Pacific Symposium on Biocomputing 13:640-651(2008)

INTRINSIC EVALUATION OF TEXT MINING TOOLS MAY NOT PREDICT PERFORMANCE ON REALISTIC TASKS

J. GREGORY CAPORASO[1], NITA DESHPANDE[2], J. LYNN FINK[3], PHILIP E. BOURNE[3], K. BRETONNEL COHEN[1], AND LAWRENCE HUNTER[1]

[1] *Center for Computational Pharmacology*
University of Colorado Health Sciences Center, Aurora, CO, USA
[2] *PrescientSoft Inc., San Diego, CA, USA*
[3] *Skaggs School of Pharmacy and Pharmaceutical Sciences*
University of California, San Diego, San Diego, CA, USA

Biomedical text mining and other automated techniques are beginning to achieve performance which suggests that they could be applied to aid database curators. However, few studies have evaluated how these systems might work in practice. In this article we focus on the problem of annotating mutations in Protein Data Bank (PDB) entries, and evaluate the relationship between performance of two automated techniques, a text-mining-based approach (MutationFinder) and an alignment-based approach, in intrinsic versus extrinsic evaluations. We find that high performance on gold standard data (an intrinsic evaluation) does not necessarily translate to high performance for database annotation (an extrinsic evaluation). We show that this is in part a result of lack of access to the full text of journal articles, which appears to be critical for comprehensive database annotation by text mining. Additionally, we evaluate the accuracy and completeness of manually annotated mutation data in the PDB, and find that it is far from perfect. We conclude that currently the most cost-effective and reliable approach for database annotation might incorporate manual and automatic annotation methods.

1. Introduction

Biomedical text mining systems have been reaching reasonable levels of performance on gold standard data, and the possibility of applying these systems to automate biological database construction or annotation is becoming practical. These systems are generally evaluated intrinsically—for example, against a gold standard data set with named entities that are tagged by human annotators, judging the system on its ability to replicate the human annotations. Systems are less commonly evaluated extrinsically—i.e., by measuring their contribution to the performance of some task. Intrinsic evaluations of text mining tools are critical to accurately assessing their basic functionality, but they do not necessarily tell us how well a system will perform in practical applications.

Hunter and Cohen (2006) list four text mining systems that are being or have been used to assist in the population of biological databases (LSAT,[2] MuteXt,[3] Textpresso,[4] and PreBIND[5]). Of these four, data on the actual contribution of the tool to the database curation effort is available for only one: the PreBIND system is reported to have reduced the time necessary to perform a representative task by 70%, yielding a 176-person-day time savings. More recently, Karamanis et al. (2007) recorded granular time records for a "paper-by-paper curation" task over three iterations in the design of a curator assistance tool, and noted that curation times decreased as user feedback was incorporated into the design of the tool. In the information retrieval (IR) domain, Hersh et al. (2002) assessed the ability of an IR tool (Ovid) to assist medical and nurse practitioner students in finding answers to clinical questions, and found that performance of the system in intrinsic evaluation did not predict the ability of the system to help users identify answers. Some tasks in the recent BioCreative shared tasks (particularly the GO code assignment task in BioCreative 2004, PPI task in BioCreative 2006, and the GN tasks in both years), and to a lesser extent, of the TREC Genomics track in some years, can be thought of as attempts at extrinsic evaluations of text mining technologies[a]. Camon et al. (2005) gives an insightful analysis of the shortcomings of the specific text mining systems that participated in the BioCreative 2004 GO code assignment task. We are not aware of any work that directly assesses the ability of an automated technique to recreate a large, manually curated data set, although the importance of such evaluations has been noted.[8]

There has recently been much interest in the problem of automatically identifying point mutations in text.[3,9–15] Briefly, comprehensive and accurate databases of mutations that have been observed or engineered in specific biological sequences are often extremely valuable to researchers interested in those sequences, but because the requisite information is generally dispersed throughout the primary literature, manually compiling these databases requires many expert hours[b]. To address this issue, we have developed MutationFinder,[9] an open source, high performance system for identifying descriptions of point mutations in text. We performed an in-depth intrinsic evaluation of MutationFinder on blind, human-annotated test data. For extracting mentions of point mutations from MEDLINE abstracts, the most difficult task it was evaluated on, it achieved 98.4% precision and 81.9% recall.

