Automatic Building of an Ontology from a Corpus of Text Documents Using Data Mining Tools

J. I. Toledo-Alvarado*, A. Guzmán-Arenas, G. L. Martínez-Luna

Centro de Investigación en Computación (CIC), Instituto Politécnico Nacional (IPN)
Av. Juan de Dios Bátiz esquina con calle Miguel Othón de Mendizabal, 07738
México, D.F., México.
jitoledo@ipn.mx

ABSTRACT
In this paper we show a procedure to build automatically an ontology from a corpus of text documents without external help such as dictionaries or thesauri. The method proposed finds relevant concepts in the form of multi-words in the corpus and non-hierarchical relations between them in an unsupervised manner.

Keywords: Ontology learning, Data Mining, Machine Learning, Apriori algorithm

RESUMEN
En este artículo mostramos un procedimiento para construir automáticamente una ontología a partir de un corpus de documentos de texto sin ayuda externa tal como diccionarios o tesauros. El método propuesto encuentra conceptos relevantes en forma de frases temáticas en el corpus de documentos y relaciones no jerárquicas entre ellos de manera no supervisada.

1. Introduction

Information in text form is generated at exponential rate [1]. Every day, organizations all over the world generate reports, articles, books, emails, and all kind of textual data concerning several topics. The increase of the storage capacity of computers and servers enable these organizations to keep all files they produce without the need of deleting anything. Although this is an obvious advantage for everybody, it also implies some problems. One mainly problem they face is to know what kind of information they have, and how it is related. One way to organize information in computer science is in ontology form. This is similar to a conceptual map in which the main topics or concepts are related to each other by some kind of relations. Ontology learning (OL) is a research area focused on discovering or constructing in a semiautomatic or automatic manner an ontology from a collection of text documents. In this paper we show an automatic way to construct an ontology. As input, our method receives a corpus of documents related to a certain context. The experiments were made with papers related to “computer tomography”. As output, the method delivers a non-hierarchical ontology in the form of a set of concepts and a set of relations between those concepts.

1.1 Ontology definition

An ontology is a formal, explicit specification of a shared conceptualization of a domain of interest [2]. In computer science, a data structure is usually used to represent knowledge about a certain domain. The most frequent representation of ontologies in computer science is with graphs, where nodes represent concepts and edges represent relations between concepts. Novak [3] gives us a useful definition of concept for our purpose:

Definition 1.1. Concept is a perceived regularity in events, objects, records, etc., designated by a label.

Relations can be considered as special kinds of concepts which relate other concepts. In general they are binary, however, there are *n*-ary relations which are hard to represent graphically due to the intervention of several related elements. They are usually tagged with verbs that describe the relation. Some works classify concepts in classes or types and instances, i.e., sets *I* (instances) and *T* (types) form a partition of *C*. Formally an ontology is defined as follows:

Definition 1.2. Ontology is a tuple $O = (C, R)$ where C is a set of nodes (referring to concepts) which some of them are relations.
R is a set of restrictions in the form of $(r[c_1, c_2, \ldots, c_k])$ between the relation r and the concepts c_1 to k. Let us call arity of r as k.

1.2 Problem statement

The main problem to solve is: *Constructing an ontology automatically from a corpus of text documents*; however, it relies on solving small problems in order to get all the elements needed to build an ontology. First, it is necessary to identify relevant concepts hidden in the corpus of documents. These concepts can be multi-words, i.e., they may be formed by *n* different words. After finding relevant concepts, it is necessary to find the relations between those concepts. This is a hard task since we do not count on external help such as dictionaries or thesauri that could help us find relevant or semantic relations between concepts. Having found both sets (concepts and relations), building the ontology is a trivial task.

The paper is organized as follows: In Section 2, we review some previous work on ontology learning; in Section 3, we explain the method of constructing an ontology automatically; in Section 4, we show an example of the model; and finally, in Section 5, we discuss the results and present the future work.

2. Previous work on ontology learning

The process of OL may be separated into 6 layers [4][5]:

a. Term Extraction.
b. Disambiguation and synonyms.
c. Finding concepts.
d. Establishing concept hierarchies.
e. Finding relations between concepts.
f. Finding rules in the ontology.

