Design of an end-to-end method to extract information from tables

Article *in* International Journal on Document Analysis and Recognition (IJDAR) · January 2006

DOI: 10.1007/s10032-005-0001-x · Source: DBLP

Ana Costa e Silva
University of Porto

Alípio Mário Jorge
University of Porto

Luís Torgo
Dalhousie University

Design of an end-to-end method to extract information from tables

Ana Costa e Silva [*]
Banco de Portugal, Portugal
acsilva@bportugal.pt

Alípio M. Jorge
Universidade do Porto, Faculdade de Economia do Porto, LIACC, Portugal
amjorge@liacc.up.pt

Luís Torgo
Universidade do Porto, Faculdade de Economia do Porto, LIACC, Portugal
ltorgo@liacc.up.pt

Abstract; This paper plans an end-to-end method for extracting information from tables embedded in documents; input format is ASCII, to which any richer format can be converted, preserving all textual and much of the layout information. We start by defining table. Then we describe the steps involved in extracting information from tables and analyse table-related research to: place the contribution of different authors, find the paths research is following, and identify issues that are still unsolved. We then analyse current approaches to evaluating table processing algorithms and propose two new metrics for the task of segmenting cells/columns/rows. We proceed to design our own end-to-end method, where there is a higher interaction between the different steps; we indicate how back loops in the usual order of the steps can reduce the possibility of errors and contribute to solving previously unsolved problems. Finally we explore how the actual interpretation of the table not only allows inferring the accuracy of the overall extraction process but also contributes to actually improving its quality. In order to do so, we believe interpretation has to consider context specific knowledge; we explore how the addition of this knowledge can be made in a plug-in/out manner, such that the overall method will maintain its operability in different contexts.

1. Introduction

Companies worldwide have the legal obligation of producing and publishing, on at least a yearly basis, financial statements where they account for their activities over the period. These documents vary considerably in length (between ten to over one hundred pages) and style (one or more columns, many or no charts), depending on how important the image of the company with the public is for the authors. The information contained in these reports is valuable to the decision-making of a variety of agents. However, much of the relevant information is contained in a heterogeneous set of tables and is currently extracted mainly by hand by all interested parties. We propose to create an integrated end-to-end method to allow automatic extraction of the information of specific tables contained in these documents. The method is designed to handle a broad range of input that we simply convert to ASCII; and to operate in diverse contexts.

We will start by defining "table" in Section 2. We then describe research related to extracting information from tables in various formats in Section 3, by following the steps of the information extraction process. How to evaluate table-related methods is described in Section 4. In Section 5, our own method is described in detail, and supporting experimental evidence is provided when available.

There are several ways to read this article: from end-to-end, obviously; but it can also be read in an encyclopaedic manner, whereby the reader can choose only those sections that address the task(s) he or she is most interested in in sections 3, 4 and 5.

[*] The opinions expressed in this article are the responsibility of the authors and do not necessarily reflect those of Banco de Portugal.

2. What is a table?

A lot of discussion has been made over the years on what a table is, The prolonging of the discussion could induce us to think that this has hardly a clear-cut answer. However, definitions have been presented in a somewhat distant past, to which more recent researchers have added little:

"*A table is an object which uses linear visual cues to simultaneously describe logical connections between the discrete content entries in the table. A content entry is the basic component of information in the table […] (and) can be any visual symbol*".

Cameron (1989)

The two main elements behind this definition are that there exist simultaneously *linear visual clues* in the form of columns and rows that represent *logical connections*; as such there is a relationship between the relative position of the items and their conceptual relationship.

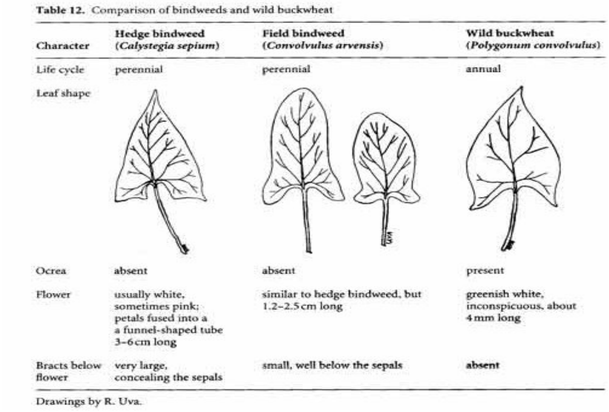


Table 12. Comparison of bindweeds and wild buckwheat

Character	Hedge bindweed (*Calystegia sepium*)	Field bindweed (*Convolvulus arvensis*)	Wild buckwheat (*Polygonum convolvulus*)
Life cycle	perennial	perennial	annual
Leaf shape			
Ocrea	absent	absent	present
Flower	usually white, sometimes pink; petals fused into a a funnel-shaped tube 3–6 cm long	similar to hedge bindweed, but 1.2–2.5 cm long	greenish white, inconspicuous, about 4 mm long
Bracts below flower	very large, concealing the sepals	small, well below the sepals	absent

Drawings by R. Uva.

