An automatic ontology-based approach to enrich tables semantically

Hélène Gagliardi, Ollivier Haemmerlé, Nathalie Pernelle, Fatiha Saïs
LRI (UMR CNRS 8623 - University of Paris-Sud),
Bâtiment 490, F-91405 Orsay Cedex, France
{gag,pernelle,sais}@lri.fr, Ollivier.Haemmerle@inapg.inra.fr

Abstract

This work aims at building automatically a thematic data warehouse composed of heterogeneous XML documents extracted from the Web. We focus on the data tables contained in these documents. This article presents how we enrich semantically those tables by means of tags and values coming from¡ the ontology of the application. First results are given for a set of real data of the e.dot project.

Introduction

Our work deals with the automatic construction of domain specific data warehouses. More precisely, our goal is to integrate automatically information found on the Web with existing information stored in different databases. The first originality of our work is that the unique external source of knowledge used to extract information is an ontology of the application domain. Then our approach is completely generic. The second originality is that the extraction of information is done in a completely automatic manner. The drawback of such a non-supervised approach is that it leads to ambiguities or misunderstandings in the information we discover. But we propose to keep the different interpretations in order to allow their use during the query processing. That flexibility is the third originality of our technique. The fourth originality is that we exclusively extract information from data tables in the documents we found on the Web. Such a choice can appear as restrictive, but in a large variety of scientific fields, we saw that data tables contain synthetic and reliable information. Finally, our approach is currently under test in a real and ambitious project concerning the microbiological risk in food products.

Our application domain concerns the microbiological risk in food products. In order to understand and to prevent such risks, the Sym'Previus project has been launched by French governmental institutions. During the Sym'Previus project, the MIEL++ system has been built (Buche *et al.* 2004). MIEL++ is a tool based on a database, containing experimental results and industrial results about the behaviour of pathogenic germs in food products depending on several parameters, such as the temperature, the pH, etc. The Sym'Previus database is incomplete by nature since the number of possible experiments is potentially infinite. The work presented in this article takes place within the e.dot project, which is a cooperation between the INA P-G/INRA MIA group, the Xyleme start-up, the IASI-Gemo team (LRI) and the Verso-Gemo team (INRIA-Futurs). The goal of the e.dot project is to palliate the incompleteness of the database by complementing it with data automatically extracted from the Web. The drawback of such a technique is that the way the data are expressed on the Web is very heterogeneous. For example, the terms used in the scientific articles in microbiology can be different from an article to another. A way of solving that heterogeneity issue can be to query the existing database and the Web documents through a mediated architecture based on a domain ontology.

In MIEL++, the database is queried through a mediated architecture (2 local bases previously developed during the Sym'Previus project, and expressed in heterogeneous formalisms are actually queried on). The mediated schema is composed of an ontology called the Sym'Previus ontology. In order to make possible the query processing on the data extracted from the Web, we need to translate these data in order to make them compatible with the Sym'Previus ontology used in the mediated schema. That mechanism is presented in this article.

In e.dot project (e.dot 2004), data are acquired by going through the following steps. First, a Web crawler is combined with a filtering tool (Mezaour 2005) that selects the Web pages that contain data useful for the warehouse. We exclusively focus on documents in Html or Pdf format which contain data tables; actually data tables are very common presentation scheme for authors in order to describe experimental results, statistical or other synthetic data in scientific articles. In our system, these tables are extracted and transformed in a generic XML representation called XTab. These documents are then semantically enriched and stored in the data warehouse.

In this paper, we present the semantic enrichment step. In our approach, we want this transformation to be as automatic and flexible as possible, only driven by the ontology and the way the data have been structured in the original table. Thus, we have defined a Document Type Definition named SML (Semantic Markup Language) which can automatically be generated using the ontology and which can

deal with additional or incomplete information in a semantic relation, ambiguities or possible interpretation errors. This approach has been implemented and tested on real data from the e.dot project.

The paper is structured as follows. In section 2, we first introduce the XTab format, the Sym'Previus ontology, and a simple example in order to explain the aims of the semantic enrichment task. Section 3 introduces the way we identify the ontology terms represented by the columns of a table. Section 4 presents the identification of semantic relations in data table, while section 5 explains the instantiation of such semantic relation. Section 6 gives an idea of the possible use of the semantic enrichment during the query processing. In section 7, some experimental results are shown. In the conclusion, we present related works and we give future directions.