Research on OL can be applied on one or several layers of the process. This research is focused mainly on term extraction, finding relevant concepts, and finding non-hierarchical relations between concepts.

a. Term Extraction is the basic process in every ontology learning research. Its objective is to obtain terms, which may be considered as linguistic realizations of domain specific concepts. The specialized literature provides many examples of term extraction methods. Most of these are based on information retrieval methods for term indexing [6]. Some researchers also consider NLP for term extraction [7].

b. Disambiguation is the process by which every term found on the corpus is assigned to a specific context. Words may refer to several contexts, for example the word *horse* may refer to an animal, to a unit of horsepower, to heroin, to an obstruction in a vein (mining), and to a frame or structure on which something is mounted [8]. There is much work and there are many methods about term disambiguation v.g. [9]. Most common methods apply clustering techniques to group similar words, using some association measure to detect term pairs statistically correlated [10]. Clustering terms in clusters with the same meaning helps to identify ontological classes. Dictionaries and thesauris are used in term disambiguation, in this sense, WordNet [11] has been widely used. In order to know more about research on term disambiguation you can consult Buitelaar 2005 [4].

c. Finding concepts. Not all terms are suitable concepts, there is controversy about what can be considered as concept; common definitions of concepts are i) an abstract idea, ii) an idea or mental picture of a group or class of objects [12].
In the context of this paper, concept is the label assigned to some knowledge regularity in the corpus of documents, conformed by set of terms, which may consist from one to *n* terms. In other words, we consider that a corpus of documents from a specific context consists on several knowledge abstractions; those abstractions are regularly found several times all over the corpus. Finding those regularities is equivalent to finding concepts. Most research focuses on finding concepts use machine learning techniques like clustering [13] in an unsupervised manner. However, some work use supervised procedures in order to improve their results [14].

d. Establishing concept hierarchies. Some kind of ontologies may consider hierarchies between concepts; this is a kind of relation of the type subset or *is-a*. Finding this kind of relations is equivalent to establishing a specific taxonomy between concepts. There are three main paradigms to induce this kind of relations. The first is the application of lexicon-syntactic patterns to detect hyponymy relations as proposed by Hearst in [15]. The second is based on Harris's distributional hypothesis; in this line, researchers have exploited hierarchical clustering algorithms to derive hierarchies from text, e.g., [13].

e. Finding relations between concepts. Also known as nonhierarchical relations, this kind of relations establish some kind of bond between two concepts. It is desired that this relations are labeled in order to give significant meaning to the ontology. Most work on finding nonhierarchical relations combine statistical analysis with some level of linguistic analysis as in Buitelaar 2004 [16]. Relation extraction through text mining was introduced first by Maedche and Staab in 2000 [17].

f. Finding rules. This layer of ontology learning focuses on discovering inference rules from text, such as "X is author of Y", "X wrote Y", "X solved Y", "X found a solution to Y", and "X caused Y", "Y is triggered by X". There is little work related to this area of research such as the work of Lin and Pantel [18], and there are not real trends on it. This layer of ontology learning is beyond the objectives of this work.

3. Building automatically an ontology

3.1 Definitions

In our model, we process a corpus of documents in order to get the most relevant concepts and the most important non-hierarchical relations between them. The relevance of a concept is determined by a data mining algorithm used originally for "market basket analysis". The algorithm is intended to find which items are suitable to be bought together. Every sale is considered as a transaction and every product which was sold is called an item.

For our purpose, the algorithm treats every document as a transaction and every word on the document as an item of the transaction

Definition 3.1. Token is a character string found in a file which is separated by spaces from other tokens.

Definition 3.2. Term is the label that represents a set of identical tokens.

Definition 3.3. Itemset is a set of terms or items.

Definition 3.4. Frequent itemset is an itemset whose items appear in the same documents at least in *S* documents or transactions. *S* is called the minimum support.

