High Precision Information Extraction

Rich Caruana
Center for Automated Learning and Discovery
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
caruana@cs.cmu.edu

Paul G. Hodor
Center for Biomedical Informatics
University of Pittsburgh
200 Lothrop Street
Pittsburgh, PA 15213
pgh@cbmi.upmc.edu

John Rosenberg
Department of Biological Sciences
University of Pittsburgh
314 Clapp Hall
Pittsburgh, PA 15213
jmr@jmr3.xtal.pitt.edu

ABSTRACT

Most fully automatic information extraction systems achieve less than 100% extraction precision and recall. On real applications these parameters typically vary between 50% to 95%, depending on the extraction method and source data.

We present an information extraction system designed for applications where little or no error can be tolerated. The system is not fully automatic. Instead, the extraction is guided by the intervention of a human expert. This "expert in the loop" approach greatly amplifies the amount of extraction an individual can accomplish, while insuring that the extraction process is nearly 100% accurate.

We used a tool we created called HPIEW (for High Precision Information Extraction Workbench) to extract several different fields from the text remark fields of the Protein Data Bank (PDB). The workbench allowed us to extract each field from more than 5,000 PDB files in an afternoon, with extraction precision and recall estimated to be greater than 99.9%. We believe this approach may be useful for other extraction problems where extreme accuracy is required.

1. INTRODUCTION

Accurate information extraction requires the maximization of both precision and recall. Manual information extraction is very accurate, but does not scale well to large sets of documents. Automatic information extraction scales well, but typically is far less accurate. Unfortunately, some problems are large yet require very high accuracy. For example, a bioinformatics project we are pursuing, which uses machine learning to address the protein folding problem, required us to extract several important numeric values from thousands of files from the Protein Data Bank (PDB) [1]. Yet our project required an error rate of less than 0.1% in the extracted values. The effort necessary to do the extraction manually is enormous. Yet the automatic extraction methods we examined were unable to achieve acceptable accuracies. This lead us to develop a semi-automatic extraction tool, called the High Precision Information Extraction Workbench (HPIEW), that amplified the ability of a domain expert to extract the values. Using the HPIEW, a user was able to extract values from 5,000 PDB files in a single afternoon.

Information extraction with the HPIEW is an interactive process where a human expert uses the workbench to extract the information from the source text. This approach is similar in spirit to the interactive extraction system proposed in [3]. A related approach that places more emphasis on *learning* to extract information from a small sample of labeled examples is discussed in [4, 5]. The approach presented in this paper does not depend on machine learning, and instead emphasizes using a human expert to make all critical decisions. While this makes the process less automatic, it also helps us obtain the very high precision needed for our problems.

This paper briefly introduces the information extraction problem that motivated our work: extracting values from the PDB. We then present the method we developed for high precision information extraction from this database. This method is guided by a human expert, and thus is semi-automatic. We present results from applying this method to the PDB, which show how a small number of user-created patterns are able to extract information from thousands of files. The extraction accuracy, however, is very high, making it difficult for us to precisely estimate it.

2. PROTEIN DATA BANK

The PDB [1] is the single international repository for data describing 3-D structure of biological macromolecules. It was established at the Brookhaven National Laboratories in 1971 and initially held seven structures. During the 1980s it began a dramatic growth which continues at present at an exponential rate. Currently the PDB is managed by the Research Collaboratory for Structural Bioinformatics and contains over 12,000 entries.

Each PDB entry contains a list of atomic coordinates for one molecular entity and additional information such as experimental procedures, literature references, author names, comments and annotations, etc. Much of the additional information was originally represented as free text in RE-

MARK records. Subsequent formats allowed for a much more structured representation, although format specifications were revised multiple times.

Non-uniform formatting of the data has been recognized for a long time as a major obstacle in introducing advanced query capabilities into the PDB. New PDB entries are required to conform to the STAR/mmCIF ontology [9], which allows for automatic management of the data. It also allows for easy conversion to the traditional PDB format, which is still widely used in most applications. However, automatic conversion of legacy data to mmCIF format is not believed to be possible [9]. This has led to a major, undergoing effort of the maintainers of the PDB to manually process legacy PDB entries to convert them to the mmCIF format [1].

