TINTIN: A System for Retrieval in Text Tables

Pallavi Pyreddy and W. Bruce Croft
Center for Intelligent Information Retrieval
Dept. of Computer Science
University of Massachusetts
Amherst, MA 01003
{pyreddy,croft}@cs.umass.edu

Abstract

Tables form an important kind of data element in text retrieval. Often, the gist of an entire news article or other exposition can be concisely captured in tabular form. In this paper, we examine the utility of exploiting information other than the key words in a digital document to provide the users with more flexible and powerful query capabilities. More specifically, we exploit the structural information in a document to identify tables and their component fields and let the users query based on these fields. Our empirical results have demonstrated that heuristic method based table extraction and component tagging can be performed effectively and efficiently. Moreover, our experiments in retrieval using the TINTIN system have strongly indicated that such structural decomposition can facilitate better representation of user's information needs and hence more effective retrieval of tables.

1 Introduction

Much of the current literature in information retrieval deals with word based retrieval[6, 1, 7]. We are interested in exploiting the information available in the regularities encountered in text to augment the capabilities of such word-based retrieval systems. Next generation information access tools need to incorporate notions of structure in a document into indexing and retrieval in order to give users additional power and flexibility in query specification. In this paper we deal with one kind of structural element in documents - text tables.

Tables form an important kind of data element in text retrieval. Often, the gist of an entire news article or other exposition can be concisely captured in tabular form. Extracting tabular data, decomposing it into its constituent elements such as table entries, captions, column and row headings, and exploiting the information in these data may lead to enhanced accuracy in word-based intelligent information retrieval systems. Such tabular data may be embedded in the text of a document and there may be no SGML markers separating it from the rest of the text. In many digital library environments, the data is likely to be highly heterogeneous and such tags are often not available. In this paper, we present the Table INformation-based Text INquery (TINTIN) system that uses heuristic methods to extract structural elements from text and separates out tables. TINTIN incorporates a capability to exploit various weighting heuristics to index and retrieve information based on the structural features of the extracted tables.

DL 97 Philadelphia PA, USA

2 Related Work

The work that is most closely related to ours is that of Rus and Subramanian and Rus and Allan[5, 4]. They are primarily interested in the notion of incorporating cues from structural regularities in a document into indexing and retrieval. Structural elements include pictures, graphs, tables, paragraph and section boundaries. Rus and Subramanian[5] use a data structure called a White Space Density Graph, similar to our Character Alignment Graph (explained later), for extracting tables from documents. However, they do not distinguish between various table components and hence their work lacks the notion of structured queries on fields in the table. Thus, while Rus and Subramanian have a more general notion of structural elements in a document, our work has a deeper notion of the constituent structural elements of a document. Much of the work in TINTIN went into distinguishing the structural elements within a table.

Some related work also exists in the automated document structuring community. Much of the work in this community deals with bit-mapped images[3, 2, 8] (unlike ASCII text documents) and it is primarily concerned with detecting structure in the documents for traditional image processing and pattern recognition tasks. On the other hand, our work is primarily motivated by the need to empower users of information retrieval systems through detection and exploitation of structure in digital documents.

3 The TINTIN System

The TINTIN system consists of an initial preprocessing stage where it looks at a regular text database and extracts table data from text documents. The preprocessing stage also tags the components of a table as caption, headline or table entries. The table output of this stage is then indexed and the users are allowed to query on this structured document database. Figure 1 shows the architecture of the TINTIN system. We will now discuss each of these components in detail.

