Text Categorization and Classification in Terms of Multi-Attribute Concepts for Enriching Existing Ontologies

Kourosh Neshatian Mahmoud R. Hejazi

Info. Society Department, Iran Telecom Research Center

P.O. Box 14155-3961

{neshat, m_hejazi}@itrc.ac.ir

Abstract: In this paper, we propose a novel comprehensive architecture for recognition of different kinds of documents and then use appropriate compoenent to extracting document information and feeding them to an existing ontology. Where by ontology, we mean a multi-attribute conceptual graph with different types of relations. This novel architecture includes two types of text processors: Information Extrators and Text Categorizers, where, each one enrichs an ontology in a different manner. We describe the details of our proposed DTR (Document Type Recognition) compoenet and a typical Information Extractor, which is able to perform knowledge acquisition from semi-structure documents (e.g. glossaries, dictionaries, and data sheets). Next, we propose a Text Categorizer component which uses a classification approach to find the related concept(s) of the text documents and attaching them together. We have evaluted the whole architecture in a question answering system named TeLQAS. The results show show a reasonable increase in accuracy and dcrease in response time of the system.

Keywords: *Information Extraction, Knowledge Acquisition, Text Categorization and Classification, Feature Selection.*

1 – INTRODUCTION

The amount of information available to a specific domain has grown drastically with the appearance of digital resources like Internet, digital document corpuses and libraries. Unfortunately this growth of available information has made the access to useful or needed information not much easier, as the access is usually based on keyword searching or simple browsing (accessing documents based on their storage location e.g. by Web site or the collection containing them). Keyword searching usualy results a lot of irrelevant information, primary because a term can have different meanings in distinct contexts or in different level of hierarchy. Inteligent tutoring systems have similar problems when they merely relies on index-based retreival engines.

Because the natural language is the most convenient and the most intuitive method of accessing the information [11], people should be able to access the information, using a system capable of understanding and answering natural language questions—in short, a system that combines human-level understanding with the infallible memory of a computer to satisfy user information needs, exactly [11]. This is why some fields of IR (Information Retrieval) like Question-Answering and Knowledge Navigation systems are allocating the most of current research activities [12].

The key element in all kinds of these systems, capable of interacting in natural language, is a knowledgebase or an ontology [13]. In the context of knowledge sharing, the term ontology is used to mean a specification of a conceptualization [5]. Enriching an ontology with finding instances of ontology classes and filling slots with appropriate values, promotes the ontology to become a knowledgebase. A knowledgebase is a form of database used in expert systems that contains the accumulated body of knowledge of human specialists in a particular field [14]. Developing an application, based on a knowledgebase instead of traditional retrieval techniques, has several advantages:

1. Using a proper inferring mechanism like plausible reasoning, the application can provide user information requirement, directly [15].
2. The abstraction given by the ontology eliminates dealing with document-specific representations [4].
3. By this abstraction robustness towards changes in content and format of the documents is gained [4].

Usually the universe ontology is considered as a set of domain specific (sub)ontologies. This division facilitates the creation and maintenance task of ontologies. In this manner the global ontology for universe is obtained by merging these domain specific ontologies [16]. The domain ontology boosts an application by providing a common understanding of the domain.

The required domain-specific ontologies for many ontology-based systems are usually created by domain experts, using some construction tools like: Ontolingua [1] and Protégé [2, 3]. The manual creatin of ontologies is time consuming and expensive task so the wide-spread usage of ontologies is still hindered. For satisfaction of this demand, some methods have been invented to automatically extend the graph of a primitive ontology to a knowledgebase or even create an ontology from scratch. Such methods are usually based on NLP (Natural Language Processing) techniques [17]. These method of ontology extension has some weakness as:

1. These methods can not understant every human litreture, because of imprecise nature of human language. So sometimes, processing different texts about a topic may lead to contradiction in the obtaind ontology.
2. The time required to process huge number of documents avaiable to a specific domain will be too long with current algorithms and processors.
3. Many documents include information which is presented by the mean of formatting styles and not linguistic structures(e.g. Gloassaries, Dictionaries and Data Sheets). Such documents can not be processed, using NLP Techniques.

