Modèle probabiliste pour l'extraction de structures dans les documents web

Guillaume Wisniewski — Francis Maes — Ludovic Denoyer 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.*

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 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, apprentissage, extraction de structure*

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

DOI :10.3166/DN.10.89-107

DN – 10/2007. Recherche d'information, pages 89 à 107

1. 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* (figure 1) ou par ordre chronologique.

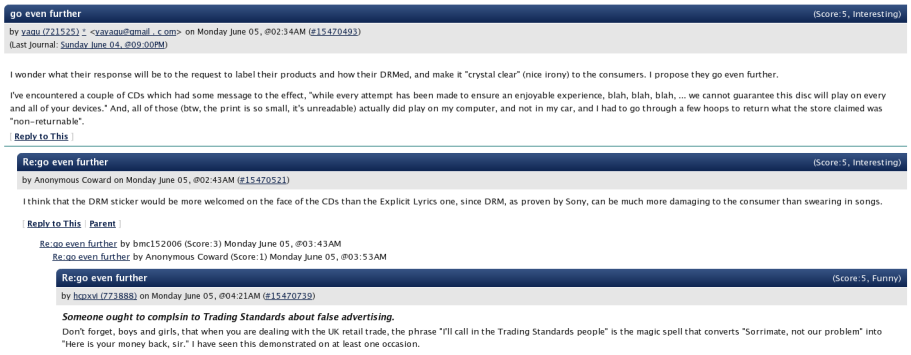


Figure 1. *Extrait d'un thread de commentaires sur Slashdot. La mise en page nous permet d'identifier facilement chaque commentaire ainsi que les relations entre commentaires. Des méta-informations, comme le nom de l'auteur ou la date du commentaire, sont également immédiatement accessibles*

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 (Baluja, 2006, Buyukkokten *et al.,* 2001), 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, au delà de l'identification des documents pertinents, il est devenu important pour un système de RI de retrouver les éléments pertinents à l'intérieur du

1. `slashdot.org`

document. 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 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. Nous parlerons par la suite de *structure pragmatique*pour définir ce concept. Intuitivement, cette structure rassemble les éléments sémantiques communs nécessaires à une classe d'application. La figure 2 décrit un exemple des différentes définitions de la structure d'un document web.

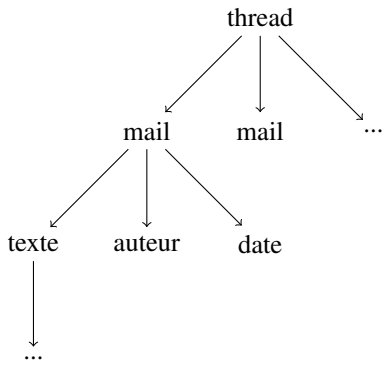
Jusqu'à présent, toutes les approches utilisant à la fois l'information de contenu et l'information de structure, que ce soit en classification (Denoyer *et al.,* 2004) ou en recherche d'information (Fuhr *et al.,* 2002, Wilkinson, 1994), n'ont pris en considération que la structure syntaxique des documents, 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 au caractère implicite 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 (Callan, 1994). Nous pensons que l'utilisation d'une structure intermédiaire de plus haut niveau, permettra de s'affranchir de ces deux problèmes.

Marc Perrin	23 oct 2005	Re: Nouvel an
Benedicte Moulin	24 oct 2005	Re: Nouvel an
Benedicte Moulin	24 oct 2005	Re: Nouvel an
Bertrand Paquet	23 oct 2005	Re: Nouvel an

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

```
<html>
  <p> Bonne [...]      <a>Bert...</a>15:06 [...] </p>
  <dd>
    <dl>Bonne [...]      <a>Jul</a> 19:46 [...]</dl>
  </dd>
</html>
```

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



(c) Structure pragmatique

Figure 2. *Différentes définitions de la structure d'un document web décrivant un échange de mails*

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 améliorer les performances de l'utilisation de celle-ci. L'écriture manuelle de médiateurs spécifiques à chacune des sources de documents est un travail long, coûteux, qui présente un grand risque d'erreur (Zhang *et al.,* 2006) et 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 défini a priori. Même si nous considérons ici plus spécifiquement la tâche de transformation de documents HTML en documents XML, l'approche étudiée permet de traiter aussi les corpus composés de documents XML qui suivent différents schéma ou même de documents dont le schéma n'est pas connu.