After filtering the contents of the text documents, we calculate the frequencies of all terms appearing in the corpus; we consider the *N* most frequent terms as possible concepts or part of concepts that are multi-words. In order to find multi-word concepts, we find the frequent itemsets with the Apriori algorithm. Frequent itemsets are sets of items or terms that appear frequently in the same documents. Then, we found the multi-word concepts with an original algorithm. After we get the relevant concepts of the corpus, we find relations between them in the form of documents that contain two concepts, i.e., we consider that concepts that appear in the same documents are somehow related. Finally, we organize them in the form of a graph.

3.2 Preprocessing

As input, we consider a corpus of documents in PDF format related to a certain topic. In the experiments, a set of documents was used related to "Computed Tomography". In order to have the documents ready for processing, they need to be "cleaned" by a series of filters. Firstly, we extract the documents' content; to achieve this goal, we use PDFBox [19], a library from apache that enables us to extract content from PDF documents. Once we have the content in string form, it is ready to be filtered. The first filter changes the contracted words into their expanded way, in other words, instead of strings of the form "I'm" it changes into "I am". The second filter removes every non-alpha character (symbols), and removes multiple spaces; the final string contains words separated by one space character between them. The next filter removes a list of stop-words [20], that is, a list of the most frequent words in the English language. In this case, we remove the list of the 2000 most frequent words in the English Language. Another filter is the stemmer, this filter reduces all the words in the string into their proper stem form; we do this in order to avoid multiple concepts for words with the same meaning, for example, *computer* and *computerized,* both words refer to computation, hence the stemmer reduces both words to their proper stem: comput. The stemmer we use is Snowball [21]. After the content is passed through all the filters, it is considered to be ready to be processed by the data mining algorithm. In Table II, the statistics of preprocessing are shown. Columns are different states of preprocessing, and rows show different metrics. For example, the cell referenced by column Ex (Extraction) and row ToC (Token Count) shows that 76,649,828 tokens wwere found just after extraction, without any filter of preprocessing. Table 2 shows the decrease of tokens, terms and length of tokens as filters are processed. Table 1 shows the acronyms used in Table 2.

cronym	Meaning
Ex	Extraction
WiSy	Without Symbols
WiStW	Without Stop-words
St	Stemming
WiSmW	Without Small Words
ToC	Token Count
TeC	Term Count
LoTo	Longest token
AvTeL	Average Term Length
AvToC	Average Token Count
AvTeC	Average Term Count
AvFL	Average File Length
ShF	Shortest File
LoF	Longest File

Table 1. Acronyms for preprocessing statistics table.

