Méthode d'apprentissage pour la conversion automatique de documents structurés

Jérôme Fuselier, Boris Chidlovskii, Jean-Charles Marty

Méthode d'apprentissage pour la conversion automatique de documents structurés

Jérôme Fuselier[1,2] **, Boris Chidlovskii**[1]**, Jean-Charles Marty**[2]
[1] Xerox Research Centre Europe,
6, chemin de Maupertuis, 38240 Meylan, France
[2] Université de Savoie - Laboratoire SysCom,
Domaine Universitaire, 73376 Le Bourget-du-Lac, France
jerome.fuselier@xrce.xerox.com, boris.chidlovskii@xrce.xerox.com,
jean-charles.marty@univ-savoie.fr

Résumé : De plus en plus de sociétés désirent moderniser leur système de gestion de fond documentaire. Le problème qui se pose à ces entreprises concerne la migration de leurs documents vers un format plus ouvert et offrant plus de possibilités. La tâche de conversion est rendue difficile d'une part à cause de la grande hétérogénéité des documents et d'autre part à cause du très grand volume de données à traiter.

Dans ce papier nous présentons une approche permettant de réaliser des conversions de documents orientés présentation vers des documents sémantiques. L'intérêt de la méthode consiste à automatiser le processus de conversion en utilisant des techniques d'apprentissage supervisé pour découvrir des règles de transformations applicables pour une collection de documents. Nous décomposons la conversion en deux étapes pour simplifier le problème, une première étape d'annotation sémantique des éléments textuels du document d'origine et une seconde étape qui consiste à faire émerger la structure sémantique du document en respectant une grammaire cible qui décrit le langage à utiliser pour les documents finaux. L'approche est probabiliste pour permettre une plus grande robustesse.

Mots-clés : Apprentissage, Extraction d'informations, Grammaires probabilistes.

1 INTRODUCTION

Les technologies liées au formalisme xml[1] sont à présent suffisamment matures et utilisées pour permettre leur utilisation dans le monde industriel. Elles offrent de nouvelles possibilités dans le domaine de la gestion documentaire, de la publication ou du multimédia. La simplicité du formalisme permet de définir des langages adaptés et facilite l'échange de données et la réutilisation du contenu. Les technologies construitent autour du formalisme offrent de nouvelles fonctionnalités, les recherches peuvent par exemple devenir plus significatives grâce à un balisage sémantique très précis sur les parties importantes du document. Il est également possible d'intégrer des données en provenance de sources diverses et la notion de document est redéfinie, un document est ainsi vu comme une aggrégation d'informations sémantisées et non plus comme un document monolithique. Ainsi modularisés, les mises à jour de ces documents sont facilitées. Le formalisme peut également permettre de définir des vues multiples sur les données et d'adapter le contenu à l'utilisateur [Fuselier et al., 2002]. L'avantage pour les entreprises est donc important mais le volume de documents à migrer vers ce nouveau formalisme crée de nombreux problèmes pour la conversion de fonds documentaires [Chidlovskii et al., 2004].

Les fonds documentaires des entreprises sont constitués d'un ensemble de documents très variés comme par exemple des documentations techniques, des manuels utilisateurs, des rapports internes, des publications, des factures, *etc.*. Ces documents sont souvent disponibles en formats électroniques, dans des formats privilégiant la présentation comme xhtml, pdf ou word. Ils décrivent correctement comment le document doit être présenté mais ne décrivent pas du tout ce qui compose effectivement le document et comment il est organisé. A l'opposé, en utilisant l'extensibilité du langage xml, il est possible d'annoter sémantiquement le contenu des documents (titres, auteurs, références, *etc.*), en laissant la tâche de présentation des documents à des composants spécialisés, qui pourront également adapter l'affichage en fonction des périphériques utilisés pour accéder aux informations.

Le processus de conversion nécessite généralement la définition d'un modèle de document cible exprimé par une grammaire xml. Cette grammaire peut être représentée sous la forme d'un schéma xml[2], d'une dtd[3] ou d'un schéma RelaxNG[4]. Elle définit les éléments structurels et sémantiques des documents propres à l'entreprise et à l'utilisation souhaitée. Un document respectant ces contraintes grammaticales peut être vu comme une instance du langage défini par la grammaire. La

[1] Voir le site du World Wide Web Consortium, http ://www.w3c.org

[2] XML Schema. http ://www.w3.org/XML/Schema

[3] Document Type Definition. http ://www.w3.org/TR/REC-xml/#elemdecls

[4] Relax NG. http ://www.relaxng.org

conversion de documents ne se résume pas seulement à une transformation d'un format à un autre mais également à la définition explicite d'informations qui se trouvaient cachées dans les documents d'entrées. En général, la conversion de fonds documentaires vers le formalisme xml est référencée comme une transformation de contenu orienté présentation vers du contenu orienté sémantique.
Pour uniformiser notre approche, nous utilisons un format pivot standard qui préserve les informations de présentation et de structure, le xhtml. Cela nous permet de ne pas travailler avec les formats propriétaires du fond documentaire (pdf, word, *etc.*) pour travailler avec un format ouvert. Pour les formats majoritairement utilisés dans l'industrie, il existe des convertisseurs qui permettent d'effectuer cette première transformation comme par exemple X-Ice[5] ou PdfToHtml[6]. Ils sont capables de reconnaître des entités structurelles comme les tableaux, les listes ou les paragraphes mais le résultat de leur conversion reste cependant insuffisant du point de vue de la grammaire cible que l'on cherche à atteindre.
La figure 1 présente les trois vues possibles d'un document (présentation, logique et sémantique). Notre approche consiste tout d'abord à transformer le fond documentaire vers le niveau présentation, ou éventuellement logique pour certaines parties du document puis à effectuer la transformation du document représenté dans le format pivot vers un document sémantique qui respecte la grammaire cible fournie par l'entreprise et les experts du domaine. Il est intéressant de noter que dans certains cas, les grammaires des documents finaux peuvent contenir des parties logiques. Dans ce cas précis, le niveau logique correspond à ce que l'utilisateur désire. Nous pouvons par exemple citer la grammaire DocBook [DocBook] qui permet de représenter des rapports techniques.