2.1 R and R_{free}

Structure determination by X-ray crystallography involves recording of X-ray diffraction patterns generated from a crystal, followed by development of a structure model based on those data. The crystallographic R factor measures how well the theoretical model fits the available data. It has been used as one of the major indicators of structure quality [7, 6]. More recently the R_{free} factor was introduced, which is more reliable and less susceptible to manipulation [2].

Extraction of R and R_{free} values across the PDB is a prerequisite for comparing structure quality among all PDB entries. Such a task is complicated due to two reasons. First, as indicated above, values are reported in a variety of formats (Table 1). Older files, such as PDB 1ldm and 3c2c in Table 1, contain R values embedded in free text. PDB 3cyt doesn't even contain the actual value, but a literature reference instead. Although newer formats are more standardized, variations exist in how values are reported even within the same format [1].

The second reason is that there are variations in the experimental methods for structure refinement. Sometimes several different types of R and R_{free} values are reported for the same experiment, representing different aspects of the experiment (PDB 4gsp and 1jdo in Table 1). For these two examples, for the purpose of quality assessment, a human expert would select the R value of 0.159 from line 4 of 4gsp and 0.1531 from line 3 of 1jdo. It would be difficult to design an automatic parser that can identify those two lines correctly among the larger number of lines containing similar text patterns. Other times a single R (and perhaps R_{free}) value is reported (PDB 1lhc in Table 1). Even in such cases the method by which it is calculated is not unique and therefore the correct line containing the value needs to be identified.

Extraction of R and R_{free} from the PDB thus poses a dilemma: It is relatively easy for a human expert to decide which values to choose for quality assessment, however, it is impractical to process manually the entire databank. On the other hand it is difficult to develop an automated procedure that can reliably extract the desired values. To satisfy the conflicting needs for speed and accuracy, we devised a mixed strategy that combines human domain expertise and ability to understand free text with a semi-automated computer workbench that greatly amplifies the speed at which an expert can do extraction.

3. HIGH PRECISION INFORMATION EXTRACTION WORKBENCH

Our basic approach to high precision information extraction is to create a workbench that amplifies the speed and accuracy of a human expert. The system, by keeping the human in the loop, can yield accuracies comparable to pure manual extraction, yet is several orders of magnitude faster. Moreover, the design allows for the incorporation of some automatic extraction techniques.

The method depends on two user capabilities. First, the system depends on a user's ability to generate regular expression-like patterns that match a subset of the documents the user has browsed. The generated patterns are not required to be perfect. Instead, the system allows the user to iteratively refine and experiment with each pattern until they are satisfied with it.

To support this experimentation, the system depends on the ability of the user to quickly scan tables/lists of items if they are presented in an appropriate format. This allows the user to quickly generate, test, and refine patterns before committing to them. Moreover, the system does not require that one super-pattern be created to match/extract all documents. Instead, the user need only find patterns that match enough of the documents to be worthwhile. Extraction from the entire set of documents is accomplished by applying the sequence of patterns built by the user in the order they were created. This breaks the extraction process down into modular portions that are easier to generate and verify.

3.1 Detailed Approach

Figure 1 shows a flowchart of the extraction process. Consider extracting the numeric values for R_{free} from a large set of PDB files. Our approach is as follows. Extraction begins by placing all PDB files (about 5,000) into a single directory. Each file has a unique name that identifies the PDB entry. The HPIEW selects a small number of random PDB files and displays them to the expert. The expert scans the displayed PDB files and locates the desired R_{free} value in the remarks in each file. Once R_{free} has been identified, the user builds a regular expression that matches one or more of the R_{free} values. The goal is to build a maximally specific regular expression that will match the occurrences of R_{free} in other PDB files that are formatted similarly to the formatting in some of the displayed files. High specificity of the regular expression is important because we do not want the regex to falsely match any values that are not R_{free}; *if the regex matches, it must match a correct R_{free} value.*