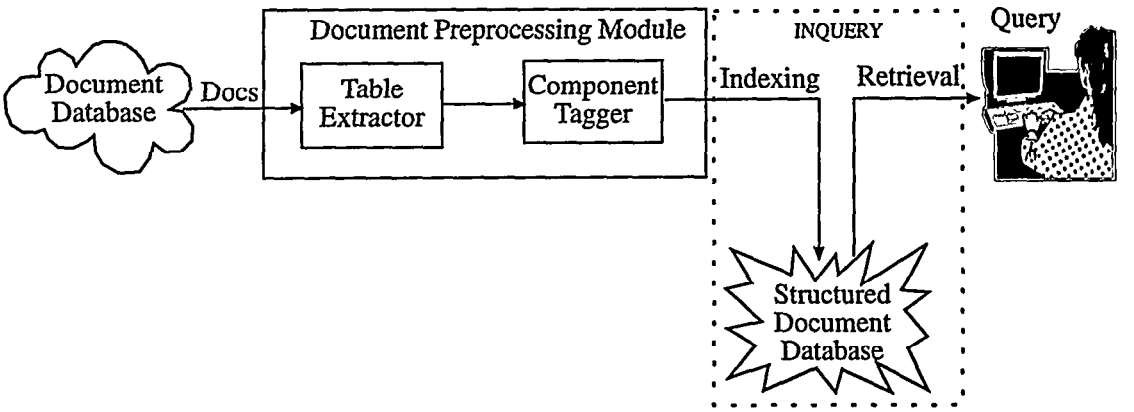

Figure 1: Architecture for the TINTIN System

3.1 Preprocessing Module

The preprocessing module uses a number of structural heuristics to separate a table from the rest of the textual data and to tag the elements of the table as captions or table entries. It consists of two stages: Table Extractor and Component Tagger.

3.1.1 Table Extractor

The Table Extractor is based on the relatively simple and intuitive idea of looking for aligned white spaces. However, text can be noisy and tables tend to have intermittent irregularities that can adversely affect a straightforward algorithm.

Our table extractor relies on a data structure called the Character Alignment Graph (CAG). A CAG is formed by checking for whitespace alignment in blocks of contiguous lines of text. A number of other heuristic parameters like holes (number of blanks between columns) and gaps (number of such contiguous blanks per line) are used to check for the gap structure in a block of data. The intent is to make the extractor as flexible and error tolerant as possible. It is important to note that the table extractor and the subsequent component tagger do not rely on SGML or similar tags to extract the tables. In fact the Wall Street Journal Database that we use does not demarcate the tables from the rest of the text in which they are embedded using any kind of SGML markers.

The CAG data structure is a histogram of the number of characters that appear in a particular location in 'k' (a tunable parameter) contiguous lines. Within the CAG array, we detect "gaps" by looking for contiguous locations whose histogram values are below a threshold. Such "gaps" are identified as potential column separators and the 'k' lines are potentially a part of a table. The tolerance of the extractor not only makes it error tolerant but also performs the important function of acquiring the potential caption data that surrounds the table lines. We experimentally determined a value of k=3 to give good results for the Wall Street Journal Database and the minimum number of gaps (number of contiguous blanks per line) to be 3 for a line to belong to a table.

Figure 2 shows an example of an extracted table. The column and row headings, and the relevant captions are extracted along with the table entries. However, they are not yet tagged as such. This is the job of the component tagger.

3.1.2 Component Tagger

The output of the table extractor is a text table along with the associated captions, column headings and legends. However, the table extractor does not distinguish between these elements. This is the job of the component tagger. Even though this initially looked like a simple thing to do, it turned out to be the most challenging and interesting part of the work due to the numerous formats in which textual tables occur. We had to appeal to a range of heuristics to filter, separate, undo the interactions between previous applications of heuristics and to take into account idiosyncrasies of the databases.

We approached the task of recognizing captions and table lines using largely syntactic heuristics as opposed to semantic ones. Column headings are also treated as captions. Table lines were tagged using <TABLE> ... </TABLE> tags and the captions were tagged using <CAPTION> ... </CAPTION>. In order to first convince ourselves that syntactic heuristics can be effective, we used a simple but interesting technique. We replaced each of the characters in the extracted tables by a "*" and left the blanks as they are. Figure 3 shows an example of such a "star table". It is apparent that the exact semantics of the words in the table are not needed to infer that the first 4 lines are captions. This technique was extremely helpful in letting us view the structural patterns in tables without being confounded by the semantics of the words during the component tagger development phase.

Some of the heuristics we found useful are:

- *Gap Structure Heuristic*: Large empty spaces (big gaps) in the middle of the line (not at beginning or end of the line) indicate that it is a potential table line. Note however, that column headings, which should be tagged as captions, may also be mistakenly tagged as table lines by this heuristic. The later heuristics take care of undoing these mistakes.

- *Alignment Heuristic*: If the gaps and characters of two or more lines are aligned in the middle (as opposed to beginning or end of the line), then it is most likely to be a table entry.