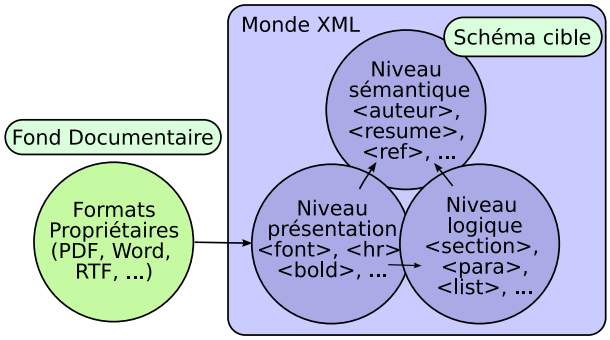

FIG. 1 – Les trois niveaux d'un document structuré.

Dans ce papier, nous suivons les concepts d'apprentissage supervisé appliqués à la conversion documentaire. Nous supposons qu'il existe une collection d'apprentissage qui fournit des exemples de transformations, composés des documents sources et de leurs annotations en xml. Cela suppose la présence d'un expert qui est capable d'extraire manuellement les informations sémantiques des documents d'origine et de les porter vers un document structuré selon les contraintes imposées par la grammaire cible. Chaque paire de l'ensemble d'apprentissage (document source, document cible) est utilisé pour mettre au point un modèle de transformation reproductible qui pourra être appliqué par la suite sur l'ensemble de la collection à convertir. Un des buts visé est de réduire le travail de l'expert et d'automatiser la conversion en utilisant un nombre minimum d'exemples. De plus, nous travaillons avec des méthodes probabilistes pour améliorer la robustesse et les performances des modèles. Cette approche nous permet par exemple de gérer les incohérences de la collection qui peuvent être introduites par différents auteurs.
Le reste de ce papier présente l'approche que nous avons mis au point pour réaliser cette conversion. La section 2 présente la décomposition du problème en deux sous-problèmes plus simples et notre apport dans la conversion de fonds documentaires. La section 3 décrit le processus d'évaluation de la conversion ainsi que les résultats des expériences que nous avons menés. La section 4 présente les approches alternatives et la section 5 conclut le papier.

2 NOTRE APPROCHE

2.1 Le problème

Le problème que nous cherchons à résoudre est l'annotation de documents sémantiques guidée par un schéma cible. Nous disposons en entrée d'un schéma cible fourni par un expert du domaine qui décrit la sémantique et la structure des documents, d'une collection de documents qui sont destinés à être visualisés et d'un sous-ensemble annoté de cette collection pour pouvoir apprendre un schéma de conversion reproductible. Ce que nous cherchons en sortie est un modèle de conversion qui pourra être appliqué à l'ensemble des documents orientés présentation pour créer des documents sémantiques appartenant au langage définit par la grammaire fournie.
La figure 2 présente un exemple de conversion avec en haut le fichier d'origine et en dessous le document sémantique qu'il faut obtenir avec la grammaire fournie. Le document est un fragment du CV d'un étudiant qui définit ses domaines de compétences et les études qu'il a suivies. Le contenu du document est présenté de manière à faciliter la compréhension des informations par des humains, il respecte une certaine nomenclature définie par un modèle de CV fourni par microsoft office. Le document cible est un fragment xml qui ne possède que les informations sémantiques du CV et qui respecte la grammaire. Toutes les informations de visualisation ont disparu pour ne conserver que les informations de contenu. Nous pouvons remarquer que certains fragments de contenu ne servent qu'à améliorer la visibilité des informations, ils ne conservent aucune information sémantique et devront être supprimés lors de la conversion.
La figure 3 présente le même exemple sous un autre point de vue avec la représentation interne des documents sous forme arborescente. L'arbre en haut est constitué de balises de présentation (b, span, *etc.*) qui vont être interprétées par un navigateur pour afficher le

[5]http ://www.turnkey.com.au/tksweb/products/xice.html
[6]http ://pdftohtml.sourceforge.net