The user has at their disposal an enhanced regular expression facility which provides a convenient set of matching and rewrite operations. The user is also able to specify range checks and type checks on matched values. For example, it can be specified that the matched value must be a number and must have a value between 0.1 and 1.0. Matches that do not satisfy these constraints are flagged to help the user refine the regex.

```
from PDB 1ldm:
    REMARK  3 REFINEMENT. BY THE RESTRAINED LEAST SQUARES PROCEDURE OF J.  1LDM 150
    REMARK  3  KONNERT AND W. HENDRICKSON (PROGRAM *PROLSQ*).  THE R       1LDM 151
    REMARK  3  VALUE IS 0.173 FOR REFLECTIONS IN THE RESOLUTION            1LDM 152
    REMARK  3  RANGE 6.0 TO 2.1 ANGSTROMS.  ATOMS WITH THERMAL FACTORS     1LDM 153
    REMARK  3  WHICH CALCULATE LESS THAN 2.00 ARE ASSIGNED THIS VALUE.     1LDM 154
    REMARK  3  THIS IS THE LOWEST VALUE ALLOWED BY THE REFINEMENT          1LDM 155
    REMARK  3  PROGRAM.                                                    1LDM 156

from PDB 3c2c:
    REMARK  3 REFINEMENT. RESTRAINED PARAMETER LEAST-SQUARES METHOD OF     3C2C  25
    REMARK  3  KONNERT AND HENDRICKSON.  THE FINAL R-FACTOR IS 0.175.      3C2C  26

from PDB 3cyt:
    REMARK  3 REFINEMENT. SIMULTANEOUS MINIMIZATION OF ENERGY AND          3CYT  60
    REMARK  3  R-FACTOR (SEE A.JACK,M.LEVITT, ACTA CRYST., V. A34,         3CYT  61
    REMARK  3  P. 931, 1978).                                              3CYT  62

from PDB 1cvc:
    REMARK  3 REFINEMENT.                                                  1CVC  19
    REMARK  3   PROGRAM                  PROLSQ                            1CVC  20
    REMARK  3   AUTHORS                  KONNERT,HENDRICKSON               1CVC  21
    REMARK  3   R VALUE                  0.170                             1CVC  22
    REMARK  3   RMSD BOND DISTANCES      0.014  ANGSTROMS                  1CVC  23
    REMARK  3   RMSD BOND ANGLES         2.6    DEGREES                    1CVC  24

from PDB 4gsp:
    REMARK  3  FIT TO DATA USED IN REFINEMENT.
    REMARK  3   CROSS-VALIDATION METHOD          : THROUGHOUT
    REMARK  3   FREE R VALUE TEST SET SELECTION  : RANDOM
    REMARK  3   R VALUE            (WORKING SET) : 0.159
    REMARK  3   FREE R VALUE                     : 0.197
    REMARK  3   FREE R VALUE TEST SET SIZE   (%) : 10.0
    REMARK  3   FREE R VALUE TEST SET COUNT      : 1169
    REMARK  3   ESTIMATED ERROR OF FREE R VALUE  : 0.02
    REMARK  3