Le plan du document est le suivant. Nous proposons un cadre général permettant l'apprentissage de transformations de documents (Section 2) et détaillons l'application de celui-ci à l'extraction de structure pragmatique en décrivant la transformation de documents HTML en documents XML (Section 3). Finalement nous présentons plusieurs séries d'expériences sur des corpus XML et HTML réels.

2. Modèle de restructuration

2.1. *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 (Abiteboul, 1997).

Nous adoptons la représentation traditionnelle des documents semi-structurés sous forme d'un arbre ordonné : un document d, tel celui de la figure 3, 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.

Les documents semi-structurés peuvent être associés à un *schéma* (DTD, XML Schema ...) qui définit un ensemble de règles et de contraintes sur les étiquettes des nœuds. Dans notre travail, nous supposerons que le schéma des documents de sortie est connu, mais pas celui des documents d'entrée.

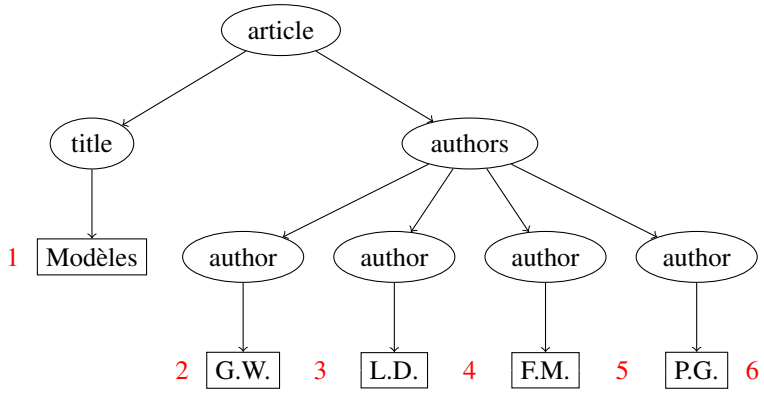


Figure 3. *Exemple de document semi-structuré. Les nœuds internes sont décrits par des cercles et les nœuds de contenu par des rectangles*

2.2. *Approche générale*

Étant donné un schéma de sortie arbitraire, nous souhaitons transformer 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 tâche de restructuration revient à identifier dans un document les éléments pertinents et, à déterminer récursivement les relations entre ces éléments. Sur l'exemple de la figure 4, l'objectif est ainsi d'identifier les noms d'acteurs et les noms de personnages, puis de déterminer le rôle de chaque acteur. Les éléments et les relations à extraire sont, tous deux, définis par un schéma cible.

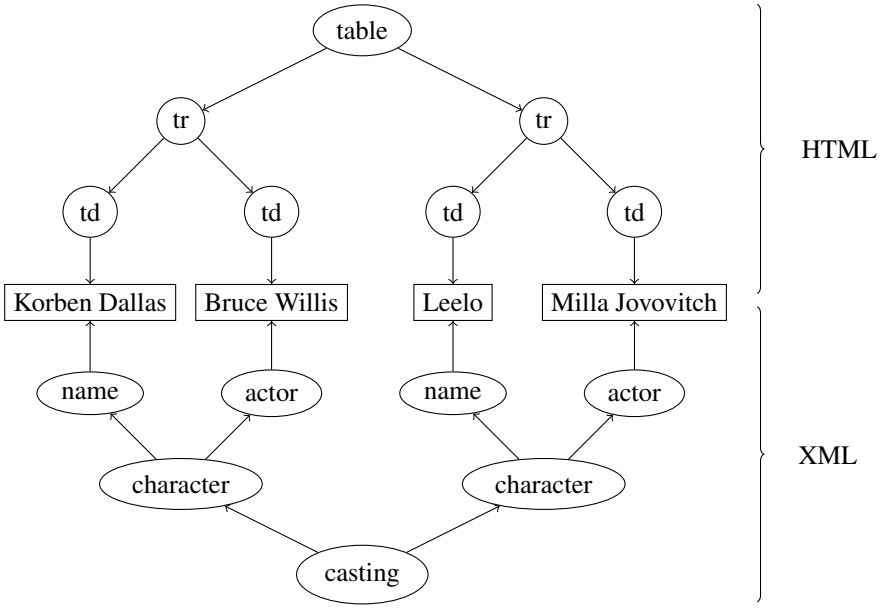


Figure 4. *Exemple d'un matching HTML-XML simple. Le document d'entrée est au format HTML et doit être converti dans le format de sortie médiateur XML. L'objectif est ici d'identifier les noms d'acteurs et les noms de personnages, puis de déterminer le rôle joué par chaque acteur*