- *Pattern Regularity Heuristic*: Context information or regularity can also be used to separate table lines from captions. For example, in the following pattern the third and fifth lines are more likely to belong to the table rather than captions. The captions occurring

Such overtime throughout the past year helped Ford boost 1988-model car and truck production 19% from 19-7 levels, the company said.

Auto Production

Fourth-quarter U.S. auto production

	1988	1987	PERCENT CHANGE
GM	964,653	843,908	+14.3%
Ford	456,954	461,150	- 0.9
Chrysler	294,998	283,062	+ 4.2
Honda	86,330	80,227	+ 7.6
Mazda	59,464	3,800	----
Nissan	24,604	15,289	+60.9
Toyota	19,600	0	----
Nummi	16,156	38,901	-58.5
Diamond Star	2,316	0	----
TOTAL	1,925,075	1,726,337	+11.6%

First-quarter U.S. auto production

	1989	1988	PERCENT CHANGE
GM	925,000	780,213	+ 18.7%
Ford	475,000	479,263	- 0.9
Chrysler	271,000	261,433	+ 3.7
Honda	99,360	92,156	+ 7.8
Mazda	61,510	16,800	+266.1
Nissan	29,522	36,324	- 18.7
Toyota	25,000	------	------
Nummi	48,000	41,479	+ 15.7
Diamond Star	10,000	------	------
TOTAL	1,945,392	1,707,673	+ 13.9%

NOTE: Estimated from industry and company sources. Vehicles are counted by manufacturer; the Mazda to-al includes cars built for Mazda and Ford;Diamond-Star includes Chrysler and Mitsubishi cars and Nummi -ncludes Chevrolet and Toyota cars.

Figure 2: A table extracted from the Wall Street Journal Database by the table extractor

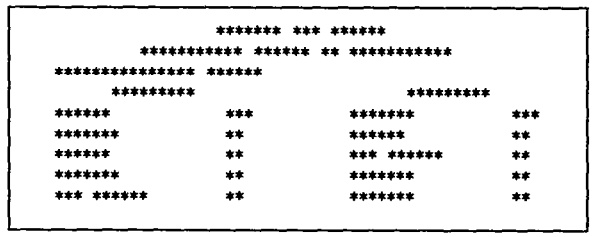

Figure 3: What are the captions here?

between table lines in such patterns are often table lines that have a different gap structure than the other table lines.

```
Caption
Table-Line
Caption
Table-Line
Caption
Table-Line
```

Figure 4 is a sample table illustrating this.

On the other hand, in the following pattern, the fourth line is more likely to be a caption.

```
Table-Line
Table-Line
Table-Line
Caption
Table-Line
Table-Line
Table-Line
```

Figure 5 is a sample table illistrating this.

- *Differential Column Count Heuristic*: If at the beginning of a table, the first few lines have fewer columns than the average columns in the table, then these lines are potential captions. For example, if there are 4 columns in a table and a line has only 3 columns and is at the beginning of the table, it is a potential caption. An illustration of the use of this heuristic can be seen in Figure 6.
- *Differential Gap Structure Heuristic*: Any lines at the beginning or end of a table whose gap alignment does not match the average gap alignment for that table are potential captions. An illustration of the use of this heuristic can be seen in Figure 7.
- If there is a line after a caption and its format is different from that of the average table format, then it is subjected to further tests by the following heuristics.
  1. Captions are text whereas the table entries are mostly numerical.
  2. The column headings are in upper case (capital letters) while table entries are lower case or mixed.
- Additional heuristics that may be applicable to several other databases include:
  1. Centered lines that do not display average gap alignment characteristics are potential captions.
  2. If the length of a line is very different from other lines in the table, then it is very likely a caption.
  3. Lines with words like "Table" or "Figure" are likely captions.
  4. For a multi-line caption, continue until a period is encountered. If the period occurs in the middle of a line, continue until the next blank line is encountered.

All of the above heuristics have been implemented in TINTIN except the last set of additional heuristics that were not relevant to the the test database we used.

3.2 Indexing

We created our structured document database by field indexing the tagged output from the preprocessing module using the "inbuild" indexing program[1]. Field indexing here involves creating terms specific to the "caption" and "table" fields.