    REMARK  3  FIT IN THE HIGHEST RESOLUTION BIN.
    REMARK  3   TOTAL NUMBER OF BINS USED           : 8
    REMARK  3   BIN RESOLUTION RANGE HIGH       (A) : 1.65
    REMARK  3   BIN RESOLUTION RANGE LOW        (A) : 1.72
    REMARK  3   BIN COMPLETENESS (WORKING+TEST) (%) : 88.86
    REMARK  3   REFLECTIONS IN BIN    (WORKING SET) : 1207
    REMARK  3   BIN R VALUE           (WORKING SET) : 0.278
    REMARK  3   BIN FREE R VALUE                    : 0.265
    REMARK  3   BIN FREE R VALUE TEST SET SIZE  (%) : 11.6
    REMARK  3   BIN FREE R VALUE TEST SET COUNT     : 158
    REMARK  3   ESTIMATED ERROR OF BIN FREE R VALUE : NULL

from PDB 1jdo:
    REMARK  3  FIT TO DATA USED IN REFINEMENT (NO CUTOFF).
    REMARK  3   R VALUE   (WORKING + TEST SET, NO CUTOFF) : 0.1550
    REMARK  3   R VALUE          (WORKING SET, NO CUTOFF) : 0.1531
    REMARK  3   FREE R VALUE                  (NO CUTOFF) : 0.2059
    REMARK  3   FREE R VALUE TEST SET SIZE (%, NO CUTOFF) : 10.0
    REMARK  3   FREE R VALUE TEST SET COUNT   (NO CUTOFF) : 1630
    REMARK  3   TOTAL NUMBER OF REFLECTIONS   (NO CUTOFF) : 16304
    REMARK  3
    REMARK  3  FIT/AGREEMENT OF MODEL FOR DATA WITH F>4SIG(F).
    REMARK  3   R VALUE   (WORKING + TEST SET, F>4SIG(F)) : 0.1378
    REMARK  3   R VALUE          (WORKING SET, F>4SIG(F)) : 0.1363
    REMARK  3   FREE R VALUE                  (F>4SIG(F)) : 0.1813
    REMARK  3   FREE R VALUE TEST SET SIZE (%, F>4SIG(F)) : 9.8
    REMARK  3   FREE R VALUE TEST SET COUNT   (F>4SIG(F)) : 1361
    REMARK  3   TOTAL NUMBER OF REFLECTIONS   (F>4SIG(F)) : 13842

from PDB 1lhc:
    REMARK  3  FIT TO DATA USED IN REFINEMENT.
    REMARK  3   CROSS-VALIDATION METHOD          : NULL
    REMARK  3   FREE R VALUE TEST SET SELECTION  : NULL
    REMARK  3   R VALUE     (WORKING + TEST SET) : 0.204
    REMARK  3   R VALUE            (WORKING SET) : NULL
    REMARK  3   FREE R VALUE                     : NULL
    REMARK  3   FREE R VALUE TEST SET SIZE   (%) : NULL
    REMARK  3   FREE R VALUE TEST SET COUNT      : NULL
    REMARK  3
    REMARK  3  FIT/AGREEMENT OF MODEL WITH ALL DATA.
    REMARK  3   R VALUE   (WORKING + TEST SET, NO CUTOFF) : NULL
    REMARK  3   R VALUE          (WORKING SET, NO CUTOFF) : NULL
    REMARK  3   FREE R VALUE                  (NO CUTOFF) : NULL
    REMARK  3   FREE R VALUE TEST SET SIZE (%, NO CUTOFF) : NULL
    REMARK  3   FREE R VALUE TEST SET COUNT   (NO CUTOFF) : NULL
    REMARK  3   TOTAL NUMBER OF REFLECTIONS   (NO CUTOFF) : NULL
```