Automatiser cette transformation revient à apprendre une fonction f, tel que $f(d^{in}) = d^{out}$. La restructuration de documents est un problème généralement sous-déterminé, puisque plusieurs documents de sortie peuvent être *compatibles* avec le document d'entrée. Ainsi, dans l'exemple du casting, n'importe quel acteur peut, a priori, jouer n'importe quel rôle. Pour déterminer la restructuration correspondant à d^{in}, nous définissons une fonction de coût paramétrée par θ, $\phi(d, d^{in}; \theta)$ permettant d'évaluer la qualité d'une solution candidate d. Cette fonction nous permet d'ordonner les éléments de $\mathcal{D}(d^{in})$, l'ensemble des *restructurations potentielles* constitué par tous les documents d respectant le schéma cible et contenant les informations de d^{in}.

La tâche de restructuration correspond alors à la recherche de la restructuration potentielle la meilleure :

$$d^{out} = \underset{d \in \mathcal{D}(d^{in})}{\operatorname{argmin}} \phi(d, d^{in}; \theta) \quad [1]$$

Le vecteur de paramètres θ est estimé à partir d'un corpus d'apprentissage constitué de documents convertis manuellement dans le schéma cible.

Dans l'Équation [1], l'argmin traduit le parcours de l'espace de toutes les restructurations potentielles $\mathcal{D}(d^{in})$ pour rechercher la meilleure solution. 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* (Tsochantaridis *et al.,* 2005) qui fournit un cadre général pour la prédiction d'éléments structurés. Il permet par exemple de traiter des problématiques de reconnaissance d'écriture (mise en correspondance de séquences) ou d'analyse syntaxique (l'objectif est alors d'associer un arbre à une séquence)

Étant donné leur complexité, la plupart des méthodes développées dans ce cadre ne sont pas directement applicables à des corpus de grande taille comme ceux habituellement utilisés en RI. Nous proposons ici un cadre stochastique à la tâche de restructuration basé sur un modèle génératif de documents.

2.3. *Modèle probabiliste de restructuration*

2.3.1. *Processus de génération des documents w@<eb*

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 pour chaque document une représentation *abstraite*, que nous noterons h. Cette représentation permet de générer toutes les structures syntaxiques possibles d'un document, chaque structure étant spécifiée par 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. La figure 5 illustre ce processus.

Formellement, ce processus est modélisé par le réseau bayésien de la figure 6 : 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 document considéré, p^{in} est un modèle de document 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 3 comment spécifier les valeurs de ces différentes variables pour la transformation de documents HTML en XML.

