Using Grammatical Inference For Structure Induction

Alexandre S. Saidi
Ecole Centrale de Lyon - Mathematics and Computer Science Department
B.P. 163. 69134 Ecully - France , alexandre.saidi@liris.cnrs.fr

Abstract

Given the huge quantity of the current available textual information, Text Mining process tackles the task of searching useful knowledge in a natural language document.

When dealing with a free-format textual corpus (e.g. a Job announcement) where the linguistic rules are not respected, the time consuming morpho-syntactic analysis is not of a great help. However, text mining techniques process may exploit linguistic sub-structures in the text.

In this paper, we present an applications of Grammatical Inference (GI) in a machine learning system applied to a text corpus. We specify and use the process of the Grammatical Inference as an instance of the Constraint Satisfaction Problem that instantiates automata in a (language inclusion) lattice.

1 Introduction

Textual databases constitute the major part of the current available information. Significant research work concentrates on the Information Extraction (IE) from these databases.

Given a textual corpus, the information extraction process applied by the techniques of *Text Mining* (e.g. [24], [27], [26], [13]) consists of the search for no-explicit informations in such corpora. As an example, Text Mining can extract significant information from Marine catastrphes bulletins like the prior event sequence of such disasters.

In a basic approach, IE task would be tedious if no *a priori* structural information is available about the text. On the other hand, given the cost of a syntactical analysis, an IE process based on a whole morpho-syntactic analysis of documents would not often be realistic. When dealing with free-format texts, such analysis would not be of a great interest in a text mining process usually based on key patterns.

In the case of free format texts, the rules of linguistic grammars are seldom respected. These texts rather contain few words without using entities such as determinant, verb and other punctuation.

In the current work, we are interested in the structures of sub-languages in free-format texts. For example, suppose an advertisement of an exposition on *Egypt* that will take place in *Paris*. The knowledge of the structure of the sub-language representing the *address* (where the exposition takes *Place*) may avoid concluding too quickly (and wrongly) on the place of the exposition upon the simple presence of *Egypt* city name.

Text Mining research field has been focused on since 1991 through MUC programs. However, it is still domain specific and time-consuming to build a new system or to adapt an existing one to a new domain. Although symbolic and statistical methods have been applied in some IE systems (e.g. [21], [28]), not a lot have combined Grammatical Inference with (naive) statistical information.

Techniques of Grammatical Inference (GI) ([15], [16], [17]) promise to be useful in this field. They carry the process of *Text Mining* to capitalize the (partial) morpho-syntactic structure of patterns (or of sub languages) with a few amount of information on the contents structure. These techniques attempt to induce the structures of a source data (flow of signs) by a set of production rules of a regular grammar. The induced grammar being an element of a (language-inclusion) lattice, the text mining is concerned by an informed search (seen as a *generalization*) within this lattice carrying required information and semantics.

This paper describes a research work on the design and implementation of GI process that was successfully applied first in a Pattern Recognition project on documents like summaries, dictionaries, scientific reports and so on. Here, whenever the *linguistic* structure of these documents are extracted, Textual DataMining technics ([24], [27], [25], [26], [20], [23]) are applied to such documents and extract valuable *knowledge* from the data.

For example, a given report document (among the corpus) can be (logically) structured by a production rule like:

report ← *abstract, outline, chapter, sub-chapter, chapter, references*.

Having one rule per example, the objective is to generalize these rules in a GI process and to propose an automaton that describes the underlying language.

A second application of this process was on a seminar announcement corpus. A seminar announcement may have the following structure (one of the possible formats):

seminar ← *heading, subject, speaker, date, hour, address, organizer*

In this second experiment, the aim is first to learn how to recongnize slot values (and their structures), and then to capture slot fillers from new announcements.

In this paper, we focus on the extraction of the structure and the content of the above announcement $corpus$ of documents using the Grammatical Inference. We apply the process of Grammatical Inference to a set of regular production rules. As in the above examples, each rule (one for each element of the sample set) represents the (logical) structure of the sample[1]. Negative descriptions (and samples) can be provided in order to denote those structures that must be rejected. The inference engine then produces a representative regular grammar that will recognize documents in their respective context.

In the following, some background about the Regular Inference is reported.

2 Grammatical Induction

The problem of grammatical inference can be considered in a Constraint Satisfaction Problem framework (see e.g. [6]). Although some work (e.g. [10]) tackled this problem as an instance of graph coloring, the proposed approach gave an interesting but a quite general idea of the question.

In [8], Gold showed that any recursively enumerable class of language is identifiable using a *complete* representation with the positive and the negative data. Hence, the class of regular languages cannot be correctly identified from only the positive examples. Although the usual case in document handling is to learn from only positive examples (given in the set I_+), the induced grammar can be drastically refined by some negative examples (the set I_-) and avoid *over generalization*[2].

It is known that any algorithm that would construct a deterministic finite automaton (DFA) with a minimum number of states compatible with all the data already processed can identify any regular language in the limit ([7]).

To achieve that, we developed an original and complete algebraic framework for the Grammatical Inference. In this framework, we define a relation over the (language inclusion) lattice of automata represented by the set of all samples $I = (I_+ \cup I_-)$ that leads to the construction of partitions over that search space. To realize that, an initial algebra A_{G_I} is assigned to a regular grammar G_I associated to the sample set I. Then we focus on the definition of a quotient algebra $A_{G_I/R}$ of A_{G_I} that leads to a uniquely defined isomorphism from $A_{G_I/R}$ to the language of the induced automaton A. This automaton is supposed to govern and generalize the language structuring the sample set.

Within this algebraic framework (describing why the logical description is processed in that way), we discuss a general Constraint Satisfaction specification that characterizes the search space of the GI problem. Then, we define a set of constraints that outlines the quotient algebra above and constructs the final induced DFA (the automaton A).

2.1 The Regular Inference

The Inductive Inference paradigm is the basis of the automatic learning problem (see also [19]). In the Syntactical Pattern Recognition (see e.g. [18]), many grammatical inference algorithms are proposed that are used in the learning step of the pattern recognition tasks ([15], [16], [17]). As mentioned above and in order to correctly identify regular languages, positive (I_+) and negative (I_-) examples are to be provided to represent the language to be learned.

