Colander: Sifting Documents for Special Terms

Shridhara Aithal
Applied Research
Wipro Technologies
Bangalore 560068, India
shridhara.aithal@wipro.com

Sudarshan Murthy
Applied Research
Wipro Technologies
Bangalore 560068, India
sudarshan.murthy@wipro.com

Abstract

We propose to demonstrate *Colander*, a set of lightweight approaches, and a tool, to mine “special terms” from a corpus of documents. In this proposal, we give an overview of three high-level approaches to extracting special terms; introduce some metrics to measure the performance of the approaches; and outline an evaluation methodology. We also provide a high-level description of the tool and its features. In this proposal, we illustrate the approaches and the tool in an application to mine both known and candidate trademarks used in documents.

1. Introduction

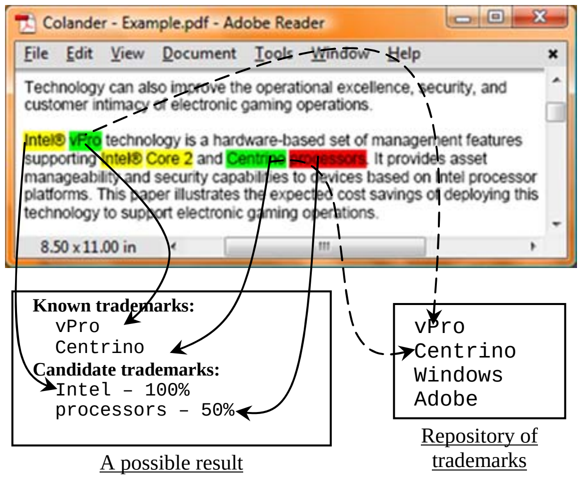
As organizations increasingly capture knowledge in document form (as against structured database form), carrying out certain knowledge-management activities requires new approaches. For example, identifying the intellectual properties (such as trademarks and product names) cited in documents, or extracting cross-references in patent applications, require new approaches.

We consider identifying intellectual properties and cross-referenced patents as instances of extracting “special terms”, where “special terms” vary across applications. Unsurprisingly, both text mining [8] and information-retrieval (IR) [1] approaches are relevant in identifying special terms, but the traditional text mining and IR approaches can be heavyweight for some applications.

We propose to demonstrate *Colander*, a set of distinct *lightweight* approaches, and a tool, to mine special terms from a corpus of documents. Colander can be used to extract many kinds of special terms, but, for illustration purposes only, we limit the discussion in this proposal to mining trademarks cited in documents. (A motivation to mine trademarks is to ensure that each trademark is attributed to its owner).

Figure 1 illustrates the use of Colander to mine trademarks. It highlights some known and candidate trademarks that appear in a source document. A trademark is *known* if it exists in a *repository* of trademarks. (In general, Colander can consult repositories of known special terms in applications where such consulting is possible.)

A trademark is a *candidate* if it is not known and if the corresponding term might be a trademark. A confidence factor is associated with each candidate trademark.



Technology can also improve the operational excellence, security, and customer intimacy of electronic gaming operations.

Intel® vPro technology is a hardware-based set of management features supporting Intel® Core 2 and Centrino processors. It provides asset manageability and security capabilities to devices based on Intel processor platforms. This paper illustrates the expected cost savings of deploying this technology to support electronic gaming operations.

Figure 1: Screenshot of a document and sample result

Figure 1 shows *one possible* result of mining trademarks: The terms vPro and Centrino are shown as known trademarks because those terms exist in the repository. Intel is shown as a candidate trademark with 100% confidence because the sign ® follows that term. The term “Core 2” is a candidate trademark, but it is not recog-

15th International Conference on Management of Data
COMAD 2009, Mysore, India, December 9–12, 2009
©Computer Society of India, 2009

nized. Finally, “processors” is not a candidate trademark, but it is recognized as one.

