Domain Term Extraction and Structuring via Link Analysis

Nasreen AbdulJaleel[1]

Amazon.com
1200 12th Avenue South, Suite 1200
Seattle WA 98144-2734
nasreenaj@gmail.com

Yan Qu

Clairvoyance Corporation
5001 Baum Boulevard, Suite 700
Pittsburgh, PA 15213-1854
yqu@clairvoyancecorp.com

Abstract

Domain ontologies contain information about the important concepts in a domain, the associated attributes and the relationships between various concepts. The manual creation of domain ontologies is an expensive and time consuming process. In this paper, we present an approach to the automatic extraction of domain ontologies from domain-specific text. This approach uses dependency relations between terms to represent text as a graph. The graph based ranking algorithm HITS is used to identify domain keywords and to structure them as concepts and attributes. Experiments on two different domains, digital cameras and wines, show that our method performs well in comparison to other approaches.

Introduction

Domain ontologies contain information about the important *concepts* in a domain, the associated *attributes* and the *relationships* between various concepts. A concept can be defined as a key idea in a domain. The term concept may refer to a physical component or to a property. Each concept may have associated attributes and attribute values. A concept may be related to one or more other concepts.

In order to create a good ontology for a particular domain, it is necessary to first identify the important keywords or domain terms. This list of keywords provides an unstructured summary of the main topics in the domain. Keywords for the digital camera domain may include "camera", "battery", "flash" and "zoom" whereas keywords for the wine domain may include "red wine", "white wine", "grape" and "winery". These keywords may be further structured as concepts, associated attributes and attribute values (See Figure 1 for the structuring of these concepts and attributes in the Wine ontology http://www.daml.org/ontologies/76).

As part of our ongoing efforts to build a system that can construct an ontology from domain specific text, we have designed an algorithm that automatically identifies domain keywords and structures them into "concept-attribute value" pairs. This algorithm uses term-based statistics and semantic role information to identify these domain keywords.

Graph based algorithms have been employed with great success in many related research areas. They can be used to identify the "important" regions in a graph.

Link analysis has been demonstrated as an effective technique for identifying the important hubs and authorities of the Web [2,6]. Recently, Mihalcea et al [8,8] have applied graphical models for analyzing the links of linguistic units, such as terms and sentences. They explore the linkage of terms with a text window for extracting important terms and present promising empirical results. Motivated by this work, we further explore whether dependency structure provides better graphs for extracting domain terms and whether the hubs and authority scores of nodes can reflect any structuring of the terms into main concepts, attributes and values.

Previous work on structuring of terms includes taxonomy extraction [5,3] and classification of certain semantic relations [4,14,15]. Such classification generally relies on linguistic features or patterns either manually constructed or (semi-) automatically constructed based on training data. Avancini et al. present an automatic approach to expand initial domain lexicons by mining the implicit associations between terms and domains [1]. Their approach, called term categorization, employs techniques used in text categorization; however, empirical results show that the performance of term categorization is much lower than that of text categorization. Our work on term structuring can be considered as the first step toward distinguishing between primary terms and secondary terms in term pairs and toward providing input to finer classification tasks, such as identifying type-of or part-of relations.

In the following section, the approach that we use is explained in detail. The next section describes the

[1] The work was done by the first author during an internship at Clairvoyance Corporation.

evaluation mechanism. In the final section, we present a discussion of the results and error analysis.

Link Analysis of Text

One of the goals of the ontology project is to process unstructured domain text into semantically meaningful structures. Toward this end, we created a system that goes through unstructured domain specific text and identifies the important keywords. Terms are chosen as keywords based on their frequency in the domain specific text collection and also based on their semantic roles.

Another type of relation often used in information retrieval is term co-occurrence or terms occurring within a fixed-size window of each other. Statistics for this kind of relation are much easier to measure than dependency relations since they do not require parsing of the text and hence are also not prone to parsing errors.

A significant proportion of domain-specific terms are noun phrases or compound nouns as observed in [12]. Based on this observation, we decided to study noun phrases for this preliminary study. Therefore, of all the term dependency relations we could have used in our experiments, we chose head noun-modifier relations.