3.3 Retrieval

We used INQUERY[7, 1], a probabilistic retrieval engine, to retrieve tables from the structured document database. Since the table documents were field indexed on caption and table fields, the user has the flexibility to query based on these fields. Occurence of words in caption versus their occurence as a table entry could carry a different import to the user. For example, a query like "health insurance united states" could represent a variety of user requirements. It could mean that the user really wants to know about health insurance in general with some association to the United States or she could be wanting to know specifically about health insurance in the United States. In the former case "health insurance" should occur in the caption and get more weight whereas in the latter case the entire set of words in the query should get equal weight. Another example is a query like "Stock prices of oil companies". A table that has all the words in the caption field is likely to be more relevant than a table with "stock prices" in captions and "oil companies" as table entry or "oil companies" in captions and "stock prices" as table entry. A table with "stock" and "companies" in captions and "prices" and "oil" as table entries is highly likely to be irrelevant. Thus, weighting the query based on fields can give the user the flexibility and power to manipulate and express her needs better.

4 Experiments

4.1 The system

The table extraction process itself is subject to a version of precision versus recall trade-off. Precision (getting only the table lines with no extraneous information) may have to be sacrificed in order to improve recall (getting all tables in the text along with some non-table lines) or vice versa. The preprocessing module can be tuned by adjusting precision and recall parameters. If the parameters are highly constrained, precision improves but recall drops. We have experimented with two variations of the system: high precision and high recall versions. For example, with the precision-oriented version of the TINTIN system, many lines of Figure 4 like "Vanderbilt one-ounce eau de toilette spray", "Fruit of the Loom men's underwear(three-pack)" were not recognized as being a part of the table. The recall-oriented version of the TINTIN system extracts all these lines along with one extraneous line "While 52% of consumers surveyed in March by Leo J. Shapiro & Associates said Sears was the best place to-buy power tools, only 1% said it was the best place to buy a woman's dress blouse."[2].

We believe that it may be better to tune the system for higher recall. The intuition behind this is that it is highly

[1] Inbuild and INQUERY (used for retrieval) are a part of a suite of tools for IR systems developed at the Center for Intelligent Information Retrieval, University of Massachusetts, Amherst.

[2] Even though the figure shows it as three lines, due to the limitations of the word processor we use, in the actual table document this was just one long line.

```
<CAPTION> While 52% of consumers surveyed in March by Leo J. Shapiro &\
Associates said Sears was the best place to-buy power tools, only 1%\
said it was the best place to buy a woman's dress blouse.</CAPTION>
<CAPTION>                New Sears prices vs. the competition </CAPTION>
<CAPTION>       SEARS       PENNEY       WAL-MART       WARD       TARGET </CAPTION>
<TABLE>      Levi 501 jeans men </TABLE>
<TABLE>       $ 19.96     $ 19.99          N.A.       N.A.          N.A. </TABLE>
<CAPTION>      Vanderbilt one-ounce eau de toilette spray </CAPTION>
<TABLE>        15.96       15.50       $ 13.84     $15.50      $ 13.95 </TABLE>
<CAPTION>      Fruit of the Loom men's underwear (three-pack) </CAPTION>
<TABLE>         3.96        N.A.          3.96       N.A.          3.99 </TABLE>
<CAPTION>      Prestone antifreeze </CAPTION>
<TABLE>         7.62        N.A.          7.47       7.99          7.99 </TABLE>
<CAPTION>      Sealy mattress queen size </CAPTION>
<TABLE>       549.95      557.75          N.A.     549.99          N.A. </TABLE>
<CAPTION>      Lego basic building set </CAPTION>
<TABLE>         9.99        N.A.          9.57       9.99          9.99 </TABLE>
<CAPTION>      Etch a Sketch </CAPTION>
<TABLE>         8.99        N.A.          6.97       8.99          8.99 </TABLE>
```