So here we focus on a novel architecture which can cover this incovenience of NLP methods. In this new way, we enrich an ontology (and shift it to a knowledgebase) by using text categoriation methods and a new generation of parsers which parse the information exist via visual features of the documents. Although the proposed architecture can be used in paralel with other tools and methods, currently exist.

Because the enrichment task did not depend on final user enquiry, all the processes required for our proposal are assumed offline. These processes are intended to enrich an existing ontology by finding and adding related slots arround the current concepts (and not by finding new concepts). The enrichment system finds the formatting style of document using a special kind of recognizer and then uses appropriate components to extract the desigred information from the document. Among these copmonents we will describe Glossary Parser, in depth. Finally we will show how the results of such enrichment may boost an online expert system like a Question-Answering or an Inteligent Tutoring system.

2 - ARCHITECTURAL DESIGN

Before processing a document to extract the facts it's comprised of, an enrichment system should determine which text processor is capable of performing this task in the best manner. As mentioned before, against the condensate information contained in some kinds of documents like glossaries and technical data sheets, they fail to be processed by two main categories of text processors which are NLP and classical classifiers. This is because:

1. NLP-based processors can not manipulate documents like glossaries, because they don't conform to any linguistic grammar (they are a sequence of entries, recognized by their different format in entry header and entry body).
2. These types of documents are noisy data for classical text classifiers like Centroid-based and Naive Bayesian because they cover a wide range of concepts and usually lead to a rejection or false classification.

To rectify this imperfection, one primary goal of our design is to redirect diverse kinds of documents to appropriate text processors. So in our proposed architecture, the first step in processing diverse kinds of documents is the document type recognition. Passing this stage, the document can be forwarded to an appropriate service for processing.

Our proposed architecture has been illustrated in Figure 1. This figure shows the main components of the enrichment system. We have considered this system as an offline service which can be incorporated in an ontology-based application like a Question-Answering or an Intelligent Tutoring system. This service is interacting with some other components like collections, ontology and document warehouse which are involved in the application.

When a new document comes from the Web or any other kind of collection, it will be given to the offline service for processing. The offline service passes this document to the *DTR (Document Type Recognition)* component. This is a rule-based component that uses visual and textual information of document to determine which document type it belongs to. We will describe this component in detail, later. A document may be directed to one or more information extractor components. For example if the document is a regular topic about the concept(s) of ontology, it can be directed to both *NLP processor* and *Text Categorization*.

The result of text processors is applied to the domain ontology. This is done by the mean of adding new attributes and relations to existing concepts. The result of a text processor like *Glossary Parser* which is responsible for glossary documents (and will be discussed later) is a set of *definition attribute*s and *definition relation*s which will be liked to ontology concepts. A *Specification Parser* component which process data sheets and technical specification generates a set of attributes defining the property of ontology classes and instances. The *Text Categorization* component uses classification methods to find the category which a regular text documents belongs to. The Text Categorizer stores the classified documents in document warehouse and corresponding relations in the ontology graph.

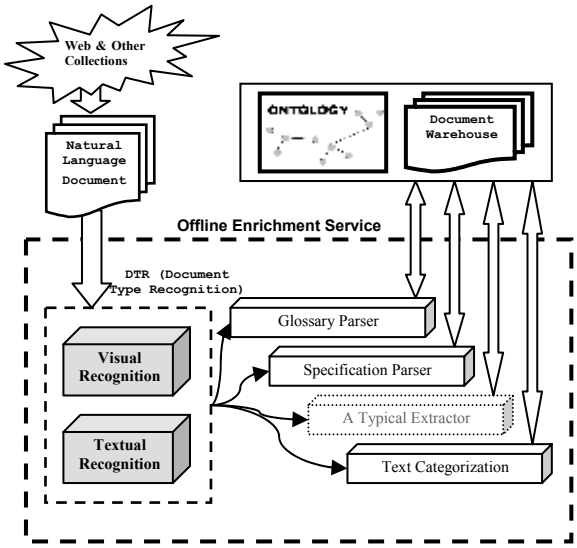


Figure 1. Component diagram of enrichment system and its interaction with other parts of an IR system (like ontology and document warehouse) has been iluustrated. All required actionss for enrichment process will be done in offline mode.

