Fuzzy annotation of web data tables driven by a domain ontology

Gaëlle Hignette, Patrice Buche, Juliette Dibie-Barthelemy, Ollivier Haemmerlé

Fuzzy annotation of web data tables driven by a domain ontology

Gaëlle Hignette, Patrice Buche, Juliette Dibie-Barthélemy, and Ollivier Haemmerlé

INRA/AgroParisTech Unité Mét@risk - Université de Toulouse le Mirail,
16 rue Claude Bernard, F-75231 Paris Cedex 5, France
{hignette, buche, dibie}@agroparistech.fr, ollivier.haemmerle@univ-tlse2.fr

Abstract. We propose an automatic system for annotating accurately data tables extracted from the web. This system is designed to provide additional data to an existing querying system called MIEL, which relies on a common vocabulary used to query local relational databases. We will use the same vocabulary, translated into an OWL ontology, to annotate the tables. Our annotation system is unsupervised. It uses only the knowledge defined in the ontology to automatically annotate the entire content of tables, using an aggregation approach: first annotate cells, then columns, then relations between those columns. The annotations are fuzzy: instead of linking an element of the table with a precise concept of the ontology, the elements of the table are annotated with several concepts, associated with their relevance degree. Our annotation process has been validated experimentally on scientific domains (microbial risk in food, chemical risk in food) and a technical domain (aeronautics).

1 Introduction

The web is usually modeled as a set of unstructured documents interconnected by a hyperlink graph. However, those documents contain a significative amount of relational data stored in tables. Those tables can be seen as small relational databases even if they lack the explicit metadata associated with a database. Those databases are of course very interesting potential external sources to feed the data warehouse of a company, dedicated to a given domain of application. In general, those data warehouses are already composed of local relational databases fed with internal data belonging to the company. The aim of the company, retrieving external data from the Web, is to complement local data and/or to compare external data to local ones. In order to realize this objective, it is necessary to integrate external data into local data. One possibility is to index external data using the vocabulary already used to index local data. This vocabulary can be easily built from explicit metadata associated with relational local databases, which correspond to the relational schemas of the databases and their attributes with their associated domains. This approach has been experimentally tested on three different domains (microbial risk in food, chemical risk in food and aeronautics): three OWL ontologies have been created within a

couple of hours thanks to preexisting information retrieved from local databases and a very simple tool which translates automatically csv files containing the metadata into an OWL ontology.

In this paper, we present a method to annotate automatically and accurately Web data tables with a domain ontology expressed in OWL. This method relies on heuristics which uses the relations signatures, the symbolic and numeric attributes names, associated domains and constraints (set of symbolic possible values for the symbolic attributes, interval of numeric values and units for the numeric attributes) of the ontology and can be compared with works in schema matching (see [1] for a review). Our original contribution in this paper compared to such works is threefold. Firstly, we instanciate accurately the recognized relations of the ontology for each row of the Web data table. By accurate, we mean that each cell of the row which corresponds to a symbolic (or numeric) attribute is associated with a set of symbolic similar values (or with an interval of possible numeric values) of the ontology. Thanks to this annotation process, it is possible, using the MIEL querying system (see [2]) and the ontology, to query simultaneously the local relational databases and the annotated Web tables thanks to adapted wrappers. Secondly, the heuristics used by our annotation method are based on a set of criteria which can be easily extended. Thirdly, in our approach, thanks to the ontology easily built from preexisting metadata, we don't need to use matching learning methods which imply to learn manually the classifiers on a part of the corpus of documents (see [3] by example). As we will present in this paper, we obtain experimental results which are similar to machine learning methods to identify the type of symbolic columns. This method has been applied with success in the three following different application fields: microbial risk in food, chemical risk in food and aeronautics.

Recent propositions in the Semantic Web community propose to extract, filter, annotate and query Web data tables (see [4–6]), but they have not been designed with the same objectives. TableSeer (see [4]) for instance permits to extract a set of predefined metadata (caption, cell content, geographical position of the table in the HTML page, ...) from Web tables, but it does not compare the schema of the Web tables to preexisting schemas defined in a ontology. We can also cite WebTables (see [5, 6]) which proposes a system to identify relational tables in the huge amount of tables included in HTML documents, index them to query and rank them. Nevertheless, the WebTables querying language is only composed of a set of key-words which are compared to the attribute names of the Web tables. The row content of the Web tables is not used in the querying process which is only based on global co-occurences frequencies statistics of attribute names. Our approach can be compared to the construction of frames from tables described in [7] but they use a generic ontology and create new relations according to the table signature, whereas we want to recognize predefined relations in an ontology specific to the target domain. In [8], relations from an ontology are instanciated using various HTML structures including tables. However, they only identify binary concept-role relations between instances that are assumed to be already annotated (manually or using another information

extraction system). Our work differs as we focus on the recognition of n-ary relations and we propose a step-by-step algorithm including the recognition of element types. From this point of view, the work presented in [9] is nearer to ours, as they transform tables of different structures into a common relational database schema with n-ary relations. However, their approach depend upon the manual definition of an "extraction ontology" that defines extraction rules for each set of objects, giving all synonyms for an attribute name or defining the context of apparition of a string to extract. Our work differs as the ontology is built independently from the annotation process.