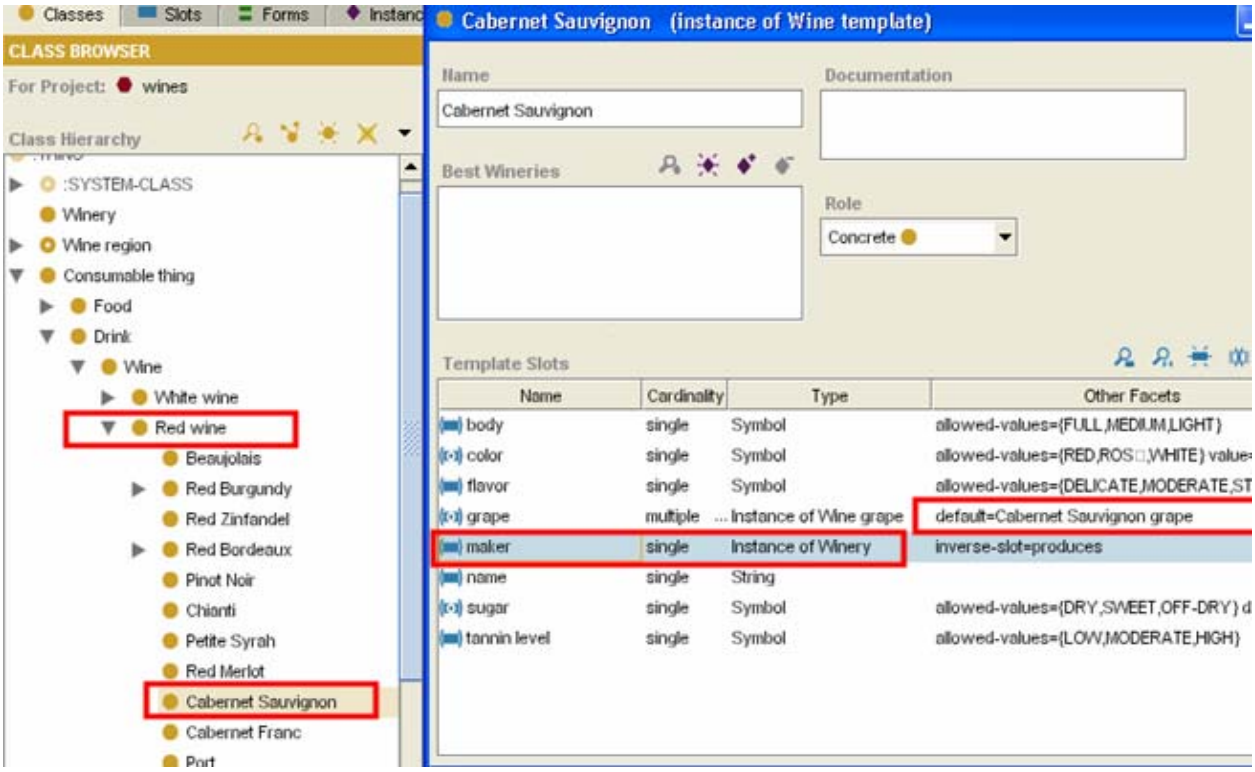

Figure 1. Structure of concepts and attributes in an online wine ontology displayed using the ontology viewer Protege

There are many kinds of relations between words in domain-specific natural language text that may be used to discover important domain terms. Some of these relations are term dependency relations. Dependencies are motivated by grammatical function, i.e., both syntactically and semantically. A word depends on another either if it is a complement or a modifier of the latter.

Term dependencies can be identified using a dependency parser. The parser that we use in our experiments is MINIPAR [10] and it is described in detail in a later section. The dependency relationships that can be extracted include *head noun-modifier*, *subject-verb*, *verb-object* etc.

Our hypotheses for detecting concepts and attributes are as follows:

- A term that is modified often and in many different ways is an important "concept". We define a concept as being a central idea in a domain. We see the number of ways a term is modified as a measure of the importance of that term. A term that has numerous modifiers is probably a basic concept.

- A term that occurs often as a modifier of one or more concepts is an important "attribute" of the concepts that it modifies. Terms that are used to modify or

describe concepts are probably important properties in the domain

In order to test our hypotheses and to see if it was portable across domains, we had to create or find two or more domain specific corpora. The process of collection creation and text processing, i.e., gathering the data, extracting the relations to be studied and using those to select possible domain terms, is described in the following section.

