
**Universidade Federal de Pernambuco**

Pós-Graduação em Ciência da Computação

# EXTRACTION D'INFORMATION ADAPTATIVE DE PAGES WEB PAR INDUCTION SUPERVISÉE D'EXTRACTEURS

Rinaldo José de Lima

DISSERTATION DE MASTER


Universidade Federal de Pernambuco
www.cin.ufpe.br/~posgraduacao

RECIFE, AGOSTO DE 2009


UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Rinaldo José de Lima

# EXTRACTION D'INFORMATION ADAPTATIVE DE PAGES WEB PAR INDUCTION SUPERVISÉE D'EXTRACTEURS

Ce mémoire a été présenté au Programme de Post-Graduation en Sciences de l'Informatique du « Centro de Informática de l'Universidade Federal de Pernambuco » comme exigence partielle pour l'obtention du grade Master en Sciences, specialité Informatique.

Directeurs: Prof. Frederico Luiz Gonçalves Freitas, Phd et
Prof. Bernard Espinasse, Phd.

Prof. Jacques Robin, PhD.
Membre du Jury

Prof. Evandro de Barros Costa, PhD.
Membre du Jury

RECIFE, JUNHO/2009

**Lima, Rinaldo José de**
**Extraction d'information adaptative de pages web par induction supervisée d'extracteurs / Rinaldo José de Lima - Recife : O Autor, 2009.**
**xx, 89 p. : il., fig., tab.**

**Dissertação (mestrado) – Universidade Federal de Pernambuco. CIn. Ciência da Computação, 2009.**

**Inclui bibliografia e apêndice.**

**1. Inteligência artificial. I. Título.**

**006.3 CDD (22. ed.) MEI2009- 070**

Dissertação de Mestrado apresentada por **Rinaldo José de Lima** Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título, ***"Extraction d'information Adaptative de Pages Web par Induction Supervisée d'exfracteurs"***, orientada pelo **Prof. Frederico Luiz Gonçalves de Freitas** e aprovada pela Banca Examinadora formada pelos professores:

Prof. Jacques Pierre Louis Robin
Centro de Informática / UFPE

Prof. Evandro de Barros Costa
Departamento de Tecnologia da Informação / UFAL

Prof. Frederico Luiz Gonçalves de Freitas
Centro de Informática / UFPE

Visto e permitida a impressão.
Recife, 16 de junho de 2009.

**Prof. FRANCISCO DE ASSIS TENÓRIO DE CARVALHO**
Coordenador da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.

*À ma mère Maria Lima et ma fille Rebeca Lima.*
*Les deux jeunes filles les plus importantes du monde pour m*oi.

# Remerciements

Cette dissertation est l'aboutissement d'un stage de recherche développé par l'auteur pour conclure son projet de Master of Sciences dans le cadre des projets MasterWeb/AGATHE et Click&Go. Ce dernier a financée cette recherche qui a été menée en accord avec l'Université Fédérale de Pernambuco (UFPE) à Recife, sous la co-orientation des Professeurs Frederico FREITAS, du Centre d'Informatique de l'UFPE ; et Bernard ESPINASSE, du Laboratoire des Sciences de l'Information et des Systèmes (LSIS) à Marseille (UMR CNRS 6168).

Je tiens à remercier mon cotuteur brésilien, Prof. Frederico Freitas, pour toutes les leçons apprises, son soutien tout au long de cette étude, et l'opportunité qu'il m'a donné pour la réalisation de ce travail.

Je tiens à remercier vivement mon cotuteur français, Prof. Bernard Espinasse, pour son soutien, ses conseils éclairés, son suivi quotidien qui m'ont été d'une aide précieuse dans l'évolution de mes travaux. Qu'il trouve ici l'expression de ma profonde gratitude.

Un grand merci au Prof. Sébastien Fournier et Shereen Bitar pour de nombreuses discussions qui n'ont permis de voir plus clair le contexte de notre sujet de recherche.

Je remercie mon ami, Luciano Cabral, que j'ai connu pendant le temps que nous avons durement travaillé ensemble au Centre d'Informatique (CIN) à Recife, et qui a beaucoup participé dans la conception de ce travail.

Je souhaite présenter mes remerciements aux membres du jury qui ont accepté de participer à ma soutenance.

Je remercie également mon ami de longues années, Gerson Henrique, qui m'a encouragé et soutenu tout au long de cette recherche.

Merci spécial à Roberta Costa pour avoir eu la patience d'écouter mes hésitations, pour les conseils et les mots d'encouragements qui m'ont vraiment faire avancer.

Je tiens à remercier Claudia Serey, ma seule amie brésilienne à Marseille qui, dès mon premier jour au LSIS, m'a aidé plusieurs fois dans mon intégration à la vie universitaire, et son époux Mathieu Detraux, pour les bons moments que nous avons tous vécus ensemble.

Finalement, je remercie tout particulièrement la merveilleuse famille Espinasse (Bernard, Sabine, Sabrina, Vanessa et Valentin) qui m'ont chaleureusement accueilli.

*« Je ne sais pas ce que je peux paraître aux yeux du monde, mais pour moi-même,*
*il me semble que je n'ai été qu'un enfant jouant sur le rivage,*
*et s'y amusant à  trouver de temps en temps,*
*un galet plus lisse ou un coquillage plus beau que les autres,*
*tandis que le grand océan de la vérité s'étendait, encore inconnu, devant moi.»*

Isaac Newton, à la fin de ces jours vers 1727.

# Resumo

A Extração de Informação (EI) compreende técnicas e algoritmos que realisam duas tarefas importantes: a identificação de informações desejadas a partir de documentos estruturados e não-estruturados, e o armazenamento dessas informações em um formato apropriado para uso futuro. Este trabalho concentra-se nos sistemas d'EI adaptativos que podem ser customizados para novos domínios através de um processo de treinamento (*Machine Learning*) usando coleções de documentos anotados como entrada. Particularmente, técnicas de *induçao automática de wrappers* são estudadas para extração de informação que se baseiam na exploração de regularidades estruturais encontradas em documentos Web. Wrappers são procedimentos para extrair dados de documentos. A indução de wrappers é definida como uma técnica de extração de informação que usa algoritmos de aprendizado de máquina para automaticamente construir wrappers a partir de um *corpus* previamente anotado e que tem mostrado bons resultados quando aplicada em textos estruturados, semi-estruturados e livres (em linguagem natural). Este trabalho propõe um sistema d'EI baseado em *Boosted Wrapper Induction* (BWI), um algoritmo de indução de wrappers supervisionado no qual um outro algoritmo, o *AdaBoost*, é usado para gerar um procedimento genérico de extração que combina, no final do processo, um conjunto de wrapers específicos por voto ponderado. Alguns autores tem estudado como as técnicas de *boosting* contribuem ao sucesso do algorithmo BWI e examinado sua perfomance tomando a direção desafiadora de usá-lo como um método de extração de informação para documentos não-estruturados em linguaguem natural. Este fato foi a principal motivação para se incluir *Parts-of-Speech (POS) tagging* na fase de pré-processamento do sistema d'EI ora proposto. Afim de se avaliar o desempenho do sistema, vários experimentos foram executados usando-se três *corpora* como *testbed* para a tarefa de extração de informação no preenchimento de esquemas de extração (*template filling task*). Outros experimentos foram também conduzidos usando-se diversas combinações de atributos para sistematicamente avaliar os efeitos que esses últimos têm no desempenho do algoritmo de aprendizado. Os resultados obtidos experimentalmente mostraram que o desempenho geral do sistema proposto é comparável a outros sistemas de EI do estado da arte.

**Palavras-chave**: Extração de Informação, Indução de Wrappers, Boosting, Classifiação Supervisionada, POS Tagging, Aprendizagem de Máquina.

# Résumé

Extraction d'Information (EI) comprend des techniques et algorithmes réalisant deux tâches importantes: l'identification des informations désirées, pertinentes à partir de documents structurés ou non structurés, et le stockage de ces informations sous une forme appropriée visant l'usage future. Ce présent travail se concentre sur les systèmes d'EI adaptatifs qui peuvent être appliqués sur de nouveaux domaines par l'apprentissage artificiel (machine learning) en utilisant une collection de documents en entrée. En particulier, des techniques d'induction automatique d'extracteurs sont examinées pour l'extraction d'information qui repose sur l'exploitation de régularités structurales présentes dans ces documents. Wrappers (extracteurs) sont définis comme des procédures pour l'extraction d'informations d'un document quelconque. L'induction d'extracteurs est une technique qui utilise des algorithmes d'apprentissage automatique pour la conception d'extracteurs à partir d'un corpus préalablement annoté, et qui donne de bons résultats lorsqu'ils sont appliqués sur des documents structurés, semi-structurés et en langage naturel (libre). Nous proposons dans ce travail un système d'EI par induction supervisée d'extracteurs reposant sur l'algorithme *Boosted Wrapper Induction* (BWI) dans lequel l'algorithme d'AdaBoost est employé pour générer une procédure d'extraction générique qui combine un ensemble d'extracteurs spécifiques par vote pondéré. D'autres auteurs ont étudié comment la technique de boosting contribue au succès de l'algorithme de BWI et ont examiné sa performance vers la direction ambitieuse de l'employer comme méthode d'IE pour les documents en langage naturel. Ce fait a motivé l'inclusion d'étiquetage POS (Parties du Discours) dans le prétraitement dans notre système des documents. Afin d'évaluer la performance de ce système, plusieurs expériences ont été menées sur trois *corpora* désignés pour la tâche d'extraction d'information classique par slot. D'autres expériences ont été également réalisées en utilisant plusieurs combinaisons d'attributs avec l'objectif d'étudier systématiquement leurs effets sur la performance de l'algorithme d'apprentissage. Les résultats obtenus empiriquement ont montré que les performances de notre système étaient comparables à d'autres systèmes de l'état de l'art.

**Mots-clés:** Extraction d'information, Induction d'extracteurs, Classification Supervisée, Parties du Discours, Apprentissage Machine.

# Abstract

Information Extraction (EI) comprises techniques and algorithms performing two important tasks: identifying the desired, relevant information from structured or non-structured documents and storing it in appropriate form for future use. This work is focused on adaptive IE systems that can be customized for new domains through training (machine learning) using annotated *corpora* as input. Particularly, automatic *wrapper induction* techniques are looked into for extraction that rest on the exploitation of structural regularities present in documents. Wrappers are procedures to extract data from information resources. Wrapper induction is defined as a technique that uses machine learning algorithms for automatically construct wrappers from a previously annotated corpus and that has shown good results when applied to structured, semi-structured and free (natural language) documents. This work proposes a supervised IE system based on Boosted Wrapper Induction (BWI), a supervised wrapper induction algorithm, in which the AdaBoost algorithm is used to generate a general extraction procedure that combines a set of specific wrappers by weighted voting. Some others authors have investigated how boosting contributes to the success of the BWI algorithm and examined its performance in the challenging direction of using it as an IE method for unstructured natural language documents. This fact became the rationale for including *Parts-of-Speech* (POS) tagging in the preprocessing phase of the proposed IE system. In order to evaluate the performance of the system, several experiments were carried out on three standard *corpora* for the template filling task. Other experiments were also conducted using various combinations of features in order to systematically study their effects on the performance of the learning algorithm. Results obtained experimentally showed that the perfomance of the proposed system is comparable with other state-of-the-art IE systems.

**Keywords**: Information Extraction, Wraper Induction, Boosting, Supervised Classification, Part-of-Speech Tagging, Machine Learning.

# Table de Matières

# Table de Figures

# Liste de Tableaux

# 1 INTRODUCTION

Une grande quantité d'informations sur divers sujets en différents formats numériques sont, de façon croissante, publiées sur le Web chaque jour. Cette croissance a été stimulée par le progrès technologique en informatique simplifiant la production, le stockage et la distribution de l'information sur le Web. Beaucoup de ces informations stockées de façon non-structurée, sont éparpillées sur des milliers d'ordinateurs individuels (*hosts*) constituant ainsi une complexe et immense base de données de portée mondiale.

Cela limite fortement l'exploitation de ces informations, et des techniques de navigation ou des recherches par mot clés se montrent inefficaces quand l'utilisateur veut trouver des informations précises sur le Web. Ces techniques sont inefficaces car elles ramènent souvent une grande quantité de document inutiles ou, dans le pire des cas, des documents pertinents ne sont mêmes pas trouvés [Aldea et al., 2003].

L'extraction d'informations dans ces documents ramenés est réalisée actuellement essentiellement par des humains. Il a été déjà argumenté que des techniques autour du format XML et ses variantes [Bray et al. 2008] ne seraient pas suffisantes pour accroître l'efficacité des moteurs de recherche existants sur Web [Espinasse et al., 2007]. Ainsi, bien que XML puisse être utilisé dans la structuration de l'information sémantique des informations, il y a toujours l'héritage de téraoctets de documents qui ne seront probablement pas convertis à ce format. [Kushmerick & Thomas, 2003] affirme que c'est impossible de déterminer un schéma d'annotation parfait qui intègre différentes sources d'informations et qui fonctionne correctement pour une grande variété d'applications. Ces problèmes alors ont stimulé la recherche de solutions qui donneraient à l'Internet le même pouvoir de consultation à des données structurées trouvées dans des bases de données actuelles.

Dans ce contexte, l'Extraction d'Information (EI) consiste d'un moyen d'obtenir et d'intégrer les données contenues dans une collection de documents d'un même domaine et, les systèmes d'IE n'ont pas pour objectif de comprendre les textes traités, mais de réduire les informations textuelles (éventuellement non structurées qui y figurent) aux structures tabulaires de manipulation plus facile [Kushmerick & Thomas, 2003]. Ainsi, il est possible d'extraire automatiquement des informations textuelles depuis des dépôts numériques, tels que les descriptions et les prix des produits dans les magasins virtuels et de les utiliser pour construire la base de données pour des analyses et comparaisons ultérieures.

Diverses recherches ont été menées pour développer des systèmes d'IE adaptés à différents types de textes, par exemple, en allant des pages HTML rigidement structurées créées à partir de bases de données [Chang & Lui, 2001] jusqu'aux des *Call for Papers* (CFP) [Ireson & Ciravegna, 2005] écrites en langage naturel. Pour ce second type de texte, en particulier, de nombreux travaux ont été développés qui montrent que de tels systèmes sont difficiles à développer et exigent un investissement important de spécialistes dans le domaine de l'application et d'EI [Siefkes & Siniakov, 2005].

De ce fait, de nombreuses recherches ont été déjà menées dans le développement de systèmes d'EI de plus en plus adaptables aux domaines les plus divers [Tang et al., 2007]. Ces systèmes d'EI adaptatifs reposent sur induction de *wrappers* (extracteurs). Un extracteur, dans le contexte d'EI, est un programme capable d'extraire des informations à partir d'un ensemble de documents (*corpus*). Ainsi, l'induction d'extracteurs utilisent des algorithmes d'apprentissage machine pour la génération de règles d'extraction à partir d'un ensemble de

documents préalablement étiquetés/annotés à une étape d'apprentissage, au lieu d'être apprises manuellement par un ingénieur de la connaissance [Kushmerick, 1997, 2000].

Un des pionniers dans l'étude et le développement de techniques d'induction d'extracteurs est le Prof. Daniel Freitag qui, en 2000, a conçu un système d'EI adaptatif reposant sur la classification supervisée comme technique de base pour l'induction d'extracteurs. Sa technique d'induction d'extracteurs, aussi appelé *Boosted Wrapper Induction* (BWI) [Freitag & Kushmerick, 2000] induisait des extracteurs d'information destinées à être appliqués sur des documents semi-structurés comme XML et des pages HTML où leur système de balises internes leur rend un certain dégrée de structuration.

Plus tard, divers travaux notamment [Kauchak et al., 2002] et [Marty & Torre, 2004] expliquent les bons résultats obtenus par le système BWI, plus précisément par les algorithmes qu'il met en œuvre. Cependant, dans le premier travail, les auteurs ont analysé comment les composants algorithmiques du BWI contribuaient à son succès. Ils ont mis en évidence que la technique de *boosting* [Freund & Schapire, 1990] est l'élément principal de la réussite de BWI. Grace à cette technique, l'algorithme d'apprentissage du BWI est capable de faire la repondération d'exemples afin d'apprendre des règles spécifiques (conduisant à des résultats plus précis).

De plus, Kauchak et al. [Kauchak et al., 2002] ont évalué le système BWI original sur des collections de documents classifiées en trois groupes : non structurés (en langage naturel), partiellement structurés et fortement structurés. Ils l'ont également expérimenté sur une collection de documents obtenue de la Libraire Nationale de Médicine (MEDLINE) et annotés avec des indications de *types de segment de phrases* ; par exemple, segment de phrases prépositionnelles, phrases nominaux, etc. De ce fait, ils sont arrivés à la conclusion que, même avec des informations grammaticales limitées, le système pourrait avoir une sensible amélioration dans les résultats.

C'est dans ce contexte que les objectifs et les contributions du présent travail consisteront à :

- étudier l'utilisation de techniques d'induction d'extracteurs en mettant l'accent sur l'algorithme d'induction d'extracteurs BWI afin de réaliser l'extraction d'information de documents de différents niveaux de structuration et dans différents domaines ;
- analyser l'influence de l'information morphosyntaxique (Parties du Discours) - en suivant l'axe des études proposées par [Kauchak et al., 2002] (*induction d'extracteurs* plus *informations grammaticale*s) - , sur les extractions faites par l'algorithme d'induction d'extracteurs BWI en utilisant une collection de documents en langage naturel pour évaluer quantitativement le gain que cette information peut apporter à l'algorithme ;
- proposer une architecture logicielle reposant sur l'induction d'extracteurs supervisée pour l'EI à partir de pages Web. Cette architecture logicielle sera présentée ainsi que les différents modules spécialisés qui la compose, dont le module central mettra en œuvre l'algorithme d'apprentissage BWI et qui sera modifié pour la prise en compte du traitement du langage naturel (annotation morphosyntaxique). Cela aboutira à l'environnement d'EI adaptatif M-TIES, facilement configurable et convivial à l'utilisateur par le biais d'une architecture modulaire destiné à l'utilisation sur plusieurs types de documents (structurés et non structurés) ;

- comparer l'architecture d'EI proposée dans cette recherche avec d'autres systèmes de l'état de l'art au travers différentes expérimentations sur trois collections de documents de référence en suivant une rigoureuse méthodologie d'évaluation de résultats bien établie dans la communauté scientifique du domaine de l'EI. Pour cela, il sera défini un protocole expérimental pour bien mener les expériences afin d'avoir de résultats plus fiables et pertinents.
- utiliser le potentiel d'induction d'extracteurs dans le contexte d'une extraction d'information intégrée - plus précise et fine - afin d'augmenter la performance du sous-système d'extraction de l'architecture MasterWeb/AGATHE [Espinasse et al., 2007; Freitas et al., 2008] (voir section 1.1). Suite à cette recherche, il sera envisagé de combiner la tâche symbolique du système MasterWeb/AGATHE qui réalise actuellement une classification des pages Web à base d'ontologies, avec une tâche d'extraction d'information adaptative, permettant d'extraire de l'information sur ces pages Web classées, ceci par l'usage de techniques d'apprentissage artificiel (machine learning) et traitement de langage naturel. Plus précisément, il est attendu une amélioration de la performance des sous-systèmes d'extraction de ces architectures en les dotant des techniques d'induction automatique d'extracteurs d'information et de traitement de langage naturel à travers l'utilisation de l'architecture d'EI proposée par ce travail.

## 1.1 Contexte : Systèmes de collecte d'information sur des domaines restreints – MasterWeb/AGATHE

Le système AGATHE [Espinasse et al., 2007; Freitas et al., 2008] est une architecture logicielle générique permettant le développement de systèmes de collecte d'information sur le Web sur un ou plusieurs domaines restreints. AGATHE met en œuvre une collecte coopérative d'information à base d'agents logiciels et d'ontologies. Ce système prend en compte des contextes de recherche en considérant des regroupements de pages Web relatifs à des domaines spécifiques (par exemple la recherche académique, le tourisme, …).

L'objectif de l'architecture AGATHE est de permettre le développement de systèmes de collecte d'information sur le Web sur des domaines restreints pouvant être progressivement étendus. Pour le développement d'AGATHE, le domaine restreint de recherche de départ est celui de la recherche académique, plus précisément la tenue d'événements scientifiques (conférences ou workshops internationaux).

L'architecture modulaire générale du système est illustrée par la Fig. 1. Trois sous-systèmes en interaction réalisent les différentes tâches :

• Le *Sous-système de Recherche* (SSR) : Il est responsable de l'interrogation des moteurs de recherche externes sur le web (comme Google). Il envoie les pages récupérées vers le sous-système d'extraction qui va en extraire les informations pertinentes.

• Le *Sous-système d'Extraction* (SSE) : Il consiste en plusieurs clusters d'extraction spécialisés dans les différents domaines. Chaque cluster réalise la validation des pages web, leur classification et l'extraction d'informations à partir de ces pages en fonction d'un domaine spécifique (la recherche académique, le tourisme, etc.).

• Le *Sous-système d'Utilisation (SSU)* : Il stock les informations extraites envoyées par le sous-système d'extraction, et fournit une interface permettant aux utilisateurs d'exécuter les requêtes sur les données stockées.

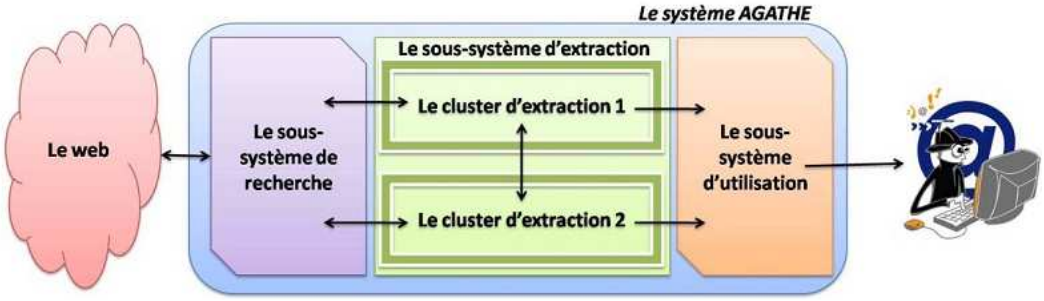



**Fig. 1.** L'architecture générique d'AGATHE.

Cette architecture logicielle tire profit du génie logiciel orienté agents afin d'assurer flexibilité et réutilisabilité. Le point de départ de cette architecture est un prototype déjà réalisé, le système MASTER-Web (*Multi-Agent System for Extraction and Retrieval over the Web*) [Freitas et al., 2000, 2001, 2003]. Ce dernier propose un agent logiciel unique qui utilise des ontologies pour réaliser des tâches de classification et d'EI sur le Web, ceci sur un seul domaine de recherche.

Un des problèmes auxquels les architectures de MasterWeb/AGATHE à dû faire face dans le développement de systèmes de collecte d'informations sur des domaines restreints, est celui du problème classique de la spécification, de l'écriture et de la mise-à-jours des base de règles, en particulier, dans l'étape d'extraction d'information. Par conséquent, afin de faciliter l'élaboration de telles règles en Jess[1], un éditeur intelligent qui aide l'utilisateur à rédiger des règles en langage Jess a été développé. Cet éditeur respect la syntaxe spécifique de ces règles et permet de se référer facilement à un élément de l'ontologie présent dans une règle, par une opération de glisser-déposer.

Cependant l'écriture de règles symboliques pour des tâches d'extraction reste très difficile et fastidieuse, même si on utilise ces outils pour les écrire. Une autre limitation est que les architectures des systèmes MasterWEB/AGATHE ne sont pas munies de techniques de traitement du langage naturel.

Par conséquent, en outre les objectifs et contributions majeurs déjà mentionnés, il est envisagé d'intégrer, dans le système MasterWeb/AGATHE, de techniques symboliques d'EI reposant sur l'apprentissage artificiel (machine learning). Plus précisément, il est attendu une amélioration des performances du sous-système d'extraction de ces architectures en les dotant des techniques d'induction automatique d'extracteurs d'information et de Traitement de Langage Naturel (TAL).

## 1.2 Structure du mémoire

Après ce chapitre introductif, le chapitre 2 donne un aperçu de la tâche d'extraction d'information à partir de différents types de documents Web, en expliquant notamment comment représenter ces documents pour en extraire, par programme, de l'information en tenant compte ou non de traitement de la langue naturelle. Ensuite, différents approches pour la conception d'extracteurs et une méthodologie traditionnelle d'évaluation de ces types de systèmes par trois mesures classiques d'évaluation de performance sont présentées.

Le chapitre 3 discute sur les fondements de l'approche d'extraction d'information retenue par cette recherche, où l'induction d'extracteurs est vue comme un problème de Classification Supervisée (CS). Il présente à la fois les techniques de CS utilisées dans ce projet et quelques systèmes d'induction d'extracteurs qui utilisent des techniques semblables en se limitant au cas de l'extraction unaire sur les documents non structurés avec prise en compte ou non de la syntaxe de la langue naturelle. De l'étude de ces systèmes existants pour cette tâche spécifique, il est dégagé une architecture logicielle et une démarche génériques adaptées au problème d'extraction d'information traité comme un problème de classification.

Le chapitre 4 décrit la principale contribution de cette recherche qui est la proposition d'un système d'EI reposant sur l'induction d'extracteurs supervisé pour l'EI à partir de pages Web. L'architecture logicielle de ce système est présentée ainsi que les différents modules spécialisés qui la compose, dont le module central TIES (*TIES – Trainable Information Extraction System*) développé par IRST de Trento, met en œuvre l'algorithme d'apprentissage BWI [Freitag & Kushemerick, 2000] et qui a été modifié pour la prise en compte du traitement du langage naturel (annotation morphosyntaxique).

Le chapitre 5 présente les *corpora* utilisés pour évaluer notre architecture logicielle d'EI. Il s'agit de trois corpora de niveaux de structuration différents : le premier corpus est structuré, le deuxième est semi-structuré, et le dernier composé de textes en langage naturel

[1] Jess (*Java Expert System Shell*) [Friedmann-Hill 97] est un outil pour le développement de règles de production (systèmes experts) fondé sur un moteur d'inférence qui emploie l'algorithme Rete [Forgy 82].

(libre). Ensuite, un protocole expérimental reposant sur une méthodologie d'évaluation rigoureuse est mis en place afin d'examiner l'influence des paramètres sur l'algorithme BWI, et analyser l'effet de la prise en compte de la syntaxe du langage naturelle sur trois collections de documents. Finalement, il est mené une évaluation comparative du système proposé avec d'autres systèmes d'EI étudiés dans l'état de l'art.

Enfin, le chapitre 6 conclut en faisant un bilan de ce travail de recherche en présentant plusieurs perspectives possibles.

# 2 EXTRACTION D'INFORMATION

Parmi les plusieurs définitions de l'extraction d'information (EI) on rencontre, d'une part, celle qui affirme que l'EI consiste à extraire de la connaissance de documents [Crespo et al., 1994]. D'autre part, [Pazienza, 1997] définit l'EI en la comparant avec la recherche d'information (RI) où la dernière consiste à trouver un ensemble de documents pertinents, tandis que la première consiste à trouver, dans ces documents, un ensemble de faits pertinents. Les documents dont on souhaite extraire de l'information peuvent présenter un certain degré dans la structuration des informations, mais ils peuvent aussi n'en présenter aucun.

En outre, [Eikvil, 1999] déclare que les systèmes d'EI n'essayent pas de comprendre le texte dans les documents d'entrée, mais plutôt d'analyser des portions de chaque document qui contiennent des informations pertinentes. La pertinence est déterminée par des schémas prédéfinis du domaine qui spécifient quel type d'information le système s'attend à trouver.

Ces définitions sont très générales car elles ne spécifient ni le type des documents d'entrée, ni la nature des éléments extraits. En revanche, [Florescu et al., 1998] propose une définition plus précise et focalisée sur Web où les pages d'un site Web sont considérées comme des conteneurs de données. Alors, l'EI consiste à produire une représentation structurée de ces données. La définition trouvée en [Cohen et al., 2003] est également spécifique au Web et appui l'EI sur l'usage d'*extracteurs*. Un extracteur est défini comme étant un programme qui permet de voir un site Web existant comme une base de données. Cette dernière définition est plus spécifique que les autres, tant au niveau du type des documents d'entrée, ici limités aux pages Web, qu'au niveau du type de la sortie, une base de données.

En comparant toutes les définitions évoquées ci-dessus, on distingue 3 caractéristiques majeures partagées : (1) l'EI a comme entrée un ensemble de documents d'un certain type ; (2) tels documents contiennent des informations ou données que l'on veut extraire et (3) la sortie d'une tâche d'EI est structurée et définie selon un schéma (*template*).

En d'autres termes, l'EI a pour objectif de construire des systèmes qui trouvent et combinent des informations pertinentes tandis qu'ils ignorent des informations insignifiantes et inutiles [Cowie & Lehnet, 1996]. Ils modélisent une fonction qui reçoit un document d'entrée et retourne un formulaire de sortie, préalablement définie, avec leurs champs remplis. De cette façon, des informations spécifiques peuvent alors être extraites de différents documents avec une représentation hétérogène et peuvent être résumées et présentées en un format fixé à l'avance. Un exemple d'une telle tâche est illustré par la Fig. 2. Dans cet exemple, le document d'entrée est un article scientifique et le formulaire (*template*) de sortie se compose des champs *auteurs* et *emails*.

Il est important de noter que des informations extraites sont déterminées par un ensemble de patrons ou de règles d'extraction spécifiques à un certain domaine. La définition de telles règles peut être effectuée manuellement, par un spécialiste du domaine concerné ou avec différents degrés d'automatisation (supervisé, semi-supervisé ou non supervisée).

Ainsi, l'EI peut concerner une collection de documents dont on veut extraire des faits précis. Le WWW est un bon exemple d'une telle collection de documents. Ici, des informations sur un sujet se trouvent fréquemment éparpillées dans de différents sites, sous de divers formats de présentation et donc, il serait très souhaitable si ces informations puissent être extraites et intégrées de manière structurée.

Dans notre recherche, nous considérons que l'EI consiste à produire automatiquement des informations structurées à partir d'un ensemble de documents. La tâche d'EI sera réalisée par

des programmes nommés *extracteurs* (*wrappers*) que l'on peut définir comme une fonction de l'espace des documents d'entrée vers l'ensemble des structures de sortie.

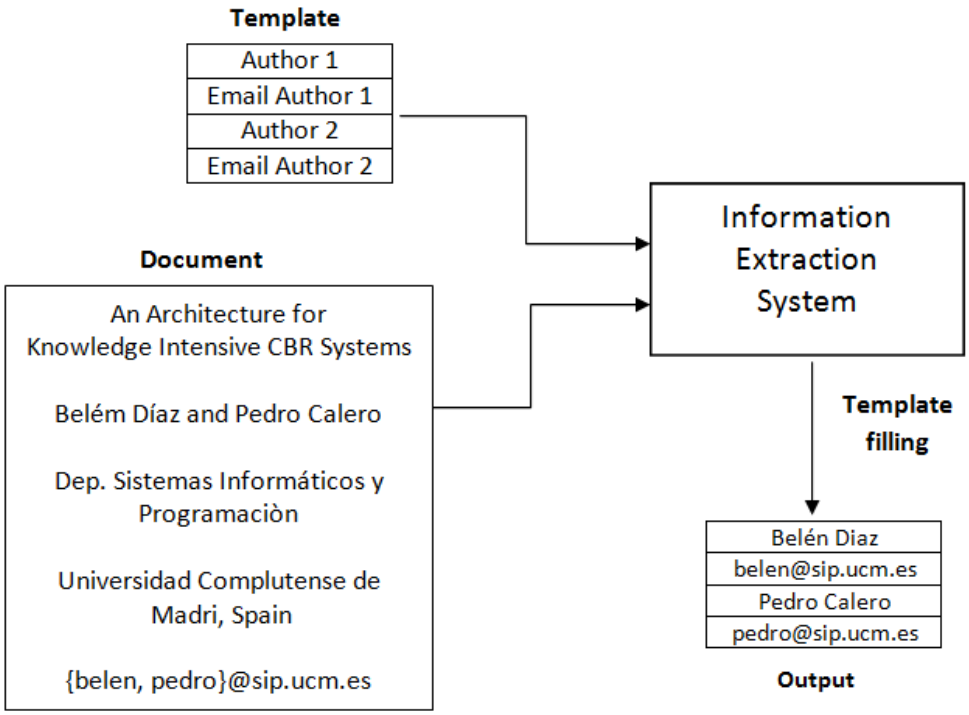



**Fig. 2.** Exemple d'extraction d'information : auteurs et emails (adapté de [Cabral, 2004]).

Dans ce chapitre nous abordons tout d'abord l'EI à partir de documents non structurés et semi-structurés, grâce au Traitement Automatique de la Langue (TAL) et à la conception d'extracteurs. Puis nous examinons les différents types de tâches d'extraction selon la représentation de documents et à la nature de sortie d'extracteurs retenues. Ensuite, nous présentons une méthodologie pour évaluer l'efficacité d'extracteurs conçus, avant de conclure.

## 2.1 Traitement Automatique de la Langue (TAL)

L'objectif du *Traitement Automatique des Langues* (*TAL*) est la conception de logiciels capables de traiter de façon automatique des données exprimées dans une langue (dite "naturelle", par opposition aux langages formels de la logique mathématique). Ces données linguistiques peuvent, selon le cas, être de différents types (textes écrits ou oraux) et de taille variable (du texte entier au mot isolé, en passant par la phrase ou le syntagme). Selon la nature de l'application, le traitement peut viser à transformer des données linguistiques existantes à des fins de correction, d'extraction d'information, de traduction, etc.

Actuellement nous avons un réel et croissant besoin de disposer d'outils et de méthodes robustes pour traiter la langue sous toutes ses formes : requêtes documentaires (moteurs de recherche), extraction d'information précises, correction orthographiques, fouille de données textuelles, etc. Les procédures d'évaluation développées initialement en RI, dans le cadre des compétitions internationales en EI et en RI (*Message Understanding Conference* – MUC [Hirschman, 1998]) ont été généralisées à d'autres domaines : Reconnaissance d'Entités Nommées (*Named Entity Recognition*), c'est-à-dire des noms de personnes, d'intuitions, de lieux ; étiquetage morphosyntaxique (*POS Tagging*) [Adda et al., 1999], pour en citer quelques-unes.

### 2.1.1 Exemple d'un système d'EI reposant sur des techniques de TAL

GATE (*General Architecture for Text Engineering*) [Cunningham & Maynard, 2002.a] est un ensemble d'outils logiciels développés en Java à l'Université de *Sheffield* à partir de 1995 et largement utilisé par de nombreuses communautés (scientifiques, entreprises, enseignants, étudiants) pour le traitement du langage naturel en différentes langues. La communauté de développeurs et de chercheurs autour de GATE est impliquée dans plusieurs projets de recherche européens comme TAO (*Transitioning Applications to Ontologies*) et SEKT (*Semantically Enabled Knowledge Technology*) [SEKT, 2006].

