RC22789 (W0305-007) May 2, 2003
Computer Science

IBM Research Report

Advances in Lexical Navigation

James W. Cooper
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Delhi - Haifa - India - T. J. Watson - Tokyo - Zurich

Visualization of relational text information for biomedical knowledge discovery

James W. Cooper
IBM Thomas J Watson Research Center
PO Box 704
Yorktown Heights, NY 10598
914-784-7285
jwcnmr@watson.ibm.com

ABSTRACT

Lexical Navigation provides users with a convenient technique for moving between related documents and terms within a collection without ever having to formulate an exact query to retrieve these related entities. It consists of a visual interface client and an index file. We discuss the algorithms we used to construct unnamed and named relations and the Java libraries we have developed. Many researchers have attempted to find relations in the Biomedical domain using strategies for recognizing protein and gene names, for example. By contrast, our strategy is to find major noun and verb phrases of all types and compute relations by recurring proximity. We then can apply biomedical term recognition as a filter against the relations we discover.

Our graphical display of the computed relations can be launched and used without reference to a database, and our XML data representation provides a portable way for other workers to access and visualize the data we have extracted.

Keywords

Text mining, Search, Document display, Databases, Lexical navigation, XML, Java.

INTRODUCTION

We have previously described the concept of Lexical Navigation [1] and the layout algorithms for the representation of a lexical network [2,3, 4]. In this paper, we discuss the browsing interface and the technical underpinnings that make a responsive navigation system that can approach the ideal of query-free document retrieval.

Other workers have detected relations between terms. For example, Roark and Charniak[5] have analyze noun phrase co-occurrence statistics by choosing seed words and finding words near them to choose need seed words. This is essentially similar to the Dual Iterative Pattern Relation Expansion (DIPRE) bootstrapping technique originally described by Brin[6]. Agichtein and Gravano [7] generated relations in a manner similar to DIPRE, but used a tagger to add more grammatical intelligence to the process.

In the Biomedical domain, Blaschke *et. al.* [8] identified protein-protein interactions using a small dictionary of common verbs, and Pustejovsky, Castano and Zhang [9] described methods for detecting the *inhibit* relation in a small number of abstracts.

There have also been any number of papers that illustrate visualizations of relations between concepts, although only a few of them seem to be interactive. Lamping and Rowe [10] described the Hyperbolic Browser and Eick and Willis [11] showed the navigation of organizational networks. Neither evaluates the effectiveness of these visualizations.

In the Biomedical domain, Stephens *et. al.*[14] detected a limited set of gene relations from Medline abstracts using small hand-built dictionaries of genes, and relation verbs. They illustrated some of these relations with graphical diagram, but did not describe how it was generated.

Enright and Ouzounis [15] described BioLayout, a graphical system for displaying similarities between proteins, Spencer and Bennett[16] described ProtInAct, an interactive system for displaying interactions between a number of proteins, using the yFiles graph drawing package[19], and Zhang *et.al.*[20] described an interactive 3D visualization system for protein interaction mapping. Jenssen *et. al.*[21] constructed a network of genes co-cited in the same abstract, but without any semantic relationship. None of these had a linguistic component, however and none of them had any evaluations of their efficacy.

Ideally we would like to construct a relations network that allows knowledge discovery such as that originally found manually by Swanson [22], where he found the relationship between "Raynaud's disease" and "fish oil." Some work along this line has also been carried out by Grell.[23], and by Ng and Wong [24] where they employed simple pattern matching and some visualization.

Current Work

In this discussion, we describe how we constructed a lexical network for 584 pharmaceutical patents. The system is in no way limited to such small collections, but this collection merely provides a convenient and interesting set of publicly available example documents. Then we discuss how we use standard ontologies to filter the relations we discover. Finally we illustrate how the relations can be exported and represented in an interactive graphical lexical navigation system.

Ideally, such a viewer should provide

- a means to illustrate both named and unnamed relations,
- allow the user to filter the results by selected hierarchies or ontologies,
- filter by relation strength, view the supporting documents, and maintain or change the selected focus[10].
- allow you to perform path analysis through a set of relations, and
- allowyou to vary the type of layout.

We have achieved some of these goals: others remain to be achieved.

The JTalent Library

Our system is constructed using our Talent (Text Analysis and Language Engineering Tools) text mining system that recognizes names [25] and multiword technical terms [26] and uses a relational database to store the terms it discovers. The most recent version (Talent 5.1) has been described in detail by Neff [18].

We have constructed the JTalent library and a set of JNI functions that enable us to call functions in Talent from Java. In addition, we have written the KSS library of functions for managing tables in databases such as IBM's DB2 from Java as well. Thus, all of the work we describe here was performed entirely in Java.