3 - DOCUMENT TYPE RECOGNITION

The DTR component uses textual and visual information of the document to find out what type of document it belongs to. If the document is classified as a pre-known type, it will be forwarded to a corresponding text processor. If the document fails to be recognized as pre-known type, it will be directed to Text Categorization component (which will be discussed later).

The DTR consists of two sub-components: one for analyzing textual features of document and another for visual features. The textual features used for determining the type of a document are different from those used in other kinds of text classifiers. For example the TF (Term Frequency) vector of a document has no meaning for type recognition but the existing of some words like 'Abstract' and 'Introduction' in a respective sequence and structure may increase the probability that the document is a paper. Some textual information which are very useful for identifying the document type and finding the structure of a document are: punctuations, markups and tags.

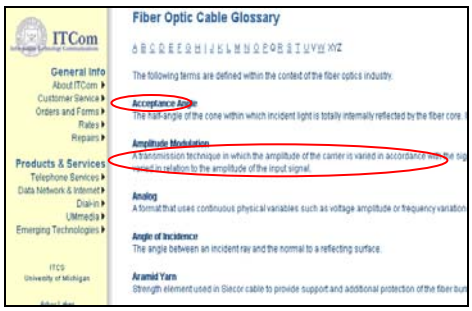
A more challengeable subject in the context of type recognition is that sometimes the type of a document relies on the visual formatting of the document (e.g. font, size, typeface and color). The DTR component makes a vector containing the visual features which are significant in type recognition. The DTR uses a fuzzy rule-base to perform its task. So, low level visual features should be fuzzified to a linguistic label.

A sample document which is of type glossary has been shown in Figure 2(a). Different styles in entry header and body have been enclosed in an ellipse. The DTR scans the document for different styles and tags them as S_i where i is an integer number and:

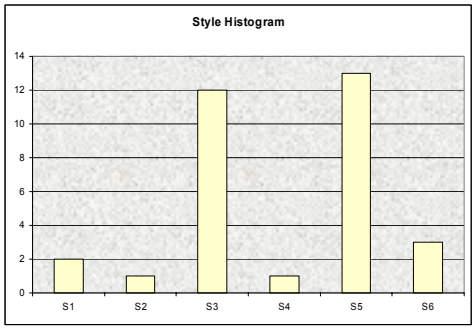
$$S_n \geq S_m \text{ where } n < m$$

The S_n will be greater than S_m where the its font size is greater or where the both of them use same font and size but the first has a text effect like bold or italic. Figure 2(b) illustrates the histogram of these styles. As seen, there exist two long bars which are for header and body styles respectively. A fuzzy classifier can detect these kinds of documents (dictionaries, glossaries and …) by this statistic and by some complementary rules such: these two types of styles must be appear one after the other. The later rule may be implemented by the aid of a FSA (Finite State Automata). We will discuss this topic in next section.

In practice, the textual and visual recognition components can be combined to a single component which is the DTR itself. In this case the feature space of the recognition system is mainly consists of styles and visual information and fewer textual features are used. Another impoetant point is that a single algorithm rarely could detect the all the document types. So it will be more reasonable to implement the DTR as a set of filters which the document sould pass through.



(a)



(b)

Figure 2. (**a**) A glossary document in which, different styles been used forentry header and body (**b**) Histogram of the document styles; header and body styles raised two tall bar in this histogram.

4 - INFORMATION EXTRACTOR COMPONENTS

When the type of the document is determined, the document will be directed to an appropriate IE (Information Extractor) component. Each IE component has its specific algorithm to find the information it is expected to. By providing appropriate IE components for different kinds of document, we can enrich the ontology from different aspects and benefiting from diverse kinds of documents. In the next

section, we will describe an IE component called *Glossary Parser* or *GP* in short.