Figure 1. A table containing images (http://www.hort.cornell.edu/extension/weeds/table.jpg)

Therefore, the example in Figure 1 is a table: the leaf types presented on the header row on the top are classified according to the set of characteristics we have on the leftmost column. To understand the data presented in the remaining cells of the table, one must connect them with cells on the header row and on the leftmost column with which they are aligned.

Let us look at other examples. Is an index (Figure 2) a table?

INDEX

image	12
implicit	22
inner product	3
input	45

INDEX

image, 12
implicit, 22
inner product, 3
input, 45

Figure 2. Indexes versus tables

To begin with, all the items are aligned, both vertically and horizontally. Besides, there is meaning in the horizontal alignment between the numbers and the words, since the number describe one characteristic of the keywords, their location, even if no header row is shown. So, one can say such an index is a "table".

However, if we eliminate the space between the words and the numbers, we still have an index but is it still a table? The vertical alignment of the numbers is lost, so there is no *linear clue* that they play the same role in all lines of the index (there are other clues, but they are not linear). So what we have is a list.

Louise
Thomas
Joanna
Josephine
Susan
Alexander

Figure 3. Lists versus tables

This might suggest that the difference between a list and a table is the existence of only one column or of more than one column. Nonetheless, lists represented in columnar format do exist; the example in Figure 3 is such a case: it is the absence of *logical connections* that dictates this is not a table and a purely graphical analysis would not be able to distinguish it. So what discriminates tables from lists is the absence of either *linear visual clues* OR *logical connections* behind the *linear visual clues* presented.

Figure 4 contains an organizational chart. It is a representation of the relationships between different components of an organization. On top, come the highest elements in the hierarchy; horizontally aligned components that are not connected with lines typically have the same power but are otherwise unrelated to each other; horizontally aligned connected elements typically provide support to the items they are connected to. Similar elements are combined in groups, e.g. under the heading "Brasil" are elements that share that location characteristic; within each group the elements may or may not be hierarchically organized. Relationships of power between different elements are depicted with colour and indentation but in particular with the use of lines and adjacent numbers that quantify the degree of power.

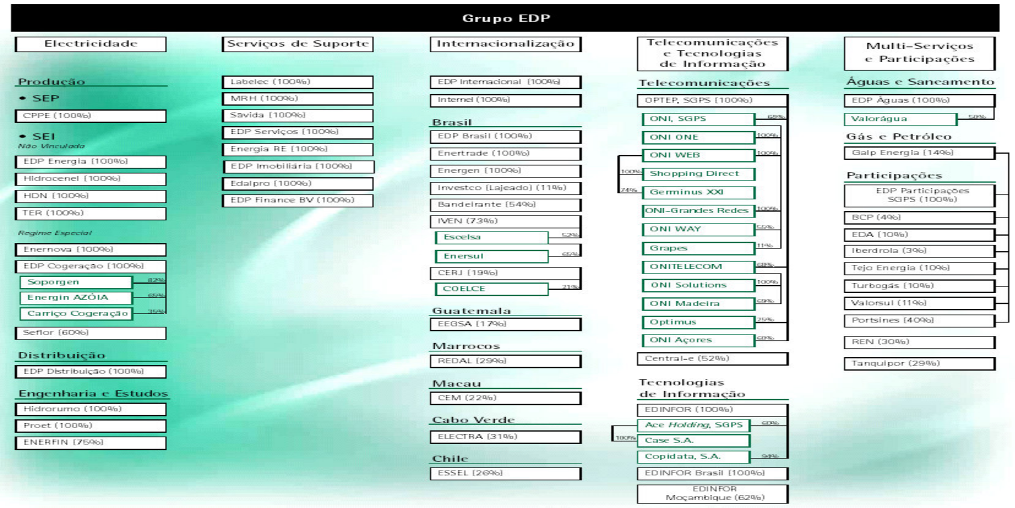


Figure 4. Diagrams versus tables, EDP group organizational chart, EDP(2000)

Is the example in Figure 4 a table or a diagram? There are *linear visual clues* to describe *logical connections* between the elements, however the contents of the cells and their alignment are not enough to understand the idea transmitted, line art is essential for that understanding. As such, this is not a table, it only looks like one, it is a diagram.