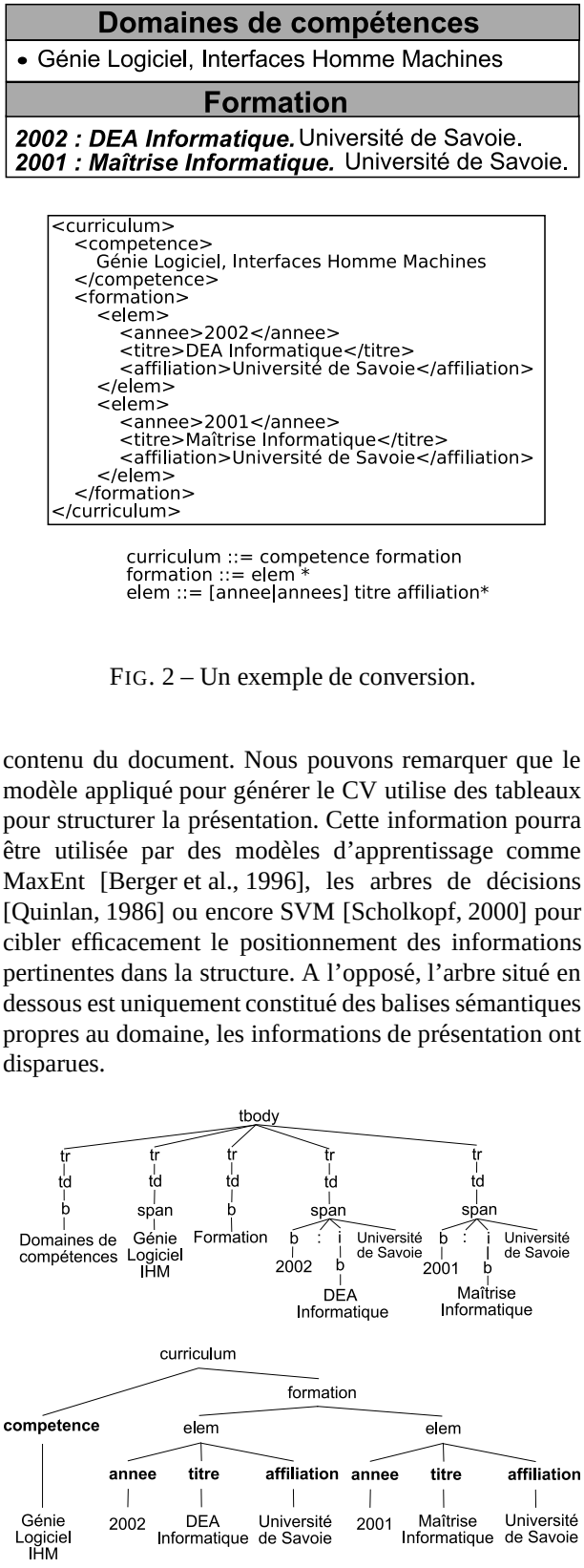

FIG. 2 – Un exemple de conversion.

contenu du document. Nous pouvons remarquer que le modèle appliqué pour générer le CV utilise des tableaux pour structurer la présentation. Cette information pourra être utilisée par des modèles d'apprentissage comme MaxEnt [Berger et al., 1996], les arbres de décisions [Quinlan, 1986] ou encore SVM [Scholkopf, 2000] pour cibler efficacement le positionnement des informations pertinentes dans la structure. A l'opposé, l'arbre situé en dessous est uniquement constitué des balises sémantiques propres au domaine, les informations de présentation ont disparues.

FIG. 3 – Représentation arborescente de l'exemple.

Dans ce papier, nous considérons le cas général de l'annotation arborescente d'un document semi-structuré. Nous ne faisons pas d'hypothèses sur la structure des documents cible et d'origine et nous ne recherchons pas des similarités de structure. Le contenu du document est représenté par une séquence d'observation $\mathbf{x} = \{x_1, \ldots, x_n\}$, où chaque observation x_i est un fragment de contenu à convertir. Dans le cas de documents html, les fragments sont les feuilles de l'arbre, elles sont entourées d'informations contextuelles sous la forme d'attributs html, de balises, etc.

Notre approche consiste à diviser le problème en deux sous-problèmes plus simples. La première étape consiste à parcourir la séquence de feuilles du document d'entrée pour estimer une séquence de classes associées à ces feuilles. Les classes possibles sont choisies parmi les éléments terminaux de la grammaire. Pour l'exemple précédent, les classes possibles sont en gras, ce sont *competence*, *annee*, *titre* et *affiliation*. La séquence de classes estimée $\mathbf{y_{est}}$ est utilisée comme point d'entrée pour le deuxième traitement, il a pour but la reconstruction d'un arbre de dérivation d associé à la séquence $\mathbf{y_{est}}$ en utilisant les contraintes grammaticales. Cette arbre de dérivation correspond à l'arbre sémantique recherché. La réalisation de ces deux étapes correspond à la conversion d'un document orienté présentation vers un document sémantique. Contrairement à l'approche la plus intuitive qui consiste à effectuer les deux étapes séquentiellement mais qui a pour principal inconvénient de cumuler les erreurs, nous verrons par la suite comment nous combinons ces deux étapes pour maximiser la probabilité jointe d'avoir le couple $(\mathbf{y}, d)$ le plus probable.

2.2 Classification probabiliste

La première étape définie par notre décomposition du problème est une étape d'apprentissage supervisé. A partir d'une séquence de feuilles $\mathbf{x}$, l'annotation consiste à estimer la séquence de classes $\mathbf{y}$ qui est la plus probable. Cette estimation est basée sur un modèle d'apprentissage entrainé avec un ensemble d'apprentissage, $S = \{(\mathbf{x}, \mathbf{y})\}$. En reprenant notre exemple, nous avons $\mathbf{x} = ($*"Domaines de compétences"*, *"Génie Logiciel, IHM"*, *"Formation"*, *"2002"*, *" :"*, *"DEA Informatique"*, *"Université de Savoie"*, *"2001"*, *" :"*, *"Maîtrise Informatique"*, *"Universite de Savoie"*$)$ et la séquence la plus probable recherchée $\mathbf{y} = ($*REMOVE*, *competence*, *REMOVE*, *annee*, *REMOVE*, *titre*, *affiliation*, *annee*, *REMOVE*, *titre*, *affiliation*$)$. Nous pouvons remarquer l'introduction d'une nouvelle classe *REMOVE* qui correspond à l'annotation des feuilles qui ne sont utiles que pour la présentation du document, sans conserver d'informations sémantiques. Ces feuilles doivent être supprimées lors de la conversion du document. Ces informations sont en général présentes de façon implicite dans le document sémantique et peuvent être regénérées pour produire une visualisation du document.