Our annotation process is divided into several steps: first, find documents on the web that are relevant to the application domain by regularly crawling sites that are specialized in the domain of interest (for example, JIFSAN[1] or EFSA[2] for risk in food purposes) or even by accessing the "hidden web" via subscription using alerting systems such as RSS feeds (for example on a site like Web Of Knowledge[3]); second, extract the tables from the documents; third annotate the tables; and finally, query the tables using an extended version of the MIEL system (see [2]). The three first steps are implemented in the @Web software, developed using the standards of the WebContent platform[4]. The Web data tables are stored in an XML document using the predefined data model of the WebContent platform which permits to exchange data between web services available on the platform. The annotations generated by the Web service are associated with the tuples of the Web data table and expressed in RDF.

This paper will focus on the third step of the @Web software: the automatic annotation of Web data tables. The structure and OWL representation of our ontology is presented in section 2. We then describe the annotation process: section 3 presents how we recognize the semantic relations represented by a table, and section 4 shows how we create the actual annotations for the table.

2 The ontology

In the current MIEL system, when a user ask a query on a database, he/she can define the selection criteria according to a predefined vocabulary. We have brought together this vocabulary and the domain information available within the databases schemas to create an ontology. The given examples come from the ontology of the food microbiology domain, but the structure we propose is generic and can be applied to many other domains such as chemical risk in food or aeronautics.

2.1 The ontology structure

The structure of our ontology relies on the structure of database schemas and is organised into numeric types, symbolic types and relations.

[1] Joint Institute for Food Safety and Applied Nutrition, http://www.jifsan.umd.edu
[2] European Food Safety Authority, http://www.efsa.europa.eu
[3] http://www.isiwebofknowledge.com/
[4] http://www.webcontent.fr

Symbolic types are described by a type name, a list of synonyms for the type name and a taxonomy of possible values. In our ontology on the food microbiology domain, the type *Food Product* is associated with a taxonomy of more than 500 food products, the type *Microorganism* with a taxonomy of more than 150 microorganisms and the type *Response* is a flat list of the possible responses of a microorganism to a treatment: growth, absence of growth or death.

Numeric types are described by a type name, a list of synonyms for the type name and the set of units in which the type can be expressed and eventually a numerical range. Our ontology on food microbiology contains 18 numeric types, for example *Temperature* which can be expressed in °C or °F and has no numerical range, or *pH* which has no unit and is restricted to the range $[0, 14]$.

Relations are described by the name of the relation and its signature. The signature of a relation is divided into one result type (the range of the relation) and several access types (the domain of the relation). Our ontology on food microbiology contains 16 relations. For example, in the relation *Growth kinetics*, the result type is *Microorganism concentration* and the access types are *Microorganism*, *Food product*, *Temperature* and *Time*.

The names of types and relations, as well as the possible values of a symbolic type defined in its taxonomy, are called terms. Those terms will be used to annotate Web tables extracted from the web.

2.2 The ontology in OWL format

The OWL representation of the ontology is divided in two parts. First, the definition of the structure of the ontology is domain independent: what is a symbolic type, a numeric type or a relation and what are the properties of these different objects. Then, the second part of the ontology presents the definition of the actual types and relations of the domain.

We have separated in the OWL representation of the ontology the concepts (i.e. the types, the relations and the elements of the taxonomies) and the vocabulary (i.e. the actual terms with their words). Any concept in the ontology can be linked to its corresponding term via a property *AssociatedTerm*. The taxonomy of a symbolic type is viewed as a hierarchy of subclasses from a generic OWL class *Taxonomy*: the symbolic type is associated to the root of its hierarchy via the property *HasForTaxonomy*. Each concept in the hierarchy (except the taxonomy root) is associated with its corresponding term. Relations in the ontology are n-ary. They are thus represented as advised by [10], a relation being an OWL class associated to the types of its signature via the properties *AssociatedKey* (for the access types) and *AssociatedResult* (for the result type). The complete OWL definition of our ontology is available for download at `http://metarisk.inapg.inra.fr/UserFiles/File/Gaelle/onto.owl`.

3 Recognising relations represented in a table

Given the ontology and given a table extracted from the web, we want to find which semantic relations described in the ontology are represented in the table.

