A Method for Vocabulary Development and Visualization based on Medical Language Processing and XML

Hongfang Liu[3], Carol Friedman[1,2]

[1]Computer Science Department, Queens College of CUNY
[2]Department of Medical Informatics, Columbia University
[3]Computer Science Division, Graduate School and University Center of CUNY

A comprehensive controlled clinical vocabulary is critical to the effectiveness of many automated clinical systems. Vocabulary development and maintenance is an important aspect of a vocabulary, and should be linked to terms physicians actually use. This paper presents a method to help vocabulary builders capture, visualize, and analyze both compositional and quantitative information related to terms physicians use. The method includes several components: an MLP system, a corpus of relevant reports and a visualization tool based on XML and JAVA.

Introduction

The use of the Electronic Medical Record (EMR), Medical Language Processing Systems (MLP) and Decision Support Systems (DSS) in the medical domain offers the potential for saving physicians time, for providing better treatment to patients, and for enhancing research. The quality of these types of systems is determined by the effective selection of a suitable medical vocabulary[1].

Cimino et.al.[2] stated that one of the requirements of a controlled vocabulary is completeness. A vocabulary that is developed without regard to physician usage may not be complete. For example, BIRADS, which was developed by a committee of mammography experts, is supposed to represent all relevant findings in mammography reports. Starren et.al.[3] analyzed mammography findings from a sample of reports, and found that critical concepts were missing from BIRADS.

Since a substantial amount of clinical input generally comes from physicians, it is important that a medical vocabulary incorporate the concepts that physicians use. Also, because of the dynamic nature of the medical domain, it is essential to continually update existing vocabularies in order to include evolving concepts.

In this paper, we present a method to help clinical system builders capture terms physicians use, and to analyze compositional and quantitative information associated with the terms. We also discuss a visualization tool that we developed along with a method that assists users in developing and refining a controlled vocabulary associated with a clinical domain. The method is based on an existing MLP system called MedLEE[4], XML[5] (the latest markup language compatible with WEB technology), JAVA[6] (an object oriented programming language designed to run seamlessly on many different kinds of platforms), and a corpus of clinical reports.

The method is introduced in detail in the Methods section. We provide some related work and background knowledge in the following section, and in the last two sections, we provide some discussion and conclude the paper.

Related Work and Background

There are several articles that discuss vocabulary development methods based on a frequency analysis of large text corpora and/or natural language processing (NLP) tools. One of them used word frequency analysis as a tool for finding the empirical basis for structured data entry design[7]. Hersh et.al.[8] employed advanced NLP tools to identify clinical findings in large corpora of narrative medical reports. He compared the findings he identified with the UMLS Metathesaurus, and determined that the breadth of modifier coverage as expressed by clinicians was not present in the Metathesaurus. Elkin et.al.[9] studied the compositional nature of clinical vocabularies and presented methods to help users compose clinical terms. Chute et.al.[10] discussed desiderata for a clinical terminology server and mentioned, "A server should propose coordinated standard terms that in combination capture the full notion intended".

In the past several years, XML has emerged as the WEB's language for data interchange. XML changes the way data move across networks since it encapsulates data inside customized tags that carry semantic information about the data. Tools for processing and browsing XML are freely available and new XML-based applications are relatively easy to build. In the health care domain, the application of XML as an interchange format for communication

1067—5027/00/$5.00 © 2000 AMIA, Inc.

standards offers a much greater flexibility and adaptability to user needs than other currently used interchange formats[11].

We have developed a natural language processing system called MedLEE[4], which has been used at New York Presbyterian Health Care (NYPH) (formerly Columbia Presbyterian Medical Center) since February 1995. MedLEE was designed as a general processor within the medical domain. It was initially developed for chest radiographs, and has since been expanded to the domains of mammography, radiology reports, pathology reports, echocardiography, electrocardiography and discharge summaries. A number of evaluations of the system were performed within the domains of chest radiographs, mammography and discharge summary reports[12;13;14] that demonstrated that it was effective in identifying specific clinical conditions, and that it was effectively used[15] for improving the quality of patient care. In the studies cited above, MedLEE was used for decision support purposes.

