Extraction of Named Entities from Tables in Gene Mutation Literature

***Wern Wong**[*], **David Martinez**[**], **Lawrence Cavedon**[**]

[**]NICTA Victoria Research Laboratory
[*]Dept of Computer Science and Software Engineering
The University of Melbourne

{*wongwl,davidm,lcavedon*}*@csse.unimelb.edu.au*

Abstract *We investigate the challenge of extracting information about genetic mutations from tables, an important source of information in scientific papers. We use various machine learning algorithms and feature sets, and evaluate performance in extracting fields associated with an existing hand-created database of mutations. We then show how classifying tabular information can be leveraged for the task of named entity detection for mutations.*[1]

Keywords Information extraction; tables; biomedical applications.

1 Introduction

We are interested in applying *information extraction* and *text mining* techniques to aiding the construction of databases of biomedical information, in particular information about genetic mutations. Such databases are currently constructed by hand: a long, involved, time-consuming and human-intensive process. Each paper considered for inclusion in the database must be read, the interesting data identified and then entered by hand into a database.[2]

However, the biomedical domain throws up many new and serious challenges to information extraction and text mining. Unusual terminology and underdeveloped standards for nomenclature present problems for tokenisation and add complexity to standard information extraction tasks, such as *named entity recognition (NER)*. A lack of resources (at least compared to other domains), such as collections of annotated full-text documents and relevance judgements for various tasks, are a bottleneck to developing and evaluating the core techniques required.

In this paper, we report on work performed on extracting information from *tables* in biomedical research papers. Tables present a succinct and information-rich format for providing information, and are particularly important when reporting results in biological and medical research papers. For example, the Human Genome Variation Society (HGVS), in its general recommendations for mutation nomenclature, recommends making use of tabular listings when several changes are described in a manuscript.[3] A specific premise of our work is that the highly-structured nature of tabular information allows leverage of some techniques that are not so sensitive to the well-reported problems inherent in biomedical terminology, which complicate NER tasks in this domain. In particular, we describe initial techniques for extending NER performance through the analysis of tables: columns/rows are classified as containing items of the entities of interest, thereby allowing those entities to be recognized as of the target type. Since a significant amount of such entities may be found in tables in biomedical scientific papers, this can have positive impact on the performance of base NER techniques.

NER tools specifically targeted at recognising mutations have been developed (e.g. (Horn et al., 2004; Baker and Witte, 2006; Caporaso et al., 2007; Lee et al., 2007)); however, they only detect a subclass of mutations, so-called *single-point mutations*,

[1] A short version of this paper was presented at the *Australasian Document Computing Symposium*, 2008. All copyrights from that event were retained by the authors.

[2] Karamis *et al* (2008) illustrate how even simple tools can have an impact on improving the database-curation process.

[3] http://www.hgvs.org/mutnomen/recs.html#general

i.e. those that affect a single base. MutationFinder (Caporaso et al., 2007) is the only publicly available tool, built with around 700 automatically-generated rules (both for different nomenclatures and natural language). However, most of the mutations that we find in our dataset are not point mutations or do not follow point-mutation nomenclature, limiting the usefulness of MutationFinder (and related tools) over our document collection.

In the next section, we describe the setting of our task, the *Mismatch Repair (MMR) Database*, and outline the task of extraction from tables. In Section 3, we describe the preparation of our document collection, and in Section 4, we analyse the amount of mutation-related information that is in the associated tables. Section 5 describes the main task, which is classifying table rows and columns as containing mutations, and Section 6 leverages this technique to detect mutations of interest to the MMR Database. We discuss the results in Section 7.

2 Background

In this section, we discuss the MMR database—the setting for our task and from which we construct our document collection—and previous approaches to table processing.

