Automatic selection of table areas in documents for information extraction

Ana Costa e Silva[1], Alípio Jorge[2], Luís Torgo[2]

[1]Banco de Portugal [*], Portugal, acsilva@bportugal.pt
[2]Universidade do Porto, Faculdade de Economia do Porto, LIACC, Portugal,
amjorge@liacc.up.pt, ltorgo@liacc.up.pt

Abstract: information contained in companies' financial statements is valuable for decision making at various levels. Much of the relevant information in such documents is contained in tables and is currently extracted mainly by hand. We propose a method that accomplishes a preliminary step of the task of automatically extracting information from tables in documents: selecting the lines of the document which are likely to belong to the tables containing the information to be extracted. Our method has been developed by empirically analyzing a set of Portuguese companies' financial statements, using statistical and data mining techniques. Empirical evaluation indicates that more than 99% of the table lines are selected after discarding at least 50% of them. The method can cope with the complexity of styles used in assembling information on paper and adapt its performance accordingly, thus maximizing its results.

1 Introduction

Companies worldwide have the legal obligation to produce and publish reports called financial statements accounting for their activities on at least a yearly basis. These reports, which are often more than 100 pages long, contain information that support the decisions of a variety of economic agents - current or potential employees, public administration, suppliers, clients, money lenders, investors, and economic policy developers. For many of these agents, analysis of such information is necessary not only at an individual level, by looking at each document separately, but also at an aggregated level, where the information from many documents is combined. However, aggregate analysis currently requires the time consuming activity of capturing the data manually to a database. If the task of extracting the relevant data from the documents could be automated, the quality of the analysis that supports economic agents' decisions would increase, and would be more timely, effective and less expensive.

Extracting relevant information from financial statements involves taking as input a potentially large document and gradually reducing it so as to have as output precisely the information items we are interested in, which are used to fill in a template. Much of the relevant information in a financial statement is in tables.

[*] The views expressed in this article are the responsibility of its authors and do not necessarily coincide with the Banco de Portugal's.

In this paper, we begin by identifying the generic components of the method for extracting information from tables in a document and propose a method that, given a text document, identifies the lines that potentially belong to tables. The documents we used in this work are plain text files obtained directly from financial statements in PDF format, made publicly available on the Web by major Portuguese companies.

In Section 2 we outline the tasks of the process of information extraction from tables. In Section 3 we briefly describe the financial statement items we consider to be the target of our extraction effort. In Section 4 we describe the analysis performed in order to determine robust criteria for selecting the relevant lines in a document. In Section 5 we present the main algorithm of the method. The method is evaluated in Section 6.

2 Tasks involved in information extraction from tables

Information extraction from tables embedded in a plain text document (untagged or poorly tagged ASCII files), is a process that can be divided in five tasks [1]:

Location: "Locating a table on a page involves differentiating tables from other elements such as body text, heading, titles, bibliographies, line drawings and bitmap graphics", [7]. The output is still a simple file, but all the elements which are not considered as part of a table have been extracted or separated. We have then a *graphical model* of a table.

Segmentation: delimiting the table's physical structures - its columns and lines, its simple and its spanning cells (cells that spread over more than one column or line). The output is a *physical model* of the table. The table's title(s) and footnotes can also be considered a part of its physical model.

Functional analysis: after having a physical model of the table, it is important to understand the function each part plays: there are basically two types of cells – the data cells, which are the purpose of the existence of the table; and the descriptor cells, which describe the data being presented. This is also the phase in which any titles and footnotes should be identified as such. The output of this phase is a *functional model* of the table.

Structural analysis: detecting the relationships between the cells, identifying the cells which have to be read conjointly, and the attribute – value threads which spread along the vertical and horizontal axis of the table. The output of this phase is a *structural model* of the table.

Interpretation: this is the ultimate goal of table analysis and it means going one step beyond simply identifying relationships to knowing what they mean and deciding upon them. More than knowing the cells containing strings "Total assets" - "5000" - "Real" have to be read conjointly, interpreting means being able to affirm "Total assets are 5000 PTE". The output of this stage is a *semantic model* of the table.

