Applying data mining for ontology building

Abd-Elrahman Elsayed[1], Samhaa R. El-Beltagy[2], Mahmoud Rafea[1], Osman Hegazy[3]

[1] The Central Laboratory for Agricultural Expert Systems, Giza, Egypt

[2] Faculty of Computers and Information, Computer Science Department, Cairo University Giza, Egypt

[3] Faculty of Computers and Information, Information System Department, Cairo University Giza, Egypt

Abstract

Ontology represents the concepts and the relationship between them for specialized domain. Building ontology is a complex work, in order to build ontology you need a domain expert to help you to declare all domain concepts and the relationship between them. In this work we propose a methodology for building ontology based on the output of data mining result. We used c4.5 decision tree algorithm to discover and extract knowledge from structure data. Then we built ontology from the generated decision tree. We represent the generated ontology in XML and OWL languages. We work in two case studies; in the first case study we work in soybean diseases, and built ontology to represent the knowledge of diseases and their symptoms. In the second case study we work in animal diseases and extracted the knowledge related to them and we built ontology from the extracted knowledge.

Keywords

Ontology building, data mining, decision tree

1. Introduction

Data Mining is the process of finding and extracting new and potentially useful knowledge from data. Data mining is also known as Knowledge discovery in databases (KDD). The terms “Data mining” and “Knowledge discovery in database” are used interchangeably [1]. Data mining is an interdisciplinary field, drawing from different areas including database system, statistics, machine learning, data visualization and information retrieval.

The task of data mining involves two primary goals; those goals are prediction and description [2]. Prediction is concerned with using some variables or fields in the database to predict unknown or future values of other variable of interest, while description focuses on finding human-interpretable patterns describing the data.

1.1 What Is the Ontology

The word "ontology" has been recognized in philosophy as the subject of existence. In Artificial Intelligence community, ontology means a formal, explicit specification of a shared conceptualization. Conceptualization refers to an abstract model of some world phenomena. Ontology concepts and the relationship among those concepts should be explicitly defined. Further, ontology should be machine-readable and the ontology should capture consensual knowledge accepted by the community [13].

Ontology is used for knowledge sharing and reuse. It improves information organization, management and understanding. Ontology has a significant role in the areas dealing with vast amounts of distributed and heterogeneous computer-based information, such as World Wide Web, Intranet information systems, and electronic commerce. Ontology will play a key role in the second generation of the web, which Tim Berners-Lee call the "Semantic Web", in which information is given well-defined meaning, and is machine-readable. Search engines will use ontology to find pages with words that are syntactically different but semantically similar [3, 4, and 5].

2. Related Work

Usually the ontology building is performed manually, but researchers try to build ontology automatically or semi automatically to save the time and the efforts of building the ontology. We survey in this section the most important approaches that generate ontologies from data.

Clerkin et al. used concept clustering algorithm (COBWEB) to discover automatically and generate ontology. They argued that such an approach is highly appropriate to domains where no expert knowledge exists, and they propose how they might employ software agents to collaborate, in the place of human beings, on the construction of shared ontologies[6].

Blaschke et al. presented a methodology that creates structured knowledge for gene-product function directly from the literature. They apply an iterative statistical information extraction method combined with the nearest neighbour clustering to create ontology structure [7].

Formal Concept Analysis (FCA) is an effective technique that can formally abstract data as conceptual structures [9]. Quan et al. proposed to incorporate fuzzy logic into FCA to enable FCA to deal with uncertainty in data and interpret the concept hierarchy reasonably, the proposed framework is known as Fuzzy Formal Concept Analysis (FFCA).They use FFCA for automatic generation of ontology for scholarly semantic web [8].

Dahab et al. presented a framework for constructing ontology from natural English text namely TextOntEx. TextOntEx constructs ontology from natural domain text using semantic pattern-based approach, and analyzes natural domain text to extract candidate relations, then maps them into meaning representation to facilitate ontology representation [11]

Wrobel et al. used different ways to build ontologies automatically, based on data mining outputs represented by rule sets or decision trees. They used the semantic web languages, RDF, RDF-S and DAML+OIL for defining ontologies [10].