The availability of this system allows us to ask subsequent questions. First, how effective are manual biological database annotation techniques in terms of accuracy and coverage; and second, does the performance of an automated annotation technique in intrinsic evaluation predict the performance of that system in an extrinsic evaluation? The first question

[a]In fact, some of the earliest work on information extraction in the modern era of BioNLP, such as Craven and Kumlein (1999) and Blaschke et al. (1999), can be thought of as having extrinsic, rather than intrinsic, evaluation.

[b]We present the problem of identifying mutations in text, our approach to addressing it, and a review of the approaches taken by other groups in Caporaso et al. (2007b).

addresses the issue of whether replacement or augmentation of manual database annotation methods with automatic methods is worth exploring, while the second addresses whether the most commonly performed evaluations of automatic techniques translate into information regarding their applicability to real-world tasks.

To address these questions, we compare and evaluate three approaches for annotating mutations in Protein Data Bank (PDB)[17] entries[c]: manual annotation, which is how mutations are currently annotated in the PDB; and two automatic approaches—text-mining-based annotation using MutationFinder, and alignment-based annotation—which we are exploring as possibilities to replace or augment manual annotation. (The PDB is the central repository for 3D protein and nucleic acid structure data, and one of the most highly accessed biomedical databases.) In the following section we present our methods to address these questions and the results of our analyses. We identify problems with all of the annotation approaches, automatic and manual, and conclude with ideas for how to best move forward with database annotation to produce the best data at the lowest cost.

2. Methods and Results

In this section we describe the methods and results of three experiments. First, we evaluate the accuracy and comprehensiveness of the manual mutation annotations in the Protein Data Bank. Then, we extrinsically evaluate our two automated techniques by comparing their results with the manually deposited mutation data in the PDB. Finally, we compare MutationFinder's performance when run over abstracts and full text to address the hypothesis that MutationFinder's low recall in extrinsic evaluation is a result of the lack of information in article abstracts. Unless otherwise noted, all evaluations use a snapshot of the PDB containing the 38,664 PDB entries released through 31 December 2006. All data files used in these analyses are available via `http://mutationfinder.sourceforge.net`.

2.1. Evaluation of manual annotations

When a structural biologist submits a structure to the PDB, they are asked to provide a list of any mutations in the structure. Compiling this information over all PDB entries yields a collection of manually annotated mutations associated with PDB entries, and this mapping between PDB entries and mutations forms the basis of our analyses. We evaluate the accuracy of these annotations by comparing mutation field data with sequence data associated with the same entries. We evaluate the completeness of this data by looking for PDB entries which appear to describe mutant structures but do not contain data in their mutation fields.

[c]Entries in the PDB are composed of atomic cartesian coordinates defining the molecular structure, and metadata, including primary sequence(s) of the molecule(s) in the structure, mutations, primary citation, structure determination method, etc.

2.1.1. Manual mutation annotations

Manual mutation annotations were compiled from the *mutation field* associated with PDB entries. The mutation field is a free-text field in a web-based form that is filled in by researchers during the structure deposition process. The depositor is expected to provide a list of mutations present in the structure (e.g., 'Ala42Gly, Leu66Thr'[d]), but the information provided is not always this descriptive. For example, many mutations fields contain indecipherable information, or simply the word *yes*. In cases where the depositor does not provide any information in the mutation field (as is often the case), differences identified by comparison with an aligned sequence are suggested to the author by a PDB annotator. The author can accept or decline these suggestions.

2.1.2. Normalization of manual mutation annotations

Because the mutation field takes free text input, automated analysis requires normalization of the data. This was done by applying MutationFinder to each non-empty mutation field. Point mutations identified by MutationFinder in a mutation field were normalized. To evaluate this normalization procedure, a non-author biologist manually reviewed a random subset (n=400) of non-empty mutation fields and the normalized mutations output by MutationFinder. Precision of the normalization procedure was 100.0%. Recall was 88.9%. This high performance is not surprising, since the task was relatively simple. It suggests that normalizing mutation fields with MutationFinder is acceptable. 10,504 point mutations in 5971 PDB records were compiled by this approach. This data set is referred to as the *manually deposited mutation annotations*.