Preliminary notions

We first present the generic XML representation of tables – called XTab. Then we introduce the ontology of the application domain. That section ends with a very preliminary example of what the result of the semantic enrichment is.

The XTab format

The data tables are first represented in XML, using purely syntactic tags that are domain-independent. The tables are automatically represented using a list of lines, each line being composed of a list of cells. Besides, when it is possible, titles are extracted. This format called XTab has been defined in the e.dot project (e.dot 2004). More complex structures of tables need heuristics such as (Pivk, Cimiano, & Sure 2004) in order to be translated into this simple XTab structure. These heuristics are not presented here. The XTab representation of Figure 1 is shown in Figure 2.

Products	pH values
Cultivated mushroom	5.00
Crab	6.60

Figure 1: *approximative pH of some food products*

The Sym'Previus ontology

The Sym'Previus project (sym) has developed an ontology dedicated to the risk assessment domain. In order to exploit the data tables and query them through the MIEL++ system – which is based on the Sym'Previus ontology – we have to express data using the vocabulary stored in that ontology. The Sym'Previus ontology is composed of:

1. a term taxonomy which contains 428 terms of the domain (food, microorganism, experimental factors, ...) which are organized by the specialization relation $\preceq$;
2. a relational schema that contains 25 semantic relations between terms of the taxonomy. A semantic relation r is characterized by its signature $attrs(r)$ composed of the set of attributes of the relation. The elements of $attrs(r)$ belong to the term taxonomy. For instance, the relation *foodFactorMicroorganism* has the signature *(food, factor, microorganism).*

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<table><title> <table-title>
approximative pH of some food products</table-title>
<column-title>Products</column-title>
<column-title>pH values</column-title></title>
<nb-col>2</nb-col>
<content>
<line>
<cell>cultivated mushroom</cell>
<cell>5.00</cell>
</line>
<line>
<cell>crab</cell>
<cell>6.60</cell>
</line>
</content></table>
```

Figure 2: *XTab Representation of figure 1*

Very preliminary example

Thus, we enrich XTab documents with tags and values provided by the ontology. More precisely, we have defined a representation formalism named SML – *Semantic Markup Language* – where table lines are not represented by cells anymore but by a set of semantic relations between columns.