3. Problem Scope and Definition

The traditional task of the knowledge engineer is to translate the knowledge of the expert into the knowledge base of the expert system. Knowledge engineer uses ontology to represent the knowledge of the domain expert. Due to of the difficulty to find a domain expert and the needing for updating the knowledge represented in the ontology frequently, we proposed a system for building ontology automatically from the database. We used data mining techniques to extract knowledge from the database and represent it as ontology.

4. System Overview

In this section we will discuss system structure. The input to the system is a database that represents a repository of raw data, while the output is the generated ontology. Figure 1 shows the overall structure of the system.

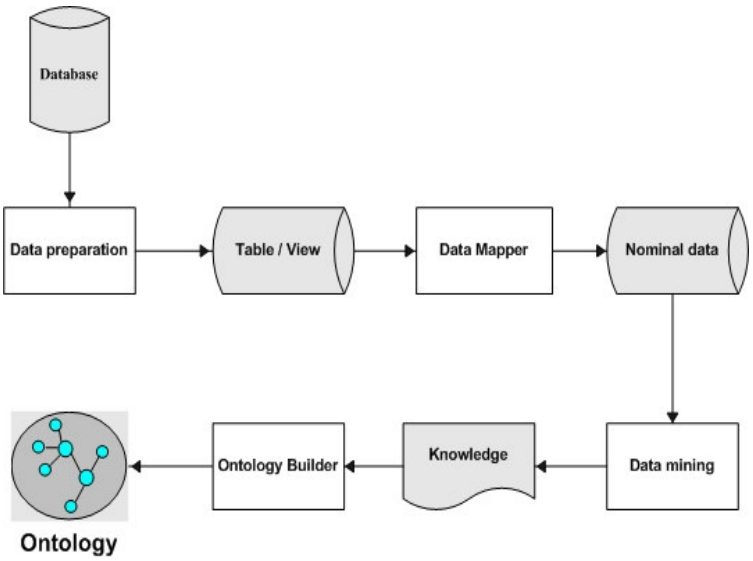

Figure 1 overall structure of the system

Ontology building from data mining will be achieved in two phases. The **data mining phase** is related to data mining process including data preparation, selection, and extraction of knowledge. The **ontology building phase** is related to the process of building the ontology from the extracted knowledge which represents the output of the data mining.

4.1 The Data mining component

As depicted in figure1 the first step in data mining phase is data preparation, the next step is data mapping, the third step is applying data mining techniques for discovering knowledge from the mapped data.

4.1.1 Data preparation

For data preparation the knowledge engineer will understand the semantics of the data and specify which tables and attributes will be used in the mining process. The Knowledge engineer may create a view in the database if he will work in a set of associated tables.

4.1.2 Data mapping

Data mapping is the process of representing raw data into format suitable to the selected data mining tool or algorithm. In the proposed system we build module for data mapping and we call (**Data Mapper**).

In the proposed system Data Mapper will be used to transform the input data into ARFF format which is used by WEKA (collection of machine learning algorithms) [12]. This module converts the input data into a nominal format to suit ontology builder requirements.

The input for this module is the database connection variables such as **severer IP**, **username**, **password**, and **the database name**.

The data-mapping module will display a list of all database tables and views as shown in figure 2.

The user of this module will specify which database table or view that will be mined; further he will select the attributes which will be used in the data mining process

