Modèle probabiliste pour l'extraction de structures dans les documents semi-structurés — Application aux documents Web

Guillaume Wisniewski — Ludovic Denoyer — Francis Maes — Patrick Gallinari

Laboratoire d'Informatique de Paris 6
8 rue du capitaine Scott
75015 Paris
{prenom. nom}@lip6. fr

RÉSUMÉ. *Le développement des systèmes de gestion de contenu a profondément changé la nature du Web : de plus en plus de documents sont créés automatiquement et leur mise en page reflète leur structure logique. Dans ce travail, nous montrons que l'information contenue dans la mise en page est suffisante pour inférer une structure sémantiquement riche, ce qui ouvre la voie à de nombreuses applications. Le passage d'une information de mise en page à une structure sémantique se heurte à deux principaux obstacles : l'hétérogénéité des données et le caractère implicite de de la structure des documents Web. Nous décrivons un modèle stochastique capable d'apprendre à transformer des documents semi-structurés vers un schéma défini a priori et présentons une instance particulière de ce modèle adaptée à la transformation de documents hétérogènes HTML en XML. Finalement, nous décrivons plusieurs expériences sur des corpus XML et HTML réels.*

ABSTRACT. *With content management system becoming mainstream the Web has changed dramatically: more and more web pages are now generated from relational databases and their design reflects the logical structure of documents. In this work, we show that there is enough information in the layout of a web document to capture the kind of data people are already producing in a more machine-friendly format. The extraction of a semantic structure from the layout of documents faces two main obstacles: structures are heterogeneous — they change with the source producing it — and often remain implicit. We introduce a general stochastic model of semi structured documents generation and transformation and detail an instance of this model for the particular task of HTML to XML conversion.*

MOTS-CLÉS : *Recherche d'information structurée, restructuration, modèle statistique, apprentissage, extraction de structure, schema matching*

KEYWORDS: *Structured Information Retrieval, Machine Learning, Document Restructuration, Schema Matching*

Introduction

Ces dernières années, le Web a profondément changé : avec le développement des blogs, des sites de nouvelles et, plus généralement, des sites basés sur des systèmes de gestion de contenu, de plus en plus de pages sont créées automatiquement à partir d'informations Stockées dans une base de données et d'un modèle de document. Désormais, la mise en page d'un document reflète sa structure logique et l'information est transmise aussi bien par le contenu du document (texte, image, ...) que par sa présentation. En effet, dès lors que celle-ci présente certaines régularités, la mise en page permet d'identifier des éléments dans un document (un titre, un commentaire, ..) et des relations entre ces éléments (on peut ainsi préciser l'auteur d'une sous-partie du document). Ce nouveau type d'information, directement lié à la présentation des documents, peut, par exemple, être utilisé pour créer automatiquement le plan d'un document ou organiser les commentaires des visiteurs d'un site en *threads* ou par ordre chronologique.

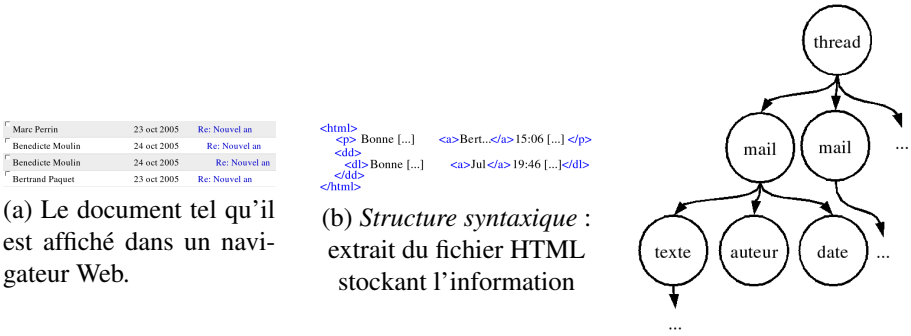
L'exploitation de la mise en page des documents a de nombreuses applications : faciliter la navigation sur le Web, en particulier sur des terminaux mobiles [BUY 00], améliorer les interfaces utilisateur, ... Elle peut aussi accroître l'efficacité de la Recherche d'Information, en permettant de cibler l'information pertinente à l'intérieur d'un document. Par exemple sur des sites d'actualité, tels Slashdot[1], les utilisateurs commentent abondamment chaque nouvelle postée et abordent souvent dans leurs commentaires des sujets n'ayant pas de lien direct entre eux. Par conséquent, il est devenu plus important pour un système de RI de retrouver les éléments pertinents, tels les commentaires, à l'intérieur d'un document plutôt que d'identifier les documents pertinents. L'utilisation de méta-informations (noms d'auteur, dates, ...) peut aussi permettre à l'utilisateur de retrouver un élément particulier. Aussi bien les éléments que les méta-informations sont identifiées par la mise en page des documents.