Un classifieur probabiliste crée un modèle d'apprentissage cohérent avec les exemples annotés par un expert qui soit le plus général possible, c'est-à-dire, qui soit le plus performant sur des données non vues en apprentissage. Le but recherché est de déterminer des caractéristiques propres à chaque classe, plus précisement des valeurs de caractéristiques, pour déterminer, à partir d'un x_i, la probabilité de chaque classe. Formellement, un classifieur

probabiliste cherche à estimer pour chaque classe la probabilité conditionnelle $P(y_i|x_i)$ qui définit la probabilité que l'observation (la feuille) x_i soit de la classe y_i. Si le classifieur probabiliste est utilisé seul, la classe estimée est la classe pour laquelle cette probabilité est maximale. Pour être utilisées avec des méthodes d'apprentissage existantes, les instances à classer (les x_i) doivent être projetées dans un modèle des données consistant. Dans ce modèle, une instance va être vue comme un vecteur de caractéristiques permettant de décrire le plus fidèlement possible les spécificités des instances d'une même classe. Nous classons ces caractéristiques suivants trois catégories différentes qui récupèrent des informations utiles pour la discrimination :

1. ***Les attributs de contenu***

   La première source d'attributs que nous pouvons utiliser concerne les fragments textuels du document d'origine, les feuilles de l'arbre. Ces attributs permettent de décrire précisement les caractéristiques spécifiques aux chaînes de caractères contenues dans les feuilles. Nous pouvons penser par exemple aux nombre de caractères de la chaîne, à la présence de caractères spéciaux ou encore au caractère numérique de la chaîne. Dans l'exemple précédent, le fait de savoir que la chaîne "2002" est numérique peut aider le classifieur à proposer une plus forte probabilité pour la classe nommée "année".

2. ***Les attributs de structure***

   La deuxième source d'attributs qui est à notre disposition concerne la structure de l'arbre html à convertir. Il peut être judicieux de connaître les balises proches d'une feuille dont nous cherchons à estimer la classe pour trouver des motifs structurels propres à chaque classe. Dans notre cas, les attributs sont à valeurs discrètes et permettent seulement de simuler les structures. Cependant, cette approche permet d'utiliser simplement la majorité des méthodes existantes et de ne pas avoir à mettre au point des méthodes plus spécifiques. Comme nous l'avons dit précédemment, la structure de tableau de l'exemple est une source d'informations structurelles pertinente. Les feuilles situées dans la première colonne par exemple peuvent être spécifiées par la description suivante : le père de la feuille est l'élément "td", il n'a pas de frère gauche et le grand père de la feuille est l'élément "tr".

3. ***Les attributs de contenu html***

   Enfin, la dernière source d'attributs est un mélange de structure et de contenu. Il s'agit des valeurs des attributs présents dans les élements html qui entourent la feuille en cours de traitement. Par exemple, il peut être intéressant de savoir que la valeur de la fonte du père de la feuille est "times" ou encore que le tableau est dessiné avec une bordure de deux pixels d'épaisseur.

En utilisant cette représentation, une feuille peut être projetée dans ce modèle et le classifieur probabiliste travaille alors sur cette représentation simplifiée. Les attributs extraits dont nous disposons sont de types hétérogènes (discrets, numériques ou booléens) mais principalement à valeurs discrètes. De plus, nous travaillons avec un nombre relativement important de classes ce qui limite les classifieurs que nous pouvons utiliser. Nous avons donc porté notre attention sur un classifieur basé sur le pricipe du maximum d'entropie [Berger et al., 1996], appelé aussi MaxEnt. Ce classifieur cherche à maximiser la probabilité conditionnelle $P(y|x)$, il fait l'hypothèse qu'elle suit une loi exponentielle :

$$P(y|x) = \frac{1}{Z_\alpha(x)} exp(\sum_\alpha \lambda_\alpha \cdot f_\alpha(x,y)), \qquad (1)$$

où $Z_\alpha(x)$ est un facteur de normalisation qui permet d'assurer que la valeur obtenue est une probabilité.

$$Z_\alpha(x) = \sum_y exp(\sum_\alpha \lambda_\alpha \cdot f_\alpha(x,y)). \qquad (2)$$

La variable α permet d'effectuer une somme sur l'ensemble des attributs choisis pour représenter le contexte d'une feuille x et la fonction $f_\alpha(x,y)$ représente la valeur de cet attribut α, pour le couple d'apprentissage (x,y). Les valeurs λ_α représentent une pondération des attributs et permettent de déterminer un modèle pour lequel la distribution définie soit la plus exacte possible pour les données de l'ensemble d'apprentissage. Pour chaque choix de $\lambda = (\lambda_{\alpha_1}, \ldots, \lambda_{\alpha_m})$ que nous pouvons faire, nous définissons donc un modèle différent, le classifieur MaxEnt va déterminer parmi toutes ces possibilités le modèle optimal, en utilisant le principe de maximum d'entropie. Ce principe privilégie les modèles les plus uniformes et permet de trouver un maximum local. Pour l'estimation itérative des paramètres du modèle, les λ_α, nous utilisons la méthode quasi Newton, appelée aussi L-BFGS qui est plus efficace que les méthodes habituelles GIS et IIS [Malouf, 2002].

2.3 Grammaires Hors-contextes probabilistes

Pour utiliser les contraintes grammaticales fournies par la grammaire xml qui définit la sémantique métier de la collection, nous utilisons des grammaires probabilistes. La partie des schémas xml (ou dtd) qui nous intéresse concerne uniquement les déclarations qui définissent la structure des arbres recherchés. Cette partie peut être transformée de manière équivalente vers le formalisme des grammaires hors-contextes probabilistes [Papakonstantinou et al., 2000].