```
<table> <title><table-title>
approximative pH of some food products </table-title>
<column-title> Products </column-title>
<column-title>pH values</column-title>...
</title> <content>
<rowRel>
<foodPH>
<food><ontoVal>mushroom</ontoVal>
<originalVal> cultivated mushroom </originalVal>
</food>
<ph><ontoVal/>
<originalVal>5.00</originalVal></ph> </foodPH>
</rowRel>
<rowRel>
<foodPH>
<food><ontoVal>crab</ontoVal>
<originalVal>crab</originalVal></food>
<ph> <ontoVal/><originalVal>6.60</originalVal> </ph>
</foodPH> </rowRel>
</content> </table>
```

Figure 3: *Simplified SML Representation of Figure 1*

Let us consider the semantic relation named *foodPH* which links a food product with its pH value in the ontology. The aim of the enrichment is to reformulate an XTab document such as Figure 2 in an SML document such as Figure 3. In this SML document, the semantic relation *foodPH* which has been recognized in the table is represented and instantiated using the table values.

In order to instantiate the relation, we try to associate one

or several terms of the taxonomy with each value of the table. If the value does not appear directly in the taxonomy, we use mapping techniques in order to find similar terms. In the example, the first column value *crab* belongs to the taxonomy. But the value *cultivated mushroom* does not appear in the taxonomy; nevertheless, we propose to associate *mushroom* with it thanks to a mapping procedure. This value is represented in the SML tag $< ontoVal >$ while the original value is kept using the tag $< originalVal >$. Thus, the original value can be shown in the result of a query, even if the query is asked on a value belonging to the ontology. This SML representation conforms to the SML DTD (Document Type Definition) we have defined in (e.dot 2004).

Identification of the columns of the data table

In order to extract the relations of the table, we perform two steps. The first one, presented in this section, consists in identifying a term of the taxonomy which represents each column of the data table. The second step, presented in the next section, will consist in discovering semantic relations between data table organized in columns.

The identification of the columns of the data table is based on two pieces of information: the content of the column which is mainly used, and the title of the column, which is used in case the content of the column is not helpful enough.

The content of the column is used as follows: we try to associate a term of the ontology taxonomy with each value belonging to the column. Then we search for common generalizers – "subsumers" of these terms. The use of a threshold allows us to associate a generalizer with a given column even if we have not recognized all the values of that column.

definition An *A-term* is a term of the taxonomy that appears at least one time as an attribute of a relation signature in the relational schema of the ontology. The set of all A-terms is noted *AT*.

We first try to find values of the columns that belong to the taxonomy or that are included[1] in one term of the taxonomy. We then look for an A-term which subsumes almost all the values in the term taxonomy. First, an A-term can be associated with a column Col if and only if the rate of the subsumed values is greater than a given threshold th. The set of all A-terms that verify this constraint is noted $ATCandidate(Col, th)$:

$$ATCandidate(Col, th) = \{t \,|t \, in \, AT \, and \, \frac{|sub(t, Col)|}{|Col|} \geq th\}$$

where *sub(t,Col)* is the set of values of Col that are subsumed by the A-term t.

Among these candidates, we select the most specific A-terms that subsume the largest set of values. This set of representative A-terms is noted *ATRep*:

$$ATRep(Col, th) = \{t \mid t \in ATCandidate(Col, th),$$
$$\neg\exists \, t' \, such \, that \, t' \in ATCandidate(Col, th)$$
$$and \, |sub(t', Col)| > |sub(t, Col)|,$$
$$\neg\exists \, t'' \, such \, that \, t'' \in ATCandidate(Col, th)$$
$$and \, |sub(t'', Col)| = |sub(t, Col)| \, and \, t'' \preceq t\}$$

If there is more than one A-Term in ATRep, we keep the first one. In fact, experiments have shown that if the threshold is high enough there is zero or one representative A-term.

If no representative A-term has been found by using this procedure, we exploit the title of the column if it is available. We exploit the values of the column first because if we are able to identify an important number of values, the A-term is often relevant. Besides, the treatment of the title can lead us to a misunderstanding association. If no A-Term has been found, we keep the column in the SML document and we associate the generic A-term named *attribute* with it.

Products	Qty	Lipids	Calories
whiting with lemon	100 g	7.8 g	92 kcal
ground crab	150 g	11.25 g	192 kcal
chicken	250 g	18.75 g	312 kcal

Figure 4: *Nutritional Composition of some food products*

In the table of Figure 4, the terms *crab* and *chicken* belonging to the ontology have been associated with the values *ground crab* and *chicken*. If the threshold is 0.5, the most specific A-term that subsumes these two terms is the A-term *Food*. The second column has not been identified because it only contains numeric values and the title is an abbreviation; the generic A-Term *attribute* is associated with it. *lipid* and *calorie* have been associated with the last two columns thanks to the exploitation of their titles.

Definition The schema *tabSch* of a table *tab*, noted *tabSch(tab)*, is the finite set of couples $(col, ATRep(col, th))$ that can be found for a given threshold th.

$$tabSch(Tab) = \{(col, t) | t \in ATRep(col, th)$$
$$or \, [(t = attribute) \, and \, ATRep(col, th) = \emptyset]\}$$

The schema of the table *Tab2* shown in Figure 4 is:

tabSch(Tab2)={(1,food)(2,attribute)(3,lipid),(4,calorie)}

Identification of the semantic relations appearing in the data table

We present now how we identify one or several semantic relations in the schema of the table. That identification is done by comparing the "natures" of the columns identified during the previous step with the attributes appearing in the signatures of the semantic relations of the ontology of the domain. Of course, an exact mapping between the schema of the table and the signature of a specific semantic relation is the ideal case. In most of the cases, we will obtain several possible mapping with subsets of the

[1] in the sense of the inclusion of sets of words, after a lemmatization step and without taking the "empty" words (determiners or prepositions) into account

attributes of the schema of the table. Or we will have only partial mapping, with only a subset of the attributes of the signature of a relation, etc. So we will see that we propose an automatic identification of the semantic relations as flexible as possible.

We say that a relation is ***completely represented*** if each attribute of its signature subsumes or is equal to a distinct A-term of the table schema.

Thus, suppose that the three relations *foodLipid*, *foodCalorie*, *foodPh* belong to the ontology and that the two relations *foodLipid* and *foodCalorie* mean "the number of lipid (or calories) contained in 100 g of the foodstuff", because the experts have considered that the weight is normalized. In table of Figure 4, the relations are extracted in the following way:

- *foodLipid*, is completely represented by the values found in the first and the third columns.
- *foodCalorie* is completely represented by the values found in the first and the fourth columns.

Since the second column *qty* is not identified and does not participate to any of these two relations, we add to each relation a generic attribute which will contain values found in this second column. If this attribute was not represented, for example, the third line of the table would be interpreted as *"**100g** of chicken correspond to 312 calories"*. When the generic attribute is taken into account, the interpretation is "***250g*** *of chicken correspond to 312 calories*". So, in such cases, the representation of additional information leads to better interpretations of the data.

Figure 5 proposes the SML representation of the relations *foodLipid* and *foodCalorie* :