Example : partially from a seminar announcement textual database (see the section 5 for an announcement example), we may have some announcement message rules in the sets I_+=$\{r_1, r_2, r_3, r_4, ...\}$, I_-=$\{r_5, r_6, ...\}$ below:

$r_1 : Message \leftarrow$ "$seminar$", Det, $Organization, Name.$
$r_2 : Message \leftarrow$ "$seminar$", $Organization, Theme.$
$r_3 : Message \leftarrow$ "$seminar$", $Pro,\ City,$ " : ", $Theme.$
$r_4 : Message \leftarrow$ "$seminar$", $Name, Theme.$
$r_5 : Message \leftarrow Det,$ " : ", $Theme.$
$r_6 : Message \leftarrow$ "$seminar$", $Organization,$ $Organization.$

Here, each valid announcement begins with the word *seminar*. The rule r_5 denotes that a message can not begin with a *Determinant* (Det) while r_6 denotes that a message with two successives *Organization*s must be rejected. Note that saying naively that *Organization* will always follow *seminar* word is wrong.

In the next section, an algebraic specification of the GI problem is stated. Then, in the sections 4 and 4.1, some practical issues, the implementation of the proposed CSP framework together with some examples are reported. Then, some relationships with other works in this field are recalled in the section 9.

3 Algebraic View of the GI

In the algebraic specification below, and relative to the sample set ($I = I_+ \cup I_-$), the properties of the partitions

[1] Various sectionslike the *address* section of a seminar announcement are handled in turn.

[2] Opposite to *overfitting*, the extreme case of the over generalization for an alphabet Σ is the language Σ*.

over the terms of A_{G_I}-algebra associated to the grammar G_I are depicted. Then we formally characterize a relation from these partitions to $L(A)$, the language of the final induced automaton. This is done by the definition of a set of constraints defining a congruence relation R over the terms of A_{G_I}. The latter produces a quotient-algebra $A_{G_I/R}$ whose terms are isomorphic to those of $L(A)$.

Quotients of the A_{G_I}-algebra give a (language inclusion) lattice. Here, our *main aim* in the Grammatical (regular) Inference is to characterize this lattice and to guide the search in it.

The Grammatical Inference problem can be specified by using the relation between an initial many sorted algebra and context-free grammars ([11])[3]. To construct the algebra associated to a context-free grammar G, each non terminal of G is assigned to a class of derivation tree. Consequently, the non terminals of G are sorts of a many sorted algebra whose operations are defined by the production rules of G. The derivation tree (and the language) of any non terminal X denotes the carriers of the sort X of the algebra.

Let $G = (N, T, P, S)$ be a context-free grammar and L_G be its language with N=non terminals, T : the terminals, P : Production rules and S : the start symbol . Let associate to G the A_G-algebra whose signature is $((N \cup T), Op)$ where Op is the set of names given to the productions in P. The terms of this algebra are (possibly partial) derivation trees starting from any non terminal of G.

An A_G-algebra is *initial* in a category C based on the same signature if for all algebra B of C, there exist a unique homomorphism $f : A_G \rightarrow B$.

Let's now consider the sample set $I = I_+ \cup I_-$ (with $I_+ \cap I_- = \emptyset$), the grammar G_I from the set I , the A_{G_I}-algebra associated to G_I and the language $L(A)$ (A is the final induced DFA). We are interested in f such that $f : A_{G_I} \rightarrow L(A)$. Consider the set of finite automata associated to elements of I (one automaton per element of I) and let ***Tree(I)*** denote the tree of all these automata.

We define the automaton $G_I = (Q \cup \{S\}, \Sigma, P_\delta, \text{S,F})$ associated to $Tree(I)$ where Q is the set of all states in $Tree(I)$, Σ is the set of terminals in I and P_δ is the set of the names given to the transitions in *Tree(I)*. The start symbole S is such that $S \rightarrow p_{01} | p_{02} | ...$ where $p_{ij} \in P_\delta$, j is the rank of the transition in the ith automaton associated to each element of I. F is the set of final states in Tree(I). G_I associated to $Tree(I)$ is possibly a no deterministic but ϵ-free (circuit-free) automaton.

If L_{I_+} is the language of the positive samples (resp. L_{I_-} for the negative ones) generated by the final induced automaton A (that accepts only L_{I_+}), then, for any partition of Q containing equivalent states (cf. the section 4), $I_+ \subseteq L_{I_+}$ and $I_- \subseteq L_{I_-}$. We have $L_{I_+} \subseteq \Sigma^*$ - L_{I_-} and $L_{I_-} \cap L_{I_+} = \emptyset$.

[3]This relation is easily extended to the regular grammars.

Let consider R a congruence relation, a partition $Tree(I)_{/R}$ from $Tree(I)$ and its regular grammar G_R, A_{G_I} and $A_{G_I/R}$ are the algebra assigned to G_I and G_R. In the following section, we will define a homomorphism $homo_R$ from A_{G_I} to $A_{G_I/R}$ that formally defines the equivalence classes of A_{G_I}-algebra. Then, we will state a constraint satisfaction specification of the (language inclusion) lattice induced by *homo*$_R$ and propose a Constraint Logic Program (CLP [6]) that will search, under some constraints, for a (not necessarily minimal[4] canonic) DFA in that lattice.

3.1 The Quotient Algebra

Let $A_{G_I} = ((Q \cup \Sigma), Op)$ be an algebra associated to the (regular) grammar of the sample set I. Terms of A_{G_I} are derivation trees (let note them by $\hat{a}$ or $\hat{b}$) of the form *ri(rj, rm(..., rk(rn)...)* and of some sort q $\in$ Q. Let R a congruence relation on A_{G_I}. Op is the set of names (like *ri*) of rules of G_I of the form *(q', $\alpha \rightarrow$ q)* or *($\alpha \rightarrow$ q)*, $\alpha \in \Sigma$, q,q' $\in$ Q. The quotient algebra induced by R is defined by $A_{G_I/R}$ = ($\overline{(Q \cup \Sigma)}$, $\overline{Op}$) with :

1- $\overline{(Q \cup \Sigma)}$ = {[$\hat{a}$] | $\hat{a}$ a derivation tree whose type is q $\in$ Q} where the congruence class [$\hat{a}$] is defined by [$\hat{a}$]={$\hat{b}$ a derivation tree of type q $\in$ Q | ($\hat{a}$,$\hat{b}$) $\in$ R};

2- $\overline{Op}$=set of $\overline{ri}$ for each element of Σ, if *ri* is the name of a production rule of the form $\alpha \rightarrow$ *q* with q $\in$ Q and $\overline{ri}$: [α] $\rightarrow$ [q];

3- $\overline{Op}$= set of $\overline{ri}$: ([q'], [q"]) $\rightarrow$ [q] if *ri* is the name of a production rule *(q', q")* $\rightarrow$ *q* with q, q', q" $\in$ Q is defined by $\overline{ri}(\overline{rj},\overline{rm}(...,\overline{rk}(\overline{rn})...)$=[ri(rj, rm(..., rk(rn)...)].