We have two high-level goals for Colander: recall 100% of the known trademarks; and achieve a “high” F-measure for candidate trademarks. The first goal is easily achieved, but the effort to achieve the goal varies based on the approach. The second goal is harder and heavily dependent on the approach used.

We consider two approaches for extracting known trademarks: string search and *n-grams* [10] constructed from input document. For the purpose of this demonstration, we illustrate the use of n-grams for extracting known trademarks. We offer *three* distinct approaches, each with an analytical model, to extract candidate trademarks.

The rest of this proposal is organized as follows: Section 2 gives an overview of the three analysis approaches. Section 3 describes the evaluation approach. Section 4 outlines the demonstration requirements. Section 5 briefly reviews related work, and Section 6 presents some concluding remarks.

2. Analysis Approaches

In this section, we describe the approaches we support to mining known and candidate trademarks from documents.

We support three analysis approaches: lexical, grammar, and linguistic. Each approach defines an analytical model and uses a different strategy to extract both known and candidate trademarks. For brevity, we omit describing the analytical models, and provide only an overview of the strategy in each approach.

At a high-level, in each approach, we extract known trademarks as follows: construct n-grams from words in the input document; and lookup each n-gram in the repository of trademarks. Though this procedure applies to all three analysis approaches, the effort to extract known trademarks varies among the approaches.

In the rest of this section, we discuss how the effort to extract known trademarks varies across the approaches. We also discuss how the approaches differ in extracting candidate trademarks.

In the rest of this proposal, we use the symbol L to denote the length (that is, the number of words) of the longest trademark in the repository of trademarks.

2.1 Lexical Analysis

Lexical analysis parses the text in a document and employs a sequence of heuristics to mine trademarks. We distinguish two methods within this approach: word-oriented analysis and sentence-oriented analysis.

Word-oriented analysis: In this naïve method, we first transform a document to a stream of words. Then, from the stream of words, we construct all n-grams of length between 1 and L, and use the n-grams to identify the known trademarks used in the document. Because the analysis is word-oriented, the n-grams constructed can span sentences, potentially increasing the number of false positives in the result.

To identify candidate trademarks, we apply a sequence of heuristics (the same heuristics Acrophile [9] uses) to each word in the input document. We apply heuristics only if the word is not part of an n-gram of a known trademark.

Sentence-oriented analysis: This method is based on the premise that a trademark cannot span sentences. Thus, in this method, we detect known trademarks using n-grams of length between 1 and L constructed only from words in a single sentence. This strategy considerably reduces the number of n-grams constructed (thereby speeding up the analysis) and also reduces the number of false positives in the result.

Both the grammar and linguistic approaches identify the known trademarks by employing same technique used by sentence-oriented analysis. Thus, all these three approaches have the same performance characteristics with respect to extracting known trademarks.

We obtain candidate trademarks using the same strategy employed in word-oriented analysis. Thus, both sentence-oriented and word-oriented lexical analyses have the same performance characteristics with respect to extracting candidate trademarks.

We emphasize that not all applications involving extraction of “special terms” might be amenable to both word-oriented and sentence-oriented lexical analyses. Also, not all applications might benefit from the sentence-oriented approach. However, we support both methods so that they can be employed where appropriate.

2.2 Grammar Analysis

Grammar analysis builds on sentence-oriented lexical analysis to let applications exploit the grammatical structure of a sentence when extracting special terms. For example, one might consider only nouns and noun sequences when extracting trademarks.

Grammar analysis first determines the part of speech (POS) of each word in a sentence, *independent* of the other words in the sentence. It then looks for specific patterns of parts of speech (such as noun sequences) to identify special terms.

A lexicon (such as WordNet [5]) may be used to determine the POS of a word as long as it is able to determine the POS of a word on the basis of the word alone, using no context information. The lexicon might assess zero or more POS to a word. If it assesses no POS to a word, it is because the word does not exist in the lexicon.

