Learning to Recognize Tables in Free Text

Hwee Tou Ng
Chung Yong Lim
Jessica Li Teng Koo
DSO National Laboratories
20 Science Park Drive, Singapore 118230
{nhweetou,lchungyo,kliteng}@dso.org.sg

Abstract

Many real-world texts contain tables. In order to process these texts correctly and extract the information contained within the tables, it is important to identify the presence and structure of tables. In this paper, we present a new approach that learns to recognize tables in free text, including the boundary, rows and columns of tables. When tested on Wall Street Journal news documents, our learning approach outperforms a deterministic table recognition algorithm that identifies tables based on a fixed set of conditions. Our learning approach is also more flexible and easily adaptable to texts in different domains with different table characteristics.

1 Introduction

Tables are present in many real-world texts. Some information such as statistical data is best presented in tabular form. A check on the more than 100,000 Wall Street Journal (WSJ) documents collected in the ACL/DCI CD-ROM reveals that at least an estimated one in 30 documents contains tables.

Tables present a unique challenge to information extraction systems. At the very least, the presence of tables must be detected so that they can be skipped over. Otherwise, processing the lines that constitute tables as if they are normal "sentences" is at best misleading and at worst may lead to erroneous analysis of the text.

As tables contain important data and information, it is critical for an information extraction system to be able to extract the information embodied in tables. This can be accomplished only if the structure of a table, including its rows and columns, are identified.

That table recognition is an important step in information extraction has been recognized in (Appelt and Israel, 1997). Recently, there is also a greater realization within the computational linguistics community that the layout and types of information (such as tables) contained in a document are important considerations in text processing (see the call for participation (Power and Scott, 1999) for the 1999 AAAI Fall Symposium Series).

However, despite the omnipresence of tables and their importance, there is surprisingly very little work in computational linguistics on algorithms to recognize tables. The only research that we are aware of is the work of (Hurst and Douglas, 1997; Douglas and Hurst, 1996; Douglas et al., 1995). Their method is essentially a deterministic algorithm that relies on spaces and special punctuation symbols to identify the presence and structure of tables. However, tables are notoriously idiosyncratic. The main difficulty in table recognition is that there are so many different and varied ways in which tables can show up in real-world texts. Any deterministic algorithm based on a fixed set of conditions is bound to fail on tables with unforeseen layout and structure in some domains.

In contrast, we present a new approach in this paper that learns to recognize tables in free text. As our approach is adaptive and trainable, it is more flexible and easily adapted to texts in different domains with different table characteristics.

2 Task Definition

The input to our table recognition program consists of plain texts in ASCII characters. Examples of input texts are shown in Figure 1 to 3. They are document fragments that contain tables. Figure 1 and 2 are taken from the Wall Street Journal documents in the ACL/DCI CD-ROM, whereas Figure 3 is taken from the patent documents in the TIPSTER IR Text Research Collection Volume 3 CD-ROM.[1]

In Figure 1, we added horizontal 2-digit line numbers "Line nn:" and vertical single-digit line numbers "n" for ease of reference to any line in this document. We will use this document to illustrate the details of our learning approach throughout this paper. We refer to a horizontal line as *hline* and a vertical line as *vline* in the rest of this paper.

Each input text may contain zero, one or more tables. A table consists of one or more hlines. For example, in Figure 1, hlines 13–18 constitute a table. Each table is subdivided into columns and rows.

[1] The extracted document fragments appear in a slightly edited form in this paper due to space constraint.

```
        1234567890123456789012345678901234567890123456789012345678901234567890
Line 01: Raw-steel production by the nation's mills increased 4% last week to
Line 02: 1,633,000 tons from 1,570,000 tons the previous week, the American
Line 03: Iron and Steel Institute said.
Line 04:
Line 05: Last week's output fell 9.5% from the 1,804,000 tons produced a year
Line 06: earlier.
Line 07:
Line 08: The industry used 75.8% of its capability last week, compared with
Line 09: 71.9% the previous week and 72.3% a year earlier.
Line 10:
Line 11: The American Iron and Steel Institute reported:
Line 12:
Line 13:                                     Net tons  Capability
Line 14:                                     produced  utilization
Line 15:   Week to March 14 .............  1,633,000   75.8%
Line 16:   Week to March 7 ..............  1,570,000   71.9%
Line 17:   Year to date ................. 15,029,000   66.9%
Line 18:   Year earlier to date ......... 18,431,000   70.8%
Line 19:   The capability utilization rate is a calculation designed
Line 20:to indicate at what percent of its production capability the
Line 21:industry is operating in a given week.
```

