CL Research's Knowledge Management System

Kenneth C. Litkowski
CL Research
9208 Gue Road
Damascus, MD 20872
ken@clres.com
http://www.clres.com

Abstract

CL Research began experimenting with massive XML tagging of texts to answer questions in TREC 2002. In DUC 2003, the experiments were extended into text summarization. Based on these experiments, The Knowledge Management System (KMS) was developed to combine these two capabilities and to serve as a unified basis for other types of document exploration. KMS has been extended to include web question answering, both general and topic-based summarization, information extraction, and document exploration. The document exploration functionality includes identification of semantically similar concepts and dynamic ontology creation. As development of KMS has continued, user modeling has become a key research issue: how will different users want to use the information they identify.

1 Introduction

In participating the TREC question-answering track, CL Research began by parsing full documents and developing databases consisting of semantic relation triples (Litkowski, 1999). The database approach proved to be quite confining, with time requirements expanding exponentially trying to maintain larger sets of documents and increasingly complex procedures to answer questions. A suggestion was made to tag text with the type of questions they could answer (e.g., tagging time phrases as answering **when** questions and person names as answering **who** questions). This led to the general approach of analyzing parse trees to construct an XML representation of texts (i.e., attaching metadata to the text) and examining these representations with XPath expressions to answer questions.

Litkowski (2003a) demonstrated the viability of this approach by showing that XPath expressions could be used to answer questions at a level above the highest performing team. Many issues and problems were identified: (1) The necessary level of analysis to meet the needs of particular applications; (2) tagging alternatives; and (3) the viability of the using the XML representation for text summarization, information extraction, novelty detection, and text mining. Subsequent efforts showed that XML representations could be effectively used in summarization (Litkowski, 2003b) and novelty detection (Litkowski, 2005).

Initially, CL Research developed an interface for examining question-answering performance. This interface has since evolved into a Knowledge Management System (KMS) that provides a single platform for examining English documents (e.g., newswire and research papers) and for generating different types of output (e.g., answers to questions, summaries, and document ontologies), also in XML representations. In this demonstration, CL Research will describe many parts of KMS, particularly the approaches used for analyzing texts.[1] The demonstration will particularly focus on the value of XML in providing a flexible and extensible mechanism for implementing the various NLP functionalities. In addition, the demonstration will identify the emerging issue of user modeling to determine exactly how knowledge will be used, since

[1] Screen shots of KMS in performing the functions as described below are can be seen at http://www.clres.com/kmsscreen.html.

Proceedings of the ACL Interactive Poster and Demonstration Sessions,
pages 13–16, Ann Arbor, June 2005. ©2005 Association for Computational Linguistics

the primary purpose of KMS is to serve as a tool that will enable users (such as scientists and intelligence analysts) to accumulate and manage knowledge (including facts, such as described in Fiszman et al., 2003) about topics of interest.[2]

2 Parsing and Creation of XML Tagging

KMS and each of its application areas is based on parsing text and then transforming parse trees into an XML representation. CL Research uses the Proximity Parser, developed by an inventor of top-down syntax-directed parsing (Irons, 1961).[3] The parser output consists of bracketed parse trees, with leaf nodes describing the part of speech and lexical entry for each sentence word. Annotations, such as number and tense information, may be included at any node. (Litkowski (2002) and references therein provide more details on the parser.)

After each sentence is parsed, its parse tree is traversed in a depth-first recursive function. During this traversal, each non-terminal and terminal node is analyzed to identify discourse segments (sentences and clauses), noun phrases, verbs, adjectives, and prepositional phrases. These items are maintained in lists; the growing lists constitute a document's discourse structure and are used, e.g., in resolving anaphora and establishing coreferents (implementing techniques inspired by Marcu (2000) and Tetreault (2001)). As these items are identified, they are subjected to a considerable amount of analysis to characterize them syntactically and semantically. The analysis includes word-sense disambiguation of nouns, verbs (including subcategorization identification), and adjectives and semantic analysis of prepositions to establish their semantic roles (such as described in Gildea & Jurafsky, 2002).

When all sentences of a document have been parsed and components identified and analyzed, the various lists are used to generate the XML representation. Most of the properties of the components are used as the basis for establishing XML attributes and values in the final representation. (Litkowski 2003a provides further details on this process.) This representation then becomes the basis for question answering, summarization, information extraction, and document exploration.