2.1.3. Accuracy of manually deposited mutation annotations

To assess the accuracy of the manually deposited mutation annotations, each mutation was validated against the sequence data associated with same PDB entry. (This process is similar to that employed by the MuteXt[3] system.) If a mutation could not be validated against the sequence data, that entry was considered to be inaccurately annotated and was reported to PDB. (Note that this discrepancy could indicate an error in the sequence, an error in the mutation annotation, or a mismatch in sequence numbering.)

Validation of mutations against sequence data was performed as follows. Sequences were compiled for all PDB entries. For a given entry, we checked whether the putative mutant residue was present at the annotated sequence position. For example, PDB entry 3CGT is annotated with the mutation `E257A`. The sequence associated with 3CGT was scanned to determine if alanine, the mutant residue, was present at position 257. In this case it was, so the annotation was retained. If alanine were not present at position 257, the

[d]This is a common format for describing point mutations, which indicates that alanine at position 42 in the sequence was mutated to glycine, and leucine at position 66 was mutated to threonine.

annotation would have been labelled as invalid. In cases where PDB entries contain multiple sequences (e.g., a protein composed of several polypeptide chains), each sequence was checked for the presence of the mutant residue.

2.1.4. Coverage of manually deposited mutation annotations

To assess the coverage of the manually annotated data, we attempted to identify PDB records of mutant structures that did not contain data in their mutation field. To identify records of mutant structures, we searched PDB entry titles for any of several keywords that suggest mutations (case insensitive search query: `muta* OR substitut* OR varia* OR polymorphi*`). MutationFinder was also applied to search titles for mentions of point mutations. If a title contained a keyword or mutation mention and the entry's mutation field was blank, the entry was labelled as insufficiently annotated. An informal review of the results suggested that this approach was valid.

2.1.5. Results of manual annotation evaluation

40.6% (4260/10504) of mutations mentioned in mutation fields were not present at the specified position in the sequence(s) associated with the same PDB entry. These inconsistencies were present in 2344 PDB entries, indicating that 39.3% of the 5971 PDB entries with MutationFinder-normalizable mutation fields may be inaccurately annotated. As mentioned, these inaccurate annotations could be due to errors in the mutation annotation or the sequence, or mismatches between the position numbers used in the mutation and the sequence. We expect that in the majority of cases the errors arise from mismatches in numbering, as there is generally some confusion in how mutations should be numbered (i.e., based on the sequence in the structure or based on the UniProt reference sequence). PDB entries now contain mappings between the structure and UniProt sequences, and in a future analysis we will use these mappings to determine if any of these apparent errors are instead inconvenient discrepancies which could be avoided automatically.

Additionally, 21.7% (1243/5729) of the PDB entries that contained a mutation keyword or mention in the title were found to contain an empty mutation field. These entries appear to be underannotated. (As a further indication of the scope of the "underannotation problem," note that 12.9% (1024/7953) of the non-empty mutation fields simply contain the word *yes.*) Again, this is likely to be an overestimate of the true number of underannotated PDB entries (due to promiscuity of the search query), but even if we are overestimating by a factor of 10, this is still a problem.

These results suggest that the manually deposited mutation data is far from perfect, and that not just the quantity, but the *quality* of manual database annotation stands to be improved. In the next section, we explore automated techniques for mutation annotation in the PDB to determine if they may provide a means to replace or augment manual annotation. These automated techniques are evaluated against the manually deposited mutation annotations, although we have just shown that it is far from perfect. Performance of the automated techniques is therefore underestimated.

2.2. Automated mutation annotation evaluated extrinsically

In this section two automated mutation annotation techniques are evaluated by assessing their ability to reproduce the manually deposited mutation annotations in the PDB. The first automated method, text mining for mutations using MutationFinder, has been shown to perform very well on blind test data (i.e., in intrinsic evaluation). Our second approach, detecting differences in pre-aligned sequences, is not inherently error-prone, and therefore does not require intrinsic evaluation. We might expect that the near-perfect and perfect abilities of these systems (respectively) to perform the basic function of identifying mutations would suggest that they are capable of compiling mutation databases automatically. Assessing their ability to recreate the manually deposited mutation annotations allows us to evaluate this expectation.