Dans ce travail, nous proposons d'extraire l'information contenue dans la mise en page pour définir une *structure de document* qui pourra être utilisée par différentes applications (aide à la navigation, moteur de recherche, ...). L'utilisation de formats semi-structurés permet d'inclure directement la mise en page dans les documents : ces formats permettent d'enrichir le texte et ainsi d'organiser l'information contenue dans un document en identifiant des *éléments* et des *relations* entre ceux-ci. Une première manière de définir la structure d'un tel document est alors d'interpréter la syntaxe utilisée par le format de fichier (par exemple, les balises XML ou les marqueurs des langages de wiki). Mais, cette *structure syntaxique*, directement liée à la manière dont l'information est stockée dans le fichier, est difficile à exploiter. En effet, sa signification reste souvent implicite : la nature exacte des éléments et des relations entre ceux-ci ne sont connus que de l'application ayant créé les documents. L'hétérogénéité de la structure syntaxique est un autre obstacle à son utilisation : chaque source de documents (chaque site Web) définit sa propre charte graphique et, bien que l'information

1. `slashdot.org`

soit identique, la structure syntaxique des documents peut varier, ce qui complique le développement de solutions indépendantes de l'origine des documents.

Même si leur structure syntaxique est différente, les documents parlant d'un sujet proche présentent certaines régularités. Par exemple, un article scientifique aura toujours une bibliographie et une description de film une distribution, même si la position de ceux-ci dans le document et leur présentation peuvent changer d'un site à l'autre. Nous proposons d'utiliser ces régularités pour définir une *structure de médiation* qui servira d'intermédiaire entre la structure syntaxique des documents et la structure de données utilisées par l'application envisagée. Cette structure de médiation va nous permettre de projeter la structure des pages Web vers une structure de plus haut niveau, directement reliée à la tâche considérée. Cette projection va nous permettre de définir un deuxième type de structure pour les pages Web, la *structure pragmatique*. La figure 1 décrit un exemple des différentes définition de la structure d'un document Web.



(a) Le document tel qu'il est affiché dans un navigateur Web.

(b) *Structure syntaxique* : extrait du fichier HTML stockant l'information

(c) *Structure pragmatique*

Figure 1. *Différentes définition de la structure d'un document web — un échange de mails*

Il est donc possible de définir plusieurs types de structure pour un document Web. Jusqu'à présent, toutes les approches utilisant à la fois l'information de contenu et l'information de structure, que ce soit en classification ou en recherche d'information [CAL 94, FUH 02], n'ont pris en considération que la structure syntaxique, notamment parce que ces travaux ne se sont intéressés qu'à des documents XML suivant un même schéma dont la sémantique était connue. Dans la plupart des cas, l'extension des méthodes développées à de nouveaux corpus issus notamment du Web se heurte à des problèmes liés à l'hétérogénéité des données ou aux caractères implicites des relations. La complexité de ces méthodes est un autre obstacle à leur mise en pratique : la complexité est généralement directement liée à la taille des documents (nombre d'étiquettes, de relations, ...) et l'utilisation de représentations trop fines rend de nombreuses approches inefficaces sur des corpus réels [CAL 94]. Nous pensons que l'utilisation d'une structure intermédaire de plus haut niveau, permettra de s'affranchir de ces deux problèmes.

Le passage de la structure syntaxique à la structure pragmatique constitue donc une première étape des systèmes de RI pouvant soit permettre d'accéder à une information de structure soit améliorier les performances de l'utilisation de celle-ci. L'écriture manuelle de médiateurs spécifiques à chacune des sources de documents est un travail long et coûteux qui est peu adapté à la nature dynamique du Web. C'est pourquoi nous considérons la tâche de restructuration qui consiste à transformer automatiquement des documents semi-structurés quelconques dans un schéma de médiation. Même si nous considérons ici plus spécifiquement la tâche de transformation de documents d'un format orienté présentation en documents XML, l'approche étudiée permet de traiter aussi les documents XML suivant différents schéma ou dont le schéma n'est pas connu.