Figure 4: An erroneously tagged table (by the earlier heuristics) that is later rectified by the Pattern Regularity Heuristic

```
<CAPTION>                        Market Share in Europe </CAPTION>
<CAPTION>                  (Figures are percentage for 1987) </CAPTION>
<CAPTION>              Central-office telephone-switching lines </CAPTION>
<TABLE>                Alcatel                                  40.0% </TABLE>
<TABLE>                Plessey/GEC                              16.7 </TABLE>
<TABLE>                Ericsson                                 13.6 </TABLE>
<TABLE>                Siemens                                  10.2 </TABLE>
<TABLE>                Italtel                                   8.4 </TABLE>
<TABLE>                Others                                   11.1 </TABLE>
<TABLE>                TOTAL SHIPMENTS:        9.6 million lines </TABLE>
<CAPTION>                              Semiconductors </CAPTION>
<TABLE>                Philips                                  14.6 </TABLE>
<TABLE>                SGS-Thomson                               8.6 </TABLE>
<TABLE>                Texas Instruments                         7.7 </TABLE>
<TABLE>                Motorola                                  7.5 </TABLE>
<TABLE>                Siemens                                   7.5 </TABLE>
<TABLE>                National Semiconductor                    5.4 </TABLE>
<TABLE>                Intel                                     4.5 </TABLE>
<TABLE>                NEC                                       3.9 </TABLE>
<TABLE>                ITT                                       3.8 </TABLE>
<TABLE>                Advanced Micro Devices                    3.7 </TABLE>
<TABLE>                Plessey                                   2.4 </TABLE>
<TABLE>                TOTAL MARKET:              $6.36 billion </TABLE>
<CAPTION>      Source: Dataquest Inc. </CAPTION>
```

Figure 5: A table with intermittent caption

```
<CAPTION>  The percentage change is since year-end.  </CAPTION>
<CAPTION>                                                    % This </CAPTION>
<CAPTION>                                 Nov 15     Nov 14    Year </CAPTION>
<TABLE>    U.S.                           248.1      247.5     + 8.1 </TABLE>
<TABLE>    Britain                        548.1      545.4     + 6.3 </TABLE>
<TABLE>    Canada                         350.7      349.6       0.0 </TABLE>
<TABLE>    Japan                         1350.8     1340.1    + 32.1 </TABLE>
<TABLE>    World index                    478.9      476.1     + 17.4 </TABLE>
```

Figure 6: Lines 2 and 3 are tagged captions by the Differential Column Count Heuristic

```
<CAPTION>                        12/31/89 LOANS  MARKET        INTEREST </CAPTION>
<CAPTION>  NAME                   (In billions)   SHARE  CARDS*    RATE </CAPTION>
<TABLE>    Citicorp                   $25.0       16.9%  V/MC     19.80 </TABLE>
<TABLE>    Chase Manhattan             9.0         6.1   V/MC     19.80 </TABLE>
<TABLE>    Sears                       8.5         5.8   D        19.80 </TABLE>
<TABLE>    First Chicago               6.6         4.5   V/MC     19.80 </TABLE>
<TABLE>    BankAmerica                 6.0         4.1   V/MC     19.80 </TABLE>
<TABLE>    MNC Financial               5.3         3.6   V/MC     19.80 </TABLE>
<TABLE>    American Express            5.2         3.5   O        16.25 </TABLE>
<TABLE>    Bank of New York            3.8         2.6   V/MC     16.98 </TABLE>
<TABLE>    Manufacturers Hanover       2.9         2.0   V/MC     19.80 </TABLE>
<TABLE>    Wells Fargo                 2.5         1.7   V/MC     19.80 </TABLE>
<TABLE>    Total for top 10          $74.8 </TABLE>
<CAPTION>  (or about 51% of industry vs. about 47% in 1987) </CAPTION>
<CAPTION>  *V=Visa; MC=MasterCard; D=Discover; O=Optima </CAPTION>
<CAPTION>  Sources: Individual issuers; RAM Research Bankcard </CAPTION>
```

Figure 7: Line 2 is tagged caption by the Differential Gap Structure Heuristic

improbable that a query term will match a randomly extracted word in an extraneous line. Hence, even if higher recall brings in extraneous lines, it may not effect the retrieval of the relevant tables. In addition, using proper weighting functions will eliminate the significance of randomly extracted sentence words versus relevant words.

4.2 Results of extraction and tagging

We ran the preprocessing module for table extraction and tagging on the 6 year (1987 – 1992) Wall Street Journal database. Table 1 shows the statistics for this process. The final structured document database was built using the collated set of 6509 tables.