GATE offre une architecture, une interface de programmation d'applications (API) et un environnement de programmation graphique. Il comporte un système d'EI, nommé ANNIE (*A Nearly-New Information Extraction System*) [Cunningham et al., 2002.b], lui-même formé de plusieurs modules, parmi lesquels, un analyseur lexical, un *gazetteer* (dictionnaire géographique), un parseur de phrases (avec désambigüisation), un étiqueteur morphosyntaxique (POS tagging), un module d'extraction d'entités nommées, et enfin un module de détection de coréférences.

GATE est déjà mis en œuvre en anglais, espagnol et français. Il existe de nombreux *plugins* d'apprentissage automatique (*Weka, RASP, MAXENT, SVM light*), d'autres pour la construction d'ontologies (*WordNet*), pour l'interrogation de moteurs de recherche comme *Google* et *Yahoo*, et pour l'étiquetage de partie de discours (*Brill Tagger* [Brill, 1992]*)*, etc.

GATE accepte en entrée divers formats de texte comme le texte brut, HTML, XML, Microsoft Word, PDF, ainsi que divers formats de bases de données comme PostgreSQL et Oracle grâce à JDBC. Il utilise également le langage JAPE (*Java Annotation Patterns Engine*) pour construire des règles d'annotation de documents. On y trouve aussi un *debugger* et des outils de comparaison de corpus et d'annotations.

## 2.2 Types de Documents

Les types de documents auxquels cette recherche s'intéresse sont présentés dans cette section. On peut distinguer trois types de documents par rapport à leur niveaux de structuration : *non structurés, semi-structurés et structurés.*

### 2.2.1 Documents non structurés

La notion de document *non structuré* se définit du point de vue des bases de données. En effet, dans une base de données, lorsque l'on parle de données structurées, on désigne les données qui possèdent une structure préalablement connue par le biais d'un schéma de base qui les organisent et qui rend aisé leur manipulation et leur interrogation par des requêtes.

Dans un document non structuré, un texte par exemple, les informations qu'il contient ont une structure a priori inconnue et très variable. Pour ce type de document on ne dispose pas d'un schéma qui indiquerait le type des données et leur organisation qui puisse guider l'interrogation directe des données. Malgré tout, un texte en langage naturel possède une structure dite grammaticale, que l'on peut exhiber à l'aide de techniques TAL.

Les conférences MUC [Hirschman, 1998] se sont focalisées sur la compréhension de textes en langue naturelle. Cette tâche d'extraction d'information est faisable à l'aide de techniques de TAL. Les figures suivantes (Fig. 3 et 4) présentent, respectivement, l'un des textes considérés par MUC 4 et les informations à extraire de ce même texte. Les informations extraites sont structurées sous la forme d'un enregistrement avec plusieurs champs.

Les tâches considérées dans les conférences MUC sont l'extraction depuis de récits d'attentats en Amérique du Sud d'informations comme la date, le lieu et le type d'attentat ou encore le nom et le type de l'organisation revendiquant l'attentat.

LIMA, 23 AUG 88 (EFE) -- [TEXT] TODAY PERUVIAN MILITARY OFFICIALS ESTABLISHED A CURFEW IN TINGO MARIA BECAUSE OF THE STRIKE BY PEASANTSAND COCA GROWERS TO PROTEST THE DESTRUCTION OF THEIR COCA FIELDS.

SINCE 20 AUGUST THE STRIKERS HAVE BEEN BLOCKING THE ROAD LINKING HUANUCO WITH TINGO MARIA, HUANUCO DEPARTMENT, IN THE NORTHERN JUNGLE WHERE THERE IS DRUG TRAFFIC ACTIVITY.

THE STRIKERS ALSO BLOCKED THE MARGINAL DE LA SELVA HIGHWAY, ISOLATING TOCACHE, UCHIZA, JUANJUI, AND AUCAYACU, WHICH ARE WELL-KNOWN CENTERS FOR DRUG TRAFFICKING.

THE STRIKE IS SUPPORTED BY THE SHINING PATH, ACCORDING TO REPORTS DISCLOSED IN LIMA TODAY.

THE COCA GROWERS OPPOSE THE USE OF THE HERBICIDE CALLED "SPIKE", WHICH THE GOVERNMENT IS PLANNING TO USE TO DESTROY THE CLANDESTINE COCA PLANTATIONS. ACCORDING TO UNOFFICIAL SOURCES, A GROUP OF TERRORISTS LINKED TO DRUG TRAFFICKING REPORTEDLY WOUNDED A POLICE OFFICER IN AN ATTACK ON A POLICE HELICOPTER. TERRORISTS ALSO ATTACKED A CIVIL GUARD POST IN NUEVO PROGRESO, NORTH OF TINGO MARIA, UCAYALI DEPARTMENT.

**Fig. 3.** Un document non structuré du MUC 4.

| | |
|---|---|
| 0. MESSAGE: ID | TST4-MUC4-0003 |
| 1. MESSAGE: TEMPLATE | 1 |
| 2. INCIDENT: DATE | 23 AUG 88 |
| 3. INCIDENT: LOCATION | PERU |
| 4. INCIDENT: TYPE | ATTACK |
| 5. INCIDENT: STAGE OF EXECUTION | ACCOMPLISHED |
| 6. INCIDENT: INSTRUMENT ID | - |
| 7. INCIDENT: INSTRUMENT TYPE | - |
| 8. PERP: INCIDENT CATEGORY | TERRORIST ACT |
| 9. PERP: INDIVIDUAL ID | "GROUP OF TERRORISTS" |
| 10. PERP: ORGANIZATION ID | - |
| 11. PERP: ORGANIZATION CONFIDENCE | - |
| 12. PHYS TGT: ID | "THEIR COCA"/ "POLICE HELICOPTER" |
| 13 TRANSPORT VEHICLE: | "POLICE HELICOPTER" |
| 14. PHYS TGT: NUMBER | - |
| 15. PHYS TGT: FOREIGN NATION | - |
| 16. PHYS TGT: EFFECT OF INCIDENT | - |
| 17. PHYS TGT: TOTAL NUMBER | - |
| 18. HUM TGT: NAME | - |
| 19. HUM TGT: DESCRIPTION | "POLICE OFFICER" |
| 20. HUM TGT: TYPE | LAW ENFORCEMENT: "POLICE OFFICER" |
| 21. HUM TGT: NUMBER | 1: "POLICE OFFICER" |
| 22. HUM TGT: FOREIGN NATION | - |
| 23. HUM TGT: EFFECT OF INCIDENT | INJURY: "POLICE OFFICER" |
| 24. HUM TGT: TOTAL NUMBER | - |

**Fig. 4.** Données à extraire du document de la Fig. 3.

### 2.2.2 Documents semi-structurés

Les données du Web ne sont pas structurées comme celles de bases de données [Abiteboul, 1997; Florescu et al., 1998]. Les pages Web sont en général des documents semi-structurés, comme les documents HTML/XHTML. Ces documents sont décrits par les noms de balises qui sont chargées d'un certain niveau de sémantique et de mise en forme. Cette caractéristique rend les documents semi-structurés compréhensibles par un humain.

Le langage HTML est le langage de mise en forme de pages Web à l'aide de paires de balises de mise en forme. Chaque paire est constituée d'une balise ouvrante, suite de caractères délimités par les symboles < et >, et d'une balise fermante, suite de caractères délimités par les symboles </ et >. Tel langage permet la conception de documents plus riches que du texte plat, en décrivant à la fois la structure du document, son contenu et sa présentation. Par exemple, dans le document HTML de la Fig. 5, la balise H1 indique un titre et la balise P délimite un paragraphe de texte.

Les pages Web sont produites manuellement ou automatiquement par programme. Dans ce cas elles intègrent souvent des informations provenant d'une base de données, par exemple les pages de résultats d'un moteur de recherche ou les pages d'un site *e-commerce*. Par

extension, on peut voir l'EI comme la transformation inverse (mais inconnue) de celle qui ont produit les pages en question.

```
<html>
<head>
<title>MUC Data Sets</title>
<META name="description" content="Listing of the various datasets available for download.">
<META name="keywords" content="Database, database, Information Extraction, MUC, muc, MUC Proceedings, MUC proceedings, MUC Data Sets, data sets, information extraction, SAIC, Science Applications International Corporation">
</head>

<BODY BGCOLOR="#FFFFFF" TEXT="#000000" link="#0033FF">
<IMG SRC="../graphics/blue_bar.gif" width="721" height="10"><P>
<H1>MUC Data Sets</H1>
<IMG SRC="../graphics/blue_bar.gif" width="721" height="10"><P>

For each evaluation, ground truth had to be established to determine the reliability of the participating systems. Datasets were typically prepared by human annotators for training, dry run test, and formal run test usage. These datasets are now being made available wherever possible on this website.

</body>
</html>
```

**Fig. 5.** Exemple d'un document HTML.

### 2.2.3 Documents structurés

Depuis sa création en 1998 par le W3C, le format XML est devenu un standard pour l'échange et le stockage de données semi-structurés. On trouve à présent divers genres de documents XML : RDF (diffusion de métadonnées sur le Web), MATHML (langage d'écriture de formules mathématiques), ODF et DOCBOOK (pour l'édition de document) et WSDL (langage de description des interfaces de services Web), etc.

Les documents XML ont une structure arborescente, traduite par l'imbrication des balises, qui décrit à la fois la structure logique du document et son contenu. Les nœuds internes ou les éléments de l'arbre représentant un document XML correspondent aux éléments de sa structure logique, tandis que son contenu est stocké dans les feuilles de l'arbre. Par exemple, le document XML de la Fig. 6 contient les éléments *catalog*, *cd*, *title*, *artist*, *country*, *company* et *year*. Chaque élément est représenté par une paire de balises ouvrante/fermante. Dans cette figure, l'élément *title* est représenté par les balises <titre> et </titre>.

Pour être bien formé, toute balise ouverte doit être fermée et les balises fermantes apparaissent dans l'ordre inverse des balises ouvrantes (cette contrainte n'est pas obligatoire dans le cas d'HTML). À la différence d'HTML où l'ensemble des balises est défini par une norme fixée, XML permet de créer l'ensemble des balises utilisées. On peut ainsi créer sa propre variante en fonction de ses besoins, en décrivant à la fois les balises et la sémantique qui leur est associée.

À l'aide d'un schéma DTD (*Document Type Definition*) ou un schéma XML qui détermine l'ensemble de balisage conçu et, conséquemment, la façon dont elles sont structurées, on peut contraindre la structure arborescence des documents XML et les types de données qui y figurent.

```
<?xml version="1.0" encoding="utf-8"?>
<CATALOG>
	<CD>
		<TITLE>Empire Burlesque</TITLE>
		<ARTIST>Bob Dylan</ARTIST>
		<COUNTRY>USA</COUNTRY>
		<COMPANY>Columbia</COMPANY>
		<YEAR>1985</YEAR>
	</CD>
	<CD>
		<TITLE>Hide your heart</TITLE>
		<ARTIST>Bonnie Tyler</ARTIST>
		<COUNTRY>UK</COUNTRY>
		<COMPANY>CBS Records</COMPANY>
		<YEAR>1988</YEAR>
	</CD>
	<CD>
		<TITLE>Greatest Hits</TITLE>
		<ARTIST>Dolly Parton</ARTIST>
		<COUNTRY>USA</COUNTRY>
		<COMPANY>RCA</COMPANY>
		<YEAR>1982</YEAR>
	</CD>
	<CD>
		<TITLE>Still got the blues</TITLE>
		<ARTIST>Gary Moore</ARTIST>
		<COUNTRY>UK</COUNTRY>
		<COMPANY>Virgin records</COMPANY>
		<YEAR>1990</YEAR>
	</CD>
	<CD>
		<TITLE>Eros</TITLE>
		<ARTIST>Eros Ramazzotti</ARTIST>
		<COUNTRY>EU</COUNTRY>
		<COMPANY>BMG</COMPANY>
		<YEAR>1997</YEAR>
	</CD>
</CATALOG>
```

**Fig. 6.** Un exemple de document XML représentant un catalogue de CD.

### 2.2.4 Bilan

Les documents XML sont considérés comme structurés car il existe des schémas DTD ou XML disponibles pour décrire les données. Les textes libres sont non structurés puisqu'ils exigent un substantiel traitement de langage naturel. Pour le grand volume de pages HTML sur le Web, elles sont considérées comme semi-structurées puisque les données incluses sont souvent récupérées grâce à l'utilisation de balises HTML. La figure suivante montre un aperçu de divers types de documents en considérant les dimensions *niveaux de structuration* et *facilité de traitement par la machine*.

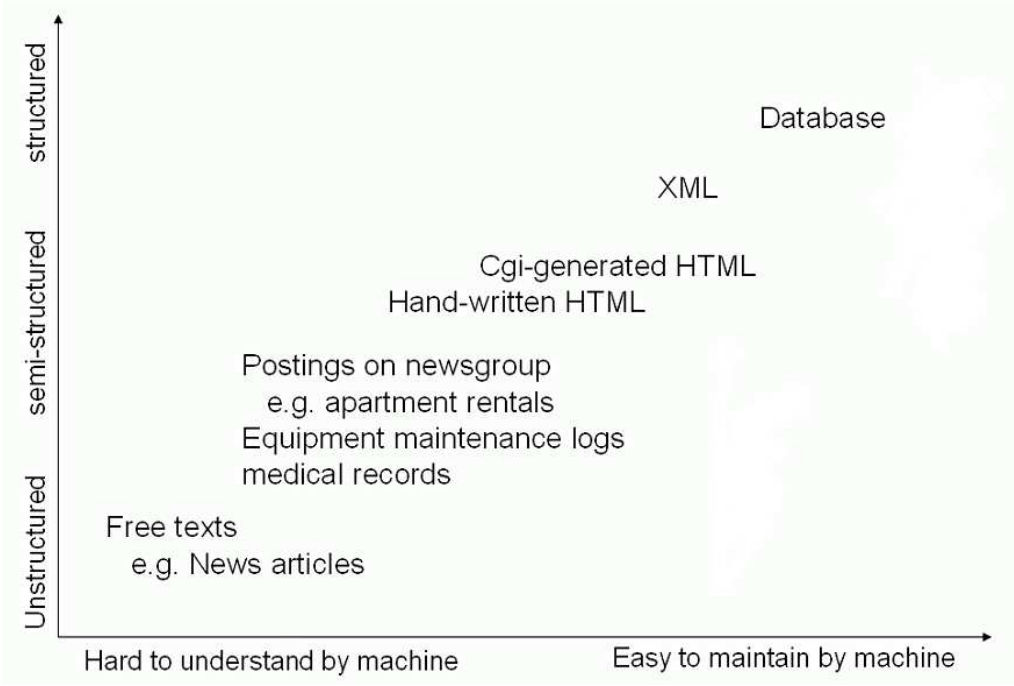



**Fig. 7.** Structuration de documents selon [Chang, 2006].

## 2.3 Concepts de base en EI

Cette section discute deux concepts de bases relatifs à la tâche d'EI, a savoir : la *représentation de documents* et les *types de sorties.*

### 2.3.1 Représentation de documents

A fin de préciser la façon de représenter des documents semi-structurés dans le contexte d'une tâche d'EI typique, deux représentations sont alors possibles : la représentation par une *séquence* et *par un arbre*.

#### 2.3.1.1 Représentation par une séquence

Cette représentation de documents sous la forme d'une séquence d'unités lexicales, ou *tokens*, est la plus courante en EI. Elle est obtenue à partir du code source de la page.

Quant à l'atomicité, un token peut se présenter sous un aspect seulement syntaxique, ou avoir aussi un aspect sémantique : dans le premier cas, des tokens caractérisent par exemple simplement des caractères individuellement considérés, ou une séquence de caractères définie par un critère syntaxique, par exemple une expression régulière qui puisse exprimer des dates, des emails, des chiffres, etc. Pour l'aspect sémantique, des tokens caractérisent des unités sémantiques plus complexes, par exemple, les mots définis à l'aide de dictionnaires ou d'outils de traitement du langage naturel (lemmatiseur, identificateur d'entités nommées).

Dans un document, un token représente soit une balise ouvrante/fermante, soit toute autre séquence de caractères comprise entre deux caractères blancs (espace, tabulation, retour chariot).

#### 2.3.1.2 Représentation par un arbre

L'imbrication de balises, comme celles présentées dans les Fig. 8 et 9, définissent naturellement une structure arborescente intrinsèque aux documents XHTML/XML. En fait, chaque paire de balises ouvrante/fermante peut définir *n* sous arbres dont la racine aura comme label, le non de la balise en question. D'autre part, les portions de textes qui ne sont pas de balises, sont les feuilles de l'arbre. Les données se sont trouvées dans les feuilles textes tandis que l'organisation de nœuds internes détermine la structure des données.

```
<html> <body>
     <b> Book Name </b> Data Mining
     <b> Reviews </b>
     <ol>
        <li>
           <b> Reviewer Name </b> Jeff
           <b> Rating </b>  2
           <b> Text </b>  ...
        </li>
        <li>
           <b> Reviewer Name </b> Jane
           <b> Rating </b> 6
           <b> Text </b>  ...
        </li>
      </ol>
</body> </html>
```

**Fig. 8.** Rendu d'une page HTML (adapté de [Chang et al., 2006]).

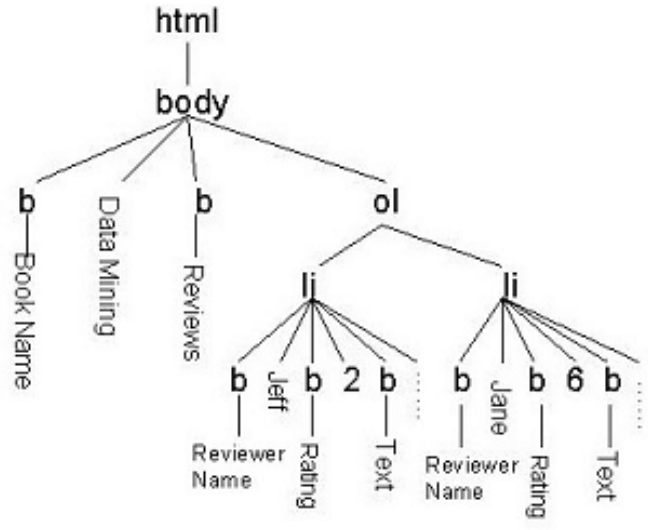



**Fig. 9**. Représentation arborescente (DOM Tree) de la page HTML de la Fig. 8 [Chang et al., 2006] .

### 2.3.2 Types de Sorties

En ce qui concerne la sortie, un système d'EI peut être : *single-slot (unaire), multi-slot (n-aire)* ou *structurée*.

**Sortie single-slot (unaire)** Les systèmes d'EI single-slot ou unaire extraient du document d'entrée seulement des données isolées, c'est-à-dire, ils ne sont pas capables de lier une instance d'un champ d'information (slot) du formulaire de sortie à une instance d'autre champ.

**Sortie multi-slot (n-aire)** Ceux-ci sont capables d'extraire du document d'entrée, des données liées les unes aux autres, c'est-à-dire, ils peuvent faire des relations entre les instances de différents slots.

La Fig. 10 illustre ce deux types de sorties.

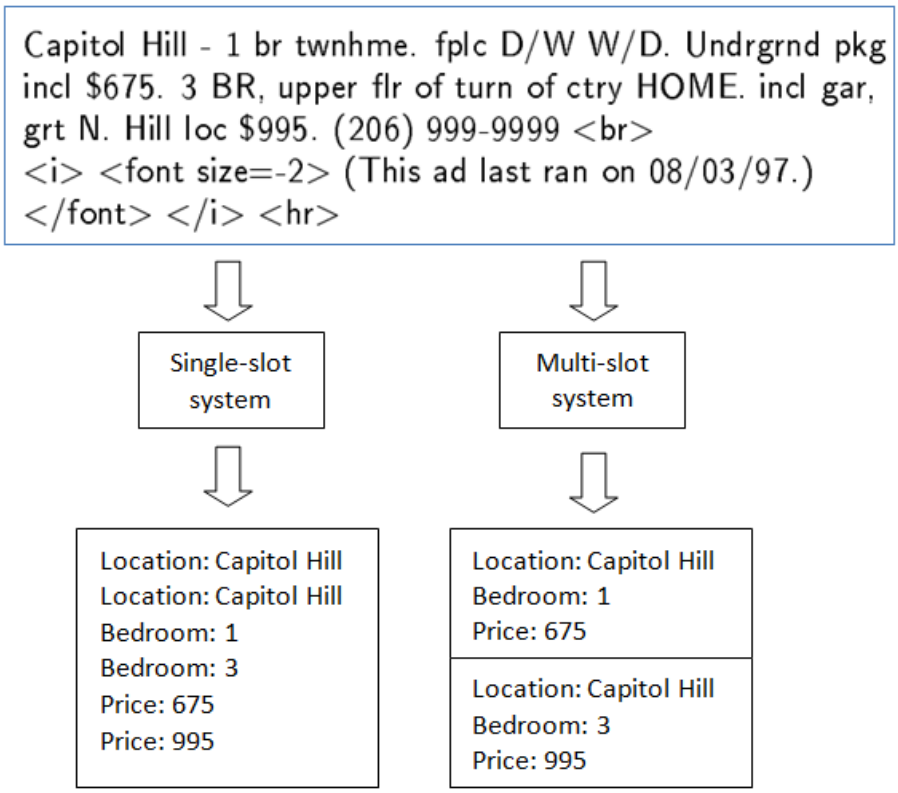



**Fig. 10.** Exemple d'extractions single-slot et multi-slot d'une page HTML (adapté de [Soderland, 99]).

**Sortie structuré**

Une sortie structurée apparaît quand le résultat d'une tâche d'extraction prend la forme arborescente. Par exemple, soit la liste de livres illustrée par la Fig. 11 obtenue du code source d'un document XHTML. Ici la tâche d'extraction consiste à obtenir, pour chaque livre, son titre et ses auteurs ou, plus précisément, à extraire la liste des livres et, pour chacun d'eux, la liste de couples (*titre, liste des auteurs*).

Le nombre d'auteurs varie d'un livre à l'autre. Dans ce cas, une représentation sous la forme d'un arbre semble plus adéquate. La Fig. 12 illustre que certaines balises du document d'entrée ont disparus (*html et body*) et les autres balises ont été renommées. Par exemple, la balise *h1* est renommée en une balise *titre* dans le document de sortie. En plus, la balise *li* devient *livre*, et la balise *em* devient *auteur*. Par contre, il existe de nouveaux éléments qui ne correspondaient pas à aucune balise du document d'entrée. Cela est le cas que l'on constate que les balises *auteurs* d'un même livre ont été regroupées sous une nouvelle balise *auteurs*. De l'exposé par cet exemple, une sortie structurée résultante d'une telle tâche d'EI peut être potentiellement complexe.

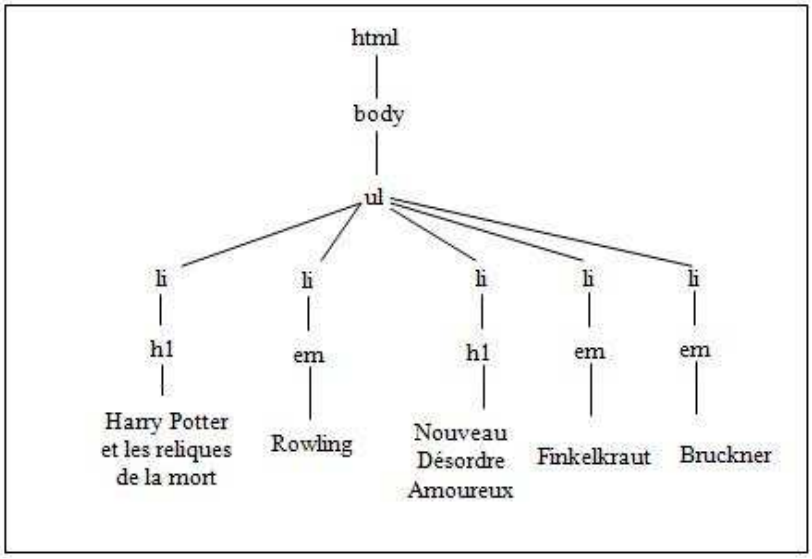



**Fig. 11.** Structure arborescente d'un document HTML contenant une liste de livres (adapté de [Marty, 2007]).

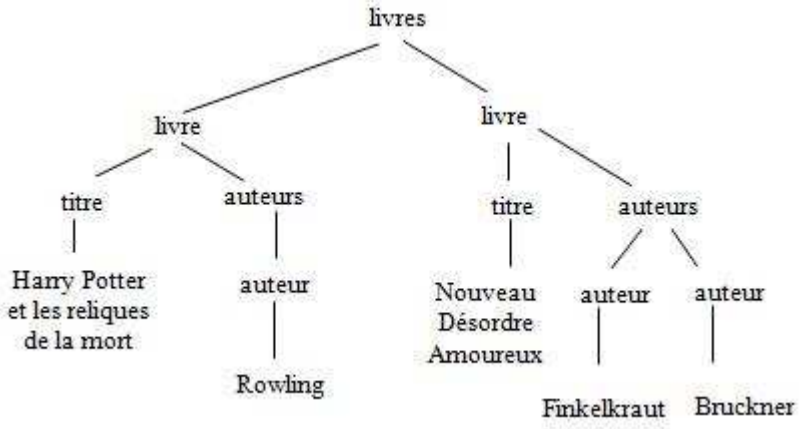



**Fig. 12.** Sortie structuré extrait du document HTML de la Fig. 11 (adapté de [Marty, 2007]).

## 2.4 Conception d'extracteurs

Rappelons qu'un extracteur est définit comme étant un programme qui produit automatiquement des informations structurées à partir d'un ensemble de documents.

D'après [Chang et al., 2006] on peut classifier les approches pour construire un extracteur en 4 classes, quant au niveau d'interaction entre le système et l'utilisateur : (1) *approches manuelle*, (2) a*pproche par spécification assistée*, (3) *approche par induction supervisée d'extracteurs* et (4) *approche par induction non supervisée.*

### 2.4.1 Approche manuelle

Dans cette approche, en employant des langages de programmation généraux tels que Perl, Java ou des expressions régulières pour l'extraction de portions de textes, le développeur programme manuellement un extracteur pour un site Web quelconque. Ainsi, une forte compétence de programmation est requise et cela peut devenir couteux.

Le langage d'interrogation sur les documents semi-structuré tel que XQUERY ou le langage XPATH peuvent également être utilisés pour le développement d'extracteurs car ils tiennent en compte la structure arborescente de documents XML. Par contre, cette approche pose de problèmes parce que c'est une tâche difficile, fastidieuse qui exige des connaissances et de l'expertise. En fait, si la structure d'une page Web change, cela rend difficile la maintenance d'extracteurs conçus de cette façon. Exemples de ce type de systèmes sont W4F [Sahuguet & Azavant, 2001] et XWRAP [Liu et al., 2000].

.

### 2.4.2 Approche par spécification assistée

À l'aide d'une interface conviviale (GUI), les systèmes qui se servent de cette approche, tels que OLERA [Chang & Kuo, 2004], IEPAD [Chang & Lui, 2001] et Lixto [Baumgartner et al., 2001], guident l'utilisateur dans la tâche de conception d'extracteurs. Ils analysent la structure de documents et indiquent à l'utilisateur des éléments à extraire en lui proposant, en général, des patrons d'extraction (adresse électroniques, des prix, des URLs, valeurs numériques, etc.) à base d'expressions régulières sur les informations pertinentes. Ensuite, c'est l'utilisateur qui les choisit selon le type de donnée à extraire.

Pendant une interaction, un extracteur peut être défini sans que l'utilisateur n'ait besoin de le manipuler directement ou de connaître le formalisme d'extraction employé par l'outil. De façon analogue à l'approche manuelle, mais dans une moindre mesure, l'approche par spécification compte sur l'expertise de l'utilisateur pour les tâches d'extraction, sans mentionner que c'est lui qui doit également faire l'analyse des documents.

### 2.4.3 Approche par induction supervisée d'extracteurs

Dans cette approche, l'induction d'extracteurs [Kushmerick, 2000] est effectuée automatiquement par un algorithme d'apprentissage machine à partir de documents annotés, aussi appelés *exemples*, par l'utilisateur. En plus, il est souhaitable de réaliser l'induction d'extracteurs en faisant en sorte que l'utilisateur annote le moins nombre possible d'exemples.

Les algorithmes d'induction d'extracteurs reposent sur l'exploitation des régularités morphosyntaxiques et/ou structurelles des documents HTML, XHTML et XML, qui permettent de repérer les données à extraire.

Les pages obtenus par des moteurs de recherche sur de nombreux sites Web, comme des sites de *e-commerce*, ont une grande régularité car elles sont construites, la plupart du temps, à partir d'une base de données. Par conséquent, dû à une grande régularité présente dans ces pages, les systèmes d'induction supervisée d'extracteurs peuvent exploiter le patron de mise en page pour l'EI. Un avantage de cette approche est qu'on peut utiliser n'importe quel algorithme d'apprentissage supervisé existant en tant que brique d'apprentissage. Ici, l'objectif est de profiter de l'existant en matière d'algorithme d'apprentissage et de faciliter son évolution future, principalement si l'on souhaite avoir un système bien modulaire.

L'induction d'extracteurs supervisée peut être réalisée depuis les exemples *positifs* (données que l'utilisateur veut extraire) et les exemples *négatifs* ou *contre-exemples* (données que l'utilisateur ne veut pas extraire).

### 2.4.4 Approche par induction non supervisée

La conception d'extracteurs par induction non supervisée, contrairement à l'approche précédente, se passe complètement de l'intervention de l'utilisateur. Généralement ces extracteurs sont reposant sur des techniques d'inférence grammaticale [Crescenzi et al., 2001] ou sur des méthodes d'alignement (de chaînes ou d'arbres) des documents [Arasu, 2003] .De cette façon, l'induction non supervisée produit un extracteur à partir des documents à extraire en analysant leurs structures et régularités.

Les extracteurs induits par ces systèmes produisent en sortie des données structurées sous la forme d'une table avec d'éventuelles imbrications. Cependant les extracteurs non supervisés sont moins précis que ceux induits par l'approche supervisée [Zhai & Liu, 2005]. Mais les extracteurs non supervisés ont l'avantage, comme l'utilisateur n'est pas nécessaire à leur fonctionnement, de pouvoir être intégrés à des chaînes de traitements automatiques de documents. *RoadRunner* [Crescenzi et al., 2001] et *DeLa* [Wang & Locovsky, 2002] sont des exemples de systèmes qui appartiennent à cette approche. En général parmi les informations extraites par ces systèmes, celles qui sont vraiment pertinentes, sont souvent soumises à un post-traitement.

## 2.5 Mesures d'évaluation de systèmes d'EI

Les mesures d'évaluation pour le problème d'EI sont apparues lors des conférences MUC (*Message Understanding* Conference) [Hirschman, 1998]. L'étude menée par les quatre premières MUC a servi de base à la définition des mesures d'évaluation existantes. Initialement ces mesures ont été développées en se fondant sur des mesures de *précision* et de *rappel* du domaine de la Recherche d'Informations (RI).

Cette section décrit la méthodologie d'évaluation de résultats d'une tâche d'EI. Avant de montrer les mesures classiques d'évaluation des performances, il faut préciser comment évaluer la correction d'une donnée extraite. Plusieurs critères sont envisageables pour définir la correction d'une donnée extraite dans le cas d'extraction *single-slot*.

### 2.5.1 Critères de correction

Afin d'évaluer la correction d'une séquence extraite, 3 critères sont applicables à tous les types de documents représentés sous la forme d'une séquence (tokens). Le premier critère est le plus lâche : une donnée extraite est correcte si elle contient une partie d'une donnée à extraire. Le second est un peu plus strict que le premier : une donnée extraite est correcte si elle contient une donnée à extraire (dans son intégralité). Finalement, le troisième critère, le plus strict : une donnée extraite est correcte si elle correspond exactement, au caractère près, à l'une des séquences à extraire. C'est ce dernier critère qui sera adopté dans les expérimentations de cette recherche.

### 2.5.2 Précision, Rappel et F-Measure

La *Précision* est définie comme la quantité d'informations correctement extraites sur toutes les informations extraites, tandis que le *Rappel* est défini comme la quantité d'informations correctement extraites sur toutes les informations à extraire contenues dans les documents. Ainsi, le rappel mesure la quantité d'informations extraites correctement parmi l'ensemble

des informations à extraire. Un rappel de 100 signifie que l'extracteur a bien reconnu toutes les valeurs à extraire. La précision mesure la qualité des extractions, tandis que le rappel met en évidence la proportion d'extractions correctes. Une précision et un rappel de 100 indique un extracteur parfait, dans le sens où il ne commet aucune erreur sur le corpus considéré.

De ce fait, précision ( $P$ ) et rappel ( $R$ ) peuvent être définis par les formules suivantes :

$$P = \frac{N_c}{N_p} \quad (1)$$

$$R = \frac{N_c}{N_t} \quad (2)$$

où, $N_c$ est le nombre de slots qui ont été extraits correctement par le système, $N_p$ est le nombre total de slots qui ont été extraits par le système, et $N_t$ est le nombre total de slots qui devraient être extraits par le système.

Ces mesures sont inversement liées, en effet quand le rappel augmente, la précision tend à diminuer et vice-versa. Une nouvelle mesure a été établie qui combine les mesures précédentes et est appelée la *F-Measure*. La F-Mesure évalue la qualité globale d'un extracteur en combinant sa précision et son rappel en une mesure unique. Cette mesure est exprimée par la formule :

$$F - Measure = \frac{(\beta^2 + 1) * R * P}{\beta^2 * (R + P)} \quad (3)$$

où, le paramètre $\beta$ quantifie la préférence du rappel sur la précision. On fixe en général $\beta = 1$ dans l'équation ci-dessus avec l'intention d'évaluer des systèmes d'IE en équilibrant les deux mesures et alors la formule ci-dessus calcule la *moyenne harmonique* entre la précision *P* et le rappel *R*.

## 2.6 Conclusion

Pour cette recherche, quant à la conception d'extracteurs, l'approche manuelle ou l'approche par spécification assistée nous sont apparues lourdes à mettre en œuvre car elles demandent un investissement important d'expertise humaine tant en ce qui concerne le système d'extraction que le domaine d'extraction. L'induction non supervisé n'est pas retenue non plus en raison de son manque de précision et de la nécessité d'un post-traitement pour filtrer ou mieux structurer les données extraites. En conséquence, nous avons retenu une conception d'extracteur selon une approche par induction supervisée, en nous limitant à des extracteurs unaires pour des documents semi-structurés et libres (format XHTML). Cette approche retenue nécessite la connaissance d'exemples de données à extraire et une annotation de ces exemples d'apprentissage réalisée par l'utilisateur. Afin que la tâche d'annotation ne soit pas fastidieuse, l'apprentissage permettant l'induction de l'extracteur doit pouvoir être fait à partir de peu d'exemples. La section suivante traitera plus en détail l'induction supervisée d'extracteurs.

# 3 INDUCTION D'EXTRACTEURS ET CLASSIFICATION SUPERVISÉE

Le domaine de l'apprentissage artificiel, qui relève de l'intelligence artificielle, propose de nombreux algorithmes dont les propriétés ont déjà été bien établies [Cornuéjols & Miclet, 2002; Mitchell, 1997 ; Russel & Norwig, 2003]. Parmi les diverses techniques d'apprentissage artificiel appliquées à l'induction d'extracteurs d'information, ce chapitre s'intéresse aux techniques relevant de la Classification Supervisée (CS).