4.1 GLOSSARY PARSER

Against the availability of some universal data formating guidlines like XML, Plenty of documents in the collections have been formed in a different format and mostly regardles of any standard. The worse problem is that many of these user defined formats are not text-based. Examples of such documents are glossaries, data sheets and specification documents which are important part of a specific domain reresources. The information comprised in many of these documents, could not be extracted using available Techniques like text-based parsers and NLP techniques, as mentioned before. Providing a way to feed these information to the domain ontology, gets the information which could not be accessible normaly. By enriching the ontology around its concepts, the performance of using IR (Information Retirieval) systems will improve.

Usually for a specific domain, there is many glossaries (or special dictionaries) available. But in practice, these glossaries are not as useful as they potentially could be. The main reasons for this problem are:

1. These glossaries are scattered among diverse type documents from different sources (e.g. in Telecom domain there are many small or large glossaries for Fiber Optics or Wireless technologies in different sites and collections).
2. Most of the times, these glossaries lack an appropriate lookup engine and if they do rarely, it is a separate application.

Here we describe our proposed *GP (Glossary Parser)* which is capable of understanding the visual features of glossary document. As we discussed before, in many glossaries, discrimination between entries and between an entry header and body is recognized by the mean of different formatting style (visual) in header and bodies. Usually there is no textual feature like colon or new line character that could be used as a separator.

The GP component has been designed as a FSA that can operate on visual features as input. The process of parsing is to scan the document, character by character and then reveal the formatting style of each character in the text. The formatting style of each character is as an input for the parser.

Table 1 shows the state transition table of the parser. The S_m and S_n are the formatting styles of the entry header and entry body, respectively. These styles can be identified by looking at formatting style histogram. The *S* symbol indicates the beginning state of the parser or positions of the document in which no header or body could be recognized. The *H* and *B* symbols indicate positions in the header and body. In some situations a commit command has been cited in parenthesis which informs the header and body couple is ready to save.

The results of GP will be stored in the ontology in the form of an attribute and a corresponding relation (of type definition) to the related concept. Regularly, only those definitions having a corresponding concept in the ontology will be added to it. But as an alternative, the administrator can notify the GP parser to add other definitions as orphan nodes.

Table 1. State transition table of Glossary Parser FSA which is based on visual features.

State / Input	S_m	S_n	S_i where $i \neq m,n$
S (Start)	H	S	S
H (Header)	H	B	H
B (Body)	H (and Commit)	B	S (and Commit)

5 – TEXT CATEGORIZATION

As a final step of DTR process, the document which is identified as regular text document (and not other special categories) will be passed to *Text Categorization* component. We can define Text Categorization obligation as taking documents and then analyzing them to find out whether they are relevant to ontology concepts. If a document relevance to an ontology concept is more than some threshold value, it will be then assigned to that concept; otherwise, the document will be rejected.

Finding document category from a large set of concepts exist in ontology, is not a usual categorization problem. Comparing document with each concept in ontology is a time consuming process. So, we use a hierarchical categorization mechanism in which the subsystem first determines to which domain and ontology a documents belongs. If the document falls into a covered ontology, it will be then analyzed precisely for finding the exact category it belongs to.

In abstraction, document categorization process is composed of two steps: Document vectorization based on interested features of a document, and classifying the vector (document) according to some predefined (learned) classes. Document vectorization problem will reduce to a feature selection problem because when important features of documents are known, constructing a feature vector will not be a costly process. Classification itself involves two sub-problems: parameter adjusting (usually in a learning process) and classification itself. This division of categorization makes it possible to parallel development of two individual but interconnected components: feature selection and classifier components.

A major characteristic, or difficulty, of text categorization problems is the high dimensionality of the feature space. The native feature space consists of documents, which can be tens or hundreds of thousands of terms for even a moderate-sized text collection. This is prohibitively high for many learning algorithms [6]. This is the motivation of automatic feature selection. Automatic feature selection methods include the removal of non-informative terms according to corpus statistics, and selecting more informative features. The more informative features, the more discriminative border between ontology concepts could be cognized.