The task we approach in this paper is a preliminary step of this process: given a text document, automatically select the lines that are likely to belong to tables. This step reduces the search effort in the subsequent step of locating whole tables.

3 The information to be extracted

The minimal content of financial statements is determined by law: they should include, among other things,

- a report of the company's activity in the period, which is freely stylized and mostly composed of text and some charts and small tables;
- and accounting information, most of which presented in tables, each often taking a whole page.

The relevant information to be extracted is located mostly in tables. Southern European legal systems suggest models for these tables, the items they should include, and their order, which companies then adapt to their specificities.

Typically, one can expect each Portuguese financial report to hold 13 tables containing the target information. These tables can have a varying number of columns (typically 2 to 7) and lines (4 to a full page); can incorporate several empty lines within the table; many are mostly composed of numeric data cells, but some include mainly text; some have several lines of heading. Financial reports are often used for testing by researchers in this field, because "while standards exist for financial reports, the layout is substantially more varied and harder to detect with current algorithms", [7]. The financial reports also contain other tables regarded as not relevant for this work.

Although it is expected that, in the future, financial statements will be published in a predefined structured format such as XBRL (eXtensible Business Reporting Language, [10]), in many countries it will most likely take a long time before those standards are fully adopted.

To serve as training examples in our definition of the methods to follow, we have downloaded from the Web 19 financial reports published by 13 Portuguese companies in a period ranging from 1997 to 2000. These varied considerably in terms of size, (12 to 196 pages); on the whole there were 87,843 lines, of which 70,584 are not table lines and 9,685 are part of the tables containing the information we wish to extract. We then proceeded to convert the original files (mostly in PDF [5]) to plain text files using the pdftotxt linux utility, and import them to a database table. Each line in the report became a distinct record.

4 Selecting target areas

Current software to locate tables in documents is generally based on detecting the alignment of certain characters, either words, e.g. [3], non-space characters, e.g. [7], or white spaces, e.g. [6], being vertical alignment often the key feature searched for. Other methods, such as [4], examine the contents of each line in the document, character by character, to detect the presence of groups of contiguous spaces, count the number of such of groups, among other characteristics, and present it to a decision tree to determine whether or not the line belongs to a table.

However, in order to classify an area as table, existing strategies take each line (either vertical or horizontal) and perform a character-by-character scan to detect the presence of the characteristics used as input. As such, all parts of the document are treated as having equal likelihood of containing tables.

4.1 The main criterion for table line location

To classify each line as being likely to belong to a table or not, our method basically relies on one feature, which can easily be measured – the total number of contiguous inner spaces in the line. To determine this, we remove the leading and trailing spaces; we then obtain all the substrings of more than one consecutive space characters; the sum of the lengths of those substrings is the total number of contiguous inner spaces. The effectiveness of this feature relies on the notion that a line having more inner spaces than what is usual in its source document will hold a distinctive graphical element (such as a table or a chart), which is worth inspecting more closely. To illustrate this empirical observation, we show in Figure 1 the stacked distributions of contiguous inner spaces per table and non-table line in our training examples.

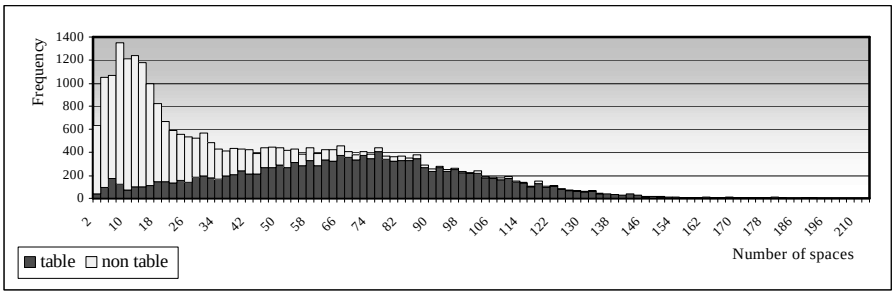

Figure 1: Distribution of the number of contiguous inner spaces per line in a sample of 19 documents of various layouts (zero is not shown). Each bar is the stacking of the frequencies obtained from table and non table lines