For that purpose, we use an aggregation approach, first looking at the contents of the cells, then computing the type of the columns and finally comparing the signature of the table (the column types) with the signatures of the relations in the ontology. We first distinguish between symbolic and numeric columns, using some of the knowledge described in the ontology (mainly the units): for better description of this step, please refer to [11] which is a preliminary version of this work. We present here how we find the type of symbolic then numeric columns, and finally the relations represented in the table.

3.1 Finding the type of a symbolic column

When a column contains symbolic data, we want to find which symbolic type of the ontology corresponds to the contents of the column. For that purpose, we compute a score of each symbolic type of the ontology for the column. This score is a combination of the score of the type for the column according to the column contents and the score of the type for the column according to the column title. Both these scores use the notion of term similarity: let a and b be two terms, represented as weighted vectors $(a_1, \ldots, a_n)$ and $(b_1, \ldots, b_n)$, the coordinate axis corresponding to the lemmatised words, the coordinate values corresponding to the weight of the word in the term (0 if the word is not part of the term); the term similarity between a and b is the cosine similarity [12] between the two vectors.

$$sim(a, b) = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2 \times \sum_{i=1}^{n} b_i^2}} \quad (1)$$

Each word, of terms from the web and in the ontology, is given a weight of 1 (except for stopwords such as articles or prepositions, as well as words that contain only one letter, which are given a weight of 0). The score of a symbolic type t for a column col according to the column title, noted $score_{title}(t, col)$, is the term similarity between the type name and the column title.

To compute the score of a symbolic type for the column according to the column contents, we first explore each cell of the column (excluding the title). For each cell of the column, we compute the score of each symbolic type for the cell as the sum of the term similarities between the content of the cell and the terms in the taxonomy of the symbolic type. Let $cell$ be the term in a cell of the column and t be a symbolic type, with $taxo(t)$ the set of terms in the taxonomy of possible values associated with t. Then the score of type t for the cell $cell$ is:

$$score(t, cell) = \sum_{x \in taxo(t)} sim(cell, x) \quad (2)$$

For each cell of the column, we compute the proportional advantage of the type having the best score: let $best$ be the type with the best score and $secondBest$ be the type with the second best score. We compute the proportional advantage of $best$ for the cell $cell$:

$$advantage(best, cell) = \frac{score(best, cell) - score(secondBest, cell)}{score(best, cell)} \quad (3)$$

The cell is affected the type *best* if its proportional advantage is higher than a specified threshold (for our experiments, this threshold is set to 10%). Otherwise, the cell is considered as having an unknown type.

Once all cells in the column are affected a type, we compute the score of a type for the column according to the column contents as the proportion of cells in the column having this type. Let *col* be a symbolic column with n_{col} the number of cells in the column (excluding the column title), t a symbolic type with $n(t, col)$ the number of cells in the column having the type t, then

$$score_{contents}(t, col) = \frac{n(t, col)}{n_{col}} \tag{4}$$

The score of a symbolic type t for the column *col* is then computed as a combination of the score from the title and the score from the contents:

$$score_{final}(t, col) = 1 - (1 - score_{title}(t, col))(1 - score_{contents}(t, col)) \tag{5}$$

The type of the column is the type having the best final score for the column, assuming that its proportional advantage (see equation 3, replacing the score for the cell by the final score for the column) is greater than a specified threshold (in our experiments, this threshold was set to 10%). Otherwise the column is considered as having an unknown type.

We have experimented our process on 81 columns containing symbolic data, extracted from tables that appeared in publications on food microbiology. Those columns were manually classified in one of the three symbolic types of our ontology, or *Other* if it corresponded to data not modeled in the ontology. Then we ran our process to automatically classify the columns according to their symbolic type. The columns that were of unknown type at the end of our type recognition process were considered as belonging to the *Other* category. In order to assess the quality of our results, we wanted to compare our method with a machine learning classifier. To our knowledge, there is no classifier that is dedicated to the classification of symbolic data using a domain ontology. We have compared our process to the SMO classifier [13]: as it is an optimized version of the well-known SVM, it allows comparing our results with a machine learning method. It is thus a comparison between two alternatives: SMO uses no domain knowledge *but* uses learning, while our method is based on domain knowledge *but* has no learning phase. For the SMO classifier, we used the following pre-treatment: each distinct lemmatized word present in a column results in an attribute; the value of this attribute for a given column is the frequency of the word in the column. The SMO classifier was evaluated using a leave-one-out cross-validation, with default parameters of the Weka[5] implementation. Results of this experiment are given in Table 1.