```
<table> <content>
<rowRel additionalAttr="yes">
<foodLipid relType ="completeRel">
<food>...</food> <lipid> ... </lipid>
<attribute> ...</attribute>
</foodLipid>
<foodCalorie relType ="completeRel">
<food>...</food> <calorie> ... </calorie>
<attribute> ...</attribute>
</foodCalorie>
</rowRel> ...
</content> </table>
```

Figure 5: *SML representation of completely represented relations*

We say that a relation is ***partially represented*** if it is not completely represented and if at least two attributes of its signature subsume or are equal to different A-terms of the schema of the table. We have considered partially represented relations in order to take the following two cases into account.

Partially represented relations with Null attributes:

This is the case when an attribute of the semantic relation has not been associated to column of the table schema. For example in the table of Figure 4, the semantic relation *foodAmountLipid*, defined in the ontology on its attributes *food*, *amount* and *lipid*, is partially represented in the table schema *tabSch*, since the attribute *amount* is not represented in the table schema. Figure 6 presents the SML representation of *foodAmountLipid* relation :

```
<table> <content>
<rowRel additionalAttr="yes">
...
<foodAmountLipid relType ="partialNull">
<food attrType="Normal">...</food>
<amount attrType="Null"/ >
<lipid attrType ="Normal"> ... </lipid>
<attribute attrType="generic"> ...</attribute>
</foodLipid>
</rowRel> ... </content> </table>
```

Figure 6: *SML representation of a partially represented relation with Null attributes*

Note that when a relation is partially represented, the attributes that do not appear in the schema are represented in the SML document by means of an empty tag like *<amount **attrType="Null"/>***. In this example, the generic attribute represents precisely the missing attribute *Amount*.

Partially represented relations with constant values:

This is the case when one of the relation attributes correspond to a constant value which appears in the title of the table.

Products	Doubling time (h)
Minced meat	30[1]
Cured raw pork	3.6[1]
Frankfurters	9[1]

Figure 7: *Doubling times of Listeria monocytogenes in foodstuffs*

Let *tabSch* the table schema computed from the table tab_3 of Figure 7: *tabSch*(tab_3)= {*(1,food),(2,factor)*}.

In this table schema, the relation *foodFactorMicroorganism* is partially represented: the attributes *food* and *factor* are represented in the table schema and the attribute *Microorganism* is represented by a constant value *Listeria Monocytogenes* which appears in the table title "Doubling time of ***Listeria Monocytogenes*** in foodstuffs".

This constant is used as a value for the corresponding attribute of the semantic relation and it is propagated into all the instances of the relation. Figure 8 presents the SML representation of the *foodFactorMicroorganism* relation.

Because we want to keep unidentified data, we also add to the semantic relations we have found the set of generic attributes of the table schema. This is done even if the relation is partial. Actually, one of these additional attributes may be

<table>
<content>
<rowRel additionalAttr="no">
...
<foodFactorMicroorganism relType ="partialConst">
<food attrType="Normal">...</food>
<factor attrType="Normal"> ... </factor>
<microorganism attrType ="Const"> listeria monocytogenes
</microorganism>
</foodFactorMicroorganism>
...
</rowRel> ...
</content> </table>

Figure 8: *SML representation of partially represented relations with attributes in constants*

a missing attribute of the relation. Besides, this attribute can add a contextual information which may modify the user's interpretation of the relation.

When no relation has been found in the table schema, a generic relation named *relation* is generated in the SML document. In this way, we keep semantic links between values even if this link has not been identified. Thus, it is possible to query the SML documents by means of lists of key-words.

Instantiation of the semantic relations

Once the relations are extracted, we instantiate them by the values contained in the table. Besides, terms of the ontology are associated with each value when it is possible. The SML formalism allows us to associate several terms that can be found by different mapping mechanisms. We have considered two kinds of mapping procedures.