Table 1: Text Fragments from the REMARK 3 records in several PDB files that illustrate the variety of ways R and R_{free} are recorded in the files.

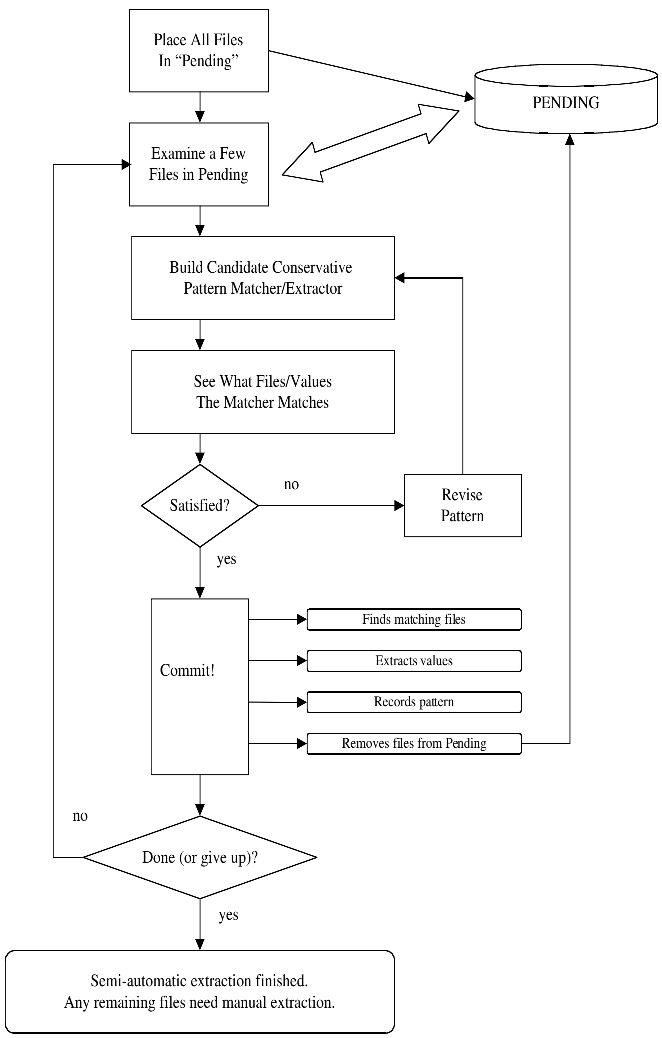

Figure 1: Flowchart of the extraction process when using the High Precision Information Extraction Workbench.

Once a plausible, maximally specific, regular expression has been constructed, the system rapidly applies it to all the PDB files in the directory. The system displays how often the regex matches, which PDB files it matches, what value it would extract from each of the files where it matches, and the text surrounding the match in each file. This allows the user to quickly assess the accuracy and coverage of the proposed regular expression. If the regex matches few or no files, it is necessary to go back and attempt to generalize the regex so it will match more cases. If the regex makes false matches (i.e., matches values that are not the correct R_{free} value in some PDB files), a refinement of the regex is needed so that it will not make any false matches.

After several iterations of this process, the user converges on a regular expression that typically matches 1%-20% of the PDB files with no false matches. Once satisfied, the user commits the matches. Commit does several things. First, it finds all PDB files that match the regex and extracts the matched value from the file. The matched values are placed in a database along with the PDB filename. Then, after extracting the values, it removes all PDB files from the pending files directory that matched the regex and places them in a directory of finished PDB files. Finally, the regex is saved in a growing file of regular expressions. This is done to document the extraction process, and to make it possible to later apply the same extraction steps to future PDB releases, when more files will be available.

Once a regular expression is committed, all PDB files that it matched have had a value extracted and have been removed from the list of pending files. The remaining files have yet to be parsed and are still pending. The user attempts to find a new regular expression that will match the remaining, as yet unextracted, PDB files. Note that the user does not need to worry about the new regex matching cases which have already had the value successfully extracted from them. Once a case has been matched by a regex, it is moved out of the directory. This allows the user to focus on finding patterns that match the currently unmatched files. It also means that the sequence in which regular expressions are applied and saved during the progress of extraction is very important. If the extraction is to be repeated on future PDB cases it has to be done in the same sequence, otherwise false matches may occur.

The user continues through cycles of finding regular expressions and extracting values from the pending PDB files that were not successfully matched by previous patterns. Each time the user commits a new regex, the number of pending PDB files remaining to be processed is reduced.

After about 10-20 regular expressions have been committed, the remaining pending PDB files may be so different from each other that it is difficult to find patterns that match more than a few of them. At this point it is no longer worth the effort to try to devise regular expressions that may only match a few files, and the user has the option of stopping the semi-automatic extraction process and continue with manual extraction.

4. RESULTS

We have tested information extraction with the HPIEW on two kinds of domains. The first is the extraction of crystalographic parameters from files in the Protein Data Bank. The second application was informal testing of the extraction of contact information such as phone numbers and addresses from email. We do not present results on the email domain because those experiments were less complete than the experiments with the PDB files.

4.1 Extracting R and R_{free} Values

In this paper we demonstrate the application of the HPIEW to the extraction of two values from the PDB: R and R_{free}. We began with about 5,200 PDB files that we were interested in. They represent a subset of the PDB consisting of protein structures solved by X-ray crystallography (other types of biomolecules, and structures solved by NMR were excluded). R and R_{free} extractions were performed independently on two different days.

Table 2 shows the augmented regular expressions developed for the extraction of R. These extraction patterns were developed by a biologist and computer scientist working side-by-side using the information extraction workbench. As should be obvious from the table, the language used to represent the extraction patterns is arcane and does not aid interpretation. There are subtleties hidden in the extraction patterns that probably would not be obvious to readers not present while developing the patterns during the actual extraction. (The extraction patterns are a poor form of documentation.)