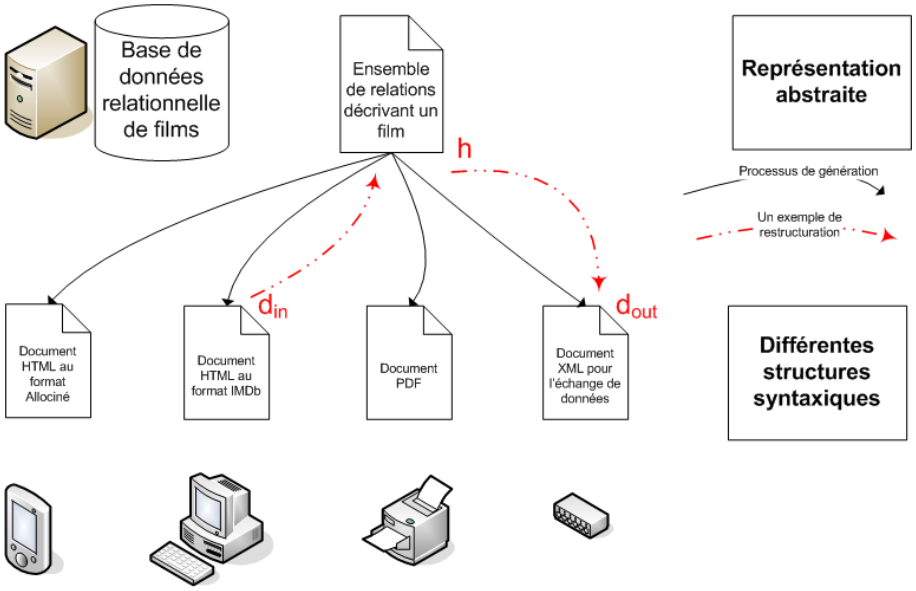


Figure 5. *Exemple de processus de génération de documents : à partir de la* représentation abstraite *on peut générer différents documents en utilisant différents modèles de document. De manière générale, seuls ces documents sont accessibles et la représentation abstraite est* cachée. *La restructuration consiste à utiliser l'information présente dans une structure syntaxique donnée pour générer un document suivant une autre structure*

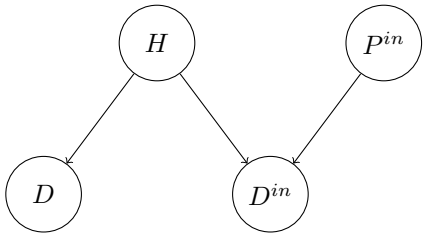


Figure 6. *Le réseau bayésien modélisant le processus de génération des différentes représentations d'un document à partir de l'observation d'un modèle de document d'entrée p^{in} et d'une source d'information h. Notre objectif est d'inférer d à partir de la connaissance de d^{in} et p^{in}*

2.3.2. *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 :

$$\phi(d, d^{in}; \theta) = 1 - p(d|d^{in}, p^{in}; \theta)$$

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 d^{out} exprimé dans le schéma de médiation qui a été généré à partir du même document abstrait h. L'Équation [1] se réécrit alors :

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

À partir du réseau bayésien décrit figure 6, nous pouvons estimer $p(d|d^{in}, p^{in}; \theta)$:

$$\begin{aligned} p(d|d^{in}, p^{in}; \theta) &= \frac{p(d, d^{in}, p^{in}; \theta)}{p(d^{in}, p^{in}; \theta} \\ &= \frac{\sum_h p(h; \theta) \cdot p(p^{in}; \theta) \cdot p(d^{in}|h, p^{in}; \theta) \cdot p(d|h; \theta)}{p(d^{in}, p^{in}; \theta)} \end{aligned}$$

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

$$d^{out} = \underset{d \in \mathcal{D}}{\operatorname{argmax}} \sum_h \underbrace{p(d|h; \theta)}_{extraction} \cdot \underbrace{p(h|p^{in}, d^{in}; \theta)}_{génération}$$

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 :

$$d^{out} = \underset{d \in \mathcal{D}(d^{in})}{\operatorname{argmax}} \sum_h p(h|d^{in}, p^{in}; \theta) \cdot p(d|h; \theta) \qquad [2]$$

3. 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.

3.1. *Modèle de représentation abstraite*

La résolution pratique de l'Équation [2] nécessite de disposer d'informations supplémentaires sur la représentation abstraite h. Nous supposerons ici 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, de nombreuses approches de restructuration ((Chidlovskii *et al.,* 2005) par exemple) font une telle hypothèse de séquentialité afin de limiter la taille de l'espace de recherche. Cette 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 définit une suite de nœuds de contenu c^h (cf : figure 3) et que :

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

L'Équation [2] peut alors se réécrire :

$$\begin{aligned} d^{out} &= \underset{d \in \mathcal{D}(d^{in})}{\operatorname{argmax}} \sum_h p(h|d^{in}, p^{in}; \theta) \cdot p(d|h; \theta) \\ &\simeq \underset{t \in \mathcal{T}(c^{in})}{\operatorname{argmax}} p(t|c^{in}; \theta) \end{aligned}$$

où $\mathcal{T}(c^{in})$ est l'ensemble des arbres respectant le schéma de médiation et dont la séquence des feuilles est c^{in}. En effet, l'hypothèse de séquentialité nous permet de limiter la recherche aux documents ayant la même séquence de feuilles : dans ce cadre, la tâche de reconstruction se définit donc comme la construction d'un arbre à partir d'une séquence d'éléments.