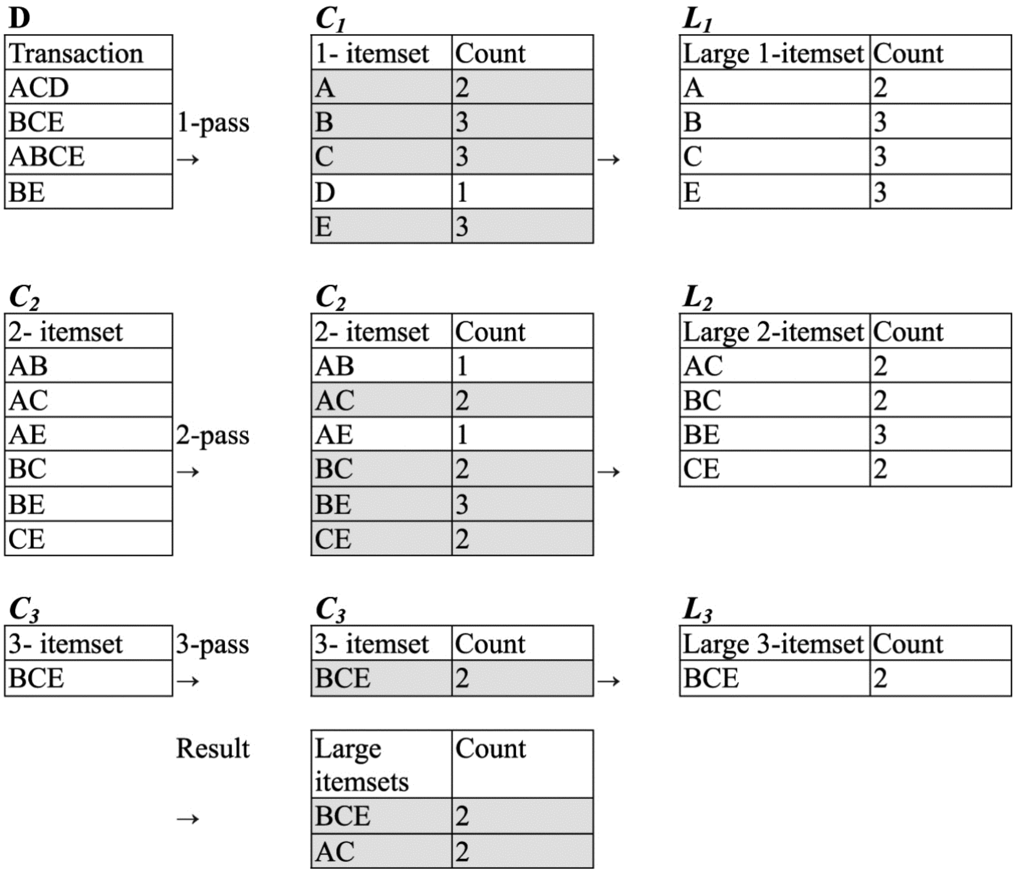
3.3 Mining frequent terms

The algorithm for mining the documents is called Apriori. In order to use it, it is necessary to convert the documents into a proper input for the algorithm. This is a binary matrix file in which the columns represent all the words in the corpus and the rows are the documents in the corpus. In the matrix character "1" means the word represented by the column appears in the document represented by the row, and "0" means the word does not appear in such document. The algorithm Apriori was proposed by R. Agrawal and R. Srikant in 1994 [22]. This algorithm first finds the set of frequent 1-itemsets by scanning the database to accumulate the count for each item and collect those items that satisfy minimum support. The resulting set is denoted *L1*. Next *L1* is used to find *L2*, the set of *frequent 2-itemsets*, which is used to find *L3*, and so on, until no more frequent *k-itemsets* can be found. In Figure 1, we show an example of how this algorithm works.

In the context of this research, each transaction is a text file, and each item is a term in the corpus. In the first step of the algorithm, the support for all items is calculated. All of the items that satisfy the support established (in the example is 2) are candidates to be frequent itemsets. Those items are then passed to the next pool to form 2-itemsets, and so on.

	Ex	WiSy	WiStW	St	WiSmW
ToC	76,649,828	70,427,042	33,753,162	33,753,162	27,443,988
TeC	924,219	933,307	928,995	849,421	848,732
LoTo	421	176	176	175	175
AvTeL	5.6	5.4	6.6	5.6	6.5
AvToC	4,508.8	4,142.7	1,985.4	1,985.4	1,614.3
AvTeC	1,604.6	1,051.4	668.6	622.4	554.3
AvFL	30,575.6	26,757.0	15,144.5	12,217.7	26,757.0
ShF	15,764	13,698	8,548	7,586	6,971
LoF	519,786,102	454,869,196	257,457,896	223,157,523	207,700,914

Table 2. Preprocessing Statistics.



D

Transaction
ACD
BCE
ABCE
BE

C_1

1- itemset	Count
A	2
B	3
C	3
D	1
E	3

L_1

Large 1-itemset	Count
A	2
B	3
C	3
E	3

C_2

2- itemset
AB
AC
AE
BC
BE
CE

C_2

2- itemset	Count
AB	1
AC	2
AE	1
BC	2
BE	3
CE	2

L_2

Large 2-itemset	Count
AC	2
BC	2
BE	3
CE	2

C_3

3- itemset
BCE

C_3

3- itemset	Count
BCE	2

L_3

Large 3-itemset	Count
BCE	2

Large itemsets	Count
BCE	2
AC	2

Figure 1. Apriori algorithm with minimum support = 2.