A derivation tree [$\hat{a}$] in $A_{G_I/R}$ is constructed using elements congruent to $\hat{a} \in A_{G_I}$.

Although a term $\hat{a}$ of A_{G_I} is like *ri(rj, rm(rl, ..., rk(rn)...)* with $r_i, r_j, ... \in Op$, for the sake of clarity, we will rewrite $\hat{a}$ by *ri(α, rm(β,..., rk(γ)...)* when *rj* (resp. *rl*, *rn*, etc.) is the name of a production rule like $\alpha \rightarrow$ *q* (resp. *rl* : $\beta \rightarrow$ *q* and *rn* : $\gamma \rightarrow$ *q*, etc.). This is also motivated by the fact that [α] denotes the equivalence class of the constant α whenever [$\hat{a}$]=[α]. We set [α]=α for each $\alpha \in \Sigma$.

Operations of $A_{G_I/R}$ are well defined since R is reflexive, symmetric, transitive and compatible such that from [$\hat{a}$]=[$\hat{b}$] we conclude that ($\hat{a}$,$\hat{b}$) $\in$ R.

Thus,

$((A_{G_I})\ r_{i1}(\alpha 1, r_{j1}(\alpha 2, ..., r_{k1}(\alpha n)...)\ ,\ (A_{G_I})\ r_{i2}(\beta 1, r_{j2}(\beta 2, ..., r_{k2}(\beta n)...)) \in R$

implies :

[4]"not necessarily minimal" is to be considered in a grammatical point of view, w.r.t. the number of states. The induced grammar is actually minimal w.r.t. all constraints specified in the Congruence predicate (section 4)

$[(A_{G_I})\ \mathrm{r}_{i1}(\alpha 1,\ \mathrm{r}_{j1}(\ ...,\ \mathrm{r}_{k1}(\alpha \mathrm{n})...)] \equiv [(A_{G_I})\ \mathrm{r}_{i2}(\beta 1,\ \mathrm{r}_{j2}(...,\ \mathrm{r}_{k2}(\beta \mathrm{n})...)]$.
Equivalently, $(\mathrm{r}_i,\ \mathrm{r}_j) \in \mathrm{R}$ implies $\overline{r_i} \equiv \overline{r_j}$ (same as $[\mathrm{r}_i] \equiv [\mathrm{r}_j]$).

Quotient algebra are characterized by the universal property (up to an isomorphism [12]). This property is stated by the following (homomorphism) theorem applied to A_{G_I} (proofs out of the scope, avoid self reference) :

Theorem 1. *Let A_{G_I} associated to Tree(I) be the algebra and R a congruence relation on A_{G_I}. Then*

$$homo_R : A_{G_I} \rightarrow A_{G_I/R}$$

defined by $homo_R(\hat{a}) = [\hat{a}]$ for $\hat{a} \in A_{G_I}$ is a homomorphism that has the following property.

Let $f : A_{G_I} \rightarrow L(A)$ a homomorphism with the (former) congruence relation R, then there exists a unique homomorphism $\bar{f}$ such that the following diagram of mapping is commutative, i.e., $f = \bar{f} \circ homo_R$.

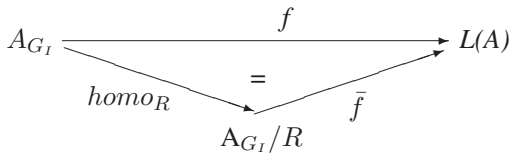


Figure 1. Commutative Mapping Diagram

It can be first showed that $homo_R$ is a homomorphism before proving the above theorem. One may note that the quotient algebra will define equivalence classes on A_{G_I}. Defining $\bar{f}$ will let us reach our goal which is to define $L(A)$ from A_{G_I}. In the next section, we define a CSP specification by the Congruence predicate that defines the congruence relation R (over the of terms A_{G_I}) and hence characterizes the search space Lat_R (see figure below) and the instantiations in it. Then we discuss the properties of I_+ and I_- with respect to L_I, L_{I_+} , L_{I_-} and $L(A)$.

In the following figure, the top element Σ^* of the lattice Lat_R represents the set of all the *words* that can be constructed over the alphabet Σ, and the button element $\emptyset$ represents the empty set. This search space contains all automata (one for each element of Σ^*) in which the final automaton $L(A)$ is searched.

4 The Congruence Predicate

Recall that R is a congruence relation over (the sorts of) A_{G_I}. Let $\hat{a}_1$, $\hat{a}_2 \in \mathrm{A}_{G_I}$ where
$\hat{a}_1 = \mathrm{r}_{i_1}(\alpha_1,\ \mathrm{r}_{i_2}(\alpha_2,\ \mathrm{r}_{i_3}(\alpha_3,\ \mathrm{r}_{i_4}(\alpha_4,...,\ \mathrm{r}_{i_{n-1}}(\alpha_{n-1},\ \mathrm{r}_{i_n}(\alpha_n)...)$
and

Figure 2. The search space Lat_R

$\hat{a}_2 = \mathrm{r}_{j_1}(\beta_1,\ \mathrm{r}_{j_2}(\beta_2,\ \mathrm{r}j_3(\beta_3,\ \mathrm{r}_{j4}(\beta_4,...,\ \mathrm{r}_{j_{n-1}}(\beta_{n-1},\ \mathrm{r}_{j_n}(\beta_n)...)$.

The Congruence predicate constructs the store θ (a set of constraints) and assigns an equivalence class $[q_i]$ to each $\mathrm{q}_i \in Q$. The set θ may contain constraints like $x\ in\ r$, = and $\neq$. Whenever the set of final constraints is satisfiable, if there is more than one solution, then we will choose the one which minimises the number of equivalence classes. Initially, $\theta = \emptyset$.

In order to extract equivalence classes, this predicate is applied to every pair of (compound) terms of A_{G_I}. Within each couple of terms, the predicate is applied to every couple of sub-term of $\hat{a}_1$ and $\hat{a}_2$. Backtracking is used to compute a consistent θ (which characterizes Lat_R). Initially, $[\mathrm{q}_i]$ is the equivalence class of each $\mathrm{q}_i \in Q$. Elements of I_+ and I_- are distinguished, hence we recognize final states (F_+ and F_- with $F = F_+ \cup F_-$ and $F_+ \cap F_- = \emptyset$) of these two sets from each other and from any other equivalence class.