2.1 The MMR Database

Our extraction task is grounded in the specific context of the *Mismatch Repair (MMR) Database* compiled at the Memorial University of Newfoundland (Woods et al., 2007)—a database of known genetic mutations related to hereditary non-polyposis colorectal cancer (HNPCC), a hereditary form of bowel cancer. The MMR Database contains information on genetic mutations known to be related to HNPCC, along with links to the research papers from which the database has been constructed.[4] From the database and its links to papers, we were able to construct a collection of tables related to HNPCC mutations, and then use the MMR database records themselves as a gold standard for evaluating our techniques. As of May 2008, the MMR database contained a total of 5,491 records on mutations that occur on any one of four genes that have been identified as related to colon cancer. An example record from the MMR database is the following:

MLH1	Exon13	c.1491delG	Yamamoto et al.	9500462

Respectively, this record contains: gene; exon; mutation; citation of the paper the information was sourced from;[5] and the paper's PubMedID. These fields are important because they contain information researchers are directly interested in (gene, exon, mutation) and the paper said information was found in. Note that if a gene/mutation pair is referenced in multiple papers, then there are correspondingly multiple entries in the database. Conversely, if a single paper mentions multiple (relevant) genes, then that paper is mentioned in multiple database records.

2.2 Table Processing

An important but less-researched sub-problem in text mining is information extraction from tables. This is particularly important in the biomedical domain since much important data is present in tabular form, such as experimental results, relations between entities, and other information that may not be contained elsewhere in the text. For example, the table shown in Figure 1 (taken from an article in our collection) contains much of the same data that was present in database records, in a similar format.

Tabular information extraction can be divided into two broad sub-tasks:

- table detection: identifying tables within documents;
- table processing: extraction of data from tables.

Several systems have been developed to handle both tasks, some are designed only to handle table detection, and others focus only on extracting data. Both machine learning and heuristic / rule-based approaches have been proposed.

Table detection techniques depend heavily on the input format. Most work that tackles this problem tends to assume one homogeneous input format, but tables generally come in one of two varieties:[6]

[4] I.e. a team of geneticists manually trawled the biomedical literature for information on HNPCC-related mutation information, and added links to any papers relevant to those mutations in the context of HNPCC.

[5] This field has been abbreviated. We have also omitted fields such as "internal id".

[6] We don't consider the possibility of processing bitmaps or other images from scanned documents.

Category	Patient No	Family No	Gene	Exon	Base change	Base No	Codon	AA change	MSI	Comment
1	1	1	MLH1	11	Del G	1046	349	Frameshift	MSI-H	Stop codon 380
Amsterdam	2	1	MLH1	11	Del G	1046	349	Frameshift	MSI-H	Stop codon 380
criteria I	3	2	MLH1	16	Del AAG	1846–8	616	Del Lys	MSI-H	Pathogenic, see text
	4	3	MSH2	12	Del AAT	1786–8	596	Del Asn	MSI-H	Pathogenic, see text
	5	4	MSH2	11	T>A	2 bases downstream	—	Splice mutation	MSI-H	Pathogenic, see text
2	12	11	MLH1	1	C>T	76	26	Gln>stop	MSI-H	Stop codon
Amsterdam	13	12	MSH2	12	Del AAT	1786–8	596	Del Asn	MSI-H	Pathogenic, see text
criteria II	16	4	MSH2	11	T>A	2 bases downstream	—	Splice mutation	MSI-H	Pathogenic, see text
6*	37	34	MSH2	12	Del AAT	1786–8	596	Del Asn	MSI-H	Pathogenic, see text
7*	42	39	MSH2	12	Del AAT	1786–8	596	Del Asn	MSI-H	Pathogenic, see text

*Category 6, proband at age 40 years or younger with at least one CRC among family members; category 7, both proband and one first degree relative with CRC <55 years.

MSI, microsatellite instability; MSI-H, high microsatellite instability; MSI-L, low microsatellite instability; MS-S, stable microsatellites; CRC, colorectal cancer.

Figure 1: Sample table containing mutation information related to HNPCC

- raw text tables: generally ASCII text in monospace font, delimited by whitespace and/or special characters;
- rich text tables: those formatted using LaTeX, PDF, HTML and other such formats.