3.2. *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 $d : p(d) = p(n_1, ..., n_{\#d}; \theta)$. 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).

Pour cela, 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 :

$$p(d;\theta) = \prod_{i=1}^{\#d} p\left(tag(n_i)|pere(n_i), frere(n_i), c_i; \theta\right)$$

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.

3.3. *Apprentissage*

Les probabilités $P(tag(n_i)|pere(n_i), frere(n_i), c_i; \theta)$ sont estimées par un classifieur maximisant l'entropie (Chen *et al.,* 2000). Ce type de classifieur nous permet de prendre en compte facilement toutes les caractéristiques que nous jugeons pertinentes, sans nécessiter d'hypothèses d'indépendance entre celles-ci. Nous pouvons donc définir aussi bien des caractéristiques décrivant le contenu des nœuds (nombre de majuscules, présence de chiffre ...) et leur contexte (les relations père-fils).

Plus précisément, on a :

$$P(tag(n_i)|pere(n_i), frere(n_i), c_i; \theta) = \frac{\exp\left(\langle\theta, F\left(pere(n_i), frere(n_i), c_i\right)\rangle\right)}{Z_\theta\left(pere(n_i), frere(n_i), c_i\right) \cdot p(c_i)}$$

où Z_θ est un facteur de normalisation, F le vecteur de caractéristiques et θ le vecteur de paramètres. $\langle\cdot,\cdot\rangle$ dénote le produit scalaire usuel. $p(c_i)$ est la probabilité du contenu du nœud i qui ne dépend pas du modèle de document et n'intervient pas dans le calcul de la meilleure restructuration.

Le vecteur des paramètres θ a été estimé en utilisant le principe de maximisation de l'entropie : l'algorithme d'apprentissage détermine, parmi toutes les distributions compatibles avec les observations, celle qui fait le moins d'hypothèse sur les valeurs non observées.

3.4. *Reconstruction d'un document*

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

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

Nous avons utilisé deux méthodes pour résoudre cette équation. La première repose sur une technique de programmation dynamique proche des algorithmes utilisés dans les analyseurs syntaxiques (Jurafsky *et al.,* 2000).

– **Intuition :** la construction du document de sortie passe par une décomposition récursive de celui-ci en sous-arbres. On reconstruit d'abord l'ensemble des sous-arbres optimaux, puis on cherche, de proche en proche, la meilleure combinaison de ces solutions partielles. Afin de garantir l'optimalité de la solution, à chaque étape, l'algorithme considère de manière exhaustive toutes les combinaisons de tous les sous-arbres possibles.

– **Complexité :** la complexité de la méthode, dans le pire 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.

– **Avantages/Inconvénients :** la méthode de programmation dynamique permet d'obtenir une solution exacte de l'Équation [3]. La construction d'une solution exacte a cependant un coût : la complexité est cubique en fonction du nombre de feuilles du document d'entrée. Cela rend, en pratique, la méthode de programmation dynamique inutilisable pour la reconstruction de grands documents.

Afin de reconstruire des documents de grande taille nous avons utilisé l'algorithme LaSO (Daumé III *et al.,* 2005) qui permet de trouver une solution approchée de l'Équation [3].

– **Intuition :** la complexité des approches fondées sur la programmation dynamique est directement liée à l'énumération exhaustive des combinaisons de solutions partielles. Pour limiter cette complexité, LaSO propose de réduire les combinaisons à considérer à l'aide d'une *heuristique*. À chaque étape l'heuristique donne des scores aux différentes combinaisons envisageables et seule la meilleure d'entre elles est choisie. En pratique, chaque étape de résolution correspond à l'ajout d'un nœud dans le document de sortie. LaSO propose une méthode générale pour apprendre l'heuristique utilisée à l'aide d'algorithmes d'apprentissage par renforcement (Sutton *et al.,* 1998).

– **Complexité :** la complexité de la reconstruction est en $\mathcal{O}(n \cdot V \cdot N)$. La reconstruction d'un document est donc réalisée en un temps linéaire par rapport à N, le nombre de nœuds du document reconstruit. Cette complexité est inférieure à celle de la méthode de programmation dynamique.

