

Codages et connaissances en extraction d'information

Patrick MARTY¹, Fabien TORRE²

¹ Mostrare - UR INRIA Futurs

² GRAppA - Université Charles de Gaulle - Lille 3

Résumé : Ce travail se place dans le cadre d'un projet de recherche sur l'induction de programmes d'extraction d'information à partir de données du WEB. Dans le contexte de ce projet, nous développons une plate-forme conjuguant la vue textuelle des documents et leur structure, permettant l'introduction de connaissances du domaine et la combinaison d'algorithmes d'apprentissage. Dans le présent article, nous nous limitons à la vue textuelle et choisissons un codage simple de la forme attribut-valeur permettant facilement d'introduire des connaissances du domaine. Nous montrons que ce codage associé à un algorithme de classification supervisée classique, en l'occurrence C4.5, permet d'obtenir des performances significatives.

Introduction

Ces dernières années, Internet et le World Wide Web ont connu un développement notable. Le WEB est devenu la plus grande source de données et d'informations de la planète. Devant le nombre croissant de documents électroniques disponibles en ligne, le besoin d'outils performants pour traiter l'information qu'ils contiennent est de plus en plus prégnant. L'extraction d'information répond en partie à ce besoin. Elle consiste à extraire automatiquement des données d'un document ou d'un corpus de documents.

Par exemple, dans le cas du corpus **Seminar Announcements**¹, composé de 485 annonces de séminaires (au format texte), il s'agit d'extraire, pour chaque document, l'heure de début (*stime*) et l'heure de fin (*etime*) du séminaire, l'endroit où il a lieu (*location*) et les noms des intervenants (*speaker*) (voir figures 1 et 2). Sur cet exemple, il y a quatre données différentes à extraire : dans la terminologie de l'extraction d'information, on parle de *champ* ou de *slot*. Par ailleurs, un champ donné peut avoir une ou plusieurs instances dans un même document (pour un séminaire, il peut y avoir plusieurs intervenants).

Un *wrapper* est un programme d'extraction d'information qui produit des données structurées à partir de documents. L'écriture manuelle d'un tel programme étant fastidieuse et source d'erreurs, les documents eux-mêmes étant changeants, les recherches se sont orientées vers leur génération automatique : l'*induction de wrappers*.

¹ disponible à l'adresse <http://www.isi.edu/info-agents/RISE/>.

Diverses techniques d'induction ont été développées, chacune dédiée à un format précis de documents, du texte libre en langage naturel au fichier XML à la structure rigide. Certains systèmes emploient des techniques de traitement du langage naturel pour utiliser le sens du texte (Soderland *et al.*, 1995; Huffman, 1995). D'autres au contraire se fondent essentiellement sur l'aspect syntaxique ou structurel du document. Dans ce dernier cas, un document peut être vu soit comme une séquence « plate » (Kushmerick, 1997; Hsu & Dung, 1998), soit comme un arbre (Sakamoto *et al.*, 2001; Kosala *et al.*, 2002), notamment dans le cas de documents HTML ou XML.

Une manière de traiter l'induction de wrapper consiste à se ramener à un problème de *classification supervisée*, puis à utiliser l'un des nombreux algorithmes dédiés à ce type de tâches (Cornuéjols & Miclet, 2002). En classification supervisée, on suppose fournis des exemples étiquetés par leurs classes ; ainsi dans le cas le plus simple (à deux classes), chaque exemple pourra être soit *positif*, soit *négatif*. À partir de ces données, il s'agit alors d'apprendre un classifieur capable de prédire la classe d'un nouvel exemple. Les premiers travaux utilisant la classification supervisée pour l'extraction d'information montrent que la piste est prometteuse (Seymore *et al.*, 1999; Freitag & McCallum, 1999; Freitag & Kushmerick, 2000; Freitag & McCallum, 2000; Kosala & Blockeel, 2000; Chieu & Ng, 2002).

Ces systèmes diffèrent sur plusieurs points :

- ils utilisent des techniques algorithmiques différentes et parfois sophistiquées (modèles de Markov cachés, boosting, algorithmes d'alignements, etc.) ;
- ils reposent sur des représentations différentes (tokenisation, utilisation de fenêtres, règles sous forme d'expressions régulières ou en attribut-valeur, etc.) ;
- ils intègrent des connaissances propres à la tâche d'extraction considérée (utilisation de dictionnaires par exemple).

Finalement, il est difficile de juger dans quelle mesure chacun de ces choix participe aux performances observées d'un système d'extraction. Pour ajouter à la confusion, précisons ici que les protocoles d'évaluation sont rarement explicités et pas toujours équitables, ce qui rend d'autant plus difficile leur comparaison (Califf *et al.*, 2004).

Nous proposons le développement d'une plate-forme qui dissocie les différentes briques impliquées dans un système d'extraction et permette à l'utilisateur de les combiner de manière interactive.

En particulier, nous voulons permettre à l'utilisateur d'introduire des connaissances propres à l'objectif visé. On observe en effet que la définition d'un problème d'extraction entraîne naturellement la donnée du type des éléments à extraire ou même de mots ou symboles importants. Par exemple la recherche de prix incite à prendre en compte les symboles monétaires, la recherche d'une heure à rechercher deux valeurs numériques séparées par le symbole de ponctuation «:». L'utilisateur peut donc faciliter la tâche d'un apprenant en intégrant ces informations directement.