To illustrate the extraction process, pattern 1 in Table 2 extracts from PDB file 4gsp in Table 1, but does not match the very similar entry for file 1lhc in Table 1 because of the value being "NULL" for that entry; Pattern 3 does the extraction for file 1lhc. A more challenging case is represented by file 1ldm in Table 2 which is not matched by any of the patterns 1-9, but is finally extracted by pattern 10 in Table 2.

Figure 2 shows the number of pending PDB files that remained after each step of the extraction process. In the case of R, the first developed and committed regular expression matched 1/3 of the documents. After the second extraction step only 20% of the documents remained in the pending state. As shown in the graph, subsequent regular expression patterns extracted much fewer documents. After 14 steps 127 files still had not had their R value extracted. They were so heterogeneous that it became difficult to find patterns that matched more than a few documents at a time. After browsing these pending files and making several failed attempts at generating usable patterns, the user decided that the pending set was small enough that manual extraction would be easier and more reliable.

The main difference in extracting R_{free} was due to the fact that only about 1/3 of the PDB files had this value recorded. One reason for this is that R_{free} is a parameter that was introduced into the crystallographic research only recently, and a large portion of files were deposited before that date. A second reason is that even in new structures R_{free} often is not calculated (see PDB 1lhc in Table 1). Therefore, the first step in our extraction process used a regular expression that

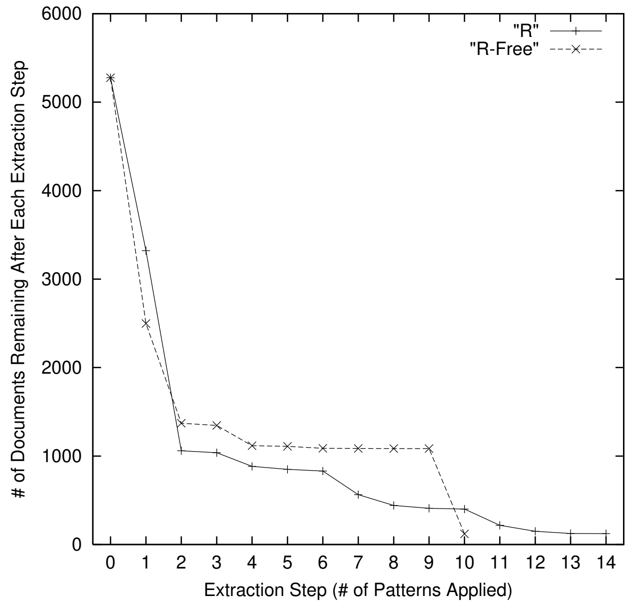

Figure 2: The number of files remaining to be extracted plotted as a function of the number of information extraction steps. (Extraction for R and R_{free} are performed independently.) Each step represents the application of a new pattern matcher/extractor. Note that after extraction for R was halted, 124 files were left that had to have the R value extracted manually – no reliable patterns could be found that matched more than a few files at a time.

eliminated all files that did not contain the word "FREE" in REMARK records, leaving less than 50% of files in the pending state. After applying 8 more patterns for extracting R_{free} it was difficult to design additional patterns to extract more values from the about 1000 remaining files. Step 10 consisted of eliminating files similar to 1lhc in Table 1, that had the value NULL recorded for R free. The remaining 121 files were processed manually and 5 additional values were extracted from them.

4.2 Accuracy Estimation

Evaluating the precision and recall of the extraction process is challenging because the accuracy is so high. While developing a pattern, the user is presented with compact tables of the values that would be extracted if that pattern were to be committed, as well as a compact presentation of the text surrounding that extracted value. This makes it easy for users to evaluate and control the precision of the extraction process while it is in progress. It is easy for the user to quickly examine 100's of matches. Because of this, we suspect that with modest attention, a user with the appropriate expertise is able to keep the error rate well under 1%. This high precision extraction phase of iterative pattern matching is followed by manual processing of the residual documents. The manual phase ensures that a high recall of the extraction process is maintained as well.