The first one uses simple syntactic criteria. Each value is considered as a set of lemmatized words M_v where empty words such as determiners or prepositions are suppressed. The same treatment is applied to the terms of the ontology. Then, we consider that there may exist a semantic similarity between a value v and a term t if :

1. equality: ($M_v = M_t$)
2. inclusion: ($M_v \subset M_t$ or $M_t \subset M_v$)
3. intersection: ($M_t \subset M_v$) or ($M_v \cap M_t \neq \emptyset$).

These three criteria are applied using the previous order.

The second mapping procedure uses more semantic criteria. Actually, we have chosen to use the unsupervised approach PANKOW – Pattern-based Annotation trough Knowledge On the Web (Cimiano, Handschuh, & Staab 2004) where patterns are used to categorize proper nouns (instances) with regard to an ontology. PANKOW applies a set of linguistic patterns including Hearst patterns (Hearst 1992) (i.e. the $< concept >< instance >$, $< concept >$ such as $< instance >$, ...) on the biggest corpus available: the World Wide Web. In fact, they exploit the google API and take the number of pages in which patterns appear as an indicator for the strength of the pattern. We have used the same approach on data table even if they are not necessarily proper nouns. We have applied the general pattern "$< value >$ is a $< term >$" in order to discover specialization relations between values and terms of the ontology using the Web corpus. For a given value, we instantiate the pattern with each term of the domain ontology and keep the best term with regard to the number of pages. Because of the specificity of our domain, the number of pages can be very low. For instance, when we try to associate the value "ice cream" to a term of the ontology, the pattern "ice cream is a dessert" is found in 35 pages. Happily, "ice cream is a microorganism" is not found. Note that the term *dessert* cannot be found by our syntactic criteria.

Figure 9 shows a part of the SML document which is automatically generated from the XTab document of Figure 4. This document is structured in the following way:

<table> <table-title>Nutritional Composition of some food products </table-title >
<column-title> Product </column-title> <column-title>Qty</column-title>
<column-title>lipids</column-title>
<column-title>calories</column-title> <column-nb> 4 </column-nb>
<content>
<rowRel additionalAttr="yes">
<foodLipid relType="completeRel">
<food indProc="yes" attrType="Normal">
<ontoVal indMap="intersection"> ***whiting Provencale***
</ontoVal>
<ontoVal indMap="intersection"> ***green lemon*** </ontoVal>
<ontoVal> ***whiting fillets*** </ontoVal>
<originalVal> **whiting with lemon** </originalVal>
</food>
<lipid indProc="no" attrType="Normal">
<ontoVal indMap="notFound"/>
<originalVal> **7.8 g**</originalVal>
</lipid>
<*attribute* indMap="notFound" indProc="no"
attrType="Generic"> <ontoVal/>
<originalVal> **100 g**</originalVal></*attribute*>
</foodLipid>
<foodCalorie relType="completeRel"> ... </foodCalorie>
<foodAmountLipid relType="partialNull"> ...
</foodAmountLipid> </rowRel> ... </content> </table>

Figure 9: *SML Representation of the nutritional composition of food products*

The main part of the document is inside the ***content*** element. It represents the table like a set of lines where each line is now a set of semantic relations (like, for example, ***foodLipide or foodCalories***).
The SML representation of a relation is composed of the set of attributes that appear in the signature of the relation described in the relational Reference Schema of the ontology (e.g. *foodLipid*(*food*, *lipid*)). Each attribute subsumes the representative term of the column or subsumes a term which has been found in its title. A set of terms represented inside the XML tag *ontoVal* is associated with each value. Thus, *crab* has been associated with *ground crab* while three different terms are proposed for *whiting with lemon* : *whiting*

Provencale, *green lemon* and *whiting fillets*. The original value is kept inside the XML tag *originalVal*.

The generality of the SML representation is ensured by the possibility of an automatic generation of the SML DTD from an ontology which contains a taxonomy and a relational reference schema. In the following, we give an example of relational schema and its corresponding SML DTD.

In the figure 10 we present an extract of the relational schema of an ontology of the risk assessment. Figure 11 is the corresponding representation of the SML DTD generated from this relational schema. The DTD is simply represented here as a graph.