Nous étudions dans le présent article l'influence du langage de représentation choisi et de connaissances auxiliaires sur les performances d'un système d'induction de wrappers, en nous limitant aux documents textuels. Nous montrons qu'un codage simple de la forme attribut-valeur, basé sur des informations basiques et sur des ressources simples, associé à un algorithme de classification supervisée classique permet d'induire des wrappers ayant de bonnes performances.

<0.23.1.95.12.06.23.kl0y+@andrew.cmu.edu.0>
 Type : cmu.andrew.official.career-center
 Topic : Marketability Talk
 Dates : 31-Jan-95
 Time : 7 :00 PM
 PostedBy : Karen Litzinger on 23-Jan-95 at 12 :06 from andrew.cmu.edu
 Abstract : Shelley Brozenick from Alumni Relations and Karen Litzinger from the Career Center would like to announce the third alumni career talk in a series of four. "Strategies to Gain Experience and Increase Marketability". The Student Alumni Relations Council (SARC) invites you to a Career Talk featuring Andrew Gault, HS'80, HNZ'94 and three university speakers Mary Francis McLaughlin(volunteer and community service opportunities), Jessie Ramey (research opportunities), and Judi Mancuso (part-time, work-study, summer and internship opportunities) on Tuesday, January 31 at 7 :00 PM in the Carnegie Conference Room, 1st Floor Warner Hall. Call #8-8451 on or before Monday, January 30 to reserve a seat.

FIG. 1 – Un document du corpus Seminar Announcements.

Champ	Donnée(s)
STIME	7 :00 PM
ETIME	
LOCATION	Carnegie Conference Room, 1st Floor Warner Hall
SPEAKER	Andrew Gault, Mary Francis McLaughlin, Jessie Ramey, Judi Mancuso

FIG. 2 – Résultat de l'extraction.

Le plan de l'article est le suivant :

- nous discutons à la section 1 des liens entre extraction d'information et classification supervisée en illustrant notre propos à l'aide du système BWI (Freitag & Kushmerick, 2000) ;
- à la section 2, nous présentons un nouveau codage d'un problème d'extraction d'information basé sur une description purement attribut-valeur, permettant une intégration facile de connaissances du domaine ;
- nous décrivons ensuite, section 3, les premières expérimentations menées avec les différents langages testés sur des données classiques en extraction d'information et présentons les résultats obtenus ;
- enfin, nous concluons à la section 4 en tirant un premier bilan de ce travail et en indiquant les perspectives qu'il ouvre.

1 Extraction d'information et classification supervisée

En extraction d'information, les wrappers sont généralement construits sur la base de textes dans lesquels l'information à extraire est marquée. Il est donc naturel d'envisager l'utilisation de la classification supervisée pour résoudre ce problème ; d'autant plus que de nombreuses méthodes sont à disposition, souvent accompagnées de garanties théoriques ou empiriques (Cornuéjols & Miclet, 2002).

La définition d'un problème d'apprentissage passe par le choix de deux langages (Cohen & Feigenbaum, 1982) :

- le langage permettant de représenter les exemples ;
- le langage permettant de représenter les hypothèses (autrement dit, l'espace de recherche de l'algorithme d'apprentissage).

Nous discutons maintenant de la définition de ces différents langages.

1.1 Des textes aux exemples

La première étape consiste à découper le texte en unités lexicales ou *tokens*. Pour ce faire, il faut définir les unités lexicales retenues : caractère, séquence de caractères dénotée par une expression régulière, etc.

Ces choix pour la *tokenisation* conduisent à définir les types des tokens : `Alpha` pour les chaînes alphabétiques, `Ponct` pour les symboles de ponctuation, `Num` pour les chaînes numériques, etc.

Une hiérarchie de ces types de tokens peut alors être construite suivant les connaissances disponibles : par exemple, `Token` dénote n'importe quel token de n'importe quel type ; `PasAnglais` désigne tout token de type alphabétique qui ne se trouve pas dans le dictionnaire anglais disponible sur `Unix/Linux`.

Finalement, chaque token sera accompagné de son type, celui-ci pouvant aller du plus général (token quelconque) au plus spécifique (le contenu du token lui-même) en passant par des types intermédiaires (*nom propre* par exemple). Ainsi, les espaces étant ignorés, le texte '*Time : 9 PM*' peut être représenté par la séquence de tokens suivante :

Time	:	9	PM
Alpha	Ponct	Num	Alpha
Token	Token	Token	PasAnglais Token

Du processus de transformation d'un document en une séquence de tokens, émerge la notion de *séparateur*. Un séparateur est soit une interposition entre deux tokens adjacents, soit la position avant (respectivement après) le premier (respectivement dernier) token de document. Ainsi, la séquence précédente contient cinq séparateurs distincts indiqués ci-dessous par des flèches :

↓	Time	↓	:	↓	9	↓	PM	↓
s_0		s_1		s_2		s_3		s_4

Si la donnée à extraire est '9 PM', alors s_2 et s_4 sont respectivement le séparateur de début et de fin de cette donnée, et s_0, s_1, s_3 sont des séparateurs quelconques.