As we can observe, the two distributions are rather different: the lines of tables tend to have considerably more spaces than the lines without tables; and the spaces outside tables are less disperse than their counterparts; in terms of symmetry, table lines show a rather symmetric distribution while the others are positively skewed. This suggests that this measure is relevant in distinguishing these two types of lines. With significance 0,0% and on the basis of this sample, a Chi-square test proved the statistical dependence of the number of inner spaces in a line and whether the lines belong to a table [1]. Thus, our main criterion will be of the form

$$E_\ell > \lambda \rightarrow \ell \text{ is a } \textit{candidate table line} \quad (1)$$

where E_ℓ is the number of contiguous inner spaces in line ℓ, and λ is a threshold to be determined.

4.2 Robustness of the criterion to different document layouts

Having this established, we have to find an appropriate threshold that distinguishes the two groups. Should such a threshold depend on the document under analysis? In that case we would like to identify features of source documents that help determine the best threshold.

Figure 2 shows the distributions of inner spaces in two documents with different layout: the one on the left derives from a single column document, while the one on

the right is a two-column document. Intuitively, one can foresee that the limit to distinguish table from non-table lines can be much smaller in a text written in one single column than in a text entirely written in two or more columns.

The Kruskal Wallis test [1] added evidence to this empirical observation by proving with 0,0% significance that the distribution of inner spaces is statistically different for the reports in the sample. To determine whether a given pair of reports had contributed to this result, we compared each pair and found that only 22,8% of the total number of possible pairs were similar. A Chi square test proved there was statistical evidence of independence between the likelihood of a line being in a table and the document it originates from. As such, the limit of inner spaces to distinguish the two classes of lines should be established differently according to the source document.

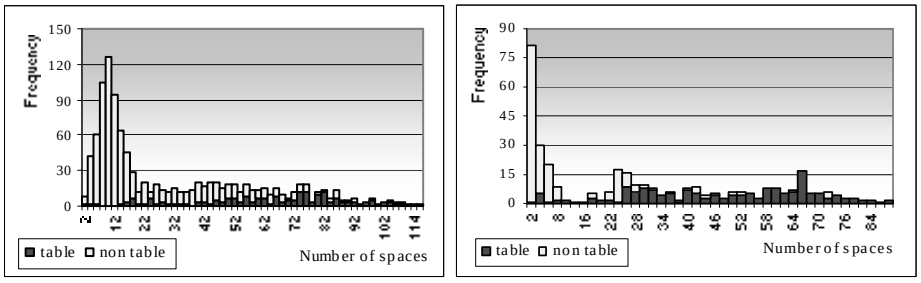

Figure 2: Distribution of the number of contiguous inner spaces per line in two documents following different styles: on the right, originating from a document written in two columns, the distribution of spaces in non-table lines is located farther from zero than that of its counterpart, which derives from a document written in one column

If we label each line of a report as being in a table or not and count the number of inner spaces within the line, we can determine the optimal limit for the document by using the J48 algorithm of the data mining suite Weka [9] to build a one-node decision tree that classifies each line as table or not given the number of contiguous inner spaces. The decision tree thus generated will classify an unlabelled line according to the logical value of a test of the form "Number of contiguous inner spaces < Threshold". The value of that threshold corresponds to the most informative test and is considered by us as the optimal threshold.

For the distributions shown in Figure 2, the optimal threshold found for the document on the left was 48, meaning that a line with more than 48 inner contiguous spaces should be investigated, whilst for the document on the right a limit of 8 spaces was set.

We now want to be able to determine how the optimal threshold for a document could be obtained given only its global features, without prior knowledge of which of its lines belong to tables. In other words, we want the criterion to be generalized to

$$E_\ell > \lambda(D) \rightarrow \ell \text{ is a } \textit{candidate table line} \quad (2)$$

where D is the document being analyzed. In Figure 3 we compare the thresholds established through the process described above with certain global features of their source reports. In the leftmost chart the global feature is *the number of lines in the*

document. As can be observed, documents with more lines tend to have higher thresholds. This happens because their authors probably invested more time and resources to make them more graphically appealing, by using a more diversified style for accommodating graphical elements on a page. The central chart shows that, for documents where the *most frequent number of positive inner spaces* is high enough (but not too high), the optimal division should be increasingly higher. This happens because the statistical mode occurs farther from zero when the text is organized in two or more columns and thus too small a limit would cause the misclassification of a considerable number of text lines. Finally, in the rightmost chart one can see a negative correlation between the optimal threshold and the *proportion of lines with zero contiguous inner spaces*, because this proportion will be higher in graphically simpler single column documents, where a low threshold leads to practically no errors.