Over the three symbolic types of the ontology, we obtain a precision of 89% and a recall of 81% with our annotation method, while the SMO classifier gives a 84% precision and a 90% recall. Our method, which uses domain knowledge but no learning phase, gives similar results to the learning classifier, with a slightly

[5] http://www.cs.waikato.ac.nz/ml/weka

manual \ computed	our method using the ontology				SMO			
	Food	Micro.	Resp.	Other	Food	Micro.	Resp.	Other
Food	34	0	0	12	45	0	0	1
Micro.	0	16	0	0	4	12	0	0
Response	0	0	1	0	0	0	0	1
Other	3	3	0	12	7	0	0	11

Table 1. Classification results on 81 symbolic columns.

better precision and lower recall. This is mainly because we have biased our annotation technique towards precision: whenever we do not know for sure the type of a column, it is considered as unknown.

Our process has been also experimented on two other corpora: chemical risk in food whose ontology contains 3 symbolic types and aeronautics whose ontology contains 4 symbolic types. Over the 22 columns of the chemical risk in food corpus containing symbolic data, we obtain a precision of 100% and a recall of 100% with our annotation method. Over the 46 columns of the aeronautics corpus containing symbolic data, we obtain a precision of 82% and a recall of 100% with our annotation method.

3.2 Finding the type of a numeric column

As it has been done for the symbolic columns, we find the type of a numeric column by computing the score of each numeric type in the ontology for the column, this score being a combination of the score of the type for the column according to the units in the column and the score of the type for the column according to the column title.

The score of a numeric type t for a column *col* according to the column title, noted $score_{title}(t, col)$ is the term similarity between the type name and the column title.

To compute the numeric type for the column according to the units in the column, we first compute the score of the numeric type for each unit that is present in the column: a numeric type has a score for each unit, depending on the number of numeric types that can be expressed in this unit. Let u be a unit and T_u the set of numeric types for which this unit has been declared in the ontology, the score of a type t for the unit u is $score(t, u) = 0$ if $t \notin T_u$, and $score(t, u) = \frac{1}{|T_u|}$ if $t \in T_u$.

The score of a numeric type for the column is computed as the maximum of the scores of the type for each unit present in the column. If no unit from the ontology was identified in the column, then the column is considered as presenting the unit "no unit", which is treated as a normal unit in the ontology.

The final score of a numeric type t for the column *col* is a combination of the score of t for *col* according to the title of the column ($score_{title}(t, col)$), and the

score of t for *col* according to the units in the column ($score_{unit}(t, col)$). However, types are filtered according to the numeric values presented in the column:

- if the column contains a numeric value outside the range of values for the type t, then $score_{final}(t, col) = 0$.
- if all values in the column are compatible with the range of type t, then

$$score_{final}(t, col) = 1 - (1 - score_{title}(t, col))(1 - score_{unit}(t, col)) \quad (6)$$

The type of the column is then the type having the best final score for the column, assuming that its proportional advantage is greater than a specified threshold (in our experiments, this threshold was set to 10%). Otherwise the column is considered as having an unknown type.

We have experimented our method of numeric column annotation on 261 columns containing numeric data extracted from tables found in publications on food microbiology. The columns were manually annotated with the 18 numeric types of our domain ontology, and we compared the manual annotation with the types computed by our method. On the 261 columns, 243 were correctly annotated, 9 were considered as unknown and 9 were annotated with a wrong type (i.e. 96% precision, 93% recall). By comparison, when considering the score from title as the final score with no use of the units defined in the ontology (as was done in [14]), precision was 96% but recall was only 83%: the use of the units defined in the ontology allows better recall with no more errors.

Our process has been also experimented on two other corpora: chemical risk in food whose ontology contains 5 numeric types and aeronautics whose ontology contains 29 numeric types. Over the 65 columns of the chemical risk in food corpus containing numeric data, we obtain a precision of 100% and a recall of 85% with our annotation method. Over the 114 columns of the aeronautics corpus containing numeric data, we obtain a precision of 96% and a recall of 86% with our annotation method.

3.3 Finding the relations represented in the table

Once all columns of the table are assigned a type, we want to find the semantic relations that are represented in the table. For that purpose, we compute a score of each relation of the ontology for the table. This score is a combination of a score of the relation for the table according to the table title and a score of the relation for the table according to the table signature.

The score of a relation for the table according to the table title is computed as the term similarity between the table title and the relation name.