2.2.1. Text-mining-based mutation annotation: MutationFinder

Two sets of MutationFinder mutation annotations were generated—with and without the sequence validation step described in Section 2.1.3. The unvalidated data should have higher recall, but more false positives. To compile the unvalidated MutationFinder annotation set, MutationFinder was applied to primary citation abstracts associated with PDB records. For each record in the PDB, the abstract of a primary citation (when both a primary citation and an abstract were available) was retrieved, and MutationFinder was applied to extract normalized point mutations. 9625 normalized point mutations were associated with 4189 PDB entries by this method, forming the *unvalidated MutationFinder mutation annotations.* To compile the *validated MutationFinder mutation annotations*, we applied sequence validation to the unvalidated MutationFinder mutation annotations. This reduced the results to 2602 normalized mutations in 2061 PDB entries.

2.2.2. Alignment-based mutation annotation

Validated and unvalidated data sets were also compiled using an alignment-based approach. Sequences associated with PDB entries were aligned with UniProt sequences using `bl2seq`. Differences between aligned positions were considered point mutations, and were associated with the corresponding entries. The alignment approach yielded 23,085 normalized point mutations in 9807 entries (the *unvalidated alignment mutation annotations*). Sequence validation[e] reduced this data set to 14,284 normalized mutations

[e]Sequence validation was somewhat redundant in this case, but was included for completeness. Surprisingly, it was not particularly effective here. The positions assigned to mutations in this approach were taken from the aligned UniProt sequence when sequence start positions did not align perfectly, or when the alignment contained gaps. This resulted in different position numbering between the manually- and alignment-produced annotations, and reduced performance with respect to the manual annotations.

in 6653 entries (the *validated alignment mutation annotations*).

2.2.3. Extrinsic evaluation of automated annotation data

To assess the abilities of the MutationFinder- and alignment-based annotation techniques to recreate the manual annotations, mutation annotations generated by each approach were compared with the manually deposited mutation annotations in terms of precision, recall, and F-measure using the `performance.py` script[f]. Two metrics were scored: *mutant entry identification*, which requires that at least one mutation be identified for each mutant PDB entry, and *normalized mutations*, which requires that each manually deposited mutation annotation associated with a PDB entry be identified by the system. *Mutant entry identification* measures a system's ability to identify structures as mutant or non-mutant, while *normalized mutations* measures a system's ability to annotate the structure with specific mutations.

Normalized mutations were judged against the manually deposited mutation annotations, constructed as described in Section 2.1.1. This set contained 10,504 mutations in 5971 PDB records from a total of 38,664 records. As we noted earlier, many non-empty mutation fields do not contain mutations (e.g., when they contain only the word *yes*). However, in the vast majority of cases, a non-empty mutation field indicates the presence of mutations in a structure. We therefore constructed a different data set for scoring mutant entry identification. We generated a *manually curated mutant entry* data set from all PDB entries which contained non-empty mutation fields. This data set contained 7953 entries (out of 38,664 entries in the PDB snapshot).

2.2.4. Extrinsic evaluation results

We assess the utility of the automated techniques (and combinations of both) for identifying mutant PDB entries (*mutant entry identification*, Table 1a) and annotating mutations associated with PDB entries (*normalized mutations*, Table 1b).

On both metrics, the highest precision results from the intersection of the validated MutationFinder mutation annotations (method 2) and the unvalidated alignment mutation annotations (method 3) data, while the highest recall results from the union of these. Generally, method 2 achieves high precision, and method 3 achieves high recall. None of these approaches achieves a respectable F-measure, although as we point out in Section 2.1.5, these performance values are likely to be underestimates due to noise in the manually deposited mutation annotations.

2.3. MutationFinder applied to abstracts versus full-text

Table 1 shows that MutationFinder (with and without validation) achieves very low recall with respect to the manually deposited mutation annotations. We evaluated the hypothesis that this was a result of mutations not

[f]Available in the MutationFinder package at `http://mutationfinder.sourceforge.net`.

Table 1. Six automated methods for identifying mutant PDB entries are assessed against (a) manually curated mutant entry data, and (b) manually deposited mutation annotations. True positives (TP), false positives (FP), false negatives (FN), precision (P), recall (R), and F-meaure (F) are presented. Sequence validation (methods 2 and 4) is described in Section 2.1.3.