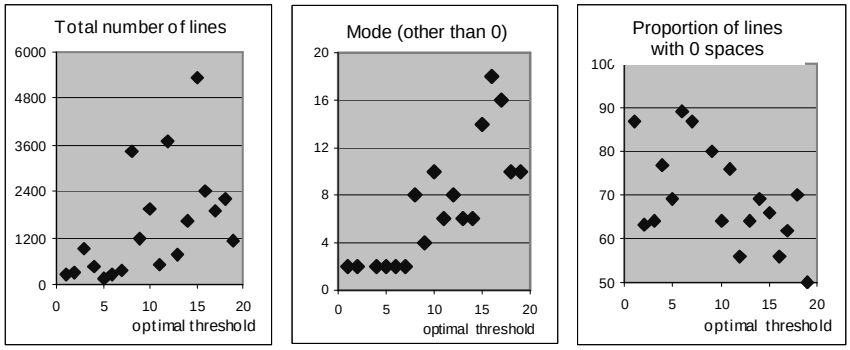

Figure 3: The ideal division versus characteristics of source documents: total number of lines (leftmost), mode (other than zero) (center) and weight of lines with no contiguous spaces over the total number of lines in the document (rightmost). At the center, one observation (4; 62) was excluded for it would reduce the readability of the chart; this document has a high weight of table lines and practically no internal spaces in non-table lines, thus the mode (other than zero) occurs in the area of lines of tables and the optimal limit is rather low

As such, it is possible to use measurable features of the source document to estimate the best threshold for deciding whether a line is likely to belong to a table or not. These features reflect the different styles used by the authors of the documents in the organization of the graphical elements on the page. All lines above the limit thus calculated are marked as candidate table lines. By establishing the limit according to the document features, the method can cope with different graphical styles. Alternative options for defining $\lambda(D)$ are given in Section 5.1.

However, one extra characteristic has to be taken into account when deciding whether a line belongs to a table: its neighbourhood. In fact, no matter how many inner spaces a line has, it is not in a table if other lines around it are plain text; and a line with no inner spaces is not plain text if it is surrounded by table lines. Ng et al. [4] found an interesting way of incorporating this aspect into their method: when deciding about each line, they took into account characteristics of the line before it and after it. That is, on deciding about an observation, characteristics of the observations which fall within a fixed 1-line band around it were used.

Our method, rather than considering a band of invariable size, determines the size of the band according to the characteristics of the page the line belongs to. This is done because the information we want to extract is more likely to be in pages which have a high concentration of table lines, being either one big table or several small tables in one page. In fact, a positive correlation, statistically significant at 1%, was found in our sample between the *proportion of table lines* in a page and a variable taking value 1 when the page contains a table with relevant information. Since we do not know in advance which lines belong to tables, we estimate that proportion by counting the lines given by criterion (2), the *candidate table lines*.

5 The algorithm

Our algorithm processes a given document page by page and within each page line by line. For each page *p* with more than a given minimum of *m* candidate table lines, we define *Windows* of lines where we look for table lines. If the proportion of candidate lines on the page, $Weight_p$, is high, all lines between the first and the last candidates, c_a and c_z respectively, are included in the window. Otherwise, we may have more than one window per page, each corresponding to a sequence of candidate lines no more than *k* lines apart. All windows include *i* lines above the first candidate line c_a and *f* lines below the last candidate line c_z.

The distance between any two lines is measured in terms of the total number of non-empty lines between them. This makes the method robust to the existence of long sequences of empty lines within tables, which can be quite common in this context.