An important component of MedLEE is the lexicon, which semantically categorizes medically relevant words and phrases, and specifies their target forms. There are two types of multi-word phrases specified in the lexicon: rigid phrases and compositional phrases. In the former, the words always occur together in the text, whereas in the later individual words can be separated from each other and the order permuted (i.e. *enlarged spleen* may occur as the *spleen* was *enlarged*). For decision support applications, multi-word phrases are treated by MedLEE as atomic units, but for other applications these phrases may be ignored and the individual words treated independently. This is discussed in more detail in the Methods section.

One of the output formats of MedLEE is XML. The XML representational model for the structured output of MedLEE is described in more detail by Friedman et.al.[16]. Figure 1 illustrates the XML output for the term *aspirate right breast*. The name of the tag corresponds to the type of information being represented. In addition, the tag may also contain embedded tags that are modifiers. For example, **procedure** is a type of information. It has an attribute **v** whose value is *aspirate* and an embedded tag that is a modifier **bodyloc** (body location: with value *breast*). **Bodyloc** also has a modifier **region** (with value *right*). The XML representation forms a tree that can be viewed graphically.

Methods

Our method consists of several steps designed according to functionality. Figure 2 shows an overview of the method.

```
<procedure v = "aspirate">
      <bodyloc v = "breast">
          <region  v = "right">
          </region>
      </bodyloc>
</procedure>
```

Figure 1. An example of the structured component of the XML output form generated by MedLEE for *aspirate right breast*

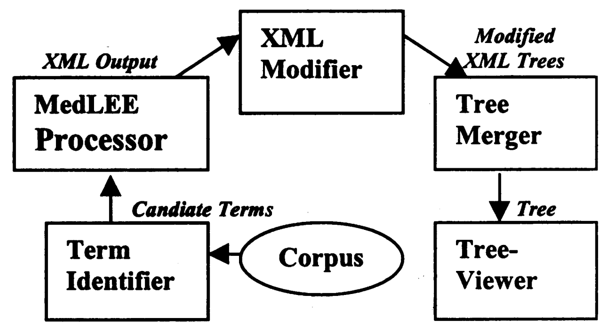

Figure 2. There are five processing steps for the vocabulary development tool. The first is the Term Identifier. It identifies candidate vocabulary terms and records the occurrences of candidates within a given corpus. The MedLEE Processor generates the compositional structure of each candidate in XML format. The XML Modifier modifies the XML format. The Tree Merger merges XML trees. The TreeViewer provides a GUI that allows the user to view the merged XML tree.

The corpus used by the method is generally a collection of medical reports from a particular domain. For our initial project we collected four years of pathology reports (a total of 366,572 reports) that were performed at NYPH. The first step of our method is the Term Identifier, which identifies candidate vocabulary terms and records the occurrences of candidates in the given corpus. In our initial project the candidate vocabulary terms were headers of the pathology reports. These consisted of a textual description of pathology procedures as expressed by pathologists. There were a total of 94,386 different headers. We disregarded those candidates that appeared less than 20 times, and obtained 833 unique candidate terms.

MedLEE was modified so that the parser could operate in a de-compositional mode of parsing. In this mode, multi-word compositional phrases (but not rigid phrases) are ignored and the individual words in the phrase are analyzed independently. In this way, MedLEE can structure the components of the candidate terms so that the outputs reflect their conceptual structures.

Each unique candidate term was parsed using the modified version of MedLEE in order to generate XML structured output forms. The XML output of MedLEE was then modified to separate the original tag (type of the information) and its **v** attribute (value of the information). This was done so that multiple values for the same type of information could be viewed as children for that type using a graphical TreeViewer, and so that a new tag **occu** could be added that specifies the number of times a particular structure occurs. Figure 3(a) shows the modified XML of **procedure** *aspirate* with **bodyloc** *breast* and **region** *right* that occurred 1,093 times. The tag **item** separates the original tag **procedure** from its **v** attribute. In addition, an attribute **occu** was added to record the number of occurrences of that tag. **Procedure** *aspirate* with **bodyloc** *breast* and **region** *right* contributes 1,093 times to the attribute **occu** of each of the corresponding modified XML tags.

The next step consists of merging the XML trees so that the similar types of information can be viewed together. Merging also provides occurrence information for all tags. If several candidates have the same XML tree, the **occu** value for any of the tree nodes will be the summation of occurrences of each of the candidates. If different XML trees have common ancestors, the merge operation will merge them into one tree, and the **occu** values for common ancestors will be the summation of occurrences of each candidate.