Dans un problème de la CS, des exemples convenablement étiquetés sont fournis à l'algorithme d'apprentissage. Les étiquettes indiquent les classes auxquelles les exemples appartiennent. Pour le cas à deux classes (cas binaire), chaque exemple peut être soit positif, soit négatif. À partir de ces exemples, il faut alors d'apprendre un classificateur capable de prédire la classe d'un exemple.

On peut se servir de la CS pour concevoir un système d'EI constitué de deux phases: *apprentissage* et *extraction*. La phase d'apprentissage utilise un ensemble d'exemples étiquetés pour produire un modèle de classification capable de remplir les slots à extraire; tandis que la phase d'extraction applique ce modèle appris sur des documents non étiquetés pour en extraire les données. Ces deux phases partagent un même prétraitement de représentation de documents afin de les segmenter en unités sur lesquelles portera la classification.

Cette recherche suit la même piste prometteuse de la CS appliquée avec succès à l'EI dans plusieurs travaux [Freitag & Kushemerick, 2000; Marty & Torre, 2003 ; Gilleron et al., 2006 ; Finn & Kushmerick, 2004 ; Giuliano et al., 2006 ; Li et al., 2004 b].

Dans ce contexte, ce chapitre présent tout d'abord la notion de la classification supervisée et puis explique comment le problème de l'EI peut être formulé comme un problème de classification supervisée. Ensuite le système BWI est examiné comme un système de référence qui emploie le modèle de la CS pour induire les hypothèses d'induction. Différentes approches et algorithmes utilisées dans le domaine d'EI sont abordés au travers de la présentation de quelques systèmes d'EI existants. Ce chapitre se termine en présentant un tableau de synthèse comparant les caractéristiques majeures des systèmes d'EI étudiés, et en dressant quelques conclusions.

## 3.1 Modèle d'Apprentissage de la Classification Supervisée

### 3.1.1 Classification Supervisée

D'après Mitchell [Mitchell, 1997] classer un objet consiste à l'affecter au groupe auquel il appartient. Autrement dit, on associe à un objet une classe. La *classification supervisée* consiste alors dans l'apprentissage de cette association objet/classe à partir d'objets dont la classe est déjà connue.

Le modèle d'apprentissage de la classification supervisée présume l'existence d'un concept cible *f* que l'on cherche à découvrir à partir d'un ensemble d'exemples déjà classés. Le concept cible *f* peut être vu comme une fonction de l'ensemble d'exemples vers l'ensemble

de classes. Dans ce contexte, l'apprentissage se produit quand on s'approche le plus possible de *f* par une fonction hypothèse *h*. La fonction *h* résultante pourra être utilisée pour prédire la classe d'exemples dont la classe est inconnue [Marty, 2007].

Soit le problème de classification sur deux classes (*classification binaire*). Soit $\{(x_1, y_1),...,(x_n, y_n)\}$ un ensemble de données d'apprentissage dans lequel $x_i$ dénote une instance (un *vecteur d'attributs*) et $y_i \in Y = \{-1,+1\}$ dénote une étiquette de classification. L'ensemble $X$ constitué par $x_1, x_2,..., x_n$ est aussi nommée de *l'espace de description* des exemples tandis que $Y$ est *l'ensemble de classes* possibles. À chaque exemple, on associe sa description $x \in X$ et sa classe $y \in Y$. Chaque couple $(x, y)$ est un exemple étiqueté et l'ensemble de tous les couples constitue l'ensemble d'apprentissage, noté $S$. On dit alors que *x* est un exemple positif si *y* = +1, et un exemple négatif (contre exemple) si *y* = -1. Par conséquent, pour chaque exemple étiqueté $(x, y)$ on a donc $y = f(x)$. La fonction cible *f* $(f : X \to Y)$ est définie sur l'ensemble $X$ et prend ses valeurs dans $Y$. Le but de l'apprentissage en classification supervisée est alors d'induire d'après un ensemble d'exemples étiquetés $S$ une hypothèse $h : X \to Y$ qui approche le plus possible la fonction cible *f* dont on ne dispose pas.

Ce modèle de classification supervisée consiste en deux étapes : *apprentissage* et *prédiction*. Dans l'apprentissage, on essaye de trouver un modèle à partir de données étiquetées qui permet de séparer les données d'apprentissage, tandis que dans la prédiction, les modèles appris servent à identifier si un exemple non étiqueté devrait être classifié comme +1 ou -1[2].

L'espace de recherche de l'algorithme d'apprentissage est aussi appelé de *l'ensemble d'hypothèses*, noté par *H*. L'apprenant opère le processus de généralisation qui est vu comme une recherche dans l'espace d'hypothèses *H* afin de trouver celle qui correspond mieux aux exemples d'apprentissage [Russel & Norwig, 2003]. Ainsi, l'objectif de l'apprenant est de trouveur un compromis entre l'hypothèse la plus générale et l'hypothèse la plus spécifique. C'est-à-dire qu'il doit bien généraliser pour qu'il puisse se détacher des données d'apprentissage (ne pas apprendre par cœur) et éviter alors l' « *overfitting* ». Par contre, il doit en même temps ne pas *sur-généraliser* pour ne pas s'éloigner du concept cible quand il apprend une hypothèse trop générale.

### 3.1.2 EI comme un Problème de Classification Supervisée

La classification supervisée a été appliquée avec succès pour l'EI dans différents travaux [Freitag & McCallum, 1999; Freitag & Kushemerick, 2000; Marty & Torre, 2004; Finn & Kushmerick, 2004; Gilleron et al., 2006]. Un problème d'EI est alors formulé en tant qu'un problème de classification supervisée et l'élaboration d'un système d'EI se fait alors en deux étapes : l'apprentissage et l'extraction. Ces deux étapes sont précédées d'un même prétraitement du document, permettant de le segmenter en unités et de représenter des exemples sur lesquels portera la classification. L'étape d'apprentissage utilise un ensemble de documents étiquetés pour produire un modèle de classification capable d'identifier les données à extraire, tandis que l'étape d'extraction applique le modèle appris à des documents non étiquetés pour en extraire les données.

[2] Dans certains cas, les résultats de prédiction peuvent être des valeurs numériques compris entre 0 et 1. Alors un exemple peut être classifié comme +1 quand la valeur de prédiction est plus grande que 0,5, par exemple.

Une base d'exemples est constituée d'un ensemble de séquence de tokens et les sous-séquences à extraire sont marquées (tâche d'annotation). Ce que nous amène à choisir comment représenter précisément un exemple particulier. On peut le représenter soit comme une *sous-séquence* [Kosala & Blockeel, 2000], éventuellement réduite à un seul token [Seymore et al., 1999; Freitag & McCallum, 1999; Giuliano et al., 2006], soit comme un *séparateur* (de début ou de fin d'une donnée à extraire) [Freitag & Kushmerick, 2000]; dans ce cas, l'apprentissage des séparateurs de débuts et des fins sera effectué indépendamment. Dans ce qui suit, on met en évidence l'apprentissage de séparateurs.

### 3.1.3 Identification de Séparateurs par la Classification Supervisée (CS)

Au lieu de performer l'extraction d'information comme l'extraction d'une sous-chaîne de tokens, on peut la réaliser au travers de *fonctions de séparation ou séparateurs*. Un séparateur est un espace entre deux tokens adjacents. Ici, un séparateur n'est pas quelque chose qui fait partie effectivement du texte (tel que l'espace blanc), mais une notion qui résulte du processus de transformation d'un document en une séquence de tokens.

En se servant de la CS, on souhaite alors appréhender deux classificateurs, autrement dit, deux fonctions d'extractions définissant des *séparateurs* de *début* ou de *fin* d'une donnée à extraire. Ces fonctions sont égales à 1 si, et seulement si, l'espace entre deux tokens adjacents quelconque sont en fait des séparateurs de début ou de fin d'une donnée quelconque à extraire, sinon ces fonctions sont égales à 0. Par exemple, la séquence « Time : 9 PM » contient cinq séparateurs qui sont illustrées respectivement par $s_0$, $s_1$, $s_2$, $s_3$, $s_4$ et $s_5$ (Fig. 13).

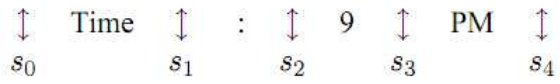



**Fig. 13.** Séquence de tokens avec des séparateurs [Marty & Torre, 2004].
$s_2$ et $s_4$ sont des séparateurs de début et fin, respectivement.

Supposons que l'on veut extraire la donnée '9 PM' de l'exemple présenté dans la figure ci-dessus, alors $s_2$ et $s_4$ seront respectivement le séparateur de début et de fin de cette donnée et les autres, des séparateurs quelconques.

Après avoir choisi la description de l'ensemble d'exemples, il reste à définir un langage d'hypothèses et à choisir un algorithme de classification supervisée permettant de trouver dans l'espace d'hypothèses, l'hypothèse qui sera la plus cohérente avec les exemples disponibles. Ensuite, une hypothèse étant apprise, il faut spécifier le *wrapper* correspondant. Enfin, une étape postérieure se fait nécessaire pour que l'on réassocie les séparateurs de début et de fin.

Dans la section suivante, afin d'illustrer la transformation du problème d'EI en un problème de classification supervisée, nous décrivons le système BWI qui réalise l'extraction single-slot à partir de documents non structurés, et qui faire l'usage de séparateurs indépendants présentés ci-dessus.

### 3.1.4 Boosted Wrapper Induction

Boosted Wrapper Induction (BWI) est un système d'EI, développé par Daniel Freitag [Freitag & Kushmerick, 2000] à l'Université de Pittsburgh, qui induit des extracteurs single-slot pour les documents structurés et non structurés. Il a été un des premiers systèmes à se servir de la classification supervisée comme algorithme de base pour l'apprentissage d'extracteurs. Plus tard, divers travaux notamment [Kauchak et al., 2002; Marty & Torre, 2004] expliquent les résultats honorables obtenus par le système BWI, plus précisément par les algorithmes qu'il met en œuvre.

Dans [Kauchak et al., 2002], les auteurs ont analysé comment les composants algorithmiques du BWI contribuent à son succès. Ils ont mis en évidence que la technique de *boosting* [Freund & Schapire, 1990] est l'élément principal de la réussite de BWI. Il a montré que cela réside dans la capacité de l'algorithme de faire la repondération des exemples afin d'apprendre des règles spécifiques (conduisant à une précision élevée), combinée avec la capacité de continuer l'apprentissage des règles, même après que tous les exemples positifs aient été couverts (conduisant à un rappel élevé).

D'autre part, [Marty & Torre, 2004] ont étudié l'influence du langage de représentation choisi et de connaissances auxiliaires sur la performance du BWI. De plus, ils ont montré qu'un codage de représentation de tokens sous la forme *attribut-valeur*[3], reposant sur des informations de base et sur des ressources simples, associé à un algorithme de classification supervisée classique, comme le BWI, permet d'induire des *wrappers* ayant performants.

Les prochaines sections détailleront l'algorithme BWI en exposant sa manière de représenter les documents, ses algorithmes d'apprentissage et d'extraction aussi bien que ses avantages et limitations.

#### 3.1.4.1 Représentation de documents et d'exemples

Dans BWI, un document est vu comme une séquence d'unités lexicales ou tokens. On distingue trois types de tokens : une séquence de caractères alphanumériques, un caractère de ponctuation, et enfin le caractère de retour chariot. Notons que pour représenter des exemples positifs, BWI emploie la notation de séparateur étudié dans la section précédente.

En effet, un séparateur est soit une interposition entre deux tokens adjacents, soit la position avant (respectivement après) le premier (respectivement dernier) token d'une valeur à extraire. Conséquemment, une donnée à extraire est une sous-séquence de la séquence de tokens représentant le document et elle est caractérisée par un couple de positions (*début, fin*) qui délimitent la séquence de tokens correspondante à cette donnée. De plus, il faut faire apprendre deux classificateurs, c'est-à-dire, deux fonctions de séparation sur {0,1}. On restreint ces fonctions pour qu'elles soient égales à 1 si, et seulement si, ils sont des séparateurs de début et de fin d'une donnée à extraire présent dans le document, sinon elles sont égales à 0.

Les exemples positifs sont ainsi codifiés d'une manière très simple et identique pour les positions de débuts et fins. Chacune de ces positions est représentée par deux ensembles : l'ensemble des tokens se trouvant à sa gauche et celui se trouvant à sa droite.

#### 3.1.4.2 Hypothèses

Les hypothèses, destinées à caractériser les séparateurs représentés par les exemples positifs, sont exprimées par des *règles*, ou des *classificateurs élémentaires*, qui prennent en entrée un séparateur et détermine s'il s'agit ou non d'un séparateur début (ou fin) selon les tokens

[3] La représentation attribut-valeur consiste d'appliquer des fonctions sur les tokens et de leurs attacher les résultats de ces fonctions: les *atributs* avec leurs *valeurs* correpondantes. Par exemple, une fonction *f* appliqué au token « = « peut renvoie: symbol-token = true.

présents à gauche et à droite. Une règle est alors constituée d'une séquence de tokens ou de jokers (*wildcards)*, ces derniers étant la conversion d'un token à une catégorie plus générale. En fait, cela rend les détecteurs plus généraux. Ainsi, au lieu de chercher des correspondances des tokens exactes des détecteurs dans un texte, les jokers correspondent à plusieurs tokens en même temps. Ci-après, des jokers utilisés par BWI sont cités avec leurs significations:

- <alph> correspond aux tokens contenant des caractères de l'alphabet.
- <ANum> correspond aux tokens contenant des caractères alphanumériques.
- <Cap> correspond aux tokens contenant des caractères en majuscule.
- <LC> correspond aux tokens contenant des caractères en minuscule.
- <SChar> correspond aux tokens contenant un seul caractère.
- <Num> correspond aux tokens contenant des chiffres.
- <Punc> correspond aux tokens de ponctuation.
- <*> correspond à n'importe quel token.

### 3.1.4.3 Algorithme d'apprentissage

L'algorithme d'apprentissage de BWI utilise une technique de *boosting* nommé *AdaBoost* [Freund & Schapire, 1996, 1997] pour l'induction de séparateur début et fin des exemples positifs. Dans [Freund & Schapire, 1996] les auteurs affirment que la technique de boosting peut être utile pour des problèmes d'apprentissage où les exemples observés tendent à avoir différent niveaux de difficultés. Pour ces problèmes, l'algorithme de boosting tend à générer des distributions qui se concentrent sur des exemples les plus difficiles, ce qui représente un challenge à un algorithme d'apprentissage faible[4], pour avoir une bonne performance sur ces parties plus difficiles de l'échantillon d'exemples.

L'algorithme d'apprentissage de BWI, présenté dans la Fig. 14, apprend deux classificateurs pour reconnaître les *positions début* (*pd*) et les *positions fin* (*pf*). En fait, on se sert de la classification supervisée (CS) pour classer les exemples de début qui jouent le rôle des positifs et tous les autres exemples sont considérés comme négatifs ; Cela aboutit à un classificateur qui détermine les exemples de type début. De façon analogue, la CS caractérise les exemples de type fin jouant cette fois-ci le rôle de positifs et tous les autres étant considérés comme négatifs. Un classificateur qui identifie les exemples de type fin est produit. Finalement, l'algorithme construit un histogramme $H_{pd,pf}$ de *fréquence des tailles* (en nombre de tokens) du champ à extraire permettant d'associer les positions débuts et fins dans une étape postérieure. En d'autres termes, pour chaque champ à extraire, l'algorithme apprend la distribution de probabilité de la longueur d'un champ, en consignant le nombre de tokens dans chacun de ses exemples de corpus, et en normalisant cet histogramme en une distribution de probabilité à la fin de la phase d'apprentissage.

procedure BWI(example sets S and E)
$F \leftarrow$ AdaBoost(LearnDetector, $S$)
$A \leftarrow$ AdaBoost(LearnDetector, $E$)
$H \leftarrow$ field length histogram from $S$ and $E$
return wrapper $W = \langle F, A, H \rangle$

**Fig. 14.** L'algorithme d'apprentissage de BWI [Freitag & Kushmerick, 2000].

[4] L'apprenant faible est celui qui fournisse une hypothèse qui fasse mieux qu'un classificateur aléatoire.

**Algorithme AdaBoost** L'idée principale de l'algorithme *Adaboost* (Fig. 15) est de concevoir un classificateur final par la combinaison de classificateurs élémentaires, ces derniers obtenus à partir d'un algorithme d'apprentissage supervisé, appelé *apprenant faible* [Freund & Schapire, 99]. Dans BWI, *LearnDetector* joue le rôle de l'apprenant faible et réalise l'apprentissage des classificateurs élémentaires (*cd, cf*) à chaque étape de boosting.

AdaBoost itère *n* fois l'apprentissage en faisant varier la pondération des exemples d'apprentissage à chaque étape, en amenant l'apprenant faible *LearnDetector* à se concentrer sur les exemples mal classés, autrement dit, sur des portions de l'ensemble d'apprentissage où les règles courantes n'ont pas eu de bonnes performances lors de l'étape précédente. En même temps, il construit itérativement une hypothèse (initialement vide) par extension des motifs à gauche et à droite du classificateur.

Un motif est étendu par l'ajout d'au plus *L* tokens à chaque itération. Cette valeur L, aussi appelée *lookahead L ou la fenêtre L,* est fournie à l'algorithme en entrée. Elle informe le nombre de tokens pris en compte pendant l'apprentissage de chaque détecteur. Si, par exemple, nous avons la fenêtre L = 3, cela veut dire que 3 tokens avant le détecteur de début du champ à extraire et 3 tokens après le détecteur de début du champ sont pris en compte pendant l'apprentissage d'un détecteur de début. Le contexte appris comprend alors les 6 tokens entourant le début d'un champ à extraire. Ce raisonnement est analogue pour un détecteur de fin.

À ce point dans l'algorithme AdaBoost, tous les motifs possibles à gauche et à droite sont énumérés, et celui qui améliore le plus les performances de la classification de l'hypothèse est conservé. Ce processus itère tant qu'il soit possible d'améliorer l'hypothèse courante. Ensuite, les classificateurs appris sont combinés par vote pondéré. Les poids d'un classificateur est déterminé par l'exactitude de son ensemble d'apprentissage pondéré. Finalement, BWI renvoie deux ensembles de classificateurs (détecteurs) appris après ces itérations, appelés *fore detector*, *F*; et *aft detector*, *A*, ainsi qu'un histogramme *H* de la longueur (en nombre de tokens) du champ cible.

```
procedure LearnDetector(example set Y)
    prefix pattern p ← []
    suffix pattern s ← []
    loop
        prefix pattern p' ← BestPreExt(⟨p, s⟩, Y)
        suffix pattern s' ← BestSufExt(⟨p, s⟩, Y)
        if score(⟨p', s⟩) > score(⟨p, s'⟩)
            if score(⟨p', s⟩) > score(⟨p, s⟩)
                p ← the last |p| + 1 tokens of p'
            else return detector ⟨p, s⟩
        else
            if score(⟨p, s'⟩) > score(⟨p, s⟩)
                s ← the first |s| + 1 tokens of s'
            else return detector ⟨p, s⟩
```

**Fig. 15.** L'apprenant faible *LearnDetector* de BWI [Freitag & Kushmerick, 2000].

### 3.1.4.4 Algorithme d'extraction

La phase d'extraction de BWI est constituée de trois étapes : (1) la classification de *séparateurs début* ($c_d$), (2) la classification de *séparateurs fin* ($c_f$) et (3) la réassociation de chaque séparateur classé comme début à un séparateur classée comme fin à l'aide d'un histogramme $h_{d,f}$.

Les classificateurs $c_d$ et $c_f$, constitués par des classificateurs élémentaires combinés par la technique du boosting, sont appliqués sur tous les séparateurs candidats des documents d'entrée. Cela correspond aux étapes (1) et (2) que l'on vient de mentionner. On obtient à ce point de l'extraction, deux ensembles des séparateurs classés comme *séparateurs début* et *séparateurs fins*, notés respectivement *D* et *F*. Freitag également définit *D* et *F* comme des ensembles de *détecteurs* [Freitag & Kushmerick, 2000]. Un détecteur est une paire de séquences de tokens < *p, s*>.

Un détecteur s'assortit à un séparateur dans le document si, et seulement si, le *préfixe* de la séquence de tokens, *p*, correspond à des tokens avant le séparateur et le *suffixe* de la séquence de tokens, *s*, correspond à des tokens après le séparateur. Par exemple, le détecteur <Who: , Dr.> peut s'assortir au fragment de texte « Who: Dr. John Smith » définissant le séparateur entre ':' et 'Dr.'. Dans ce contexte, soient des détecteurs de début et fin, l'extraction est réalisée en identifiant le début et la fin d'un champ et en prenant les tokens entre ces deux points. La figure suivante illustre un exemple d'un *wrapper* avec ses détecteurs *D* et *F* appris.

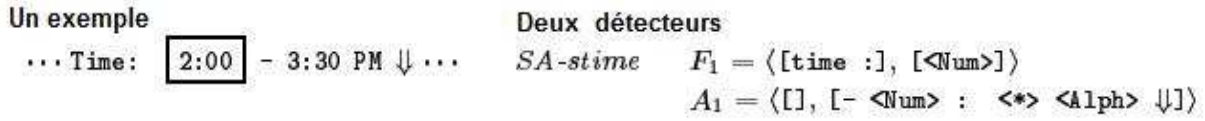



**Fig. 16.** Deux détecteurs (début et fin) pour le slot « stime » du corpus Seminars [Freitag & Kushmerick, 2000]. Le symbole ⇓ signifie une nouvelle ligne.

Finalement pendant l'étape (3), BWI enregistre les longueurs des séquences de tokens d'un champ observées sur les données d'apprentissage. Deux détecteurs, 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. En fait, l'algorithme apprend la distribution de la probabilité (histogramme $H_{d,f}$) de la longueur des champs à extraire rencontrés dans l'ensemble d'apprentissage. En d'autres termes, on peut associer à chaque détecteur de début et fin, une valeur numérique de confiance $C_{dd}$ et $C_{df}$, respectivement. Pour effectuer une extraction en utilisant un extracteur (wrapper) $W = \langle F, A, H \rangle$, on attribue à chaque détecteur *i* dans le document un score « fore » $F(i) = \sum_k C_{F_k} F_k(i)$ et un score « aft » $A(i) = \sum_k C_{A_k} A_k(i)$. Le wrapper *W* classifie une séquence de tokens comme :

$$W(i, j) = \begin{cases} 1, \text{ si } F(i)A(j)H(j-i) > \tau \\ 0, \text{ autrement} \end{cases}$$

Où $\tau$ est une valeur seuil qui représente la préférence entre la précision et le rappel.

La raison est que W compare $\tau$ avec l'estimation de la probabilité d'une classification correcte. En faisant varier $\tau$ on peut ajuster la préférence entre précision et rappel, où $\tau = 0$ désigne la configuration avec le maximum rappel possible et $\tau = 1$ le maximum de précision.

#### 3.1.4.5 Limitations de BWI

D'après les résultats montrés par [Kauchak et al., 2002], l'apprentissage de l'algorithme BWI et des méthodes similaires peuvent avoir de honorables performances quand elles sont appliquées sur différents types de *corpora*. Cependant, il identifie deux majeures limitations de l'algorithme BWI :

*a) Expressivité limitée de séparateurs*

Les séparateurs sont conçus pour capturer le contexte plat et voisin d'un champ à extraire en apprenant de courtes séquences de tokens qui les entourent. BWI obtient une haute précision grâce à la capture de l'information séquentielle et régulière autour de la donnée à extraire. Toutefois, pour des textes partiellement structurés et en langage naturel, les séparateurs ne sont pas aussi efficaces à cause de régularités dans le contexte qui sont moins consistant et fiables. Normalement, dans ces types de documents, de nombreux séparateurs couvrent seulement un ou quelques exemples, et collectivement, les séparateurs peuvent avoir un faible rappel.

Une seconde limitation des séparateurs est qu'ils ne peuvent pas représenter la structure grammaticale des phrases, ou la plupart des informations structurales dans les documents libre. La raison de cette limitation est qu'ils ne peuvent que capturer les informations sur le tokens qui apparaissent *avant* ou *après* un champ cible dans la séquence linéaire de tokens dans un document. En particulier, BWI ne peut représenter ou apprendre aucune information sur les nœuds de parent, les enfants de mêmes parents, ou la position d'enfant dans l'arbre d'un document XML, auquel les champs cible appartiennent implicitement.

*b) L'apprentissage lent*

L'apprentissage d'un seul champ avec quelques centaines de documents peut prendre plusieurs heures, même sur un ordinateur performant. La lenteur dans l'apprentissage rend l'utilisation de grandes valeurs pour le paramètre *lookahead L*, aussi prohibitif.

Pour Kauchak [Kauchak et al., 2002] cette lenteur est due à la boucle la plus interne de l'algorithme d'apprentissage chargée de trouver des extensions pour les séparateurs. Toutes les combinaisons de motifs possibles d'au plus *L* tokens juste avant e après un séparateur quelconque sont considérées jusqu'à ce qu'aucune meilleure règle ne puisse être trouvée. Trouver une extension de séparateurs est exponentiel en *L* parce que chaque combinaison de tokens est énumérée et évaluée, même des valeurs modestes de *L* sont coûteuses.

Dans [Freitag & Kushemerick, 2000] la valeur $L = 3$ a été normalement employée pour obtenir un équilibre entre l'efficacité et la performance. Les auteurs déclarent également que $L = 3$ est généralement suffisant, mais pour certaines tâches, une valeur jusqu'à $L = 8$ peut se faire nécessaire pour atteindre les meilleurs résultats.

## 3.2 Autres systèmes d'induction supervisée d'extracteurs

Cette section présente brièvement d'autres systèmes d'induction supervisée d'extracteurs. Pour une étude plus détaillée des systèmes d'EI, nous renvoyons le lecteur à [Kushmerick &

Thomas, 2003; Siefkes & Siniakov, 2005; Chang et al., 2006; Tang et al., 2007]. Les systèmes examinés sont tous des systèmes d'EI qui s'appliquent aux documents HTML et XML.

Il faut rappeler qu'un système d'EI supervisé prend en entrée des documents dans lesquels les informations à extraire sont annotées. Il fournit en sortie l'extracteur induit au cours d'un processus d'apprentissage à partir d'exemples positifs.

Les sections suivantes présentent les systèmes *WIEN, SoftMealy, STALKER, Amilcare, SIE et TIES.*

### 3.2.1 WIEN

Le système WIEN [Kushmerick, 1997, 2000], développé par Kushemick au *Departement of Computer Science* de l'Université de Washington en 1997, a été le premier système d'induction d'extracteurs. Ce système traite les documents HTML annotés comme des chaînes de caractères. Il est défini comme un vecteur $<l_1, r_1, ..., l_k, r_k>$ de 2K délimiteurs, où chaque paire $<l_i\ R_i>$ correspond à un type d'information et par la fonction *extraireLR* [Kushmerick, 1997]. Cette fonction prend en entrée un extracteur $W = ((l_1, r_1), . . ., (l_n, r_n))$ et un document *d* et applique l'extracteur *W* au document *d*.

L'extraction d'un champ d'information est réalisée en repérant ses délimiteurs gauche et droit. Un délimiteur est une séquence de caractères qui se trouve soit avant la donnée à extraire, dans ce cas, on parle de délimiteur gauche et on le note *l* ; soit après elle, il s'agit alors d'un délimiteur droit, noté *r*. Ainsi, une donnée à extraire (composante) est représentée par ses indicateurs de début *b* et de fin *e* dans la séquence de caractères du document.

On trouve dans WIEN 6 classes d'extracteurs à base de délimiteurs où la plus simple d'entre elles est la classe *LR*. Les extracteurs de la classe *LR* exigent alors que tous les délimiteurs indiquent correctement les limites à gauche et à droite des segments à extraire. Les autres 5 classes d'extracteurs sont : *HLRT, OCLR, HOCLRT, NLR et NHLRT*. Nous renvoyons le lecteur à [Kushmerick, 1997] afin de connaître en détail ces autres classes d'extracteurs.

Un exemple d'un extracteur défini dans le WIEN est illustré par la Fig. 17. On veut extraire « Country » et « Area Code » de deux pages HTML : D1 et D2.

```
D1: <B>Congo</B> <I>242</I><BR>
D2: <B>Egypt</B> <I>20</I><BR>

Rule: *'<B>'(*)'</B>'*'<I>'(*)'</I>'
Output: Country_Code {Country@1}{AreaCode@2}
```

**Fig. 17.** Exemple d'induction d'extracteurs et extraction dans WIEN.

La règle de la Fig. 17 a la signification suivante : ignorer tous les caractères jusqu'à ce qu'il soit trouvée la première occurrence de ' <B> ' et extraire le nom du pays comme une chaîne qui se termine par le premier '</B>'. Ensuite ignorer tous les caractères jusqu'à ce que soit trouvé '<I> ' et extraire la chaîne qui se termine par ' <I>'. Afin d'extraire des informations sur *Country* et *Area Code*, la règle est appliquée de façon répétée jusqu'à ce

qu'elle ne s'applique plus. De cet exemple, on note qu'une règle de WIEN peut être appliquée avec succès sur les deux documents: *D1* et *D2*.

Cependant WIEM présente certaines limitations qui ont été soulignées par [Hsu, 1998]. Tout d'abord, une première limitation est liée au fait que WIEN fait une hypothèse forte sur l'organisation des tuples dans les documents : les tuples sont consécutifs, et leurs composantes sont obligatoirement dans le même ordre. Cette hypothèse est vérifiée lorsque les données à extraire sont présentées dans une table car l'ordre des colonnes est naturellement fixe et identique dans toute la table. Cependant elle n'est plus nécessairement lorsque les n-uplets sont dans une liste dans laquelle l'ordre des composantes peut changer d'un tuple à l'autre. Ensuite, une autre limitation est que dans WIEN, les valeurs manquantes ne sont pas gérées convenablement en extraction. Ainsi, l'induction d'un extracteur est tout simplement impossible lorsque les documents contiennent des tuples dont certaines valeurs sont manquantes.

### 3.2.2 SoftMealy

Similaire à WIEN, SoftMealy est un système d'EI qui produit des règles d'extraction en utilisant un type spécial d'automates appelés *Finite State Transducers* (transducteurs à états finis) [Hsu, 1998]. Un transducteur à états finis comporte des alphabets d'entrée/sortie, des états et des transitions. Hsu et Dung ont présenté l'idée de transducteurs à états finis pour permettre plus de variation sur les structures d'extracteurs conçues par SoftMealy.

Dans SoftMealy, un transducteur à états finis est composé d'un « body transducer », qui extrait la partie de la page qui contient les tuples (similaire à HLRT dans WIEN), et de plusieurs *transducteurs de tuples* qui extraient itérativement les tuples à partir de la portion de textes définies par le body transducer. Les transducteurs de tuples acceptent un tuple et renvoient ses attributs. Ils produisent une sortie, un mot de l'alphabet de sortie, en fonction de la lecture d'un mot en entrée et de l'état dans lequel ils se trouvent.

L'algorithme d'extraction de SoftMealy est conçu dans un esprit similaire à celui des extracteurs HLRT de WIEN. Il fait intervenir deux transducteurs : un pour déterminer la zone du document qui contient les n-uplets et un autre pour extraire les n-uplets de ladite zone. Chaque permutation distincte d'attribut dans la page peut être encodée comme un chemin partant d'un état de début à un état final d'un transducteur de tuples ; et les transitions d'état sont déterminées en assortissant les règles contextuelles qui décrivent le contexte délimitant deux attributs adjacents. Les règles contextuelles se composent des différents séparateurs qui représentent les frontières invisibles entre des tokens adjacents et un algorithme de généralisation inductif est employé pour induire ces règles depuis les exemples d'entraînement annotés.

Avant d'extraire des données d'un document, le système segmente la page HTML d'entrée en tokens. Le transducteurs à états finis qui en résulte prend une séquence de tokens comme entrée et assortit les séparateurs de contexte avec des règles contextuelles pour déterminer des transitions d'état. Un FST peut être construit pour chaque type de tuples dans un document.

### 3.2.3 STALKER

STALKER est un système d'extraction multi-slot reposant sur *n* extracteurs single-slot. Il utilise un formalisme appelé *Embedded Catalog Tree* (ECT) [Muslea et al, 1998] pour représenter un schéma de sortie. ECT est une structure arborescente, où les nœuds-feuilles

sont des données à extraire et les nœuds non-feuilles sont des listes de tuples, dont les composantes représentent soit un nœud feuille, soit un nœud liste. Cet arbre décrit l'organisation logique des données dans le document, mais également un processus d'extraction hiérarchique. En plus, le formalisme ECT guide le processus d'extraction d'informations multi-slot, reliant par le biais de sa structure hiérarchique les données extraites.

La racine contient la séquence de tokens *S* du document dans son intégrité, et chacun de ses fils à une sous-séquence de *S*. Un document est vu par ce système comme une séquence de tokens. Un token est une séquence de caractères alphanumérique, non- alphanumériques, ou une balise HTML.

L'algorithme d'induction consiste à déterminer les règles d'extraction des nœuds de l'arbre EC fourni par l'utilisateur. Les valeurs des n-uplets annotés par l'utilisateur sont reliées aux feuilles adéquates de l'arbre. Cet apprentissage est fait à partir de documents annotés. Il consiste à apprendre les règles d'extraction en utilisant un algorithme de couverture spécifique [Muslea et al., 2001].

L'algorithme d'extraction procède en parcourant l'arbre et en appliquant à chaque nœud la règle d'extraction. La séquence de tokens extraite est le point de départ de la règle d'extraction suivante. Il existe pour chaque nœud de l'ECT une règle single-slot spécifique, formée par une paire d'automates finis non-déterministes pour la localisation des délimiteurs gauche et droit des nœuds du document d'entrée. Cette localisation se produit de manière indépendante de ses nœuds voisins ce qui facilite ainsi la manipulation d'attributs désordonnés et/ou absents.

STALKER spécifie deux types de règles : celles pour l'extraction des nœuds feuille et celles pour l'itération de listes pour les nœuds non-feuille. Les bases de ces règles d'extraction sont les commandes *SkipTo* et *SkipUntil* qui définit la transition de l'automate d'un état A à l'état B à partir de la localisation d'un *landmark l* : A → B. Un landmark est une séquence de tokens et/ou classes sémantiques dans un document. Parmi les classes sémantiques qui sont prédéfinies par le système, on peut citer des classes : numérique, alphanumérique, alphabétique (en majuscules), des balises HTML, ponctuation, etc. ; ou celles définies par l'utilisateur lui-même, telles que nom, email, numéro de téléphone, etc. La règle suivante, illustré par la Fig. 18, sert à localiser un délimiteur gauche du slot <résumé> en HTML.