Predicate Congruence(r_1, r_2) :
adds constraints to the constraint store θ
Let r_1 and r_2 be transition rules (for $\hat{a}_1$, $\hat{a}_2$) with $\alpha,\beta \in \Sigma$
$\mathrm{r}_1 : [\alpha] \times s'_1 \rightarrow s_1$ $\quad$ $\mathrm{r}_2 : [\beta] \times s'_2 \rightarrow s_2$

(1) if s_1 and s_2 are different final states in ($F_+ \times F_-$) then
add $[s_1] \neq [s_2]$.
(2) if $[\alpha] = [\beta]$ then
add $([s'_1] = [s'_2] \Rightarrow [\mathrm{s}_1] = [\mathrm{s}_2])$ *(The DFA condition)*
(3) if $[\alpha] \neq [\beta]$ then add $[s_1] \neq [s_2]$

Givens the rules r_1 and r_2 above (depicted in the figure 3 below) , the application of the Congruence predicate can produces 3 different configurations (i.e. $[\mathrm{s'}_1]=[\mathrm{s'}_2] \wedge [\mathrm{s}_1]=[\mathrm{s}_2]$, $[\mathrm{s'}_1]=[\mathrm{s'}_2] \wedge [\mathrm{s}_1] \neq [\mathrm{s}_2]$, $[\mathrm{s'}_1] \neq [\mathrm{s'}_2] \wedge [\mathrm{s}_1] \neq [\mathrm{s}_2]$).

Although $[\alpha]=\alpha$ is in its simplest form, we introduced the notion of equivalence class for the alphabet using the lexical class function $\mathrm{CL}(\alpha)=[\alpha]$ where:

$[\alpha]=[\beta]$ iff $\alpha = \beta$ or $\mathrm{CL}(\alpha)=\mathrm{CL}(\beta)$, $\alpha,\beta \in \Sigma$.

For example, different city names or two (possibly different) organizations (university, research laboratory) are equivalent.

Figure 3. transitions for r_1 and r_2

Obtaining the final induced automaton is a matter of search in Lat_R. This automaton is the solution of a consistent instantiation in the constraint store θ. Among all solutions, we pick up the one the minimizes the number of states.

The Congruence predicate is implemented as a (compiled) CLP program in GNU-Prolog[5].

This predicate takes as input the sets I_+ and I_- and generates the final DFA which is in turn an executable CLP program representing the induced Grammar. In the application (i.e. test) phase, we try to match the automaton of a new input seminar announcement. In the case of a success, further processing can take place (e.g. slot fillers value assignment in the announcement corpora, as stated briefly in the next section).

The following figure shows a more general case. Note that if we consider α_1 (resp. β_1) as the *left context* of α_2 (resp. β_2) and α_3 (resp. β_3) as its *right context*, we will cover, to some extent, the case studied in [21]:

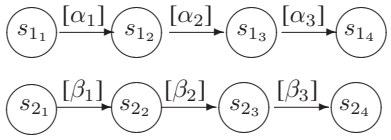


Figure 4. contextes and states

Applying the Congruence predicate to above case will produce 5 different configurations (depending on the equivalence classes of α_i,β_i) with various number of states in which the final induced minimal DFA has 4 states. Constraint store then will decide the final induced DFA considering all transitions and the negative examples.

It is worth emphasize that the Grammatical Induction applied only to positive examples (I_+) tends to overgeneralize L_+ (see e.g. [20]). Hence, one may express negative descriptions that are representative of the *words* to be rejected. For example, we may state that a *seminar announcement heading containing the Hour value* must be rejected. The I_- set of the section 5.5 contains some negative examples for an announcement heading.

[5]The final induced automaton is an extended DCG ([5])

4.1 An Example

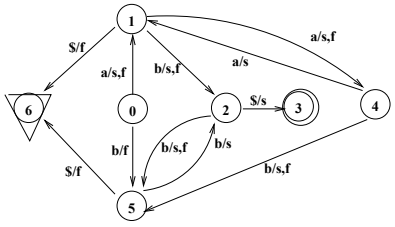
The theoretical aspects and the implementation issues of the related work were validated first by using the experimental protocol cited in [9]. The following example reports an original one that shows some interesting aspects of the grammatical inference engine.

Example : Consider that I denotes the regular language $\mathbf{a}^n\mathbf{b}^m$, $n, m > 0$. We have, Σ={a, b} with I_+ = {ab, aabb, aaabbb, aaaabbbb, ...} and I_- = {a,b, aab, abb, baa, bab, ...}.

The generated automaton A and $L(A)$ are given below.

L_+ = {a (aa)* b(bb)*} $\cup$ {aa (aa)* bb(bb)*}

L_- = {aa (aa)* b(bb)*} $\cup$ {a(bb)$^+$} $\cup$ {b(bb)$^+$} $\cup$ {a(aa)$^+$}



Here, notations like *a/s,f* over an arc means that the transition is a part of success (s) or failure (f) derivation. The tag $< s, f >$ means that the transition can possibly take to a success or to a failure. It is interesting to note that even though I contains words of the Context Free language $\mathbf{a}^n\mathbf{b}^n$, the above DFA extracts the following underlying knowledge from the sample set I (cf. $\mathbf{a}^n\mathbf{b}^m$): the DFA recognizes either `an even number of` a`'s followed by an even number of` b`'s` or `an odd number of` a`'s followed by an odd number of` b`'s`. But it rejects an even number of a's (resp. b's) followed by an odd number of b's (resp. a's) at the same time (which are words in L_-). Obviously, A can not generate that context free language but *learns* a part of it.

Relative to this induction is the notion of grammatical *enrichment*[6] that may be defined as follows. Suppose that the state [q] is originated from I_-. If there is any successful derivation of $\omega \in L(A)$ containing $L([q])$, then we say that I_- *enriches* L_+. In the above example, For example, bb derived through $q_0q_5q_2q_3$ is in the enriched L_+ if ever we do not constrain that derivation to only use success ($< s >$ or $< s, f >$ tag) edges.

5 The Text Mining Application

As mentioned above, we used the GI system to extract linguistic structure of different parts of a seminar announce-

[6]similar to the well known *version spaces.*

ment database. An example of such announcement is given below.

```
Seminar of the Institute of Nuclear
                  physics of Lyon
problem of the mode conversions
Yves  Colin de Verdiere
Fourier Institute of Grenoble
14:30 H - Room 27
Paul Dirac Building
```

We want to extract various information such as the *Date* or the *Subject* in a seminar. Finals measurements like the research fields of a university (or a researcher, etc.) can then be extracted. In this process whose goal is to extract slot fillers, valuable template slot fillers are already defined by an expert[7] : he/she knows in advance which kind of information is contained (and sought) in the data base.