Dans ce travail, nous considérons le problème de la conversion automatique d'un corpus de documents semi-structurés hétérogènes vers un schéma XML prédéfini. Nous proposons un cadre général permettant l'apprentissage de telles transformations (section 1) et détaillons l'application de celui-ci à l'extraction de structure pragmatique en décrivant la transformation de documents HTML en documents XML (section 2). Finalement nous présentons plusieurs séries d'expériences sur des corpus XML et HTML réels.

1. Modèle de restructuration

1.1. *Modèle de documents Web*

Aujourd'hui, la plupart des documents que l'on trouve sur le Web — par exemple les documents au format HTML ou PDF — peuvent être considérés comme des documents semi-structurés. Dans ce travail, nous nous limitons à ce type de documents.

Nous adoptons la représentation traditionnelle des documents semi-structurés sous forme d'un arbre ordonné : un document d, tel celui de la figure 2, est décrit par $\#d$ nœuds $(n_1, ..., n_{\#d})$. Deux types de nœuds peuvent être distingués : les *nœuds de contenu* qui segmentent le document en éléments et les *nœuds internes* qui décrivent les relations entre les différents éléments du document. Chaque nœud de contenu est associé à une information de contenu (texte, image, ...) ; chaque nœud interne est associé à une étiquette, à une liste d'enfants et à une information de contenu constituée par la concaténation de toutes les informations de contenu de ses enfants. Nous noterons c l'ensemble des nœuds de contenu et t l'ensemble des nœuds internes. Ils sont associés à un *schéma* qui définit un ensemble de règles et de contraintes que devront respecter les documents qui *valident* celui-ci.

1.2. *Approche générale*

Pour extraire la structure pragmatique d'un document, nous considérons la tâche générale de restructuration : étant donné un schéma arbitraire, nous souhaitons trans-

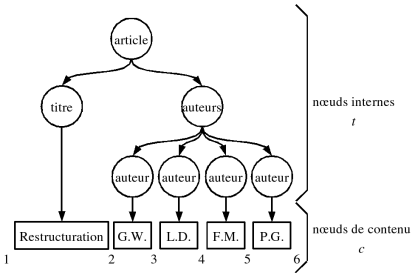


Figure 2. *Exemple de document semi-structuré. Les nœuds internes sont décrits par des cercles et les nœuds de contenu par des rectangles.*

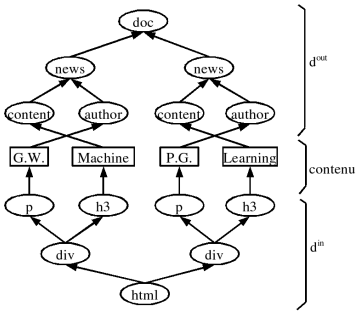


Figure 3. *Exemple d'un matching XML-HTML simple : les nouvelles sont identifiées par un tag div, les auteurs par un tag p et les titres par un tag $h3$*

former un document d'entrée d^{in} en un document de sortie d^{out} conforme à ce schéma. Cette transformation d'arbre peut inclure différents types d'opérations : réorganisation d'éléments — une bibliographie peut être présentée soit par auteurs, soit par année — , regroupement d'éléments — le nom et le prénom d'un auteur peuvent être stockés dans un élément ou dans deux —, etc. La Figure 3 montre un exemple de matching simple. La formalisation des transformations d'arbres est une problématique à part entière, dont la difficulté est bien connue [BEX 02]. C'est pourquoi, nous préférons utiliser des techniques d'apprentissage statistique à la fois pour éviter de décrire les transformations entrant en jeu et pour permettre de traiter les structures irrégulières.