Tables in plain text tend to be more difficult to detect, as the detection system must be sensitive to whitespace and symbols used to align cells in tables. Efforts to handle rich text formats generally focus on HTML-based representations. Raw HTML is easier to parse than raw LaTeX or PDF, and most formats are easily converted to HTML. HTML tables can theoretically be trivially detected using `<table>` tags. However, Lerman *et al* (2004) note that in HTML files taken from the web, only a fraction of tabular data was presented using `<table>` tags, and those tags were also used to format multi-column text, images and other non-table applications. Hurst (2001) attests that less than 30% of HTML tables on the web contain actual tabular content; for many, the HTML table tags are often used simply for formatting purposes.

Zanibbi *et al* (2004) present a survey of table recognition in general. Of greatest relevance to us here are approaches that adopt a machine learning approach to detecting and/or extracting table data.

Cohen *et al* (2002) use features based on HTML table tags, number of rows and columns of specific lengths, and ratios of single-square cells to total number of cells, to perform table detection, and then form a geometric representation of the data using algorithms based on table-rendering techniques implemented by browsers.

Pinto, Wei, and their colleagues have used *conditional random fields (CRFs)* to both detect and process tables simultaneously. Pinto *et al* (2003) compare the output of their CRF system with a previous effort using hidden Markov machines (HMMs). These systems use features such as: presence of whitespace of varying length (different lengths of whitespace are used as separate features); domain-specific lexical features (such as month names, year strings, specified keywords); separator characters (e.g. '+', '-', etc). In subsequent work they develop a system for performing question answering over table data (Wei et al., 2004) by treating each extracted data cell as a discrete document.

To our knowledge, no previous system has attempted to extract data from tables in biomedical literature. This is possibly because of a combination of the lack of resources for this domain (e.g.

collections of full-text documents; relevance judgements), as well as the lesser focus on text mining in general in this area. As will be seen in the next section, the vagaries of the construction of our collection of tables means we were effectively able to ignore the issue of table detection and focus directly on the problem of processing.

3 Experimental Setting

Our experiments were designed to identify mentions of mutations in the biomedical literature, focusing on tabular content. In this section, we first describe our target dataset, built from the hand-curated MMR database (Woods et al., 2007); we then explain the table extraction process; finally, we introduce the task design.

3.1 Mutation Mention Dataset

We relied on the MMR Database and MEDLINE in order to build our test collection. First we collected all the information available in the hand-curated MMR records, obtaining a total of 5,491 mutations linked to 719 distinct PubMedIDs[7].

Our next step was to crawl the full-text articles from MEDLINE. We used an automatic crawler that followed the links from the PubMed interface, and downloaded those papers that had a full-text HTML version, and which contained at least one content table.

The tables were then extracted from the full text HTML files. It is important to note that the tables were already present as links to separate HTML files rather than being presented as inline tables, making this process easier. Papers that did not contain tables in HTML format were discarded.

Our final collection consisted of 70 papers out of the original 719 PubMedIDs. Some of the papers were not available in full text, and for others our crawling script failed to extract the full version. Our approach was conservative, and our collection could be augmented in the future, but we decided to focus on this dataset for the experiments presented in this paper. This set of articles is linked to 717 MMR records (mutations), which constitutes our gold standard hand-curated annotation. The collection contains 197 tables in all.

[7] Data was downloaded from the web interface in May 2008.

3.2 Table extraction

Once scraped, the tables were then pre-processed into a form that more readily allowed experimentation. The tables were therefore split into three parts: column headers, row headers, and data cells. This was done based on the HTML formatting, which was consistent throughout the data set as the tables were automatically generated.

The first step was to deconstruct the HTML tables into nested lists of cells based on HTML table tags. Inconsistencies introduced by `colspan` and `rowspan` attributes were resolved by replicating a cell's contents across its spanned lengths. That is, a cell with `colspan=3` would be duplicated across the three columns, and likewise for cells spanning multiple rows. Single-cell rows at the top or bottom of a table were assumed to be captions and discarded.

The remaining HTML was stripped, save for the following tags which contained important information:

- `img` tags were replaced by their *alternate* text, where available. Such images often represent a mathematical symbol, which is important information to retain;
- `hr` tags proved to be an important indicator for dividing header cells from data cells.