The utility of the XML representation does not stem from an ability to use XML manipulation technologies, such as XSLT and XQuery. In fact, these technologies seem to involve too much overhead. Instead, the utility arises within a Windows-based C++ development environment with a set of XML functions that facilitate working with node sets from a document's XML tree.

3 Question Answering

As indicated above, the initial implementation of the question-answering component of KMS was designed primarily to determine if suitable XPath expressions could be created for answering questions. CL Research's XML Analyzer was developed for this purpose.[4] XML Analyzer is constructed in a C++ Windows development environment to which a capability for examining XML nodes has been added. With this capability, a document can be loaded with one instruction and an XPath expression can be applied against this document in one more instruction to obtain a set of nodes which can be examined in more detail. Crucially, this enables low-level control over subsequent analysis steps (e.g., examining the text of a node with Perl regular expressions).

XML Analyzer first loads an XML file (which can include many documents, such as the "top 50" used in TREC). The user then presents an XPath expression and discourse components (typically, noun phrases) satisfying that expression are returned. XML Analyzer includes the document number, the sentence number, and the full sentence for each noun phrase. Several other features were added to XML Analyzer to examine characteristics of the documents and sentences (particularly to identify why an answer

[2]The overall design of KMS is based on requirements enunciated by intelligence analysts and question-answering researchers in a workshop on Scenario-Based Question Answering sponsored by the Advanced Research and Development Agency in 2003.

[3]An online demonstration of the parser is available at http://www.zzcad.com/parse.htm. A demo version of the parser is available for download at http://www.clres.com/demos.html.

[4]A demo version of XML Analyzer is available for download at http://www.clres.com/demos.html.

wasn't retrieved by an XPath expression).

XML Analyzer does not include the automatic creation of an XPath expression. KMS was created for TREC 2003 as the initial implementation of a complete question-answering system. In KMS, the question itself is parsed and transformed into an XML representation (using the same underlying functionality for processing documents) and then used to construct an XPath expression.

An XPath expression consists of two parts. The first part is a "passage retrieval" component, designed to retrieve sentences likely to contain the answer. This basic XPath is then extended for each question type with additional specifications, e.g., to ask for noun phrases that have time, location, or other semantic attributes. Experiments have shown that there is a tradeoff involved in these specifications. If they are very exacting, few possible answers are returned. Backoff strategies are used to return a larger set of potential answers and to analyze the context of these potential answers in more detail. The development of routines for automatic creation of XPath expressions is an ongoing process, but has begun to yield more consistent results (Litkowski, 2005).

In preparation for TREC 2004, KMS was further extended to incorporate a web-based component. With a check box to indicate whether the web or a document repository should be used, additional functionality was used to pose questions to Google. In web mode, an XML representation of a question is still developed, but then it is analyzed to present an optimal query to Google, typically, a pattern that will provide an answer. This involves the use of an integrated dictionary, particularly for creating appropriate inflected forms in the search query. KMS only uses the first page of Google results, without going into the source documents, extracting sentences from the Google results and using these as the documents. (A user can create a new "document repository" consisting of the documents from which answers have been obtained.) Many additional possibilities have emerged from initial explorations in using web-based question answering.

4 Summarization

Litkowski (2003a) indicated the possibility that the XML representation of documents could be used for summarization. To investigate this possibility, XML Analyzer was extended to include summarization capabilities for both general and topic-based summaries, including headline and keyword generation. Summarization techniques crucially take into account anaphora, coreferent, and definite noun phrase resolutions. As intimated in the analysis of the parse output, the XML representation for a referring expression is tagged with antecedent information, including both an identifying number and the full text of the antecedent. As a result, in examining a sentence, it is possible to consider the import of all its antecedents, instead of simply the surface form.

At the present time, only extractive summarization is performed in KMS. The basis for identifying important sentences is simply a frequency count of its words, but using antecedents instead of referring expressions. Stopwords and some other items are eliminated from this count.

In KMS, the user has the option for creating several kinds of summaries. The user specifies the type of summary (general, topic-based, headline, or keyword), which documents to summarize (one or many), and the length. Topic-based summaries require the user to enter search terms. The search terms can be as simple as a person's name or a few keywords or can be several sentences in length. Topic-based summaries use the search terms to give extra weight to sentences containing the search terms. Sentences are also evaluated for their novelty, with redundancy and overlap measures based on examining their noun phrases. KMS summarization procedures are described in more detail in Litkowski (2003b); novelty techniques are described in Litkowski (2005).