The score of a relation r for the table *tab* according to the table signature is computed as follows:

- if the result type of the relation r was not recognized as a type of a column of the table, then $score_{signature}(r, tab) = 0$
- else, the score of the relation for the table is the proportion of types in its signature that were recognized in the table columns. Let $Sign(r)$ be the set

of types in the signature of relation r and $Sign_{tab}$ the set of types that were recognized for the table columns, then

$$score_{signature}(r, tab) = \frac{\mid Sign(r) \cap Sign_{tab} \mid}{\mid Sign(r) \mid} \quad (7)$$

Then the final score of a relation r for the table tab is computed as:

$$score_{final}(r, tab) = 1 - (1 - score_{title}(r, tab))(1 - score_{signature}(r, tab)) \quad (8)$$

When the scores of all relations of the ontology have been computed for the table, we choose the relation(s) to keep for the table. A table can represent several relations at a time: for example, if a table gives the pH and the water activity of a food product, we will consider it as two separate relations: *food pH* and *food water activity*. We define the notion of concurrent relations: two relations are called concurrent if they have the same result type. If a relation has a non-zero score for the table and has no concurrent relation, this relation is considered as represented in the table. If there are several concurrent relations with non-zero scores for the table, then we only keep the one with the highest score for the annotation of the table. If several concurrent relations have the same highest score, we keep them all.

We have experimented our method on 60 tables extracted from publications on food microbiology. Those tables were manually annotated with the 16 relations of the ontology: one table was typically annotated with 1 to 5 relations, which gives a total of 123 relations (the manually annotated tables are available at `http://metarisk.inapg.inra.fr/UserFiles/File/Gaelle/tables.zip`).

We ran the different steps of our relation recognition system without validating the intermediate steps, i.e. even columns that were wrongly recognized were further annotated and used for the relation annotation. Over the 123 relations in the manual annotation, 119 were correctly recognized in our process, 4 were not recognized and there were 30 relations that were kept for the tables while they should not have been recognized. This gives a precision of 80% for a recall of 97%. The relation recognition is biased towards recall, as we accept partially recognized relations (only the result type is mandatory): it often happens that we recognize two different binary relations for one result type (for example *Food property: pH* and *Growth parameter: pH*) — one of the two relations is the one represented in the table, the other is false.

Our process has been also experimented on two other corpora: chemical risk in food whose ontology contains 4 relations and aeronautics whose ontology contains 26 relations. Over the 34 relations in the manual annotation of the chemical risk in food corpus, 27 were correctly recognized in our process, 7 were not recognized and there were 2 relations that were kept for the tables while they should not have been recognized. This gives a precision of 93% for a recall of 79% with our annotation method. Over the 113 relations in the manual annotation of the aeronautics corpus, 100 were correctly recognized in our process, 13 were not recognized and there were 2 relations that were kept for the tables while they should not have been recognized. This gives a precision of 98% for a recall of 88% with our annotation method.

4 Instanciating relations and annotating the tables

To annotate the tables, we annotate each row of the table with an instance of each relation that was recognized for the table. These instances of relations are linked to the instances of numeric and symbolic types of their signature, corresponding to the data available in the table row. The annotations we generate are fuzzy: they allow one to take into account the imprecision of the initial data in the table (for example an interval for a numeric type) as well as the problems of matching between the vocabulary of the table and the vocabulary of the ontology, and the uncertainty of the recognition of relations. We first present briefly the theory of fuzzy sets that we use for our annotations, then we present how we instanciate numeric types, symbolic types and relations.

4.1 Fuzzy sets

We use the definition of fuzzy sets given by [15, 16]. The notion of fuzzy set is an extension of classical subsets. In the classical case, elements of a reference set X that have some properties belong to a subset A, and elements that do not have these properties belong to the complementary subset of A in X. In a fuzzy set, elements can belong partially to the fuzzy set, with a membership degree comprised between 0 (element which is not part of the fuzzy set) and 1 (element which is completely part of the fuzzy set). The membership degree of an element $x \in X$ for the fuzzy set A is noted $\mu_A(x)$. When X is a continuous domain, we talk about a continuous fuzzy set; if X is discrete, we talk about a discrete fuzzy set.

The support of a fuzzy set A defined on a reference set X is the set (in the classic definition) of elements $x \in X$ so that $\mu_A(x) > 0$. The kernel of a fuzzy set A defined on a reference set X is the set (in the classic definition) of elements $x \in X$ so that $\mu_A(x) = 1$. A trapezoid fuzzy set TFS is a special continuous fuzzy set that is described only by its support $sup = [min_{sup}, max_{sup}]$ and its kernel $ker = [min_{ker}, max_{ker}]$. The membership degree of a numeric value x in the reference set, is then defined as follows:

- if $x \leq min_{sup}$ or $x \geq max_{sup}$ then $\mu_{TFS}(x) = 0$;
- if $min_{ker} \leq x \leq max_{ker}$ then $\mu_{TFS}(x) = 1$;
- if $min_{sup} \leq x \leq min_{ker}$ then $\mu_{TFS}(x) = \frac{x - min_{sup}}{min_{ker} - min_{sup}}$;
- if $max_{ker} \leq x \leq max_{sup}$ then $\mu_{TFS}(x) = \frac{x - max_{ker}}{max_{ker} - max_{sup}}$.