– **Avantages/Inconvénients :** le principal avantage de LaSO est sa faible complexité. Néanmoins, LaSO ne permet que de reconstruire une solution approchée et, expérimentalement, on observe une faible baisse de performances. De plus, comme le montrent nos expériences, le temps d'apprentissage de l'heuristique n'est pas négligeable, même si le temps de reconstruction est lui très raisonnable.

4. Expériences

4.1. *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 2004 (Fuhr *et al.,* 2005) 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 de description de films 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 IMDb. Cette méthode de reconstruction n'est pas applicable au troisième corpus.

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 feuilles couvertes par l'étiquette e. Par exemple, sur la figure 3, 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.

4.2. *Résultats*

Le Tableau 1 et la figure 7 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 IMDb

2. `www.imdb.com`

et Shakespeare et près de 70% des nœuds d'Inex, qui est un corpus ayant un plus grand nombre d'étiquettes possibles et présentant 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 apparaît que le score obtenu sur ces derniers est plus faible que le 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, même si les résultats sur des bases assez régulières comme IMDb sont plutôt bons. Sur l'ensemble des expériences, 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 7 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.

5. État de l'art

Des problèmes semblables (*schema matching*, intégration de données ...) sont traités depuis plusieurs années par la communauté base de données et sont apparus, plus récemment, pour la recherche documentaire, la conversion (Chidlovskii *et al.,* 2005) et l'intégration de documents (Chung *et al.,* 2002), l'alignement d'ontologies. Plusieurs techniques d'apprentissage ont été employées : classifieur multi-classes, spectre de graphe ... Plusieurs travaux récents abordent la problématique de la transformation de documents, notamment avec des grammaires formelles (Chidlovskii *et al.,* 2005, Wisniewski *et al.,* 2006).

Une comparaison des différentes approches proposées en base de données est faite dans (Doan *et al.,* 2005). (Doan *et al.,* 2003) présente une des approches les plus abouties pour travailler sur différents types de données (SQL, XML, ontologies). La tâche 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 présentant peu de récursion et ne contenant que très rarement des données textuelles.

La tâche de restructuration est aussi proche de l'extraction d'information (McCallum, 2005) dont le but est de repérer certains éléments et les relations entre ceux-ci dans du texte brut et plus récément dans des pages web. Des travaux récents ((Viola *et al.,* 2005))dans ce domaine ont mis en évidence l'importance de traiter simultané-

corpus	méthode	feuilles	nœuds internes	arbre complet	durée apprentissage	durée reconstruction
INEX	DP	**79.6**%	51.5%	**70.5**%	30 mn	$\simeq$ 4 jours
	LaSO	75.8%	**53.1**%	67.5%	$>$ 1 semaine	**3h20min**
IMDb	DP	**95.3**%	77.1%	**90.4**%	20 mn	$\simeq$ 2 jours
	LaSO	90.5%	**86.8**%	89.6%	$>$ 1 semaine	**1h15min**
Shakespeare	DP	—	—	—	—	—
	LaSO	95.3%	77.0%	92.2%	$\simeq$ 5 jours	**30 min**

Tableau 1. *Résultats des expériences de restructuration : mesure de l'erreur de reconstruction. Les tirets indiquent que l'expérience ne peut être réalisée. Une estimation de la durée d'apprentissage et de reconstruction est donnée (Les expériences ont été menées sur des Pentium 3,2 GHz). LaSO permet d'avoir des temps d'inférence plus rapide, mais ceci au prix d'une baisse de performance et d'un temps d'apprentissage très long*