En vue de l'utilisation d'un algorithme de classification supervisée, il faut maintenant utiliser la séquence de tokens et les sous-séquences marquées à *extraire* pour constituer une base d'exemples. Un préalable est de décider de ce que va représenter un exemple. Pour cela, deux choix sont possibles :

- un exemple code un séparateur (de début ou de fin d'une donnée à extraire ou tout autre séparateur) (Freitag & Kushmerick, 2000) ; dans ce cas, les apprentissages des débuts et des fins se feront indépendamment ;
- un exemple code une sous-séquence (Kosala & Blockeel, 2000), éventuellement réduite à un seul token (Seymore *et al.*, 1999; Freitag & McCallum, 1999; Freitag & McCallum, 2000; Chieu & Ng, 2002) ; par rapport au premier, ce choix revient à apprendre en même temps les caractérisations des débuts et des fins.

La description des exemples étant faite, il reste à définir un langage d'hypothèses et à choisir un algorithme de classification supervisée qui aura la charge de trouver dans cet espace une hypothèse cohérente avec les exemples disponibles. Enfin, une hypothèse étant apprise, il faut spécifier le wrapper correspondant. En particulier dans le cas où les exemples codent des séparateurs, il faut préciser comment réassocier les séparateurs de début et de fin.

Dans la section suivante nous décrivons BWI qui utilise le codage de séparateurs indépendants.

1.2 Le système BWI

Le système BWI (Freitag & Kushmerick, 2000) distingue trois types de token :

- séquence de caractères alphanumériques ;
- un caractère de ponctuation ;
- le caractère 'retour chariot'.

Un exemple décrit un séparateur qui partitionne le texte en deux séquences de tokens (la séquence avant le séparateur considéré et la séquence qui vient après ce séparateur). On apprend deux fonctions : une première pour reconnaître les séparateurs de début d'une information à extraire, une autre pour reconnaître les séparateurs de fin. Dans le premier cas, les séparateurs de début sont les exemples positifs et les autres séparateurs constituent les exemples négatifs ; dans le second, ce sont les séparateurs de fin qui jouent le rôle des exemples positifs et les autres séparateurs celui des exemples négatifs.

Les hypothèses, destinées à caractériser les séparateurs représentés par les exemples positifs, utilisent ici des motifs comparables à des expressions régulières et portant sur les tokens et leurs types. Précisément, une hypothèse est un couple (g, d) de tels motifs : cette hypothèse couvre un séparateur donné si g matche la séquence de tokens avant le séparateur et si d matche la séquence de tokens après le séparateur. Par exemple, l'hypothèse suivante :

('Time' Token , Num Ponct Num PasAnglais)

couvre le séparateur noté par \uparrow dans le texte 'Time : \uparrow 9 :30 PM'. La forme d'un motif, et donc sa longueur, ne sont pas fixées *a priori* mais découvertes par apprentissage.

TAB. 1 – Performances d'extraction de BWI.

Problème	Rappel	Précision	F -mesure
stime	99.6 %	99.6 %	99.6 %
etime	94.9 %	94.4 %	94.6 %
location	69.6 %	85.4 %	76.7 %
speaker	59.2 %	79.1 %	67.7 %

Pour effectuer les apprentissages des début et fin, BWI utilise l'algorithme de boosting ADA-BOOST (Freund & Schapire, 1997; Schapire & Singer, 1998). Cette méthode nécessite la donnée d'un *apprenant faible*, c'est-à-dire d'un algorithme d'apprentissage faisant un peu mieux que l'aléatoire. Dans BWI, cet apprenant se nomme LEARNDETECTOR. Il construit itérativement une hypothèse par extension de ses motifs, initialement vides. À chaque étape, un motif peut être étendu par l'ajout d'au plus L tokens ou types. Tous les motifs possibles, à gauche et à droite, sont énumérés et celui contribuant le plus à améliorer les performances de classification de l'hypothèse est conservé. Ce processus est répété tant qu'il est possible d'améliorer l'hypothèse courante.

Les deux classifieurs produits permettent d'identifier dans un nouveau texte les séparateurs de début et de fin : il reste à les ré-associer. Pour cela, BWI enregistre les longueurs des séquences de tokens d'un champ observées sur les données d'apprentissage. Deux séparateurs, le premier identifié comme un début et le second comme une fin, sont associés si le nombre de tokens entre les deux a déjà été observé lors de l'apprentissage.

Force est de constater que BWI est l'un des meilleurs systèmes d'extraction d'information sur les données textuelles (la table 1 présente les performances de BWI sur le corpus Seminar Announcements).