Figure 3(b) shows the result of merging two modified XML trees. One was obtained from a structure consisting of **procedure** *aspirate* with **bodyloc** *breast* and **region** *right* (occurring 1,093 times), and the other from a similar structure where **region** corresponded to the value *left* (occurring 1,189 times). After merging, the **occu** values for those common ancestors are 2,282: 1,189 times from **region** *left*, and 1,093 times from **region** *right*. When performing the merging operation, all the different types of information and their different substructures are combined and summations are performed appropriately.

After the individual trees are merged into one XML tree, the TreeViewer can be used to view the tree graphically. The TreeViewer is a JAVA application that provides a GUI that allows the user to view the merged XML tree, and also to choose several additional helpful functions. Figure 4 illustrates a visual presentation of the TreeViewer. It shows a partial tree of pathology concepts from the corpus that was used. Note that the **occu** value of each node records the total occurrences of the corresponding candidate terms in the corpus. For example, the **occu value** of tag **item** with **v** value *aspirate* is 3,670, which is the summation of the **occu** values of its children: the tag **bodyloc** with the **occu** value 2,866, the tag '**.**' with the **occu** value 676 (here '**.**' represents a concept structure tree without any children), the tag **region** with the **occu** value 70, the tag **certainty** with **occu** value 34 and the tag **descriptor** with **occu** value 24.

```
<procedure occu = "1093">
    <item v = "aspirate" occu = "1093">
        <bodyloc occu = "1093">
            <item v = "breast" occu = "1093">
                <region occu = "1093">
                    <item v = "right" occu = "1093"/>
                </region>
            </item>
        </bodyloc>
    </item>
</procedure>
```

(a)

```
<procedure occu = "2282">
    <item v = "aspirate" occu = "2282">
        <bodyloc occu = "2282">
            <item v = "breast" occu = "2282">
                <region occu= "2282">
                    <item v = "left" occu = "1189"/>
                    <item v = "right" occu = "1093"/>
                </region>
            </item>
        </bodyloc>
    </item>
</procedure>
```

(b)

Figure 3. (a) Modified XML for **procedure** *aspirate* with **bodyloc** *breast* and **region** *right*. (b)Subsequent merging of two modified XML trees, one corresponds to the example in (a) and the other has a similar structure but with **region** *left*.

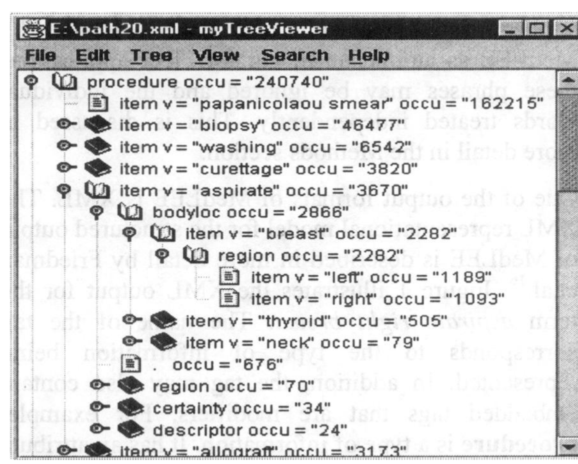

Figure 4. A partial structural tree of pathology terms from a corpus of pathology report headers.

The menu-bar of TreeViewer contains: <u>File</u>, <u>Edit</u>, <u>Tree</u>, <u>View</u>, <u>Search</u> and <u>Help</u> choices. The <u>File</u> menu performs standard file operations, the <u>Edit</u> menu provides some editorial functions, and the Tree menu provides different tree layout functions. The View menu provides several useful functions that allows the user to obtain different views of the tree and to retrieve source terms, and the Search menu performs standard search operations. The Help menu provides help information for the TreeViewer. In this paper we discuss some features of the View menu only.

There are two main features in the View menu: Sort and View Source. Sort sorts the tree *by occurrence* or *by alphabetic* order and presents the result in *descending* or *ascending* order as specified by the user. Figure 4 is a presentation of the choice *sort by occurrence* using descending order. The function View Source retrieves original source terms, whose modified XML outputs contain the structure represented by the paths from the root to the selecting node and its children. To demonstrate this feature, we selected the node *item v = "right" occu = "1093"* in Figure 4 and choose View Source under the View menu (the selected node is shown to the user highlighted in purple but this can not be seen in the figure); the result is shown in Figure 5. The source terms are shown in alphabetic order. The numbers in brackets are the occurrences of the associated source terms. Note the summation of occurrences for source terms is equal to the value of attribute **occu.**

Discussion

The method we have described was designed to help vocabulary development. MedLEE breaks down components of phrases and shows the relations among the components. Therefore, the method shows which components occur with other components, and the frequencies of the co-occurrences. The method is based on structural similarity and therefore textual variants that have the same structure are considered to be equivalent.