Each line in a window will be classified as a *target area* if it has any contiguous inner spaces or if its total string length is smaller than *x*% of the maximum length of the candidate lines on the page. The procedure is iterated until there are no more candidate lines on the page and no more pages with enough candidate lines in the whole document. The values of *k*, *i* and *f*, and consequently the size of the windows, are determined according to the proportion of candidate lines on the page. We now give the pseudo-code of the algorithm.

Symbols used:

L : the lines ℓ in a document

Ind_ℓ : absolute index given to each non-empty line which identifies the distance between it and the beginning of the document in terms of the number of the total non-empty lines

E_ℓ : # of contiguous inner spaces in line ℓ

Len_ℓ : # of characters in line ℓ which are not contiguous inner spaces

P : the set of pages *p* in a document which have more than *m* candidate lines (default for *m* is 3, see Section 5.1)

y : the weight of candidate lines on a page that is large enough for us to allow less proximity between candidate lines

x : maximum string length considered acceptable in a table for a line with no inner spaces, measured as a percentage of the maximum line length found on the page, *max*

Algorithm:

Estimate the optimal threshold λ (see Section 5.1)
$C \leftarrow$ the set of candidate lines c such that $\{c \in L: \#S_\ell > \lambda\}$
For each $p \in P$
  $Weight_p \leftarrow$ # of candidate lines / # of non-empty lines in p
  $k \leftarrow$ acceptable distance between candidate lines for $Weight_p$
$i \leftarrow$ number of non-empty lines which could in principle be part of a table and lie before its first identifiable candidate line for $Weight_p$
$f \leftarrow$ number of non-empty lines which could in principle be part of a table and lie after its last identifiable candidate line for $Weight_p$
  $max \leftarrow$ maximum Len_c in page p

  While there are candidate lines in p not yet marked as *seen*
    $c_a \leftarrow$ the first candidate in p not yet marked as *seen*
    If $Weight_p > y$ then
      $c_z \leftarrow$ the last candidate in p
    Else
      Find the sequence of consecutive candidates c_w in p starting from c_a such that $Ind_{cw+1} – Ind_{cw} < k$
      $c_z \leftarrow$ the last candidate in that sequence
    End if
    If # of candidates in sequence $> m$ then
      Window $\leftarrow$ the sequence of lines from i lines above c_a to f lines below c_z
      For each line ℓ in *Window*
        Mark ℓ as *seen*
        If $E_\ell > 0$ or $Len_\ell < x.\ max$ **then** classify ℓ as *target area*
      Next ℓ
    Else Mark all candidates in the sequence as *seen*
    End if
  Loop
Next p

Instead of simply marking each line as belonging to a target area or not, an absolute index can be assigned to each distinct target area identified, that serves as a first approach to the delimitation of different tables. Thus, consecutive non-empty lines classified by the algorithm as table lines are assigned the same table identifier n. Any non-empty line not classified as table or a change in page will make the algorithm consider the current table is over, and a new identifier n+1 will be assigned to the next one. This is very important for the subsequent information extraction steps.

5.1 Parameters

We have considered three manners of estimating the optimal threshold, λ given global characteristics of a given source documents [7]:

- γ standard deviations, $S_E(D)$, from the mean, $\overline{E}(D)$, of inner spaces in the report D, determining γ as the value that minimizes the average square error when estimating the ideal threshold for the sample of 19 reports;

$$\lambda_1(D) = \overline{E}(D) + 0{,}46 * S_E(D) \quad (3)$$

- an equation, based on the most common number of positive inner spaces found in the lines of the report ($Mode_E$), whose coefficients were determined by the ordinary least square method based on the sample of 19 financial statements and as such are those that minimize the average square error of the estimates;

$$\lambda_2(D) = 17{,}9 + 1{,}4 * Mode_E(D) \quad (4)$$

- and the minimum of both equations (3) and (4). Better experimental results have been achieved with this last alternative.

The parameters of the algorithm were set at default values, which have been empirically tested: we have tested the method for $m = 3$ and $m = 2$ ($m = 3$ was found better) and for $x = 75\%$ and $y = 50\%$. We have determined k, i, and f as a function of the weight of table candidates on the page as described in Figure 4.