Nous dirons qu'un document d est une *restructuration potentielle* d'un document d^{in} si, et seulement si, d contient les mêmes informations que d^{in} et respecte le schéma cible. Soit $\mathcal{D}$ l'ensemble des restructurations possibles d'un document. Le nombre d'éléments de $\mathcal{D}$ est potentiellement très grand. C'est pourquoi, nous définissons une *fonction de coût jointe* $\phi(d, d^{in})$ qui évalue à quel point un document d de $\mathcal{D}$ est une bonne restructuration de d^{in}. ϕ permet donc d'ordonner l'ensemble des restructurations potentielles. La tâche de restructuration peut alors être définie comme la recherche de la restructuration potentielle de plus petit coût :

$$d^{out} = \underset{d \in \mathcal{D}}{\operatorname{argmin}}\, \phi(d, d^{in}) \tag{1}$$

Le choix de la fonction de coût ϕ dépend de l'application envisagée. Cette formulation de la tâche de restructuration permet de considérer celle-ci comme un problème *d'apprentissage structuré* [TSO 04] et de bénéficier des différentes techniques d'apprentissage dévéloppées dans ce cadre. Étant donné leur complexité, ces méthodes ne sont pas applicables à des corpus de grande taille comme ceux habituellement utilisés en RI. Nous proposons donc un cadre stochastique à la tâche de restructuration basé sur un modèle génératif de documents.

1.3. *Modèle probabiliste de restructuration*

1.3.1. *Processus de génération des documents Web*

La régularité des structures pragmatiques et l'utilisation croissante de systèmes de gestion de contenu, nous amènent à supposer que tous les documents d'un même domaine (l'ensemble des articles scientifiques par exemple) sont générés à partir d'une unique source d'informations. Cette source définit une représentation *abstraite* de chaque document à partir de laquelle il est possible de générer toutes les structures syntaxiques possibles de ce document en spécifiant un *modèle* de document. Dans le cas des sites de cinéma, h pourrait, par exemple, représenter une base de données relationnelle à partir de laquelle on générerait les pages des différents sites (AlloCiné, IMDb, ...) ou des documents dans différents formats de présentation (Word, PDF, ...), chaque site spécifiant un modèle de documents qui décrit comment mettre en forme les données fournies par la source d'information.

Plus formellement, ce processus est modélisé par le réseau bayésien de la figure 4 : en connaissant un patron de document p^{in} et une représentation abstraite h, il est possible de générer une représentation d^{in} de cette information. h est une variable aléatoire cachée dont la nature exacte dépend de l'application et du type de documents considérés, p^{in} est un modèle de documents permettant de décrire les contraintes et les régularités qui caractérisent les sources de documents considérées et d et d^{in} sont des variables aléatoires qui représentent les documents suivant, respectivement, le schéma de médiation et le schéma spécifié par le modèle de document p^{in}. Nous verrons au paragraphe 2 comment spécifier les valeurs de ces différentes variables pour la transformation de documents HTML en XML.

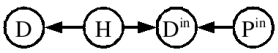


Figure 4. *Le réseau bayésien modélisant le processus de génération des différentes représentations d'un document à partir d'un template d'entrée p^{in} et d'une source d'information h.*

1.3.2. *Application à la tâche de restructuration*

À partir du processus de génération de documents Web décrit au paragraphe précédent, nous proposons d'utiliser la fonction de coût suivante : $\phi(d, d^{in}) = 1 - p(d|d^{in}, p^{in})$. Cette fonction correspond à une mesure de la dissimilarité entre un document d'entrée d^{in} généré à partir d'un patron p^{in} et le document m exprimé dans le schéma de médiation qui a été généré à partir du même document abstrait h. L'équation [1] se réécrit alors :

$$m = \underset{d \in \mathcal{D}}{\operatorname{argmax}}\, p(d|d^{in}, p^{in})$$

l'argmax traduit le parcours de l'espace de toutes les restructurations possibles. Cette recherche est contrainte à la fois par le contenu et la structure du document source et le modèle de document cible.

À partir du réseau bayésien décrit figure 4, nous pouvons estimer $p(d|d^{in}, p^{in})$:

$$p(d|d^{in}, p^{in}) = \frac{p(d, d^{in}, p^{in})}{p(d^{in}, p^{in}} = \frac{\sum_h p(h) \cdot p(p^{in}) \cdot p(d^{in}|h, p^{in}) \cdot p(d|h)}{p(d^{in}, p^{in})}$$

la somme se faisant sur toutes les représentations abstraites possibles. En appliquant la règle de Bayes, l'équation précédente peut se réécrire :

$$p(d|d^{in}, p^{in}) = \sum_h p(h|d^{in}, p^{in}) \cdot p(d|h)$$

Le problème de restructuration peut donc se décomposer en une étape d'*extraction* permettant de construire la représentation abstraite à partir d'une observation et en une étape de *génération* ré-exprimant les informations extraites dans le schéma de médiation ; ces deux étapes sont intimement liées.