Définition : Une grammaire hors-contexte probabiliste G est définie par un 5-uplet $< N, T, R, S, P >$ où :
- N est l'ensemble des symboles non terminaux,
- T est l'ensemble des symboles terminaux,
- R est l'ensemble des règles r_i de la forme : $A \rightarrow \alpha$, $A \in N$, $\alpha \in (N \cup T)*$,
- S est l'axiome de départ,

– P est l'ensemble des probabilités p_i associées aux règles r_i telles que :

$$\sum_{\alpha} p(A \rightarrow \alpha) = 1, \forall A \in N. \quad (3)$$

Définition : On note $A \stackrel{*}{\Rightarrow} B$ la fermeture transitive de la relation $\rightarrow$. $A \stackrel{*}{\Rightarrow} B$ si $A = A_0 \rightarrow A_1 \rightarrow \cdots \rightarrow A_n = B$, où $n \geq 0$.

Définition : On dit que $A \in N$ domine une chaîne $\mathbf{y} = (y_1, \ldots, y_n)$ si $A \stackrel{*}{\Rightarrow} \mathbf{y}$.

Dans notre cas, nous cherchons à trouver une séquence $\mathbf{y}$ qui soit dominée par S, l'axiome de départ qui est aussi la racine de l'arbre sémantique. La suite de règles de production $S \stackrel{*}{\Rightarrow} \mathbf{y}$ utilisée pour produire la séquence définit un arbre de dérivation d qui est équivalent à un arbre xml. La figure 4 schématise la domination de la racine par rapport à une séquence de classes $\mathbf{y} = (y_1, \ldots, y_n)$.

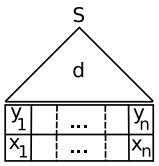

FIG. 4 – Dérivation de la grammaire.

Les règles de la grammaire hors-contexte probabiliste peuvent être écrites manuellement en se référant à la grammaire xml. Elles peuvent également être inférées automatiquement à partir des documents cible de la collection d'apprentissage. Les probabilités de chaque règles peuvent être calculées automatiquement en les dénombrant, en utilisant la formule suivante :

$$P(A \rightarrow \alpha) = \frac{nombre(A \rightarrow \alpha)}{\sum_{A \rightarrow \beta \in R} nombre(A \rightarrow \beta)}. \quad (4)$$

L'utilisation de grammaires probabilistes permet de proposer plusieurs arbres de dérivations pour une séquence d'éléments terminaux et donc d'introduire la notion d'arbre le plus probable qui correspond à l'arbre qui possède la plus grande probabilité. La probabilité d'un arbre de dérivation est calculée en effectuant le produit des probabilités des règles de production qui ont été utilisées pour le créer.

2.4 Combinaison de méthodes

La spécificité de nos travaux consiste à combiner les deux technologies précédentes, un classifieur probabiliste et des grammaires probabilistes, pour réussir à convertir automatiquement des documents semi-structurés [Chidlovskii et al., 2005]. Formellement, nous cherchons à trouver le couple $(\mathbf{y}, d)$ qui soit le plus probable étant donné une grammaire probabiliste G et un document d'origine projeté sous une forme vectorielle d'attributs $\mathbf{x}$. Cela peut s'écrire :

$$(y, d)_{max} = \underset{(y,d)}{argmax}\, P(y, d|x, G). \quad (5)$$

L'avantage de cette approche est que nous cherchons à maximiser une probabilité jointe entre y et d. En utilisant le théorème de Bayes et des hypothèses d'indépendances entre x et d et entre y et G, il est possible de reformuler l'équation 5 en :

$$(y, d)_{max} = \underset{(y,d)}{argmax}\, P(d|y, G) \cdot P(y|x). \quad (6)$$

Dans l'équation 6, la première partie correspond à la partie grammaticale et à la recherche de l'arbre de dérivation le plus probable pour une séquence d'éléments terminaux (des classes) fixée. La deuxième partie correspond à la classification probabiliste d'une séquence de feuilles pour estimer une séquence de classes (les terminaux).
La formule indique qu'il faut calculer la probabilité jointe pour tous les couples (y,d) et prendre le couple dont la valeur est maximale. La théorie impose d'effectuer le test pour tous les couples de valeurs possibles afin de trouver un maximum global. Ce n'est malheureusement pas réalisable en pratique. Afin de pallier à ce problème, nous avons mis en place une modification de l'algorithme inside-outside pour les grammaires hors-contextes probabilistes. Il permet de calculer efficacement la probabilité d'un arbre de dérivation à partir d'une séquence donnée [Lari et al., 1990]. Plus spécifiquement, nous modifions la partie inside de l'algorithme en injectant la distribution de probabilité estimée par le classifieur probabiliste. Cet algorithme est dynamique et a une complexité en $O(n^3 \cdot |N| + n \cdot |T| \cdot |N|)$, avec n la longueur de la séquence, $|N|$ le nombre de non-terminaux de la grammaire et $|T|$ le nombre de terminaux. Cependant, il est important de noter que l'utilisation de cet algorithme impose de travailler sur des grammaires en formes normales de Chomsky, ce qui revient à binariser les documents cibles. L'algorithme situé dans le tableau 1 présente une méthode qui permet de transformer une grammaire exprimé par une dtd vers une grammaire hors-contexte en forme normale de Chomsky. Les probabilités associées à chaque règle peuvent être calculées en observant les règles utilisées dans les documents de l'ensemble d'apprentissage.

2.5 Exemple

Considérons un schéma cible donné par la dtd suivante :

```
<!ELEMENT  Book      (author, Section+)>
<!ELEMENT  Section   (title, (para | footnote)+)>
<!ELEMENT  author    (#PCDATA)>
<!ELEMENT  title     (#PCDATA)>
<!ELEMENT  para      (#PCDATA)>
<!ELEMENT  footnote  (#PCDATA)>
```

La réduction de la définition du schéma précédent à une forme normale de Chomsky introduit de nouveaux non terminaux. Nous obtenons la grammaire hors-contexte probabiliste $G = (T, N, \mathcal{S}, R)$, où l'ensemble des terminaux est T = {`author, title, para, footnote`},

Entrée : Une grammaire xml (dtd)
Sortie : Une grammaire hors-contexte

1 - *Conversion en grammaire hors-contexte.*
Un algorithme de réécriture peut être trouvé dans [Hopcroft et al., 2001].