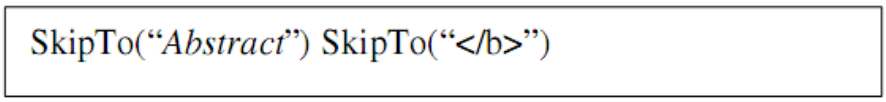



**Fig. 18.** Exemple d'une règle par conjonction [Cabral, 2004].

Cet exemple est constitué par la combinaison de deux *SkipTo* commandes. La première commande ira ignorer tous les caractères à l'intérieur d'un nœud parent, jusqu'à ce qu'elle trouve le token « Abstract», en s'arrêtant au prochain token. De ce point, la seconde commande *SkipTo* va effectuer la même procédure pour trouver "</b>". Dans le cas de règles pour des délimiteurs du côté droit, l'idée continue la même, en se modifiant seulement l'ordre qui tokens seraient cherchés, c'est-à-dire, de droite à gauche à l'intérieur d'un nœud parent.

Dans le cas de l'itération de listes, le système applique une règle d'extraction pour identifier les délimiteurs gauche et droite de la liste, de façon similaire à celle pour l'identification de nœuds feuille. Ensuite, les règles de nœuds-fils pour les délimiteurs gauches sont appliquées itérativement par toute l'extension de la liste, de façon à trouver tous ses items. La même procédure est effectuée vers la direction opposée, avec les délimiteurs droits

des nœuds fils. Différemment à WIEN et SoftMealy, les règles d'extractions de STALKER peuvent exprimer des disjonctions.

La principale limitation de STALKER est de déléguer la procédure de réassociation des valeurs des composantes en n-uplets à l'utilisateur, qui se voit alors chargé de la construction de l'arbre EC fourni à l'algorithme d'induction. L'imposition de la conception de cette procédure à l'utilisateur est contraire à l'esprit de l'induction d'extracteurs à partir d'exemples annotés.

Une mise en œuvre du STALKER a été développée en [Cabral, 2004] en utilisant Document Object Model (DOM) pour implémenter le formalisme ECT.

### 3.2.4 Amilcare

Amilcare [Ciravegna, 2003a] est un système d'induction de règles d'extraction développé par Fabio Ciravegna de l'Université de Sheffield. Il repose sur l'algorithme $(LP)^2$ (*Learning Pattern by Language Processing*), un algorithme supervisé qui appartient à la classe de systèmes d'induction d'extracteurs utilisant *LazyNLP* [Ciravegna, 2001]. LazyNLP, aussi nommé *Shallow NLP*, faire l'usage de TAL pour généraliser des règles d'extraction au-delà de la structure plate de mots, tout en conservant l'efficacité sur les textes fortement structurés.

Ce système essaie d'apprendre le meilleur (le plus fiable) niveau d'analyse linguistiques utile pour une tâche spécifique d'EI en combinant des stratégies peu ou très profondes d'analyse linguistiques. L'apprenant commence à induire des règles qui ne font aucune utilisation de l'information linguistique, comme dans les systèmes d'EI classiques. Puis, il ajoute progressivement l'information linguistique à ses règles, s'arrêtant quand l'utilisation de telles informations devient incertaine ou inefficace.

Les modules de TAL fournissent des informations linguistiques et de ressources définies une fois pour toutes, ils ne peuvent pas être modifiés par l'utilisateur lors de son emploie dans une application spécifique.

Les apprenants LazyNLP apprennent la meilleure stratégie pour chaque information/contexte séparément. Cela s'est révélé très efficace pour l'analyse de documents avec un mélange de genres, par exemple, les pages Web contenant à la fois du matériel structuré et non structuré, ce qui est fréquent dans des documents Web [Ciravegna & Lavelli, 2001].

L'architecture d'Amilcare est liée avec ANNIE, un système de TAL de l'environnement GATE. ANNIE effectue la tokenisation, l'étiquetage morphosyntaxique (*POS tagging*), *gazetteer lookup* (dictionnaires du domaine) et la reconnaissance d'entités nommées (*Named Entity Recognition*).

Les règles d'Amilcare sont apprises par la généralisation d'un ensemble d'exemples trouvés dans un corpus d'apprentissage annoté avec des balises XML. Le système apprend comment reproduire une telle annotation par l'extraction d'information.

Trois types de règles sont définis dans l'Amilcare : *règles d'étiquetage (tagging rules)*, *règles contextuelles (contextual rules)* et *règles de correction (correction rules)*. Une règle d'étiquetage se compose d'un ensemble de conditions sur une séquence de mots reliés, et d'une action déterminant si la position actuelle est un délimiteur d'une instance. La Fig. 19 montre un exemple d'une telle règle. La première colonne représente une séquence de mots. De la seconde à la cinquième colonne de cette figure représentent, respectivement, étiquetage POS, le type de mot, *gazeteer lookup* et la classe des entités nommées à laquelle le token appartient. La dernière colonne représente l'action. L'action « <Speaker> » indique que si le texte s'assortie avec le patron, le token « Patrick » sera identifié comme un délimiteur de début de « speaker ».

| Pattern | | | | | Action |
|---|---|---|---|---|---|
| **Word** | **POS** | **Kind** | **Lookup** | **Name Entity** | |
| ; | : | Punctuation | | | |
| Patrick | NNP | Word | Person's first name | Person | <Speaker> |
| Stroh | NNP | Word | | | |
| , | , | Punctuation | | | |
| assistant | NN | Word | Job title | | |
| professor | NN | Word | | | |
| , | , | Punctuation | | | |
| SDS | NNP | Word | | | |

**Fig. 19.** Exemple d'une règle étiquetage (*taging rule*) [Tang, 2007].

Les règles d'étiquetage sont induites de la façon suivante : tout d'abord, une étiquette du corpus d'apprentissage est sélectionnée, une fenêtre de *w* mots à gauche et *w* à droite est prise en suivant le patron initial de la règle. Ensuite, toutes les règles initiales sont généralisées. Par exemple, en utilisant TAL, les deux règles « *at 4 pm* » et « *at 5 pm* »peuvent être généralisées par « *at DIGIT pm* ». Chaque règle généralisée est testée dans le corpus d'apprentissage et un score d'erreur est calculé. Finalement, les meilleurs *k* généralisations pour chaque règle initiale sont conservés dans le groupe de meilleures règles. Cet algorithme est également utilisé pour induire les deux autres types de règles. La Fig. 20 indique une règle d'étiquetage généralisée pour l'identification du délimiteur initial de « Speaker ».

Des règles contextuelles sont appliquées pour améliorer la performance du système. L'idée principale est qu'un <$tag_x$> pourrait être utilisé comme un indicateur de l'apparition de <$tag_y$>. Considérons, par exemple, une règle qui reconnaît un délimiteur final entre un mot avec une majuscule initiale et un mot avec toutes les lettres minuscules. Cette règle ne fait pas partie du groupe des meilleures règles en raison de sa faible précision dans le corpus, mais elle est fiable seulement si elle est utilisée lors de la fermeture d'un délimiteur <speaker>.

| Pattern | | | | | Action |
|---|---|---|---|---|---|
| **Word** | **POS** | **Kind** | **Lookup** | **Name Entity** | |
| ; | : | Punctuation | | | |
| | | Word | Person's first name | Person | <Speaker> |
| | | Word | | | |
| | | Punctuation | | | |
| assistant | NN | Word | Jobtitle | | |
| professor | NN | Word | | | |

**Fig. 20.** Exemple d'une règle d'étiquetage généralisé [Tang, 2007].

Enfin, les règles de correction sont identiques aux règles d'étiquetage, mais ses patrons assortissent aussi des balises insérées par des règles d'étiquetage et ses actions tout simplement déplacent les balises mal positionnées plutôt que d'ajouter de nouvelles règles. Un exemple d'une règle de correction initiale pour le déplacement de </stime> dans « at <stime> 4 </stime> pm » est illustré par la figure suivante.

| condition | | action |
|---|---|---|
| word | wrong tag | move tag to |
| at | | |
| 4 | *</stime>* | |
| pm | | *</stime>* |

**Fig. 21.** L'action déplace la balise de la mauvaise position à la bonne [Ciravegna, 2001].

### 3.2.5 SIE (Simple Information Extraction)

SIE (*Simple Information Extraction*) [Giuliano et al., 2006] est un système d'EI fondé sur une technique d'apprentissage supervisé où la tâche d'EI est traité comme un problème de classification en appliquant des *Machines à Vecteurs de Support* ou *Séparateur à Vastes Marges* (*SVM*) [Cortes & Vapnik, 1995], pour établir un ensemble de classificateurs qui détectent les délimiteurs d'entités à extraire. Il a été conçu avec le but d'être rapidement portable à différents domaines. Une série d'expérimentations sur plusieurs domaines ont prouvé que SIE est comparable en performance aux systèmes déjà introduits dans l'état de l'art, et plus performant que quelques systèmes qui ont été conçus pour des domaines spécifiques. En outre, ce système a été testé sur différents domaines en utilisant la même configuration de base sans employer aucune connaissance spécifique, telles que des *gazetteers* et pré/post-traitement *ad hoc*.

Une caractéristique majeure de SIE est de réduire l'effort du traitement de tokens en exploitant une nouvelle technique nommée *Instance Pruning* [Gliozzo et al, 2005] (filtrage d'instances) qui permet le traitement de nombreux documents en appliquant un filtrage sur les tokens d'un document. Cela s'est avéré très efficace pour le traitement des données en bioinformatique.

SIE a une architecture modulaire où les composants du système sont combinés en *pipeline*, où chaque module restreint les structures de données fournies par les modules précédents. Cette architecture modulaire apportent des avantages significatifs : elle est d'abord plus simple à implémenter, ensuite elle permet d'intégrer facilement différents algorithmes d'apprentissage, enfin, elle permet, si nécessaire, un réglage fin à une tâche spécifique, en spécialisant simplement quelques modules.

L'architecture du système est illustrée par la Fig. 22. La tâche d'EI est réalisée en deux phases : dans la première, le système fait des hypothèses sur un ensemble de modèles d'apprentissage à partir d'un corpus étiqueté ; dans la seconde, ces modèles sont appliqués pour étiqueter de nouveaux documents. Dans ces deux phases, le module de filtrage d'instances est utilisé pour enlever certains exemples (*tokens*) du corpus en question afin d'accélérer le processus entier, tandis que le module d'extraction d'attributs (*Feature Extraction*) est utilisé pour l'extraction d'un ensemble prédéfini d'attributs à partir des exemples.

Lors de l'étape de classification, le module *Tag Match* est utilisé pour combiner les prévisions du module classificateur. Tous les modules sont implémentés en langage Java, à l'exception des composants aux tiers. Pour voir une description complète de chaque module présent dans la figure ci-dessous, nous renvoyons le lecteur à [Giuliano et al., 2006].

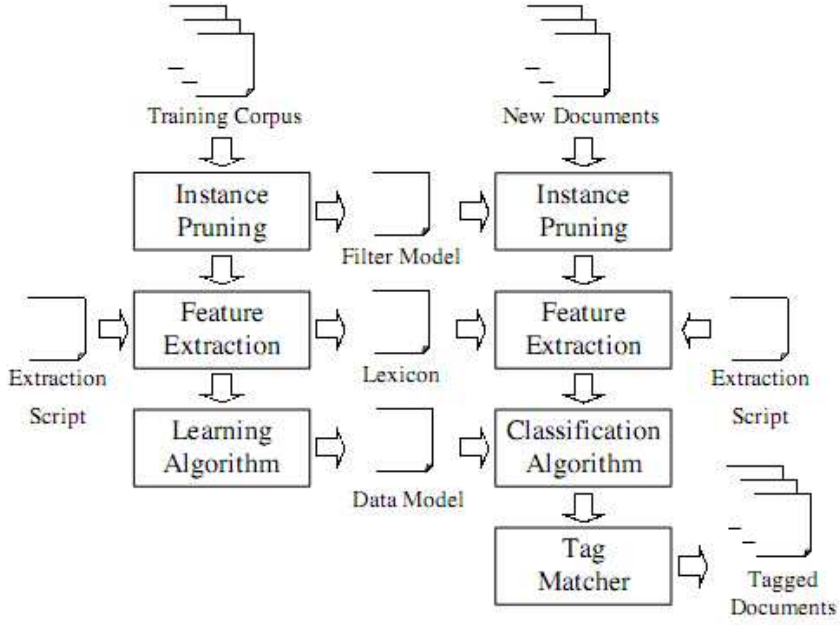



**Fig. 22.** Aperçu du système SIE.

### 3.2.6 TIES (Trainable Information Extraction System)

Le système TIES (*Trainable Information Extraction* System) [ITC-IRST, 2004], développé en 2004 par le *Centro per la Ricerca Scientifica e Tecnologica* (ITC-irst), est un système d'EI adaptatif. Il a été développé en langage Java et il met en œuvre l'algorithme *Boosted Wrapper Induction* (BWI) proposé par Dayne Freitag et Nicholas Kushmerik [Freitag & Kushemerick, 2000] (section 3.2.2).

L'algorithme d'apprentissage de BWI utilise lui-même l'algorithme de boosting *AdaBoost* pour générer des extracteurs qui combine un ensemble spécifique d'inducteurs d'extracteurs à partir de documents annotés. L'algorithme BWI a démontré être très performant dans de nombreuses tâches d'EI à partir de documents structurés et semi-structurés. De plus, d'après [Kauchak et al., 2002] l'utilisation de l'algorithme BWI donne aussi de bons résultats pour le traitement de documents non structuré, par exemple des textes en langage naturel.

TIES étiquète les documents avec un ensemble de balises XML prédéfinies, en exploitant des règles apprises automatiquement grâce à un corpus préalablement annoté. Ces balises XML permettent d'identifier les instances d'entités qui proviennent d'un ensemble d'éléments définis par l'utilisateur. Ce système a été retenu dans cette recherche et sera présenté en détail dans la section 4.2.

### 3.2.7 Tableau de synthèse

Un résumé des caractéristiques majeures de tous les systèmes d'EI par induction supervisée d'extracteurs, examinés dans cette section est présenté dans le tableau suivant.

**Tab. 1.** Résumé comparatif de caractéristiques des systèmes.

| ***Système*** | ***Type de documents*** | ***Représentation de documents*** | ***Type d'extraction*** | ***Algorithme d'apprentissage*** | ***Faire l'usage de TAL*** |
|---|---|---|---|---|---|
| **WIEN** | Semi structuré (HTML) | Token | unaire | Induction d'extracteurs | Non |
| **SoftMealy** | Semi structuré (HTML) | Token | unaire | Transducteurs à état finis | Non |
| **STALKER** | Semi structuré (HTML) | *Embedded Catalag* (arborescence) | n-aire | Algorithme de couverture ad-hoc | Non |
| **Amilcare** | Texte Libre | Token | unaire | Algorithme de couverture séquentiel | Oui (*LazyNLP*) |
| **SIE** | Texte libre | Token | unaire | Classification Supervisée (SVM) | Non |
| **TIES** | Texte libre[5] | Token | unaire | Classification supervisée (*Adaboost*) | Non[6] |

Ce tableau montre que la plupart de ces systèmes étudiés adoptent une représentation de document à base de tokens (sauf STALKER). Excepté STALKER, ces systèmes font de l'extraction unaire (single slot). Ils mettent en œuvre différents algorithmes d'apprentissage pour traiter des textes semi-structurés, voire libre pour certains de ces systèmes (SI, TIES, Amilcare).

## 3.3 Conclusion

Dans ce chapitre, tout d'abord il a été étudié la classification supervisée sur laquelle repose la plupart des techniques d'EI par induction supervisée d'extracteurs. Ainsi un extracteur a été défini comme un processus de classification où le problème d'induction d'extracteurs se ramène alors à un problème d'apprentissage de classification supervisée.

[5] TIES accepte des textes libres à condition qu'ils soient constitués comme documents XHTML ou XML.
[6] La deuxième version du TIES décrite dans ce mémoire utilise tagage POS (section 4.3).

La transformation d'un problème d'extraction en un problème de classification supervisée nécessite que l'on spécifie la représentation de documents, la définition des éléments de documents qui seront considérés comme les exemples du problème de classification, et le codage de ces exemples. En étudiant en détail un système d'induction d'extracteurs reposant sur la classification supervisée (BWI), il a été illustré les différentes étapes élémentaires associées à une telle transformation.

Ensuite, différents systèmes d'EI par induction supervisée d'extracteurs ont été présentés, notamment les systèmes WIEN, SoftMealy, STALKER, Almicare, SIE et TIES. Ces différents systèmes presque tous adoptent une représentation de document à base de tokens, font de l'extraction unaire, mettent en œuvre différents algorithmes d'apprentissage pour traiter des textes semi-structurés, voire libre pour certains de ces systèmes (SIE, TIES[7] Amilcare).

Dans le chapitre suivant ce travail propose un système d'EI adaptatif modulaire permettant l'extraction d'information de documents semi-structurés ou libres, reposant sur l'induction supervisée d'extracteurs et intégrant le système TIES mettant lui-même en œuvre l'algorithme BWI.

[7] Brièvement présenté dans la section 3.3.6

# 4 UN SYSTEME D'EI ADAPTATIF PAR INDUCTION SUPERVISEE D'EXTRACTEURS

Dans ce chapitre nous proposerons un système d'EI adaptatif modulaire permettant l'extraction d'information de documents semi-structurés ou libres. Ce système, reposant sur l'induction supervisée d'extracteurs, est composé de divers modules permettant la préparation d'un ensemble d'apprentissage annoté manuellement par un utilisateur - ensemble pouvant aussi intégrer l'étiquetage morphosyntaxique permettant le traitement de la syntaxe du langage naturel - et un module réalisant l'induction supervisée d'extracteurs. Pour ce dernier module, le système TIES mettant lui-même en œuvre l'algorithme BWI, a été retenu. Il sera détaillé dans la section 4.2 et modifié afin de pouvoir traiter l'étiquetage morphosyntaxique.

Pour obtenir de meilleurs résultats sur des collections de documents moins structurés, il a fallu modifier l'architecture de TIES pour qu'il puisse traiter des documents en texte libre annotés par étiquetage morphosyntaxique fournie par une analyse de Parties du Discours (*Part of Speech*).

Ainsi, dans un premier temps, nous présenterons chacun des différents modules ou composants du système d'EI proposé. Ensuite, les nouveaux modules de l'architecture de notre solution seront présentés : le module d'annotation de documents XHTML, le module de validation des pages de *corpora* qui les transforme en fichiers XML bien formés et, enfin, le module en charge de l'étiquetage morphosyntaxique (POS tagging).

## 4.1 Architecture générale d'un système d'EI adaptatif

L'architecture classique d'un système d'EI adaptatif par l'induction supervisée d'extracteurs est illustrée par la Fig. 23. C'est une architecture modulaire en pipeline qui comporte un prétraitement (linguistique), les étapes d'apprentissage et d'application et, pendant l'étape d'application, un post-traitement sémantique comme les trois modules les plus importants (Fig. 23). Un corpus de documents comprenant les textes du domaine d'application et une structure cible (*template*) qui définit les informations pertinentes à extraire constituent l'entrée minimale pour un système d'IE.

### 4.1.1 Prétraitement des textes d'entrée

Un corpus est souvent constitué de textes non structurées, en langage naturel. Une grande partie des informations pertinentes peut être caractérisée par une certaine régularité trouvée dans les propriétés linguistiques des textes. Par conséquent, l'analyse linguistique peut donner des suggestions utiles et déterminer des attributs importants pour identifier le contenu pertinent. Les composants linguistiques suivants se sont révélée utiles pour EI: tokenisation, étiquetage morphosyntaxique (*POS tagging*), reconnaissance des entités nommées (*NER*), segmentation en phrases (*Sentence Spliting*), reconnaissance de structures de phrases grammaticaux (*Chunking*), et enfin résolution d'anaphores (*Coreference Resolution*).

### 4.1.2 Apprentissage et application du modèle d'extraction

Les systèmes d'EI modernes utilisent un composant d'apprentissage pour réduire la dépendance de domaines spécifiques et pour diminuer la quantité de ressources fournies par un humain. Un modèle d'extraction est défini en fonction de la démarche à suivre et ses paramètres sont «appris» (optimisés), par une procédure d'apprentissage. Les approches statistiques apprennent, par exemple, les attributs de classement les plus pertinents, des probabilités, des séquences d'états ; des approches reposant sur règles apprennent un ensemble de règles d'extraction et les approches reposant sur connaissance acquièrent des structures pour augmenter et interpréter leur connaissance pour l'extraction. Le défi est de trouver un modèle d'extraction permettant l'apprentissage de tous les paramètres en utilisant la même plate-forme d'extraction pour chaque domaine d'application.

### 4.1.3 Post-traitement de la sortie

L'une des possibilités pour structurer les données extraites est de modeler le *template* en tant qu'une relation de base de données. Après que les informations importantes aient été trouvées par l'application d'un modèle d'extraction, les extraits identifiés des textes sont assignés aux attributs correspondants de la structure cible.

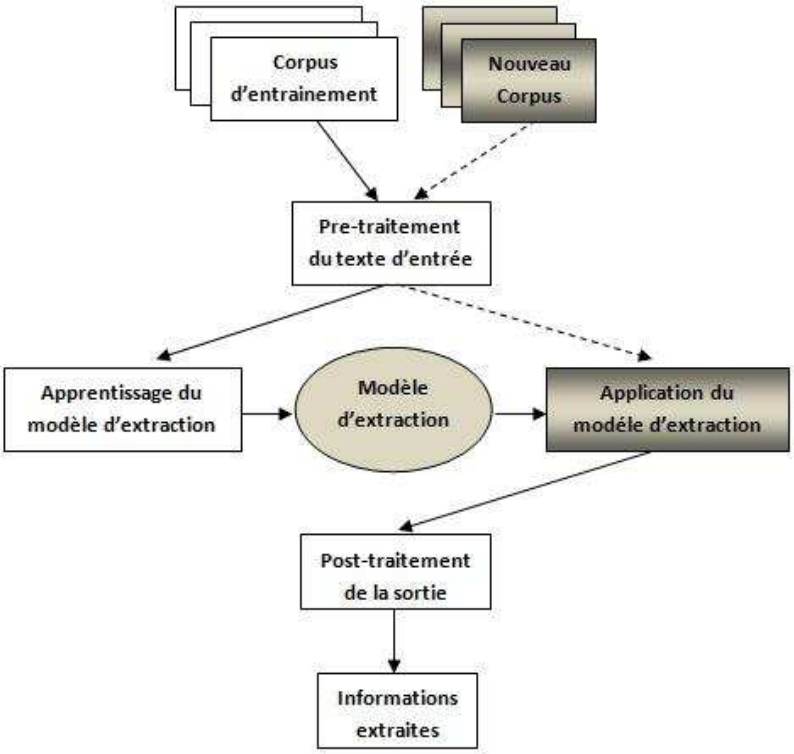



**Fig. 23.** Architecture d'un système d'EI adaptatif.

## 4.2 TIES : un système d'induction supervisée d'extracteurs

Le système d'induction supervisée d'extracteurs proposé par le présent travail est constitué par le système TIES [ITC-IRST, 2004], vu comme un module dans l'architecture générale de notre proposition (voir section 4.3). Ce dernier consiste d'un système d'EI adaptatif reposant sur des techniques d'apprentissage supervisé.

Pour qu'un système d'EI soit considéré adaptatif, il doit, d'après [Ciravegna, 2003a]:

- s'adapter à de nouveaux domaines d'information : en employant (ou modifiant) leurs bases de connaissances, en concevant de nouveaux *templates* d'extraction, de sorte que les systèmes soient capables de manipuler des concepts du domaine en question ;
- s'adapter aux sous-langages des attributs : en modifiant des grammaires et des lexiques à fin de faire face aux constructions linguistiques spécifiques qui sont typiques à l'application ;
- s'adapter aux différents types de textes : les documents Web peuvent varier des documents rigidement structurés (par exemple des pages en XML et des tableaux) aux textes libres. Par conséquent, chaque type de texte peut avoir des exigences différentes en termes d'analyse du langage.

Le système TIES apprend automatiquement des règles à partir d'un corpus annoté avec un ensemble prédéfini de balises XML. Ces balises XML permettent d'identifier les instances d'entités qui proviennent d'un ensemble d'éléments définis par l'utilisateur. Le système TIES fournit un ensemble d'interfaces et de classes pour l'apprentissage, le test et l'application d'un modèle d'extraction aussi bien sur des textes libres que des textes fortement structurés. Il incorpore l'algorithme BWI pour l'apprentissage. L'implémentation de TIES par défaut emploie seulement des attributs orthographiques mais des attributs plus complexes (par exemple, des attributs morphosyntaxiques) peuvent être ajoutés pour améliorer les performances en utilisant un préprocesseur personnalisé. Un des objectifs de cette recherche sera d'intégrer un module traitant l'étiquetage morphosyntaxique (POS tagging) intégrée dans l'architecture originale du TIES.

Cette section montre comment la mise en œuvre de l'algorithme BWI par TIES s'applique dans le cadre de notre recherche. En effet, il est examiné en détail comment utiliser TIES pour générer des règles d'extraction à partir d'un corpora donné. Dans un premier temps, l'architecture originale de TIES est présentée, en précisant comment les documents sont y représentés. La procédure à suivre pour mettre en marche le système à l'aide de fichiers de configuration est ensuite expliquée, ainsi que des règles induites et leurs correspondantes informations extraites.

### 4.2.1 Description détaillée du système TIES (version originale)

L'architecture originale du système TIES est illustrée par la Fig. 24. On peut noter que cette architecture suit de près le modèle d'architecture générique présenté dans la section 4.1 de ce chapitre. Les sections suivantes détaillent chaque module en analysant leurs aspects les plus importants.

### 4.2.2 Représentation de documents

Dans TIES les documents d'un corpus qui seront traités doivent être sous un format spécifique nommé TIESIF (*TIES Input Format*). Ce format permettra la tokenisation et l'extraction d'information.

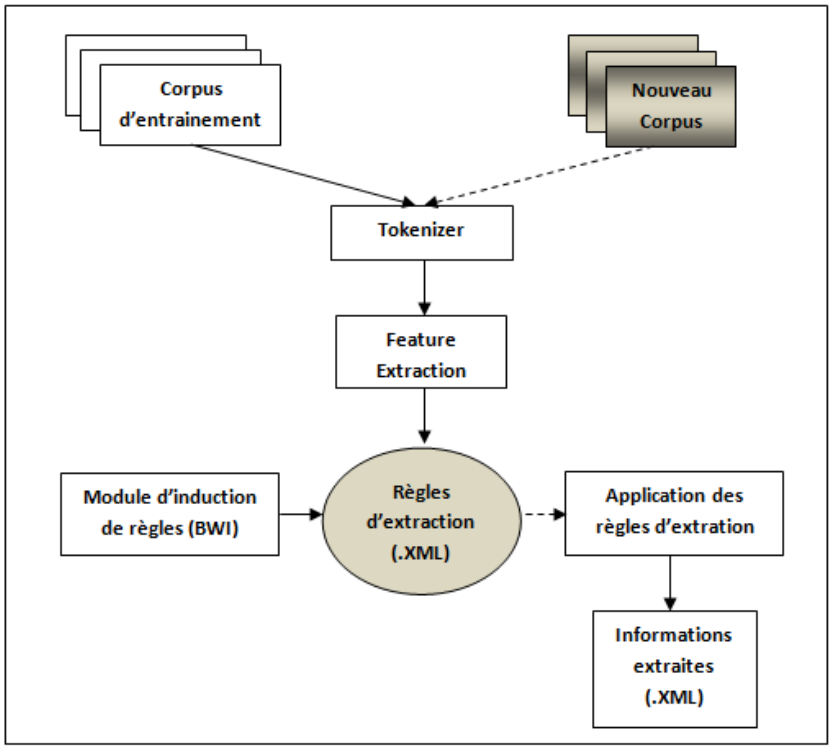



**Fig. 24.** Architecture originale du TIES.

### 4.2.2.1 Tokenisation

À partir d'une séquence de caractères, l'objectif est d'identifier les parties élémentaires du langage naturel : des mots, des signes de ponctuations et des séparateurs. La séquence résultante de tokens significatifs est la base pour tous les types de traitements linguistiques postérieurs. Ci-dessous un extrait d'un fichier du corpus *Seminars* déjà tokenisé avec des balises d'annotations <*speaker*> qui indique les exemples positifs et sa représentation en format TIESIF.

```
University of Pittsburgh
Name: <speaker> Dr. Jefrey D. Hermes </speaker>
Affiliation: Department
…
```

```
<?xml version="1.0"?>
<corpus>
  <text path="./input/seminar" name="cmu.andrew.org.cfa.cfa-today-19:0">
    <body>
      <token id="146" type="word" start="438" len="10" alpha_token="true"
                capitalized_token="true">University</token>
      <token id="148" type="word" start="449" len="2" alpha_token="true"
                lower_case_token="true">of</token>
      <token id="150" type="word" start="452" len="10" alpha_token="true"
                capitalized_token="true">Pittsburgh</token>
```

```
      <token id="151" type="nl" start="462" len="1"
                  nl_token="true">\n</token>
      <token id="153" type="word" start="464" len="4" alpha_token="true"
                  capitalized_token="true">Name</token>
      <token id="154" type="punct" start="468" len="1"
                  punct_token="true">:</token>
      <token id="156" type="tag" start="470" len="9"
                  open_tag="true">speaker</token>
      <token id="157" type="abbrev" start="479" len="3"
                  abbr_token="true">Dr.</token>
      <token id="159" type="word" start="483" len="7"

      alpha_token="true"capitalized_token="true">Jeffrey</token>
      <token id="161" type="abbrev" start="491" len="2"
                  abbr_token="true">D.</token>
      <token id="163" type="word" start="494" len="6" alpha_token="true"
                  capitalized_token="true">Hermes</token>
       <token id="164" type="tag" start="500" len="10"
                  close_tag="true">/speaker</token>
      <token id="165" type="nl" start="510" len="1"
                  nl_token="true">\n</token>
      <token id="167" type="word" start="512" len="11" alpha_token="true"
                  capitalized_token="true">Affiliation</token>
      <token id="168" type="punct" start="523" len="1"
                  punct_token="true">:</token>
      <token id="170" type="word" start="525" len="10" alpha_token="true"
                  capitalized_token="true">Department</token>
          ...
    </body>
  </text>
  <text>
  </text>
</corpus>
```

Nous renvoyons le lecteur au guide d'utilisateur de TIES [ITC-IRST, 2004] pour en savoir plus.

#### 4.2.2.2 Feature Extraction

Les exemples sont définis en termes d'attributs qui sont les résultats des fonctions appliquées sur les tokens. Ainsi, chaque token a une image $f(x)$ qui sont des valeurs discrètes. C'est-à-dire, un exemple est transformé en une collection d'attributs, produisant ainsi un vecteur $N$-dimensionnel. Chaque token est traité comme un exemple, de cette façon, on peut avoir un attribut *capitalized* qui associe un token à l'ensemble {*true, false*}. Étant donné cet attribut, nous pouvons exprimer de simples propositions sur un token spécifique : *capitalized("Home") = true et capitalized("work") = false.*

L'implémentation par défaut du tokeniseur du TIES encapsule un simple *feature extractor* de 12 attributs, à savoir :

- *alpha_token = true* pour des tokens qui contiennent seulement des caractères alphabétiques
- *num_token = true* pour un nombre
- *perc_token = true* pour des pourcentages
- *capitalized_token= true* pour des tokens qui commencent avec une lette en majuscule
- *lower_case_token= true* pour des tokens qui contiennent seulement lettres en minuscules
- *punc_token= true* pour des signes de ponctuation

- *upper_case_token = true* pour des tokens qui contiennent seulement lettres en majuscules
- *single_char_token true* pour un token composé d'un caractère
- *date_token= true* pour des dates
- *time_token= true* pour des heures
- *abbr_token= true* pour des abréviations
- *symb_token= true* pour des symboles

Le *feature extractor* génère seulement des attributs jugés actifs (true) dans un exemple. Tous les autres attributs sont jugé inactifs, ou false.

### 4.2.3 Configuration du système TIES

TIES est implémenté en utilisant un ensemble de modules. Les modules ont un certain nombre de propriétés paramétrables et ils implémentent une ou plusieurs interfaces. Ces modules peuvent être configurés de façon flexible en employant des fichiers XML. Chaque module est décrit par un élément XML avec des sous-éléments et des attributs utilisés pour définir les propriétés de modules. En spécifiant quels modules et leurs attributs à employer, on atteint à une grande flexibilité dans le contrôle des caractéristiques d'une instance de TIES.

#### 4.2.3.1 Étape d'apprentissage : génération d'un modèle

Le fichier de configuration pour l'étape d'apprentissage est le fichier TIES-CONFIG.XML. Dans ce fichier *<ties-config>* est l'élément principal. Il est constitué de plusieurs balises décrivant les modules du TIES pour étape de génération d'un modèle : *stratégie de validation, stratégie d'extraction, weak learner, boosting, tokeniseur et corpus loader*. Les directives de contrôle d'entrée et de sortie sont aussi mises dans ce fichier de configuration, ainsi que la mémoire cache (essentielle pour optimiser les performances du système et l'utilisation de la mémoire). Ci-dessous on voit un extrait du fichier de configuration TIES-CONFIG.XML relatif au corpus *CMU Seminar announcements*. Le manuel d'utilisateur [ITC-IRST, 2004] décrit en détail la signification de tous les paramètres définis par ce fichier.

```
<!-- Configuration file for the standard "CMU seminar announcements" extraction
task -->

<ties-config>
  <validation-strategy>
    <validation-
class>org.itc.irst.tcc.ties.validation.NFoldCrossValidation</validation-class>
    <init-param>
      <param-name>n</param-name>
      <param-value>10</param-value>
    </init-param>
    <init-param>
      <param-name>hypothesis-file</param-name>
      <param-value>./bwi/sa/out.xml</param-value>
    </init-param>
    <init-param>
      <param-name>eval-file</param-name>
      <param-value>./bwi/sa/bwi-eval.csv</param-value>
    </init-param>
  </validation-strategy>
       …
</ties-config>
```

#### 4.2.3.2 Étape d'application de règles : extraction

Le fichier de configuration pour l'étape d'extraction est le fichier CLASSIFIER-CONFIG.XML, dans lequel *<ties-config>* est l'élément principal. Il est constitué de plusieurs balises décrivant les modules du TIES employés pour l'extraction : *classificateur, stratégie d'extraction, tokeniseur et corpus loader*. Les directives de contrôle d'entrée et de sortie sont aussi mises dans le fichier de configuration, ainsi que la mémoire cache (toujours essentielle pour optimiser les performances du système et l'utilisation de la mémoire). Ci-dessous un extrait du fichier de configuration d'extraction CLASSIFIER-CONFIG.XML concernant le corpus *CMU Seminar announcements*. Le manuel d'utilisateur [ITC-IRST, 2004] contient des explications sur la façon de paramétrer le système.

```
<!-- Configuration file for the standard "CMU seminar announcements" extraction
task -->
<ties-config>
  <bwi>
    <classifier>
      <extract>
        <entity>
          <wrapper>./bwi/sa/new-out0-speaker.xml</wrapper>
          <output>./bwi/sa/res0-speaker.xml</output>
        </entity>
        <entity>
          <wrapper>./bwi/sa/new-out0-location.xml</wrapper>
          <output>./bwi/sa/res0-location.xml</output>
        </entity>
      </extract>
    </classifier>
      …
  <corpus-loader>
     …
  </corpus-loader>
</ties-config>
```

### 4.2.4 Exécution du système

TIES réalise deux tâches différentes : la tâche d'*apprentissage-test* et celle d'*extraction*. Les étapes à suivre pour ces tâches sont présentées ici d'une façon algorithmique.