There are several semantics for fuzzy sets, defined in [17]:

- preferences: the elements with the higher membership degrees are the prefered elements. This is used in the MIEL querying system for the user to define query preferences;
- uncertainty or imprecision: there exists a "true" value, but we do not know it. The higher is the membership degree of a value x, the more probable is x to be the "true" value. This is used in our annotations to represent the imprecision of the original data in the tables (in the instanciation of numeric types);

– similarity: a new object is represented by its similarity with known objects. The higher is the membership degree of a known object x, the more it is similar to the new object. This is used in our annotations to represent the similarity between a term from the web and terms from the ontology.

4.2 Instanciating numeric types

When instanciating a numeric type t in the signature of a relation that has been recognised for the table, there are three possibilities:

1. There is one column in the table (thus one cell in the row to annotate) that has been recognized as having the numeric type t. In this case, the values in the cell are used to instanciate the type: it can be an isolated value, an enumeration of isolated values, an interval or a mean with a standard error. Intervals and mean with standard error are recognized using specific patterns; if those patterns are not recognized, then all numeric values in the cell are considered as isolated.
2. There are several columns in the table that have been recognized as having the numeric type t. In this case, we have to find the relations between the columns: it is done by looking for keywords in the column title. A column can represent a minimum value, a maximum value or an optimum value (comprised between the minimum and maximum); it can also represent a mean value or a standard error.
3. There is no column in the table that was recognized as having the type t. If the numeric type t has a defined unit, we search for occurences of a numeric value followed by this unit, in the title of the table or in the titles of columns: those occurences are then considered as isolated values.

An instance of a numeric type is represented as a fuzzy set, the reference set being the value range defined in the ontology for the numeric type. The fuzzy set used for annotation is a union of trapezoid fuzzy sets. The trapezoid fuzzy sets used for the instanciation of a numeric type are constructed as follows:

– when recognizing an isolated value x in the table, we construct a trapezoid fuzzy set with $sup = ker = [x, x]$;
– when recognizing an interval $[a, b]$ in the table, either in one cell or when a is the value in a cell recognized as minimum and b is the value in a cell recognized as maximum with no cell recognized as optimum, we construct a trapezoid fuzzy set with $sup = ker = [a, b]$;
– when recognizing a cell as minimum, its minimum numeric value being min, a cell as maximum, its maximum numeric value being max and a cell as optimum, its values being comprised in the minimum interval $[a, b]$, we construct a trapezoid fuzzy set with $sup = [min, max]$ and $ker = [a, b]$;
– when recognizing a mean m and a standard error e, we construct a trapezoid fuzzy set with $sup = [m - e, m + e]$ and $ker = [m, m]$.

Once all trapezoid fuzzy sets are created, the annotation is constructed as the union of all those sets (for example, there can be a union of several isolated values).

Example 1. Table 2 is annotated with the relation of the ontology *Growth parameter: pH*, with the access type *Microorganism* and the result type *pH*. This result type is instanciated, for the first row of data in the table, as a unique trapezoid fuzzy set with $sup = [5, 8.8]$ and $ker = [6, 7]$.

Species	pH Min	pH Opt.	pH Max
Bacillus cereus	5	6–7	8.8

Table 2. Excerpt of a table: Ecologic values for some foodborne bacterial pathogens (extracted from [18])

Below is the RDF representation of the fuzzy set used for the instanciation of type *pH* in the first data row of Table 2.

```
<onto:CFS rdf:about="Row:2/GrowthParameterPH/PH/CFS">
   <onto:IsComposedOf rdf:resource="Row:2/GrowthParameterPH/PH/CFS/TFS:1"/>
</onto:CFS>
<onto:TrapezoidFuzzySet rdf:about="Row:2/GrowthParameterPH/PH/CFS/TFS:1">
   <onto:HasForMinSupport>5</onto:HasForMinSupport>
   <onto:HasForMaxSupport>8.8</onto:HasForMaxSupport>
   <onto:HasForMinKernel>6</onto:HasForMinKernel>
   <onto:HasForMaxKernel>7</onto:HasForMaxKernel>
</onto:TrapezoidFuzzySet>
```

To evaluate our annotation system, we have instanciated the 119 relations that were correctly recognized in the 60 tables used for the experiment on relation recognition. The instanciation of numeric values was analyzed for the first data row of each table: we assume that the structure is homogeneous inside a table, so the instanciation of the first row is representative of what happens for the whole table. On the 119 relations, there were 2 errors on the extraction of numeric values (one was an error of type recognition, one was an error of numeric value recognition). For 5 tables (corresponding to 13 relations), the numeric type *Temperature* was not instanciated because its value was not present in the table but in the textual environment of the table in the original publication. There also were 3 errors in interval reconstruction (values were considered as isolated while it was an interval) and one error in the construction of a minimum/optimum/maximum trapezoid fuzzy set (values were considered as isolated). For all 100 remaining relations, all numeric values were correctly instanciated.