Finalement, dans le cadre de notre description probabiliste, la tâche de restructuration peut donc se définir comme la résolution de :

$$m = \underset{d \in \mathcal{D}}{\operatorname{argmax}} \sum_h p(h|d^{in}, p^{in}) \cdot p(d|h) \quad (2)$$

2. Application de la restructuration au passage HTML-XML

Nous considérons maintenant une application du modèle de restructuration présenté au paragraphe précédent à l'extraction des structures pragmatiques des documents Web en convertissant ceux-ci vers un schéma XML défini a priori.

2.1. *Modèle de représentation abstraite*

La résolution de l'équation [2] nécessite de disposer d'informations supplémentaires sur la représentation abstraite h. Nous supposerons que les nœuds de contenu sont les mêmes dans tous les documents générés à partir d'une représentation cachée et que l'ordre de ces nœuds n'est pas modifié. À l'heure actuelle, toute les approches de restructuration proposées ([CHI 05] par exemple) font une telle hypothèse de séquentialité afin de limiter la taille de l'espace de recherche. De plus, une telle hypothèse est généralement vérifiée dans les corpus de documents textuels, dans lesquels le contenu est naturellement ordonné. De manière plus formelle, nous supposons que la representation cachée h définie une suite de nœuds de contenu c^h (cf : Figure 2) et que :

$$p(h|d^{in}, p^{in}) = \begin{cases} 0 & \text{si } c^h \neq c^{in} \\ 1 & \text{sinon} \end{cases}$$

L'équation 2 peut alors se réécrire :

$$m = \underset{d \in \mathcal{D}}{\operatorname{argmax}} \sum_d p(h|d^{in}, t^{in}) \cdot p(d|h) = \underset{d=(c,t) \in \mathcal{D}}{\operatorname{argmax}} p(c, t|h) = \underset{t \in \mathcal{T}(c^{in})}{\operatorname{argmax}} p(t|c^{in})$$

où $\mathcal{T}(c^{in})$ est l'ensemble des arbres respectant le schéma de médiation et dont la séquence des feuilles est c^{in}. Dans ce cadre, la tâche de reconstruction se définit donc comme la construction d'un arbre à partir d'une séquence d'éléments.

2.2. *Modèle de document semi-structuré*

Pour résoudre le problème de la restructuration tel que nous l'avons défini dans le paragraphe précédent, nous avons besoin de définir un *modèle de documents* pour estimer la probabilité de générer un document d : $p(d) = p(n_1, ..., n_{\#d})$. Ce modèle de document doit nous permettre de caractériser les éléments sémantiquement équivalents de deux documents. Cette équivalence est un concept difficile à définir. Dans ce travail, nous proposons d'utiliser la définition suivante : deux éléments sont équivalents si leur contenu est similaire (une taille est définie par un nombre et une unité) ou s'ils sont constitués d'éléments équivalents (une date est composée d'un jour, d'un mois et d'une année même si la représentation et l'ordre de ces éléments peuvent varier). Par conséquent, nous proposons de modéliser les documents par les relations d'inclusion de leurs éléments (ie : par les couples $(parent, enfant)$). Pour limiter la taille de l'espace des paramètres, nous supposons aussi que les paramètres d'un nœud ne dépendent que de l'étiquette de celui-ci. Plus formellement, nous faisons à la fois une hypothèse d'indépendance verticale et d'indépendance horizontale en supposant que la probabilité qu'un nœud n_i apparaisse dans un document d ne dépend que du père de n_i, de son prédécesseur et du contenu c_i du nœud : $p(d) = \prod_{i=1}^{\#d} p\left(tag(n_i)|pere(n_i), frere(n_i), c_i\right)$ où $tag(n_i)$ est l'étiquette du nœud n_i, $pere(n_i)$ l'étiquette de son père, $frere(n_i)$, l'étiquette du prédécesseur de celui-ci et c_i son contenu.