*1- Apprentissage-test*

1. Instancier un algorithme d'EI
2. Spécifier un corpus d'entrée
3. Choisir une stratégie de validation
4. Choisir une stratégie d'extraction
5. Apprendre un modèle
6. Tester le modèle appris

*2 - Extraction*

1. Spécifier un nouveau corpus d'entrée
2. Exécuter un classificateur utilisant le modèle appris

Pour l'instant TIES ne disposant pas d'une interface conviviale de paramétrage, tous les paramètres de configuration doivent être saisis d'une des façons suivantes : invite de

commande, configuration XML ou API. Le lecteur est renvoyé au [ITC-IRST, 2004] pour savoir comment ajuster précisément les paramètres de configuration.

### 4.2.5 Règles induites et information extraites

#### 4.2.5.1 Règles induites

Les règles induites par TIES sont exprimées par des *wrappers* essentiellement formés par un *préfixe* suivi de l'information à extraire et se terminant par un *suffixe*. En fait, une règle détermine un *pattern* (patron de tokens) dans le corpus d'entrée qui entoure le slot d'information à extraire. Les *wrappers* appris sont stockés en format XML, et ils pourront être employés plus tard pendant une tâche d'extraction. La Fig. 25 montre un extrait d'un *wrapper* appris pour le slot <*speaker*> du corpus *CMU Seminar announcements* (sans traitement de POS) et la Tab. 2 présente leurs significations.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
<wrapper label="speaker">
<fore-detector>
    <detector>
      <pattern type="prefix">
        <feature name="token" value="Who"/>
        <feature name="single_char_token" value="true"/>
      </pattern>
      <pattern type="suffix">
        <feature name="alpha_token" value="true"/>
      </pattern>
    <confidence-value>2.7587264482323546</confidence-value>
    </detector>
    <detector>
      <pattern type="prefix">
        <feature name="token" value="speaker"/>
        <feature name="single_char_token" value="true"/>
      </pattern>
      <pattern type="suffix">
      </pattern>
      <confidence-value>2.2216808574759974</confidence-value>
    </detector>
  ...
</wrapper>
```

**Fig. 25.** Un extrait d'un *wrapper* appris en XML.

**Tab. 2.** Un extrait du *wrapper* appris après l'application d'une transformation XSL**.**

| | | |
|---|---|---|
| *1* | *P[token='Who', single_char_token='true']* | *S[alpha_token='true']* |
| 2 | *P[token='speaker', single_char_token='true']* | *S[]* |

Ainsi, la ligne 1 de la table ci-dessus exprime une règle, où le nom d'un *speaker* peut être trouvé juste après les tokens « Who » suivi d'un token constitué d'un seul caractère (préfixe) et juste avant un token formé par des caractères alphabétiques (suffixe). Il en va de même

pour la deuxième ligne qui exprime une règle qui n'a qu'un préfixe (« speaker » suivi d'un token formé par un seul caractère) et n'importe quel suffixe.

#### 4.2.5.2 Information extraites

Les entités extraites d'un nouveau corpus sont aussi stockés en format XML. La figure suivante montre ces entités avec l'endroit exact au millier de documents indiqués par les balises <*entity*> et ses attributs (*name, src*).

```
<entity-list>
  <entity name="speaker" src=".\CMUAN~3G.DER" start="142" end="151">
    <token start="142" len="3">Mr.</token>
    <token start="146" len="5">Okada</token>
  </entity>
  <entity name="speaker" src=".\CMUAN~EU.ACE" start="330" end="348">
    <token start="330" len="3">Dr.</token>
    <token start="334" len="9">Stephanie</token>
    <token start="344" len="4">Shaw</token>
  </entity>
  <entity name="speaker" src=".\CMUAN~G2.CAR" start="624" end="640">
    <token start="624" len="3">Mr.</token>
    <token start="628" len="6">Andrew</token>
    <token start="635" len="5">Gault</token>
  </entity>
  <entity name="speaker" src=".\CMUAN~G2.CAR" start="810" end="826">
    <token start="810" len="3">Mr.</token>
    <token start="814" len="6">Jessie</token>
    <token start="821" len="5">Ramey</token>
  </entity>
  <entity name="speaker" src=".\CMUAN~G2.CAR" start="880" end="896">
    <token start="880" len="3">Dr.</token>
    <token start="884" len="4">Judi</token>
    <token start="889" len="7">Mancuso</token>
  </entity>
</entity-list>
```

**Fig. 26.** Un extrait d'une sortie d'extraction obtenu du corpus d'annonces de conférences CMU : séquences de tokens extraits pour le slot <speaker>.

Le guide du TIES explique en plus de détail la configuration nécessaire pour exécuter des extractions.

## 4.3 M-TIES : nouvelle version de TIES étendue à l'annotation morphosyntaxique

Comme nous avons déjà évoqué dans l'introduction de ce chapitre, il a fallu d'étendre la version originale du système TIES afin de pouvoir évaluer l'influence de l'annotation morphosyntaxique (POS tagging) dans l'extraction d'information dans un document plus ou moins structuré. De plus, certaines limitations dans cette version du système ont été constatées, notamment un prétraitement insuffisant de documents d'entrée pour bien préparer et valider les documents avant qu'ils puissent être traités par le module d'apprentissage. Ces limitations seront corrigées dans la nouvelle version proposée par ce projet: M-TIES.

La figure suivante illustre l'architecture modifiée du système présentant les nouveau modules y rajoutés. Nous exposerons ci-après les tâches qui ont été nécessaires pour adapter le système TIES aux besoins de cette recherche.

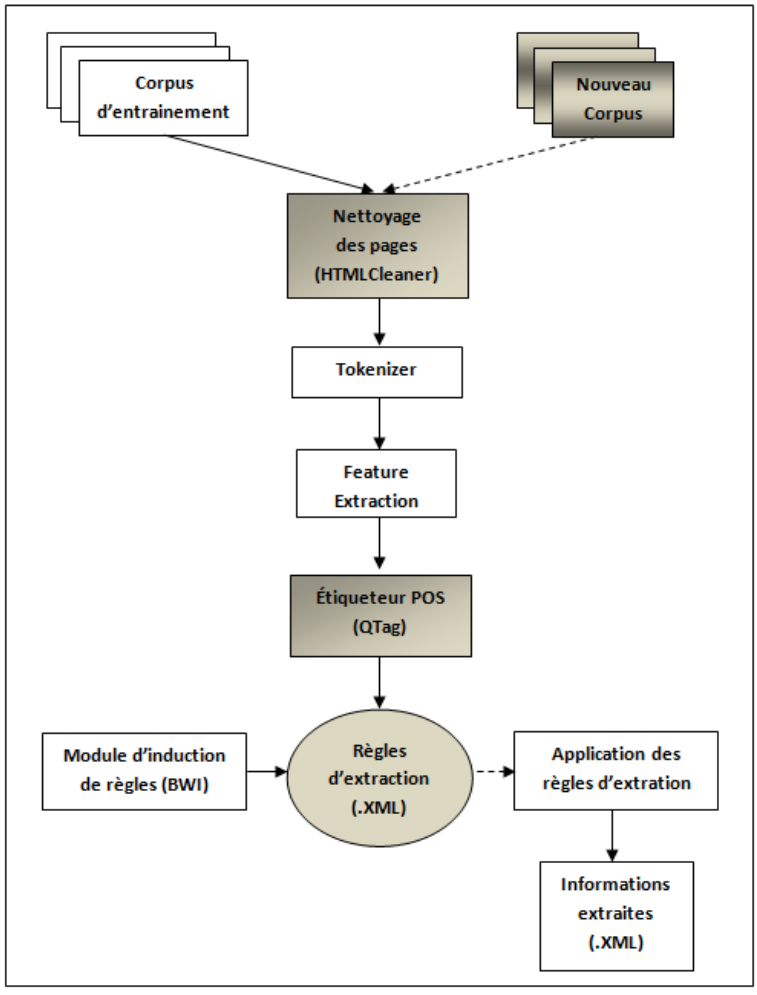



**Fig. 27**. Nouvelle architecture du TIES (M-TIES).

### 4.3.1 Amélioration du prétraitement

#### 4.3.1.1 Module d'annotation de documents : MnM

Bien que ce module ne soit pas vraiment intégré dans l'architecture de M-TIES et pour cela il ne soit pas illustré dans la Fig. 27, il a été retenu pour bien s'adapter aux besoins de notre recherche. MnM consiste d'un outil d'annotation automatique de documents qui intègre un navigateur Web et un éditeur d'ontologie [Vargas-Vera, 2002]. En plus, il fournit une API pour faire le lien avec des serveurs d'ontologies et pour l'intégration d'outils d'EI. Cet outil peut être considéré comme un exemple précoce de la prochaine génération des éditeurs d'ontologies orientés vers l'étiquetage sémantique et, en plus, qui fournit des mécanismes pour l'étiquetage automatique à grande échelle de pages Web.

La Fig. 28 montre la fenêtre principale de l'interface du MnN : à gauche, il présente l'ontologie de référence qui guide l'utilisateur dans le processus d'annotation. La fenêtre à droite affiche les pages Web du corpus à annoter avec les balises qui sont définies par les classes de l'ontologie de référence. Des guides de l'utilisateurs/développeur qui présentent l'outil MnM en détails peuvent être trouvé sur le site Internet dédie a son projet [MnN, 2008].

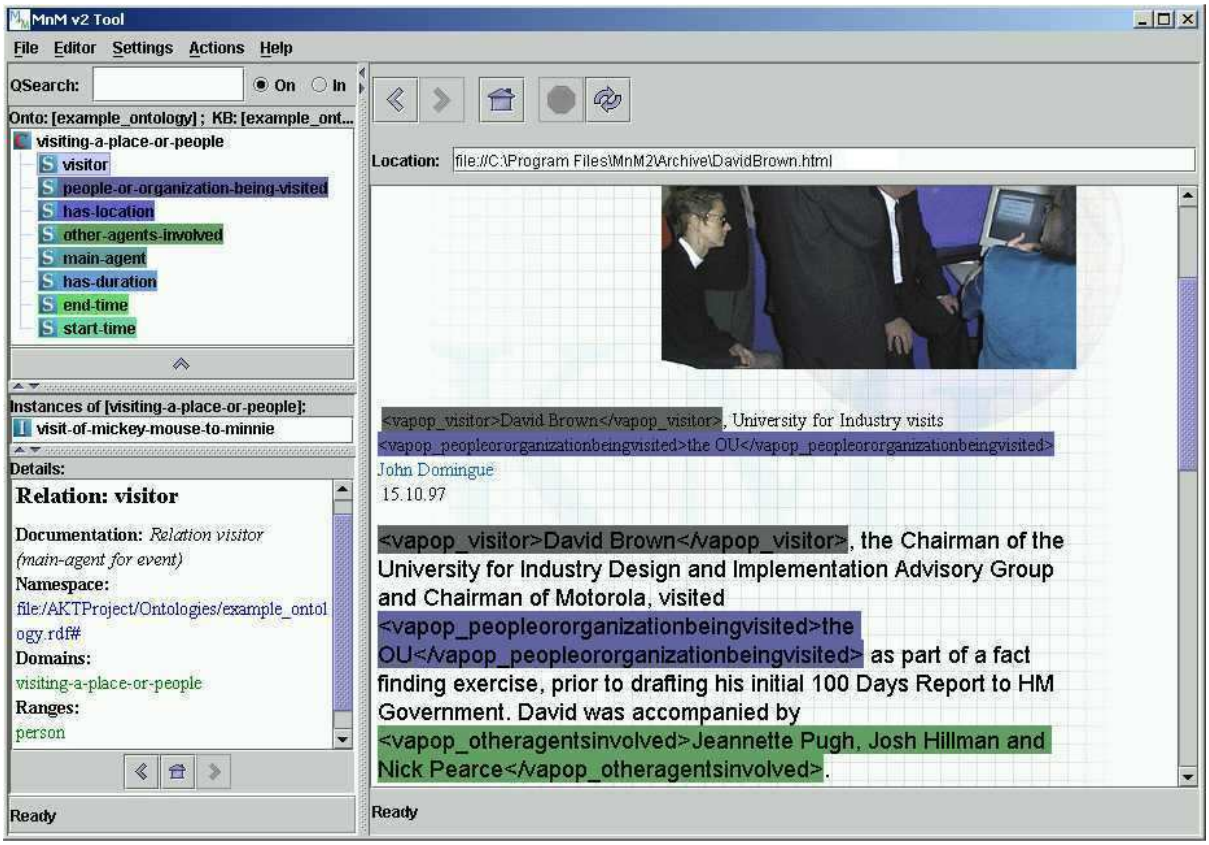



**Fig. 288.** Fenêtre principale de l'outil d'annotation MnM.

#### 4.3.1.2 Module de Nettoyage de pages HTML : HTMLCleaner

Étant donné que TIES exige des pages XHTM/XML bien formés et que l'on a constaté des problèmes avec les pages Web dans les *corpora* choisis pour les expérimentations, on s'est servir de l'outil de nettoyage et transformations de pages HTMLCleaner [Girardi, 2007] pour

les résoudre. La Fig. 27 illustre que cet outil est appliqué sur les documents d'entrée avant toute chose.

HtmlCleaner est un outil qui sert à nettoyer automatiquement des fichiers en HTML/XHTML. Il enlève des balises et des parties du texte non pertinent (comme certains mots utilisés comme menu de navigation, l'en-tête et le titre de bas de page commun dans toutes les pages d'un site, etc.). Il également restructure le texte avec un codage de base de la structure de la page à l'aide d'un ensemble minimum de symboles pour marquer le commencement d'en-têtes, de paragraphes et d'éléments de listes.

Cet outil peut être très utile pour construire un corpus à partir des pages Web. Il a été évalué dans la première compétition *CleanEval* en septembre 2007. CleanEval une compétition d'évaluation sur le thème de nettoyage de pages Web arbitraires, avec l'objectif de préparer des données Web comme un corpus, pour la recherche et le développement de la technologie linguistique.

Dans ce projet, *HTMLCleaner* a été utiliser comme outil de nettoyage de pages Web des *corpora* et pour vérifier et corrigés les balises HTML manquantes, en les transformant donc, en fichiers XHTML bien formés. En plus, il a fallu le développement d'un programme (en Java) pour adapter *HTMLCleaner* aux besoins de services de nettoyage de pages HTML et de mise en forme en suivant le standard XML.

### 4.3.2 Extension du module de tokenisation par l'ajout de tagage POS

Le traitement de texte libre par TIES s'est rendu possible en ajoutant un étiqueteur POS à son module de prétraitement (Fig. 27). L'outil QTAG [Mason & Tufis, 1998] a été adopté pour réaliser l'analyse morphosyntaxique automatique d'un corpus d'entrée après l'étape de tokenisation.

Par défaut, la tokenisation de TIES génère un fichier en format TIESIF sans prise en compte de l'annotation morphosyntaxique. L'idée ici est de prendre ce fichier et y insérer les étiquettes POS de chaque token proposés par QTAG, tout en respectant le format TIESIF. De ce fait, le module d'apprentissage de TIES peut enfin exploiter un corpus enrichi avec des informations morphosyntaxiques. Ci-après, nous décrivons l'étiqueteur QTAG.

#### 4.3.2.1 Module d'étiquetage morphosyntaxique (POS tagging): QTag

QTag [Mason & Tufis, 1998] est un étiqueteur POS (*tagger*) stochastique de parties du discours. Il crée le lexique, l'ensemble d'étiquettes (*tags*), les probabilités lexicales et contextuelles à partir du corpus manuellement étiqueté. Grâce à cette base d'apprentissage, l'étiqueteur POS peut trouver les étiquettes possibles avec leur fréquence pour les assigner à chaque unité lexicale dans un nouveau corpus déjà segmenté. Enfin, l'étiqueteur POS effectue la tâche de désambiguïsation en utilisant les distributions probabilistes apprises à partir du corpus. Cela signifie qu'il lit le texte et pour chaque *token* y présent, il renvoie à partie du discours qu'il appartient (par exemple : nom, verbe, ponctuation, etc.). Il fonctionne en utilisant des méthodes statistiques, d'où le «stochastique» vient. Par conséquence, il peut commettre des erreurs (comme tous les étiqueteurs POS), mais il est assez robuste en étiquetant des textes avec une précision élevée.

Il y a deux façons d'utiliser le QTag : soit en tant que logiciel, lorsqu'on prépare les textes et puis on exécute le QTAG là-dessus, ou intégré en tant que module à travers une API dans nos propres programme.

QTAG est implémenté en Java, ce qui signifie qu'il fonctionne sur plusieurs systèmes d'exploitions. Il est, en principe, indépendant de la langue, bien que la version actuelle ne vient qu'avec des fichiers de ressources pour l'anglais. Si on a besoin de l'utiliser avec d'autres langues, on devra avoir un ensemble pré-étiqueté de texte pour créer les ressources nécessaires. Le logiciel pour créer ces ressources est aussi inclus dans le *package* de distribution sur le site [QTag, 2008].

Dans cette recherche, QTAG est utilisé comme module d'étiquetage morphosyntaxique a fin d'étiqueter automatiquement des documents avec un ensemble d'étiquettes POS qui est listé (avec leurs significations) dans l'appendice A.

### 4.3.3 Génération de diagrammes de classes

Le guide utilisateur du system TIES est considérablement résumé. De plus, sa documentation concernent l'API (*Application Program Interface*) est très légère. Aussi il a été nécessaire d'utiliser un outil de retro-ingénierie (reverse engineering) pour obtenir, à partir du code source de TIES, les diagrammes de classes de tous les *package* du système.

### 4.3.4 Décompilation de code source

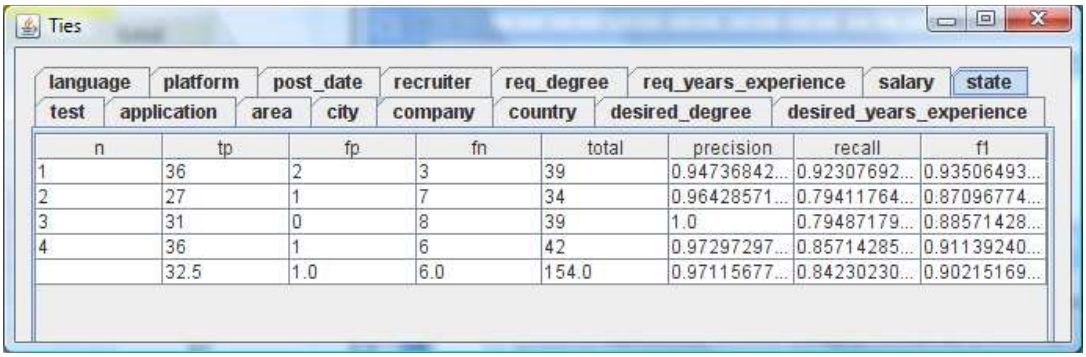
Le système TIES a été écrit en langage Java. Le code source de l'un de ces packages, celui responsable pour le prétraitement (tokenisation et feature extraction) du corpus d'entrée, n'était pas disponible directement. Ce package se présentait comme un fichier JAR. Un fichier JAR (*Java ARchive*) est un ensemble de code source java compilés en *bytecode*[8] et groupés dans un seul fichier pour distribution. Il a fallu alors, grâce à des outils de décompilations de bytecode java existants sur le Web, décompiler ce package afin d'accéder au code source et faire les modifications pertinentes aux besoins de la nouvelle version du système TIES. Désormais le code source révisé de ce module est disponible pour qu'on puisse le changer en futures versions du système.

### 4.3.5 Sortie de résultats en format CSV

Les résultats de l'apprentissage exprimés par les mesures de Précision, Rappel et F-Measure sont affichés par TIES à la fin d'une session d'apprentissage comme illustré par la Fig. 29. Afin de contrôler la mise en page de la sortie du système et faciliter l'analyse ultérieure des résultats, on a développé de nouvelles classes Java qui produisent des fichiers en format CVS[9] permettant l'usage de tableurs (spreadsheet) pour les exploiter.

[8] Bytecode est créé lors de la compilation de code source java par le biais de compilateur *javac* de l'environnement JDK. Le compilateur javac produit alors un fichier *.class.*

[9] Ce type de fichier est compatible avec Excel.




| n | tp | fp | fn | total | precision | recall | f1 |
|---|---|---|---|---|---|---|---|
| 1 | 36 | 2 | 3 | 39 | 0.94736842... | 0.92307692... | 0.93506493... |
| 2 | 27 | 1 | 7 | 34 | 0.96428571... | 0.79411764... | 0.87096774... |
| 3 | 31 | 0 | 8 | 39 | 1.0 | 0.79487179... | 0.88571428... |
| 4 | 36 | 1 | 6 | 42 | 0.97297297... | 0.85714285... | 0.91139240... |
| | 32.5 | 1.0 | 6.0 | 154.0 | 0.97115677... | 0.84230230... | 0.90215169... |

**Fig. 29.** Fenêtre de résultats d'une session d'apprentissage de TIES.

## 4.4 Conclusion

Ce chapitre a présenté un système d'EI modulaire que, reposant sur l'induction supervisée d'extracteurs, permet l'extraction d'information à partir d'un corpus d'apprentissage manuellement annoté par l'utilisateur et qui peut prendre en compte la syntaxe du langage naturel par le moyen d'un nouveau module responsable pour l'étiquetage morphosyntaxique sur ces documents.

Dans le prochain chapitre, afin d'évaluer les performances de l'architecture proposée, plusieurs expériences seront réalisées sur des *corpora* bien connus dans la communauté scientifique du domaine de l'EI : Seminars, Jobs et CFP (Pascal Challenge). L'objectif sera d'évaluer les performances d'un système d'induction supervisée d'extracteurs (M-TIES) sur des *corpora* de différents niveaux de structuration, avec ou non prise en compte de la syntaxe du langage naturel, c'est-à-dire, avec ou non étiquetage morphosyntaxique.

# 5 EXPERIMENTATIONS

Ce chapitre réalise une séries d'expériences avec l'objectif d'évaluer la version modifiée du système TIES, dorénavant *M-TIES*, sur 3 *corpora* constitués de documents à différents niveaux de structuration. Après la description de ces *corpora*, le protocole expérimental défini par une méthodologie d'évaluation et quelques recommandations afin d'avoir de résultats plus fiables sont présentées. Tout d'abord des expériences pour déterminer les meilleurs paramètres de l'algorithme BWI lors de la génération de modèles sont réalisées. Ensuite, on évalue les modèles appris sur ces *corpora* afin de vérifier le gain que l'information POS peut apporter à la performance du système. Ce chapitre se termine en présentant une évaluation comparative de système proposé dans cette recherche avec d'autres systèmes existants.

## 5.1 Corpora choisis

Cette section présente les *corpora* choisis pour évaluer le système d'EI proposé sans prise en compte d'analyse morphosyntaxique ou avec une telle prise en compte. Pour cela, trois *corpora* de niveaux de structuration: *Seminars*, *Jobs* et *Call For Papers* ont été retenu. Dans un premier temps, le processus d'annotation de documents de ces *corpora* est décri ; puis, ces *corpora* sont examinés en détail, notamment leurs descriptions, leurs *templates (*schéma) d'extraction et quelques statistiques pertinentes.

### 5.1.1 Annotation de documents

A fin d'utiliser des algorithmes d'EI supervisés du M-TIES sur une collection de documents, il faut disposer d'un ensemble d'apprentissage d'exemples positifs annotés. Il faut réaliser cette annotation et faire en sorte que celle-ci soit compatible avec le format TIESIF, le seul format d'entrée accepté par M-TIES. Par conséquent, les exemples positifs qui serviront pour l'apprentissage du système seront annotés de la façon suivante (Fig. 30) :

<*doc* id='276' filename='cmu.cs.proj.vision-273_0'><0.25.4.84.12.33.15.???@???.0>
Type: cmu.cs.proj.vision
Topic: Sanderson group seminar
Dates: 27-Apr-84
Time: **<stime>**2:30**</stime>**
PostedBy: ??? on 25-Apr-84 at 12:33 from ???
Abstract:
**<speaker>**Alberto Elfes**</speaker>** will be speaking about "A Wide-Beam Sonar Mapping System" on Friday the 27th in **<location>**WeH 4623**</location>** at **<stime>**2:30**</stime>.**
*</doc>*

**Fig. 30.** Exemple d'un document correctement annoté en XML.

La Fig. 30 illustre comment il faut annoter les *exemples positifs*. Ainsi, il faut signaler chaque champs (*slot*) d'information, en les entourant avec des balises *ouvrantes (<stime>)* et celles *fermantes* correspondantes *</stime>*, en respectant la syntaxe XML [Bray et al., 2008]. Les balises *<doc>* et *</doc>*, indiquant le début et respectivement la fin du document, sont optionnelles.

Cette tâche préliminaire d'annotation de documents peut être faite manuellement : il ne faut qu'annoter chaque *slot* d'information trouvé dans les documents du corpus. Cependant, pour cette recherche, il est envisageable d'automatiser cette tâche d'annotation car à chaque fois que l'on change de corpus ou de domaine, il faudra annoter tous les documents du nouveau corpus. En tenant en compte cette possibilité, le système d'annotation *MnN*, déjà présenté dans la section 4.3.1, sera en charge de rendre cette tâche moins fastidieuse.

## 5.1.2 Corpus SEMINARS

### 5.1.2.1 Description du corpus

Le corpus *Seminars* est constitué par une collection d'annonces de conférences prise des panneaux d'affichage électronique de l'Université Carnegie Mellon, États-Unis, pendant la période de septembre/1982 à août/1995 et proposée par [Freitag, 1997]. Cette collection comporte 485 documents aux mises en forme très différentes. La figure suivante en illustre un exemple :

```
Name: Dr. Jeffrey D. Hermes
Affiliation: Department of Autoimmune Diseases
Research & Biophysical Chemistry Merck Research
Laboratories
Title: "MHC Class II: A Target for Specific
Immunomodulation of the Immune Response"
Host/e-mail: Robert Murphy, murphy@a.cfr.cmu.edu
Date: Wednesday, May 3, 1995
Time: 3:30 p.m.
Place: Mellon Institute Conference Room
Sponsor:  MERCK RESEARCH LABORATORIES
```

---

```
Professor John Skvoretz, U. of South Carolina,
Columbia, will present a seminar entitled
"Embedded Commitment," on Thursday, May 4th
from 4-5:30 in PH 223D.
```

---