4.3 Instanciating symbolic types

To instanciate a symbolic type t in the signature of a relation that has been recognized for the table, we construct a fuzzy set. The reference set of this fuzzy set is the set of all terms in the taxonomy of the type. The membership degree of a term x in the annotation fuzzy set is the term similarity between x and the term in the cell that has been recognized as having type t. In our 60 tables, it did not happen that several columns in a table were recognized as having the

same type t, however, would that happen, we would construct a union of fuzzy sets (one fuzzy set for each column).

Below is the RDF representation of the fuzzy set used for the instanciation of type *Microorganism* in the first data row of Table 2 (we only represent the elements of the taxonomy for which the membership degree is not equal to 0).

```
<onto:DFS rdf:about="Row:2/GrowthParameterPH/Microorganism/DFS">
  <onto:HasForElement rdf:resource="Row:2/GrowthParameterPH/Microorganism/DFS
      /elt:1"/>
  <onto:HasForElement rdf:resource="Row:2/GrowthParameterPH/Microorganism/DFS
      /elt:2"/>
</onto:DFS>
<onto:BacillusCereus rdf:about="Row:2/GrowthParameterPH/Microorganism/DFS/elt
    :1">
  <onto:HasForMembershipDegree>1.0</onto:HasForMembershipDegree>
</onto:BacillusCereus>
<onto:Bacillus rdf:about="Row:2/GrowthParameterPH/Microorganism/DFS/elt:2">
  <onto:HasForMembershipDegree>0.7</onto:HasForMembershipDegree>
</onto:Bacillus>
```

The quality of the annotation has been evaluated on 185 instances of food products. For those food products, the "best match" in the ontology (i.e. the term in the ontology that was the nearest to the meaning of the term in the table) was manually defined. The evaluation is done by looking at the position of the "best match" in the automatic annotation, by order of descending membership degree. The position is evaluated at worse, i.e. if there are several terms in the ontology having the same membership degree in the fuzzy set used for the annotation, the "best match" is always considered as being in last position. This evaluation at worse is due to the need for manual validation of the annotations: if we present a user with the 5 terms having the best membership degree for him to choose the best, we want to be sure that the "best match" will be among those 5. On the 185 terms from the web, 78% had a "best match" for which the term similarity with the term from the web was not null. 46% of the terms from the web had their "best match" in first position in the computed annotation, while 66% had their "best match" among the five best positions. This validates the approach of keeping a fuzzy set for instanciating the symbolic types, instead of keeping only the concept in the taxonomy having the best term similarity with the term in the table.

4.4 Representing the relations in the table annotation

Once all types of the signature of a relation have been instanciated for a row of the table, we can annotate the row with the relation: we create an instance of the relation, which is related to a degree of certainty (the score that has been computed during the relation recognition phase) via the property *HasForScore*, and is related to the instances of numeric and symbolic types of its signature that were created for the row.

Below is the annotation of the first data row of Table 2, given the DFS and CFS that were already defined during the intanciation of symbolic and numeric types.

```
<onto:GrowthParameterPH rdf:about="Row:2/GrowthParameterPH" >
   <onto:HasForScore>1.0</onto:HasForScore>
   <onto:AssociatedKey rdf:resource="Row:2/GrowthParameterPH/Microorganism"
       />
   <onto:AssociatedResult rdf:resource="Row:2/GrowthParameterPH/PH" />
</onto:GrowthParameterPH>
<onto:Microorganism rdf:about="Row:2/GrowthParameterPH/Microorganism">
  <onto:IsConstructedFrom rdf:resource="Row:2/Text:1"/>
  <onto:IsAnnotatedBy>
    <onto:DFS rdf:resource="Row:2/GrowthParameterPH/Microorganism/DFS"/>
  </onto:IsAnnotatedBy>
</onto:Microorganism>
<onto:PH rdf:about="Row:2/GrowthParameterPH/PH">
  <onto:IsConstructedFrom rdf:resource="Row:2/Text:2"/>
  <onto:IsConstructedFrom rdf:resource="Row:2/Text:3"/>
  <onto:IsConstructedFrom rdf:resource="Row:2/Text:4"/>
  <onto:IsAnnotatedBy rdf:resource="Row:2/GrowthParameterPH/PH/CFS" />
</onto:PH>
```