	(a) Mutant Entry Id.	TP	FP	FN	P	R	F
1	MutationFinder	2690	1499	5263	0.642	0.338	0.443
2	MutationFinder + validation	1665	396	6288	**0.808**	0.209	0.333
3	Alignment	6079	3728	1874	0.620	**0.764**	0.685
4	Alignment + validation	4104	2549	3849	0.617	0.516	0.562
5	2 *and* 3	1403	275	6550	**0.836**	0.176	0.291
6	2 *or* 3	6258	3816	1695	0.621	**0.787**	0.694

	(b) Normalized Mutations	TP	FP	FN	P	R	F
1	MutationFinder	2575	7050	7929	0.268	0.245	0.256
2	MutationFinder + validation	1803	799	8701	**0.693**	0.172	0.275
3	Alignment	7681	15404	2823	0.333	**0.731**	0.457
4	Alignment + validation	5059	9225	5455	0.354	0.482	0.408
5	2 *and* 3	1584	532	8920	**0.749**	0.151	0.251
6	2 *or* 3	7900	15671	2604	0.335	**0.752**	0.464

being mentioned in article abstracts, but rather only in the article bodies. A PDB snapshot containing the 44,477 PDB records released through 15 May 2007 was used for this analysis.

2.3.1. Compiling and processing full-text articles

PubMed Central was downloaded through 15 May 2007. XML tags and metadata were stripped. All articles were searched for occurrences of a string matching the format of a PDB ID. (IDs are four characters long: a number, a letter, and two letters or numbers, e.g. 3CGT.) If such a string was found, it was compared to a list of valid PDB IDs; if the string matched a valid PDB ID, the article was retained. This returned 837 articles. From this set, articles that were primary citations for PDB structures were selected, resulting in a set of 70 PDB entries (with 13 manually annotated mutations) for which full text was available.

2.3.2. Comparing abstracts versus full text

MutationFinder with sequence validation (as described in Section 2.1.3) was applied to the abstracts and full-text articles, yielding two mutation data sets. The results were compared against the manually annotated mutation data, allowing us to directly assess the contribution of the article bodies to MutationFinder's performance.

2.3.3. Abstract versus full text results

A 10-fold increase in recall was observed when the article body was provided to MutationFinder in addition to the abstract, with no associated degradation of precision (Table 2). While 70 PDB entries with 13 mutations is a small data set, these data strongly suggest that access to full text is critical for automated mutation annotation by text mining tools. When sequence validation was not applied, normalized mutation and mutant entry identifi-

Table 2. MutationFinder with sequence validation was applied to abstracts and full articles (abstract + article body) for 70 PDB entries. Results are compared with manually annotated data. True positives (TP), false positives (FP), false negatives (FN), precision (P), recall (R), and F-meaure (F) are presented, describing each approach's ability to replicate manually curated data.

Metric	Input	TP	FP	FN	P	R	F
Normalized Mutations	Abstracts	1	0	12	1.000	0.077	0.143
	Full text	10	0	3	1.000	0.769	0.870
Mutant Entry Id.	Abstracts	1	0	9	1.000	0.100	0.182
	Full text	7	0	3	1.000	0.700	0.824

cation recall were perfect, but precision was 11.7% and 38.5%, respectively.

3. Conclusions

These experiments address two questions. First, how effective are manual biological database annotation techniques in terms of accuracy and coverage; and second, does the performance of an automated annotation technique in intrinsic evaluation predict the performance of that system in an extrinsic evaluation? We now present our conclusions regarding these questions, and discuss their implications for database curation.

3.1. Reliability of mutation annotation approaches

The manual *and* automatic approaches to annotating mutations appear to yield significant Type I and II errors when analyzed on the PDB as a whole. This suggests that these methods may be insufficient to generate the required quality and quantity of annotation that is necessary to handle the barrage of data in the biomedical sciences.

Manual annotation of PDB entries is error-prone, as illustrated by our sequence-validation of these data described in Section 2.1.5, and does not guarantee complete annotation. (It should be noted that many of the results that are classified as errors in the manually annotated data are likely to be due to sequence numbering discrepanices. Mappings between PDB sequences and UniProt sequences in the PDB can be used to identify these, and in a future analysis these mappings will be used to reevaluate the manually annotated data.)

The automated mutation annotation approaches also appear to have limitations. MutationFinder (with validation against sequence data) performs well, but full text is probably required for any text mining approach to achieve sufficient recall. Conversely, the alignment-based approach is comprehensive, but overzealous. The manual and automatic methods do frequently validate and complement one another (data not shown due to space restrictions)—in parallel, they may provide a means for improving the quality, while reducing the cost (in person-hours), of database annotation.