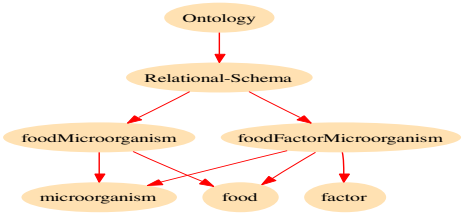


Figure 10: Extract of a risk assessment ontology

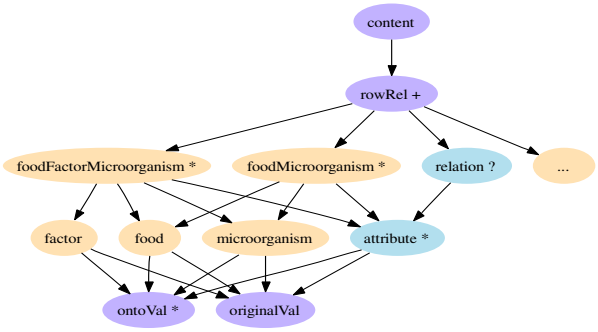


Figure 11: Extract of SML DTD (risk assessment)

Interrogation of SML documents

Some indicators that can be exploited in the queries

Our approach allows one to extract data from tables even if we are not sure of their representation using the vocabulary of the ontology. It is the reason why we have defined a list of indicators that are represented in the SML document and that will be exploited during the query evaluation.

We present now the two main treatment indicators represented in SML as XML attributes attached to lines or to relation attributes. The first one is related to the structure of the relations (presence or absence of additional attributes).

additionalAttr: it informs on the presence of one or several additional attributes that represent the columns of the table which could not be associated with an identified relation. It is added to the tags $< rowRel >$ of SML document. For example in the table of Figure 4, this indicator allows the query engine to use the generic attribute associated with the *Quantity* column.

The following indicator make it possible to specify the kind of mapping procedure used to find a term of the ontology; it can thus be used to evaluate the risk of a mapping error. It is added to the $< ontoVal >$ tags of the SML document.

indMap: it indicates the name of the mapping procedure (inclusion, intersection or PANKOW) used to find the term of the ontology which corresponds to the original value of the table. Several mapping operators can exist in the application, this indicator allows us to modulate a trust degree, relating to enrichment, according to mapping operators. Besides, it can be used to visualize the original value if necessary.

These treatment indicators can be used by the query engine to adapt and find other answers for the user in cases of dissatisfaction.

An example of interrogation

To query SML documents, XQuery queries have been written. They rely on the SML DTD. In the following, we describe a query example where the user looks for the quantity of lipid in 100 g of crab. The evaluation of this query consists in searching in the SML document for the subtrees – SML fragments – such that the parent node is *foodLipid* and such that there is an element *ontoVal* that contains the value "crab". The indicators *indMap* and *indProc* are used to check the validity of the semantic enrichment of the data. As the indicator *additionalAttr* has the value "yes", the query engine displays the additional information *150g*. This example shows how the unidentified attributes that are kept in the SML representation can increase the accuracy of the user interpretation. Besides, the original value *ground crab* is displayed since *indProc* indicates that a treatment was carried out on the original value. The evaluation of this query performed on the document of Figure 9 is presented in Figure 12.

<table>
<title> Nutritional composition of some food products </title>
<food> **ground crab**</food> <lipid>**11.25 g**</lipid>
<validity>inclusion</validity>
<additionalattr>**150 g**</additionalattr>
<category> unknown</category> </table>

Figure 12: *A possible structure of the query answer*

First results

We present in this section the results of the first experimentation of our method. The approach has only be tested on the risk assessment domain represented in the Sym'Previus ontology. In this evaluation, we show the capacity of our system to recognize relations of the ontology in the XTab

tables . Our goal was to compare the results provided with our automatic method with a manual one done by an expert. We compared the results in terms of the well-known information retrieval measures Precision, Recall and F-Measure.

Test set

Among two hundred real XTab tables collected from the Web, we have selected 33 tables. One table is selected in the test set if and only if we identify, among its columns at least one semantic relation attribute represented in the ontology.

Evaluation methodology

In order to evaluate our approach, we have distinguished the results found for the three kinds of semantic relations: the Completely represented Relations (*CR*), the Partially represented Relations where all the missing attributes are identified by Constants in the table title (*PRC*) and the Partially represented Relations which contain at least one attribute which is not identified – Null attributes – (*PRN*). Note that PRC relation can only found in the tables which are associated with a table title.