Table 2 and Table 3 show the accuracy statistics for a set of 100 documents with 50 tables in them. The table extractor did not miss any table in its entirety. The errors were few and spread out. In the case of the component tagger, the errors were highly localized. Given the low error rate, this is a positive trait as this implies that most of the tables were correctly tagged for all their components.

4.3 Retrieval Results

We evaluated the TINTIN system using two types of queries - natural language query and structured query (with field retrieval on caption and table fields). Query Set I contained 25 natural language queries. The relevance judgement set for each query was obtained by looking at the top 100 tables retrieved for that query. Query Set II contained 25 structured queries derived by manually reformulating the natural language queries from Set I. Table 4 shows the precision statistics[3]. The precision measure is defined as follows:

$$Precision = \frac{Number\ of\ relevant\ tables\ retrieved}{Total\ tables\ retrieved}$$

It can be seen from Table 4 that the structured queries perform significantly better than the natural language queries. The average precision values improved by 20.9%. More importantly, it can be seen that the structured queries do even better in the top 20 tables. This strongly indicates that they are very good for interactive querying where the users are usually interested in the top 15 to 20 documents.

We would now like to discuss a few queries in detail to illustrate the additional searching power field retrieval operators give to the user.

1. Natural Language Query:

   `Stock prices of oil companies`

   Reformulated Query:

```
#wsum (6.0  (2.0 #field(CAPTION oil))
            (2.0 #field (CAPTION stock))
            (1.0 #field(CAPTION company))
            (1.0 #field(CAPTION prices)))
```

   The reformulated query should be interpreted as follows: `(2.0 CAPTION oil)` means that the occurrence of "oil" in the CAPTION field gets a weight of 2.0 on a scale of 6.0.

   The average precision value for the natural language query was 49.5% and for the structured query was 57.7%. The structured query performed 16.4% better than the natural language query. With the natural language query, many irrelevant tables related to oil matters other than stock prices and tables concerning stock prices of companies other than oil companies were retrieved. The structured query has the power to specify, that "oil" and "stocks" should get more weight if they occur in caption and that "company" and "price" are not as important.

2. Natural Language Query:

   `Export of computers`

   Reformulated Query:

```
#wsum  (5.0 (2.0 #field (CAPTION exports))
            (2.0 #field (CAPTION computers))
            (1.0 #field (TABLE computers)))
```

   The average precision value for the natural language query was 80.6% and for the structured query was 64.5%. The natural language query performed 20.1% better than the structured query. The natural language query performs better because the reformulated query gives more weight to computers. So anything related to computers such as computer parts, computer prices, and computer companies were all retrieved and there were a large number of computer related tables in the database.

   This query illustrates an important caveat about the structured queries. While it is true that they give better expressibility to the user it is also true that they are susceptible to bad reformulations leading to poor retrieval results. Also note that a structured query, at the very least, can be made to perform the same as the natural language query by giving equal weight to all the words occurring in all the fields.

   For example, the following structured query is the same as the natural language query from which it was derived:

```
#wsum (4.0 (1.0 #field (CAPTION exports))
           (1.0 #field (CAPTION computers))
           (1.0 #field (TABLE computers))
           (1.0 #field (TABLE exports)))
```

5 Conclusion

The central point of this paper is to highlight the utility of exploiting information other than the key words in a document to empower the users with more flexible and powerful query capabilities. More specifically, we exploit the structural information in a document to identify tables and their component fields and let the users query based on these fields. Our empirical results have demonstrated that heuristic method based table extraction and component tagging can be performed effectively and efficiently. Moreover, our experiments in text table retrieval using the TINTIN system have strongly indicated that such structural decomposition can facilitate better representation of user's information needs and hence more effective retrieval of tables.

Presently, the system distinguishes between caption and table entries of a table. Enhancing the system to make further distinctions like the caption, column and row headings and table entry components could lead to a more effective

[3] We did not look at the recall statistics because we did not have an exhaustive relevance judgement set.