We start with this collection of patent documents and run the Talent processor on this collection. This gives us

- A database load file of all the salient terms per document, and their relative token positions in the document.
- A load file of the patent documents, along with their dates, titles, and authors.

We load the Documents table with a series of document key numbers, along with the title and filename of each document.

We load the TermDocs table with the terms in each document, the document key, paragraph number, sentence number and offset of the term. By putting these data into a database where we can fetch them rapidly, we can look up the principal multiword terms in a document or the documents which contain any specified term.

We can then use a few simple database queries to construct a Terms database table of all the unique terms in the document collection, and compute their frequencies, and the number of documents in which they appear once and more than once. Then we can compute the Information Quotient (IQ) [12] or salience of each term based on these frequencies.

COMPUTING RELATIONS

We describe here the Java library code which carries out the computation of relations. The computation is similar to and derived from that described by Byrd and Ravin [13].

We can compute relations between terms in the collection in two ways. First, for each abbreviation whose long form is detected by Talent[17], we compute a "same-as" relation, such as NO for "nitric oxide," and store it in a table as a *named relation*. We can also compute relations between terms based on their proximity. If two terms occur near each other on several occasions within the collection of documents they have a stronger relation than those which co-occur but once. We refer to these as *unnamed relations*, but we regard them as relations for which we have not yet been able to discover a name.

Since we store the document number, and token position for each term in the database, it is a simple matter to compute the occurrence of terms that co-occur within any specified distance. Further, we can tune these relations to select only those where one or both of the terms have a salience above a specific value.

We compute the weights of these relations using the mutual information formula

$$m = \log\left(\frac{totalterms \bullet paircount}{freq1 \bullet freq2}\right)$$

where *totalterms* is the total number of unique terms in the collection, *paircount* is the number of documents in which both terms occur, and *freq1* and *freq2* are the frequencies of the two terms in the collection. After computing all the mutual information values *m* for the term pairs, we scale them to lie between 0 and 100.

We can then generate a database load file for the terms and their weights of their unnamed relations. We construct the Relations database table to contain both of the related terms, the strength of the relation and the relation name or “none” for unnamed relations. Named relations are assigned a weight of “100” automatically.

Use of Ontologies

In addition to co-occurrence and salience measures, it is particularly useful to relate discovered terms to those in a known dictionary or ontology of terms in a particular domain. Our group has developed code which matches terms with those in the MeSH[28] ontology and assigns MeSH IDs and pathways to each recognized term.

This dictionary matching need not be limited to a single source, however, and it is not unreasonable to search several such dictionaries for term matches. Then, we can further filter the relations we discover by whether one or both of them belong to a particular ontology.

Exporting Term Relations

Once we have loaded the database with the computed relations we have in the past built a server that returns relations on request to a client on the same or a different system using, for example, a web service protocol. However, we have further found that for moderate sized collections, it is tractable to export all of the discovered relations to an XML file, which then represents a portable version of this knowledge.

Each relation is represented by a simple XML statement block such as the following:

```
  <relation>
  <rdef name="unnamed" strength="90" />
    <name>none</name>
  <term>
         polypeptide product
    <iq value="92" />
    <tdef source="shallowParser" name="NP" />
    <tdef source="Talent" name="UWORD" />
  </term>
  <term>
         eucaryotic host cell
  <iq value="92" />
  <tdef source="shallowParser" name="NP" />
  <tdef source="Talent" name="UWORD" />
 </term>
  <relationDocuments>
         <doc>34</doc>
         <doc>54</doc>
  </relationDocuments>
</relation>
```

Such a data structure can also contain references to any number of ontology sources, such as

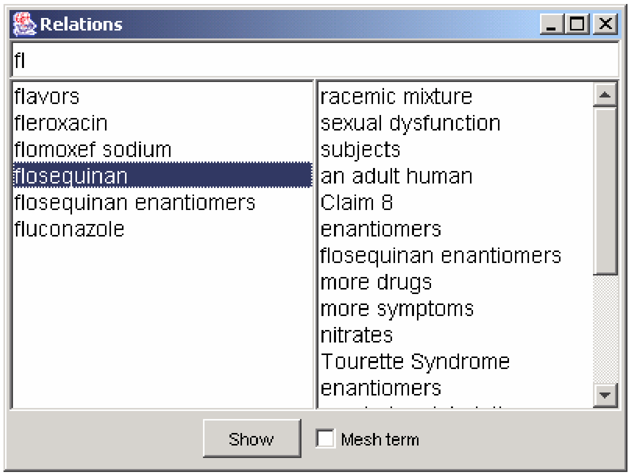
```
<tdef source="MeSH" name="D001076"
canon="Aptitude"/>
```

Thus, this XML file represents a complete, portable source of relation data that can be viewed using any sort of viewer that might be developed. Further, this portable form completely decouples the data representation from any graphical viewing system, and allows researchers to develop any number of different viewers and data objects for different purposes.