In first step we run our prototype on the real test set of XTab documents. In second step a domain expert checks the relevance of each semantic relation provided by our system.

To identify the semantic relations represented in the table represented in the XTab document, the expert has access to the whole information of the original – *HTML or Pdf* – document but he only considers information which are contained in the XTab document (ie. the table title and the table content). The expert considers that a semantic relation is *correct* if the relation is represented in the table and if all its attributes are correctly identified. If he recognizes in the XTab document one semantic relation which is not found by our system, he considers that the relation is *forgotten*. By this way, he can determine which semantic relations provided by our system are incorrect and which are forgotten.

In Figure 13, we show the result of this step for each kind of relation (CR, PRC and PRN) : number of semantic relations which have been found by the system, incorrect semantic relations and forgotten semantic relations.

	Found rels	Incorrect rels	Forgotten rels
CR	30	22	11
PRC	6	3	5
PRN	23	2	3

Figure 13: *Expert results after semantic relations checking step*

On these results we have computed Recall, Precision and F-Measure. Let *T, T'* be two variables that represent the semantic relation type considered in the three measures calculations. It gets values in : {CR, (CR and PRN), (CR, PRN and PRC)}. Let *Correct_Rels(*T*)* be the number of semantic relations of type *T*, correctly found by our system.

[1]The XTab tables are the result of an automatic transformation applied on HTML and PDF documents found on the Web

$$Correct\_Rels(T) = Found\_Rels(T) - Incorrect\_Rels(T)$$

Recall is the percentage of relations (all types) actually represented in the data tables and correctly found by our system. Here we suppose that *T'*={*CR, PRN and PRC*}.

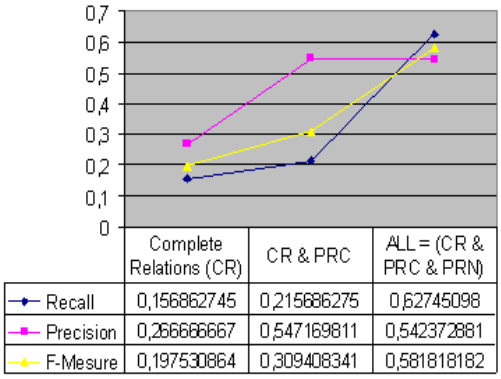
$$Recall = \frac{Correct\_Rels(T)}{(Correct\_Rels(T')) + Forgotten\_Rels(T'))}$$

Precision Is the percentage of relations found in the data tables by our system and with a correctly assigned relation signature.

$$Precision = \frac{Correct\_Rels(T)}{Found\_Rels(T)}$$

F-Measure as usual we balance Recall and Precision against each other.

$$F - Measure = \frac{2 * Recall * Precision}{Recall + Precision}$$



	Complete Relations (CR)	CR & PRC	ALL = (CR & PRC & PRN)
Recall	0,156862745	0,215686275	0,62745098
Precision	0,266666667	0,547169811	0,542372881
F-Mesure	0,197530864	0,309408341	0,581818182

Figure 14: Recall, Precision and F-Measure for a threshold at 0.3

Results

The diagram presented in Figure 14 gives the results in term of precision, recall and F-Measure of our semantic enrichment system approach. The first interesting observation is that the recall value increases significantly when our system takes into account partially represented relations. This result shows clearly the interest of the partially identified semantic relations kept in the SML documents, even when missing attributes are not identified by constants. If we restrict the relation types on the complete relations, we would have only 0.15 for the recall value, whereas in the case where we keep all the identified relations (ie. completely and partially represented relations) we have 0.62 for the recall value. Note that our aim is precisely to obtain a satisfying *Recall* value. Because we have chosen to keep all the identified pieces of information as well as information which are not completely identified such as partial relations, generic attributes end partial relations. We can also note that the precision is increasing as well. This result is globally shown by the increasing of the F-Measure value.

Conclusion

Our method allows one to enrich semantically documents found on the Web which present the specificity of a tabular structuring. The semantic enrichment is completely automatic and it is guided by an ontology of the domain. Thus, that processing cannot lead to a perfect and complete enrichment. The XML representation we propose keeps all the possible interpretation in order to let the possibility of using them during the query step, for example by allowing a query processing based on keywords or by exhibiting some relevant information to the user in order to help him/her during the interpretation of the results.