Figure 1: Wall Street Journal document fragment

```
       How Some Highly Conditional 'Bids' Fared
               Stock's
   'Bid'*      Initial
   Date**      Reaction***       Outcome
Bidder (Target Company)
TWA/Carl Icahn (USAir Group)
   $52         +5 3/8 to 49 1/8  Bid, seen a ploy to get
   3/4/87                        USAir to buy TWA, is
                                 shelved Monday with
                                 USAir at 45 1/4; closed
                                 Wed. at 44 1/2
Columbia Ventures (Harnischfeger)
   $19         +1/2 to 18 1/4    Harnischfeger rejects
   2/23/87                       bid Feb. 26 with stock
                                 at 18 3/8; closed Wed.
                                 at 17 5/8
```

Figure 2: Wall Street Journal document fragment

Each column of a table consists of one or more vlines. For example, there are three columns in the table in Figure 1: vlines 4–23, 36–45, and 48–58. Each row of a table consists of one or more hlines. For example, there are five rows in the table in Figure 1: hlines 13–14, 15, 16, 17, and 18.

More specifically, the task of table recognition is to identify the boundaries, columns and rows of tables within an input text. For example, given the input text in Figure 1, our table recognition program will identify one table with the following boundary, columns and rows:

1. Boundary: hlines 13–18
2. Columns: vlines 4–23, 36–45, and 48–58
3. Rows: hlines 13–14, 15, 16, 17, and 18

Figure 1 to 3 illustrate some of the difficulties of table recognition. The table in Figure 1 uses a string of contiguous punctuation symbols "." instead of blank space characters in between two columns. In Figure 2, the rows of the table can contain caption or title information, like "How Some Highly Conditional 'Bids' Fared", or header information like "Stock's Initial Reaction***" and "Outcome", or

```
 side walls of the tray to provide even greater protection from convective
 heat transfer. Preferred construction materials are shown in Table 1:
          TABLE 1
--------------------------------------
Component
         Material
--------------------------------------
Stiffener
         Paperboard having a thickness of about 6 and 30
         mil (between about 6 and 30 point chip board).
Insulation
         Mineral wool, having a density of between 2.5
         and 6.0 pounds per cubic foot and a thickness of
         between 1/4 and 1 and 1/4 inch.
Plastic sheets
         Polyethylene, having a thickness of between 1
         and 4 mil; coated with a reflective finish on the
         exterior surfaces, such as aluminum having a
         thickness of between 90 and 110 Angstroms
         applied using a standard technique such as
         vacuum deposition.
--------------------------------------
The stiffener 96 makes a smaller contribution to the insulation properties
 of the blanket 92, than does the insulator 98. As stated above, the
```

Figure 3: Patent document fragment

body content information like "$52" and "+5 3/8 to 49 1/8". Each row containing body content information consists of several hlines — information on "Outcome" spans several hlines. In Figure 3, strings of contiguous dashes "-" occur within the table. Furthermore, the two columns within the table appear right next to each other — there are no blank vlines separating the two columns. Worse still, some words from the first column like "Insulation" and "Plastic sheets" spill over to the second column. Notice that there may or may not be any blank lines or delimiters that immediately precede or follow a table within an input text.

In this paper, we assume that our input texts are plain texts that do not contain any formatting codes, such as those found in an SGML or HTML document. There is a large number of documents that fall under the plain text category, and these are the kinds of texts that our approach to table recognition handles. The work of (Hurst and Douglas, 1997; Douglas and Hurst, 1996; Douglas et al., 1995) also deals with plain texts.

3 Approach

A table appearing in plain text is essentially a two dimensional entity. Typically, the author of the text uses the <newline> character to separate adjacent hlines and a row is formed from one or more of such hlines. Similarly, blank space characters or some special punctuation characters are used to delimit the columns.[2] However, the specifics of how exactly this is done can vary widely across texts, as exemplified by the tables in Figure 1 to 3.