2 - *Réduction de la grammaire hors-contexte.*
Une grammaire réduite ne contient pas de règles de production inutiles ni de non terminaux inutiles.

3 - *Binarisation de la grammaire réduite.*
L'algorithme se décompose en deux étapes :
a) Pour tous les terminaux a :
- rajouter T_a dans N,
- remplacer a par T_a dans les parties droites des règles,
- rajouter la règle $T_a \rightarrow a$ dans R.
b) Tant qu'il existe une règle de la forme $X \rightarrow Y\beta$, $X, Y \in N, \beta \in NN^+$:
- remplacer $X \rightarrow Y\beta$ par $X \rightarrow YZ$ et $Z \rightarrow \beta, Z \in N$, avec $Z \in N$ un nouveau non terminal.

TAB. 1 – Algorithme de binarisation des grammaires.

l'ensemble des non terminaux est N = {Book, Author, SE, Section, TI, ELS, EL}, $\mathcal{S}$ = Book, et R est composé de douze règles de production.
Nous supposons avoir appris un classifieur probabiliste C et les paramètres de la grammaire G sur un ensemble d'apprentissage et avoir obtenu les probabilités suivantes pour les règles de production de R (les probabilités sont entre parenthèses) :

(0.3) Book → AU Section	(0.7) Book → AU SE
(0.4) SE → Section Section	(0.6) SE → Section SE
(0.8) Section → TI ELS	(0.2) Section → TI EL
(0.4) ELS → EL EL	(0.6) ELS → EL ELS
(1.0) AU → `author`	(1.0) TI → `title`
(0.8) EL → `para`	(0.2) EL → `footnote`.

Nous supposons maintenant avoir testé le classifieur probabiliste C sur une séquence de cinq feuilles $\mathbf{x} = \{x_1, \ldots, x_5\}$. Il estime la distribution des probabilités pour chaque terminal de T pour chaque feuille et fournit le tableau suivant :

	x_1	x_2	x_3	x_4	x_5
`author`	0.3	0.2	0.1	0.1	0.2
`title`	**0.4**	**0.4**	0.3	0.3	**0.3**
`para`	0.1	0.2	**0.5**	0.2	0.2
`footnote`	0.2	0.2	0.1	**0.4**	0.2

En suivant la distribution de probabilité estimée, la séquence de terminaux la plus probable $\mathbf{y}_{max}$, représentée en gras dans le tableau, est composée des terminaux les plus probables pour tous les x_i, $i = 1, \ldots, 5$. Il s'agit de `'title title para footnote title'` avec une probabilité $p(\mathbf{y}_{max}) = p(\mathbf{y}_{max}|\mathbf{x}) = \Pi_i \cdot p(\mathbf{y}_i^{max}|x_i) = 0.4 \cdot 0.4 \cdot 0.5 \cdot 0.4 \cdot 0.3 = 0.0096$. Cependant, $\mathbf{y}_{max}$ ne possède pas d'arbre de dérivation dans G. Par contre, il existe deux arbres de dérivations valides pour $\mathbf{y}$, $(\mathbf{y}_1, d_1)$ and $(\mathbf{y}_2, d_2)$, comme le montre la figure 5. Dans la figure 5.b, la séquence de terminaux $\mathbf{y}_2$=`'author title para title para'` avec l'arbre de dérivation d_2=Book(AU SE(Section (TI EL) Section (TI EL))) fournit la probabilité jointe $p(\mathbf{y}, d|\mathbf{x}, G) = p(\mathbf{y}) \cdot p(d)$, avec $p(\mathbf{y}_2) = 0.3 \cdot 0.4 \cdot 0.5 \cdot 0.3 \cdot 0.2 = 0.0036$, et

$$\begin{aligned} p(d_2) =& p(\text{Book} \rightarrow \text{AU SE}) \cdot p(\text{AU} \rightarrow \text{author}) \times \\ & p(\text{SE} \rightarrow \text{Section Section}) \cdot p(\text{Section} \rightarrow \text{TI EL}) \times \\ & p(\text{TI} \rightarrow \text{title}) \cdot p(\text{TI} \rightarrow \text{title}) \times \\ & p(\text{EL} \rightarrow \text{para}) \cdot p(\text{EL} \rightarrow \text{para}) \times \\ & p(\text{Section} \rightarrow \text{TI EL}) \\ =& 0.7 \cdot 1.0 \cdot 0.4 \cdot 0.2 \cdot 1.0 \cdot 1.0 \cdot 0.8 \cdot 0.8 \cdot 0.2 \\ =& 0.007172. \end{aligned}$$

Ainsi, nous avons $p(\mathbf{y}_2) \cdot p(d_2) \approx 2.58 \cdot 10^{-5}$. De la même façon, pour l'arbre de dérivation de la figure 5.a, nous avons $p(\mathbf{y}_1) \cdot p(d_1) = 0.0048 \cdot 0.0018432 \approx 8.85 \cdot 10^{-6}$. La dérivation 2 est donc celle qui maximise la probabilité jointe et produit le résultat optimal de la conversion.