Cependant, les critiques suivantes peuvent être émises sur BWI :

- Considérer les séparateurs de début et de fin indépendamment rend délicate leur ré-association. En effet, si BWI n'a rencontré que des séquences de longueur 1, 3, 4, et 5, aucune séquence de taille 2 ne sera extraite. De plus, il y a une perte d'information à dissocier les débuts et les fins pendant l'apprentissage : la description d'un séparateur n'est pas informée du séparateur qui lui est associé ce qui empêche de découvrir des règles comme *le séparateur de début est précédé du caractère «:» et le séparateur de fin suivi du caractère «.»* ;
- l'algorithme LEARNDETECTOR peut se révéler très coûteux : celui-ci génère les règles par extension en essayant d'ajouter à chaque fois au motif une fenêtre de taille L . Au pire, un motif peut couvrir l'ensemble du texte et la complexité pour le construire est alors exponentielle en L .
- BWI utilise des connaissances du domaine pour atteindre de bonnes performances. Ces connaissances sont internes à l'algorithme et ne peuvent être modifiées. D'une part, le type de motifs est fortement intégré dans l'apprenant faible LEARNDETECTOR. Changer de motifs signifiait concevoir un nouvel apprenant spécifique. D'autre part, la modification des ressources modélisant les connaissances du domaine (comme le dictionnaire des mots an-

glais) n'est pas non plus possible aisément. L'utilisateur ne peut pas par exemple rajouter un dictionnaire personnel contenant des valeurs clés pour résoudre son problème d'extraction d'information.

et al. (Kauchak *et al.*, 2002) ont essayé de comprendre pourquoi les performances de BWI sont bonnes. Pour ce faire, ils ont construit SWI un système basé sur BWI en remplaçant l'algorithme de boosting par une boucle de couverture sur les exemples positifs, et en utilisant l'apprenant LEARNDETECTOR. D'après les expériences menées, les auteurs concluent que SWI est inférieur à BWI, et que le boosting est la raison du succès de BWI.

Cependant, on peut leur opposer que l'écart de performances entre les deux systèmes est faible sur un certain nombre de problèmes et que les candidats pour expliquer le succès de BWI restent les mêmes :

- l'algorithme d'apprentissage (pour trancher il aurait fallu essayer un autre apprenant que LEARNDETECTOR, avec et sans boosting) ;
- les connaissances du domaine utilisées (dictionnaires) ;
- la forme des hypothèses et la valeur de L .

Dans la suite, nous discutons des deux derniers points, autrement dit de l'influence sur les performances d'un système des connaissances du domaine injectées et du langage choisi pour les hypothèses.

2 Un système d'extraction d'information avec codage attribut-valeur et ressources

Les réflexions sur le système BWI nous conduisent à définir les caractéristiques suivantes pour notre plateforme :

- apprentissage simultané des débuts et des fins des informations à extraire ;
- possibilité d'utiliser des vues (texte, balises, arborescence, etc.) sur le document et des langages de description différents ;
- intégration facilitée des connaissances du domaine ;
- choix entre plusieurs algorithmes d'apprentissage.

Dans cet article, nous voulons distinguer ce qui est propre au codage et aux ressources de ce qui provient de l'algorithme d'apprentissage. Pour cela, nous explorons les possibilités d'un langage d'hypothèses plus simple que celui de BWI en adoptant une représentation attribut-valeur et nous utilisons un algorithme de classification supervisée classique, en l'occurrence C4.5 (Quinlan, 1993).

2.1 Représentation des exemples

Le codage choisi porte sur des couples de séparateurs. L'information portée par un couple de séparateurs est beaucoup plus riche et l'on peut donc espérer découvrir avec ce codage des régularités qui seraient perdues dans le codage par séparateurs indépendants. De plus, l'extraction est simplifiée car il suffit de prédire si un couple de positions correspond à une information à extraire. Il n'est donc plus nécessaire de réassocier début et fin. La contrepartie est que le

speaker	stime	etime	location
Dr	:	:	conference
Professor	Time	Time	room
University	at	-	located
School	Dates	Dates	Dr
Research		at	Professor
			University
			School
			Research

TAB. 2 – Dictionnaires *utilisateur* utilisés.

nombre d'exemples est quadratique en fonction de la taille du texte à traiter et le déséquilibre entre classes s'accroît.

La description d'un exemple est ici un vecteur d'attributs portant sur les deux séparateurs du couple. Chaque séparateur du couple est représenté par les tokens se situant à sa gauche et à sa droite, dans une fenêtre de taille T . Chaque token à l'intérieur de ladite fenêtre est codé par un à plusieurs groupes d'attributs.

Le codage de base consiste à représenter chaque token de la fenêtre par un seul attribut indiquant son type. Ainsi un couple de séparateurs est représenté par $4.T$ attributs correspondant aux T tokens à gauche et à droite des deux séparateurs du couple.

Les autres langages expérimentés sont construits par enrichissement de ce codage, en combinant différents groupes d'attributs. Nous présentons maintenant les groupes utilisés.

Deux groupes d'attributs sont fondés sur l'utilisation de dictionnaires.

- Le principe du groupe d'attributs *dictionnaire* est de tester si la valeur du token est présente ou non dans un certain nombre de dictionnaires. Pour ce faire, un attribut booléen par dictionnaire utilisé est introduit pour chaque token de la fenêtre. Les trois dictionnaires de BWI sont disponibles (dictionnaire des mots anglais², dictionnaire des prénoms les plus fréquents aux États-Unis, et dictionnaire de noms de famille³), et il est possible d'utiliser un à plusieurs d'entre eux. Par exemple, si on utilise seulement le dictionnaire anglais, on aura pour chaque token de la fenêtre un attribut `ISEW` qui vaut vrai si le mot est dans le dictionnaire anglais, et faux sinon.
- Le groupe *dictionnaire utilisateur* permet à l'utilisateur de définir un dictionnaire comportant des valeurs (de tokens) qui lui semblent importantes pour trouver l'information à extraire, et qu'on s'attend généralement à trouver dans son voisinage. Un attribut catégoriel est introduit par token. Si le token est dans le dictionnaire, la valeur de l'attribut est celle du token. Sinon l'attribut prend la valeur `autre`. On définit par exemple un dictionnaire `DICotime` contenant les valeurs `Time`, `:`, `at`, `Date` et pour chaque token de la fenêtre un attribut catégoriel `ISinDICotime` qui vaut la valeur du token si elle est présente et `autre` sinon.