3.2. MutationFinder: intrinsic versus extrinsic evaluations

In an intrinsic evaluation against blind gold standard data, MutationFinder achieved 97.5% precision and 80.7% recall on normalized mutations extraction, and 99.4% precision and 89.0% recall on document retrieval.[9, 10] In our extrinsic evaluation against manually deposited mutation annotations in the PDB, the exact same system achieved 26.8% precision and 24.5% recall for normalized mutation extraction, and 64.2% precision and 33.8% recall for mutant entry identification (the equivalent of document retrieval in this work). While these are likely to be underestimates of the true utility (Section 2.1.5), the large difference in performance cannot be explained completely by the imperfect extrinsic evaluation. The discrepancy appears to be chiefly explained by two factors: introduction of a systematic source of false positives, and missing data. These issues illustrate that accurately and comprehensively pulling desired information from text is just the beginning of deploying a text mining system as a database curation tool.

False positives were systematically introduced when a single article was the primary citation for several PDB entries, and MutationFinder associated all mutations mentioned in the article with all the citing entries[g]. Our sequence validation step addressed this issue, and improved normalized mutation precision by 42.5 percentage points with an associated degradation in recall of 7.4 percentage points.

False negatives were most common when the targeted information was not present in the primary citation abstracts. In our abstract versus full text analysis, we found that processing the full text with MutationFinder plus sequence validation resulted in a nearly 70 percentage point increase in recall, with no precision degradation. These data result from an analysis of only a small subset of the PDB, but they clearly illustrate the importance of full text for high-recall mutation mining.

We conclude that while it is an essential step in building a text mining system, evaluating a system's performance on gold standard data (intrinsic evaluation) is not necessarily indicative of its performance as a database curation tool (extrinsic evaluation). Identifying and gaining access to the most relevant literature, and identifying and responding to sources of systematic error, are central to duplicating the performance observed on a (well-chosen) gold standard data set in an extrinsic evaluation.

3.3. Alignment-based mutation annotation: extrinsic evaluation

Compiling differences in aligned sequences is not inherently error prone, unlike text mining—beyond unit testing to avoid programming errors, no intrinsic evaluation is necessary. However, this method does not perform perfectly for annotating mutations in the PDB, but rather achieves high recall with low precision.

[g]For example, PDB entries 1AE3, 1AE2, and 1GKH all share the same primary citation (PMID: 9098886). This abstract mentions five mutations, all of which MutationFinder associates with each of the three PDB entries. Each of the structures contains only one of the five mutations, so four false positives are incurred for each entry. (The other two mutations referred to are in other structures.) Sequence validation eliminated all of these false positives while retaining all of the true positives.

Error analysis suggests that the primary cause of both false positives and false negatives obtained by alignment-based mutation annotation with respect to the manually deposited mutation annotations is differences in sequence position numbering between the PDB sequence and the UniProt sequence. In PDB entry 1ZTJ, for example, the authors annotated mutations S452A, K455A, T493A, and C500S, while sequence comparison identified S75A, K78A, T115A, and C123S. The (almost identical) relative sequence positions and wild-type and mutant residues suggest that these are the same mutations, but the sequence position offset results in four false positives and four false negatives. Utilizing the mappings between PDB sequence positions and UniProt sequence positions in the PDB should help to alleviate these discrepancies in position numbering. This will be explored in future work, and is expected to significantly reduce these types of errors.

False positives additionally occur as a result of slight differences in the sequence of the solved structure and the closest sequence in UniProt. Differences in the sequences are not *necessarily* mutations induced for analysis, and are therefore not annotated as such. For example, sequence comparison identified six mutations in the PDB entry 1QHO, and the primary citation authors acknowledge several of these as sequence 'discrepancies.' False negatives can also occur when a sequence cannot be aligned to a UniProt sequence, and the alignment-based method cannot be applied, or alternatively if inaccurate information was provided by the depositor. For example, PDB entry 1MWT is annotated with the Y23M mutation, but valine is present at position 23 in the associated sequences. In this case the classification as false negative is an artifact of a problematic manual annotation, rather than a statement about the performance of the annotation technique.

4. Discussion

Automatic annotation can not yet replace manual database curation, even for the relatively simple task of annotating mutations in molecular structures. We evaluated manual curation and two automated methods, and showed that all three are unreliable. Genomic data and their reliable annotation are essential to progress in the biomedical sciences. It has been shown empirically that manual annotation cannot keep up with the rate of biological data generation;[20] furthermore, we have shown here that even if manual annotation could keep pace with data generation, it is still error prone.