It is also appropriate to note that a seminar announcement can be incomplete. For instance, the *Hour* may be missing within an announcement or it can be expressed in a different form (for example, by the "Friday afternoon" expression).

The reminder of this paper describes the use of the Grammatical Inference engine (consolidated by a Bayesian analysis, see the section 5.3) with respect to the textual IE task applied to the seminar announcement corpus.

5.1 Slots and Fillers of the Corpus

The following slots are defined for the seminar announcements corpus (abbreviations are further used in the paper).

$<Sub>$	the (general) $Topic$ and the $Subject$ of the seminar,
$<Org>$	the organizer, i.e. a university, lab.,...
$<Adr-Plc>$	the address and/or the place where the seminar takes place,
$<Sp>$	the person who will make the talk,
$<OrgSp>$	the organization of the Speaker (e.g. the research lab. of the Speaker),
$<Date>$	the date of the seminar,
$<Hr>$	the beginning hour (or the time range) of the seminar.

An announcement starts with the `seminar` ($séminaire$ in French) keyword.

5.2 Related Grammatical Inference

In the IE process applied to natural language texts, there are major differences between the Sentence Analysis and the traditional NLP parsers . The goal of syntactic analysis in an IE system is not to produce a complete parse tree for each sentence in the text. Instead, our system needs only to perform partial parsing. That is, it needs only to construct as much structures as the IE task requires.

Current methods (see e.g. [2], [1]) use generally global constraints to resolve local ambiguities. But because of the gaps in the grammatical and lexical coverage, full sentence parsers may end up making poor local decisions about structures in order to create a parse spanning the entire sentence.

Furthermore, the syntactic analysis in a text mining process is avoided for several more reasons:
- the cost and the complexity of this analysis,
- the very few use of the results of this analysis (the goal is not to correct errors or to translate the text),
- the texts may not follow the correct and complete syntax rules (of French in our case), etc.

A partial parser looks up for fragments of text that can be reliably recognized, e.g., *noun* and *verb* groups. Because of its limited coverage, a partial parser can rely on general pattern-matching techniques, particularly finite-state machines, to identify these fragments deterministically based on pure local syntactic elements. Partial parsing is well suited for information extraction applications for an additional reason : the ambiguity resolution decisions that makes full parsing difficult can be postponed until later stages of the processing where top-down expectations from information extraction task can guide the system's actions.

In our seminar announcement corpus, the subject is similar to a *noun group* but may not follow its rigorous syntax. Then, the inference stage helps, in this case, to retain *effective* rules used in the examples. Therefore, the corresponding text mining process will rather be a syntax directed process.

Starting from a sample set (positive examples and negative cases description, see the section 5.5 for an example of GI), the Grammatical Inference (GI) induces a regular grammar[8] (a DFA) of this sample set. In the test phase, sentences presented to the grammar will be regarded as pertaining (or not) w.r.t. the language generated by induced grammar.

The Grammatical Inference carries out a classification of the sentences (*accept* or *reject* means belonging or not to a given language) but, in its original form, it does not handle the semantics of these constructions. Hence, Bayesian measures will guide the process by predicting the slot and its value (in its context) to be submitted to the grammar. The IE process is then achieved with more precision and reliability (see also [35]).

[7] The expert in this domain is just a scientifique-researcher familiar with such seminar announcements

[8] We note that the Context-Free grammar induction is an actual and active research filed facing hard constraints making the general Context Free induction problem no-decidable.

5.3 Naive Bayesian use

Several techniques of text mining use the Bayesian analysis that (even in its naive form) gives interesting results. In the method known as *naive Bayesian*, the document is presented as a vector of characteristics (e.g. various sections of an announcement). Other presentations such as *bag of words* consider the text in the form of a collection of words where any internal structure (physical, logical, morpho-syntactic or semantics) is inhibited.

The Bayesian rule is recalled below. Given a hypothesis (e.g. to have such a section of the class C in such a context inside a seminar announcement) and an announcement E over C, we have:

$$Pr(C/E) = \frac{Pr(E/C)\,.\,Pr(C)}{Pr(E)}$$

The idea is to express the weighted probability of the membership of a pattern or a sub-language within a class C according to the characteristic of the text E and those of other texts classified as such.

To summarise the current process, key patterns leading to recognize the various (but not all) fillers of an announcement are first defined during the training stage. Together with the key patterns, the frequency measurements and the regular production rules will help to decide (to *classify*) a section of the announcement. During the test phase, a pattern p first gets a probability to belong to a slot filler by the presence of a deterministic keyword (100 %) and/or by the probability (from the frequency table) of its (possibly left and right) contexts. p is then submitted to the induced grammar according to these probabilities. Failure cases are postponed to the postprocessing step[9]. The process uses the backtracking to consider other possibilities (see section 6 for the Sub filler).

5.4 Details of the GI process

It is easy to note that a simple textual search (based on keywords) cannot be appropriate for extracting knowledge from our seminar announcements. Methods of knowledge extraction based on the Bayesian analysis allow to predict the position of a given information in the text together with its average length (see e.g. [35]). This technique, based on the learning of the position of a section (e.g. the $<Sub>$ section) would not be appropriate here because of the free format of the announcements. In addition, an announcement can be incomplete. Thus, getting the induced grammar of e.g. the $<Adr - Place>$ section will make it possible to analyse the content of that sub-language.

We use the grammatical inference in various sub-languages (e.g. the *heading* or the *subject* of an announcement) that may contain relevant information. As an example, the heading can contain a topic, a subject or an organizer that can be possibly extended in the reminder of the announcement. The subject (Sub) can add precise details to the Topic of the seminar and vice versa. Such complementary data are registered both in the frequency table and hard coded in the production rules. The sequence of operations is governed by the key patterns, the probabilities from the frequency table (table 1) and, finally, by the production rules [10].

It may be noted that if the Grammatical Induction is processed only upon positive examples (the set I_+ set below), then the result tends to over-generalise the language induced. Hence, the expert may express negative descriptions that are representative of the *words* that must be rejected[11]. For example, he may state that a *seminar announcement heading containing the Hour value* must be rejected. The following example contains some negative examples for an announcement heading (the set I_-).

5.5 An Example of GI

As an example, the results of the grammatical inference on the **heading** of announcement follows. The grammar below partially describes what the headings of the sample set contained. Hence, the following I_+ does not cover all possible headings in all seminar announcements, but those of the sample set.