In addition to pathology report headers, radiology report headers have been successfully processed and viewed. Processing textual reports such as discharge summaries can also be performed, but will require modification of the candidate term identifier in order to isolate terms within sentences. This method can also use a controlled vocabulary such as SNOMED or the UMLS as the input corpus. In those cases, the compositional structure of the components of the vocabulary terms will be presented to the user, and the frequency information will reflect the occurrences of structures in the controlled vocabulary.

When the structural tree is imported to the TreeViewer, the user manipulates the tree and captures different kinds of information. The most frequent type of information in the tree formed by structuring pathology headers is **procedure**.

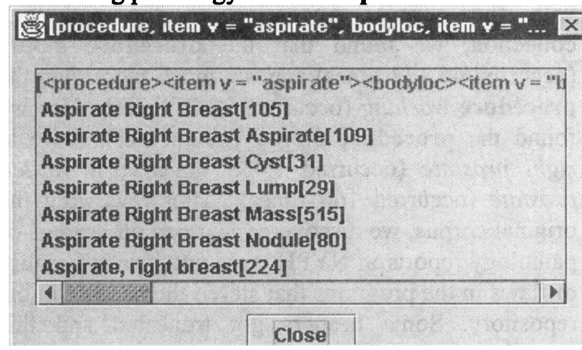

Figure 5. An example of view source for concept **procedure** *aspirate* with **body location** *breast* and **region** *right* from a corpus of pathology report headers.

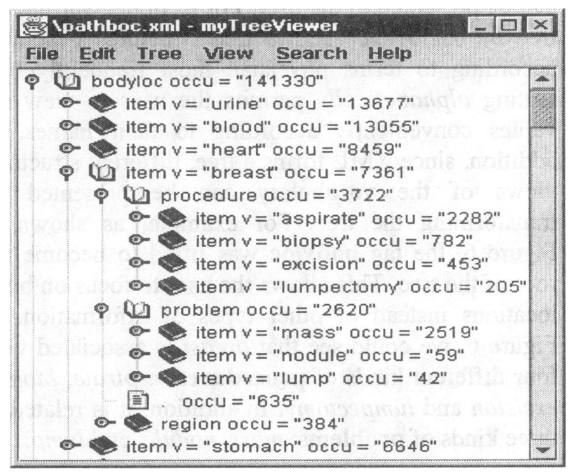

Figure 6. A partial body location tree from a corpus of pathology report headers.

The full tree associated with Figure 4 has 40 children under **procedure**, i.e. 40 kinds of **procedures**. The most frequent one is *papanicolaou smear* (occurring 162,215 times). The second one is *biopsy* that occurs 46,477 times. *Aspirate* is the fifth most frequent item under **procedure**. Under *aspirate*, there are 4 children which represent four different types of modifiers: **bodyloc**, **region**, **certainty** and **descriptor**. Under the modifier **bodyloc** (occurring 2,866 times), there are three values: *breast, thyroid* and *neck*. From the tree, we can see that 2,282 out of the 3,670 *aspirate* **procedures** occur at *breast*, 505 at *thyroid*, 79 at *neck*, and 804 *aspirate* **procedures** do not specify a body location. When the user is building a controlled vocabulary in the selected domain, it is helpful to traverse the tree to determine which atomic terms and compositional terms (i.e. terms with modifiers) should be considered for inclusion in the vocabulary.

The **occu** values of nodes can also be used to find important information and shortcomings associated with the corpus. According to the pathology collection, we found that the **procedure** *biopsy* (occurred 46,477 times) appears much more than the **procedure** *washing* (occurred 6,542 times). Also we found the **procedure** *biopsy* occurs much more at *right prostate* (occurred 1,438 times) than at *left prostate* (occurred 160 times). After reviewing the original corpus, we discovered that certain headers of pathology reports in NYPH were not correct because of flaws in the programs that stored the headers in the repository. Some headers got truncated and this caused parsing errors because the truncated words, such as *LEF* for *left*, were unknown to MedLEE.