A reasonable approach to pursue is the incorporation of automated techniques into manual annotation processes. For example, when a scientist deposits a new PDB structure, their primary citation and sequences can be scanned for mutations. The depositor could be presented with suggestions: *In your abstract, you mention an A42G mutation—is this mutation present in your structure?* Additionally, these tools can be applied as quality control steps. Before a mutation annotation is accepted, it could be validated against sequence data. Responses to such prompts could be recorded and used to generate new gold standards that could be used to improve existing or future tools for automating annotation procedures. 'Smart' annotation deposition systems could be the key to improved quality of data in the present and improved automated techniques in the future.

Acknowledgments

The authors would like to acknowledge Sue Brozowski for evaluating the MutationFinder normalization of PDB mutation fields, Jeffrey Haemer, Kristina Williams, and William Baumgartner for proof-reading and helpful discussion, and the four anonymous reviewers for their insightful feedback. Partial funding for this work came from NIH grants R01-LM008111 and R01-LM009254 to LH.

References

1. Hunter, L. and Cohen, K.B., *Mol Cell* 21, 589–594 (2006).
2. Shah, P.K. and Bork, P., *Bioinformatics* 22, 857–865 (2006).
3. Horn, F., Lau, A.L. and Cohen, F.E., *Bioinformatics* 20, 557–568 (2004).
4. Müeller, H., Kenny, E.E. and Sternberg, P.W., *PLoS Biol* 2, e309 (2004).
5. Donaldson, I., Martin, J., de Bruijn, B., Wolting, C., Lay, V., Tuekam, B., Zhang, S., Baskin, B., Bader, G.D., Michalickova, K., Pawson, T. and Hogue, C.W.V., *BMC Bioinformatics* 4, 11 (2003).
6. Hersh, W.R., Crabtree, M.K., Hickam, D.H., Sacherek, L., Friedman, C.P., Tidmarsh, P., Mosbaek, C, and Kraemer, D. *J American Medical Informatics Association* 9, 283–293 (2002).
7. Karamanis, N., Lewin, I., Sealy, R., Drysdaley, R., and Briscoe, E., *Pacific Symposium on Biocomputing* 12, 245–256 (2007).
8. Cohen, A.M. and Hersh, W., *Briefings in Bioinformatics* 6, 57–71 (2005).
9. Caporaso, J.G., Baumgartner Jr., W.A., Randolph, D.A., Cohen, K.B. and Hunter, L., *Bioinformatics* 23, 1862–1865 (2007).
10. Caporaso, J.G., Baumgartner Jr., W.A., Randolph, D.A., Cohen, K.B. and Hunter, L., *J. Bioinf. and Comp. Bio. (accepted, pub. Dec. 2007)*, (2007b).
11. Rebholz-Schuhmann, D., Marcel, S., Albert, S., Tolle, R., Casari, G. and Kirsch, H., *Nucl. Acids Res.* 32, 135–142 (2004).
12. Baker, C.J.O. and Witte, R., *Journal of Information Systems Frontiers* 8, 47–57 (2006).
13. Lee, L.C., Horn, F. and Cohen, F.E., *PLoS Comput Biol* 3, e16 (2007).
14. Bonis, J., Furlong, L.I. and Sanz, F., *Bioinformatics* 22, 2567–2569 (2006).
15. Witte, R., Kepler, T., and Baker, C.J.O., *Int J Bioinformatics Research and Methods* 3 389–413 (2007).
16. Camon, E.B., Barrell, D.G., Dimmer, E.C., Lee, V., Magrane, M., Maslen, J., Binns, D. and Apweiler, R., *BMC Bioinformatics* 6 Suppl 1, S17 (2005).
17. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N. and Bourne, P.E., *Nucleic Acids Res* 28, 235–242 (2000).
18. Craven, M. and Kumlien, J., *ISMB 1999*, (1999).
19. Blaschke, C., Andrade, M.A., Ouzounis, C. and Valencia, A., *ISMB 1999*, 60–67 (1999).
20. Baumgartner Jr., W.A., Cohen, K.B., Fox, L.M., Acquaah-Mensah, G. and Hunter, L. *Bioinformatics* 23, 14 (2007).