I_+={'SDON', 'S:T', 'S', 'ST','SDT','SàV:T','SN:T', ...}
I_-={'Sa', 'SS', 'S::T', 'S::L', 'S::N','SDD','SOO', 'Sàà',
'Sa:', 'SD:', 'SaVV', ...} where

S : the "séminaire" keyword (seminar in English),

D : $< Det >$, a determinant (e.g., *'du'*, *'de la'*, *'des'*) like *'of'* or *'of the'* in English

T : *<Thème>* , an exposed $Topic - Subject$ (e.g. *Algorithm*, *Complexity , Internet and Security*, etc.),

N :$< Nom >$, a Noun, e.g. name of a research laboratory ,

O : $< Org >$, an organization name (e.g. *institute, laboratory, university, school...*)

V : Ville, name of a City, e.g. *Toulouse*

':' : this character,

'à' : this character (stands for *'at'* or *'in'*, ... in English).

[9] e.g. in the case of ambiguity (or failure on p), if a pattern p' ($p' \neq p$) has been successfully recognized to fulfill a slot filler, the pattern p is *tried* against other related sections. Several lookup may be necessary in more complex cases. A *blind* application of the induced production rules is the last chance.

[10] However, we are not in the context of the so-called *Probabilistic Grammars*

[11] For the seminar announcement case, negative examples are quite straightforward.

The induced grammar accepts the language L_+ (the induced language of I_+) and rejects those of L_- (the induced language of I_-). The final induced automaton accepts the language given below[12]. The rules that reject unsuitable constructions (i.e. words in L_-) are not reported here for the sake of clarity. However, one may observe that a rejection takes place in the induced DFA when a derivation (upon a token) leads to a final *failure* state F_- (see the section 5.4).

The language of the induced finite state automaton

The language induced from the set $I = (I_+ \cup I_-)$ for the *heading* part of announcements is given below. Recall that this definition gives only the successful derivation paths.

L+ = "Séminaire" . L1
L1 = ϵ || (':' | 'à') . L3 || $<Nom>$. L5 || <*Thème*> . L6
L3 = $<Org>$. L6 || ($<Thème>$ | | $<Ville>$) . L1
L5 = ':'. L3 L6 = ϵ || $<Nom>$. L1

Nota Bene: the induced grammar is an operational logical grammar (extended DCG). Predicates expressing constraints and actions are then added to its rules (see the *Date* example below). As an example of action, while recognizing (in their context):

- a <*Thème*> may contain a part of the $Subject$; then the value corresponding to the $Subject$ will be added to the $<Sub>$ filler;
- for a $<Ville>$, the corresponding city value will be added to $<ADR - Place>$ filler[13].

Other possible adjustment actions are achieved during the post-processing phase.

5.6 An example: the $Date$ analyser DCG

Below, some of the induced grammar rules (annotated by their *semantical actions*[14] given inside brackets) for the $< Date >$ filler are given. The lack of any part of a $Date$ (e.g., the *day-name*) is not reported here[15].

$< \mathbf{Date} >$:: ["$date$"][" : "]["le"][$< Day - name >$]
[$< Mid - day >$]["le"] $< Day >$ [$< Sep >$]
$< Month >$ [$< Sep >$] $< Year >$.

$<\mathbf{Mid - day}>$:: $<Word>$
{\$1 ∈ {"$matin$"} ;
$add(part\_of\_Heure$, "$8h - 12h$", 100) OR
\$1 ∈ {"$après - midi$"} ;
$add(part\_of\_Heure$, "$14h - 18h$", 100)}.

$< \mathbf{Month} >$:: $< Number >$
{\$1 ∈ {1..12} ; $add(part\_of\_Date, \$1, 100)$}
\|\| $< Word >$
{\$1 ∈ {"jan"..."dec"} ; $add(part\_of\_Date, \$1, 100)$}.

$<\mathbf{Day - name}>$:: $< Word >$
{\$1 ∈ {"lun"..."sam"} ; $add(part\_of\_Date, \$1, 100)$}.

$< \mathbf{Day} >$:: $< Number >$
{\$1 ∈ {1..31}; $add(part\_of\_Date, \$1, 100)$}.

$< \mathbf{Year} >$:: $< Number >$
{\$1 ≥ 1990}; $add(part\_of\_Date, \$1, 100)$}.

$< \mathbf{Sep} >$:: '/' \|\| ' :' \|\| '−' \|\| ... $--$ *a separator*

Nota Bene: the value 100 (parameter of the predicate add) indicates the confidence coefficient of the filler assigned to the slot. Here, the case of $< Date >$ is rather simple and follows a known format. We may however note that the presence of "matin/après-midi" (AM/PM in English) of the $< Date >$ will complete the $< Hr >$ slot filler.

5.7 Frequency Measurements

The following percentage values is constructed from the input samples. Here, $OrgSp$ abbreviates $'Organizer - Speaker'$, $Pres$ stands for $'Present'$, Sub for $'Subject'$, Plc for $'Place'$, Hr for $'Hour'$ and Sp for $'Speaker'$:

	Sub	Org	Date	Hr	Plc	Adr	Sp	OrgSp	End	Pres
Ance	14	9	41	4	14	14	9	0	0	100
Sub	0	0	4	0	9	0	23	0	9	45
Org	0	0	4	0	4	0	0	0	0	9
Date	0	0	0	77	9	0	4	0	4	95
Hr	9	0	4	0	41	0	18	0	14	86
Plc	4	0	23	0	0	36	4	0	23	91
Adr	4	0	9	0	4	0	0	0	32	50
Sp	4	0	9	0	4	0	0	36	4	59
OrgSp	14	0	0	4	4	0	0	0	14	36

Table 1. Frequency table of an announcement sections

In the above table, a cell C_ij gives the frequency (or the *Support*, see below for a definition) of the column j that followed the line i in the training set. The *Pres* (Presence) column (the last one) gives the frequency of each element of the line in the training set (e.g. the Sub is present only in 45% of the announcements). We add to this table two other values: 77% of the announcements contain a $Topic$ in their heading, and 18% of the headings contain an indication on the organizer (Org).

The cells containing 0% are of a particular interest because they give indications on the cases that do not occur. For example, $< OrgSp >$ never follows the heading of an announcement.

As an example, we apply the conditional probability to the section Sub of the example of section 5 where the slot of the second line is not determined. This example shows how the post-processing will help deciding that slots filler. Given the table 1 above, the probability so that the unknown section (in the example given in the section 5) is a $Subject$ (surrounded by the *Heading* and the *Speaker*) is 12%. However, this announcement does not contain a Subject in its heading and, the Speaker is the successor of a $Subject$ in $\frac{23}{45}$ cases. Therefore, the filler is predicted at 23% (weighted 51%) to be the $Subject$.

[12] Notation: $(X\|Y)$ means (X or Y). The dot denotes the monoid concatenation and ε denote the empty string.