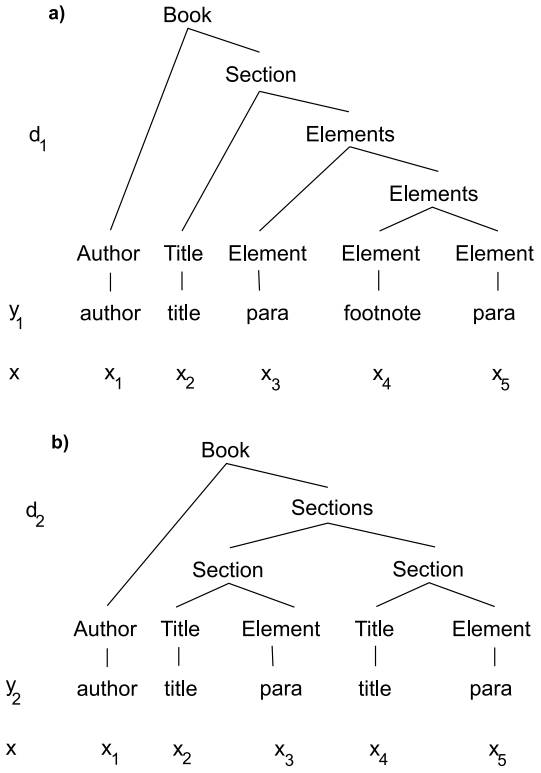

FIG. 5 – Annotations d'arbre pour la séquence d'exemples.

3 RÉSULTATS

Nous avons testé notre méthode pour l'annotation xml sur deux collections. La première est une collection de 39 pièces de shakespeare, disponibles dans les formats html et xml sur le web[7] . Nous avons extrait aléatoirement 60

[7]Les fichiers html sont disponibles ici : http ://www-tech.mit.edu/Shakespeare, les fichiers xml sont présents ici : http ://www.ibiblio.org/xml/examples/shakespeare

scènes de ces pièces pour l'évaluation, elles possèdent de 17 à 189 feuilles. Le fragment de la dtd correspondant aux scènes est composé de 4 terminaux et de 6 non-terminaux. Après la transformation en forme normale de Chomsky, la grammaire contient 8 non-terminaux et 18 règles de production.

La seconde collection, appelée TechDoc, est constituée de 60 documents techniques décrivant des opérations de maintenance. Les documents cibles ont une granularité sémantique bien plus fine que pour la collection des pièces de Shakespeare et ont une profondeur plus importante. Le plus long document possède 218 feuilles. Le schéma cible est donné par une dtd complexe qui possède 27 terminaux et 35 non-terminaux. La tranformation en forme normale de Chomsky augmente le nombre de non-terminaux à 53.

Pour évaluer la précision de notre annotation, nous utilisons deux métriques différentes. Le *Pourcentage d'Erreurs Terminales* (PET) est similaire au pourcentage d'erreurs sur les mots en traitement du langage naturel, il calcule le pourcentage d'éléments terminaux (classes) qui ont été correctement annotés dans les documents de test [Lehnert et al., 1991]. La deuxième métrique est le *Pourcentage d'Erreurs des Non-terminaux* (PEN) qui calcule le pourcentage de sous-arbres correctement annotés. Nous considérons un sous-arbre bien annoté si l'estimation du symbole N^j dominant la séquence $y_i \dots y_j$ correspond effectivement au symbole dominant la même séquence dans le document à obtenir. La précision PEN correspond donc au rapport du nombre de nœuds corrects sur le nombre de nœuds total.

Pour le modèle d'apprentissage de MaxEnt, nous extrayons 38 attributs de contenu pour chaque observation comme le nombre de mots, sa longueur, etc. Ensuite, nous extrayons 14 attributs de structure et de présentation qui incluent les balises entourant la feuille ainsi ques les attributs xml associés.

Pour chaque test, nous effectuons une validation croisée. Nous avons testé la classification de séquence de terminaux seuls avec différents classifieurs, MaxEnt en utilisant ensemble les 52 attributs et deux classifieurs basé sur Naive Bayes qui utilisent respectivement les attributs de contenu et les attributs de structure et de présentation. Ces classifieurs servent de référence pour la comparaison avec notre approche qui rajoute des contraintes grammaticales pour guider la classification de MaxEnt, ils fournissent les résultats pour PET. Les trois dernières lignes présentent les résultats de la combinaison des classifieurs avec les grammaires, cette combinaison permet de calculer PET et PEN.

Les résultats des différents tests sont collectés dans le tableau 2. La méthode combinatoire permet de montrer une amélioration de chaque classifieur simple et nous montre que les contraintes grammaticales permettent de récupérer certaines erreurs de classification. Cependant, pour réaliser une conversion complète et produire un arbre sémantique correct, nous pouvons remarquer qu'un classifieur performant (supérieur à 85 %) est nécessaire pour permettre de lever les ambiguïtés de la grammaire et utiliser les bonnes règles de production.

Method	TechDoc		Shakespeare	
	PET	PEN	PET	PEN
MaxEnt	92.68	–	100.0	–
NB - contenu	71.84	–	81.90	–
NB - structure	76.37	–	99.95	–
MaxEnt + G	93.39	80.00	99.97	99.81
NB - contenu + G	78.40	65.53	81.94	28.38
NB - structure + G	88.58	76.44	99.95	99.84

TAB. 2 – Résultats de l'évaluation.

4 AUTRES APPROCHES

La transformation de documents définis dans un schéma source (basé sur la présentation dans notre cas) vers des documents définis dans un schéma cible (fourni par un utilisateur dans notre cas) a fait l'objet de plusieurs langages de transformations d'arbres comme xpath ou xslt [XSLT] par exemple. Ils fournissent tous des outils de programmation très puissants qui permettent de réaliser un grand nombre de tâches liées à la transformation de documents.