A GRAPHICAL RELATIONS VIEWER

We have developed a viewer that illustrates unnamed (and named) relations in a pair of list boxes and in a Lexical Navigation window.

Our RelationViewer program creates a Java Relations object for each relation it reads in from the XML file. Then, it inserts them into a Trie structure[27] based on the lower case representation of the first term. Now, since any given term may have several relations, we extend the common Trie structure to include a Vector of relations, all of which have the same first term.



Relations

fl

flavors
fleroxacin
flomoxef sodium
flosequinan
flosequinan enantiomers
fluconazole

racemic mixture
sexual dysfunction
subjects
an adult human
Claim 8
enantiomers
flosequinan enantiomers
more drugs
more symptoms
nitrates
Tourette Syndrome
enantiomers

Figure 1 – A display of relations to “flosequinan.”

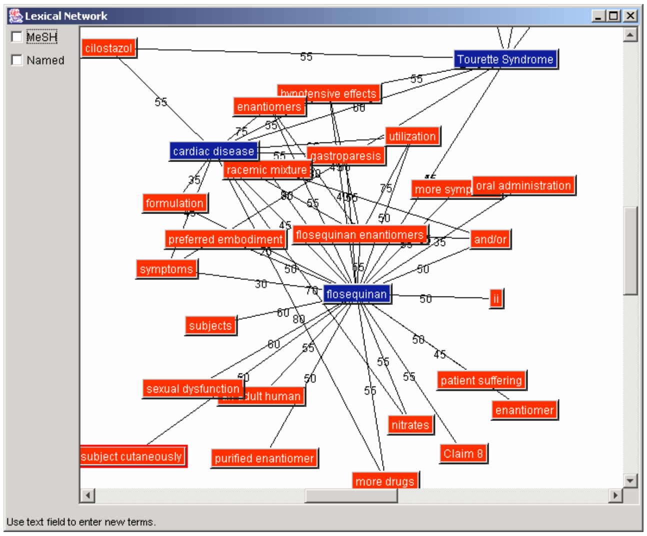
Figure 1 shows the first part of our relations viewer. We index an alphabetical list of all the terms in the relations and store it in a standard Trie. Then, you can bring up any portion of the alphabetical list of terms by typing part of that term into the entry field at the top. When you select one of the terms, it triggers a lookup into the Relations Trie and displays all the relations in the right hand list box. If you select the “MeSH term” checkbox, only those relations are shown which are to MeSH terms.

Lexical Navigation

Lexical Navigation is our term for the ability to explore term relations graphically, without needing to formulate specific queries. The system starts by displaying the single term you selected, and then expands each node when you double-click on it. Nodes which have been expanded turn to a darker blue color.

Figure 2 illustrates one such navigation, that illustrates a relation between “flosequinan,” “Tourette syndrome” and “cardiac disease” that could not have been observed in the simple two-column display of Figure 1. Since the XML data also contains a reference to the actual document(s) where the relation has been discovered, you could right click to see a list of these documents and view them if necessary.

Since we collect and store both noun phrases and verb groups, we plan to expand this display to show noun-verb-noun relationships such as X *inhibits* Y.



Use text field to enter new terms.

Figure 2 – A Lexical Navigation screen, showing relations found to the drug “flosequinan.”

EVALUATIONS

Evaluation of the efficacy of detecting relations and filtering them by ontology categories is still underway. In general, we have observed that we are not alone in finding it difficult to provide unambiguous evaluations of the concepts of lexical networks and of the associated visualization technique of Lexical Navigation. However, it is at least intuitively clear by inspection that relations which are more than one step apart are best viewed in a display similar to that in Figure 2. Evaluations for such systems are not trivial to undertake and as yet few have been done in any area relating to text mining and visualization.

CONCLUSION

We have developed a Java-based text mining system that stores terms and their relative positions in a database. You can query that database using various cutoffs including distance, salience and ontology membership to generate a set of relations which can be stored in the database. These relations can also be extracted and stored in a portable XML file. We have developed a viewer for these XML data that allows you to explore these relations both in list form and in graphical form. While these sorts of graphical displays are fascinating technically, it remains to be established that users find them useful.

ACKNOWLEGEMENTS

We thank Bhavani Iyer for writing the XML extractor from our database representation, Eric Brown for the use of his DictMatcher code for detecting MeSH terms, and Bob Mack for numerous helpful discussions. The original force-based layout algorithm was written by Daniel Tunkelang while a summer student in our lab.

REFERENCES

1. Cooper, J. W. and Byrd, R J, “Lexical Navigation: Visually Prompted Query Refinement,” ACM Digital Libraries Conference, Philadelphia, 1997.