Regarding a comparative study of feature selection methods in statistical learning of text categorization, IG (Information Gain) yields the most increase of classification accuracy.

Information gain is frequently employed as a term-goodness criterion in the field of machine learning [7]. It measures the number of bits of information obtained for category prediction by knowing the presence or absence of a term in a document. If $C_1, C_2, ..., C_m$ are a set of categories in target space, then the information gain of term t is defined by following expression.

$$G(t) = -\sum_{i=1}^{m} \Pr(c_i) \log \Pr(c_i)$$

$$+ \Pr(t) \sum_{i=1}^{m} \Pr(c_i \mid t) \log \Pr(c_i \mid t)$$

$$+ \Pr(\bar{t}) \sum_{i=1}^{m} \Pr(c_i \mid \bar{t}) \log \Pr(c_i \mid \bar{t})$$

After gain computation, features with higher scores are selected. The feature selection process will be performed over training documents only once and will be repeated if training documents have been updated. Once the features have been extracted (in the form of terms), vectorization becomes a simple problem. Counting selected terms over a document and storing number in a vector sequentially, results in the corresponding vector of the document.

Feature selection process can be improved by some heuristics. We know that information gain of common terms are very low and have no chance in selection process. So, we have added a preprocessing module to feature selection component. In preprocessing, stop words (like articles, auxiliary verbs and so on) are eliminated from documents. In special domains, other general terms in that domain (e.g. 'system' in computer domains) can be appended to the list of stop words.

Text classification is the task of assigning text documents to pre-specified classes [8]. Analysis and experimental results have shown that Centroid-based document classification has better performance in contrast with other classification methods [9].

In Centroid-based classification, each document d is considered to be a vector in the term-space. Each document represented by the *term-frequency* (TF) vector $dtf = (tf1, tf2, \ldots, tfn)$, where tfi is the frequency of the i-th term in the document. In the vector-space model, the similarity between a document d and the centroid of class C is commonly measured using the cosine function, given by:

$$\cos(d, C) = \frac{d \cdot C}{\| d \|_2 * \| C \|_2} = \frac{d \cdot C}{\| C \|_2}$$

where centroid of class C is calculated as:

$$C = \frac{1}{| S |} \sum_{d \in S} d$$

These centroids are calculated over the training documents during learning phases. They will be then updated only if training documents change. Class of a new coming document x is determined as follows:

$$\arg\max_{j=1...k} \cos(\vec{x}, \vec{c}_j)$$

6 – EXPERIMENTAL RESULTS

The offline system proposed here, has been incorporated in a question-answering system for telecommunications domain, called TeLQAS (Telecommunication Literature Question Answering System) [10]. A prototype of TeLQAS, before and after considering the proposed offline system, has been implemented for the Optical Technology. For assessing the performance of TelQAS, we manually stored 200 text documents related to the 80 concepts of the existing concepts in the ontology graph in the local database. Then, 100 sample questions related to this domain submitted to the system, where some of the questions were in the noisy form (i.e. having syntactical or grammatical errors).

The average of the assessment parameters based on TREC (i.e. precision, recall, and accuracy), has been brought in table 1. The results in the first row have been calculated before the system tries to automatically extract the documents information in an offline process. In the second row, the average values have been calculated after the incorporation of our proposed offline serivce – composed of DTR, Glossary Parser and Text Categorizaiton components. Comparing the first and second rows, determines that the accuracy has been reasonably increased after the offline process performed.

It is important to notice that the values in the second row have been obtained as a result of applying all the information extractor components. However, components like GP leads increase the accuracy of exact answers while some components like Text Categorization lead to an increase in the accuracy of acceptable answers.

In addition, the overall time elapsed for the process has had a noticeable drop, as there have been lots of documents and text fragments ready before the process of retrieval starts.