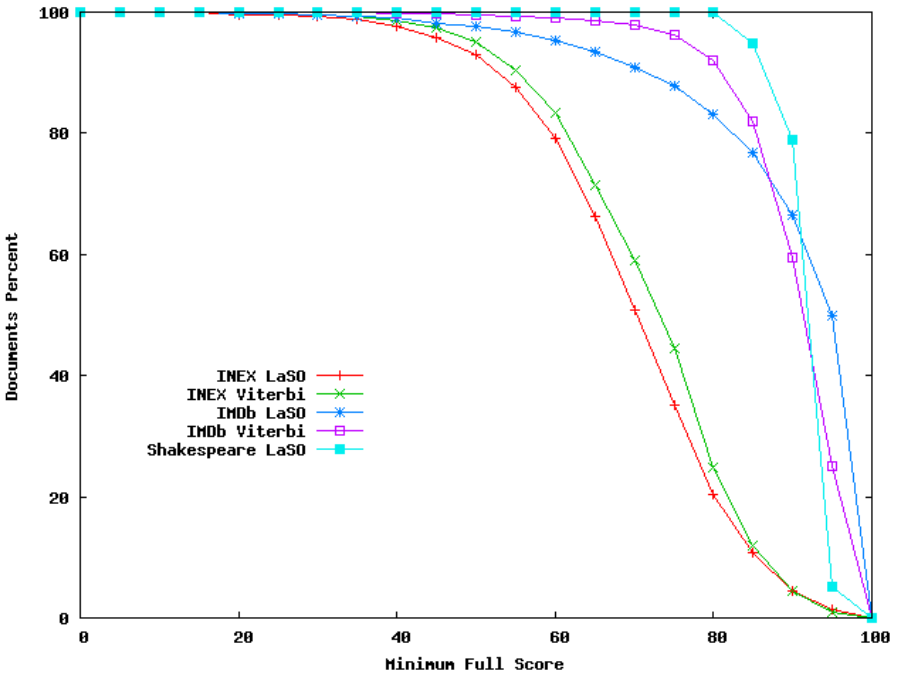


Figure 7. *Résultat des expériences : mesure du pourcentage de documents (ordonnée) reconstruit correctement à x% (abscisse). Par exemple, sur les données Mini-Shakespeare, LaSO reconstruit environ 75 % des documents avec une correction de 80 %*

ment l'extraction des éléments et des relations entre ces éléments ce qui est une des principales caractéristiques du modèle présenté dans ce travail.

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 utilisées dans les HMMs hiérarchiques ((Fine *et al.,* 1998)). De nombreux travaux ((Young-Lai *et al.,* 2000), (Chidlovskii *et al.,* 2005)) 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 (Wisniewski *et al.,* 2005) 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 notre est (Chidlovskii *et al.,* 2005) : ils caractérisent le contenu par un classifieur maximisant le critère d'entropie et utilisent des grammaires probabilisées pour reconstruire la structure.

À côté de la transformation de documents, un deuxième type d'approche a été développé pour tirer avantage de la structure des données web. Il consiste à adapter

les langages de requêtes utilisés pour prendre en compte les spécificités de la structure syntaxique (Amer-Yahia *et al.,* 2002). Cette adaptation repose, en général, sur une *relaxation* des méthodes développées pour les bases de données (langage de requêtes, représentation des données ...). Mais, étant donné la nature implicite de leur structure et leur hétérogénéité, ces techniques sont peu adaptées aux documents web.

6. Conclusion

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. Ce cadre permet de distinguer deux étapes dans la transformation d'un document semi-structuré : une première étape *extrait* du document d'entrée un ensemble d'informations pertinentes et une deuxième étape utilise ces informations pour générer un nouveau document conforme au schéma souhaité. 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.

Plusieurs améliorations sont envisageables. Une première amélioration consiste à introduire dans la première phase un modèle permettant d'extraire des informations plus riches qui pourraient ensuite être utilisées dans la phase de génération. Plusieurs approches d'extraction d'information (Champs Conditionnels Aléatoires (Lafferty *et al.,* 2001), grammaires probabilistes (Viola *et al.,* 2005) ...) pourraient être utilisées. L'amélioration des performances de prédiction, notamment sur les nœuds internes, nécessitent l'utilisation de caractéristiques plus riches (dépendances non-locales, prise en compte de la structure d'entrée ...). Ces dépendances sont difficiles à introduire dans un modèle génératif comme le notre. Une solution serait de formuler l'extraction de structure dans un cadre discriminatif.

Remerciements

Ce travail a été en parti financé par le Programme IST de la Communauté européenne, dans le cadre du réseau d'excellence Pascal (IST-2002-506778). Cette publication ne reflète que le point de vue des auteurs.

7. Bibliographie

Abiteboul S., « Querying Semi-Structured Data », *ICDT*, p. 1-18, 1997.