Ces approches sont déclaratives et nécessitent une écriture manuelle des règles de transformation. Des méthodes d'apprentissages comme [Curran, 1999] peuvent apprendre des règles simples de transformation. Elles supposent que des documents sources peuvent être transformés dans des documents xml grâce à une série d'opérations de transformations élémentaires comme l'insertion, le remplacement, la supression et l'échange. Le modèle de traduction apprend un ensemble d'opérations qui minimisent une erreur donnée par une fonction d'évaluation.

Dans le domaine de l'analyse de la présentation des documents, l'utilisation de balises xml ou html peuvent faciliter la récupération de documents sur le web. Des systèmes comme [Wisdom++, 2001][Wang, 1999] sont ainsi capables de transformer des documents scannés sous la forme de documents bien structurés. Cependant, le résultat de ces systèmes restent orientés présentation et contiennent très peu d'informations sémantiques. L'objectif principal est de préserver une visualisation qui soit la plus proche possible du document original dans un navigateur web.

Une autre catégorie de système adresse le problème de conversion de documents. Ces méthodes, comme [Chung, 2002], traite plus particulièrement de la conversion de documents html vers des documents xml. En analysant les collections et en utilisant des techniques d'apprentissage non supervisées, l'auteur définit des méthodes manuelles d'extraction et des règles de composition qui sont capable de trouver des motifs structurels représentatifs dans l'arbre d'entrée, de définir un label à affecter à un élément extrait et de finalement restructurer les éléments pour former un arbre converti.

5 CONCLUSION

Nous proposons une méthode probabiliste pour l'annotation xml de documents semi-structurés. Le problème de l'annotation d'arbre est réduit à la dérivation hors-contexte probabiliste d'une séquence d'observation. Nous déterminons l'arbre d'annotation le plus probable en maximisant la probabilité jointe d'estimer une séquence de symboles terminaux et de dériver un arbre pour cette séquence.
Nous avons étendu l'algorithme inside-outside pour les grammaires hors-contextes probabilistes et avons défini un algorithme dynamique efficace pour estimer cette probabilité jointe.
Les résultats expérimentaux sont encourageants. Dans le futur, nous envisageons d'adresser de nouveaux challenges dans l'automatisation de la conversion de documents html vers xml. Nous sommes plus particulièrement intéressés dans la prise en compte des structures d'arbres d'entrée dans le modèle d'apprentissage. Nous envisageons également de rendre les algorithmes actifs pour minimiser la tâche de l'annotation des documents pour l'apprentissage supervisé

BIBLIOGRAPHIE

[Berger et al., 1996] Berger, A.L., Della Pietra, S., Della Pietra, V.J. : A Maximum Entropy Approach to Natural Language Processing. Computational Linguistics, 22(1), 39-71, 1996.

[Chidlovskii et al., 2004] Chidlovskii, B., Fuselier, J. : Supervised Learning for the Legacy Document Conversion. The ACM Symposium on Document Engineering (DcEng'04), Milwaukee, Wisconsin, USA, October 28-30, 2004.

[Chidlovskii et al., 2005] Chidlovskii, B., Fuselier, J. : A Probabilistic Learning Method for XML Annotation of documents. International Joint Conference on Artificial Intelligence. Edimbourgh, Ecosse, 2005. A paraitre.

[Chung, 2002] Yip Chung, C., Sundaresan, N., Gertz, M. : Reverse engineering for web data : From visual to semantic structures. 18th International Conference on Data Engineering (ICDE'02), San Jose, California, 2002.

[Curran, 1999] Curran, J.R., Wong, R.K. : Transformation-Based Learning for Automatic Translation from HTML to XML". Proceedings of the Fourth Australasian Document Computing Symposium (ADCS99), 1999.

[DocBook] Guide to the DocBook DTD Documentation version 1.0 for release 2.2.1. citeseer.ist.psu.edu/50447.html.

[Fuselier et al., 2002] Fuselier, J., Marty, J.C., Vignollet, L. : Une génération automatique de documents virtuels personnalisables guidée par des contraintes. Documents Virtuels Personnalisables (DVP'02), Brest, France, Juillet 2002.

[Hopcroft et al., 2001] Hopcroft, J.E., Motwani, R., Ullman, J. : Introduction to Automata Theory, Languages, and Computation - 2nd Edition.

[Lari et al., 1990] Lari, K., Young, S.J. : The estimation of stochastic context-free grammars using the inside-outside algorithm. Computer Speech and Language, 4, 35–56, 1990.

[Lehnert et al., 1991] Lehnert, W., Sundheim, B. : A Performance Evaluation of Text-Analysis Technologies. AI Magazine, 12(3), 81-94, 1991.

[Malouf, 2002] Malouf R. : A comparison of algorithms for maximum entropy parameter estimation. Proceedings of the 6th Conference on Natural Language Learning, 49–55, 2002.

[Papakonstantinou et al., 2000] Papakonstantinou, Y., Vianu, V. : DTD Inference for Views of XML Data. Proceedings of the 19th ACM Symposium on Principles of Database Systems (PODS), 35-46, Dallas, Texas, USA, 2000.

[Quinlan, 1986] Quinlan, R. : Induction of decision trees. Machine Learning, 1, 81–106, 1986.

[Scholkopf, 2000] Schölkopf, B. : Statistical learning and kernel methods. MSR-TR 2000-23, Microsoft Research, 2000.

[Wang, 1999] Wang, Y., Phillips, I.T., Haralick, R. : From image to SGML/XML representation : One method. International Workshop on Document Layout Interpretation and Its Applications (DLIAP'99), Bangalore, India, September 1999.

[Wisdom++, 2001] Oronzo, A., Esposito, F., Malerba D. : Transforming paper documents into XML format with WISDOM++. IJDAR,4(1), 2-17, 2001.

[XSLT] XSL Transformations. www.w3.org/TR/xslt.