Once the candidate terms have been identified, they are represented as nodes and edges in a graph. A graph-based algorithm is used to rank the nodes by their importance.

After the list of concepts and attributes has been obtained, they are visualized using the tool TouchGraph [18].

Text processing

The text collections used in these experiments are collections of customer reviews in the digital camera domain from www.epinions.com and for the wine domain from http://www.klwines.com.

For each of these websites, a web crawling program was written that extracted (using wget) the html content from web pages related to the domain being studied. The html markup was cleaned and the text was extracted. The cleaned text was processed so that each sentence was on a different line and then put into an in-house XML document format.

Lexical atoms

Many important domain terms happen to be phrases and not single words. A phrase processing step is included in order to ensure that important phrases are not split up in the later steps of the algorithm. A list of the most frequent compound noun phrases in the collection is compiled. All phrases in this list are ranked by their mutual information content. Mutual information is calculated as follows:

$$\textit{Mutual Information} = P(phrase)/\Pi_{w \in phrases} P(w)$$

P(phrase) is the frequency of the phrase in the corpus and is approximated as the number of times the phrase appears in the corpus divided by the number of terms in the corpus. *P(w)* is the probability of a word appearing in the corpus

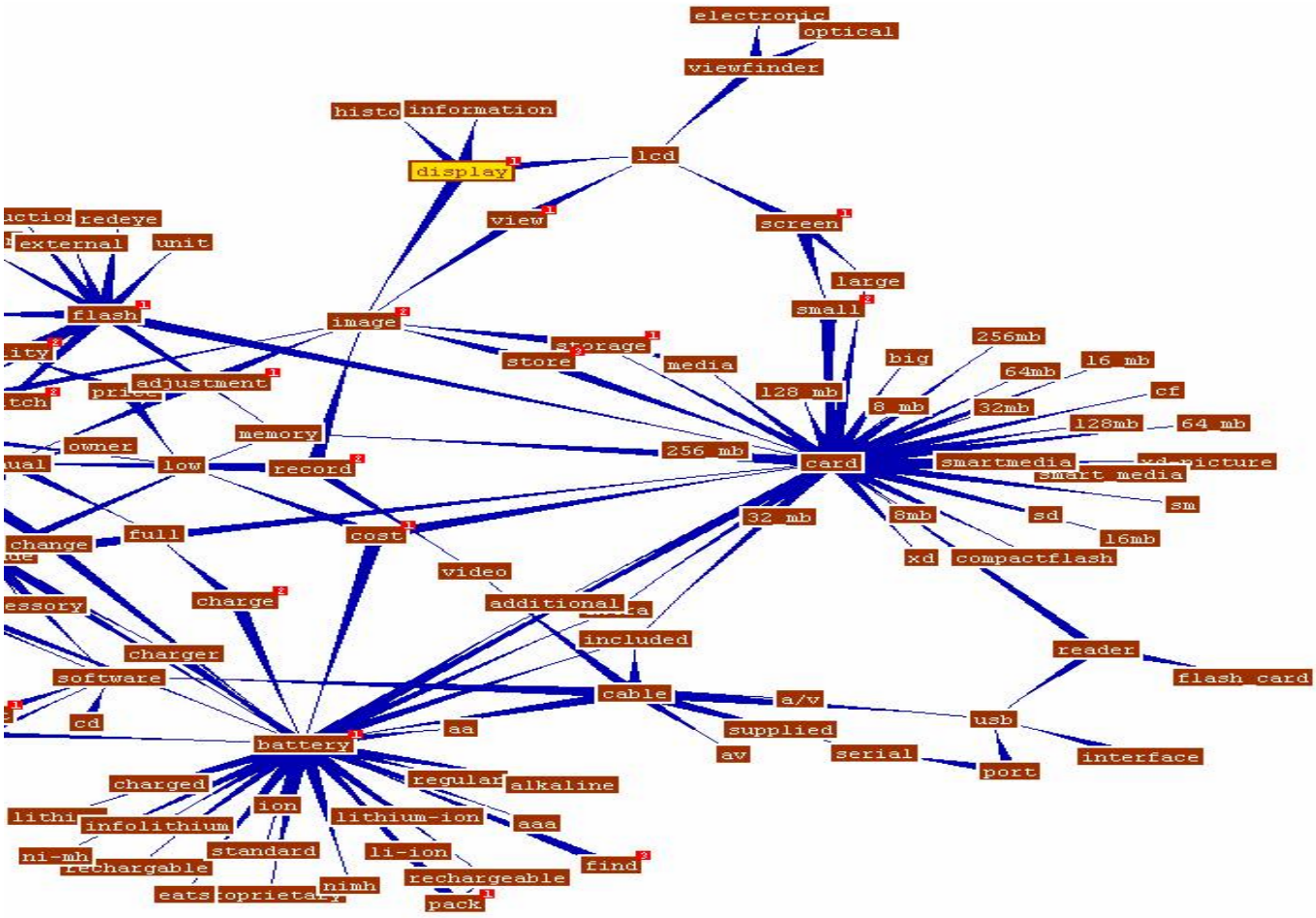

Figure 2. Structuring of attributes and concepts from the digital camera domain using TouchGraph

and is calculated as the number of times the word occurs over the number of terms in the corpus.

The top 200 phrases are replaced in the collection by a new token, which is the terms in the phrase chunked together. E.g., *Cabernet sauvignon* is replaced with *cabernet_sauvignon*.

Dependency parsing

For dependency-based parsing, we use MINIPAR, which is a statistical parser trained based on the Penn Treebank [11]. The parser is freely downloadable [10].

MINIPAR identifies dependency relations and grammatical functions. For example, for the sentence *Automatic mode is basically idiot proof*, MINIPAR produces the following parse:

```
E0    (()    fin C   *      )
1     (Automatic     ~ A    2      mod    (gov mode))
2     (mode  ~ N    3      s      (gov be))
3     (is    be VBE  E0     i      (gov fin))
4     (basically     ~ A    3      guest  (gov be))
E2          (()       mode  N   6          subj     (gov  proof)
(antecedent 2))
5     (idiot  ~ N    6      nn     (gov proof))
6     (proof  ~ N    3      pred   (gov be))
```

The parser fails in many instances when run on the test corpus because customer reviews often contains ungrammatical sentences, badly formatted text with unrecognized characters from the original Web documents, and out-of-vocabulary terms such as proper names, acronyms, technical terms etc.