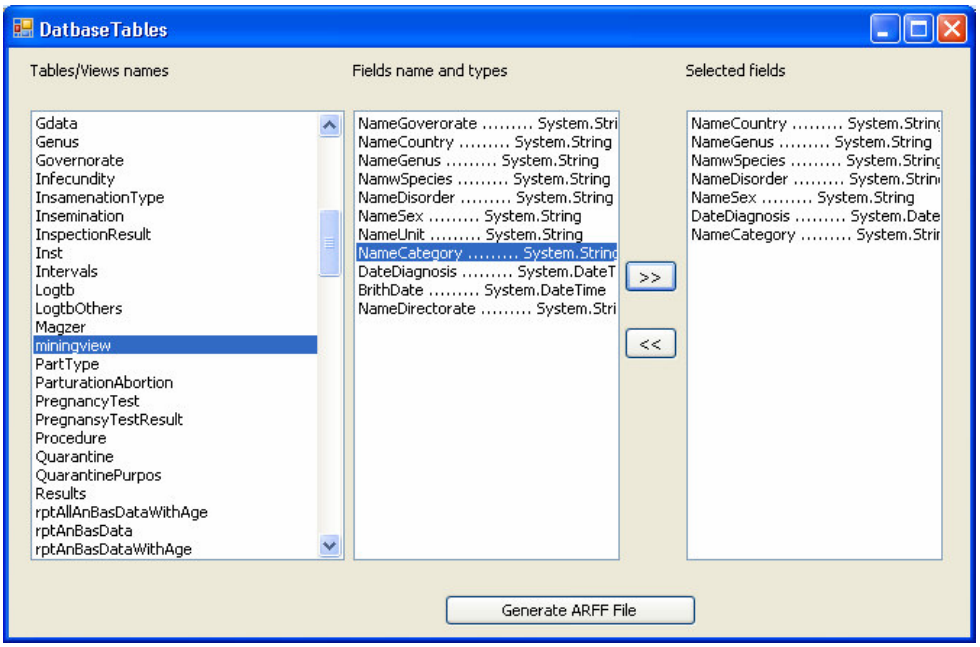

Figure 2 Data mapper module

The output of this module will be an ARFF file that contains the mined data.

4.1.3 Applying Data mining techniques for discovering knowledge from data

The third step is discovering knowledge from the preprocessed data. We used Weka framework because it is an open source package. The task of data mining will be classification. Many algorithms can be used for classification such as Support vector machine, Neural Network and decision tree. We select the decision tree algorithm because it introduces the discovered knowledge in readable format.

We select (j4.8) decision tree algorithm, which is Weka's implementation of c4.5 decision tree learner.

C4.5 is an extension to id3 algorithm. It addresses issues not dealt with ID3 such as:

- Avoiding over fitting the data, by determining how deeply to grow a decision tree.
- Handling continuous attributes.
- Handling training data with missing attribute values.

4.2 The Ontology Building Phase

At this phase the ontology builder will be used to generate the ontology automatically from the data mining output (extracted knowledge). In the next section we will discuss the ontology builder, and the algorithm that is used to generate ontology form data mining output.

4.2.1 The Ontology Builder

The Ontology builder is the main component in our system. It is used for parsing the output of the data mining result and generating an ontology. The ontology builder will generate ontology in two languages (XML & OWL).

In the first phase of our work we have generated ontology in XML format but to keep our work more standard and to support the semantic web vision we extended the tool to generate ontology in OWL. The input of the ontology builder is the file that contains the decision tree represented in textual format. This decision tree represents the output of the data mining process.

Figure 3 displays the components of the decision tree and its corresponding representation in the generated OWL ontology**.**

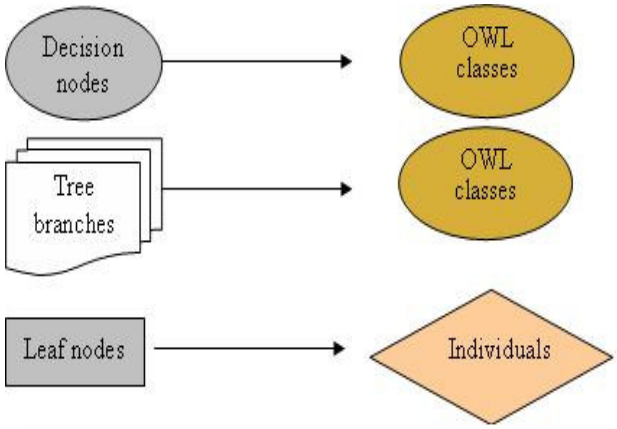

Figure 3 mapping decision tree to OWL ontology

In decision trees, decision nodes refer to the root node and internal nodes. As can be seen in figure 3, decision nodes can be mapped to OWL classes. Decision tree branches can also be represented in OWL as classes. Each branch in the decision tree may have a set of leaves. Each leaf in the decision tree represents a classification rule. Each rule can be represented as an individual (instance) of the class that represents its tree branch.