In the trademark application, we treat a word with multiple POS as a noun if noun is one of the POS associated with the word. We also assume that a word without a POS is a noun. Then, we analyze only *noun sequences*.

We identify the candidate trademarks using the same sequence of heuristics used in sentence-oriented lexical analysis, but only over each noun. This strategy provides significant speed up over lexical analysis, but it can reduce recall of candidate trademarks.

Grammar analysis has one key advantage over the other two analysis approaches: Its performance can improve when used over a document corpus, and the improvement itself can grow as the corpus gets larger. This phenomenon is due to the ability to reuse POS information for different documents in the corpus. The reuse itself is possible because the POS information for each word is independent of its use context.

2.3 Linguistic Analysis

Linguistic analysis is a highly context-sensitive approach to mining special terms. It employs intimate knowledge of the source language, accounting for factors such as polysemy, word frequency, word placement, the context of words, and formatting. For example, in the trademark application, this approach can determine that the word “technology” cannot be a candidate trademark because that word is too common. An application to extract acronyms might identify a parenthetical word placed immediately after some italicized text as a candidate acronym.

Linguistic analysis depends heavily on POS taggers that consider the structure of the containing sentence (or a document) while determining the POS of each word in the sentence. Stanford POS tagger [13, 14] is one such tagger.

When determining candidate trademarks, the trademark application considers formatting information (such as italics and font name), in addition to using a sequence of heuristics over words in noun phrases. Thus, when formatting conventions are known and used, recall is likely to be higher for candidate trademarks.

3. Evaluation Approach

In this section, we give an overview of an implementation of Colander, introduce the metrics to evaluate performance, and summarize an approach to empirical evaluation.

Implementation: Colander is implemented as a Microsoft® Windows® application using Microsoft Visual Basic®. Currently, Colander uses Microsoft Word to parse a document, but this dependency can be easily removed. (Our implementation approach has been to reuse existing components as much as possible.)

Colander is designed to extract special terms from a document corpus, not just a single document. Currently, the corpus to be processed should be placed in a directory, but we expect to provide means of processing together documents whose locations are distributed. Regardless of how the document corpus is conveyed, the algorithms we employ in Colander complete the analysis of each document in just one pass. (Caveat: A POS tagger used in linguistic analysis might make a full pass over a document to complete tagging. However, none of our algorithms require more than one pass over a document.)

In the trademark application, we presort the repository of known trademarks alphabetically and partition it by the number of words in the trademarks. At initialization time, we load each partition into a separate list. At runtime, to determine if an n-gram is a known trademark, we perform binary search over the list that contains trademarks whose length is the same as the length of the n-gram.

For grammar analysis, Colander uses WordNet and the WordNet COM server [2] to determine the POS of a word. (The trademark application only needs to know if a word can be a noun.) It caches the results of POS determination and reuses it over a document corpus. For efficiency, it also maintains a predefined list of frequently used words (such as the articles and some prepositions) and their POS. The POS cache and the list of frequent words reduce the cost of consulting WordNet.

Evaluation Metrics: We use the Big-O notation to represent the complexity of the algorithms the three analysis approaches employ. We estimate time complexity analytically and verify it empirically.

We use the common IR metrics precision, recall, and F-measure to determine the effectiveness of the analysis approaches. We also use the number of n-grams processed as a key metric because that number significantly influences the speed of performance.

Empirical Analysis: We carry out empirical analysis in two parts. First, we ask humans to identify known and candidate trademarks from a *gold set* of documents. We then analyze the same set of documents using the different approaches in Colander and compare the two sets of results.

We use different sets of documents to empirically verify the analytical models of each approach. We also use document sets of different sizes to verify scalability.

4. Demonstration Setup

During the demonstration, we intend to showcase the utility of Colander in extracting trademarks from a representative document corpus we have crafted. The demonstration will be designed to illustrate the performance of each analysis approach and its efficacy compared to the manual process.