[13] In the $<Adr - Plc>$ context.

[14] a semantical action is a term from the *syntax directed* and the *Attributed Grammars* paradigm which denotes (no syntactic) actions based on the attribute values. Distinguished from the pure syntactical analysis, such actions take place in a production rule if the rule applied.

[15] $[x]$ means an optional x; \$k denotes the value of the k^{th} literal (a.k.a. $yacc$ compiler compiler).

Note that the strongest probability of the section that follows the *heading* is the $Date$ section. However, one can recognize a $Date$ by the keywords in the induced grammar.

The depth of the Morpho-Syntactic analysis engine is a system parameter. In some cases, the (partial) linguistic class from this filler can be extracted giving a (partial) Noun Group (even without any initial determinant, see e.g. [1]).

6 Results for the Example

This section describes briefly some experimentations on the seminar announcement corpus.

For the grammatical induction, the GI process is applied within the morphological step in order to learn to reject useless combinations like those constructions that are liguistically ambiguous and useless for us[16]. Once the learning step is achieved on the seminar corpus, we obtained the following results for the seminar example of the section 5 (confidence coefficient for a filler value is reported at right when it is less than 100; the original database is in French):

```
Org     = "Institut de Physique Nucleaire de Lyon"
Sub     = "Le probleme des conversions de modes" (51)
Sp      = "Yves Colin de Verdiere"               (51)
OrgSp   = "Institut Fourier Grenoble"            (61)
Hr      = "14:30 H"
Adr_Plc= "Salle 27-Rez de chaussee-Bât. Paul Dirac"
Adr     = "Institut de Physique Nucleaire de Lyon"
Date    = ""
```

7 Performances Evaluation

Several textual IE systems, notably those of MUCs, involved large training corpora with thousands of documents (see e.g. [26]). However, such large training corpora (and their associated templates) may not be available for most real tasks.

Experiments with smaller training collections (such as the 100 documents provided for MUC-6) suggest that fully automated learning techniques applied to a few text examples with minimal automatic syntactic processing may not be able to achieve sufficient coverage (see e.g. [34]).

We paid a special attention to the over generalization pitfall of the GI engine. An amount of work was done in testing the GI engine on several different corpora (bibliography, abstract, table of content, etc.) in order to improve the induction algorithm. The GI engine is parametric such that several different degrees of generalization[17] can be set (by varying the constraints over the language-inclusion lattice of automata). The output automaton is then tested against the training set and the one (that accepts all positive examples rejecting all negative one) with the least number of states is chosen. One may observe that the refinement operator is hard-coded within the the *Congruence Predicate* of section 4

In addition, another parameter is available in the GI engine the turns on-off the so-called *enrichment* issue (section 4.1).

However, we are aware that larger sample sets (and other domain specific corpora such as *abstract* scanning) are needed to improve the system. Larger sample set has however an inconvenience. Recall that the search space is given by the lattice of language-inclusion specified by the GI process and illustrated by the *Congruence Predicate*. This search space grows exponentially with the size of sample set I.

Starting with 300 examples, we applied a ten-fold cross validation and observed that the results were not significantly changed for more examples.

Metric used : in the **IE task** (i.e. the corpus is known to contain announcements), evaluation metrics are based on the filler presence and prediction.

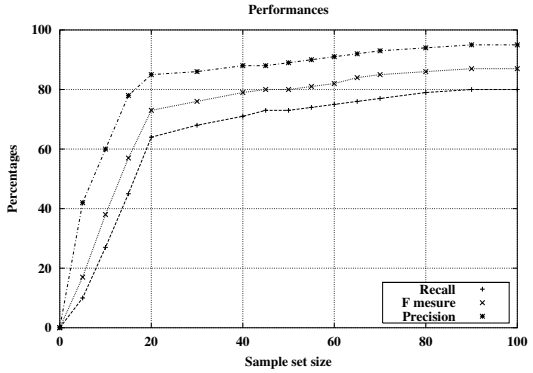
$$Precision = \frac{Number\ of\ Correctly\ assigned\ slots}{Number\ of\ assigned\ slots}$$

$$Recall = \frac{Number\ of\ Correctly\ assigned\ slots}{Number\ of\ correct\ present\ slots}$$

In addition, an harmonic measure called F-measure (see e.g. [29]) is used to give the mean of the above values:

$$F - measure = \frac{Precision \times Recall}{\frac{1}{2}(Precision + Recall)}$$

Figure 5. Performance evaluation



The diagram of the figure 5 shows the performance percentages we obtained.For the seminar announcements corpus, it is not surprising to have high performance values (95% and 80%) given the intended slots and the relative low

[16] Here, some linguistic knowledge is required to eliminate useless lexical class combinations from morphological analysis.

[17] Three for the moment

risk of error. The system is quite domain specific and may even be enhanced. Student work is currently done to adapt the system to other corpora.

8 The Related Work

Several textual IE system have been proposed since the focus on researches started by MUC program of DARPA (e.g. [22], [29]).

The use of pattern dictionary is common to many systems. Some uses clustering to create patterns by generalizing those identified by an expert (see e.g. [33]). The dictionnary used during the analysis. step contains basically keywords (and their lexical class).

Syntactic information can be used as in Autoslog ([31], [32]) that uses a set of general syntactic patterns validated by an expert. Among these systems, some uses advanced syntactic analysis to identify the relationship between the syntactic elements and the linguistic entities (e.g. in [28]). This analysis is costly (when the semantic information is not used) and may limit the system specially if linguistic rules are not respected (like in our seminar examples).

In many IE systems, human interaction is highly required through different phases of training. Machine Learning techniques like decision trees are used ([30]) to extract coreferences using the annotated coreference examples.

Among these systems, the current work is closed to PAPIER system ([21]). RAPIER is an ILP system that takes pairs of documents and filled templates and induces rules that directly extract fillers for the slots in the template. This system uses constraints on words and part-of-speech tags surrounding the fillers' left and right contexts. To some extent, our system can be seen from this point of view since, as mentioned in the GI (section 4), our grammatical Inference engine implements this technique implicitly. In addition, these results should be compared with those of the *Named-Entity* research work (see e.g. [3]) and aims to learn *names* by identifying all named locations, persons, organizations dates and so on.

9 Conclusion

In this paper, a new constraint satisfaction framework for GI has been presented and implemented by an operational constraint logic program that outputs the final DFA. Here, the algebraic specification gives a theoretical framework to state **why** some processing are done (e.g. merged states) before explaining **how** we do process. This specification allows to show that the homomorphism f (section 3.1) exists and we gave an implementation of it by the *Congruence* predicate which produces a set of constraints. If this set is satisfiable, then we choose a solution with the fewest number of states.