4.2.2 The Ontology Building Algorithm

The ontology building algorithm from decision tree is represented as follows:

Input:

- A decision tree.
- decision-nodes, the set of distinct decision nodes
- tree-branches, the set of distinct tree branches
- target-attribute, the target attribute
- Get-Branches, a function to get all branches which include specific node
- GetLeaveBranch , a function to get the branch of the leaf node.
- Get-Class, a function to get the class that represent decision tree branch
- Create-Individual, a function to create an individual for the leaf node.

Output: ontology

Method:

```
BEGIN
for each node N of decision-nodes
   Class C=new (owl:Class)
              C.Id= N.name
        DatatypeProperty DP=new (owl:DatatypeProperty)
              Dp.Id= N.name+"_Value"
              Dp.AddDomain(C)
              for each branch B of Get-Branches(N)
                 Dp. AddDomain (B.Get-Class ())
    endfor
endfor
```

//Generate an OWL class that represents the target-attribute

```
Class TargetClass= new (owl:Class)
TargetClass.Id= target-attribute.name
DatatypeProperty TargetDP=new (owl:DatatypeProperty)
//Generate DatatypeProperty for the target attribute
TargetDP.Id= target-attribute.name+"_Value"
TargetDP.AddDomain (TargetClass)
//Generate DatatypeProperty that represent certainty
DatatypeProperty CertaintyDP=new (owl:DatatypeProperty)
CertaintyDP.Id= "Certainty
//Generate classes that represent decision tree branches
```

```
for each branch B of tree-branches
  Class BranchClass= new (owl:Class)
  BranchClass.Id=""
  for each node N of B
    BranchClass.Id += N.name
  endfor
  BranchClass.Id+="determine"+ target-attribute.name
  TargetDP.AddDomain(BranchClass)
  CertaintyDP.AddDomain(BranchClass)
endfor
//Representing leaves nodes as individuals
for each leave-node LN of the decision tree
  Branch B= GetLeaveBranch(LN)
  Create-Individual (B, LN)
endfor
END
```

5. System Evaluation

In order to evaluate the proposed system we conducted two case studies. The first case study is concerned with plant diseases and the second case study is concerned with veterinary diseases.

The goal of the first case study is to assess the performance of the system and approve the validity of the generated ontology. The goal of the second case study is to build ontology for real live system.

5.1 Building Ontology for Soybean Diseases

We used the proposed system for generating ontology of soybean diseases. We get the data from the samples that are augmented to WEKA "data mining tool". There are 35 categorical attributes, Number of instance 683; number of classes (diseases) 19.

We used classification algorithm c4.5 for soybean data. The result of c4.5 is a decision tree.

Number of Correctly Classified Instances 625 (91.5081%), Number of incorrectly Classified Instances 58 (8.4919 %).

We built ontology for the knowledge represented in the decision tree.

To evaluate the knowledge represented in the generated ontology we compare the symptoms of sample of the most common diseases (7 diseases for simplicity) in the generated ontology with the domain expert knowledge.

We used the standard measures of precision, recall, and F-score (which represents the harmonic mean of precision and recall) to evaluate otology [14]. The calculations were based on a global contingency table shown in table 1.

Symptoms		Domain expert	
		YES	NO
The Generated Ontology	YES	TP	FP
	NO	FN	TN

Table 1 Global contingency table

- **TP (true positives)** represents symptoms that are identified by the domain expert and the generated ontology.
- **FP (false positives)** represents symptoms that are identified by the generated ontology but are not identified by the domain expert.
- **FN (false negatives)** represents symptoms that are identified by the domain expert but are not identified by the generated ontology.
- **TN (True Negatives)** represents symptoms that are not identified by both domain expert and the generated ontology.