Database	Size	Wall-Clock Time	Table db Size	Number of Tables
WSJ87	131MB	17 min	3.7MB	1857
WSJ88	109MB	15 min	3.4MB	1701
WSJ89	38MB	4 min	.3MB	222
WSJ90	73MB	10 min	2.4MB	930
WSJ91	145MB	20 min	4.3MB	1417
WSJ92	34MB	5 min	1.1MB	382
TOTAL	530MB	71 min	15 MB	6509

Table 1: Table Extraction & Component Tagging Statistics

Total number of lines in 100 docs	6205	–
Total number of table and caption lines	1005	–
Total number of lines extracted by table extractor	1041	–
Total number of table and caption lines missed	18	1.8 %
Total number of extraneous lines extracted	54	5.4 %

Table 2: Accuracy results for the Table Extractor

Actual number of captions	265	–
Number of captions mis-tagged as table lines	25	9.4%
Actual number of table lines	740	–
Number of table lines mis-tagged as captions	55	7.4 %

Table 3: Accuracy results for the Component Tagger

Precision (% change) – 25 queries		
Tables	Natural Language Query	Structured Query
5	72.0	84.0 (+16.7)
10	58.8	71.6 (+21.8)
15	48.0	62.4 (+30.0)
20	42.0	51.8 (+23.3)
30	33.3	39.1 (+17.4)
100	12.8	13.6 (+6.3)
200	6.4	6.8 (+6.3)
500	2.6	2.7 (+3.8)
1000	1.3	1.4 (+7.7)
Average	52.6	63.6 (+20.9)

Table 4: A comparison of precision values of natural language and structured queries

query formulation and retrieval. The weighting function can be modified in such a way that caption gets the highest weighting, column and row headings get the next highest weighting and table entries get the least weighting. There are also many other possibilities like slicing the table into columns and treating each column as a document. The column header and body content occurring together indicates more specificity and could be a source of multiple evidence for the corresponding table. For example, if the query is "China Exports Slippers" and we have a table with "China" and "slippers" occurring together in a column, this should get more weight than the case where "Romania" and "slippers" occur in one column and "China" occurs in another one. Here co-occurrence in a column indicates higher relevance.

Incorporating the kinds of enhancements discussed above into TINTIN remains high on our agenda for future work.

Acknowledgments

We would like to express our gratitude to James Allan and M V Nagendra Prasad for their insights and helpful comments. We would like to thank Vasantha Kumar, Stephen Harding and Sandhya Kasera for their help with the internals of INQUERY and also many helpful discussions. Our thanks go to Daniela Rus, James Callan and Fred Lenherr for their encouragement to work on these ideas. We would also like to thank Jay Ponte, Warren Greiff, Lisa Ballesteros, and Russell Swan for providing useful feedback on the draft versions of this paper.

This material is based on work supported in part by the National Science Foundation, Library of Congress and Department of Commerce under cooperative agreement number EEC-9209623. This material is based on work supported in part by NRaD Contract Number N66001-94-D-6054.

Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor(s).

References

[1] J. P. Callan, W. Bruce Croft, and S. M. Harding. The INQUERY retrieval system. In *Proceedings of the 3rd International Conference on Database and Expert Systems Applications*, pages 78–83, 1992.

[2] H Fujisawa, Y Nakano, and K Kurino. Segmentation methods for character recognition: From segmentation to document structure analysis. *Proceedings of the IEEE*, 80(7), 1992.

[3] G Nagy, S Seth, and M Vishwanathan. A prototype document image analysis system for technical journals. *Computer*, 25(7), 1992.

[4] Daniela Rus and James Allan. Does Navigation Require More than One Compass. In *AAAI 95 Fall Symposium on AI Applications in Knowledge Navigation and Retrieval*, pages 116–122, Cambridge, MA, November 1995.

[5] Daniela Rus and Devika Subramanian. Customizing Information Capture and Access. *ACM Transactions on Information Systems*, 1997. To appear.

[6] G Salton. The smart document retrieval project. In *Proceedings of the Fourteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 356–358, 1991.

[7] H. R. Turtle. *Inference Networks for Document Retrieval*. PhD thesis, University of Massachusetts, 1990.

[8] D. Wang and S. Srihari. Classification of newspaper image blocks using texture analysis. *Computer Vision, Graphics, and Image Processing*, 47, 1989.

USER COMMUNITIES

Chair: Paul Mosher