Enfin, le groupe *position* ne porte pas sur les tokens de la fenêtre considérée mais est simple-

²/usr/share/dict/words

³disponibles à <http://www.census.gov/genealogy/names/>

ment constitué de deux attributs, l'un indiquant la position, relative à la taille du texte, et l'autre le numéro de ligne du premier séparateur du couple.

Par exemple, si la valeur T de la fenêtre est 2 et si l'on choisit d'utiliser les attributs *position*, *ISEW* et *dictionnaire stime* en plus des attributs du codage de base, le séparateur noté par \updownarrow dans le texte '*Time : \updownarrow 9 :30 PM*' est représenté par le vecteur suivant :

$$(0.31, 4, \text{Alpha}, \text{Vrai}, \text{'Time'}, \text{Ponct}, \text{Faux}, \text{' : '}, \updownarrow, \text{Num}, \text{Faux}, \text{'Autre'}, \text{Ponct}, \text{Faux}, \text{'Autre'})$$

Aux combinaisons des différents attributs s'ajoutent la variation du paramètre T (taille de la fenêtre sur les tokens).

Un langage étant choisi, tout couple de séparateurs (s_d, s_f) présent dans le texte est décrit selon lui et l'exemple associé a une des deux classes : *positif* si s_d identifie le début d'une donnée à extraire et s_f en identifie la fin, *négatif* sinon.

2.2 Apprentissage et programme d'extraction d'information

À partir d'un ensemble de documents marqués, nous sommes donc maintenant en mesure de constituer un échantillon d'apprentissage. Nous avons choisi d'utiliser ensuite C4.5 sur ces données pour obtenir un arbre de décision capable de classer tout nouveau couple comme *positif* ou *négatif*.

Il reste à définir un wrapper basé sur cet arbre de décision. Nous nous plaçons dans le cadre du corpus *Seminar Announcements* où l'extraction d'un champ a au plus une valeur. Le processus est donc le suivant : à partir du texte en entrée, on code les couples de positions dans le langage de description choisi, l'arbre de décision préalablement appris est utilisé pour étiqueter positif ou négatif chacun des couples. Il est alors possible d'avoir plusieurs couples reconnus comme positifs. Pour fournir la donnée extraite, il faut donc choisir un couple parmi ceux classés positif. Nous proposons pour cela les stratégies suivantes :

- *aléatoire* choisit le couple à extraire aléatoirement parmi les propositions de C4.5 ;
- la stratégie *confiance* qui choisit la séquence correspondant à l'exemple identifié comme *positif* par C4.5 avec la plus grande confiance ;
- la stratégie *oracle* qui sélectionne la séquence à extraire si elle est présente et échoue sinon.

Avec la stratégie *aléatoire*, nous mesurons la capacité de C4.5 à identifier des couples positifs, autrement dit sa précision. Elle permet de déterminer une borne inférieure de ce que nous pouvons attendre.

Quand à la stratégie *oracle*, elle n'est pas réellement utilisable puisque elle accède à l'étiquetage des exemples, information qui n'est pas disponible dans le cadre normal d'utilisation du wrapper. Cependant, les performances de cette stratégie traduisent la capacité de C4.5 à détecter les séquences à extraire. Nous obtenons donc, cette fois, une évaluation du rappel de C4.5 et une borne supérieure des performances pour le langage de représentation choisi et l'algorithme utilisé.

TAB. 3 – Effectifs pour les problèmes du corpus Seminar Announcements.

Problème	Positifs	Négatifs	Total
stime	982	1 238 851	1 239 833
etime	433	1 239 400	1 239 833
location	643	2 455 258	2 455 901
speaker	757	1 936 882	1 937 639

3 Expérimentations

Nous utilisons les corpus Seminar Announcements.

3.1 Déséquilibre des problèmes d'apprentissage

Les exemples d'apprentissage sont des couples de séparateurs. Tenir compte de tous les couples de séparateurs des documents du corpus conduirait à 50 millions d'exemples. Afin de réduire le nombre d'exemples, une information supplémentaire est utilisée. Pour chaque problème d'extraction d'information, on observe la plus grande longueur (en nombre de tokens) de la donnée et seuls les couples de séparateurs délimitant des sous-séquences d'au plus cette longueur sont considérés comme des exemples du problème d'apprentissage. Ainsi, toutes les tailles possibles de séquences positives sont représentées, mais le nombre d'exemples négatifs diminue. Cela revient à faire l'hypothèse que la taille d'une donnée à extraire est bornée, et que l'on connaît cette borne à travers les exemples dont nous disposons. Cette hypothèse est raisonnable comparée à celle de BWI qui suppose que toutes les tailles possibles pour la donnée à extraire ont été observées.