Precision = TP / (TP + FP)
Recall = TP/ (TP +FN)
F-score = (2* Precision * Recall) / (Precision + Recall)

Table 2 shows contingency table for soybean disease symptoms

Symptoms		**Domain expert**	
		YES	NO
Data mining	YES	115	1
	NO	10	0

Table 2 contingency table for soybean disease

This case study resulted precision=99.13%, a recall=92%, F-score =95.43%

From this result we conclude that the generated ontology is similar to the expert knowledge and we can apply our idea to build ontology automatically using data mining techniques when the expert is not available or to help us to get knowledge from the expert to build ontology in semi automatic manner.

5.2 Building Ontology for BOVIS

In this case study we build ontology from **BOV**ine **I**nformation **S**ystem (BOVIS using data mining techniques. BOVIS has been developed by CLAES (Central Laboratory of Agriculture expert system) in co-operation with The Public Institute for Veterinary Services at Egypt. BOVIS is a system that enables decision makers to obtain statistical data about cattle and buffaloes on the national level. It helps in the tracing and management of contagious diseases. [15]. Data mining will help decision maker to discover useful knowledge and hidden pattern from the data. For mining the BOVIS database we focus in tables and attributes which are related to animal diseases. The animal diseases data includes information about country, governorate, Directorate, units, species, genus, sex in addition to diseases which infect the animal and the date of the diagnosis. In this case study we used the data mapping component to generate ARFF file to be mined by WEKA.

Figure (5) shows the generated decision tree for BOVIS diseases data. We used the classification algorithm J4.8 which is updated version of C4.5 algorithm.

```
J48 pruned tree
------------------
Category = ١٣٩٥,٠/١٨٩٧,٠) (أمراض-باطنة: نزلة-معوية
Category = أمراض-الطفيليات-الداخلية
|  Year = 2002: ١,٠) (كوكسيديا
|  Year = 2003: ١٠٣,٠/٢٧٩,٠) (نيماتودا
|  Year = 2004: ٢٥٩,٠/١٠٢٣,٠) (فاشيولا
Category = جراحة
|  Year = 2002: ٠,٠) (فتح-خراج
|  Year = 2003: ٢٢,٠/٤٦,٠) (خياطة-جروح
|  Year = 2004: ٦٤,٠/١٤٥,٠) (فتح-خراج
Category = أمراض-جهاز-بولى
|  Gender = أنثى
|  |  Genus = ٢,٠/٨,٠) (بلدى: التهابالكلى
|  |  Genus = ١,٠/٣,٠) (خليط-فريزيان: التهاب-المثانة
|  Gender = ١,٠/٤,٠) (ذكر: احتباس-بول
Category = ٣٦٨,٠/١٧٥٥,٠) (أمراض-تنفسية: نزلة-شعبية
Category = ١٣,٠/١٥٥,٠) (أمراض-الطفيليات-الخارجية: قراد
Category = ١٠,٠/١١٣,٠) (أمراض-الضرع: التهاب-الضرع
Category = أمراض-ضعف-الخصوبة-و-العقم-الدائم
|  Year = 2002: ١,٠/٢,٠) (التهاب-رحمى
|  Year = 2003
|  |  Genus = ٢٤٢,٠/٥٢٨,٠) (بلدى: خمول-مبايض
|  |  Genus = ٩٧,٠/١٩٦,٠) (خليط-فريزيان: التهاب-رحمى
|  Year = 2004
|  |  Genus = ٦١٣,٠/١٤٨٤,٠) (بلدى: خمول-مبايض
|  |  Genus = خليط-فريزيان
|  |  |  Governorate = ١٧,٠/٣٩,٠) (البحيرة: التهاب-رحمى
|  |  |  Governorate = ١٩٩,٠/٣٧٣,٠) (الشرقية: خمول-مبايض
Category = ١,٠) (أمراض-العجول-حديثة-الولادة: جهاز-هضمى
Category = ١٩,٠/٥٩,٠) (أمراض-الولادة-و-مضاعفتها: احتباس-مشيمة
Category = ٧٣,٠/٢٨٠,٠) (أمراض-طفيليات-الدم: بابيزيا
Category = ٧٧,٠/٢١٨,٠) (أمراض-جلدية: جرب
```

Figure 5 the generated decision tree For BOVIS diseases

In the generated decision tree Each leaf node are fallowed by a number (sometimes two) in parenthis. The first number tells how many instances in the training set are correctly classified by this node. The second number, if it exists (if not, it is taken to be 0.0), represents the number of instances incorrectly classified by the node[16].