XML is an extremely flexible mechanism. By manipulating the XML tree, many different views can be presented to the user. For example, sorting the tree *by occurrence* shows tree structures generated according to terms physician most frequently use; sorting *alphabetically* permits the user to view the values conveniently according to their names. In addition, since XML forms a tree, different structural views of the vocabulary can be presented by transforming the tree. For example, as shown in Figure 6, the tag **bodyloc** was lifted to become the root of the tree. This allows the user to focus on body locations instead of other types of information. In Figure 6, we could see that *breast* is associated with four different kinds of **procedures**: *aspirate*, *biopsy*, *excision* and *lumpectomy*. In addition, it is related to three kinds of **problems**: *mass*, *nodule*, and *lump*.

Our method has several limitations. The words in the candidate terms have to be known to the MedLEE system. If a word is unknown, structural information associated with it will be lost. Additionally, instances of structural information may be incorrect or lost if MedLEE parses a term incorrectly or fails to parse a term.

Future work will involve adding functions in the GUI to help users link terms from the corpus with related controlled vocabulary terms. The GUI will allow users to associate codes for the terms when possible, and to select terms for inclusion in the controlled vocabulary that are missing.

Conclusion

We have presented a method that uses MLP, XML, JAVA and a corpus of medical reports to facilitate vocabulary development. We believe the method can provide substantial help for creating and enhancing vocabularies, and for mapping to different vocabularies.

Acknowledgment This study was supported in part by grants LM06274 from the National Library of Medicine.

References

1.Cimino JJ. Desiderata for Controlled Medical Vocabularies in the Twenty-First Century. Meth Inf in Med. 1998; 37:394-403.

2. Cimino JJ, Clayton PD, Hripcsak G, Johnson SB. Knowledge Based approaches to the maintenance of a large controlled medical terminology J Am Med Inf Assoc. 1994; 1:35-40.

3. Starren, J. and Johnson, SB. Expressiveness of the Breast Imaging Reporting and Database System (BI-RADS). Proc AMIA Symp 1997:655-659.

4. Friedman C, Hripcsak G, DuMouchel W, Johnson SB, Clayton PD. Natural language processing in an operational clinical information system. Nat. Lang. Eng. 1995; 1:83-108

5. XML Web http://www.w3.org/XML.

6. Java Web. http://java.sun.com.

7. Kreis C and Gorman P. Word frequency analysis of dictated clinical data: a user-centered approach to the design of a structured data entry interface. Proc AMIA Symp 1997:724-728.

8. Hersh, WR., Campbell, EM., Evans, DA., and Brownlow, ND. Empirical, Automated Vocabulary Discovery Using Large Text Corpora and Advanced Natural Language Processing Tools. Proc. AMIA Symp 1996:159-163.

9. Elkin PL, Tuttle MS, Keck K, Campbell K, Atkin G, and Chute, CG The role of compositionality in standardized problem list generation. Proc. of MEDINFO 98. 660-664.

10. Chute, CG, Elkin, PL, Sherertz, DD, and Tuttle, MS. Desiderata for a Clinical Terminology Server. Proc. AMIA Symp 1999:42-46.

11. Dudeck J. Aspects of implementing and harmonizing healthcare communication standards. Intl J of Med Info 1998; 48:163-71.

12. Jain, NL., Knirsch, CA., Friedman, C., and Hripcsak, G. Identification of suspected tuberculosis patients based on natural language processing of chest radiograph reports. Proc AMIA Symp 1996:542-546.

13. Johnson, SB. and Friedman, C. Integrating data from natural language processing into a clinical information system. Proc AMIA Symp 1996:577-581.

14. Friedman,C., Knirsch,CA., Shagina,L., and Hripcsak,G. Automating a severity score guideline for community-acquired pneumonia employing medical language processing of discharge summaries. Proc. AMIA Symp 1999:256-260.

15. Knirsch CA, Jain NL, Pablos-Mendez A, Friedman C, Hripcsak G. Respiratory isolation of tuberculosis patients using clinical guidelines and an automated decision support system. Infection Control and Hospital Epidemiology 1998; 19:94-100.

16. Friedman C, Hripcsak G, Shagina L, Liu H. Representing information in patient reports using natural language processing and the extensible markup language. J Am Med Inf Assoc. 1999; 6:76-87.