2. Cooper, James W. and Byrd, Roy J., OBIWAN – “A Visual Interface for Prompted Query Refinement,” Proceedings of HICSS-31, Kona, Hawaii, 1998.

3. Cooper, J. W. “The Technology of Lexical Navigation,” JCDL-2001.

4. Tunkelang, D. D., Byrd, R. J., and Cooper, J. W., “Lexical Navigation: Using Incremental Graph Drawing for Query Refinement,” Graph Drawing 97.

5. Roark, B. and Charniak, C., “Noun phrase co-occurrence statistics for semi-automatic lexicon construction.” Proceedings of the 36th Annual Meeting of Association for Computational Linguistics, 1998.

6. Brin, S “Extracting Patterns and Relations for the World Wide Web,” Proceedings of the 6th Annual WebDB Workshop, EBDT98, 1998.

7. E. Agichtein and L Griavano, “Snowball: extracting Relations from Large Plain-text collections.”

Proceedings of the 19th IEEE Conference on Data Engineering, 2003.

8. Blaschke, C., Andrade, M.A., Ouzounis, C. and Valencia, A., “Automatic extraction of biological information from scientific text: protein-protein interactions,” *BioInformatics* 4(7), 1998.

9. Pustejovsky, J, Castano, J. and Zhang, J. “Robust Relational Parsing over Biomedical Literature: Extracting Inhibit Relations,” Proceedings of the Pacific Symposium on Biocomputing (PSB) 2002.

10. Lamping, J., and Rao, R., “The Hyperbolic Browser: A Focus+Context Technique for Visualizing Large Hierarchies,” in *Readings in Information Visualization,* S K. Card, J.K. Mackinlay and B. Schneiderman, Morgan-Kauffman, 1999.

11. Eick, S. G. and Willis, G.J., “Navigating Large Networks with Hierarchies,” in *Readings in Information Visualization*.

12. Prager, John M., Linguini: Recognition of Language in Digital Documents, *in Proceedings of the 32nd Hawaii International Conference on System Sciences*, Wailea, HI, January, 1999.

13. Byrd, R.J. and Ravin, Y. Identifying and Extracting Relations in Text. *Proceedings of NLDB 9*9, Klagenfurt, Austria.

14. Stephens, M., Palkal, M., Mukhopadhyay, R and Mostafa, J., “Detecting Gene Relations from Medline Abstracts,” Proceedings of the Pacific Symp osium on Biocomputing, 2001, Honolulu, HI.

15. Enright, A.J. and Ouzounis, C. A., “BioLayout –An automatic graph layout algorithm for similarity visualization,” *Bioinformatics* 17(9), 853-854 (2001).

16. Spencer, H. and Bennett, S.P., “Visualizing Protein-Protein Interactions on a Genomic Scale,” IEEE Conference on Information Visualization, Boston, 2002.

17. Park, Y. and Byrd, R. J., “Hybrid text mining for finding abbreviations and their definitions,” Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2001.

18. Neff, Mary, Byrd, Roy J. and Boguraev, B. “The Talent System: TEXTRACT Architecture and Data Model,” NAACL Workshop on Software Engineering and Architecture of Language technology Systems, Edmonton, Canada, 2003.

19. Wiese, R., Eiglesperger, M., and Schaber, P. “yFiles Graph Drawing Package, 2002. www.yWorks.com

20. Zhang,Y., Tian, H., Kraemer, E. and Arnold, J. “A visualization System for Protein Interaction Mapping Using Java 3D Technology,” submitted to *BioInformatics*. 2003. Nissan.cs.uga.edu/ ~yozhang/protein3D/.

21. Jenssen, T-K., Komorowski, A-L, Hovig, E., A literature network of human genes for high throughput analysis of gene expression. *Nature Genetics*. 28, 21-28, May 2001.

22. Swanson, D.R., “Fish oil, Raynaud’s syndrome and undiscovered public knowledge,” *Perspectives in Biology and Medicine* 30(10 7-18, (1986)

23. Grell, Stephan, unpublished Master’s thesis, University of Heidelberg.

24. Ng, S-K. and Wong, M., “Toward routine automatic pathway discovery from on-line scientific text abstracts.” *Genome Informatics. 10: 104-112*. 1999.

25. Ravin, Y. and Wacholder, N. 1996, “Extracting Names from Natural-Language Text,” IBM Research Report 20338.

26. Justeson, J. S. and S. Katz "Technical terminology: some linguistic properties and an algorithm for identification in text.” *Natural Language Engineering*, 1, 9-27, 1995.

27. Goodrich, Michael and McGeoch, Catherine, eds., *Lecture Notes in Computer Science* 1619, Springer-Verlag, 1999.

28. Medical Subject Headings, National Library of Medicine, www.nlm.nih.gov/mesh/meshhome.html.