From the generated decision tree that represents animal diseases in BOVIS database we generate ontology in XML and OWL languages. Here we will display description of the generated ontology in OWL.

5.2.1 Description of the generated ontology

The classes that represent decision nodes (**Category**, **Year**, **Gender, Genus, and Governorate**) and the class that represents a target attribute (**Disorder**) are displayed in figure (6)

```
<owl:Class rdf:ID="Category" />
<owl:Class rdf:ID="Year" />
<owl:Class rdf:ID="Gender" />
<owl:Class rdf:ID="Genus" />
<owl:Class rdf:ID="Governorate" />
 <owl:Class rdf:ID="Disorder" />
```

Figure 6 Classes that represent decision nodes and target attribute

Part of The classes that represent decision tree branches are displayed in figure (7)

```
<owl:Class rdf:ID="CategoryDetermineDisorder" />
<owl:Class rdf:ID="CategoryYearDetermineDisorder" />
<owl:Class rdf:ID="CategoryGenderGenusDetermineDisorder" />
```

Figure (7) classes that represent decision tree branches

Ontology builder created a Data type Property for each distinct decision nodes. For example ontology builder created a data type property for the decision node "**Gender**". The ID of this property will be "**Gender _value**" as displayed in figure (8).

```
<owl:DatatypeProperty rdf:ID="Genus_Value">
<rdfs:domain>
 <owl:Class>
<owl:unionOf rdf:parseType="Collection">
<owl:Class rdf:about="#Genus" />
<owl:Class rdf:about="#CategoryGenderGenusDetermineDisorder" />
<owl:Class rdf:about="#CategoryYearGenusDetermineDisorder" />
<owl:Class rdf:about="#CategoryYearGenusGovernorateDetermineDisorder" />
</owl:unionOf>
 </owl:Class>
</rdfs:domain>
</owl:DatatypeProperty>
```

Figure (8) example for the generated data type property of the decision node

The domain of the "**Gender _value**" will be the class "**Gender**" in addition to the classes that represent tree branches which includes "**Gender** "node.

Each rule generated by the decision tree will be represented as an individual (instance) for the class that represents its decision tree branch. Figure (9) shows part of the OWL syntax for the individuals that represents these rules

```
<CategoryYearDetermineDisorder rdf:ID="CategoryYearDetermineDisorder2">
    <Category_Value
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">أمراض-الطفيليات-
الداخلية</Category_Value>
    <Year_Value
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">2003</Year_Value>
    <Disorder_Value
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">نيماتودا</Disorder_Value>
     <Certainty
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">279.0/103.0</Certainty>
  </CategoryYearDetermineDisorder>
```

Figure (9) OWL syntax for the generated individuals

6. Comparison between the proposed system and Wrobel et al work

The proposed system is similar to the approach presented by Wrobel et al. The contribution of this paper is the method of ontology representation and building. We build ontology in a way that is suitable to all cases of data. We represent distinct decision nodes as an OWL classes. Also we represent decision tree branches as OWL classes. In our approach each leaf in the decision tree represents a classification rule. Each rule can be represented as an individual (instance) of the class that represents its tree branch.

But Wrobel et al represent each node as a class. And they represent the tree as class hierarchies. And we find that this representation is not suitable to represent all types of data. For example in the case of BOVIS the representation of class Year will be a class for example Year_3_2 which is sub class of Category1_3. But in realty Year is not a sub class of Category. In Our system we will not define the class Year as a sub class of Category. We just define the relation between classes that occur in same branch as a class whose name is a concatenated string of all classes names that appear in the decision tree branch plus the word determine.