As the parser is pre-compiled and detailed documentation of the parser is not available, we did not try to correct the parsing errors. We expect the laws of large numbers will help us get reasonable results.

An XML parser in Perl was developed for the MINIPAR output. The parsing program converts the data from the dependency parser into XML format. The resulting XML file stores the noun-modifier, subject-verb and verb-object relations found in the collection. The program analyzes each sentence and extracts the relevant relations being studied. These relations are accumulated in a hash table. After all the text has been analyzed, the information is written out in the following XML format.

The parent array **<terms>** is a collection of many **<term>** entries. Each entry of the type **<term>** has the following associated fields:

- **<name>** : The root form of this term

Each of the following has a **<name>** subfield with the associated root form and a **<count>** subfield with the number of terms the relation was observed

1. **<subject_verb>** : The root form of a verb that has this term as a subject
2. **<object_verb>** : The root form of a verb that has this term as an object
3. **<modifier>** : The root form of a term used to modify this term
4. **<target>** : The root form of a term that is modified by this term

A snippet of the XML file that is produced by the program is shown below.

```
<terms>
    <term>
     <name>mode</name>
        <subject_verb><name>is</name>
              <count>378</count></subject_verb>
        …
        <object_verb><name>select</name>
             <count>32</count></object_verb>
        …
        <modifier><name>automatic</name>
           <count>727</count></modifier>
        <modifier><name>manual</name>
                  <count>578</count></modifier>
        …
   </term>
   <term>…</term>
   …
</terms>
```

The XML file generated as described above is used as the input to the next step in the algorithm.

Graphical representation of dependencies

The dependencies extracted from the collection are represented as nodes and edges in a graph. Using a diagram to represent text has a number of advantages. When we examine the graphical representation of text we utilize our visual cognitive apparatus which has an advantage over our text-reading abilities. The two-dimensional representation of a diagram is a lot more expressive than text, which is typically scanned from the left to the right and the top to the bottom.

Diagrams can be viewed following different directions to gain distinct insights, while the use of a larger symbol set makes them more expressive. In addition, one can obtain different levels of detail from the same diagram: a bird's eye view will easily convey a system's structure, while examining a class in detail can reveal its collaborators. Finally, a diagram can allow us to identify patterns; again

two-dimensional pattern-matching is an activity we humans are particularly good at.