Instead of resorting to an ad-hoc method to recognize tables, we present a new approach in this paper that learns to recognize tables in plain text. Our learning method uses purely surface features like the proportion of the kinds of characters and their relative locations in a line and across lines to recognize tables. It is domain independent and does not rely on any domain-specific knowledge. We want to investigate how high an accuracy we can achieve based purely on such surface characteristics.

The problem of table recognition is broken down into 3 subproblems: recognizing table boundary, column, and row, in that order. Our learning approach treats each subproblem as a separate classification problem and relies on sample training texts in which the table boundaries, columns, and rows have been correctly identified. We built a graphical user interface in which such markup by human annotators can be readily done. With our X-window based GUI, a typical table can be annotated with its boundary, column, and row demarcation within a minute.

From these sample annotated texts, training ex-

[2] We assume that any <tab> character has been replaced by the appropriate number of blank space characters in the input text.

amples in the form of feature-value vectors with correctly assigned classes are generated. One set of training examples is generated for each subproblem of recognizing table boundary, column, and row. Machine learning algorithms are used to build classifiers from the training examples, one classifier per subproblem. After training is completed, the table recognition program will use the learned classifiers to recognize tables in new, previously unseen input texts.

We now describe in detail the feature extraction process, the learning algorithms, and how tables in new texts are recognized. The following classes of characters are referred to throughout the rest of this section:

- Space character: the character " " (i.e., the character obtained by typing the space bar on the keyboard).
- Alphanumeric character: one of the following characters: "A" to "Z", "a" to "z", and "0" to "9".
- Special character: any character that is not a space character and not an alphanumeric character.
- Separator character: one of the following characters: ".", "*", and "-".

3.1 Feature Extraction

3.1.1 Boundary

Every hline in an input text generates one training example for the subproblem of table boundary recognition. Every hline H within (outside) a table generates a positive (negative) example. Each training example consists of a set of 27 feature values. The first nine feature values are derived from the immediately preceding hline H_{-1}, the second nine from the current hline H_0, and the last nine from the immediately following H_1.[3]

For a given hline H, its nine features and their associated values are given in Table 1.

To illustrate, the feature values of the training example generated by line 16 in Figure 1 are:

$$f, 3, N, \%, N, 4, 3, 1, 1,$$

$$f, 3, N, \%, N, 4, 3, 1, 1,$$
$$f, 3, N, \%, N, 3, 3, 1, 1$$

Line 16 generated the feature values $f, 3, N, \%, N, 4, 3, 1, 1$. Since line 16 does not consist of only space characters, the value of $F1$ is f. There are three space characters before the word "Week" in line 16, so the value of $F2$ is 3. Since the first non-space character in line 16 is "W" and it is not one of the listed special characters, the value of $F3$ is "N". The last non-space character in line 16 is "%", which becomes the value of $F4$. Since line 16 does not consist of only special characters, the value of $F5$ is "N". There are four segments in line 16 such that each segment consists of two or more contiguous space characters: a segment of three contiguous space characters before the word "Week"; a segment of two contiguous space characters after the punctuation characters "..." and before the number "1,570,000"; a segment of three contiguous space characters between the two numbers "1,570,000" and "71.9%"; and the last segment of contiguous space characters trailing the number "71.9%". The values of the remaining features of line 16 are similarly determined. Finally, line 15 and l7 generated the feature values $f, 3, N, \%, N, 4, 3, 1, 1$ and $f, 3, N, \%, N, 3, 3, 1, 1$, respectively.

The features attempt to capture some recurring characteristics of lines that constitute tables. Lines with only space characters or special characters tend to delimit tables or are part of tables. Lines within a table tend to begin with some number of leading space characters. Since columns within a table are separated by contiguous space characters or special characters, we use segments of such contiguous characters as features indicative of the presence of tables.