For Example in the BOVIS case study the relation between Category, Year, are represented as OWL class CategoryYearDetermineDisorder. This class has three data type properties , The first data type property represent the “Category” class, and the second data-type property represents the class “Year” and the third data type property represent the class “Disorder”

Also our system generates ontology in OWL but Wrobel et al represent ontology in RDF or DAML+OIL.

OWL facilitates machine interpretability of web content greater than that supported RDF and DAML + OIL.

7. Conclusion and Future Work

In this paper we proposed a methodology for building ontology that represent the knowledge of specific domain using data mining techniques. We proposed a system that represents the discovered knowledge in OWL format. This system will help us in building an expert system based on the data mining result. In this paper we introduce two cases study, one of them for representing plant diseases and the other for representing animal disease

For future work we propose the idea of helping the knowledge engineer to acquire knowledge from the domain expert. Knowledge engineer will use the extracted knowledge as a guide in acquiring knowledge from the domain expert. The domain expert will validate the extracted knowledge, and remember the missed knowledge. Also, for future work we will investigate the methodology for building ontology from unstructured data such web pages and documents.

8. References

[1] Frawley, W., Piatetsky-Shapiro, G., and Matheus, C., Knowledge Discovery in Databases: An Overview. Ai Magazine, Vol. 13 (1992), pp. 57-70.

[2] Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. From data mining to knowledge discovery: An overview. In Advances in Knowledge Discovery and Data Mining, pp. 1 --34. AAAI Press, Menlo Park, CA, 1996.

[3] Berners-Lee, T., Weaving the Web, Harper, San Francisco, 1999

[4] Decker, S., Melnik, S., Van Harmelen, F., Fensel, D., Klein, M., Broekstra, J., Erdmann, M. and Horrocks, I. (2000) 'The semantic web: the roles of XML and RDF', IEEE Internet Computing, Vol. 4, No. 5, pp.63–74.

[5] Ding, Y., and Foo, S., (2002). Ontology Research and Development: Part 1 – A Review of Ontology Generation. Journal of Information Science 28 (2).

[6] Clerkin, P., Cunningham, P., and Hayes, C., Ontology Discovery for the Semantic Web Using Hierarchical Clustering, , Trinity College Dublin, Ireland, TCD-CS-2002-25

[7] Blaschke, C., & Valencia, A., Automatic Ontology Construction from the Literature, Genome Informatics, Vol. 13, pp 201–213, 2002.

[8] Quan, T. T., Hui, S. C., Fong, A. C. M., and Cao, T. H. (2004). Automatic generation of ontology for scholarly semantic Web. In: Lecture Notes in Computer Science. Vol. 3298. (pp. 726–740).

[9] Ganter, B.; Stumme, G.; Wille, R. (Eds.) (2005). Formal Concept Analysis: Foundations and Applications. Lecture Notes in Artificial Intelligence, no. 3626, Springer-Verlag. ISBN 3-540-27891-5.

[10] Wuermli, O., Wrobel, A., Hui S. C. and Joller, J. M. "Data Mining For Ontology Building: Semantic Web Overview", Diploma Thesis–Dep. of Computer Science WS2002/2003, Nanyang Technological University.

[11] Dahab, M. Y. Hassan, H., and Rafea, A.., TextOntoEx: Automatic ontology construction from natural English text, Expert Systems with Applications (2007), doi:10.1016/j.eswa.2007.01.043.

[12] Garner, S.R., (1995), WEKA: The Waikato Environment for Knowledge Analysis. In Proc. of the New Zealand Computer Science Research Students Conference, pages 57-64.

[13] Gruber, T.R. (1993). A translation approach to portable ontology specifications. Knowledge Acquisition, 5, 199-220.

[14] El-Beltagy, S.R., Maryam, H., and Rafea, Ontology Based Annotation of Text Segments. To appear in Proceedings of the 2007 ACM symposium on Applied computing, Seoul, Korea.

[15] http://www.claes.sci.eg/project/proj_view.asp?id=20

[16] http://grb.mnsu.edu/grbts/doc/manual/J48_Decision_Trees.htm