Among other works in the field, [13] and [14] proposed similar methods for document analysis. But in the algebraic and constraint satisfaction frameworks of the Grammatical Inference, the logical aspects for the direct grammar extraction have, as well as known, not yet been investigated.

This work was initiated in a (paper) document processing project where GI results are used to classify and then translate documents into machine readable form. Other applications dealing with more general multimedia contents (video in particular) are under the study.

The code in GNU-Prolog of the realization is available from the author.

On top of the GI part, we designed and implemented an IE system that fills slots of a template associated to seminar announcements using Bayesian measurements. Once the template are slots filled, usual techniques of Data Mining can applied to the results since the resulting values of the slots describe simply a relational database scheme. One current use of the system is to extract information like the research field of universities, laboratories or researchers. That is, to guide PHD students in their researches.

This is a work in progress and the performance results are encouraging to continue the project. We plan to first enhance and then extend the system to other corpora like job announcements and marine weather announcements. The aim is to establish statistics on marine catastrophes and previsions. The system will be integrated to a (database) Datamining engine in order to establish valuable information on marine events.

References

[1] D. Appelt et all. : *FASTUS system : MUC-6 test results and analysis*. In Proc. 6^th MUC, 1995, Morgan Kaufmann.

[2] R. Weischedel *BBN : Description of the PLUM system as used for MUC-6.* In Proc. 6^th MUC. 1995, Morgan Kaufmann.

[3] D. Bikel, R. Schwartz, R. Weischedel *An Algorithm that Learns What's in a Name*, 1999

[4] D.D. Palmer, D.S. Day, *A Statistical Profile of the Named Entity Task*, in proc. of th 5th. Conf. on Applied Natural Language Processing, Washington D.C., 1997 ACL.

[5] A. Colmerauer. "Metamorphosis grammars". In : Natural Communication with Computer. Berlin : Springer-Verlag 1977, LNCS 63, pp. 133-189.

[6] P. Van Hentenryck. "Constraint Satisfaction in Logic Programming". MIT Press. Ehud Shapiro Ed. 224 p. 1989.

[7] E.M. Gold. "Language identification in the limit". Inf. and Control, 10(5)- 1967.

[8] E.M. Gold. "Complexity of automaton identification from given data". Information and Control, 37- 1978.

[9] P. Dupont. "Regular Grammatical Inference from Positive and Negative Samples by Genetic Search : the GIG method". ICGI'94, Grammatical Inference and Applications. Springer-Verlag-94, LNCS 862.

[10] F. Coste, J. Nicols : "Regular Inference as a graph coloring Problem". ICML'97. 1997.

[11] J. A. Goguen, J.W. Tatcher, E.G. Wagner, J.B. Wright. "Initial Algebra Semantics and Continuous Algebra". JACM 24(1). 1977.

[12] H. Ehrig, B. Mahr. " Fundamentals of Algebraic Specification". Vol-1 & 2. Springer-Verlag1985.

[13] H. Ahohen, H. Mannila. "Forming Grammars for structured documents". Research report. University of Helsinki. 1994.

[14] G. LindŽn "Structured Document Transformation". PhD Thesis. University of Helsinki. Finland june 1997.

[15] "Grammatical Inference : Algorithms and Applications", 4th Int. Col. ICGI 1998, Springer-Verlag LNCS. 1433.

[16] "Grammatical Inference : Algorithms and Applications", 5th Int. Col. ICGI 2000, Lisbon, Portugal, Springer-Verlag LNCS. 1891.

[17] "Grammatical Inference : Algorithms and Applications", 6th Int. Col. ICGI 2002, Springer-Verlag ISBN 3-540-44239-1.

[18] ICCSA: International Workshop on Advances in Structural and Syntactical Pattern Recognition, LNCS, 2002.

[19] David J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*, Cambridge University Press, 2003, available from http://www.inference.phy.cam.ac.uk/mackay/itila/.

[20] H. Ahonen, O. Heinonen, M. Klemettinen, and A. Verkamo. *Applying data mining techniques for descriptive phrase extraction in digital documents*. In Proc. Advances in Digital Libraries (ADL98), Santa Barbara, CA, 1998.

[21] M.E. Califf, R.J. Mooney, *Relational Learning of Pattern-Match Rules for I.E.*, Proc. of AAAI Symposium on Applying Machine Learning to Discourse Processing, 1997.

[22] ed., *Proc. of 4th and 5th DARPA Message Understanding Evaluation and Conference*. Morgan Kaufman. 1992, 1993.

[23] M. Dixon,*An Overview of Document Mining Technology*, 1997, http://citeseer.nj.nec.com/dixon97overview.html.

[24] U. fayyad & all *From DataMining to Knowledge discovery : An overview*. in Advances in Knowledge Dicovery and DataMining, MIT Press, Cambridge, Mass 1996.

[25] R. feldman, I. Dagan *Knowledge Discovery in textuel databases (KDT)*. Proc. of the 1st Int. Conf. on Knowledge Discovery and DataMining (KDD-95), Montreal, Ca, AAAI Press, 1995.

[26] R. Grishman, *Information Extraction: Techniques and Challenges*, http://citeseer.nj.nec.com/ grishman97information.html.

[27] M.A. Hearst, *Text Data Mining : Issues, Techniques and Relationship to Information Axxess*, Presentation Notes for UW/MS Workshop on data mining, July 1997.

[28] S. B. Huffman, *Learning information extraction from examples*. in Wermter, Riloff, Scheler ed. Berlin, 1996.

[29] W. Lehnert, B. Sundheim, *A performance evaluation of text-analysis technologies*, AI Magazine 12(3), 1991.

[30] J. McCartht, W. Lehnert, *Using decision trees for coreference resolution.*, in Proc. of 4th Int. Conf. on IA, 1995.

[31] E. Riloff, *Atomatically constructing a dictionnart for information extraction tasks*. in Proc. of 11th National Conf. on AI, 1993.

[32] E. Riloff, *Atomatically generating extraction patterns from untagged text.*. in Proc. of 13th National Conf. on AI, 1996.

[33] S. Soderland, D. Fisher, J. Aseltine and W. Lehnert, *Crystal : inducing a conceptual dictionnary*, in Proc. of the 14th Int. Conf. on AI, 1995

[34] D. Fisher'96 et all. *Description of UMass system as used for MUC-6*. In Proc.. 6^th MUC. 1996, Morgan Kaufmann.

[35] D. Freitag. *Using Grammatical Inferenec to Improve Precision in*. ICML'97. 1997.