2.3. *Apprentissage*

Nous avons choisi de modéliser les probabilités $P(tag(n_i)|pere(n_i), frere(n_i), c_i)$ par une distribution de la forme :

$$P(tag(n_i)|pere(n_i), frere(n_i), c_i) = \frac{\exp\left(\langle W_{n_i}, F_{pere(n_i),frere(n_i),c_i}\rangle\right)}{Z_{pere(n_i),frere(n_i),c_i}}$$

où Z est un facteur de normalisation, F le vecteur de caractéristiques et W_α le vecteur de paramètres. $\langle\cdot,\cdot\rangle$ dénote le produit scalaire usuel. Les caractéristiques utilisées pour décrire le contexte de chaque nœud prennent en compte des informations sur la structure (les relations père-fils) et sur le contenu (ponctuation, nombre de majuscules, ... et autres critères utilisés habituellement en extraction d'information).

L'apprentissage du modèle nécessite que l'on estime, pour chaque étiquette α apparaissant dans le corpus d'apprentissage, la valeur du vecteur de paramètres W_α. Pour cela, nous avons utilisé BFGS, une méthode fondée sur le principe de maximisation de l'entropie [BER 96].

2.4. *Reconstruction d'un document*

La restructuration d'un document d^{in} est donc obtenue en résolvant l'équation :

$$m = \underset{d \in \mathcal{D}}{\operatorname{argmax}} \prod_{i=1}^{\#d} \frac{\exp\left(\langle W_{n_i}, F_{pere(n_i),frere(n_i),c_i} \rangle\right)}{Z_{pere(n_i),frere(n_i),c_i}} \quad (3)$$

Nous avons utilisé deux méthodes pour résoudre cette équation. La première méthode utilise une technique de programmation dynamique proche des algorithmes utilisés dans les analyseurs syntaxiques et permet de déterminer une solution exacte de l'équation [3]. Sa complexité dans le pire des cas est en $\mathcal{O}(n^3 \cdot V)$ où n est le nombre de feuilles du document d'entrée et V le nombre d'étiquettes du schéma de médiation. Bien que fournissant une solution exacte, sa complexité rend cette méthode inutilisable pour la reconstruction de grands documents. La deuxième méthode utilise l'algorithme LaSO [Dau 05], qui permet d'apprendre à rechercher de manière efficace une solution approchée de [3]. La complexité de la reconstruction est alors en $\mathcal{O}(n \cdot V \cdot N)$ où N est le nombre de nœuds du document reconstruit.

3. Expériences

3.1. *Corpus et mesures d'évaluation*

Le modèle de restructuration présenté dans ce travail a été testé sur trois corpus différents. Dans chaque cas, les documents des corpus ont été transformés en HTML et le fichier XML a été reconstruit à partir des informations contenues dans le fichier HTML.

Le premier corpus est le corpus Inex qui rassemble près de 12 000 articles scientifiques. Les documents ont, en moyenne, près de 500 nœuds de contenu et 200 nœuds internes. Il y a 139 tags différents. C'est une collection de petite taille pour la RI, mais dont la taille pose déjà problème dans la tâche de restructuration. Le deuxième corpus a été généré à partir de la base de données d'IMDb[2] : les 10 000 plus gros documents ont été exportés à la fois en XML et en HTML. Ces documents ont, en moyenne, 100 nœuds de contenu et 35 nœuds internes. Les documents sont beaucoup plus réguliers que ceux d'Inex. Le dernier corpus est constitué de 39 pièces de Shakespeare. Ce corpus n'a qu'un très petit nombre de documents, mais ceux-ci sont particulièrement grands : plus de 4 100 nœuds de contenu et 850 nœuds internes en moyenne.

Chaque corpus a été séparé aléatoirement en un ensemble de test et un ensemble d'apprentissage de même taille. À cause de sa complexité, la méthode de reconstruction basée sur la programmation dynamique n'a été utilisée que sur les plus petits documents (moins de 150 nœuds de contenu) de chaque collection, ce qui correspond à 2 200 documents du corpus INEX et à 4 000 documents du corpus d'IMDb. Cette méthode de reconstruction n'est pas applicable au troisième corpus.