Amer-Yahia S., Cho S., Srivastava D., « Tree Pattern Relaxation », *EDBT '02 : Proceedings of the 8th International Conference on Extending Database Technology*, Springer-Verlag, London, UK, p. 496-513, 2002.

Baluja S., « Browsing on small screens : recasting web-page segmentation into an efficient machine learning framework », *n Proceedings of the 15th International Conference on World Wide Web*, ACM Press, p. 33-42, May, 2006.

Buyukkokten O., Garcia-Molina H., Paepcke A., « Seeing the Whole in Parts : Text Summarization for Web Browsing on Handheld Devices », 2001.

Callan J. P., « Passage-level evidence in document retrieval », *SIGIR '94 : Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, Springer-Verlag New York, Inc., New York, NY, USA, p. 302-310, 1994.

Chen S., Rosenfeld R., « A survey of smoothing techniques for ME models », *IEEE Trans. on Speech and Audio Processing*, 2000.

Chidlovskii B., Fuselier J., « A Probabilistic Learning Method for XML Annotation of Documents », *IJCAI*, 2005.

Chung C. Y., Gertz M., Sundaresan N., « Reverse Engineering for Web Data : From Visual to Semantic Structures », *ICDE*, 2002.

Daumé III H., Marcu D., « Learning as search optimization : approximate large margin methods for structured prediction », *ICML '05 : Proceedings of the 22nd international conference on Machine learning*, ACM Press, New York, NY, USA, p. 169-176, 2005.

Denoyer L., Gallinari P., « Bayesian Network Model for Semi-Structured Document Classification », *Information Processing and Management*, 2004.

Doan A., Domingos P., Halevy A., « Learning to Match the Schemas of Data Sources : A Multistrategy Approach », *Mach. Learn.*, vol. 50, n° 3, p. 279-301, 2003.

Doan A., Halevy. A., « Semantic Integration Research in the Database Community : A Brief Survey », *AI Magazine, Special Issue on Semantic Integration*, 2005.

Fine S., Singer Y., Tishby N., « The Hierarchical Hidden Markov Model : Analysis and Applications », *Machine Learning*, vol. 32, n° 1, p. 41-62, 1998.

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.

Fuhr N., Lalmas M., Malik S., Szlavik Z., Fuhr N., Lalmas M., Malik S., Szlavik Z., « Advances in XML Information Retrieval : Third International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX) », vol. 3493, Springer-Verlag GmbH, may, 2005. http ://www.springeronline.com/3-540-26166-4.

Jurafsky D., Martin J. H., *Speech and Language Processing : An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, 2000.

Lafferty J., McCallum A., Pereira F., « Conditional Random Fields : Probabilistic Models for Segmenting and Labeling Sequence Data », *Proc. 18th International Conf. on Machine Learning*, Morgan Kaufmann, San Francisco, CA, p. 282-289, 2001.

McCallum A., « Information extraction : distilling structured data from unstructured text », *Queue*, vol. 3, n° 9, p. 48-57, 2005.

Sutton R., Barto A., *Reinforcement l earning : an introduction*, MIT Press, 1998.

Tsochantaridis I., Joachims T., Hofmann T., Altun Y., « Large Margin Methods for Structured and Interdependent Output Variables », *Journal of Machine Learning Research*, vol. 6, p. 1453-1484, 2005.

Viola P., Narasimhan M., « Learning to extract information from semi-structured text using a discriminative context free grammar », *SIGIR '05*, 2005.

Wilkinson R., « Effective retrieval of structured documents », *SIGIR '94 : Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, Springer-Verlag New York, Inc., New York, NY, USA, p. 311-317, 1994.

Wisniewski G., Denoyer L., Gallinari P., « Restructuration automatique de documents dans les corpus semi structurés hétérogènes », *EGC'2005*, 2005.

Wisniewski G., Denoyer L., Maes F., Gallinari P., « Modèle probabiliste pour l'extraction de structures dans les documents semi-structurés — Application aux documents Web », *CORIA'06*, Lyon, p. 169-180, March, 2006.

Young-Lai M., Tompa F. W., « Stochastic Grammatical Inference of Text Database Structure », *Machine Learning*, 2000.

Zhang S., Dyreson C., « Polymorphic XML restructuring », *IIWeb'06 : Workshop on Information Integration on the Web*, 2006.