3.1.2 Column

Every vline within a table generates one training example for the subproblem of table column recognition. Each vline can belong to exactly one of five classes:

1. Outside any column
2. First line of a column
3. Within a column (but neither the first nor last line)
4. Last line of a column
5. First and last line of a column (i.e., the column consists of only one line)

Note that it is possible for one column to immediately follow another (as is the case in Figure 3). Thus a two-class representation is not adequate here, since there would be no way to distinguish between two adjoining columns versus one contiguous column using only two classes.[4]

The start and end of a column in a table is typically characterized by a transition from a vline of

[3] For the purpose of generating the feature values for the first and last hline in a text, we assume that the text is padded with a line of blank space characters before the first line and after the last line.

[4] For the identification of table boundary, we assume in this paper that there is some hline separating any two tables, and so a two-class representation for table boundary suffices.

Feature	Description
$F1$	Whether H consists of only space characters. Possible values are t (if H is a blank line) or f (otherwise).
$F2$	The number of leading (or initial) space characters in H.
$F3$	The first non-space character in H. Possible values are one of the following special characters: ()[]{}<> +−*/=~!@#$%∧& or N (if the first non-space character is not one of the above special characters).
$F4$	The last non-space character in H. Possible values are the same as $F3$.
$F5$	Whether H consists entirely of one special character only. Possible values are either one of the special characters listed in $F3$ (if H only consists of that special character) or N (if H does not consist of one special character only).
$F6$	The number of segments in H with two or more contiguous space characters.
$F7$	The number of segments in H with three or more contiguous space characters.
$F8$	The number of segments in H with two or more contiguous separator characters.
$F9$	The number of segments in H with three or more contiguous separator characters.

Table 1: Feature values for table boundary

space (or special) characters to a vline with mixed alphanumeric and space characters. That is, the transition of character types across adjacent vlines gives an indication of the demarcation of table columns. Thus, we use character type transition as the features to identify table columns.

Each training example consists of a set of six feature values. The first three feature values are derived from comparing the immediately preceding vline V_{-1} and the current vline V_0, while the last three feature values are derived from comparing V_0 with the immediately following vline V_1.[5]

Let V_j and V_{j+1} be any two adjacent vlines. Suppose $V_j = c_{1,j} \dots c_{i,j} \dots c_{m,j}$, and $V_{j+1} = c_{1,j+1} \dots c_{i,j+1} \dots c_{m,j+1}$ where m is the number of hlines that constitute a table.

Then the three feature values that are derived from the two vlines V_j and V_{j+1} are determined by counting the proportion of two horizontally adjacent characters $c_{i,j}$ and $c_{i,j+1}$ $(1 \leq i \leq m)$ that satisfy some condition on the type of the two characters. The precise conditions on the three features are given in Table 2.

To illustrate, the feature values of vline 4 in Figure 1 are:

$$0.333, 0, 0.667, 0.333, 0, 0$$

and its class is 2 (first line of a column). In deriving the feature values, only hlines 13–18, the lines that constitute the table, are considered (i.e., $m = 6$). For the first three feature values, $F1 = 2/6$ since there are two space-character-to-space-character transitions from vline 3 to 4 (namely, on hlines 13 and 14); $F2 = 0$ since there is no alphanumeric character or special character in vline 3; $F3 = 4/6$, since there are four space-character-to-alphanumeric-character transitions from vline 3 to 4 (namely, on hlines 15–18). Similarly, the last 3 feature values are derived by examining the character transitions from vline 4 to 5.

[5] For the purpose of generating the feature values for the first and last vline in a table, we assume that the table is padded with a vline of blank space characters before the first vline and after the last vline.

3.1.3 Row

Every hline within a table generates one training example for the subproblem of table row recognition. Unlike table columns, every hline within a table belongs to some row in our formulation of the row recognition problem. As such, each hline belongs to exactly one of two classes:

1. First hline of a row
2. Subsequent hline of a row (not the first line)

The layout of a typical table is such that its rows tend to record repetitive or similar data or information. We use this clue in designing the features for table row recognition. Since the information within a row may span multiple hlines, as the "Outcome" information in Figure 2 illustrates, we use the first hline of a row as the basis for comparison across rows. If two hlines are similar, then they belong to two separate rows; otherwise, they belong to the same row. Similarity is measured by character type transitions, as in the case of table column recognition.