```
Laura Petitto
Department of Psychology
McGill University

Thursday, May 4, 1995
12:00 pm
Baker Hall 355
```

**Fig. 31.** Exemple d'un document du corpus Seminars [Freitag, 1997].

#### 5.1.2.2 Définition du Template d'extraction

Pour chaque document de ce corpus, il faut identifier et extraire les informations suivantes :

- *location* (endroit de la conférence);
- *speaker* (nom du conférencier) ;
- *stime* (heure de début) ;
- *etime* (heure de la fin de la conférence).

La table suivante montre des statistiques sur le corpus *Seminars*.

**Tab. 3.** Nombre d'exemples positifs pour les slots (location, speaker, stime, etime) *et* d'exemples négatifs non annotés (*non-entity*) du corpus *Seminars*.

| ***Seminars*** | *Location* | *Speaker* | *Stime* | *Etime* | *Non-Entity* |
|---|---|---|---|---|---|
| | 643 | 754 | 980 | 433 | 157.647 |

#### 5.1.2.3 Exemples de sorties d'extractions

La Fig. 32 illustre deux sorties d'une même tâche d'extraction sur un document du corpus Seminars. Nous pouvons noter que certaines instances du template peuvent être complètement remplies et d'autres, sont partiellement remplies.

| ***speaker*** | *Prof. John Skvoretz* |
|---|---|
| **location** | PH 223D |
| **stime** | 4 :00 ou 4 |
| **etime** | 5 :30 |

(a)

| ***speaker*** | *Mike Greenberg* |
|---|---|
| **location** | - |
| **stime** | 3 :30 PM |
| **etime** | - |

(b)

**Fig. 32.** Exemples de template d'extraction complète rempli (a) et partiellement rempli (b) à la fin d'une étape d'extraction.

### 5.1.3 Corpus JOBS

#### 5.1.3.1 Description du corpus

Le corpus *JOBS* concerne des offres d'emploi dans le domaine de l'Informatique. Ce corpus est composé de 300 documents qui contiennent des informations sur des employeurs, entreprises, salaires et exigences d'offres d'emploi. Plusieurs slots, tel que ceux concernant les langages et les plate-formes logiciels requis pour l'emploi, peuvent avoir plusieurs valeurs.

Dans les expérimentations, il a été considéré le corpus proposé originalement par [Califf & Mooney, 1999]. La Fig. 33 présente un document originaire de ce corpus et son template d'extraction rempli.

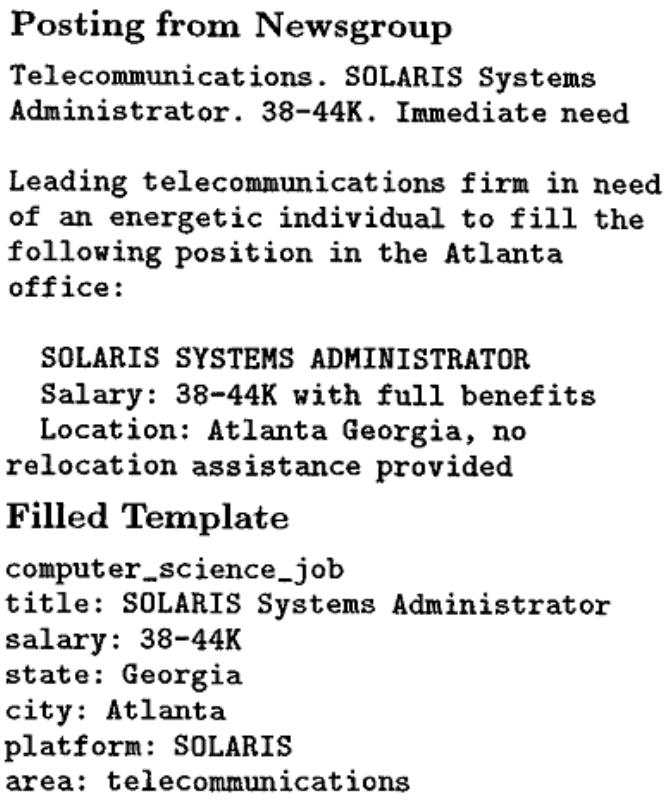



**Fig. 33.** Exemple d'une offre d'emploi avec son template d'extraction rempli [Califf, 1999].

### 5.1.3.2 Définition du Template d'extraction

Chaque document de ce corpus comporte 17 *slots* suivants à extraire : *id, title, salary, company, recruiter, state, city, country, language, platform, application, area, req_years_experience, desired_years_experience, req_degree, desired_degree et post_date.* N'importe quel de ces slots peut apparaître aucune, une ou plusieurs fois dans un même document. Le tableau ci-dessous en présente des statistiques sur ce corpus :

**Tab. 4.** Nombre d'exemples pour chaque entité (*slot*) du corpus *JOBS*.

| | | | | | | |
|---|---|---|---|---|---|---|
| **JOBS** | *Platform* | *Language* | *Area* | *City* | *State* | *Application* |
| | 709 | 851 | 1005 | 659 | 452 | 590 |
| | *Title* | *Recruiter* | *Post date* | *Country* | *Salary* | *Req-years-e* |
| | 457 | 312 | 302 | 345 | *141* | *166* |
| | *Company* | *Des-years_e* | *Req-degree* | *Des-degree* | *Id* | |
| | *298* | *43* | *83* | *21* | 304 | |

### 5.1.4 Corpus CFP (CALL FOR PAPERS) - Pascal Challenge 2005

#### 5.1.4.1 Description du corpus

Ce corpus est constitué de 1.100 documents contenant 850 *Workshop Call for Papers* (CFP) et 250 Conference CFP, établi lors de la compétition de systèmes d'EI de l'état de l'art, *Pascal Challenge*, en 2005 **[**Ireson & Ciravegna, 2005]. La grosse majorité de documents relèvent du domaine de l'Informatique, bien que d'autres domaines (biomédecine et linguistique) y soient représentés.

Les documents sont divisés en trois corpora : un *corpus d'apprentissage* (400 Workshop CFP), un *corpus de test* (200 Workshop CFP), et un *corpus enrichi* (250 Workshop CFP and 250 Conférence CFP). Ce dernier corpus a été annoté par le system GATE qui fournit la tokenisation, étiquetage POS, NER et attributs de token de textes (type de token, taille, etc).

#### 5.1.4.2 Définition du template d'extraction

Chaque document peut avoir 11 slots à extraire : 8 concernant des Workshops (*name, acronym, homepage, location, date, paper submission date, notification date* et *camera-ready copy date*), et 3 relatifs aux Conferences (*name, acronym* et *homepage*).

**Tab. 5.** Distribution de fréquences des slot annotés [Ireson & Ciravegna, 2005].

| ANNOTATION TYPE | CORPUS FREQUENCY | | | |
|---|---|---|---|---|
| | TRAIN | % | TEST | % |
| workname | 543 | 11.8 | 245 | 10.8 |
| workacro | 566 | 12.3 | 243 | 10.7 |
| workhome | 367 | 8.0 | 215 | 9.5 |
| workloca | 457 | 10.0 | 224 | 9.9 |
| workdate | 586 | 12.8 | 326 | 14.3 |
| workpape | 590 | 12.9 | 316 | 13.9 |
| worknoti | 391 | 8.5 | 190 | 8.4 |
| workcame | 355 | 7.7 | 163 | 7.2 |
| confname | 204 | 4.5 | 90 | 4.0 |
| confacro | 420 | 9.2 | 187 | 8.2 |
| confhome | 104 | 2.3 | 75 | 3.3 |
| **TOTAL** | **4583** | **100** | **2274** | **100** |

La Tab. 5 présente la distribution de fréquences des slots annotés dans le deux premiers corpus (apprentissage et test) ; comme on peut bien constater, les deux distributions sont à peu près semblables. Il faut noter que, comme ni tous les workshops ont une conférence qui leurs sont associées, les slots moins représentatifs sont les slots relatifs aux conférences.

### 5.1.5 Comparaison et spécificités des Corpora

Un système d'EI adaptatif doit s'adapter à un nouveau domaine ou application avec un minimum d'effort. Du point de vue de l'algorithme d'apprentissage, un tel système est nécessaire pour apprendre un premier modèle à partir d'un petit nombre d'exemples d'apprentissage.

Dans ce contexte, afin d'évaluer le système M-TIES dans un scénario d'IE adaptatif, on évalue son algorithme d'apprentissage (qui sera détaillé dans le chapitre suivant) sur 3 *corpora* déjà annotés, avec différents degrés de structuration pouvant être classés en 2 groupes : *partiellement structuré et naturel* (non structuré).

Les documents partiellement structurés des *corpora Seminars* et *Jobs* contiennent des structures au niveau de document et des régularités dans la mise en forme et dans les annotations. Par exemple, il est commun pour quelques slots d'être précédés d'une étiquette d'identification (ex. « Speaker : Dr. X »), bien que ce ne soit pas toujours le cas.

Les documents en langage naturel du corpus CPF contiennent peu de structuration de mise en forme. Les principales raisons du choix de ce corpus sont dû à la connaissance du domaine et parce qu'il offre une gamme raisonnable de difficulté. Une autre caractéristique souhaitable de ce corpus, c'est que leurs documents ont un certain degré de mise en forme semi-structuré. Par conséquent, les algorithmes d'apprentissage devraient exploiter telle régularité. En outre, il existe certaines différences entre les types d'annotation du corpus. Les *dates importantes* du CPF (*paper submission date, notification* date *et camera-ready copy date*) sont généralement bien prescrites par les textes qui les entourent, tandis que les noms de *workshop* et *conference* sont plus définis par leurs positions dans le document et ils ont une longueur plus variable. De telles différences influenceront évidemment la capacité des algorithmes d'apprentissage à identifier les types d'annotation.

## 5.2 Protocole Expérimental

Tout d'abord les travaux de préparations des *corpora* qui ont été nécessaires pour mener les expériences sont présentés. Les résultats accompagnés de ses respectives discussions sont finalement exposés après la présentation de la méthodologie d'évaluation adoptée.

### 5.2.1 Préparations des corpora

Dans cette recherche nous avons utilisé HTMLCleaner pour vérifier et corrigés les balises HTML manquantes de pages Web, en les transformant en fichiers XHTML bien formés. En plus, il a fallu le développement d'un programme (en Java) pour adapter HTMLCleaner aux besoins de services de nettoyage de pages HTML et de mise en forme sous le format XML. Les *corpora* ont été annotés selon l'exemple déjà présenté dans la section 5.1.

### 5.2.2 Méthodes d'évaluation

Utiliser les données d'apprentissage pour concevoir un classificateur et puis estimer la précision de la classification sur ces mêmes données peut aboutir à des estimations trompeusement suroptimistes à cause de la surspécialisation de l'algorithme d'apprentissage.

*Hold-out* et *k-fold cross-validation* sont deux techniques d'évaluation d'exactitude de classificateurs, reposant sur des partitions aléatoires d'échantillonnage de données. Dans la méthode *hold-out*, les données sont aléatoirement partitionnées en deux ensembles, un ensemble *d'apprentissage* et un autre de *test*. Typiquement, deux-tiers des données sont attribuées à l'ensemble d'apprentissage, et le tiers restant est attribué à l'ensemble de test. L'ensemble d'apprentissage est employé pour construire le classificateur, dont l'exactitude est estimée avec l'ensemble de test. L'évaluation ici est pessimiste puisque seulement une partie de données initiales est employée pour construire le classificateur. *Random subsampling* est une variante de la méthode holdout dans laquelle la méthode holdout est répétée *k* fois. L'évaluation globale d'exactitude est prise comme la moyenne des précisions obtenues à partir de chaque itération.

Dans *k-fold cross-validation*, les données initiales sont aléatoirement divisées en *k* sous-ensembles mutuellement exclusifs ou *fold*, $S_1$, $S_2$, ..., $S_k$, chacun d'une taille approximativement égale. L'apprentissage et test sont effectués *k* fois. Dans l'itération *i*, le sous-ensemble $S_i$ est réservé comme l'ensemble de test, et les sous-ensembles restants sont collectivement employés pour faire apprendre le classificateur. On définit alors l'estimation d'exactitude comme le nombre global de classifications correctes de *k* itérations, divisé par le nombre total d'instances présentes dans les données initiales. Kohavi [Kohavi, 1995] défend ce dernier méthode d'évaluation comme la meilleur parce qu'elle fournit des estimations plus impartiales et avec une variance minimum.

Les expériences de cette section ont été menées en prenant en compte les méthodes hold-out ou k-fold-cross-validation.

#### Macroaverage et Microaverage

Pour chaque slot d'information, des résultats sont évalués en comptant *true positives tp* (slots correctes), *false positives fp* (slots incorrectes), *false negatives fn* (slots absents) et en calculant la *précision* $P = \frac{tp}{tp + fp}$ et le *rappel* $R = \frac{tp}{tp + fn}$.

Pour un corpus contenant plusieurs slots d'information à extraire, il y a diverses manières de combiner les résultats de toutes les slots en une seule mesure, parmi elles, on peut mentionner les mesures *micro-average* et *macro-average*. La mesure micro-average est calculée en additionnant les *tp, fp et fn* pour tous les slots, et en calculant ensuite *P, R et F*. Par conséquent, des slots plus fréquents ont un impact plus élevé sur la mesure finale que des slots rares. D'autre part, la mesure macro-average est calculée en faisant la moyenne de toutes les valeurs *P* et *R* par slots, de sorte que toutes les slots sont considérés comme d'égale importance, peu important combien de fois ils se produisent.

Certains chercheurs soutiennent que la mesure macro-average est meilleure que la micro-average [Yang & Liu, 1999] parce que la dernière peut être dominée par les classes plus nombreuses de telle façon qu'elle exprime moins la performance de l'algorithme sur les classes moins représentatives. D'autre part, si toutes les classes sont d'une taille comparable, comme c'est souvent le cas de corpora en EI, alors la mesure macro-average n'est pas très différente de la micro-average.

Dans ce chapitre, la performance globale du système proposé sur un corpus quelconque est mesurée en termes de *F-Measure (micro et macro)* avec le paramètre $\beta$ = 1.

## 5.3 Expériences

Les résultats expérimentaux sur trois *corpora* décrits dans la section 5 sont présentés. Toutes les expériences ont été menées avec le système M-TIES. Pour chaque expérience, les respectifs paramètres utilisés pendant l'apprentissage du modèle sont établis. En plus, il a été décidé de reprendre les mêmes protocoles expérimentaux utilisés par d'autres travaux de recherche en l'EI afin d'avoir des comparaisons plus pertinentes.

### 5.3.1 Influence des Paramètres de l'algorithme BWI et information POS

#### 5.3.1.1 Nombre d'itérations de boosting : Corpus JOBS

L'objectif visé des expériences de cette section, dont les résultats sont présentés dans la Tab. 6 et la Fig. 34, est de mesurer la sensibilité de l'algorithme au nombre d'itérations de *boosting*. Le paramètre look-ahead $L$ = 3 a été fixé tandis que nous faisons varier le nombre de boosting de 10 jusqu`à 200.

**Tab. 6.** Influence du nombre d'itérations de *boosting* sur le corpus *Jobs*.

| ***Nombre d'Itérations*** | ***10*** | ***20*** | ***30*** | ***40*** | ***70*** | ***100*** | ***200*** |
|---|---|---|---|---|---|---|---|
| *application* | 0,544 | 0,593 | 0,656 | 0,666 | 0,690 | 0,685 | 0,695 |
| *area* | 0,383 | 0,455 | 0,492 | 0,534 | 0,581 | 0,618 | 0,631 |
| *city* | 0,965 | 0,964 | 0,967 | 0,964 | 0,965 | 0,968 | 0,967 |
| *country* | 0,987 | 0,991 | 0,986 | 0,988 | 0,988 | 0,984 | 0,982 |
| *language* | 0,773 | 0,821 | 0,847 | 0,859 | 0,881 | 0,883 | 0,893 |
| *plataform* | 0,737 | 0,795 | 0,823 | 0,848 | 0,864 | 0,869 | 0,871 |
| *post_date* | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| *recruiter* | 0,794 | 0,855 | 0,872 | 0,869 | 0,885 | 0,868 | 0,879 |
| *state* | 0,967 | 0,974 | 0,964 | 0,977 | 0,973 | 0,977 | 0,971 |
| *title* | 0,479 | 0,604 | 0,630 | 0,672 | 0,706 | 0,690 | 0,703 |
| ***F1 macro*** | 0,763 | 0,806 | 0,824 | 0,838 | 0,853 | 0,854 | 0,859 |

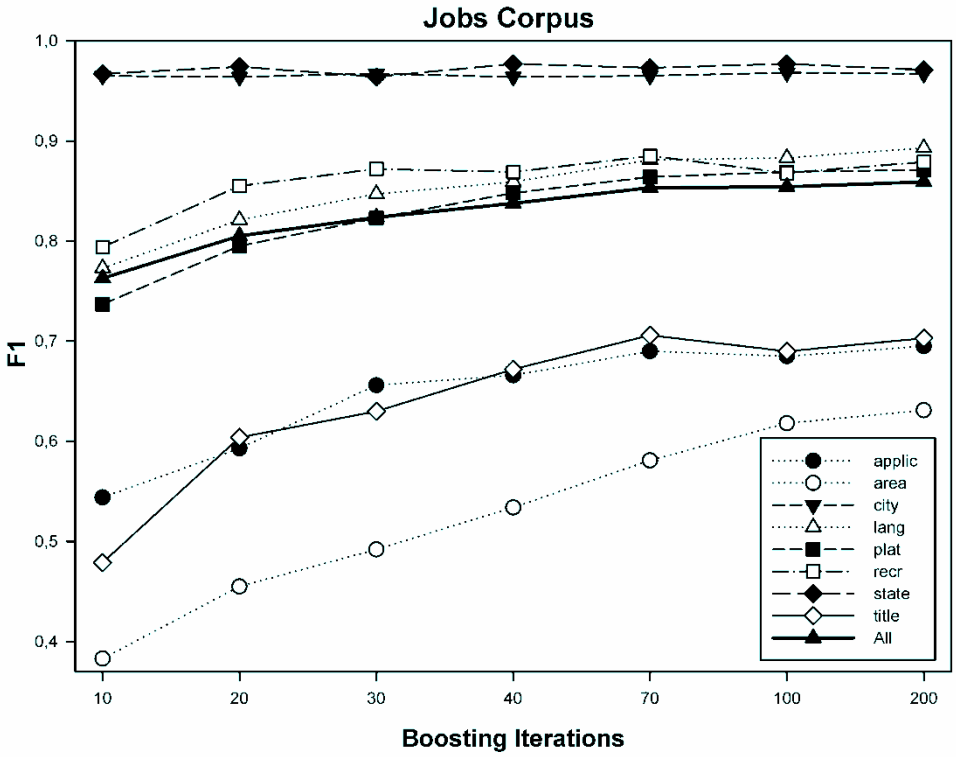



**Fig. 34.** Performance F1 comme fonction du nombre d'itérations de *boosting* sur le corpus Jobs.

Comme la Fig. 34 suggère, le nombre d'itérations exigés par l'algorithme BWI pour atteindre la performance maximal dépend de la difficulté de la tâche. Par exemple, pour les slots *city* et *state*, très peu d'itérations ont été nécessaires pour achever la performance maximale, dont nous pouvons conclure que pour les slot plus réguliers, très peu d'itérations sont suffisantes. D'autre part, quand on considère les slots *recruiter*, *language* et *plataform*, on s'aperçoit qu'il faut un nombre plus grand d'itérations pour atteindre un niveau similaire de performance. Les slots *area*, *title* et *application* se montrent encore plus difficiles pour l'algorithme BWI car on observe la tendance croissante de gain au fur et à mesure que l'algorithme utilise plus d'itérations.

À ce point il est important de discuter que le gain faible de performance du slot *area*, quand on considère le nombre d'itérations entre 100 et 200, ne compense pas le double de temps qu'il le faut. En plus, Kauchak [Kauchak et al., 2002] affirment que un nombre très élevé d'itérations peut amener l'algorithme à une situation de sur-apprentissage ou à des règles redondantes.

Enfin, la courbe correspondante du slot « All », vue ici comme la courbe moyenne de performance, montrent qu'un nombre de 100 itérations nous semble un bon compromis entre la performance et le temps d'exécution de l'algorithme BWI.

### 5.3.1.2 Paramètre Lookahead *L*

Il a été réalisé des expériences en utilisant différent valeurs pour le paramètre *look-ahead L* afin d'analyser son effet sur la performance. La section 3.2.4.5, sur les limitations de l'algorithme BWI, a expliqué que trouver une extension de séparateurs est exponentiel en *L* parce que chaque combinaison des motifs de tokens à droite et a gauche d'un séparateur quelconque est énumérée et évaluée. En revanche, dans [Freitag & Kushmerik] on trouve la

remarque que dans la grosse majorité de cas des tâches d'extraction, la valeur $L$ = 3 est pertinente pour avoir un bon équilibre entre la performance et le temps pris par l'algorithme pour induire d'extracteurs. Conséquemment, pour rendre les expériences de cette section viable pour des valeurs plus élevées de $L$, nous les avons réalisées avec les attributs défauts du système, c'est-à-dire, sans utiliser des attributs supplémentaires pour caractériser les tokens.

La Fig. 35 montre les résultats des expériences sur le corpus CFP pour une validation croisée (avec $k$ = 4) et le nombre d'itérations = 100. En examinant ces résultats, ils confirment l'importance du *contexte* en EI. En plus, le graphique ci-dessous montre que, dans un même corpus, différents slots peuvent avoir différentes valeurs optimales de $L$. Par exemple, le slot *workshop acronym* nécessite d'une *fenêtre de token* plus large de 7 pour avoir la meilleure performance. Après cette valeur, la performance se stabilise. Les slots *conference, homepage et conference acronym* atteignent la F1 measure optimale avec $L$ = 5 tandis que la valeur optimale pour le slot *workshop camera-ready-copy date* a été $L$ = 4. Finalement, il a été constaté que, pour la presque moitié de slots de ce corpus, notamment les slots *workshop date, workshop location, workshop home, conference name, workshop date*, $L$ = 3 donne la F1 measure optimale ou bien très proche d'elle.

Le chois de corpus CPF pour la réalisation de ces expériences a eu l'objectif d'évaluer l'influence du paramètre $L$ sur un corpus moins structuré. Pour les autres *corpora*, Seminars et Jobs, étant donné qu'ils sont plus structurés, les gains sont marginaux pour une fenêtre plus large que 3 comme a été démontré par [Li, 2004] et [Freitag & Kushmerick, 2000].

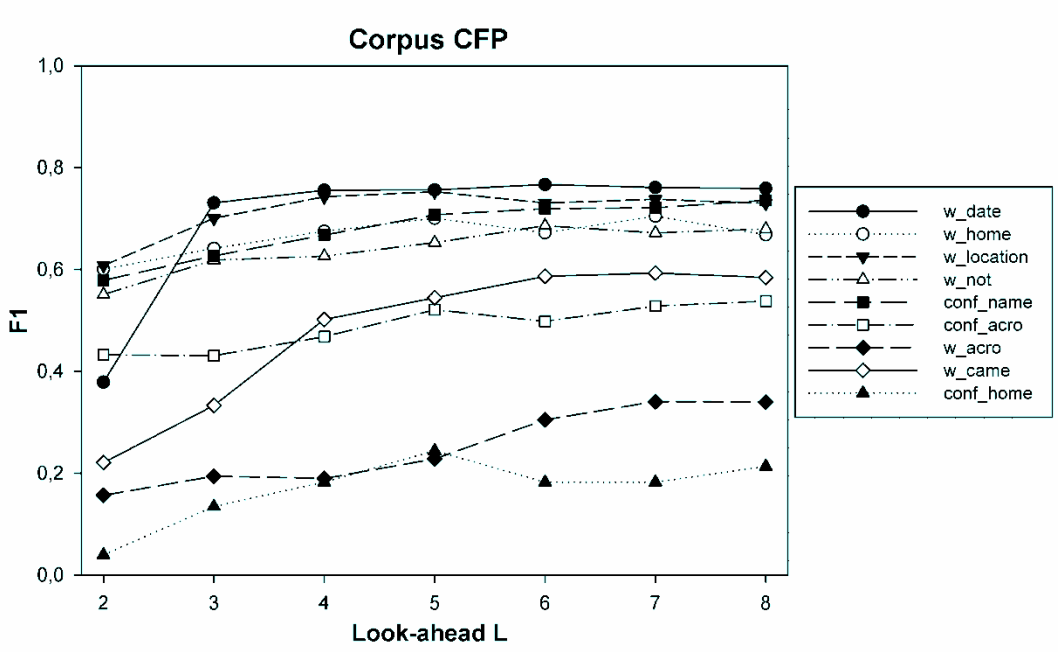



**Fig. 35.** Évolution de la F-measure en fonction du look-ahead L sur le corpus CFP.

### 5.3.1.3 Information POS

Les expériences menées dans cette section examinent l'influence du tagage POS sur chaque slot de tous les *corpora*. On a fixé le nombre d'itérations = 100 et look-ahead $L$ = 3 en suivant les suggestions de résultats antérieurs de ce deux paramètres qu'on vient d'analyser. Il a été utilisé la validation croisée avec $k$ = 10 dans les expériences menés sur les *corpora* Seminars et Jobs, et avec k = 4 sur le corpus CFP.

**Corpus SEMINARS : Résultats par slot avec et sans POS**

| *Slot* | *Préc* | *Rappel* | *F1* |
|---|---|---|---|
| *stime* | 0,985 | 0,979 | 0,982 |
| *etime* | 0,989 | 0,969 | 0,979 |
| *location* | 0,961 | 0,912 | 0,936 |
| *speaker* | 0,962 | 0,944 | 0,953 |

(a) **sans** POS

| *Slot* | *Préc* | *Rappel* | *F1* |
|---|---|---|---|
| *stime* | 0,984 | 0,983 | 0,983 |
| *etime* | 0,988 | 0,974 | 0,981 |
| *location* | 0,953 | 0,924 | 0,938 |
| *speaker* | 0,960 | 0,965 | 0,962 |

(b) **avec** POS

**Fig. 36.** Résultats sur le Corpus *Seminars* sans (a) et avec (b) POS.

**Corpus JOBS : Résultats par slot avec et sans POS**

| *Slots* | *Préc* | *Rappel* | *F1* |
|---|---|---|---|
| *application* | 0,903 | 0,618 | 0,734 |
| *area* | 0,848 | 0,491 | 0,622 |
| *city* | 0,993 | 0,944 | 0,968 |
| *company* | 0,943 | 0,759 | 0,841 |
| *country* | 0,998 | 0,974 | 0,986 |
| *des_degree* | 0,922 | 0,379 | 0,537 |
| *des_y_exp* | 0,942 | 0,835 | 0,885 |
| *id* | 1,000 | 0,956 | 0,977 |
| *language* | 0,934 | 0,840 | 0,885 |
| *plataform* | 0,957 | 0,802 | 0,872 |
| *post_date* | 1,000 | 1,000 | 1,000 |
| *recruiter* | 0,976 | 0,774 | 0,864 |
| *req_degree* | 0,915 | 0,805 | 0,857 |
| *req_y_exp* | 0,932 | 0,806 | 0,864 |
| *salary* | 0,920 | 0,854 | 0,886 |
| *state* | 0,995 | 0,963 | 0,979 |
| *title* | 0,855 | 0,582 | 0,693 |

(a)

| *Slots* | *Préc* | *Rappel* | *F1* |
|---|---|---|---|
| *application* | 0,905 | 0,599 | 0,720 |
| *area* | 0,831 | 0,502 | 0,626 |
| *city* | 0,988 | 0,941 | 0,964 |
| *company* | 0,945 | 0,772 | 0,850 |
| *country* | 1,000 | 0,966 | 0,983 |
| *des_degree* | 0,910 | 0,432 | 0,586 |
| *des_y_exp* | 0,893 | 0,856 | 0,874 |
| *id* | 1,000 | 0,952 | 0,975 |
| *language* | 0,931 | 0,841 | 0,883 |
| *plataform* | 0,952 | 0,807 | 0,873 |
| *post_date* | 1,000 | 1,000 | 1,000 |
| *recruiter* | 0,984 | 0,797 | 0,881 |
| *req_degree* | 0,903 | 0,767 | 0,830 |
| *req_y_exp* | 0,930 | 0,842 | 0,884 |
| *salary* | 0,895 | 0,870 | 0,882 |
| *state* | 0,991 | 0,960 | 0,975 |
| *title* | 0,833 | 0,601 | 0,698 |

(b)

**Fig. 37.** Résultats sur le Corpus *Jobs* sans (a) et avec (b) POS.

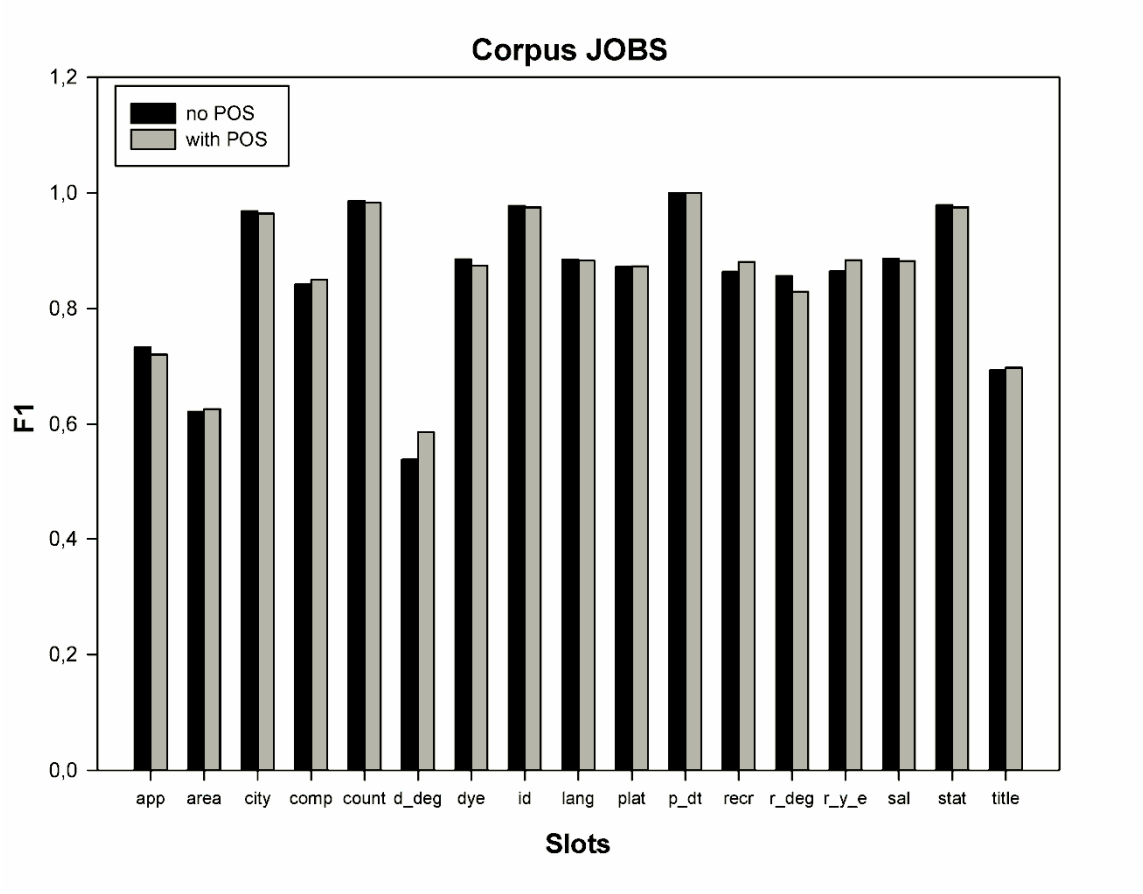



**Fig. 38.** Comparaisons par slot de F-measure avec et sans POS sur le corpus JOBS.

**Corpus CFP : Résultats par slot avec et sans POS**

Comme le montrent les Fig. 39 et 40, le tagage POS a apporté une légère augmentation sur la F-measure pour la majorité des slots. Par exemple, le slot *confacro* (*conference acronym*) a présenté le gain de plus de 5%.

Le plus bas résultat de l'algorithme en considérant tous les slots individuellement a été celui du slot *confhome* (*conference homepage*). On peut expliquer ce score à cause du nombre moins représentatif (100) d'exemples dans le corpus tout entier. De plus, on a constaté que le module responsable pour la tokenisation de documents du M-TIES, ne reconnais pas les adresses électronique comme une entité. Une possibilité d'avoir meilleurs résultats pour ce type de slot, serait de permettre le tokeniseur pour le reconnaître en augmentant la taille de la fenêtre *w*, par exemple.

| Slot | Préc | Rappel | F1 |
|---|---|---|---|
| *confacro* | 0,935 | 0,386 | 0,547 |
| *confhome* | 0,745 | 0,143 | 0,240 |
| *confname* | 0,969 | 0,683 | 0,801 |
| *workacro* | 0,902 | 0,283 | 0,431 |
| *workcame* | 0,869 | 0,392 | 0,541 |
| *workdate* | 0,930 | 0,725 | 0,815 |
| *workhome* | 0,718 | 0,729 | 0,724 |
| *workloca* | 0,988 | 0,906 | 0,945 |
| *workname* | 0,875 | 0,761 | 0,814 |
| *worknoti* | 0,902 | 0,545 | 0,679 |
| *workpaper* | 0,882 | 0,464 | 0,608 |

(a)

| Slot | Préc | Rappel | F1 |
|---|---|---|---|
| *confacro* | 0,930 | 0,442 | 0,600 |
| *confhome* | 0,717 | 0,122 | 0,208 |
| *confname* | 0,951 | 0,685 | 0,796 |
| *workacro* | 0,904 | 0,280 | 0,428 |
| *workcame* | 0,855 | 0,425 | 0,559 |
| *workdate* | 0,919 | 0,731 | 0,814 |
| *workhome* | 0,718 | 0,739 | 0,728 |
| *workloca* | 0,990 | 0,917 | 0,952 |
| *workname* | 0,890 | 0,826 | 0,857 |
| *worknoti* | 0,927 | 0,572 | 0,708 |
| *workpaper* | 0,911 | 0,478 | 0,627 |

(b)

**Fig. 39.** Résultats sur le Corpus *CFP* sans (a) et avec (b) POS.

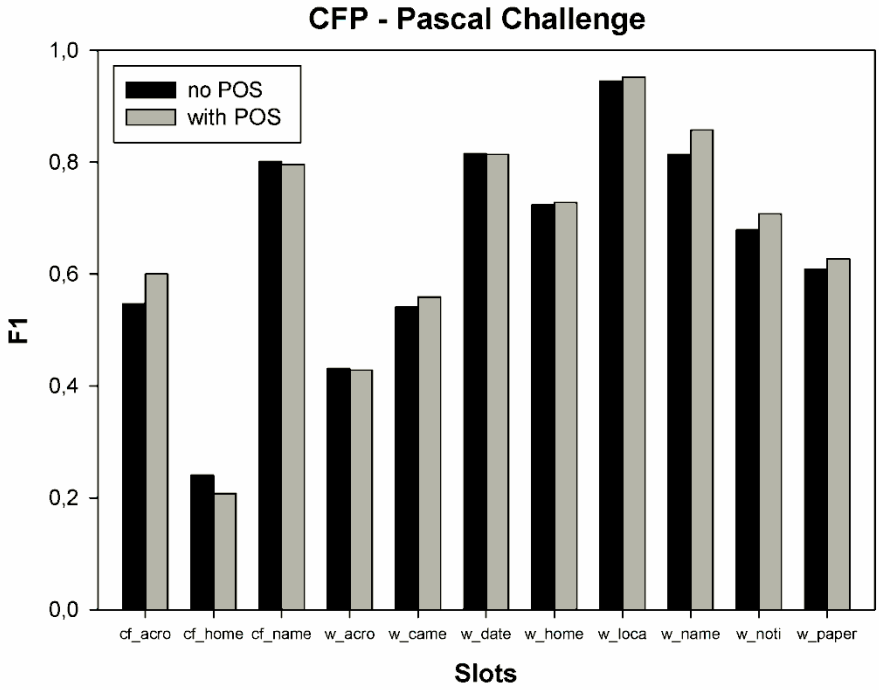



**Fig. 40.** Influence du POS sur le Corpus *CFP :* sans (a) et avec (b) POS.

**Bilan sur les corpora avec information POS**

La Fig. 41 présente le bilan général pour apprécier le gain effectif du tagage POS sur les *corpora*. On constate que pour le corpus CFP on a obtenu de meilleurs résultats avec POS. En revanche, pour les deux autres *corpora* (Fig. 36, 37 et 38) la différence a été pratiquement nulle. Ces derniers résultats peuvent surprendre mais on peut les justifier on analysant la nature très structuré des corpora Seminars et Jobs. En effet, l'algorithme d'induction d'extracteurs peut déjà avoir de très bonnes performances sans se servir d'un espace d'hypothèses plus large, sans la prise en compte de l'information POS, par exemple. On peut

aussi remarquer que ces résultats sont semblables à ceux obtenus par [Li et al., 2004]. Li et al. ont réalisé la même expérience qu'on vient de discuter sur les corpora Seminars et Jobs en utilisant l'algorithme SVM et ils ont également constaté un gain très faible sur le corpus Seminars et même un impact négative du tagage POS sur le corpus Jobs. Plus précisément, la baisse de performance a été de moins de 1% pour ce dernier cas.

De 17 slots qui constituent le schéma d'extraction pour le corpus JOBS, on note que plus de la moitié des champs ont un contenu très structuré, voire régulier. C'est le cas des slots *post_date*, *country*, *id*, *state*, *city*, *req_years_experience*, *langage*, *salary* et *recruiter*. Le système a eu un score parfait de 100% sur le premier slot de la liste (post_date).

| *Corpus* | *Préc* | *Rappel* | *F1* |
|---|---|---|---|
| *Seminars* | 0,974 | 0,953 | 0,963 |
| *Jobs* | 0,945 | 0,778 | 0,853 |
| *CFP* | 0,891 | 0,571 | 0,696 |

**(a)**

| *Corpus* | *Préc* | *Rappel* | *F1* |
|---|---|---|---|
| *Seminars* | 0,971 | 0,964 | 0,967 |
| *Jobs* | 0,939 | 0,780 | 0,853 |
| *CFP* | 0,896 | 0,591 | 0,712 |

**(b)**

**Fig. 41.** Perfomance général d'extraction de M-TIES sur les *corpora* : Seminars, Jobs et CFP sans (a) et avec (b) POS.

### 5.3.2 Différents ensembles d'attributs

Un autre aspect de grande importance à analyser est d'étudier l'influence des attributs (espace d'hypothèses) sur le *corpora*. On a choisi la validation *hold-out* en divisant chaque corpus en deux moitiés : la première, pour l'apprentissage ; la deuxième, pour le test. On a fixé le paramètre L = 3 et le nombre d'itérations en 100.

Comme le montrent les résultats de la Tab. 7, les attributs concernant les informations de *tokenkind* (catégorie de tokens) ont été utiles pour améliorer le score du système d'environ 3% pour les corpus Seminars et Jobs par rapport à l'information simple du *mot*. La Fig. 42 montre aussi que, quand on considère seulement le corpus CFP, l'information POS a été encore plus utile avec un écart de performance de plus de 5% par rapport à la simple information du token. Ces résultats sont explicables en raison de la nature moins structurée du corpus CFP et, conséquemment, l'avantage d'avoir des patrons linguistiques fournis par le l'étiquetage POS. Par contre, le corpus JOBS a obtenu le gain plus faible parmi les *corpora*.

**Tab. 7.** Influence de différentes *features* sur le corpora résultats exprimés en F-Measure (micro).

| *Features* | *Word* | *Word + Case* | *Word + Case +Tokenkind* | *Word + Case +Tokenkind + POS* |
|---|---|---|---|---|
| Seminars | 0,933 | 0,938 | 0,963 | 0,967 |
| Jobs | 0,822 | 0,832 | 0,853 | 0,853 |
| CPF | 0,656 | 0,670 | 0,696 | 0,712 |

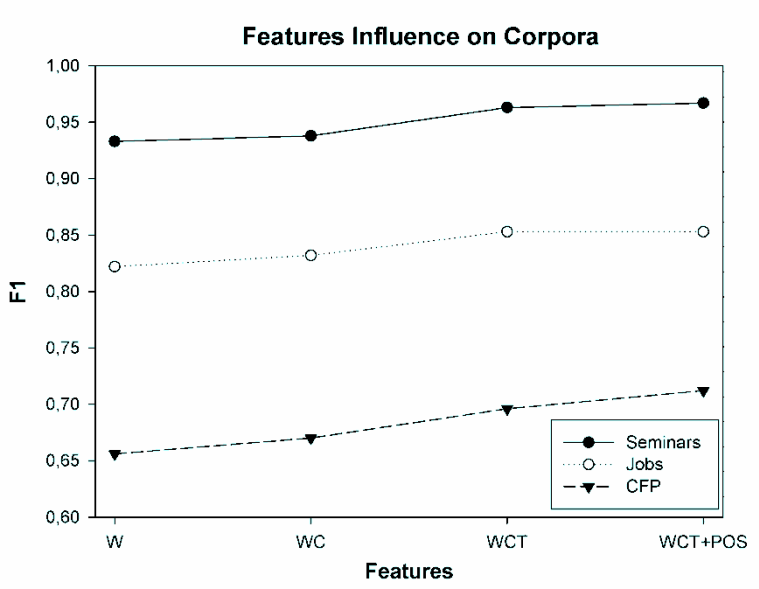



**Fig. 42.** L'effet de différents ensembles d'attributs utilisés sur les *corpora.*

### 5.3.3 Courbe d'apprentissage

La Tab. 8 et la Fig. 43 montrent, respectivement, les scores individuels et la *courbe d'apprentissage* de l'algorithme. La tâche d'EI adaptative est requise pour concevoir un modèle initial à partir d'un petit nombre d'exemple. Dans ce cas, il est attendu que la performance du système s'améliore progressivement au fur et à mesure que de plus en plus d'instances soient disponibles pour l'apprentissage. Par exemple, cela peut avoir lieu quand l'utilisateur annote de nouveaux documents. En fait, on veut ici simuler ce scénario avec l'objectif d'évaluer l'algorithme d'apprentissage sur un nombre croissant d'exemples. Le paramètre $L$ = 3 et le nombre d'itérations en 100 ont été utilisés. Ensuite, les documents du corpus ont été répartis aléatoirement en deux partitions égales : une partition sera désignée *l'ensemble de test* avec un nombre invariable de documents. De l'autre partition, *l'ensemble d'apprentissage*, il est choisi un nombre croissant de documents en faisant en sorte que les documents choisis dans l'étape précédente soient compris dans l'ensemble d'exemples d'une étape postérieure. De cette façon, il est simulé le scénario où l'utilisateur qui annote des plus en plus de documents et les rajoutent au corpus.

A chaque étape on mesure la F-mesure du système, ce qui amène aux résultats suivants :

**Tab. 8.** Résultats en F-measure par slots du corpus Seminars en augmentant le nombre de documents.

| ***Slots*** | ***10*** | ***20*** | ***40*** | ***60*** | ***80*** | ***100*** |
|---|---|---|---|---|---|---|
| *stime* | 0,914 | 0,949 | 0,954 | 0,980 | 0,982 | 0,985 |