One way to evaluate the accuracy of the extraction process is to randomly sample the documents, extract the values manually, and compare these manually extracted values

```
1   s/REMARK   3   R VALUE            (WORKING SET) : \([0-9\.]*\).*/\1/p
2   s/REMARK   3   R VALUE                          \([0-9\.]*\)                  .*/\1/p
3   s/REMARK   3   R VALUE     (WORKING + TEST SET) : \([0-9\.]*\).*/\1/p
4   s/REMARK   3   R VALUE          (WORKING SET, NO CUTOFF) : \([0-9\.]*\).*/\1/p
5   s/REMARK   3   R VALUE         (NO SIGMA CUTOFF) : \([0-9\.]*\)       .*/\1/p
6   s/REMARK   3   R VALUE       (WITH SIGMA CUTOFF) : \([0-9\.]*\)       .*/\1/p
7   s/.*THE R VALUE IS \([0-9]*\.[0-9]*\).*/\1/p
8   s/REMARK   3 [ ]*R[ -]VALUE[ ]*[:]*[ ]*\([0-9]*\.[0-9]*\).*/\1/p
9   s/REMARK   3   R VALUE   (WORKING + TEST SET, NO CUTOFF) : \([0-9][0-9]*\.[0-9][0-9]*\).*/\1/p
10  s/.* R [^z]*zREMARK   3 [ ]*VALUE IS \([0-9][0-9]*\.[0-9][0-9]*\).*/\1/p
11  s/.* R[ -]VALUE[^z]*zREMARK   3 [ ]*IS \([0-9][0-9]*\.[0-9][0-9]*\).*/\1/p
12  s/.* R[ -]VALUE IS[^z]*zREMARK   3 [ ]*\([0-9][0-9]*\.[0-9][0-9]*\).*/\1/p
13  s/.* R[ -]VALUE IS \([0-9][0-9]*\.[0-9][0-9]*\).*/\1/p
```

Table 2: Extraction patterns (regular expressions augmented with SED-style string editing operations) developed to extract R from the PDB files. Note that the order of the extraction patterns is critical.

with those extracted using the HPIEW. We did this for a few hundred PDB files and found no errors in the extracted values. Also, no values were missed. Manual extraction is so tedious and time consuming that it was impractical to do this for a sample large enough to detect error rates less than about 0.5% (we suspect the actual error rates are less than 0.1%). The PDB is so important, however, that there is an effort by the PDB maintainers to use manual extraction by experts to build a parallel structured database. When this database becomes available we will be able to find discrepancies between the values we extracted with HPIEW, and the manually extracted values.

5. DISCUSSION AND FUTURE WORK

The HPIEW currently requires the user to generate and refine the regular expression patterns. Regular expression learning methods might be able to assist the user in this process [8]. For example, the user might be able to select a subset of documents and highlight the text to be extracted from each of those, and then ask the system to induce a maximally specific regular expression to match those cases. The user might also select some negative instances that are near misses, and these might be used to help refine the regular expression somewhat automatically.

An important issue when designing and evaluating high precision information extraction systems is how to measure the accuracy of the system. The high accuracy of these systems makes it very difficult to measure. One approach that might be used to facilitate evaluating such systems is to have the system automatically find near matches or ambiguous matches. These are cases that either barely match the pattern that will be applied to them, or cases which match, or nearly match, several patterns. We suspect that these are some of the cases that are most likely to be extracted incorrectly. Being able to focus in on such cases might make it easier to find errors the system makes, and thus better evaluate its performance. Of course these methods might also be useful in helping the user construct better patterns that make fewer errors, thus improving the system's accuracy.

The augmented regular expression language we used for the PDB may not be the most appropriate pattern matching language for other extraction problems. Other applications might do better with other pattern matching grammars. While we have designed the HPIEW to be compatible with other extraction grammars, we have only tested it with the augmented regular expressions.