Bien que le nombre d'exemples négatifs soit ainsi considérablement réduit, les problèmes de classification restent fortement déséquilibrés. La table 3 présente, pour chaque problème d'extraction d'information, le nombre de couples retenus par notre hypothèse. Ce fort déséquilibre existe de part la nature même du problème mais aussi à cause de notre codage des exemples qui représentent des couples. Dans ce cas, le nombre d'exemples négatifs augmente de manière quadratique avec la taille du texte (linéaire seulement si les exemples représentent des séparateurs comme dans BWI).

Pour remédier à cela, une version modifiée de C4.5 a été utilisée : celle-ci rééquilibre les classes en pondérant les exemples de telle sorte que les sommes des poids sur chaque classe soient égales.

3.2 Protocole expérimental

Le même protocole a été mis en œuvre pour mener toutes les expériences. Nous avons repris le protocole utilisé par BWI (Freitag & Kushmerick, 2000), afin d'avoir des comparaisons pertinentes. Il faut cependant noter que ce protocole est discuté par KUSHMERICK et FREITAG

eux-mêmes (Califf *et al.*, 2004). Par exemple, il n'est pas adapté à l'évaluation d'un système lorsque plusieurs informations sont à extraire.

Le protocole utilisé fait l'hypothèse suivante : il y a au plus une information à extraire par texte, pour chaque champ (ou problème). Ceci est vérifié pour les champs heure de début *stime*, heure de fin *etime* et lieu *location*. Par contre, pour le champ *speaker*, il y a en général plusieurs valeurs à extraire. La mesure utilisée évalue donc le fait d'être capable d'extraire l'un des orateurs. Ceci conduit, bien évidemment, à une surévaluation des performances. Il faut également noter que si l'information à extraire est présente plusieurs fois dans le texte (par exemple l'heure de début), extraire une occurrence suffit.

Pour toutes les expériences, les documents du corpus ont été répartis aléatoirement en dix groupes de validation croisée. Les représentations des séparateurs des neuf groupes de documents utilisés en apprentissage constituent la base d'exemples à partir de laquelle est généré le wrapper. Les performances du wrapper sont ensuite évaluées sur le dixième groupe de documents.

Le Rappel (R) est le pourcentage de données à extraire du corpus qui sont correctement extraites, ce qui met en évidence la proportion d'extractions correctes. La Précision (P) est le pourcentage de données extraites par le wrapper qui sont correctes (*i.e.* qui devaient être extraites), ce qui mesure la qualité des extractions. La F-mesure (F) permet de combiner les deux mesures précédentes : c'est la moyenne harmonique du Rappel et de la Précision. Elle est définie par :

$$F = \frac{2PR}{P + R}$$

Plus formellement, soient TP le nombre d'extractions correctes, FP le nombre d'extraction incorrectes, et FN le nombre de données à extraire qui ne l'ont pas été. On peut alors définir le rappel et la précision par :

$$R = \frac{TP}{TP + FN} \quad \text{et} \quad P = \frac{TP}{TP + FP}$$

Le pseudo-code suivant présente la mise à jour de ces quantités après l'examen d'un texte de l'ensemble de tests. On note `NbPresent` le nombre de séquences à extraire du texte (dans notre cas, 0 ou 1); `NbExtrait` le nombre de séquences extraites par notre wrapper et enfin, `KExtrait` la classe de la séquence extraite (défini uniquement dans le cas où `NbExtrait` vaut un).

```

si (NbPresent = 0)
alors si (NbExtrait > 0)
    alors FP = FP + 1
sinon si (NbExtrait = 0)
    alors FN = FN + 1
    sinon si (KExtrait = 'positif')
        alors TP = TP + 1
        sinon FP = FP + 1
            FN = FN + 1

```

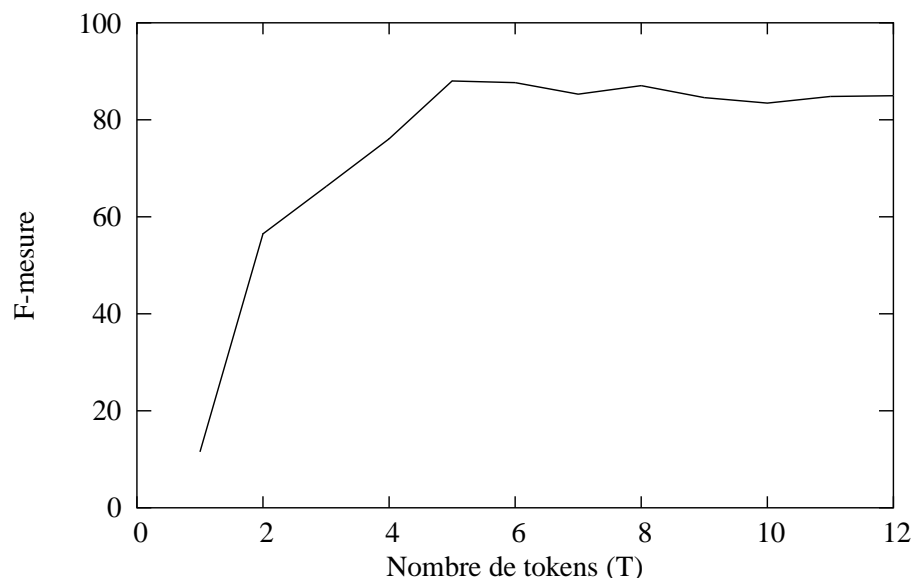


FIG. 3 – Évolution de la F-mesure en fonction de T pour le problème *stime*.

