysis. Technical Report 01–40, Department of Computer Science, University of Minnesota, Minneapolis, MN.

Summary

In this work, we propose a new clustering method for semi-structured documents collections. Our approach consists on a representation of XML documents based on their sub-paths, defined according to some criteria (length, root beginning, leaf ending) using the structure only or both the structure and the content. By considering those sub-paths as words, we can use standard methods for vocabulary reduction, and simple clustering methods such as K-means that scale up well. We actually use an implementation of the clustering algorithm known as *dynamic clouds* that can work with distinct groups of independent variables. This is necessary in our model since embedded sub-paths are not independent. For validation and evaluation of our method, two collections are used: the INEX[2] corpus and the activity reports of INRIA[3], and a set of metrics well-known in Information Retrieval.

[2]http://inex.is.informatik.uni-duisburg.de
[3]http://www.inria.fr