2. `www.imdb.com`

Nous proposons d'évaluer la qualité de la restructuration, non pas par rapport à une application, mais en mesurant la similitude entre le document reconstruit et le document utilisé pour générer le HTML. Deux types de mesures d'évaluation ont été utilisés. La première mesure permet d'évaluer la qualité globale de la reconstruction en comparant les *couvertures* de chaque nœud : nous considérons l'erreur de classification sur les triplets (e, i, j) où i et j sont les indices respectifs de la première et de la dernière feuille couverte par l'étiquette e. Par exemple, sur la figure 2, on peut voir que la couverture de $auteurs$ est $(2, 6)$ et que celle de $titre$ est $(1, 2)$. C'est la mesure d'évaluation traditionnellement utilisée en TAL pour évaluer la qualité des arbres construits par un analyseur syntaxique. La deuxième est constituée par le pourcentage de documents dont plus d'un certain pourcentage de nœuds a été correctement reconstruit. Cette mesure d'évaluation est plus adaptée à un système développé dans le cadre de la RI pour lequel il n'est pas nécessaire de reconstruire parfaitement un document.

3.2. *Résultats*

Les figures 5 et 6 rassemblent les résultats des deux méthodes de reconstruction sur les différents corpus. Les performances sur les différents corpus sont encourageantes : on arrive à reconstruire plus de 90% des nœuds des corpus les corpus IMDb, Shakespeare et près de 70% des nœuds d'Inex, qui est un corpus ayant nettement plus d'étiquette possible et présentant nettement moins de régularité. Il semblerait donc qu'il y ait suffisamment d'information de structure dans les documents HTML pour pouvoir reconstruire les documents XML originaux. Toutefois, en distinguant la reconstruction des feuilles de celle des nœuds internes, il apparait que le score obtenu sur ces denières est nettement plus faible que les score obtenu sur les feuilles. Ainsi, bien que notre approche permette d'identifier les éléments correctement, l'extraction des relations entre ces éléments pose encore problème. De manière générale, la reconstruction utilisant une méthode à base de programmation dynamique est plus efficace que la méthode utilisant une approche LaSO. Un parcours exhaustif de l'espace de recherche est nécessaire pour obtenir de bonnes performances de reconstruction. Toutefois cette amélioration des performances à un coût : le temps de reconstruction est beaucoup plus important pour les méthodes DP que pour les méthodes LaSO, ce qui rend cette méthode de reconstruction inutilisable en pratique.

Les résultats de la Figure 6 montrent que l'on peut facilement reconstruire des documents dont la structure est approximativement la même que celle du document original. Il reste toutefois à évaluer l'impact d'une reconstruction imparfaite sur l'application utilisant les données.

4. État de l'art

L'automatisation de l'intégration de données — tâche de *schema matching* — a été étudiée depuis longtemps par la communauté base de données. Une comparaison des différentes approches proposées récément est faite dans [DOA 05]. [DOA 03] présente

Corpus	Méthode	Feuilles	Nœuds internes	Arbre complet	durée apprentis-sage	durée recons-truction
INEX	DP	**79.6**%	51.5%	**70.5**%	30 mn	$\simeq$ 4 j.
	LaSO	75.8%	**53.1**%	67.5%	$>$ 1 s.	**3h20min**
Movie	DP	**95.3**%	77.1%	**90.4**%	20 mn	$\simeq$ 2 j.
	LaSO	90.5%	**86.8**%	89.6%	$>$ 1 s.	**1h15min**
Shak.	LaSO	95.3%	77.0%	92.2%	$\simeq$ 5 j.	**30 min**

Figure 5. *Résultats des expériences de restructuration : mesure de l'erreur de reconstruction. Une estimation de la durée d'apprentissage et de reconstruction est donnée. Les expériences ont été menées sur des Pentium 3,2 GHz*

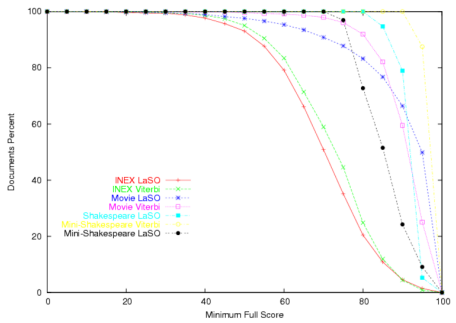


Figure 6. *Résultat des expériences : mesure du pourcentage de documents reconstruit correctement à x%*

une des approches les plus abouties pour travailler sur différents types de données (SQL, XML, ontologies, ...). La tâche de schema matching y est présentée comme un problème de classification supervisée multi-étiquettes. Toutefois les corpus considérés sont très différents des corpus auxquels nous nous intéressons : l'évaluation des méthodes développées a, généralement, été faite sur des corpus de petite taille, ayant une structure très stricte et ne contenant que très rarement des données textuelles.