In KMS, summaries are saved in XML files as sets of sentences, each characterized by its source and sentence number. Each summary uses XML attributes containing the user's specifications and the documents included in the search. generated quickly but in whole form.

5 Document Exploration

KMS includes two major components for exploring the contents of a document. The first is based on the semantic types attached to nouns and verbs. The second is based on analyzing noun phrases to construct a document hierarchy or ontology.

As noted above, each noun phrase and each verb is tagged with its semantic class, based on WordNet. A user can explore one or more documents in three stages. First, a semantic category is specified. Second, the user pushes a button to obtain all the instances in the documents in that category. The phraseology in the documents is examined so that similar words (e.g., plurals and singulars and/or synonyms) are grouped together and then presented in a drop-down box by frequency. Finally, the user can select any term set and obtain all the sentences in the documents containing any of the terms.

KMS provides the capability for viewing a "dynamic" noun ontology of a document set. All noun phrases are analyzed into groups in a tree structure that portrays the ontology that is instantiated by these phrases. Noun phrases are reduced to their base forms (in cases of plurals) and grouped together first on the basis of their heads. Synonym sets are then generated and a further grouping is made. Algorithms from Navigli & Velardi (2004) are being modified and implemented in KMS. The user can then select a node in the ontology hierarchy and create a summary based on sentences containing any of its terms or children.

6 Dictionaries and Thesauruses in KMS

KMS makes extensive use of integrated dictionaries and thesauruses, in addition to a comprehensive dictionary used in parsing (which makes use of about 30 subcategorization patterns for verbs). This dictionary is supplemented with other dictionaries that are first used in dynamically extending the parser's dictionary for parsing, but then more extensively in semantic analysis. WordNet is used for many functions, as is a Roget-style thesaurus. KMS also uses a full machine-readable dictionary, dictionaries and semantic networks from the Unified Medical Language System, and a specially constructed dictionary of prepositions for semantic role analysis.

7 Summary

The preceding sections have focused on particular prominent functionalities (question-answering, summarization, and document exploration) in KMS. Each of these components is part of the whole, in which the main objective is to allow a user to explore documents in a variety of ways to identify salient portions of one or more documents. KMS is designed to identify relevant documents, to build a repository of these documents, to explore the documents, and to extract relevant pieces of information.

References

Fiszman, M., Rindflesch, T., & Kilicoglu, H. (2003). Integrating a Hypernymic Proposition Interpreter into a Semantic Processor for Biomedical Texts. Proceedings of the AMIA Annual Symposium on Medical Informatics.

Gildea, Daniel, and Daniel Jurafsky. (2002) Automatic Labeling of Semantic Roles. *Computational Linguistics, 28* (3), 245-288.

Irons, E. T. (1961) A Syntax Directed Compiler for ALGOL-60. *Communications of the ACM, 4*, 51-55.

Litkowski, K. C. (1999). Question Answering Using Semantic Relation Triples. In E. M. Voorhees & D. K. Harman (eds.), *The Eighth Text Retrieval Conference (TREC-8)*. NIST Special Publication 500-246. Gaithersburg, MD., 349-56.

Litkowski, K. C. (2002). CL Research Experiments in TREC-10 Question-Answering. In E. M. Voorhees & D. K. Harman (eds.), *The Tenth Text Retrieval Conference (TREC 2001)*. NIST Special Publication 500-250. Gaithersburg, MD., 122-131.

Litkowski, K. C. (2003a). Question Answering Using XML-Tagged Documents. In E. M. Voorhees & L. P. Buckland (eds.), *The Eleventh Text Retrieval Conference (TREC 2002)*. NIST Special Publication 500-251. Gaithersburg, MD., 122-131.

Litkowski, K. C. (2003b). Text Summarization Using XML-Tagged Documents. Available: http://nlpir.nist.gov/projects/duc/pubs.html.

Litkowski, K. C. (2005). Evolving XML and Dictionary Strategies for Question Answering and Novelty Tasks. Available: http://trec.nist.gov/pubs/trec13/t13_proceedings.html.

Marcu, Daniel. (2000) The Rhetorical Parsing of Unrestricted Texts: A Surface-based Approach. *Computational Linguistics, 26* (3), 395-448.

Navigli, R. & P. Velardi (2004) Learning Domain Ontologies from Document Warehouses and Dedicated Web Sites. *Computational Linguistics 30*, 151-180.

Tetreault, Joel. (2001) A Corpus-Based Evaluation of Centering and Pronoun Resolution. *Computational Linguistics, 27* (4), 507-520.