More specifically, to generate a training example for a hline H, we compare H with H', where H' is the first hline of the immediately preceding row if H is the first hline of the current row, and H' is the first hline of the current row if H is not the first hline of the current row.[6]

Each training example consists of a set of four feature values $F1, \dots, F4$. $F1$, $F2$, and $F3$ are determined by comparing H and H' while $F4$ is determined solely from H. Let $H = c_{i,1} \dots c_{i,j} \dots c_{i,n}$

[6] $H' = H$ for the very first hline within a table.

Feature	Description
$F1$	$c_{i,j}$ is a space character and $c_{i,j+1}$ is a space character; or $c_{i,j}$ is a special character and $c_{i,j+1}$ is a special character
$F2$	$c_{i,j}$ is an alphanumeric character or a special character, and $c_{i,j+1}$ is a space character
$F3$	$c_{i,j}$ is a space character, and $c_{i,j+1}$ is an alphanumeric character or a special character

Table 2: Feature values for table column

and $H' = c_{i',1} \ldots c_{i',j} \ldots c_{i',n}$, where n is the number of vlines of the table. The values of $F1, \ldots, F3$ are determined by counting the proportion of the pairs of characters $c_{i',j}$ and $c_{i,j}$ $(1 \leq j \leq n)$ that satisfy some condition on the type of the two characters, as listed in Table 3. Let $c_{i,k}$ be the first non-space character in H. Then the value of $F4$ is k/n.

To illustrate, the feature values of hline 16 in Figure 1 are:

$$0.236, 0.018, 0.018, 0.018$$

and its class is 1 (first line of a row). There are 55 vlines in the table, so $n = 55$.[7] Since hline 16 is the first line of a row, it is compared with hline 15, the first hline of the immediately preceding row, to generate $F1$, $F2$, and $F3$. $F1 = 13/55$ since there are 13 space-character-to-space-character transitions from hline 15 to 16. $F2 = F3 = 1/55$ since there is only one alphanumeric-character-to-space-character transition ("4" to space character in vline 19) and one space-character-to-special-character transition (space character to "." in vline 20). The first non-space character is "W" in the first vline within the table, so $k = 1$.

3.2 Learning Algorithms

We used the C4.5 decision tree induction algorithm (Quinlan, 1993) and the backpropagation algorithm for artificial neural nets (Rumelhart et al., 1986) as the learning algorithms to generate the classifiers. Both algorithms are representative state-of-the-art learning algorithms for symbolic and connectionist learning.

We used all the default learning parameters in the C4.5 package. For backpropagation, the learning parameters are: hidden units = 2, epochs = 1000, learning rate = 0.35 and momentum term = 0.5. We also used log n-bit encoding for the symbolic features and normalized the numeric features to $[0 \ldots 1]$ for backpropagation.

3.3 Recognizing Tables in New Texts

3.3.1 Boundary

Every hline generates a test example and a classifier assigns the example as either positive (within a table) or negative (outside a table).

3.3.2 Column

After the table boundary has been identified, classification proceeds from the first (leftmost) vline to the last (rightmost) vline in a table. For each vline, a classifier will return one of five classes for the test example generated from the current vline.

Sometimes, the class assigned by a classifier to the current vline may not be logically consistent with the classes assigned up to that point. For instance, it is not logically consistent if the previous vline is of class 1 (outside any column) and the current vline is assigned class 4 (last line of a column). When this happens, for the backpropagation algorithm, the class which is logically consistent and has the highest score is assigned to the current vline; for C4.5, one of the logically consistent classes is randomly chosen.

3.3.3 Row

The first hline of a table always starts a new active row (class 1). Thereafter, for a given hline, it is compared with the first hline of the current active row. If the classifier returns class 1 (first hline of a row), then a new active row is started and the current hline is the first hline of this new row. If the classifier returns class 2 (subsequent hline of a row), then the current active row grows to include the current hline.

4 Evaluation

To determine how well our learning approach performs on the task of table recognition, we selected 100 Wall Street Journal (WSJ) news documents from the ACL/DCI CD-ROM. After removing the SGML markups on the original documents, we manually annotated the plain-text documents with table boundary, column, and row information. The documents shown in Figure 1 and 2 are part of the 100 documents used for evaluation.