Bien que ce procédé d'évaluation paraisse évident, il est important de remarquer qu'il est souvent omis des articles d'extraction d'information. Ainsi on ne peut savoir comment tel ou tel système d'extraction d'information est évalué. Toute comparaison de performances avec les autres systèmes devient délicate en raison des différences entre les protocoles expérimentaux et méthodes d'évaluation utilisés (Califf *et al.*, 2004).

3.3 Expériences

Nous menons ici des expériences avec l'idée de partir d'un langage simple et d'observer l'évolution des performances du système selon l'enrichissement du langage.

Rôle du paramètre T dans le codage de base

Le codage de base consiste à représenter un couple de séparateurs par $4.T$ attributs prenant leurs valeurs dans l'ensemble des types de tokens. Ces attributs correspondent aux T tokens à gauche et à droite des deux séparateurs. Nous étudions l'évolution des performances mesurées par la F-mesure sur le problème *stime* en fonction de la valeur de T . Les résultats sont présentés dans la figure 3. On constate que la valeur $T = 5$ conduit aux meilleurs résultats avec 88 %, et qu'au delà de cette valeur les courbes se stabilisent autour de 85 %. Dans la suite, le paramètre T est fixé à 5.

TAB. 4 – Codage de base et les stratégies *oracle* et *confiance*, pour $T = 5$.

	Problème	Rappel	Précision	F-mesure
<i>oracle</i>	stime	100.00 %	100.00 %	100.00 %
	etime	98.25 %	75.17 %	85.17 %
	location	66.16 %	67.47 %	66.81 %
	speaker	70.90 %	59.92 %	64.95 %
<i>confiance</i>	stime	88.01 %	88.01 %	88.01 %
	etime	88.52 %	67.73 %	76.74 %
	location	46.79 %	47.71 %	47.24 %
	speaker	31.34 %	26.48 %	28.71 %

TAB. 5 – Codage de base ($T = 5$) et stratégie *confiance* et avec les attributs *position*.

Problème	Rappel	Précision	F-mesure
stime	88.52 %	88.52 %	88.52 %
etime	89.99 %	69.08 %	78.16 %
location	44.58 %	48.10 %	46.28 %
speaker	28.96 %	24.83 %	26.74 %

Pertinence du codage de base

Nous souhaitons mesurer le pouvoir de prédiction du codage de base. Pour cela, il nous faut vérifier que les informations à extraire sont effectivement classées comme positives après apprentissage. Nous utilisons donc la stratégie *oracle*. Les performances, pour les quatre problèmes d'extraction d'information du corpus **Seminar Announcements** sont présentées dans la table 4. On constate qu'avec un codage aussi simple, on obtient des performances proches de celles de BWI (voir la table 1), cela bien sûr sous réserve que l'on sache choisir « correctement » le couple à extraire parmi les propositions de C4.5. Malheureusement, ce contrat n'est pas tout à fait rempli par les valeurs de confiance fournies par C4.5, comme le montrent les résultats de la stratégie *confiance*, présentés dans la table 4.

Enrichissement du langage de base

On ajoute les deux attributs de la famille *position*. Les résultats sont présentés dans la table 5. On constate une légère amélioration sur les problèmes *stime* et *etime*, et une dégradation sur les deux autres. Cela est certainement dû au fait que les données *stime* et *etime* sont souvent à des positions fixes dans les documents, alors que les données *speaker* et *location* n'ont pas de position privilégiée. Par la suite, les attributs *position* ne seront utilisés qu'avec les problèmes *stime* et *etime*.

TAB. 6 – Ajout des dictionnaires anglais, des noms et des prénoms et de l'utilisateur.

Problème	Rappel	Précision	F-mesure
stime	93.49 %	93.49 %	93.49 %
etime	89.91 %	83.00 %	86.32 %
location	52.55 %	56.44 %	54.43 %
speaker	51.50 %	49.67 %	50.57 %

Introduction de connaissances spécifiques au domaine

Les dictionnaires disponibles sont :

- dictionnaire des mots anglais ;
- dictionnaire des noms propres ;
- dictionnaire de l'utilisateur.

Pour les problèmes *speaker* et *location* nous avons utilisé tous les dictionnaires. Pour les problèmes *stime* et *etime* seuls les dictionnaires *mots anglais* et *utilisateurs* ont été utilisés.

L'utilisation des dictionnaires permet d'améliorer les performances de manière significative. Les résultats sont présentés dans la table 6. On constate une augmentation de la F-mesure de 10 points pour *location*, de 24 points pour *speaker*, 5 points pour *stime* et 8 points pour *etime*. L'augmentation très forte sur *speaker* est principalement due à l'utilisation des noms et prénoms. De façon générale, cette augmentation de performances n'est pas surprenante mais montre bien les biais qui peuvent exister lorsque l'on compare des systèmes d'extraction.

Une borne supérieure

Le codage final est le codage utilisant tous les groupes d'attributs (sauf *position* pour *stime* et *etime*). La table 7 montre que ce codage permettrait d'atteindre les performances de BWI à partir du moment où l'on sait choisir correctement la donnée à extraire parmi les propositions. On constate que sur les problèmes *stime*, *etime* et *speaker* le codage final est au moins aussi bon que BWI.