Table1. The average of TREC assessment parameters for 200 documents and 100 questions

	Precision	Recall	Accuracy
Without offline process	74%	88%	81%
With offline process	82%	95%	88%

7 – CONCLUDING REMARKS AND FUTURE WORK

As the experimental results show, the performance of a question-answering system could be increased using an offline service like the one proposed here. Increase in performance is the result of more acceptable answers and more direct answers which are the result of offline process themselves. Some parts of offline process like Glossary Parser component enrich the system ontology by the mean of relations and slots. This enrichment leads to more accuracy in direct answers. Some other components like Text Categorization enrich the system ontology by the mean of related documents and corresponding relations to concepts.

This later type of enrichment causes more acceptable answers.

As indicated before, an NLP component can be uses in parallel with Text Categorization and other components of this offline system. By adding more text processing module covering different type of documents, the more information can be extracted for the ontology. For example, a component which is able to process the technical data sheets can extract the attributes and properties of domain concepts.

REFERENCES

1. Farquhar, R. Fikes, & J. Rice. The Ontolingua Server: A Tool for Collaborative Ontology Construction. Knowledge Systems Laboratory, 1996.
2. J. Gennari, M. A. Musen, R. W. Fergerson, W. E. Grosso, M. Crubézy, H. Eriksson, N. F. Noy, S. W. Tu The Evolution of Protégé: An Environment for Knowledge-Based Systems Development. 2002.
3. N. F. Noy, M. Sintek, S. Decker, M. Crubezy, R. W. Fergerson, & M. A. Musen. Creating Semantic Web Contents with Protege-2000. *IEEE Intelligent Systems* 16(2):60-71, 2001.
4. JSrg-Uwe Kietz and Raphael Volz, Extracting a Domain-Specific Ontology from a Corporate Intranet, Proceedings of CoNLL-2000 and LLL-2000, pages 167-175, Lisbon, Portugal, 2000.
5. T.R. Gruber and Olsen G.R. An ontology for engineering mathematics. Technical Report KSL-94-18, Knowledge Systems Laboratory, Stanford University, Palo Alto, CA., 1994.
6. Y. Yang, J.O. Pederson. A Comparative Study on Feature Selection in Text Categorization. 1997.
7. T. Mitchell, Machine Learning. McGRAW Hill, 1996.
8. Y. Yang and X. Liu. A re-examination of text categorization methods. In SIGIR-99, 1999.
9. E.H. Han and G. Karypis Centroid-Based Document Classification: Analysis & Experimental Results. University of Minnesota, 2000.
10. M. R. Hejazi, M.S.Mirian, K. Neshatian, A.Jalali, and B.R. Ofoghi, TeLQAS: A Telecommunication Literature Question/Answering Benefits from a Text categorization Mechanism, Accepted for Publication in the Proceedings of IKE'03, July 2003, USA.
11. Boris Katz, Jimmy Lin, and Sue Felshin. Gathering Knowledge for a Question Answering System from Heterogeneous Information Sources. Proceedings of the ACL 2001 Workshop on Human Language Technology and Knowledge Management, July, 2001.
12. William Hersh, TREC Genomics Track – Background, Oregon Health & Science University, November 2003.
13. Vinay Chaudhri and Richard Fikes, *Question Answering Systems*, AAAI Fall Symposium, November 1999.
14. Jeffrey D. Ullman, *Principles of Database & Knowledge-Base Systems Vol. 1*, December 1988
15. Drew V. McDermott: Comments on Cheeseman: why plausible reasoning. Computational Intelligence 4: 91-92 (1988)
16. Kourosh Neshatian, Mahmoud R. Hejazi. *An Object Oriented Ontology Interface for Information Retrieval Purposes in the Domain of Telecommunications*, International Symposium on Telecommunications 2003.
17. Sabrina T., Rosni A., T. Enyakong Extending Ontology Tree Using NLP Techniques, 2000
18. Mahmoud R. Hejazi, Kourosh Neshatian, Bahador R. Ofoghi, *A System Developed for Automatic Extraction and Categorization of Telecommunication Literatures Used in TeLQAS,* International Symposium on Telecommunications, Augost 2003.