| *etime* | 0,940 | 0,940 | 0,940 | 0,959 | 0,959 | 0,959 |
| *location* | 0,547 | 0,562 | 0,753 | 0,723 | 0,715 | 0,703 |
| *speaker* | 0,513 | 0,775 | 0,844 | 0,912 | 0,907 | 0,922 |

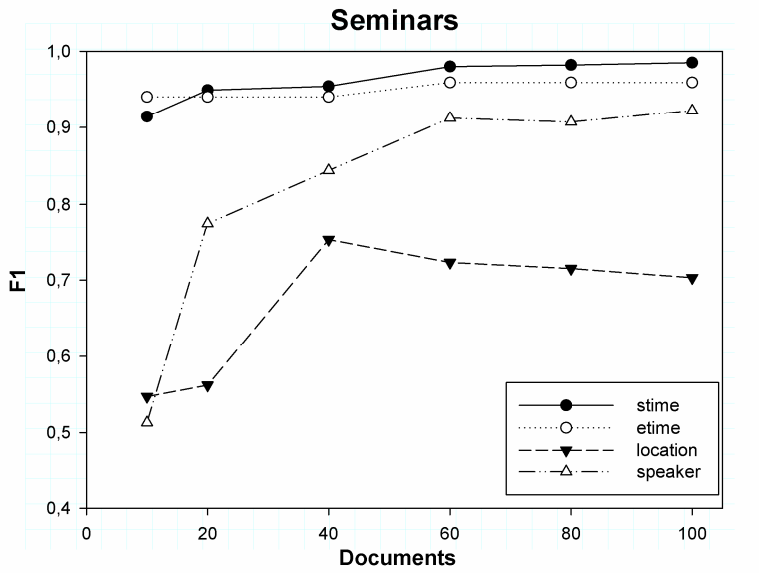



**Fig. 43.** Courbe d'apprentissage sur le corpus Seminars.

En analysant la courbe d'apprentissage de l'algorithme, on s'aperçoit qu'en général la performance de l'algorithme augmente progressivement au fur et à mesure que de nouveaux documents sont utilisés pour l'apprentissage. Plus particulièrement, les slots de temps (*stime* et *etime*) sont les plus faciles à être généralisé que les autres, vu que le système peut les apprendre dès le premier ensemble de 10 documents. En revanche, pour le slot *speaker*, il a fallu 60 documents pour atteindre un score proche de ceux des slots de temps. De plus, on note que juste après 60 documents, la courbe se stabilise pour les slots *stime, etime et speaker*. Le gain est mineur à partir de ce point sur la courbe. On note aussi une légère dégradation de performance pour le slot *location* après un nombre de 40 documents. Cela signifie que l'algorithme présente un problème de sur-apprentissage (*overfitting*), une réduction de son efficacité quand plus de documents sont utilisés pour l'apprentissage.

## 5.4 Évaluation comparative

### 5.4.1 Critères d'évaluation recommandés

La définition d'une méthodologie d'évaluation et la disponibilité des *corpora* standards annotés ne garantissent pas que les expériences réalisées avec de différentes approches et algorithmes proposés dans la littérature soient comparées d'une façon juste et fiable [Lavelli et al., 2004]. Voici les points les plus importantes examinées par Lavelli et al.:

**Définition exacte des partitions de corpus**
Il est bien connu que différentes partitions de corpus peut influencer les résultats. Ainsi il est crucial que l'on définisse le point de séparation exacte entre l'ensemble d'apprentissage et l'ensemble de test, vu les proportions numériques entre les deux ensembles (par exemple, un

*random split* de 50/50 contre un de 80/20) et la procédure adoptée pour partitionner les documents, par exemple, validation croisée vs random subsampling.

**Tâches de prétraitement**
Certaines sous-tâches de préparation d'un corpus (par exemple, tokenization) sont souvent considérées comme évidentes et non problématiques, mais il n'est pas le cas ici et cela peut influencer la performance des algorithmes d'IE. Cette question importante distingue un algorithme et les attributs qu'il emploie dans leur contribution à sa performance. En EI, par exemple, certains algorithmes ont utilisé des attributs orthographiques simples, tandis que d'autres emploient des attributs plus complexes tels que le tagage POS ou étiquettes sémantiques extraites de *gazeteers* [Califf, 1998; Ciravegna, 2001; Peshkin & Pfeffer, 2003].

**Le système « scorer »**
Ce point concerne le logiciel qui a été utilisée pour l'évaluation. Le seul outil publiquement disponible pour tel but est le *MUC Scorer* [Douthat, 1998]. Normalement, les chercheurs d'IE ont mis en œuvre leurs propres systèmes *scorer*, en s'appuyant sur un certain nombre d'hypothèses implicites qui ont une forte influence sur les résultats de l'évaluation.

**Résultats rapportés**
Quelques articles rapportent seulement la F-measure, mais pas la précision et le rappel, alors que la différence entre la précision et le rappel est un aspect fondamental de la performance.

### 5.4.2 Description des systèmes à comparer

Cette section fait une brève présentation des systèmes à comparer avec M-TIES qui n'ont pas déjà été mentionnés dans la section 3.3.

**Rapier** C'est un système d'EI single-slot qui vise à extraire des informations de textes libres [Califf & Mooney, 1999]. Son algorithme d'apprentissage incorpore des techniques de la *programmation logique inductive* [Muggleton, 1994; Thomas, 2005] et il apprend des patrons qui ne sont pas limités par une fenêtre fixe mais que inclut des contraints sur les mots et sur le tagage POS que entoure la donnée à extraire. Ces patrons consistent de trois slots distincts: *Pre-*, *Post-* et *Filler*. Les premiers jouent le rôle de délimiteurs gauche et droite, tandis que le dernier définit la structure du champ à extraire.

**GATE-SVM** [Li et al., 2004a, 2004b] est un système d'EI développé dans le cadre du Project SEKT qui utilise l'algorithme SVM pour la classification supervisée de tokens. Ce système emploie une variante du SVM, le SVM avec *uneven margins* qui a une meilleure performance de généralisation que le SVM original sur un ensemble où le nombre d'exemples positifs sont beaucoup moins représentatifs que les négatifs [Li et al., 2003]. Il a été testé sur plusieurs *corpora*: Seminars, Jobs, CoNLL-2003, pour en citer quelque uns. Il peut utiliser plusieurs combinaisons d'attributs : *word*, *capitalisation*, POS, *gazeteers*, lemmatisation, etc. L'extraction d'attributs est performée par GATE.

**Yaoyong** Le système Yaoyong [Ireson et al., 2005] est le prédécesseur du GATE-SVM et ses classificateurs utilisent une fenêtre de contexte de 10 tokens à gauche et à droite. Cette version-ci faisait l'usage d'attributs des tokens suivants: *token,* c*apitalisation, tokenkind* et informations des entités. Le tagage POS ne figurait pas dans cette liste. Cette version a

participé dans las compétition Pascal Challenge sur l'évaluation d'apprentissage machine pour l'EI.

Par la suite, le système M-TIES est comparée avec les systèmes SIE, $(LP)^2$ (section 3.3), GATE-SVM, Yaoyong et Rapier sur le *corpora* choisis en essayant de suivre les même protocole expérimental (résultats rapportés, méthode d'évaluation, etc.) ou les plus proche possible pour que l'on puisse avoir de comparaisons plus pertinentes. Il faut rappeler que des comparaisons vraiment justes et fiables sont problématiques en raison de critères d'évaluations déjà présentés dans la section précédente.

### 5.4.3 Comparaison sur les corpora Seminars et Jobs

Les *corpora* Seminars et Jobs ont été utilisés par plusieurs système d'apprentissage, soit ceux orientés vers l'induction d'extracteurs, soit ceux plus orientés au traitement linguistique. Les comparaisons sur ces *corpora* deviennent problématiques parce que différents systèmes suivent différents protocoles expérimentaux.

Pour les expériences avec le système M-TIES, la méthode d'évaluation hold-out (avec 50% de documents pour l'apprentissage et 50% pour le test) a été utilisée; puis, le système est exécuté (apprentissage et test) 10 fois et le résultat final est la moyenne de tous les exécutions - *random split* (50/50) 10 fois. De nombreux systèmes d'EI évalués sur ces *corpora* ont employé cette méthode d'évaluation, donc les résultats de cette section a suivi la même méthodologie avec l'objectif d'obtenir des résultats plus fiables et justes.

Les tables 9 et 10 résument la configuration des systèmes à comparer sur les *corpora* Seminars et Jobs.

**Tab. 9.** Résumé de configuration des systèmes évalués sur le corpus SEMINARS.

| ***Système*** | ***Méthode d'évaluation*** | ***Attributs utilisés*** | ***W*** |
|---|---|---|---|
| $(LP)^2$ | Random split (50/50) - 10 fois | Word, capitalisation et POS | 5 |
| GATE-SVM | Random split (50/50) - 10 fois | Word, capitalisation, token type, lemma et POS | 5 |
| Rapier | Random split (50/50) - 10 fois | Word, POS et wordnet | - |
| SIE | 2-fold cross validation - 5 fois | Word, capitalisation, lemma, alpha/numerique et ponctuation | 10 |
| M-TIES | Random split (50/50) - 10 fois | Word, capitalisation et POS | 3 |

**Tab. 10.** Résumé de configuration des systèmes évalués sur le corpus JOBS.

| ***Système*** | ***Méthode d'évaluation*** | ***Attributs utilisés*** | ***W*** |
|---|---|---|---|
| $(LP)^2$ | Random split (50/50) – 10 fois | Word, capitalisation et POS | 5 |
| GATE-SVM | Random split (50/50) - 10 fois | Word, capitalisation, token type, lemma et POS, NER et gazeteer | 3 |
| Rapier | 10-fold cross validation | Word, POS et wordnet | - |
| M-TIES | Random split (50/50) - 10 fois | Word, capitalisation et POS | 3 |

**Résultats comparatifs sur les corpora Seminars et Jobs**

Les tables 11 et 12 présentent les performances des systèmes en termes de F measure sur le corpora Seminars et Jobs, respectivement. Les meilleurs résultats (F-measure) pour chaque slot sont en gras. Pour les performances des systèmes SIE, GATE-SVM, $(LP)^2$ et Rapier, on s'appuie sur les résultats donnés dans [Giuliano et al., 2006; Li et al., 2004a; Ciravegna, 2003b].

**Tab. 11.** Perfomances par slot de 5 systèmes sur le corpus *Seminars*.

| | ***speaker*** | ***location*** | ***stime*** | ***etime*** | ***All Slots*** |
|---|---|---|---|---|---|
| SIE | - | - | - | - | 86,6 |
| GATE-SVM | 69,0 | 81,3 | 94,8 | 92,7 | 86,2 |
| $(LP)^2$ | 77,6 | 75,0 | **99,0** | 95,5 | 86,0 |
| Rapier | 53,0 | 72,7 | 93,4 | 96,2 | 77,3 |
| M-TIES | **86,2** | **88,8** | 93,9 | **96,7** | **91,4** |

**Tab. 12.** Perfomances par slot de 4 systèmes sur le corpus *Jobs* en utilisant un ensemble d'attributs composé d'information de capitalisation et POS.

| ***Slot*** | ***$(LP)^2$*** | ***GATE_SVM*** | ***Rapier*** | ***M-TIES*** |
|---|---|---|---|---|
| id | **100,0** | 97,7 | 97,5 | 98,1 |
| title | 43,9 | 49,6 | 40,5 | **67,4** |
| company | 71,9 | 77,2 | 69,5 | **78,9** |
| salary | 62,8 | 86,5 | 67,4 | **89,2** |
| recruiter | 80,6 | 78,4 | 68,4 | **86,1** |
| state | 86,7 | 92,8 | 90,2 | **96,9** |
| city | 93,0 | 95,5 | 90,4 | **96,5** |
| country | 81,0 | 96,2 | 93,2 | **98,8** |
| language | **91,0** | 86,9 | 80,6 | 88,5 |
| plataform | 80,5 | 80,1 | 72,5 | **86,9** |
| application | **78,4** | 70,2 | 69,3 | 73,1 |
| area | **66,9** | 46,8 | 42,4 | 51,6 |
| req_y_exp | 68,8 | 80,8 | 67,1 | **86,4** |
| des_y_exp | 60,4 | 81,9 | 87,5 | **89,9** |
| req_degree | 84,7 | **87,5** | 81,5 | 78,6 |
| des_degree | **65,1** | 59,2 | 72,2 | 47,6 |
| post date | 99,5 | 99,2 | 99,5 | **100,0** |
| **All slots** | **84,1** | **80,8** | **75,1** | **83,8** |

**a) Corpus Seminars**

En analysant les résultats de la Tab. 11, où « all slot » est la F-measure (micro), on note que l'évaluation du M-TIES a été notamment supérieur sur trois slots tandis que pour les systèmes SIE, GATE-SVM (qui utilisent l'algorithme SVM comme leur composant d'apprentissage) et $(LP)^2$ ont obtenu une performance globale similaire [Giuliano et al., 2006; Li et al., 2004a; Ciravegna, 2003b]. On veut remarquer que le système GATE-SVM a utilisé un ensemble d'attributs plus riche que les autres systèmes [Li et al., 2004b]. Avec l'ensemble d'attributs complet du M-TIES, ce atteint une performance de 96,7%. De façon similaire $(LP)^2$ obtient 89,7% sur ce corpus en employant NER et gazeteers. On n'a pas de scores détaillé par slots du SIE [Giuliano et al., 2006], ce qui explique les valeurs manquantes dans la table de résultats.

Les honorables scores des slots *speaker* et *location* confirment l'adéquation de l'algorithme BWI pour les tâches d'EI sur des documents plus structurés. Pour l'algorithme BWI, si un slot cible (slot à extraire) est tout simplement précédé ou suivi d'un ensemble de tokens, ou par des tokens d'un type distinct, représenté par les caractères *wildcard* (*joker*) disponibles dans son espace d'hypothèses, les détecteurs de séparateurs apprennent aisément ce contexte.

Pour les documents fortement structurés ou partiellement structurés (le cas du corpus Seminars), les slots sont souvent précédés par l'identification des étiquettes (par exemple « Speaker: Dr. X »), ou suivi par des éléments d'information facilement identifiables.

Alors que d'autres méthodes d'EI reposant sur règles sont principalement conçues pour identifier des contextes en dehors des slots cibles, BWI apprend davantage certaines régularités qui se produisent à *l'intérieur* de slots cibles. Conséquemment, les détecteurs de séparateurs peuvent se « prolonger » dans le bord du slot cible aussi bien que dans le contexte local. Autrement dit, les détecteurs à gauche (détecteurs *fore*) peuvent apprendre à ce que ressemblent les premiers tokens d'un slot cible, si les slots tendent à avoir un début régulier, et les détecteurs à droite (détecteurs *aft*) peuvent apprendre les derniers tokens si ils ont une forme également régulière. De plus, pour des slots « courts », des détecteurs individuels mémorisent souvent des exemples du slot cible quand le contexte n'est pas utile.

La version actuelle du tokeniseur du M-TIES est optimisée pour identifier le plut tôt possible des instances de dates, heures et abréviations plus courantes. Cela pourrait expliquer le meilleur résultat pour le slot *etime* qu'il a obtenu.

Enfin, le slot *stime*, il semble qu'il faut plus de contexte pour obtenir un bon score. L'approche d'induction d'extracteurs plus orienté au traitement linguistique du $(LP)^2$ obtient alors le meilleur score.

**b) Corpus Jobs**

Dans la Tab. 12, les meilleurs scores pour chaque slot sont en gras, il faut remarquer que le score « all slots » de tous les systèmes sont exprimés en F-measure micro, sauf pour le GATE-SVM qui emploie la F-measure macro [Li et al., 2004a; Ciravegna, 2003b].

D'une façon générale, tous les systèmes ont présenté des performances uniformes sur ce corpus. M-TIES a atteint les meilleurs scores en 11 slots sur 17 tandis que le système Amilcare a été plus performant en 6 slots. Cependant, ces différences de performance sont très faibles. D'autres slot tels que *id* (identification du message) et *post-date* sont fortement réguliers (ils font partie des métadonnées de message), ce qui explique les résultats supérieurs de tous les systèmes. En particulier, M-TIES et $LP^2$ ont obtenu un score parfait pour les slot *post-date* et *id*, respectivement.

Pour M-TIES, le plus grand écart positif de performance a été celui du slot *title*. Par contre, le plus grand écart négatif a été pour le slot *des-degree*. En analysant les annotations pour le premier slot, on voit qu'il a une taille très variable et qui son contenu est plus

important que son contexte pour bien l'identifier. D'autre part, la faible représentation d'occurrences du slot *des-degree* (21 au total) explique ce score faible.

Des tests de signification statistiques peuvent montrer que les systèmes comparés sur ce corpus ne sont pas nettement différents les uns des autres.

### 5.4.4 Comparaison sur le corpus Call For Papers (CFP)

La Tab. 13 montre les résultats obtenus par les systèmes SIE, Yaoyong, Amilcare et M-TIES. Tous les systèmes ont utilisés la validation croisée (k = 4) comme méthode d'évaluations pour la tâche de *template*. Pour les performances des systèmes comparés dans cette section, on s'appuie sur les résultats donnés dans [Ireson et al., 2005].

Le corpus CFP a subi un prétraitement par le système GATE qui fournit la tokenisation, des attributs orthographiques, taggage POS et NER (*Location, Person, Date, etc.)*. Ces attributs constituent un ensemble de base en termes de traitement linguistique.

Les systèmes SIE et Yaoyong utilisent des classificateurs SVM, chaque balise a été apprise indépendamment et elles sont ensuite combinées. Le premier emploie la technique *instance filtering* pour réduire le nombre d'instance négatives en supprimant jusqu'à 50% des instances à fin d'alléger le déséquilibre entre les classes et accélérer le traitement. Une fenêtre de 10 tokens (un contexte gauche/droite de 20 tokens au total) a été utilisée pour ces deux systèmes. En revanche, Amilcare et M-TIES on utilisé une fenêtre de 5 et 3, respectivement.

En considérant les attributs utilisés, Yaoyong et SIE ont utilisé tous les attributs disponibles, sauf l'information POS. Par contre, M-TIES se sert d'informations sur *word* (token), capitalisation, *token types* (abréviation, alpha numérique, symboles, ponctuation), entités (date et l'heure seulement) et taggage POS. Almicare utilise tous les attributs du GATE.

La Fig. 44 résume les résultats de la Tab. 13 en montrant la performance (en F-measure) par slot des systèmes Amilcare $(LP)^2$, Yaoyong et SIE.

**Tab. 13.** Performance des systèmes sur le corpus CFP par slot en termes de PRE - Précision, RAP - Rappel et FME – F-measure.

| | | **WORKSHOP** | | | | | | | | **CONFERENCE** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Système** | **Score** | **name** | **acro** | **date** | **home** | **loca** | **pape** | **noti** | **came** | **name** | **acro** | **home** |
| Amilcare $(LP)^2$ | PRE | 0,656 | 0,887 | 0,769 | 0,864 | 0,621 | 0,876 | 0,889 | 0,876 | 0,792 | 0,922 | 0,656 |
| | RAP | 0,241 | 0,884 | 0,632 | 0,619 | 0,402 | 0,851 | 0,889 | 0,865 | 0,422 | 0,888 | 0,280 |
| | FME | 0,352 | **0,865** | 0,694 | 0,721 | 0,488 | **0,864** | **0,889** | **0,870** | 0,551 | **0,905** | **0,393** |
| Yaoyong | PRE | 0,629 | 0,738 | 0,810 | 0,656 | 0,611 | 0,719 | 0,867 | 0,764 | 0,649 | 0,619 | 0,368 |
| | RAP | 0,539 | 0,523 | 0,666 | 0,870 | 0,674 | 0,763 | 0,821 | 0,736 | 0,411 | 0,348 | 0,093 |
| | FME | 0,580 | 0,612 | 0,731 | **0,748** | 0,641 | 0,740 | 0,843 | 0,750 | 0,503 | 0,445 | 0,149 |
| SIE | PRE | 0,852 | 0,733 | 0,850 | 0,672 | 0,812 | 0,841 | 0,921 | 0,911 | 0,795 | 0,667 | 0,556 |
| | RAP | 0,539 | 0,259 | 0,451 | 0,419 | 0,406 | 0,617 | 0,795 | 0,687 | 0,344 | 0,235 | 0,067 |
| | FME | 0,660 | 0,383 | 0,589 | 0,516 | 0,542 | 0,712 | 0,853 | 0,783 | 0,481 | 0,348 | 0,119 |
| M-TIES | PRE | 0,889 | 0,906 | 0,918 | 0,718 | 0,990 | 0,906 | 0,925 | 0,849 | 0,953 | 0,930 | 0,706 |
| | RAP | 0,825 | 0,275 | 0,729 | 0,735 | 0,916 | 0,477 | 0,569 | 0,414 | 0,691 | 0,443 | 0,122 |
| | FME | **0,856** | 0,422 | **0,813** | 0,726 | **0,952** | 0,625 | 0,705 | 0,556 | **0,801** | 0,600 | 0,209 |

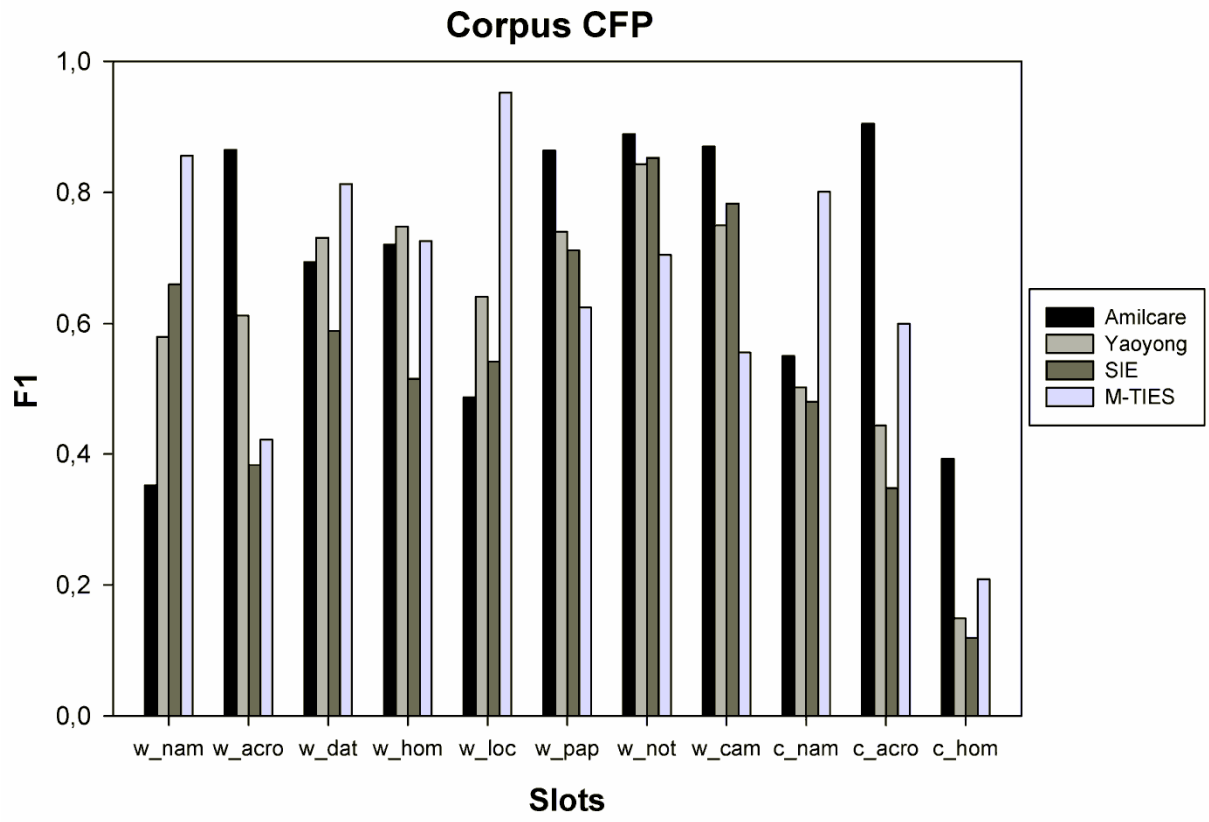



**Fig. 44.** Score F-Measure des systèmes par slot sur le corpus CFP.

Tous les systèmes ont présenté une grande variation concernant la capacité d'identifier certains slots (Fig. 44). Amilcare atteint les meilleurs scores en 6 sur 11 slots tandis que M-TIES le fait pour 4 slots. Le système Yaoyong a obtenu la meilleur F-measure juste pour 1 slot cible.

Lors de l'examen des F-measures, la meilleure performance est observée sur les 4 dates du corpus (*workshop date, workshop papper submission date, workshop camera ready copy date et worshop notification acceptance date*). Ces dates sont relativement faciles à extraire car elles sont sous en format bien défini et fortement prescrites par le texte qui les entoure. D'autre part, la plus baisse performance de tous les systèmes a eu lieu pour les 3 slots relatifs aux *Conferences*, ceux-ci ont un nombre d'exemples relativement bas dans le corpus. Ce qui indique un nombre insuffisant d'exemples (baisse représentation) pour s'achever à de bonnes généralisations.

Amilcare a obtenu le plus bas scores par rapport aux autres systèmes pour les slots *workshop name, workshop location et conference name*, ce qui montre d'une façon claire que leurs techniques ne garantissent pas la bonne performance sur tous les types de slots. En examinant les documents, on peut noter que ces slots problématiques pour Amilcare ne sont pas spécifiés par leurs contextes, mais ils sont plutôt déterminés par leurs contenus et leurs places dans le document. Au contraire, pour M-TIES, il montre un grand écart de performance pour ces slots grâce à la façon de l'algorithme BWI qui peut *voir* le contenu du champ à extraire[10].

[10] Voir les commentaires de la perfomance du M-TIES sur le corpus Seminars.

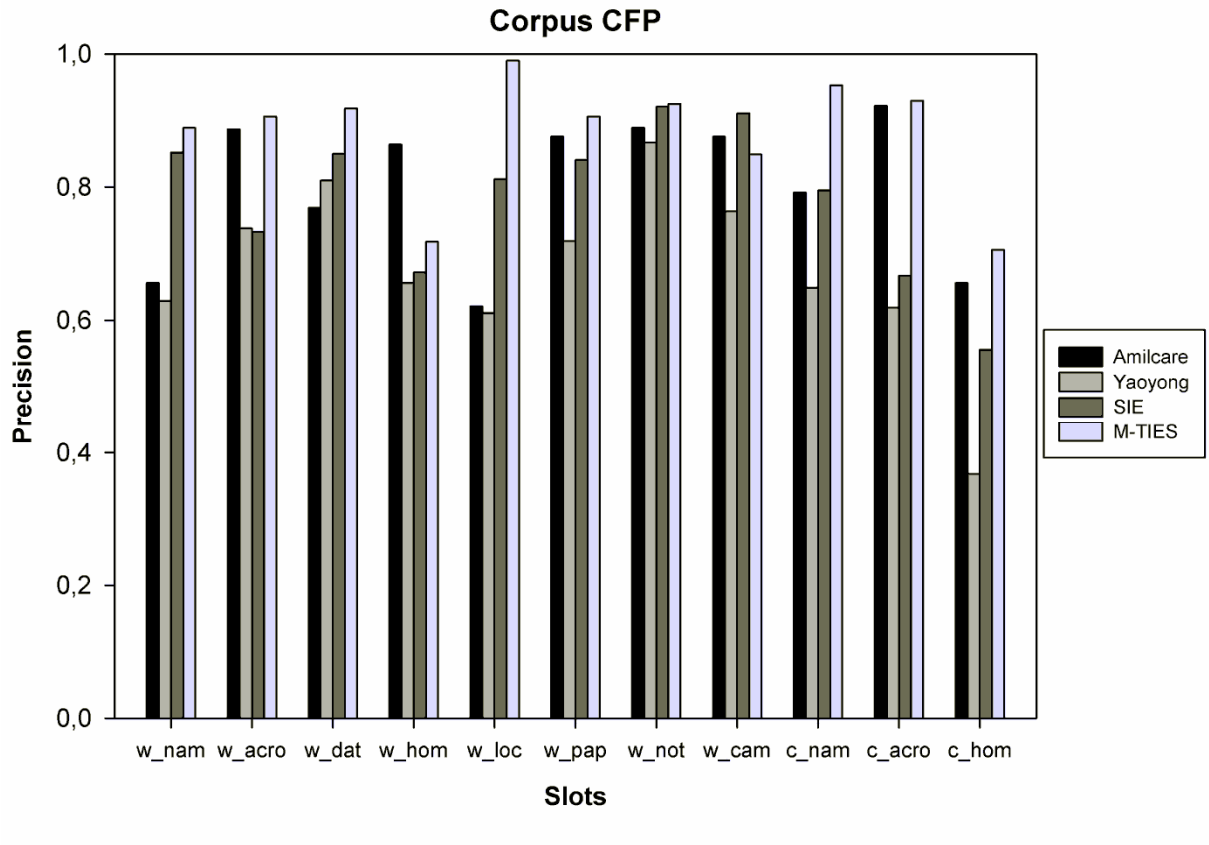



**Fig. 45.** Scores de Précision des systèmes par slot sur le corpus CFP

Les figures 45 et 46 présentent, respectivement, la performance des systèmes en termes de la précision et du rappel pour les 4 systèmes à comparer. Comme le montrent ces résultats, les systèmes préfèrent d'avoir plus de précision que de rappel. Ce fait est probablement dû aux systèmes d'IE qui, en général, sont destinés à performer une extraction qui attribuent un coût plus élevé aux faux positifs. En effet, on constate plus de variations dans le rappel que dans la précision pour tous les systèmes, à l'exception de Yaoyong qui favorise le rappel. Cela nous amène à une forte indication que ces systèmes poursuivent de différentes stratégies en termes de mesures de performance. Enfin, il faut aussi noter que l'on peut seulement avoir un système d'EI avec plus de rappel en détriment de la précision et vice-versa.

L'explication pour le plus bas rappel du M-TIES pour le slot *c-hom* (home page de la conférence) (Fig. 46), c'est que les liens de pages sont découpé en plusieurs tokens et la taille de seulement 3 tokens ne se faire pas suffisante pour générer de bonnes règles pour ce type d'information. En plus, ce slot est le plus représentatif de tout le corpus CFP, en ayant seulement 100 exemples annotés. Une façon d'améliorer ce résultat serait alors de faire le tokeniseur à reconnaître ce type information comme une entité, ou en augmentant les nombres d'exemples d'apprentissage.

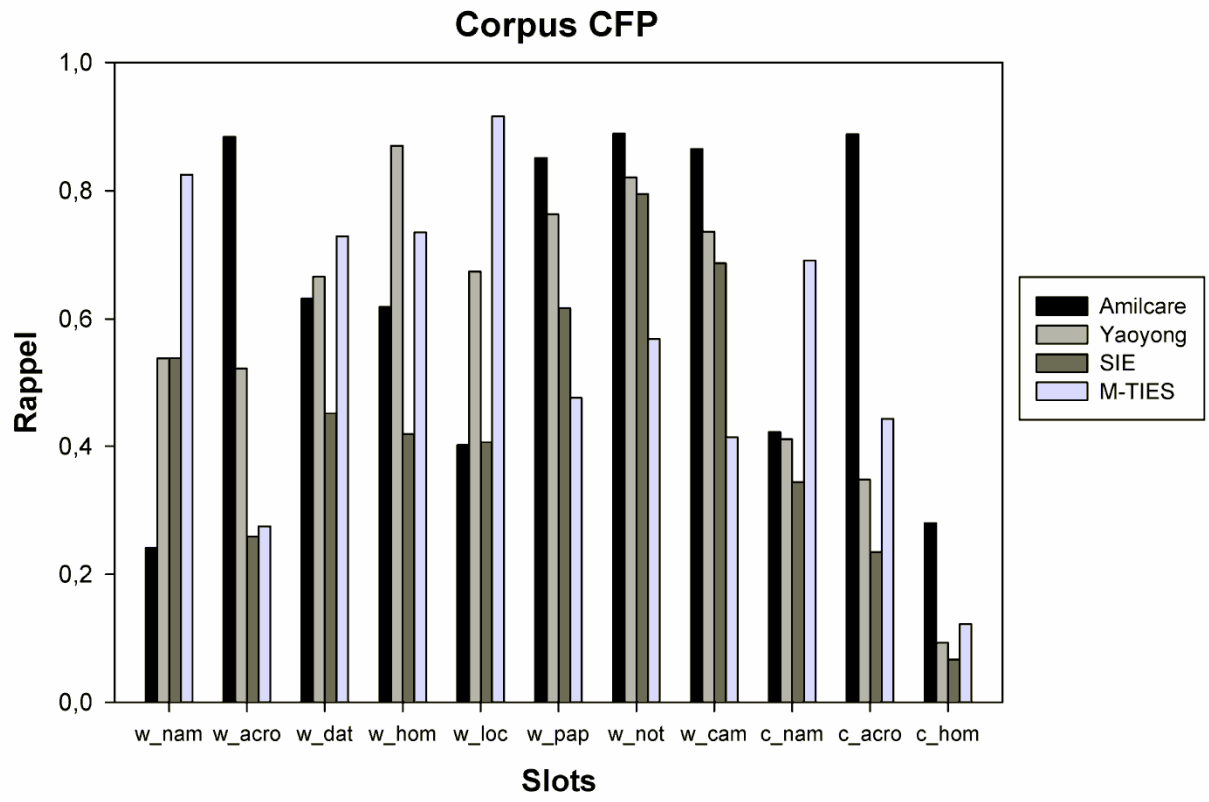



**Fig. 46.** Scores de Rappel des systèmes par slot sur le corpus CFP.

La Tab. 14 est illustrée par la Fig. 47 qui montre que le système M-TIES a été le plus précis de tous les systèmes participants à cette évaluation comparative, mais il a eu le plus bas score de rappel. En somme, sa performance en terme de F-measure a été comparable aux systèmes SIE et Yaoyong et un peu plus bas que le système Amilcare sur le corpus CFP.

**Tab. 14.** Comparaison entre les 4 systèmes sur le corpus CFP.

| | ***Préc*** | ***Rappel*** | ***F1*** |
|---|---|---|---|
| Amilcare | 84,3 | 70,3 | **76,7** |
| Yaoyong | 70,2 | **71,7** | 70,9 |
| SIE | 75,5 | 65,2 | 70,0 |
| M-TIES | **89,6** | 59,1 | 71,2 |

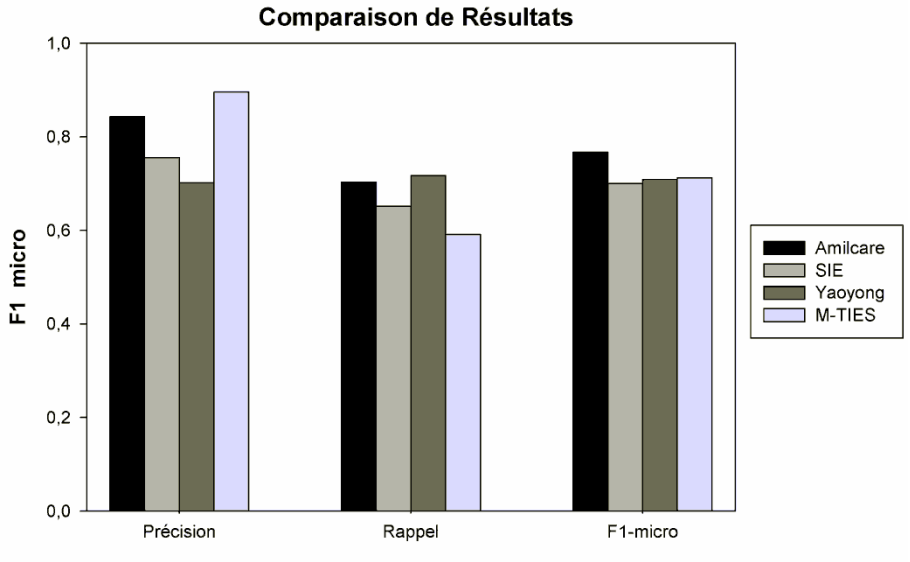



**Fig. 47.** Comparaisons des mesures de précision, rappel et F1-measure (micro) entre les 4 systèmes

**Bilan sur l'évaluation comparative**

Les expériences d'évaluation comparative ont montré que le système M-TIES est supérieur à d'autres systèmes de l'état de l'art sur le corpus plus structuré Seminars, et qu'il atteint des score comparables sur le corpus semi-structuré Jobs. En plus, les extracteurs produits par le système proposé ont la tendance à atteindre plus de précision que les autres systèmes tout en tenant un rappel raisonnable.

La raison pour cette réussite est que le système M-TIES, qui repose sur l'algorithme BWI, performe l'extraction avec plus de précision, car les règles contextuelles apprises sont fortement précises, mais il a également un rappel raisonnable dans de nombreux domaines, étant donné que des dizaines ou de centaines de règles suffisent pour avoir un bon rappel. En revanche, même avec l'aide d'étiquetage POS, M-TIES a obtenu un score inférieur sur le corpus CFP (en langage naturel). Cependant, tous ces résultats ne font que confirmer expérimentalement ce qu'il était prévu pour cette recherche.

En effet, l'algorithme BWI emploie un formalisme de règles plus expressif grâce à l'utilisation de wildcards qui généralisent mieux que l'algorithme $LP^2$, par exemple, sur des documents plus structurés. Ce formalisme de règles inclut un certain nombre de wildcards (jokers) qui contribuent radicalement aux résultats expérimentaux de l'algorithme. BWI apprend des règles simples d'extraction qui sont en grande partie équivalentes aux meilleures règles de $LP^2$ [Ciravegna, 2001]. D'ailleurs, au sein du BWI, la technique de boosting est utilisée pour mettre l'accent sur des exemples sur lesquels l'apprenant a une faible performance afin d'en tirer des règles supplémentaires; contrairement à l'algorithme $LP^2$ qui emploie une approche d'apprentissage machine plus simple reposant sur un algorithme de couverture [Ciravegna, 2003b].

Enfin, pour un ensemble de document moins structurés, notamment en langage naturel, on peut conclure que l'avantage de l'algorithme $LP^2$ sur les autres réside dans l'effet combiné de l'étape d'étiquetage contextuel, de l'étape de correction et de l'utilisation plus avancée d'information TAL. Ce dernier étant le plus important d'entre eux.

# 6 CONCLUSION ET PERSPECTIVES

## 6.1 Conclusion

L'un des objectifs visés de cette recherche a été d'étudier l'utilisation de techniques d'induction d'extracteurs permettant l'extraction d'information à partir de documents de différents niveaux de structuration (documents structurés et non structurés). Pour cela, il a été distingué tout d'abord les concepts de base, comme: les types de documents, la représentation de ces documents et le types de sorties qui sont obtenues pendant une tâche d'EI. Ensuite, différentes approches de conception d'extracteurs ont été étudiées et l'approche par induction supervisée a été retenue, approche nécessitant qu'un ensemble d'exemples de documents avec données à extraire soient annotées par l'utilisateur.

Puis, nous avons vu comment la classification supervisée sur laquelle repose la plupart des techniques d'EI par induction supervisée, permet de définir un extracteur en tant qu'un processus de classification : le problème d'induction d'extracteurs se ramène alors à un problème d'apprentissage de classification supervisée. Cette transformation d'un problème d'extraction en un problème de classification supervisée nécessite que soit spécifié la représentation des documents, la définition des éléments des documents qui seront considérés comme les exemples du problème de classification, et le codage de ces exemples. En analysant en détail le système BWI d'induction d'extracteurs reposant sur la classification supervisée, on a pu distinguer les différentes étapes élémentaires associées à une telle transformation. En plus, il a été présenté différents systèmes d'EI par induction supervisée d'extracteurs qui adoptent une représentation de document à base de tokens et réalisent l'EI en mettant en œuvre différents algorithmes d'apprentissage pour traiter des textes semi-structurés.

La contribution principale de cette recherche a été la proposition d'un système d'EI adaptatif, facilement configurable et convivial à l'utilisateur par le biais d'une architecture modulaire destinée à l'utilisation sur plusieurs types de documents. Il a été constaté que tel système modulaire reposait sur l'induction supervisée d'extracteurs, et qui permettait l'extraction d'information à partir d'un corpus d'apprentissage manuellement annoté et qui pouvait prendre en compte la syntaxe du langage naturel par le moyen d'un nouveau module responsable pour l'étiquetage morphosyntaxique sur ces documents.

Un autre objectif de ce travail était de comparer l'architecture de notre système d'EI avec d'autres systèmes de l'état de l'art au travers différentes expérimentations sur trois *corpora* de documents de référence en suivant une rigoureuse méthodologie d'évaluation de résultats bien établie dans la communauté scientifique du domaine de l'EI. Pour cela, il a été alors défini un protocole expérimental pour bien mener les expériences afin d'avoir de résultats plus fiables et pertinents. Particulièrement, ce protocole a consisté (i) à définir la tâche d'EI sur les 3 corpora de documents, (ii) à définir une méthodologie d'évaluation, (iii) à réaliser différentes expériences visant à déterminer les meilleurs paramètres du modèle pour chaque type de corpus (en prenant en compte l'information POS ou non) et, (iv) finalement faire une évaluation comparative parmi plusieurs systèmes d'EI existants.

Les expériences ont montré que, par rapport au gain effectif d'étiquetage POS sur les corpora, le corpus CFP on a obtenu le meilleur résultat. On a obtenu une amélioration de plus de 5% pour certains slots. Par contre, pour les autres deux *corpora* plus structuré, la