4 Discussion et Perspectives

Dans cet article, nous avons proposé de dissocier le langage de description et les connaissances du domaine de l'algorithme d'apprentissage.

Nous avons montré qu'un codage attribut-valeur simple associé à un algorithme de classification supervisée classique permet d'obtenir un système d'induction de wrapper ayant des performances intéressantes. De plus, l'introduction de connaissances spécifiques au domaine est aisée à réaliser avec notre codage et améliore les résultats.

À court terme, nous souhaitons étendre ce travail aux documents HTML et XML, en enrichissant le langage de description avec des informations de structure.

TAB. 7 – Codage final et l'aide de l'oracle.

	Problème	Rappel	Précision	F-mesure
<i>codage final et oracle</i>	stime	100.00 %	100.00 %	100.00 %
	etime	98.68 %	91.09 %	94.74 %
	location	68.32 %	73.38 %	70.76 %
	speaker	76.77 %	74.06 %	75.39 %
BWI	stime	99.6 %	99.6 %	99.6 %
	etime	94.9 %	94.4 %	94.6 %
	location	69.6 %	85.4 %	76.7 %
	speaker	59.2 %	79.1 %	67.7 %

Par ailleurs, on a vu que la valeur de confiance de C4.5 n'est pas entièrement satisfaisante elle ne correspond en effet qu'à une probabilité estimée d'arriver dans une feuille de l'arbre. Nous envisageons donc d'utiliser d'autres algorithmes générant des valeurs de confiance meilleures que celles de C4.5. Un candidat peut être un algorithme de boosting sur les arbres de décision à un noeud (boosted stumps). Cela s'intègre dans la perspective plus large de faire varier l'apprenant sur des langages de description finés.

Remerciements

Ces travaux de recherche ont été soutenus par :

- « CPER 2000-2006, Contrat de Plan état - région Nord/Pas-de-Calais : axe TACT, projet TIC » ;
- « ACI masse de données – ACI-MDD – Accès au Contenu Informationnel pour les Masses de Données et Documents ».

Références

- CALIFF M. E., CIRAVEGNA F., FREITAG D., GIULIANO C., KUSHMERICK N., LAVELLI A. & ROMANO L. (2004). A critical survey of the methodology for i.e evaluation. In *Proceedings of LREC 2004*. To appear.
- CHIEU H.-L. & NG H.-T. (2002). A maximum entropy approach to information extraction from semi-structured and free text. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI 2002)*, p. 786–791.
- COHEN P. R. & FEIGENBAUM E. A. (1982). *The Handbook of Artificial Intelligence*, volume 3. HeurisTech Press and William Kaufmann.
- CORNUÉJOLS A. & MICLET L. (2002). *Apprentissage artificiel*. Eyrolles.
- FREITAG D. & KUSHMERICK N. (2000). Boosted wrapper induction. In *AAAI/IAAI*, p. 577–583.
- FREITAG D. & MCCALLUM A. (1999). Information extraction with HMMs and shrinkage. In *Proceedings of the AAAI-99 Workshop on Machine Learning for Information Extraction*.

- FREITAG D. & MCCALLUM A. (2000). Information extraction with HMM structures learned by stochastic optimization. In *AAAI/IAAI*, p. 584–589.
- FREUND Y. & SCHAPIRE R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, **55**(1), 119–139.
- HSU C.-N. & DUNG M.-T. (1998). Generating finite-state transducers for semi-structured data extraction from the web. *Information Systems*, **23**(8), 521–538.
- HUFFMAN S. B. (1995). Learning information extraction patterns from examples. In *Learning for Natural Language Processing*, p. 246–260.
- KAUCHAK D., SMARR J. & ELKAN C. (2002). Sources of success for information extraction methods. Technical Report No. CS2002-0696, January, UCSD.
- KOSALA R. & BLOCKEEL H. (2000). Instance-based wrapper induction.
- KOSALA R., DEN BUSSCHE J., BRUYNNOOGHE M. & BLOCKEEL H. (2002). Information extraction in structured documents using tree automata induction.
- KUSHMERICK N. (1997). *Wrapper Induction for Information Extraction*. PhD thesis, University of Washington.
- QUINLAN J. R. (1993). *C4.5 : Programs for Machine Learning*. Morgan Kaufmann.
- SAKAMOTO H., MURAKAMI Y., ARIMURA H. & ARIKAWA S. (2001). Extracting partial structures from HTML documents. In H. SAKAMOTO, H. ARIMURA & S. ARIKAWA, Eds., *Proc. the 14th International FLAIRS Conference*, p. 264–268 : AAAI Press.
- SCHAPIRE R. F. & SINGER Y. (1998). Improved boosting algorithms using confidence-rated predictions. In *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT-98)*, p. 80–91, New York : ACM Press.
- SEYMORE K., MCCALLUM A. & ROSENFELD R. (1999). Learning hidden Markov model structure for information extraction. In *AAAI 99 Workshop on Machine Learning for Information Extraction*.
- SODERLAND S., FISHER D., ASELTINE J. & LEHNERT W. (1995). CRYSTAL : Inducing a conceptual dictionary. In C. MELLISH, Ed., *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, p. 1314–1319, San Francisco : Morgan Kaufmann.