Then, in case of ambiguity, it is possible to associate several terms of the ontology or several semantic relations with a same set of columns. In order to allow the query processor to adapt its answers or to evaluate their relevance, we log the processes by means of a set of indicators. The generality of our approach is ensured by the fact that the SML DTD can be automatically generated from the ontology.

The approach we propose is currently under testing in the domain of the food risk assessment, by means of a Java prototype. In order to query SML documents, we wrote *XQuery* queries which take advantage of the treatment indicators inserted in the SML documents. Those queries have been tested by means of the MIEL++ query engine.

Some works like (Kushmerick 2000), (Muslea, Minton, & Knoblock 2001) and (Hsu & Dung 1998) allow to extract knowledge by learning rules from a sample of manually annotated documents. Our goal is quite different since our approach is completely automatic and exclusively guided by the ontology.

Moreover, the documents we use to fill the data warehouse are heterogeneous and, contrarily to previous approaches like (Crescenzi, Mecca, & Merialdo 2002) and (Arasu & Garcia-Molina 2003), we cannot base the search for information on a common structure discovered among a set of homogeneous documents.

The techniques we use to identify the columns of the table are based first on the values contained in those columns. (Rahm & Bernstein 2001) and (Doan *et al.* 2003) showed that those techniques give good results in the framework of the search for schema mappings for relational databases or XML. In our case, we do not have the schema of the tables we work on: we have to discover it first before searching for mappings with the semantic relations of the ontology.

We can now enhance our mapping operators, for example by using external resources such as WordNet or by using more sophisticated similarity measures (Robertson & Willett 1998). Moreover, we can think about using linguistic tools allowing to process the table content (cells, titles) represented in a more complex way. We also want to check the generality of our approach by applying it to another application domain.

References

Arasu, A., and Garcia-Molina, H. 2003. Extracting structured data from web pages. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, 337–348. ACM Press.

Buche, P.; Dibie-Barthélemy, J.; Haemmerlé, O.; and Houhou, M. 2004. Towards flexible querying of xml imprecise data in a dataware house opened on the web. In *Flexible Query Answering Systems (FQAS)*. Springer Verlag.

Cimiano, P.; Handschuh, S.; and Staab, S. 2004. Towards the self-annotating web. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, 462–471. ACM Press.

Crescenzi, V.; Mecca, G.; and Merialdo, P. 2002. Automatic web information extraction in the roadrunner system. In *Revised Papers from the HUMACS, DASWIS, ECOMO, and DAMA on ER 2001 Workshops*, 264–277. Springer-Verlag.

Doan, A.; Lu, Y.; Lee, Y.; and Han, J. 2003. Profile-based object matching for information integration. *Intelligent Systems, IEEE* 18(5):54– 59.

e.dot. 2004. Progress report of the e.dot project. http://www-rocq.inria.fr/gemo/edot.

Hearst, M. A. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics*, 539–545. Association for Computational Linguistics.

Hsu, C.-N., and Dung, M.-T. 1998. Generating finite-state transducers for semi-structured data extraction from the web. *Inf. Syst.* 23(9):521–538.

Kushmerick, N. 2000. Wrapper induction: efficiency and expressiveness. *Artif. Intell.* 118(1-2):15–68.

Mezaour, A. D. 2005. Filtering Web Documents for a Thematic Warehouse, case study : eDot a Food Risk Data Warehouse (extended). In *to Appear in Proceedings of New Trends in Intelligent Information Processing and Web Mining Conference (IIPWM'05), Gdansk, Poland*. Springer Verlag series–Advances in Soft Computing–.

Muslea, I.; Minton, S.; and Knoblock, C. A. 2001. Hierarchical wrapper induction for semistructured information sources. *Autonomous Agents and Multi-Agent Systems* 4(1-2):93–114.

Pivk, A.; Cimiano, P.; and Sure, Y. 2004. From tables to frames. In *International Semantic Web Conference*, 166–181.

Rahm, E., and Bernstein, P. A. 2001. A survey of approaches to automatic schema matching. *The VLDB Journal* 10(4):334–350.

Robertson, A., and Willett, P. 1998. Applications of n-grams in textual information systems. In *Journal of Documentation*, 48–69.

http://www.symprevius.net.