Tables were broken up into row headers, column headers, and data cells by making use of the `hr` tags, denoting horizontal lines, to detect column headers. Such tags tend to be present as a separator between column header cells and data cells; in fact, the only tables in our collection that did not have the separators did not have column headers either. The `hr` tags were subsequently stripped after this use. Detecting row headers was performed by checking if the top left cell of the table was blank, a pattern which occurred in all row-major tables. The vast majority of tables had column headers rather than row headers, although some had both and a small proportion had only row headers. We acknowledge that this processing may be specific to the vagaries of the specific format of the HTML generation used by PubMed (from which we sourced the tables). However, our whole task is specific to this domain; further, our focus is on the extraction task rather than the actual detection of row/column headers.

Class	Class Freq.	Cell Freq.
Gene	64	1,618
Exon	48	1,004
Codon	23	435
Mutation	90	2,174
Statistic	482	8,788
Other	576	14,324
Total	1,283	28,343

Table 1: Frequency per class and number of cells in the collection.

3.3 Task Design

In order to extract mutation information from tables, we first performed classification of full columns/rows into relevant entities. The content of a column (or row, depending on whether the table was row- or column-oriented) tends to be homogeneous; this allowed us to build classifiers that can identify full vectors of relevant entities in a single step. We refer to this task as *table vector classification*.

We identified the following classes as relevant: Gene, Exon, Mutation, Codon, and Statistic. The first four were chosen directly from the MMR Database. We decided to include "Statistic" after inspecting the tabular dataset, since we found that this provides relevant information about the importance of a given mutation. Of the five classes, Mutation is the most informative for our final information extraction goal.

The next step was to hand-annotate the headers of the 197 tables in our collection by using the five classes and the class "Other" as the tagset. Some headers belonged to more that one class, since the classes were collapsed into a single field of the table. The frequency per class and the number of cells, across the collection of tables, is shown in Table 1.

3.4 Evaluation

We evaluated our systems in two ways:

- Header classification: performance of different systems on predicting the classes of each column/row of the tables;
- Mutation extraction: recall of our system over the subset of the hand-curated MMR database.

Evaluation for the header classification step was performed using precision, recall and f-score, micro-averaged amongst the classes. Micro-averaging involves multiplying the score of a class by the number of instances of the class in the gold standard, and dividing by the total number of instances. For the machine learning algorithms, evaluation was performed using 10-fold cross-validation. For mutation extraction we focus on a single class, and produce recall and a lower-bound on precision.

4 Mutation Mentions in Tables

In order to determine the value of processing tabular data for mutation-mining purposes, we obtained a sample of 100 documents that were hand-annotated by curators prior to their introduction in the database—the curators highlighted relevant mutations found in each paper. We found that for 59 of the documents, only the tabular parts of the paper were selected; 33 of the documents had only textual parts highlighted; and for 8 documents both tables and text were selected. This is an indicator of the importance of tabular data in this context.

Our next step was to measure the amount of information that we could potentially extract from the tables in our collection. Since we are interested in mutations, we extracted all cells from the vectors that were manually annotated as "Mutation" in order to compare them to the goldstandard, and measure the recall. This comparison was not straightforward, because mutation mentions have different nomenclatures. Ideally we would normalise the different references into a standard form, and then perform the comparison. However, normalisation is a complex process in itself, and we resorted to evaluation by hand at this point.

We found that 198 of the 717 goldstandard mutations were present in tables (28%). This is a significant amount, taking into account that their extraction should be much easier than parsing the raw text. We also tested MutationFinder over the full text, and found that only 6 of the goldstandard mutations were retrieved (0.8%), which indicates that point mutation identification is not sufficient for this task.

Finally, we measured the amount of information that could be extracted by a simple string look-up system separately over the tabular and textual parts

of the articles. We were looking for mutation mentions that correspond exactly to the goldstandard record from each article, which meant that mentions in different nomenclatures would be missed. We found that a total of 177 mentions (24.7%) could be found with the same spelling; of those 142 (80.1%) were found in tables only, and the remaining 35 (20.9%) were found in both tables and text; i.e., no mention was found in text only.