4.1 Accuracy Definition

To measure the accuracy of recognizing table boundary of a new text, we compare the class assigned by the human annotator to the class assigned by our table recognition program on every hline of the text. Let A be the number of hlines identified by the human annotator as being part of some table. Let B

[7] In generating the feature values for table row recognition, only the vlines enclosed within the identified first and last column of the table are considered.

Feature	Description
$F1$	$c_{i',j}$ is a space character and $c_{i,j}$ is a space character
$F2$	$c_{i',j}$ is an alphanumeric character or a special character, and $c_{i,j}$ is a space character
$F3$	$c_{i',j}$ is a space character, and $c_{i,j}$ is an alphanumeric character or a special character
$F4$	k/n

Table 3: Feature values for table row

be the number of hlines identified by the program as being part of some table. Let C be the number of hlines identified by both the human annotator and the program as being part of some table. Then recall $R = C/A$ and precision $P = C/B$. The accuracy of table boundary recognition is defined as the F measure, where $F = 2RP/(R+P)$. The accuracy of recognizing table column (row) is defined similarly, by comparing the class assigned by the human annotator and the program to every vline (hline) within a table.

4.2 Deterministic Algorithms

To determine how well our learning approach performs, we also implemented deterministic algorithms for recognizing table boundary, column, and row. The intent is to compare the accuracy achieved by our learning approach to that of the baseline deterministic algorithms. These deterministic algorithms are described below.

4.2.1 Boundary

A hline is considered part of a table if at least one character of hline is not a space character and if any of the following conditions is met:

- The ratio of the position of the first non-space character in hline to the length of hline exceeds some pre-determined threshold (0.25)
- Hline consists entirely of one special character.
- Hline contains three or more segments, each consisting of two or more contiguous space characters.
- Hline contains two or more segments, each consisting of two or more contiguous separator characters.

4.2.2 Column

All vlines within a table that consist of entirely space characters are considered not part of any column. The remaining vlines within the table are then grouped together to form the columns.

4.2.3 Row

The deterministic algorithm to recognize table row is similar to the recognition algorithm of the learning approach given in Section 3.3.3, except that the classifier is replaced by one that computes the proportion of character type transitions. All characters in the two hlines under consideration are grouped into four types: space characters, special characters, alphabetic characters, or digits. If the proportion of characters that change type exceeds some predetermined threshold (0.5), then the two hlines belong to the same row.

4.3 Results

We evaluated the accuracy of our learning approach on each subproblem of table boundary, column, and row recognition. For each subproblem, we conducted ten random trials and then averaged the accuracy over the ten trials. In each random trial, 20% of the texts are randomly chosen to serve as the texts for testing, and the remaining 80% texts are used for training. We plot the learning curve as each classifier is given increasing number of training texts. Figure 4 to 6 summarize the average accuracy over ten random trials for each subproblem. Besides the accuracy for the C4.5 and backpropagation classifiers, we also show the accuracy of the deterministic algorithms.

The results indicate that our learning approach outperforms the deterministic algorithms for all subproblems. The accuracy of the deterministic algorithms is about 70%, whereas the maximum accuracy achieved by the learning approach ranges over 85% – 95%. No one learning algorithm clearly outperforms the other, with C4.5 giving higher accuracy on recognizing table boundary and column, and backpropagation performing better at recognizing table row.

To test the generality of our learning approach, we also evaluated it on 50 technical patent documents from the TIPSTER Volume 3 CD-ROM. To test how well a classifier that is trained on one domain of texts will generalize to work on a different domain, we also tested the accuracy of our learning approach on patent texts after training on WSJ texts only, and vice versa. Space constraint does not permit us to present the detailed empirical results in this paper, but suffice to say that we found that our learning approach is able to generalize well to work on different domains of texts.