Another domain where we have tested the HPIEW using augmented regular expressions is an email application where the goal was to extract phone numbers, dates, and addresses from email archives. The system was effective for these extraction tasks. Interestingly, the ability to quickly locate emails that contained extractable fields made the tool surprisingly useful for information retrieval. For example, if you are looking for an email that contains a phone number located in the body of the email, the HPIEW could be used to quickly find the subset of documents matching any of the phone number patterns, and display the text surrounding these matched patterns to the user. Thus the user was able to quickly identify most of the name/phone number pairs in their email, and thus could locate a particular phone number they were searching for. Dayne Freitag built a prototype user interface for an HPIEW customized for text applications and for using the system for information retrieval. Freitag's user interface is far more pleasant to use than the HPIEW we used for the PDB applications. His system still enables users to build, test, and evaluate new patterns, but the emphasis is on using these to aid rapid interactive information retrieval rather than information extraction.

The importance of the user interface for an high precision information extraction workbench can not be over emphasized. The goal of an HPIEW is to amplify the ability of a human expert to do semi-automatic extraction. Doing this requires that the user receive rapid feedback in a form that enables them to quickly scan for possible extraction errors. The interface also needs to provide the user with enough information for them to quickly understand the variety of text remaining to be extracted. The workbench we created for the PDB extraction was fairly crude. It used a number of cooperating shell scripts running in side-by-side windows on a workstation. It was not an integrated workbench with a uniform MMI. Despite it's simplicity, however, this interface did provide the right information to the user in a compact and easily interpreted form. It was also quick enough to allow the user to rapidly try different extraction patterns and evaluate how well they performed. The interface developed by Freitag for information extraction from email was more tightly integrated and easier to use than the PDB interface. We believe developing custom workbenches for different types of extraction applications will be important for maximum efficiency and accuracy.

We used the high precision information extraction workbench to extract two different values, R and R_{free}, from over 5000 PDB files. We believe the extractions for these fields is comparable in accuracy to manual tagging the files. This creates an interesting opportunity. The high-precision extraction has created two tagged data sets that may be useful for others doing research in information extraction who would benefit from labeled data sets. One characteristic of these data sets that makes them different from other data sets is that the text is not completely free-form. The nature of the entries in the REMARK fields of the PDB files probably makes extracting the right values more difficult because there are other text fields that are very similar. For example, the text surrounding the R and R_{free} values in the files tends to be very similar, and thus might introduce confusion when trying to extract these values correctly. We would be happy to make our labelled data sets available to other researchers in text extraction.

6. ACKNOWLEDGMENTS

We thank Dayne Freitag for many useful discussions, and for his development of a prototype HPIEW for extraction from email. This work was supported in part by NLM grants LM07059 and LM06759 and by NCRR grant RR10447.

7. ADDITIONAL AUTHORS

Bruce Buchanan
Department of Computer Science
University of Pittsburgh
200 Lothrop Street
Pittsburgh, PA 15213
buchanan@cs.pitt.edu

8. REFERENCES

[1] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein Data Bank. *Nucleic Acids Res.*, 28(1):235–242, 2000.

[2] A. T. Brünger. Free R value: a novel statistical quantity for assessing the accuracy of crystal structures. *Nature*, 355:472–475, 1992.

[3] C. Cardie and D. Pierce. Proposal for an interactive environment for information extraction. In *Technical Report TR98-1702, Cornell University Computer Science*, September 1998.

[4] O. Glickman and R. Jones. Examining machine learning for adaptable end-to-end information extraction systems. In *AAAI 1999 Workshop on Machine Learning for Information Extraction*, 1999.

[5] R. Jones, K. Nigam, A. McCallum, and E. Riloff. Bootstrapping for text learning tasks. In *IJCAI-99 Workshop on Text Mining: Foundations, Techniques and Applications*, 1999.

[6] G. J. Kleywegt. Validation of protein crystal structures. *Acta Cryst.*, D56:249–265, 2000.

[7] R. A. Laskowski, M. W. MacArthur, and J. M. Thornton. Validation of protein models derived from experiment. *Curr. Opin. Struct. Biol.*, 8:631–639, 1998.

[8] E. Riloff. Automatically generating extraction patterns from untagged text. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 1044–1049, 1996.

[9] J. D. Westbrook and P. E. Bourne. STAR/mmCIF: An ontology for macromolecular structure. *Bioinformatics*, 16(2):159–168, 2000.