A compelling reason for us to use a graphical representation for these dependencies is that this allows the use of graph-based ranking algorithm to identify the important sections of the graph.

As mentioned previously in this report, we chose noun phrases for this preliminary study. We focus on the noun-modifier relations and, for the time being, ignore the subject-verb and verb-object relations. Nouns and modifiers are represented by nodes in the graph. If a noun-modifier relation exists between a pair of nodes, the nodes are connected by directed edge from the modifier to the noun, ie. Modifier→Noun. Directing the edges from the modifier to the noun is more intuitive. Also, a preliminary test showed that directing edges in this way, as opposed to Noun → Modifier, gave better results. For example, the noun-modifier relations extracted from "manual mode", "auto mode" and "mode button" are represented as follows.

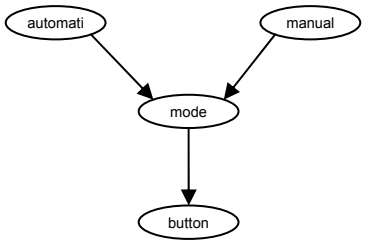

Figure 3. Representing noun-modifier relations as nodes and edges in a graph

Each edge is also associated with an edge weight that corresponds to the number of times a relation occurs in the corpus. The relations extracted from the collection are converted into a graphical representation as described above. The keywords are selected from the nodes in this graph.

Graph based algorithms

Graph-based ranking algorithms are used to decide the importance of a vertex in a graph based on information computed recursively from the entire graph. These algorithms have been successfully used in several web-based tasks. Google's PageRank [2] is an example of graph-based algorithms used in the web domain. They have also been shown to be effective for natural language processing applications [8] such as sentence and keyword extraction. We applied one such algorithm, HITS, to our noun-modifier dependency graphs.

The HITS algorithm [6] developed by Kleinberg gives each node in the graph a hub score and an authority score. A hub is a document that points to many important documents. An authority is a document that is pointed to by many important documents. For our problem, and as per our hypothesis, an authority corresponds to a domain concept and a hub corresponds to an attribute. Nodes are ranked by their authority score and the top ranked ones are chosen as important concepts in the domain. The HITS scores for each node are calculated as follows:

$$HITS_H(V_i) = \sum_{V_j \in Out(V_i)} HITS_A(V_j)$$

$$HITS_A(V_i) = \sum_{V_j \in In(V_i)} HITS_H(V_j)$$

The advantage of using HITS to calculate the hub and authority scores is that this algorithm was designed for directed graphs. However this version of the HITS algorithm does not take into account edge weights. Therefore a relation that occurs very rarely in the collection is treated on par with a relation that occurs often. Our solution to this problem is to consider only the edges above a threshold weight. This threshold can be tuned for each domain.

Graph construction

The list of hubs and authorities from the previous step is used to structure the keywords into concepts and attributes. For each authority term, the list of its modifiers ($In(V_{term})$) is pruned so that only those remain that are either present in the ranked list of hubs or occur above a threshold frequency. A small section of the resulting graph for the digital camera domain is shown below in Figure 2 using the visualization tool TouchGraph http://www.touchgraph.com.

In the section of the graph that appears in Figure 3, there are two nodes, "card" and "battery" with a large number of incoming edges. These are important concepts in this domain. Some of the attribute values for battery are *AA, AAA, lithium_ion, alkaline* etc. The aforementioned attribute values can actually be classified into two based on the class of the attribute value, namely into "size" and "chemical_type". However we do not tackle that problem in this work. A more detailed explanation of this issue is presented in the section on future work.

The attribute values for "card" include *8_MB, 128_MB, compact_flash* etc. The term "card" in this domain implicitly refers to a memory card. Since the dependencies are drawn from online customer reviews, the official term "memory card" is not used often and, therefore, is not extracted by our algorithm.

Evaluation

In order to test the robustness of our approach we tested it on real world data. The data that we used for our experiments was drawn from the World Wide Web. We created two domain-specific collections

1. Digital camera reviews from http://www.epinions.com

2. Wine-related articles from http://www.klwines.com

Each of these collections was processed and parsed as described earlier. The extracted dependencies were converted into a graphical structure. The link structure was analyzed and the hubs and authorities were identified.