3.4 Finding concepts

Every itemset is a suitable candidate to be a concept. We will consider a concept as a maximal frequent itemset whose members appear together in at least *u* documents, where *u* most be defined previously by the user.

Definition 3.5. Maximal Frequent Itemset. An itemset is called maximal if it is not a subset of another itemset.

Because frequent itemsets do not take into account the order of the items within the set, we have to generate all possible combinations of the items within each frequent itemset in order to find the multi-word concepts. That is, for a *k- itemset* there are *k!* possible concepts. The algorithm to find concepts is as follows.

Algorithm 3.1. Finding concepts

```
/*Array of frequent itemsets */
fis : ARRAY ;
/*Array of combinations of items within an itemset */
com : ARRAY ;
/*Array of files that will be searched for the concepts */
fil : ARRAY ;
/*Array of items that represent the best combination, i.e., the one with more occurrences */
bestComb : ARRAY
/*Array of concepts */
concepts : ARRAY ;
begin
    for i := 0 to FIS:length 1 step 1 do
        com = GENERATE COMBINATIONS(fis[i]);
        for c := 0 to COM:length 1 step do
```

```
            best = GET BEST COMB(com[c], fil);
            if best! = null ^ IS NOT SUBSET (bestComb, concepts) then
                concepts:add(best);
            fi
        od
    od
end
```

3.5 Finding relations

The relations we find to form the ontology are not hierarchical. The relations are established when two concepts appear in the same documents.

Definition 3.6. Relation is an undirected link between two concepts.

After finding all concepts, we look for relations within all documents. This is accomplished by creating pairs of concepts and looking for every pair in all the documents of the corpus. The order of the concepts is irrelevant to establish the relation since relations are not directed.

4. Experiments

The experiments were made on an Intel Core 2 Duo Processor running at 2.8 GHz and 4 GB of RAM. The experiments were made with a subset of documents already preprocessed. We consider 1000 documents and take the 100 most frequent terms as items for the Apriori algorithm, that is, the file used as input for the Apriori algorithm consists on a binary matrix of 100x1000, the file size was 196 Kb. With a support of 80 percent, the Apriori algorithm output were 222 frequent itemsets in less than a second. The largest itemsets contained 5 items. This means that those 5 terms appear in the same documents in at least 80 percent of the documents of the corpus. The algorithm for finding concepts achieves to find 9 concepts of 2 items, 25 concepts of 3 items and 13 concepts of 4 items. None concepts of 5 items were found. With these concepts, 135 relations were found in the corpus. The set of relations and concepts form an undirected graph that represents the ontology learned from the corpus.

5. Discussion

As with every parameterized method, it is difficult to find optimal values for the method to work as expected. The parameters used in the experiments were found on a trial-and-error basis, including the minimum support, and the threshold to consider an itemset as multi-word. In the future, we pretend to give better understanding on the impact of these parameters. We found a non-hierarchical ontology in a corpus of documents without any help of dictionaries or thesauri in an automatic way. Most work on this field needs either the supervision of an expert that validates the ontology generated in every step of the method or metadata such as dictionaries to find relations between concepts. With this research, we proved that it is possible to establish unknown relations between new concepts of the domain of study in an unsupervised way and without external help. We proposed a new manner to learn multi-words (concepts). Instead of looking for multiwords in all the vocabulary, we look them up only with the most frequent terms reducing the computational effort. This is accomplished with data mining tools such as the Apriori algorithm. As future work, we want to tag relations either with some kind of weight or with a label that describes the relation between the concepts. We will also provide qualitative and quantitative measures to validate the ontology generated. This method will be proved with a largest corpus to validate its scalability.

References

[1] Victoria Barbosa José Alfredo and Ávila Aoki Manuel. Patrones de crecimiento en la generación de información en discos duros, Revista Digital Universitaria, Vol. 10 No. 6, June, 2009.

[2] Thomas R. Gruber, A translation approach to portable ontology specifications, Knowledge Acquisition - Special issue: Current issues in knowledge modeling, Vol. 5, No. 2, June, ,1993, pp. 199-220.