Weight < 30%	1	1	1
30% ≤ *Weight* <50%	2	2	2
Weight ≥ 50%	3	4	2

Figure 4: Default rule for the definition of parameter values.

Roughly, k corresponds to the maximum number of contiguous table lines with few inner spaces that we consider could appear within a table, i corresponds to the maximum number of contiguous table lines with few inner spaces that could appear in the beginning of a table, and f similarly for the end of the table.

The user can manipulate the values of m, k, i and f, as well as their relationship with the concentration of table candidates in the page. As such, the user can influence the performance of the method by providing the parameters which can best seize the types of tables he knows to hold the information for extraction. For example, if the context were such that the tables holding the information to be extracted were equally likely to be in pages with high or low concentration of table candidates, then the values of k, i and f should be the same for all values of weight. We can see examples of how the setting of these parameters works in Figure 5.

6 Evaluation

If the aim of our work was to detect all tables in a document, then the aim of the target selection algorithm would be to identify areas which contain as many table lines and as few non-table lines as possible. However, since the purpose of the method is to extract information, we in fact want the areas selected to include as many lines from tables with relevant information as possible and as few of all other

lines. Let us call the objective of only recognizing tables with relevant information our *context-specific purpose* and the objective of recognizing all tables our *generic purpose*.

A	When a column holds long vertically justified strings, there will be large vertical distances between the lines identified as candidates, therefore *k* needs to be bigger	150
B	The rest	200

Themes	Days for the meeting	Months
Demographic growth	10	March
Big bang	15	April
Greenhouse effect	20	April

Figure 5: Examples of tables that require parameter adaptation: for the table on the left, *k* should be set to at least 9, because there are 9 non candidate table lines between the two identifiable candidate lines, A and B; on the right, *i* should be at least 5, because there are 5 lines with too few inner spaces before the first identifiable candidate line

Either way, the cost of not identifying an interesting table line is much higher than that of classifying as target a line that is not one, since the targeted areas will undergo a large number of subsequent procedures to allow the isolation of the information we aim at extracting. The detection of the error and its subsequent correction is much harder in the first case. But the cost of correcting a misclassification of an interesting table line is not the same for all lines: if no portion of the table it belongs to has been identified, the cost is much higher than otherwise; or if the line missed is within a table otherwise spotted, its consequences on the rightful interpretation of the table are much smaller than if it is the first line of the headings of the table.

Therefore, we will consider as adequate metrics to test the quality of the results the percentage of lines we have been able to withdraw from the attention of the method (*economy*) and the proportion of the table lines in the document we wish to detect that are correctly identified by the method (*recall*).

Using default values, the algorithm was able to discard in average 74,4% of the lines in 17 of 19 reports, while keeping 99,6% of the interesting lines. In the other two reports, by manipulating the default values, the algorithm detected 100% of the desired information while discarding in average 62,3% of the examples. In an unseen test group of three reports the average performance was of 99,4% recall and 73,6% economy. No interesting table was entirely undetected.

For a generic purpose evaluation, and because we wish to detect the presence of any type of table, whatever type of page it belongs to, we have chosen for parameter values $k = f = 31$, $i = 5$, $m = 3$, independently of the weight of candidate lines on the page. The average results were 98,5% recall and 62,6% economy over the 19 examples. In the unseen group, average quality was of 99,7% recall and 63,6% economy. The only types of tables the method cannot detect were very narrow tables standing horizontally isolated in its page (e.g. a slightly narrower Figure 4), when in source documents mostly written in two or more columns and in pages with no other graphical elements. These are undetected because not enough candidate lines are

detected on the page. However, if we used $m = 1$, we would be able to detect 99,9% of table lines, with a 47,7% economy.

Figure 6 shows the performance of the methods parameterised for context specific and generic evaluation as described above. As can be seen, economy is practically always above 50% and recall is consistently above 95%, except for one financial statement which showed a table using dots as delimiters, instead of inner spaces. The method can easily be adapted to detect column separators other than white space, including dots.