Two types of evaluation were performed. The list of extracted concepts and attributes was evaluated for

- keyword extraction
- keyword structuring for ontology creation

The second type of evaluation required a hand-crafted Gold Standard ontology which was created as described in the following section.

Gold standard

Since creating a domain ontology from scratch is a painstaking and time consuming process, we started from an ontology that already exists.

An online wine ontology [16] in the OWL format was the starting point for the gold standard. This ontology was modified by the removal of some classes and by the addition of attributes to some classes. The predefined version of some class names, attribute names and attribute values were replaced by the surface forms that commonly occur in text.

Evaluation measures and results

The algorithm presented in this paper was evaluated with respect to two tasks: keyword extraction and ontology creation. The results are presented in separate tables for each task. The method that has the highest precision has its name displayed in bold font in the corresponding result table.

The keyword and concept extraction approach was applied to the digital camera corpus. For the first task, the list of terms was shown to a human expert who judged whether each term was a domain key word or not. Terms were ranked by keyword score, which was assigned by the algorithm. This score was the sum of the hub and authority scores returned by the HITS algorithm. Several other combinations of the hub and authority scores were tested but did not give better performance than $HITS_A$+$HITS_H$. The hubs and authorities extracted by our method were compared to the list of top terms extracted using Clarit Thesauri extraction [9]. The Clarit terms were ranked using the Rocchio method. Precision at rank *n* was defined as the number of domain keywords seen in the ranked list at rank *n*. As you can see from the results in Table 1, HITS performed better than Rocchio on this task.

$$\text{Precision at rank } n = \frac{\text{Number of keywords seen}}{n}$$

Top term	Precision@20
Rocchio	70%
$HITS_A$+$HITS_H$	**80%**

Table 1. Precision on keyword extraction for HITS and Rocchio in the digital camera domain

For the digital camera domain, a human expert judged whether the concepts extracted by our method matched concepts in the domain. The list generated by the HITS algorithm was compared to what we call a non-iterative baseline (NoGraph) as well as the Rocchio method. In the non-iterative method, terms are ranked by the number of immediate in-links (modifiers) to identify concepts and are ranked by the number of immediate out-links (targets) to identify attributes. Results for this experiment are shown in
Table 2. The HITS algorithm gives better performance than Rocchio and the non-iterative method on both these tasks, suggesting that recursive information from the graph beyond immediate neighbors is beneficial. The NoGraph method performs surprisingly well, indicating that noun-modifier relations are a good feature for concept and attribute extraction.

Method	Concepts	Attributes
NoGraph	45%	55%
Rocchio	50%	25%
HITS	**55%**	**60%**

Table 2. Precision@20 for concept and attribute extraction at top 20 terms in the digital camera domain

The algorithm described in this study was applied to the wine domain. For evaluation purposes, the gold standard wine ontology was used for truth judgments instead of a human expert. The results for keyword extraction in the

wine domain are in Table 3 and the results for concept and attribute extraction are in Table 4. These tables include results for the NoGraph method and Rocchio method when appropriate.

In the Window-n method in Table 4, a graph is created from text by drawing an edge from term t_1 to term t_2 if term t_1 occurs immediately preceding t_2 in the corpus. The HITS algorithm is then applied to this graph and concepts and authorities are selected as described earlier. This method was found to be inferior to the approach using term dependencies.

For the HITS-phrases method, lexical atoms were identified and grouped together at the text processing stage. It was found that this did not improve the performance of the algorithm, but it reduced errors due to mistakes in chunking lexical atoms.

Top term	Precision@20
Rocchio	75%
Window-n	65%
$HITS_A$+$HITS_H$-phrases	70%
$HITS_A$+$HITS_H$	**95%**

Table 3. Precision on keyword extraction for HITS and Rocchio in the wine domain

Method	Concepts	Attributes
NoGraph	65%	50%
Rocchio	40%	65%
Window-n	65%	60%
HITS-phrases	60%	70%
HITS	**70%**	**75%**

Table 4. Precision@20 for concept and attribute extraction at top 20 terms in the wine domain

Discussion and Future Work

From our experiments, it appears that there is a performance advantage to using noun-modifier relations and link analysis to identify keywords, concepts and attributes in a domain. Our hypotheses about concepts and attributes are supported by the results of our experiments. Words that occur in domain specific text as targets for many different modifiers are probably concepts and those that occur as modifiers for many different targets are probably attributes. The HITS scores help to rank these candidate concepts and attributes.