différence a été pratiquement nulle. Ceci peut s'expliquer par le fait que l'algorithme d'induction d'extracteurs a déjà de très bonnes performances sans se servir d'un espace d'hypothèses plus large (avec la prise en compte de l'information POS), quand il est employé sur de documents plus structurés (corpus Seminars et Jobs). En plus, il y a eu le cas où un slot avait un score parfait.

Concernant les expériences sur l'évaluation comparative, elles ont montré que l'architecture développé dans cette recherche est supérieure à d'autres systèmes d'EI de l'état de l'art sur le corpus plus structuré Seminars, et qu'elle atteint des scores comparables sur le corpus semi-structuré Jobs. En plus, les extracteurs produits par le système proposé semblent atteindre plus de précision que les autres systèmes tout en ayant un rappel raisonnable. En revanche, même avec l'aide du tagage POS, le système proposé a obtenu un score inférieur par rapport aux autres systèmes comparés, notamment $(LP)^2$, sur le corpus *CFP* (en langage naturel). Dans M-TIES, l'algorithme BWI utilise un formalisme de règles plus expressif grâce à l'utilisation de *wildcards* qui généralise mieux que l'algorithme $(LP)^2$, par exemple, sur des documents plus structurés. Pour une collection de document moins structurés, notamment en langage naturel, on peut conclure que l'avantage de l'algorithme $(LP)^2$ sur les autres réside dans l'effet combiné de l'étape d'étiquetage contextuel, de l'étape de correction et de l'utilisation plus avancée d'information TAL. Ce dernier étant le plus important d'entre eux.

## 6.2 Perspectives relatives au système d'EI proposé

Plusieurs travaux pour étendre la version actuelle du système d'EI proposé dans cette recherche peuvent être envisagés à court et moyen terme. En voici quelques uns :

**1. Extraction d'information à partir de documents en format PDF**

Le système M-TIES ne traite que des documents en format HTML/XHTML. Cependant, le format PDF devient de plus en plus utilisé sur Web comme un format standard pour une gamme de documents. Ainsi, il est envisagé d'avoir un module capable d'extraire des informations de tel type de document.

**2. Séparation des modules tokeniseur et extracteur d'attributs**

Il a été constaté que, dans M-TIES, les modules tokeniseur et d'extraction d'attributs sont fortement liés entre eux. Pour avoir plus de flexibilité et, conséquemment, rendre ce système plus adaptatif à l'apprentissage sur de nouveaux corpus (en fournissant différents types de tokeniseur et/ou extracteur d'attributs, par exemple), un travail de réingénierie sur les codes source de M-TIES devrait être réalisé.

**3. Ajoutement d'un module de Normalisation**

Il est très fréquent d'avoir certains type d'information tels que : adresses de courrier électronique, URL de home page, dates, numéros de téléphone, pour en citer quelques uns, présents dans des pages d'intérêt. Il est fort probable qu'une étape préalable à celle de la tokenisation pourrait être très utile, puisque l'on pourrait obtenir une représentation plus homogène de tokens appartenant à ces catégories d'informations qui sont assez courantes. Par conséquent, cela pourrait augmenter les performances de l'algorithme BWI.

**4. Enrichissement de l'étape d'extraction d'attributs**

Le système M-TIES repose sur une représentation de documents par séquence de tokens. Ainsi, pour identifier chaque token, il fait l'usage d'un ensemble de treize attributs. Un enrichissement de la représentation d'un document pourrait être fait en rajoutant d'autres attributs pour aider le système lors de l'apprentissage de règles d'extraction.

**5. Expérimentation avec d'autres algorithmes d'apprentissage**

Dans le domaine de l'apprentissage automatique, il existe de nombreux algorithmes supervisés, notamment SVM (*Support Vector Machines* et C4.5 (arbre de décision), etc. Ces derniers devraient pouvoir être utilisés comme des modules d'apprentissage automatique indépendants dans M-TIES.

**6. Evaluer le processus de filtrage *Instance Filtering* (SIE)**

Le système SIE (*Simple Information Extraction*) utilise un algorithme de classification supervisée SVM pour classifier les tokens, après un traitement de *filtrage d'attributs*. Ce filtrage est réalisé par l'algorithme *Instance Filtering* [Gliozzo et al., 2005] qui indique, avec une valeur vraie ou fausse, si le token doit être considéré par l'algorithme de classification. Il a été démontré que cette approche donne de bons résultats en diminuant la quantité de tokens à tenir en considération par l'algorithme d'apprentissage lors de la génération des hypothèses et, ce qui a été le plus important, que ce filtrage a très peu d'influence sur les résultats en les comparant avec les approches qui utilisent toute l'intégralité de tokens d'un corpus.

**7. Améliorer l'étape de tokenisation**

Afin d'améliorer la tokenisation, il serait intéressant d'ajouter au système un module tokeniseur spécialisé dans les sous-tâches de lemmatisation, NER et *chunking* en anglais et français. Dans l'EVALITA 2007, une nouvelle initiative consacrée à l'évaluation des outils de TAL, un l'outil appelé TextPro [Pianta et al., 2008] a obtenu le deuxième classement comme outil plus performant dans quasiment toutes les sous-tâches de TAL (en anglais et italien) évoqué ci-dessus. Pour le français, l'outil TreeTagger [Schmid, 1994] est un bon candidat en tant qu'étiqueteur POS et lemmatiseur. On pourrait envisager ainsi l'intégration de ces modules au M-TIES afin d'avoir d'autres expérimentations sur des *corpora* en langage naturel.

**8. Persistance de règles XML dans une base de données**

Afin de faire la mise en application des règles extraites (sous format XML actuellement) sur des documents d'entrée, il pourrait être développé un module de post-traitement pour sauvegarder les règles générées en utilisant une base de données, par exemple. En plus, une étude postérieure plus minutieuse pourrait être faite pour définir d'autres façons de représenter ces règles en utilisant un autre langage ou formalisme de représentation.

**9. Modélisation d'une base de données pour les informations extraites**

De façon similaire à ce qu'on vient de mentionner, on envisage aussi un module de post-traitement pour stocker les informations extraites qui sont générées par le système en plusieurs fichiers XML. En fait, M-TIES crée séparément un fichier XML pour chaque slot du schéma d'extraction. Ainsi, on peut réorganiser les règles extraites, en les regroupant selon une structure dictée par le schéma d'extraction en question. Un possible choix serait de bien modéliser une base de données pour atteindre cet objectif.

**10. Annotation automatique guidée par une ontologie de domaine**

Il a été proposé dans la section 4.3.1 l'usage d'un système d'annotation semi-automatique pour rendre moins fastidieuse l'annotation de nouveaux *corpora*. En fait, la version originale de MnN (de 2004) avait un module très important qui guidait tout le processus de suggestions de règles pour l'annotation assistée de documents (mode interactif) ou même pour les règles d'annotation sans intervention de l'utilisateur. Ce scenario plus avantageux du point de vue de l'utilisateur n'existe plus car le module extracteur Amilcare, une mise en œuvre de l'algorithme $LP^2$, n'est plus disponible à cause des droits réservés de son créateur.

Ainsi, nous pourrions essayer de rajouter au MnM un algorithme d'apprentissage supervisé (SVM ou C4.5), par exemple, pour remplacer ce module manquant.

## 6.3 Perspectives relatives à l'architecture MasterWeb/AGATHE

En outre les perspectives présentées dans la section précédente, un autre travail envisagé dans cette recherche repose sur l'hypothèse d'utiliser le potentiel d'induction d'extracteurs, avec tous les avantages déjà listés, dans le contexte d'une extraction d'information intégrée - plus précise et fine - afin d'augmenter la performance du sous-système d'extraction de l'architecture MasterWeb/AGATHE (section 1.1).

Dans les systèmes MasterWeb/AGATHE, les ontologies peuvent réunir et combiner, sous un même environnement, les trois types de connaissances nécessaires à l'extraction d'information: (1) la connaissance destinée à l'identification des structures syntaxiques et sémantiques du texte (avec l'aide du tagage POS); (2) la connaissance pour identifier les différents formats de textes traités sans utiliser de techniques TAL - comme les wrappers, par exemple; et enfin (3) la connaissance pour réaliser les inférences avec un engagement ontologique, c'est-à-dire, en employant les faits connus sur les entités extraites.

Suite à cette recherche, il est envisagé de combiner la tâche symbolique du système MasterWeb/AGATHE qui réalise actuellement une classification des pages Web à base d'ontologies, avec une tâche d'extraction d'information adaptative, permettant d'extraire de l'information sur ces pages Web classées, ceci par l'usage de techniques d'apprentissage artificiel (machine learning) utilisées dans M-TIES avec l'algorithme BWI.

Cette combinaison de techniques symboliques et de techniques d'apprentissage artificiel devrait permettre d'une part une amélioration significative de la performance de ces architectures en les dotant de techniques d'induction automatique d'extracteurs d'information et de techniques TAL ; et, d'autre part, de faciliter la mise en œuvre de ces architectures sur de nouveaux domaines du Web, en évitant le développement fastidieux de bases de règles symboliques d'extraction d'information.

# REFERENCES


[Adda et al., 1999] Adda G. M., Paroubek J., Leconte J. Metrique et premier résultats de l'évalution GRACE des étiqueteurs morphosyntaxiques pour le français. Amsili, P. (ed.), Actes de TALN'99, pages 15-24, 1999.

[Abiteboul, 1997] Abiteboul S. Querying semistructured data. In ICDT, pages 1-18, 1997.

[Aldea et al., 2003] Aldea A., Bañares-Alcántara R., Bocio J., Gramajo, D., Isern D., Kokssis A., Jiménez L. Moreno A., Riaño D. An Ontology-Based Knowledge Management Plataform. In IJCAI. IIWEb-03, 2003.

[AKT, 2009] MnmMnM, Ontology Driven Semi-Automatic and Automatic Support for Semantic Web. Mnm Developer Guide.
Disponible à: http://projects.kmi.open.ac.uk/akt/MnM/MnM_Developer_Guide.html (dernier accès en juin 2009).

[Apache, 2009] JFex - java Feature Extractor, User Guide. Disponible à: http://tcc.itc.it/research/textec/tools-resources/jfex/quickstart.html (dernier accès en juin 2009).

[Arasu, 2003] Arasu A., Garcia-Molina H. Extracting structured data from web pages. In Proceedings of internation conference on Management of data, pp. 33--348, 2003.

[Baumgartner et al., 2001] Baumgartner R., Flesca S., Gottlob G. Visual web information extraction with Lixto. In 28th International Conference on VLDB, pp. 119-128, 2001.

[Bray et al., 2008] Bray T., Paoli J., Sperberg-McQueen C., Maler E., Yergeau F. Extensible Markup Language (XML) 1.0 (Fifth Edition). In W3C Recommendation, November, 2008.

[Brill, 1992] Brill E. A simple rule-based part of speech tagger. In Proceedings Of the 3th conference on Applied NLP, pp. 152-155. Association for Computational Linguistics, 1992.

[Cabral, 2004] Cabral D. M. Um framework para extração de informações: uma abordagem baseada em XML. Dissertação de Mestrado. UFPE, CIN, Recife, 2005.

[Califf, 1998] Califf, M. E. Relational Learning Techniques for Natural Language Information Extraction. Ph.D. Dissertation, University of Texas at Austin, 1998.

[Califf & Mooney, 1999] Califf M. E, Mooney R. J. Relational learning of pattern-match rules for information extraction. In Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99), pp. 328-334, 1999.

[Callif et al., 2004] Califf M. E., Ciravegna F., Freitag D., Giuliano C., Kushmerick N., Lavelli A. Romano. A critical survey of the methodology for IE evaluation. In Proceedings of LREC, 2004.

[Chang & Lui, 2001] Chang C. H., Lui S. C. IEPAD: Information extraction based on pattern discovery. Proceedings of the Tenth International Conference on World Wide Web (WWW), Hong-Kong, pp. 223-231, 2001.

[Chang & Kuo, 2004] Chang C. H., Kuo S. C. OLERA: A semi-supervised approach for Web data extraction with visual support. IEEE Intelligent Systems, 19(6):56-64, 2004.

[Chang et al., 2006] Chang C-H., Kayed M., Girgis M. R., Shaalan K. F. A Survey of Web Information Extraction Systems. IEEE Trans. Knowl. Data Eng. 18(10): 1411-1428, 2006.

[Ciravegna, 2001] Ciravegna, F. $(LP)^2$. An adaptive algorithm for information extraction from web-related texts. In Proceedings of the IJCAI-2001. Workshop on Adaptive Text Extraction and Mining held in conjunction with the 17th International Joint Conference on Artificial Intelligence (2001), Seattle, 2001.

[Ciravegna & Lavelli, 2001] Ciravegna, F., Lavelli, A. LearningPinocchio: Adaptive information extraction for real world applications. In Proceedings of 3rd Romand Workshop. Frascati, Italy, 2001.

[Ciravegna, 2003a] Ciravegna, F. Designing adaptive information extraction for the semantic web in Amilcare. In S. Handschuh and S. Staab, editors, Annotation for the Semantic Web, Frontiers in Artificial Intelligence and Applications. IOS Press, Amsterdam, 2003.

[Ciravegna, 2003b] $(LP)^2$, Rule Induction for Information Extraction Using Linguistic Constraints. Technical Report CS-03-07, Departament of Computer Science, Univ. of Sheffield, Sheffield, September 2003.

[Cohen et al., 2003] Cohen W., Hust M., Jensen L. Web Document Analysis: Challenge and Opportunities. World Scientific, 2003.

[Cornuéjols & Miclet, 2002] Cornuéjols A., Miclet L. Apprentissage Artificielle: concepts et algorithms. Eyrolles, 2002.

[Cowie & Lehnet, 1996] Cowie J., and Lehnet W. Information Extraction, Communication of ACM vol.39, 1996.

[Cortes & Vapnik, 1995] Cortes C., Vapnik V. Support-Vector Networks. Machine Learning, 20(3):273–297, 1995.

[Crescenzi et al., 2001] Crescenzi V., Mecca G., Merialdo P. Roadrunner: Towards automatic data extraction from large web sites. In Proceedings of 27th International Conference on VLDB, 2001.

[Crespo et al., 1994] Crespo A., Jannink J., Neuhold E., Rys M., Studer R. A survey of semi-automatic extraction and transformation. Techinical report, 1994.

[Cunningham et al., 2002.a] Cunningham H., Maynard D., Tablan V., Ursu C., Bontcheva K. Developing language processing components with GATE, www.gate.ac.uk, 2002.

[Cunningham et al., 2002.b] Cunningham H., Maynard D., Tablan V., Ursu C., Bontcheva , K. The GATE User Guide, 2002. Disponible à: http://gate.ac.uk (dernier accès en juin 2009).

[Douthat, 1998] Douthat, A. The message understanding conference scoring software user's manual. In Proceedings of the 7th Message Understanding Conference (MUC-7), 1998.

[Eikvil, 1999] Eikvil, L. Information Extraction form the World Wide Web: a Survey. In Technical Report 945. Norweigian Computing Center, 1999.

[Espinasse et al., 2007] Espinasse B., Fournier S., Freitas F. AGATHE : une architecture génerique à base d'agents et d'ontologies pour la collecte d'information sur domaines restreints du Web. CORIA 2007. Quatrième conférence francophone en Recherche d'Information et Applications, 2007.

[Finn & Kushmerick, 2004] Finn A., Kushmerick N. Multi-Level boundary classification for information extraction. In Proceeding of the European Conference on Machine Learning, Pisa, 2004.

[Florescu et al., 1998] Florescu D., Levy A., Mendelzon A. O. Database techniques for the World Wide Web: A survey, SIGMOD Rec., 1998.

[Forgy 82] Forgy, C. L. Rete: A Fast Algorithm for the Many Pattern/ Many Object Pattern Match Problem, Artificial Intelligence 19(1982), 17-37, USA, 1982.

[Fred Freitas et al., 2008] Freitas F., Cabral L., Lima R., Espinasse B., Palmeira E., Fournier S., Guilherme Bittencourt. From MASTER-Web to AGATHE: the evolution of architecture for manipulating information over the Web using ontologies. RECIIS Journal, vol. 2, no. 1, pp. 73-84, 2008.

[Freitag, 1997] Freitag D. Using grammatical inference to improve precision in information extraction. In ICML-97 Workshop on Automation Induction, Grammatical Inference, and Language Acquisition, Morgan Kaufmann, 1997.

[Freitag & McCallum, 1999] Freitag D, McCallum A. K. Information Extraction with HMMs and shrinkage. In Proc. Of the AAAI-99. Workshop on Machine Learning for Information Extraction, 1999.

[Freitag & Kushemerick, 2000] Freitag D., Kushmerick N. Boosted Wrapper Induction. In Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000), 2000.

[Freund & Schapire, 1990] Freund Y., Schapire R. E. A decision-theoric generalization of on-line learning and an application to boosting. Proc. Of the 2th European Conf. on Computational Learning Theory, Rochester, NY, ACM Press, pp. 202-216, 1990.

[Freund & Schapire, 1996] Freund Y., Schapire R. E. Experiments with a new boosting algorithm. In International Conference on Machine Learning, pp. 148–156, 1996.

[Freund & Schapire, 1997] Freund Y., Schapire R. E. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 55(10):119-139, August, 1997.

[Freund & Schapire, 1999] Freund, Y., Schapire R. E. A short introduction to boosting**.** *Journal of Japanese Society for Artificial Intelligence*, 14(5):771-780, September, 1999.

[Friedmann-Hill 2000] Friedmann-Hill, E. Jess, The Java Expert System Shell, 2000. Disponible à: http://herzberg.ca.sandia.gov/Jess (dernier accès en juin/2009).

[Gilleron et al., 2006] Gilleron R., Marty P., Tommasi M., Torre F. Extraction de relations from semi-structured data. In Revue RNTI - Actes de EGC'06, pages 415-420, 2006.

[Girardi, 2007] Girardi, C. HtmlCleaner: Extracting Relevant Text from Web Pages. In Proceedings of WAC3 2007 - 3rd Web as Corpus Workshop. Louvain-la-Neuve, Belgium, September 15-16, 2007.

[Giuliano et al., 2006] Giuliano C., Lavelli A., Romano L. Simple Information Extraction (SIE): A Portable and Effective IE System. In Proceedings of the EACL-06 Workshop on Adaptive Text Extraction and Mining (ATEM-2006) , Trento, Italy, 2006..

[Gliozzo et al., 2005] Gliozzo A. M., Giuliano C., Rinaldi R. Instance pruning by filtering uninformative words: an Information Extraction case study. In Proceedings of the Sixth International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2005), Mexico City, Mexico, 13-19, February, 2005.

[Hirschman, 1998] Hirschman L. The evolution of evaluation: Lessons from the Message Understanding Conferences. Computer Speech and Language, 12, pp. 281-305, 1998.

[Hsu, 1998] Hsu C. N., Dung M. Generating finite-state transducers for semi-structured data extraction from the web. Journal of Information Systems, 23(8): 521-538, 1998.

[Ireson & Ciravegna, 2005] Ireson, N., F. Ciravegna. Pascal Challenge: The Evaluation of Machine Learning for Information Extraction. Machine Learning for the Semantic Web Dagstuhl Seminar, Dagstuhl, DE, 2005.

[Ireson et al., 2005] Ireson N., Ciravegna F., Califf M. E., Freitag D., Kushmerick N., Lavelli A. Evaluating machine learning for information extraction. In Proceedings of the 22nd international conference on Machine learning table of contents, Vol. 119, pp. 345 - 352 , Bonn Germany, 2005.

[ITC-IRST, 2004] TIES. Trainable Information Extraction System. Dot.Kom project, 2004. Disponible à : http://tcc.itc.it/research/textec/tools-resources/ties.html (dernier accès en juin 2009).

[Kauchak et al., 2002] Kauchak D., Smarr J., Elkan C. Sources of Success for Information Extraction Methods, Technical Report CS2002-0696. Department of Computer Science and Engineering, University of California, San Diego, January, 2002.

[Kohavi, 1995] Kohavi R. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. IJCAI, pp. 1137-1145, 1995.

[Kosala & Blockeel] Kosala R., Blockeel H.Instance-based wrapper induction, 2000.

[Kushmerick, 1997] Kushmerick N. Wrapper Induction for Information Extraction. PhD. Thesis, University of Washington, 1997.

[Kushmerick, 2000] Kushmerick N. Wrapper induction: Efficency and expressiveness. Artificial Intelligence, 118(1-2):15-68, 2000.

[Kushmerick & Thomas, 2003] Kushmerick, N., Thomas B. Adaptive Information Extraction: Core Technologies for Information Agents, 2003.

[Laporte, 2000] Laporte M. Mots et niveau lexical. Pierrel, J-M (ed.), Ingénerie des langues, Informatique et systemes d'information, ch. 1, pages 25-50, Hermes Science, Paris, 2000.

[Lavelli A. et al., 2004] Lavelli A., Califf M. E, Ciravegna F., Freitag D., Giuliano C., Kushmerick N., Romano L. IE Evaluation: Criticisms and Recommendations. In AAAI-2004 Workshop on Adaptive Text Extraction and Mining, 2004.

[Li et al., 2004a] Li Y., Bontcheva K., Dowman M.; Roberts I., Cunningham, H. D2.1.1 Ontology Based Information Extraction (OBIE) v.1., SEKT deliverable, University of Sheffield, 2004.

[Li et al., 2004b] Li Y., Bontcheva K., Cunningham H.: SVM Based Learning System for Information Extraction. Deterministic and Statistical Methods in Machine Learning 2004: 319-339, 2004.

[Li et al., 2003] Li, Y., Shawe-Taylor, J.: The SVM with uneven margins and Chinese document categorization. In Proceedings of The 17th Pacific Asia Conference on Language, Information and Computation (PACLIC17), pages 216–227, Singapore, Oct. 2003.

[Liu et al., 2000] Liu L., Pu C., Han W. XWRAP: An XML-enabled wrapper construction system for web information sources. In ICDE, pp. 611-621, 2000.

[Marty, 2007] Marty Patrick. Induction d'extraction n-aire pour les documents semi-structurés. Thèse Doctorat. Université Charles de Gaulle, Lille 3, 2007.

[Marty & Torre, 2003] Marty P., Torre F. Classer pour extraire : représentation et méthodes. Technical Report Grappa report 0103, GRAPPA, 2003.

[Marty & Torre, 2004] Marty P., Torre F. Codages et connaissances en extraction d'information. In M. Liquière and M. Sebban, editors. Actes de la Sixième Conférence d'Apprentissage (CAp 2004), Montpellier, pp. 207-222, juin 2004.

[Mason & Tufis, 1998] Mason O., Tufis D. Tagging Romanian Texts: a Case Study for QTAG, a Language Independent Probabilistic Tagger. In Proceddings of First International Conference on Language Resources and Evaluation (LREC), Granada (Spain), 28-30 May, pp.589-596, 1998.

[Mitchell, 1997] Michell T. Machine Learning. McGraw-Hill, 1997.

[MnN, 2008] Ontology driven semi-automatic and automatic support for semantic web, décembre, 2008. Disponible à : http://projects.kmi.open.ac.uk/akt/mnm/ (dernier accès en juin 2009).

[Muslea et al, 1998] Muslea I., Minton S., Knoblock C. STALKER: Learning extraction rules for semistructured, web-based information sources. In AAAI Workshop on AI and Information Integration. pp.74-81, 1998.

[Muggleton, 1994] Muggleton S., Raedt L. D. Inductive logic programming: Theory and methods, J. Log. Programming, 19/20:629-679, 1994.

[Muslea et al., 2001] Muslea I., Minton S., Knoblock C. Hierarchical wrapper induction for semistructured information sources. Automomous Agents and Multi-Agent Systems, 4(1/2): 93-114, 2001.

[Paroubek & Rajman, 2000] Paroubek, P., Rajman, M. Etiquetage morpho-syntaxique. Pierrel, J-M (ed.), Ingénerie des langues, Informatique et systemes d'information, ch. 5, pages 131-150. Hermès Science, Paris, 2000.

[Pazienza, 1997] Pazienza M. T. Information Extraction: Towards scalable, adaptable systems. In Lecture Notes in Artificial Intelligence, 1997.

[Peshkin, 2003] Peshkin, L., and Pfeffer, A.Bayesian information extraction network. In Proceedings of 18th International Joint Conference on Artificial Intelligence (IJCAI 2003), 2003.

[Pianta et al., 2008] Pianta E., Girardi C., Zanoli R. In Proceedings of LREC, 6th edition of the Language Resources and Evaluation Conference, 28-30 May , Marrakech, Morocco, 2008.

[QTag, 2008] QTag, a portable POS tagger, 2008. Disponible à : http://morphix-nlp.berlios.de/manual/node17.html (dernier accès en juin 2009).

[Russel & Norwig, 2003] Russell S., Norwig P. Artificial Intelligence: A Modern Approach.Pearson Education, 2003.

[Sahuguet & Azavant, 2001] Sahuguet A., Azavant F. Building intelligent web applications using lightweight wrappers. Data Knowledge Eng. 36(3): 283-316, 2001.

[Schmid, 1994] Schmid H. Probabilistic Part-of-Speech Tagging Using Decision Trees. In International Conference on New Methods in Language Processing, pp. 44-49, Manchester,UK, 1994.

[SEKT, 2006] SEKT project, Semantically-Enabled Knowledge Technologies. 2006. Disponible à : http://www.sekt-project.com/ (dernier accès en juin 2009).

[Seymore et al., 1999] Seymore K., McCallum A., Rosenfeld. Learning hiddem Markov Model strutucture for Information Extraction. In AAAI 99 Workshop on Machine Learning for Information Extraction.

[Siefkes & Siniakov, 2005] Siefkes C., Siniakov, P. An overview and classification of adaptive approaches to information extraction. Journal on Data Semantics IV. Berlin, Germany:Springer, 2005.

[Soderland, 1999] Soderland S. Learning information extraction rules for semi-structured and free text. Machine Learning, 34(1-3):233-272, 1999.

[Tang et al., 2007] Tang J., Hong M., Zhang D., Liang B., Li, J. Information Extraction: Methodologies and Applications. DCS-Tsinghua University, 2007.

[Thomas, 2005] Thomas B. Machine Learning of Information Extraction Procedures - An ILP Approach, PhD. Thesis, Universität Klobentz-Landau, 2005.

[Vargas-Vera, 2002] Vargas-vera M., Motta E., Domingue J., Lanzoni M., Ciravegna F. MnM: Ontology Driven Semi-Automatic and Automatic Support for Semantic Markup. Springer Verlag, pp.379-391, 2002.

[Yang & Liu, 1999] Yang Y., Liu X. A Re-Examination of Text Categorization Methods. In Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 42–49, 1999.

[Wang & Locovsky, 2002] Wang J., Lochovsky F. H. Wrapper induction based on nested pattern discovery. Technical Report HKUST-CS-27-02. Department of Computer Science, Hong Kong, University of Science and Technology, 2002.

[Zhai & Liu, 2005] Zhai Y., Liu B. Extracting web data usning instance-based learning. In Proceedings of Web Information Systems Engineering WISE, pages 318-331, 2005.

# **Appendice A** - Légendes d'étiquettes POS du QTAG (en anglais)

| Tag | Meaning | Tag | Meaning |
|---|---|---|---|
| ??? | no tag assigned | NNS | noun, plural |
| "" | simple double quote | NP | proper noun, singular |
| # | pound sign | NPS | proper noun, plural |
| $ | dollar sign | PDT | predeterminer |
| ' | right single quote | POS | possessive ending |
| ` | left single quote | PP | personal pronoun |
| ( | left parenthesis (round, square, curly or angle) | PP$ | possessive pronoun |
| ) | right parenthesis (round, square, curly or angle) | RB | adverb |
| , | comma | RBR | adverb, comparative |
| . | sentence-final punctuation | RBS | adverb, superlative |
| : | mid-sentence punctuation | RP | particle |
| BE | be | SYM | symbol |
| BED | were | TO | infinitive marker 'to' |
| BEDZ | was | UH | interjection |
| BEG | being | VB | verb, base form |
| BEM | am | VBD | verb, past tense |
| BEN | been | VBG | verb, gerund or present participle |
| BER | are | VBN | verb, past participle |
| BEZ | is | VBP | verb, non-3rd person singular present |
| CC | coordinating conjunction | VBZ | verb, 3rd person singular present |
| CD | cardinal number | WDT | wh-determiner |
| DO | do | WP | wh-pronoun |
| DOD | did | WP$ | possessive wh-pronoun |
| DOG | doing | WRB | wh-adverb |
| DON | done | XNOT | not and n't |
| DOZ | does | | |
| DT | determiner | | |
| EX | existential 'there' | | |
| FW | foreign word | | |
| HV | have | | |
| HVD | had (past tense) | | |
| HVG | having | | |
| HVN | had (past participle) | | |
| HVZ | has | | |
| IN | preposition or subordinating conjunction | | |
| JJ | adjective | | |
| JJR | adjective, comparative | | |
| JJS | adjective, superlative | | |
| LS | list item marker | | |
| MD | modal | | |
| NN | noun, singular or mass | | |