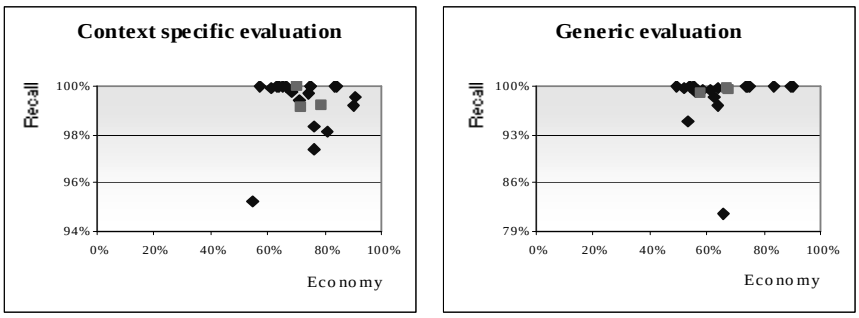

Figure 6: Performance of the methods parameterised for context specific or generic evaluation in the 22 documents (the three squares represent unseen documents). The only case where recall was below 95% was due to the presence in one document with two tables, which did not contain interesting information, using dots instead of spaces as delimiters. The method can easily be adapted to accommodate this

7 Conclusion

We have developed a method to locate portions of ASCII documents which have high likelihood of holding tables displaying the information we wish to extract. This is a necessary step in a longer process of information extraction from tables in text. We have applied the method in the context of financial statements – mandatory reports published by companies accounting for their activities on at least a yearly basis. Information extraction from such reports is relevant as a support for the decisions of many economic agents, but current approaches are mainly manual and time consuming.

The method has been developed by using data analysis techniques from statistics to data mining on a set of 19 financial statements published on the Web mostly in PDF by Portuguese companies. By deciding on each line based on the characteristics of itself, its neighbourhood, its page and its source document as a whole, the method is able to adapt its performance to the styles used by the authors in organizing graphical components on page, styles which can be very diverse in this highly and increasingly marketing influenced environment. Apart from this, the method includes some parameters which allow the user to affect the performance of the algorithm and adapt it further to the specificities of the tables it wishes to detect. The performance of the method was measured on the given set of reports, as well as on a separate test set. Results show that a lot of work can be saved right away, without losing relevant information.

Another main advantage of this method is that virtually any type of document can be converted to ASCII, including paper documents, after digitalizing and using optical character recognition software (OCR) that preserves the relative positions of the items on the page. Awareness to certain keywords and future recognition of titles has the potential of bringing recall even closer to 100% with very low economy costs, and will be sought in future work. The remaining tasks for information extraction from tables will also be researched in the future.

References

1. Conover,W. J. *Practical nonparametric statistics*. John Wiley, New York, (1999)
2. Hurst, Mathew: *The interpretation of tables in text*. PhD. Thesis, School of Cognitive Science, Informatics, The University of Edinburgh, United Kingdom (2000)
3. Kieninger, Thomas: Table structure recognition based on robust block segmentation. In *Proceedings of IS&T/SPIE's* 10th *Annual Symposium Electronic Imaging* '98: Science Technology – Document Recognition V, San Jose, California, USA (1998)
4. Ng, Hwee Tou, Chung Yong Lim and Jessica Li Teng Koo: Learning to recognize tables in free text. In *Proceedings of the* 37th *Annual Meeting of the Association for Computational Linguistics*, Maryland, USA (1999) 443-450
5. PDF, http://www.adobe.com
6. Pyreddy, Pallavi e W. Bruce Croft: A system for retrieval in text tables. Technical report 105, Centre for Intelligent Information Retrieval, Department of Computer Science, University of Massachussets, USA (1997)
7. Silva, Ana Costa e, Extracção da informação de tabelas em texto, MSc Thesis (in Portuguese), Faculdade de Economia, Universidade do Porto, Portugal (2003)
8. Tupaj, Scott, Zhongwen Shi, C. Hwa Chang and Hassan Alan: Extracting tabular information from text files. EECS Department, Tufts University, Medford, USA (1996)
9. Witten, I, Frank, E., *Data mining: practical machine learning tools and techniques with java implementations*, Morgan Kaufmann (2000)
10. XBRL, http://xbrl.org