One aspect of this problem that we do not address is coverage. When designing a domain, ontology, it is important to cover all important areas of a domain. However one requires a comprehensive list of keywords from a domain in order to evaluate the coverage of a list of extracted keywords. As part of future work, we hope to create such a list for the digital camera domain. Once this list is created, we can evaluate our algorithm for coverage.

The next step would be to classify the type of relation between a concept and an attribute. This would enable the clustering of attribute values that share the same type of relation with a concept. For example, the concept battery has the possible attribute values *Lithium ion, NiMh* and *alkaline*. These attribute values are all of the class "Material" for the concept battery. Similarly, the concept "Memory card" has the following attribute values from the class "capacity": *16MB, 32MB, 64MB, 128MB and 256MB*. There is ongoing work in this area.

In this study, the grouping of lexical atoms did not improve performance. However, there appears to be some benefit to using lexical atoms as this reduced errors caused by multi-word lexical atoms. It is possible that a different threshold criteria or a different scoring metric for the lexical atoms may improve system performance over the current method.

For this study, we focused our attention on noun-modifier relations. The other types of dependencies that we extracted may also be used to help ontology creation. The subject-verb and verb-object dependencies may be used to describe the relations between concepts. This is another possible direction for future work.

References

1. Avancini, H., Lavelli, A., Magnini, B., Sebastiani, F., Zanoli, R. Expanding Domain-Specific Lexicons by Term Categorization. In Proceedings of the 2003 ACM Symposium on Applied Computing, pages 793-797.

2. Brin, S., Page, L. 1998. The anatomy of a large-scale hypertextual Web search engine. Computer Networks and IDSN Systems, 30(1-7).

3. Caraballo, S. A., Charniak, E. 1999. Determining the Specification of Nouns from Text, in Proceedings of the Joint SIGDAT conference on Empirical Methods in Natural Language Processing and Very Large Corpora.

4. Hearst, M., A. Automated discovery of WordNet relations.

5. Ide, N., Véronis, J. (1993). Refining taxonomies extracted from machine-readable dictionaries. In Hockey, S., Ide, N. Research in Humanities Computing II, Oxford University Press, 145-59.

6. Kleinberg, J.M. 1999. Authoritative sources in a hyperlinked environment. Journal of the ACM, 46(5):604-632.

7. Mihalcea, R. 2004. Graph-based Ranking Algorithms for Sentence Extraction, Applied to Text Summarization. In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, companion volume (ACL 2004).

8. Mihalcea, R., Tarau, P. 2004. TextRank: Bringing Order into Texts, in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004), Barcelona.

9. Milic-Frayling, N. Evans, D.A., Tong, X., Zhai, C.X: 1996. CLARIT Compound Queries and Constraint-Controlled Feedback in TREC-5 Ad-Hoc Experiments. In Proceedings of TREC 1996.

10. Minipar: http://www.cs.ualberta.ca/~lindek/minipar.htm

11. Building a large annotated corpus of English: the Penn Treebank Mitchell P. Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz, Computational Linguistics, vol 19, 1993

12. Automatic term recognition based on statistics of compound nouns and their components. Hiroshi Nakagawa and Tatsunori Mori. Terminology, 9:2, pages 201-219, 2004

13. Owl guide/tutorial: http://www.w3.org/TR/owl-guide/

14. Barbara Rosario and Marti Hearst, Classifying the Semantic Relations in Noun Compounds via a Domain-Specific Lexical Hierarchy, in the Proceedings of EMNLP '01, Pittsburgh, PA, June 2001.

15. Turney, P., Litman, M. Learning Analogies and Semantic Relations NRC/ERB-1103. July 10, 2003. 28 pages NRC Publication Number: NRC 46488.

16. Wine ontology: http://protege.stanford.edu/plugins/owl/ontologies.html

17. WordNet. An electronic lexical database. Edited by Christiane Fellbaum, with a preface by George Miller. Cambridge, MA: MIT Press; 1998. 422 p. $50.00

18. TouchGraph: http://www.touchgraph.com