These results indicate that we can find relevant information in tables that is not easy to detect in running text.

5 Table Vector Classification

We built automatic classifiers to detect relevant entities in tables. Two separate approaches were attempted for vector classification: applying heuristic rules, and machine learning (ML) techniques. These are described here, along with an analysis of their performance.

5.1 Heuristic Baseline

As a baseline method, we approached the task of classifying headers by matching the header string to the names of the classes in a case-insensitive manner. When the class name was found as a substring of the header, the class would be assigned to it. For example, a header string such as "Target Mutation" would be assigned the class "Mutation". Some headers had multiple annotations (E.g. "Gene/Exon").

For better recall, we also matched synonyms for the class "Mutation" (the terms "Variation" and "Missense") and the class "Statistic" (the terms "No.", "Number" and "%"). For the remaining classes we did not identify other obvious synonyms.

The results are shown in Table 2. Precision was reasonably high for the "Codon", "Exon" and "Statistic" classes. However, this was not the case for "Mutation", and this illustrates that different types of information are provided under this heading; illustrative examples include the heading "Mutation Detected" on a "Gene" vector, or the heading "Germline Mutation" referring to "Statistics". The recall was also low for "Mutation" and most other classes, showing that more sophisticated approaches are required in order to exploit the information contained in the tables. Notice also that the micro-average is highly biased by the classes "Statistic" and "Others", since they contain most of the test instances.

Class	Precision	Recall	FScore
Gene	0.537	0.620	0.575
Exon	0.762	0.615	0.681
Codon	0.850	0.654	0.739
Mutation	0.283	0.301	0.292
Statistic	0.911	0.324	0.478
Other	0.581	0.903	0.707
Micro Avg.	**0.693**	**0.614**	**0.651**

Table 2: Naive Baseline results across the different classes and micro-averaged

Class	Precision	Recall	FScore
Gene	0.537	0.611	0.571
Exon	0.762	0.615	0.681
Codon	0.850	0.654	0.739
Mutation	0.600	0.452	0.515
Statistic	0.911	0.340	0.495
Other	0.579	0.910	0.708
Micro Avg.	**0.715**	**0.633**	**0.672**

Table 3: Results integrating MutationFinder across the different classes and micro-averaged

Our second step was to build a more informed classifier for the class "Mutation" using the point mutation NER system MutationFinder (Caporaso et al., 2007). We applied this tool to the text in the table-cells, and identified which table-vectors contained at least one mutation mention. These vectors were also classified as mutations. The results are shown in Table 3. This approach caused the "Mutation" results to improve, but the overall f-score values are still in the range 50%-70%.

We considered other heuristic rules that could be applied, such as looking for different kinds of patterns for each class: for instance, numbers for "Exon", or the normalised form `c.N[symbol]N` for mutation, or trying to match against term lists (e.g. using Gene dictionaries). Future work will explore extending the ML approach below with features such as these.

5.2 Classification Techniques

For the ML experiments we used the Weka (Witten and Frank, 2005) toolkit, as it contains a wide selection of in-built algorithms. We selected a variety of well-known approaches in order to obtain a better picture of the overall performance. As a baseline, we applied the majority class from the training data to all test instances. We applied the following ML systems:[8] Naive Bayes (NB); Support Vector Machines (SVM); Propositional Rule Learner (JRip); and Decision Trees (J48). We did not tune the parameters, and relied on the default settings.

In order to define our feature sets, we used the text in the headers and cells of the tables, without tokenisation. Other possible sources of information, such as captions or the running text referring to the table were not employed at this stage. We applied four feature sets:

- **Basic (Basic):** Four basic features, consisting of the header string, the average and median cell lengths, and a binary feature indicating whether the data in the cells was numeric.
- **Cell Bag-of-Words (C_bow):** Bag of words over the tokens in the table cells.
- **Header Bag-of-Words (H_bow):** Bag of words over the tokens in the header strings.
- **Header + Cell Bag-of-Words (HC_bow):** Combination of bags of words formed by the tokens in headers and cells, represented as separate types of features.