Le modèle de documents utilisé a de nombreuses similarités avec les modèles utilisés en apprentissage dans les domaines structurés. Les hypothèses d'indépendance que nous avons faites sont proches de celles des HMMs hiérarchiques [FIN 98]. De nombreux travaux tels [YOU 00] ont proposé d'utiliser les méthodes développées en TAL (PCFG, analyse syntaxique, ...) pour modéliser la structure d'un document, mais une première série d'expériences [WIS 05] a montré que ces méthodes, bien qu'efficaces, avaient une complexité qui les rendait inutilisables sur des corpus de grande taille, tel Inex. Le travail le plus proche du nôtre est [CHI 05] qui s'intéresse à une problématique proche de la nôtre avec une méthode similaire : ils caractérisent le contenu par un classifieur maximisant le critère d'entropie et utilisent des grammaires probabilisées pour reconstruire la structure.

Conclusion [3]

Nous avons motivé et décrit la tâche de restructuration. Un cadre stochastique général, reposant sur la modélisation d'un processus de génération des documents Web a été proposé. L'utilisation d'une structure abstraite permet de convertir des données hétérogènes vers un schéma de médiation. Plusieurs expériences utilisant une méthode de reconstruction exacte et une méthode approchée ont été menées sur des corpus réels. À notre connaissance, cette méthode est la première à pouvoir convertir des corpus de grande taille. Les résultats de ces premières expériences sont encourageants et montrent clairement que l'information de structure contenue dans les pages HTML est suffisante pour inférer une structure sémantiquement riche.

5. Bibliographie

[BER 96] BERGER A., DELLA PIETRA S., DELLA PIETRA V., « A maximum entropy approach to natural language processing », *Computational Linguistics*, 1996.

[BEX 02] BEX G. J., MANETH S., NEVEN F., « A formal model for an expressive fragment of XSLT », *Inf. Syst.*, vol. 27, n° 1, 2002, p. 21–39, Elsevier Science Ltd.

[BUY 00] BUYUKKOKTEN O., GARCIA-MOLINA H., PAEPCKE A., « Seeing the Whole in Parts : Text Summarization for Web Browsing on Handheld Devices », rapport, 2000.

[CAL 94] CALLAN J. P., « Passage-Level Evidence in Document Retrieval », *SIGIR'94*, New York, NY, USA, 1994.

[CHI 05] CHIDLOVSKII B., FUSELIER J., « A Probabilistic Learning Method for XML Annotation of Documents », *IJCAI'05*, 2005.

[Dau 05] DAUMÉ III H., MARCU D., « Learning as Search Optimization : Approximate Large Margin Methods for Structured Prediction », *ICML'05*, 2005.

[DOA 03] DOAN A., DOMINGOS P., HALEVY A., « Learning to Match the Schemas of Data Sources : A Multistrategy Approach », *Maching Learning*, , n° 3, 2003.

[DOA 05] DOAN A., HALEVY A. Y., « Semantic Integration Research in the Database Community : A Brief Survey », *AI Magazine, Special Issue on Semantic Integration*, , 2005.

[FIN 98] FINE S., SINGER Y., TISHBY N., « The Hierarchical Hidden Markov Model : Analysis and Applications », *Machine Learning*, vol. 32, n° 1, 1998, p. 41-62.

[FUH 02] FUHR N., GOVERT N., KAZAI G., LALMAS M., « INEX : Initiative for the Evaluation of XML Retrieval », *SIGIR'02 Workshop on XML and Information Retrieval*, 2002.

[TSO 04] TSOCHANTARIDIS I., HOFMANN T., JOACHIMS T., ALTUN Y., « Support vector machine learning for interdependent and structured output spaces », *ICML'04*, 2004.

[WIS 05] WISNIEWSKI G., DENOYER L., GALLINARI P., « Restructuration automatique de documents dans les corpus semi structurés hétérogènes », *EGC'2005*, 2005.

[YOU 00] YOUNG-LAI M., TOMPA F. W., « Stochastic Grammatical Inference of Text Database Structure », *Mach. Learn.*, vol. 40, n° 2, 2000, p. 111–137.

3. This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.