5 Future Work

Currently, our table row recognition does not distinguish among the different types of rows, such as title (or caption) row, header row, and content row. We would like to extend our method to make such

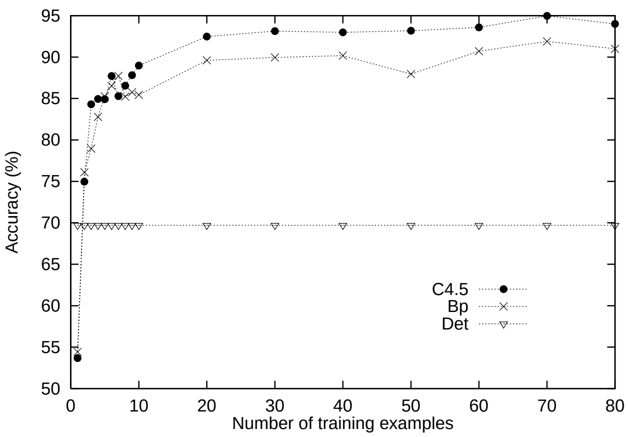

Figure 4: Learning curve of boundary identification accuracy on WSJ texts

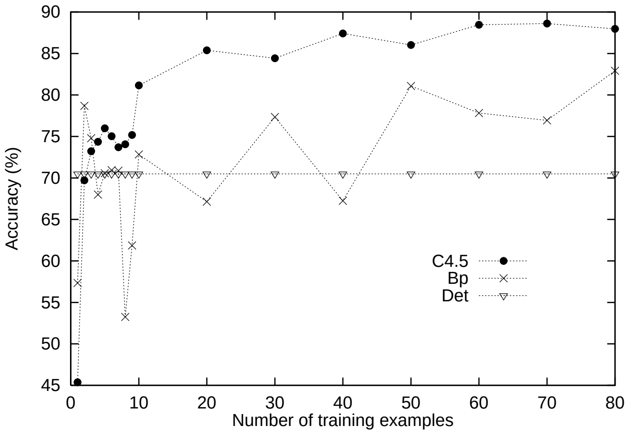

Figure 5: Learning curve of column identification accuracy on WSJ texts

distinction. We would also like to investigate the effectiveness of other learning algorithms, such as exemplar-based methods, on the task of table recognition.

6 Conclusion

In this paper, we present a new approach that learns to recognize tables in free text, including the boundary, rows and columns of tables. When tested on Wall Street Journal news documents, our learning approach outperforms a deterministic table recognition algorithm that identifies tables based on a fixed set of conditions. Our learning approach is also more flexible and easily adaptable to texts in different domains with different table characteristics.

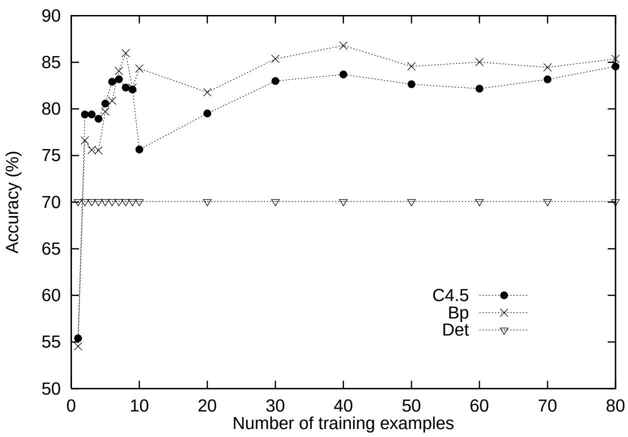

Figure 6: Learning curve of row identification accuracy on WSJ texts

References

Douglas Appelt and David Israel. 1997. Tutorial notes on building information extraction systems. Tutorial held at the Fifth Conference on Applied Natural Language Processing.

Shona Douglas and Matthew Hurst. 1996. Layout & language: Lists and tables in technical documents. In *Proceedings of the ACL SIGPARSE Workshop on Punctuation in Computational Linguistics*, pages 19–24.

Shona Douglas, Matthew Hurst, and David Quinn. 1995. Using natural language processing for identifying and interpreting tables in plain text. In *Fourth Annual Symposium on Document Analysis and Information Retrieval*, pages 535–545.

Matthew Hurst and Shona Douglas. 1997. Layout & language: Preliminary experiments in assigning logical structure to table cells. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 217–220.

Richard Power and Donia Scott. 1999. Using layout for the generation, understanding or retrieval of documents. Call for participation at the 1999 AAAI Fall Symposium Series.

John Ross Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco, CA.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning internal representation by error propagation. In David E. Rumelhart and James L. McClelland, editors, *Parallel Distributed Processing, Volume 1*, pages 318–362. MIT Press, Cambridge, MA.