[3] Novak, J. D. & Cañas, A. J., The Theory Underlying Concept Maps and How to Construct and Use Them (2006-01 Rev 2008-01), Technical report, Florida Institute for Human and Machine Cognition, 2006.

[4] Paul Buitelaar, Philipp Cimiano, Bernardo Magnini, Ontology Learning from Text: An Overview, in Ontology Learning from Text: Methods, Evaluation and Applications / Frontiers in Artificial Intelligence and Applications volume 123, Paul Buitelaar, Philipp Cimiano, Bernardo Magnini Editors, IOS Press, 2005, pp. 1-10.

[5] Cimiano, P.; Völker, J. & Studer, R., 'Ontologies on Demand? -A Description of the State-of-the-Art, Applications, Challenges and Trends for Ontology Learning from Text Information', Information, Wissenschaft und Praxis, Vol. 57, No. 6-7, October, 2006, pp. 315-320.

[6] Manning, C. D.; Raghavan, P. & Schütze, H., An Introduction to Information Retrieval, Cambridge University Press, 2008.

[7] Kyo Kageura, B Atrice Daille, Hiroshi Nakagawa & Lee-Feng Chien, Recent advances in computational terminology, John Benjamins, 2004, pp. 1-22.

[8] http://www.wordreference.com/definition/horse.

[9] José Francisco Martínez Trinidad, Beatriz Beltrán Martínez, Adolfo Guzmán-Arenas & José Ruiz-Shulcloper, CLASITEX: A Tool for Knowledge Discovery from Texts, In Proceedings of the Second European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD '98), Jan M. Zytkow and Mohamed Quafafou Editors, 1998, pp. 459-467, Springer-Verlag, London, UK.

[10] Christopher D. Manning and Hinrich Schütze, Foundations of Statistical Natural Language Processing, MIT Press, Cambridge, MA, USA, 1999.

[11] http://wordnet.princeton.edu/

[12] http://www.wordreference.com/definition/concept

[13] Philipp Cimiano, Andreas Hotho, and Steffen Staab, Learning concept hierarchies from text corpora using formal concept analysis, Journal of Artificial Intelligence Research, Vol. 24, No. 1, August, 2005, pp. 305-339.

[14] David Faure & Claire Nedellec, Knowledge Acquisition of Predicate Argument Structures from Technical Texts Using Machine Learning: The System ASIUM, In Proceedings of the 11th European Workshop on Knowledge Acquisition, Modeling and Management (EKAW '99), Dieter Fensel and Rudi Studer Editors, Springer-Verlag, London, UK, 1999, pp. 329-334.

[15] Marti A. Hearst, Automatic acquisition of hyponyms from large text corpora, In Proceedings of the 14th conference on Computational linguistics - Volume 2 (COLING '92), Vol. 2, 1992, pp. 539-545 Association for Computational Linguistics, Stroudsburg, PA, USA.

[16] Buitelaar, P.; Olejnik, D. & Sintek, M., A Protégé Plug-In for Ontology Extraction from Text Based on Linguistic Analysis, in 'The Semantic Web: Research and Applications', Christoph Bussler; John Davies; Dieter Fensel & Rudi Studer Editors, Springer, Berlin / Heidelberg, 2004, pp. 31-44.

[17] Maedche, A. & Staab, S., Discovering Conceptual Relations from Text, in Proceedings of the 14th European Conference on Artificial Intelligence (ECAI), 2000, IOS Press, Amsterdam.

[18] Dekang Lin and Patrick Pantel, DIRT @SBT@discovery of inference rules from text, In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '01), 2001, pp. 323-328, ACM, New York, NY, USA.

[19] http://pdfbox.apache.org/

[20] http://www.cs.utexas.edu/users/ear/cs378NLP/EnglishWordFrequencies.txt

[21] http://snowball.tartarus.org/

[22] Rakesh Agrawal and Ramakrishnan Srikant, Fast Algorithms for Mining Association Rules in Large Databases, In Proceedings of the 20th International Conference on Very Large Data Bases (VLDB '94), 1994, pp 487-499, San Francisco, CA, USA.