The micro-averaged results of the different learning methods and feature sets are shown in Table 4. Regarding the feature sets, we can see that the best performance is obtained by using the headers as bag-of-words, while the content of the cells seems to be too sparse to guide the learning methods. SVM is the best algorithm for this dataset, with JRip and J48 following, and NB performing worst of the four in most cases.

Overall, the results show that the ML approach is superior to the baselines when using the header bag of words feature to classify the relevant entities.

[8] We applied a number of other ML algorithms as well, but these showed significantly lesser performance.

Method	Feature Sets			
	Basic	**C_bow**	**H_bow**	**HC_bow**
Mj. Cl.	0.288			
NB	0.614	0.454	0.678	0.581
SVM	**0.717**	**0.599**	**0.839**	**0.816**
JRip	0.564	0.493	0.790	0.749
J48	0.288	0.532	0.793	0.782

Table 4: Results for ML Algorithms - Micro-Averaged FScores. Mj.Cl.: Majority Class. The best results per column are given in bold.

Class	Precision	Recall	FScore
Gene	0.778	0.737	0.757
Exon	0.786	0.707	0.745
Codon	0.833	0.882	0.857
Mutation	0.656	0.679	0.667
Statistic	0.919	0.853	0.885
Other	0.82	0.884	0.850
Micro Avg	**0.839**	**0.841**	**0.839**

Table 5: Results for SVM and the feature set *H_bow* per class and micro-averaged.

SVM is able to reach a high f-score of 83.9%, which has been found to be significantly better than the best baseline after applying a paired t-test (p-value under 0.0001).

We break down the results per class in Table 5, using the outputs from SVM and feature-set *H_bow*. We can see that all classes show an improvement over the heuristic baselines. There is a big increase for the classes "Gene" and "Statistic", and all classes except mutation are above 70% f-score. "Mutation" is the most difficult class to predict, but it still reaches 66.7% f-score, which can be helpful for some tasks, as we explore in the next section.

6 Automatic Mutation Extraction

We applied the results of our classifier to a practical application, i.e., the detection of mutations in the literature for the MMR Database project. Table vector classification allows us to extract lists of candidate mutation names from tables to be added to the database. We would like a system with high recall that identifies all relevant candidates, but also acceptable precision so that not all the tables need to

System	Mut. Found	TP	% in MMR	Rec.
Automatic	1,702	153	9.0	77.3
Gold standard	1,847	198	10.7	100

Table 6: Results for Mutation detection. TP indicates the number of true positives, "% in MMR" shows the percentage of positives found in the database.

be hand-checked.

In order to test the viability of this approach, we measured the results of the system in detecting the existing hand-curated mutations in MMR. We calculated the recall in retrieving those mutations, and also the rate of false positives; however, note that we also consider as false positives those valid mutations that were not relevant for MMR, and therefore the reported precision is artificially low.

Results for the automatic extraction and the gold-standard annotation are given in Table 6. As expected, there is a high rate of false positives in the goldstandard and automatic systems; this shows that most of the mutations detected are not relevant for the MMR database. More interestingly, we were able to retrieve 77.3% of relevant mutation mentions automatically using the ML approach, which corresponds to 21.3% of all the hand-curated data.

The vector classifier discriminates 1,702 mutation cells out of a total of 27,700 unique cells in the table collection, and it effectively identifies 153 out of the 198 relevant mutations present in the tabular data. This means that we only need to hand-check 6.1% of the tabular content to retrieve 77.3% of relevant mutations, saving the curators a significant amount of time. The classifiers could also be biased towards higher recall by parameter tuning—this is an area for further investigation.

Finally, after the evaluation process we observed that many false mutation candidates could be removed by discarding those that do not contain two consecutive digits or any of the following n-grams: "c.", "p.", '>', "del", "ins", "dup". This heuristic reduces the number of mutation candidates from 1,702 to 989 with no cost in recall.