We intend to demonstrate Colander on one of our own portable computers, but it would be helpful to connect a large external monitor so that the demonstration is easily visible to the audience. It would also be helpful if we can display a large poster.

The demonstration would not require connection to the Internet.

5. Related Work

Tools such as UIMA [6] focus on performing semantic analysis of unstructured data to create and store structured information. This approach is useful in applications that require repeated access to the same information. An example is a search engine such as IBM® OmniFind™ [7].

Techniques (such as GATE [4]) which use natural language processing (NLP) for information extraction, are extremely sophisticated and tend to be heavyweight.

Much work [9, 12, 15] related to automatic extraction of acronyms and abbreviations (and their definitions) exists, but such work is different from our approach in two key ways: First, they tend to be specific to extracting acronyms. Second, they tend to be domain specific. For example, significant work [3, 11] has been done to find acronyms and their definitions in the biomedical domain.

In contrast to the aforementioned systems, our approach is lightweight and it allows developers to choose the sophistication that is appropriate for their applications. (Use just word-oriented lexical analysis or use sophisticated linguistic analysis.) Our approach is generic enough to extract any kind of "special terms". Finally, our approach is also quite simple to implement.

6. Summary

We have proposed to demonstrate Colander, a set of abstract lightweight approaches to extract special terms from a document corpus. We have introduced three approaches Colander supports to mine special terms from a document corpus. We have also outlined our evaluation approach and briefly reviewed related work.

In this proposal, we have illustrated the use of Colander in an application to extract known and candidate trademarks, but Colander can be used to extract several kinds of special terms.

References

1. Baeza-Yates, R., Ribeiro-Neto, B. 1999. Modern Information Retrieval. Addison Wesley. ISBN: 020139829X

2. Bou, B., Champollion, L. 2007. WordNet COM Server: http://sourceforge.net/projects/wncom/

3. Bracewell, D. B., Ren, F., Kuroiwa, S. 2006. Dealing with Acronyms in Biomedical Texts. Engineering Letters 13(2):216-224

4. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics, New Brunswick, USA

5. Fellbaum, C. 1998. WordNet: An Electronic Lexical Database. MIT Press. ISBN: 026206197X

6. Ferrucci, D., Lally, A. 2004. UIMA: an architectural approach to unstructured information processing in the corporate research environment, Natural Language Engineering, v.10 n.3-4, p.327-348

7. IBM® OmniFind™. IBM Corporation. http://www-01.ibm.com/software/data/enterprise-search/omnifind-enterprise/

8. Kao, A., Poteet, S. R. 2006. Natural language processing and text mining. Springer. ISBN: 184628175X

9. Larkey, L. S., Ogilvie, P., Price, A. M., Tamilio, B. 2005. Acrophile: an automated acronym extractor and server. In Proceedings of the fifth ACM conference on Digital libraries, Denver, USA, pp 205-214

10. Manning, C. D., Schütze, H. 1999. Foundations of statistical natural language processing. MIT Press. ISBN: 0262133601

11. Schwartz, A. S., Hearst, M. A. 2003. A Simple Algorithm for Identifying Abbreviation Definitions in Biomedical Text. In Proceedings of the Pacific Symposium on Biocomputing, Lihue, USA, pp. 451-462

12. Taghva, K., Gilbreth, J. 1999. Recognizing Acronyms and their Definitions. ISRI (Information Science Research Institute) UNLV, 1:191-198

13. Toutanova, K., Klein, D., Manning, C., Singer, Y. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In Proceedings of HLT-NAACL 2003, Edmonton, Canada, pp. 252-259

14. Toutanova, K., Manning, C. D. 2000. Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. In Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, Hong Kong, pp. 63-70

15. Yeates, S. 1999. Automatic extraction of acronyms from text. In Proceedings of the Third New Zealand Computer Science Research Students' Conference, Hamilton, NZ, pp 117-124