5 Conclusion and perspectives

In this paper we have described a complete annotation system to annotate data tables extracted from the web. It is implemented in the @Web software which has been developed using the standards of the WebContent platform. In @Web, the workflow is divided into three steps: first, find documents on the web that are relevant to the application domain; then, extract the tables from the documents; finally annotate the tables. @Web proposes very efficient and powerful tools which permit to transform semi-automatically the structure of a Web data table before its automatic annotation using a domain ontology. To the best of our knowledge, @Web is the only software which permits first to deal efficiently with the heterogeneity and the complexity of the table structures available on the Web and second to annotate accurately a Web table with a domain ontology. This annotation entirely relies on the domain knowledge given in the ontology, with no learning phase. After segregating between numeric and symbolic columns, we recognize the types of the columns using both the column title and the contents of the column. Then, we recognize the relations represented in the table: the use of both the table title and the table signature helps finding the relations with a very good recall and an acceptable precision. The relations are then instanciated for each row of the table: the instanciation of numeric types gives very good results, the instanciation of symbolic types being more difficult due to language variation from one author to another. Annotations associated with each row are expressed in a fuzzy extension of RDF representing either imprecises data or the semantic distance between Web data tables and the ontology. This fuzzy representation allows the use of approximate reasoning techniques in the querying step of the annotated Web data tables using the ontology (see [2]).

In the very next future, we want to explore two ideas. The first one consists in enhancing the performance of the annotation system using machine learning techniques on the knowledge of the ontology but without manual training on a subset of the corpus. The second perspective will be to exploit the annotated Web data tables in three different applications. The Web data tables annotated

by our method will be incorporated in two applications in the field of microbial and chemical risk in food. A third application in economic watch in the field of aeronautics will be also realized with EADS.

Acknowledgments

Financial support from the French National Research Agency (ANR) for the project WebContent in the framework of the National Network for Software Technology (RNTL) is gratefully acknowledged.

References

1. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. VLDB J. **10**(4) (2001) 334–350
2. Buche, P., Dibie-Barthélemy, J., Hignette, G.: Flexible querying of fuzzy rdf annotations using fuzzy conceptual graphs. In: 16th International Conference on Conceptual Structures. Volume 5113 of LNCS. (2008) 133–146
3. Doan, A., Domingos, P., Halevy, A.Y.: Learning to match the schemas of data sources: A multistrategy approach. Machine Learning **50**(3) (2003) 279–301
4. Liu, Y., Bai, K., Mitra, P., Giles, C.L.: Tableseer: automatic table metadata extraction and searching in digital libraries. In: JCDL, ACM (2007) 91–100
5. Cafarella, M.J., Halevy, A.Y., Zhang, Y., Wang, D.Z., 0002, E.W.: Uncovering the relational web. In: WebDB. (2008)
6. Cafarella, M.J., Halevy, A.Y., Wang, D.Z., 0002, E.W., Zhang, Y.: Webtables: exploring the power of tables on the web. PVLDB **1**(1) (2008) 538–549
7. Pivk, A., Cimiano, P., Sure, Y.: From tables to frames. In: ISWC. (2004) 116–181
8. Tenier, S., Toussaint, Y., Napoli, A., Polanco, X.: Instantiation of relations for semantic annotation. In: Int. Conf. on Web Intelligence. (2006) 463–472
9. Embley, D.W., Tao, C., Liddle, S.W.: Automatically extracting ontologically specified data from HTML tables of unknown structure. In: Conceptual Modeling - ER 2002. (2002) 322–337
10. Noy, N., Rector, A., Hayes, P., Welty, C.: Defining n-ary relations on the semantic web. W3C working group note (2006) http://www.w3.org/TR/swbp-n-aryRelations.
11. Hignette, G., Buche, P., Dibie-Barthélemy, J., Haemmerlé, O.: An ontology-driven annotation of data tables. In: WISE Workshops 2007. Web Data Integration and Management for Life Sciences., Nancy, France (December 2007) 29–40
12. Van Rijsbergen, C.J.: Information Retrieval, 2nd edition. Dept. of Computer Science, University of Glasgow (1979)
13. Platt, J.C. In: Fast training of support vector machines using sequential minimal optimization. MIT Press (1999) 185–208
14. Gagliardi, H., Haemmerlé, O., Pernelle, N., Saïs, F.: An automatic ontology-based approach to enrich tables semantically. In: AAAI Context and Ontologies Workshop. (2005)
15. Zadeh, L.: Fuzzy sets. Information and control **8** (1965) 338–353
16. Zadeh, L.: Fuzzy sets as a basis for a theory of possibility. Fuzzy Sets and Systems **1** (1978) 3–28
17. Dubois, D., Prade, H.: The three semantics of fuzzy sets. Fuzzy Sets and Systems **90**(2) (1997) 141–150
18. Cliver, D.O., Hajmeer, M.N., Jay-Russell, M.: Foodborne infections and intoxications. School of Veterinary Medicine, University of California (2004)