7 Discussion

While this is early work, our preliminary results on the task of identifying relevant entities from gene mutation literature show that targeting tables can be a fruitful approach for text mining. By relying on ML methods and simple bag-of-words features, we were able to achieve good performance over a number of selected entities, well above header word-matching baselines. This allowed us to identify lists of mentions of relevant entities with minimal effort. An advantage of our approach is that the annotation of examples for training and evaluation is considerably easier, since many entities can be annotated in a single step, opening the way to faster annotation of other entities of interest in the biomedical domain.

The approach of using table vector classification for the named entity task also has promise. In particular, the wide variety and non-standard terminology of biomedical entities (i.e. genes, proteins, mutations) is one of the challenges to NER in this domain. However, since a column of homogeneous information may include representatives of the heterogeneous nomenclature schemes, classification of a whole column or row potentially helps nullify the effect of the terminological variability.

For future work, we plan to study different types of features for better representing the entities targeted in this work. Specially for mutation mentions, we observed that the presence of certain ngrams (e.g. "del") can be a strong indicator for this class. Another issue we plan to address is that of the normalisation of mutation mentions into a standard form, for which we have started developing a collection of regular expressions. Another of our goals is to increase the size of our dataset of articles by improving our web crawler, and by hand-annotating the retrieved table vectors for further experimentation. Finally, we also aim to explore the potential of using tabular data for NER of different entities in the biomedical domain, such as gene mentions.

Acknowledgements NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.
Thanks to Mike Woods and his colleagues at the Memorial University of Newfoundland for making the MMR database and their curation data available to us. Eric Huang wrote several of the scripts mentioned in Section 3 for creating the table collection.

References

C. J. O. Baker and R. Witte. 2006. Mutation mining–a prospector's tale. *J. of Information Systems Frontiers*, 8(1):45–57.

J. G. Caporaso, W. A. Baumgartner Jr., D. A. Randolph, K. B. Cohen, and L. Hunter. 2007. Mutationfinder: A high-performance system for extracting point mutation mentions from text. *Bioinformatics*, 23(14):1862–1865.

W. W. Cohen, M. Hurst, and L. S. Jensen. 2002. A flexible learning system for wrapping tables and lists in html documents. In *WWW '02: Proc. 11th Int'l Conf. on World Wide Web*, pages 232–241, Honolulu.

F. Horn, A. L. Lau, and F. E. Cohen. 2004. Automated extraction of mutation data from the literature: Application of MuteXt to g protein-coupled receptors and nuclear hormone receptors. *Bioinformatics*, 20(4):557–568.

M. Hurst. 2001. Layout and language: Challenges for table understanding on the web. Technical report, WhizBang!Labs.

N. Karamanis, R. Seal, I. Lewin, P. McQuilton, A. Vlachos, C. Gasperin, R. Drysdale, and T. Briscoe. 2008. Natural language processing in aid of flybase curators. *BMC Bioinformatics*, 9:193–204.

Lawrence C. Lee, Florence Horn, and Fred E. Cohen. 2007. Automatic extraction of protein point mutations using a graph bigram association. *PLoS Computational Biology*, 3(2):e16+, February.

K. Lerman, L. Getoor, S. Minton, and C. Knoblock. 2004. Using the structure of web sites for automatic segmentation of tables. In *SIGMOD'04*, pages 119–130, Paris.

D. Pinto, A. McCallum, X. Wei, and W. B. Croft. 2003. Table extraction using conditional random fields. In *SIGIR '03*, pages 235–242.

X. Wei, W.B. Croft, and D. Pinto. 2004. Question answering performance on table data. *Proceedings of National Conference on Digital Government Research*.

I. H. Witten and E. Frank. 2005. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2nd edition.

M.O. Woods, P. Williams, A. Careen, L. Edwards, S. Bartlett, J. McLaughlin, and H. B. Younghusband. 2007. A new variant database for mismatch repair genes associated with lynch syndrome. *Hum. Mut.*, 28:669–673.

R. Zanibbi, D. Bolstein, and J. R. Cordy. 2004. A survey of table recognition. *Int'l J. on Document Analysis and Recognition*, 7(1).