We can then conclude that the difference between a table and a diagram is that in a diagram the presence of clues other than the content of the items and their relative position is essential to the understanding of the relations being transmitted; as such, a diagram is a more irregular manner of conveying information.

Finally, a form is a document with a predefined template that is repeated thousands of times. On top of this template, users add handwritten or machine-printed data that can overlap the template. Forms may or may not contain tables. However, the almost compulsory presence of line-art to delimit each cell in a table-form and the overlapping of user written data with the template make it a different research problem from that of tables embedded in non-form like documents.

Summing up, for working reasons, we propose a definition of table that captures the essence of Cameron's view by establishing verifiable properties:
Definition: A *table* is a graphical grid like representation of a matrix M*i*,*j*, where:

1. each element *i*, *j* of the matrix is atomic;
2. there are *linear visual clues*, i.e. the elements of each row *i* (column *j*) of the matrix tend to be horizontally (vertically) aligned;
3. *linear visual clues* describe *logical connections*, i.e. some of the elements of each row *i* (column *j*) are bound in their meaning to each other and are not independent; as such, each row (column) of a table is unbreakable, in that if two elements on a given row (column) exchange places, then the two corresponding rows (columns) must do so;
4. eventual line-art does not add meaning otherwise not present in the relative positioning of the cells in the table.

So is Figure 5 a table? No, it is a diagram.

3. Table related research

According to Mathew Hurst (2000), several models can be used to represent a table. These may be the output of common table-related tasks (Figure 5).

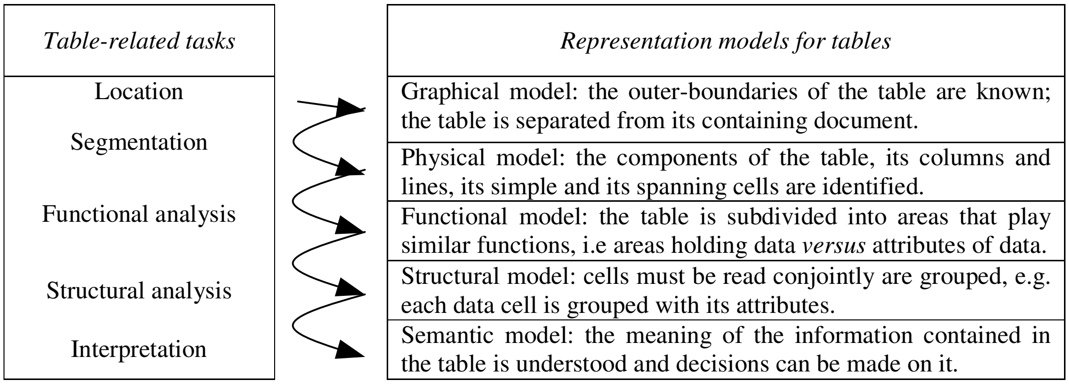


Table-related tasks

Location

Segmentation

Functional analysis

Structural analysis

Interpretation

Representation models for tables

Graphical model: the outer-boundaries of the table are known; the table is separated from its containing document.

Physical model: the components of the table, its columns and lines, its simple and its spanning cells are identified.

Functional model: the table is subdivided into areas that play similar functions, i.e areas holding data *versus* attributes of data.

Structural model: cells must be read conjointly are grouped, e.g. each data cell is grouped with its attributes.

Semantic model: the meaning of the information contained in the table is understood and decisions can be made on it.

Figure 5. Table related tasks as transformations of table representation models

Different representation models serve as input to different research fields. For example, physical models are usually required for document analysis, but a graphical model may be sufficient. For information retrieval functional models are sufficient. Information extraction usually requires tables to be represented structurally or semantically. Authors from all of these fields have often studied tables with the purpose of converting them into a model that can be used in their own areas. As such, it is only natural that more authors have focused on the basic table related tasks than on the more knowledge based ones. In Figure 6 we position selected papers along the span of tasks they have addressed (we use a diagonal line to mark the papers that only partially address a task).

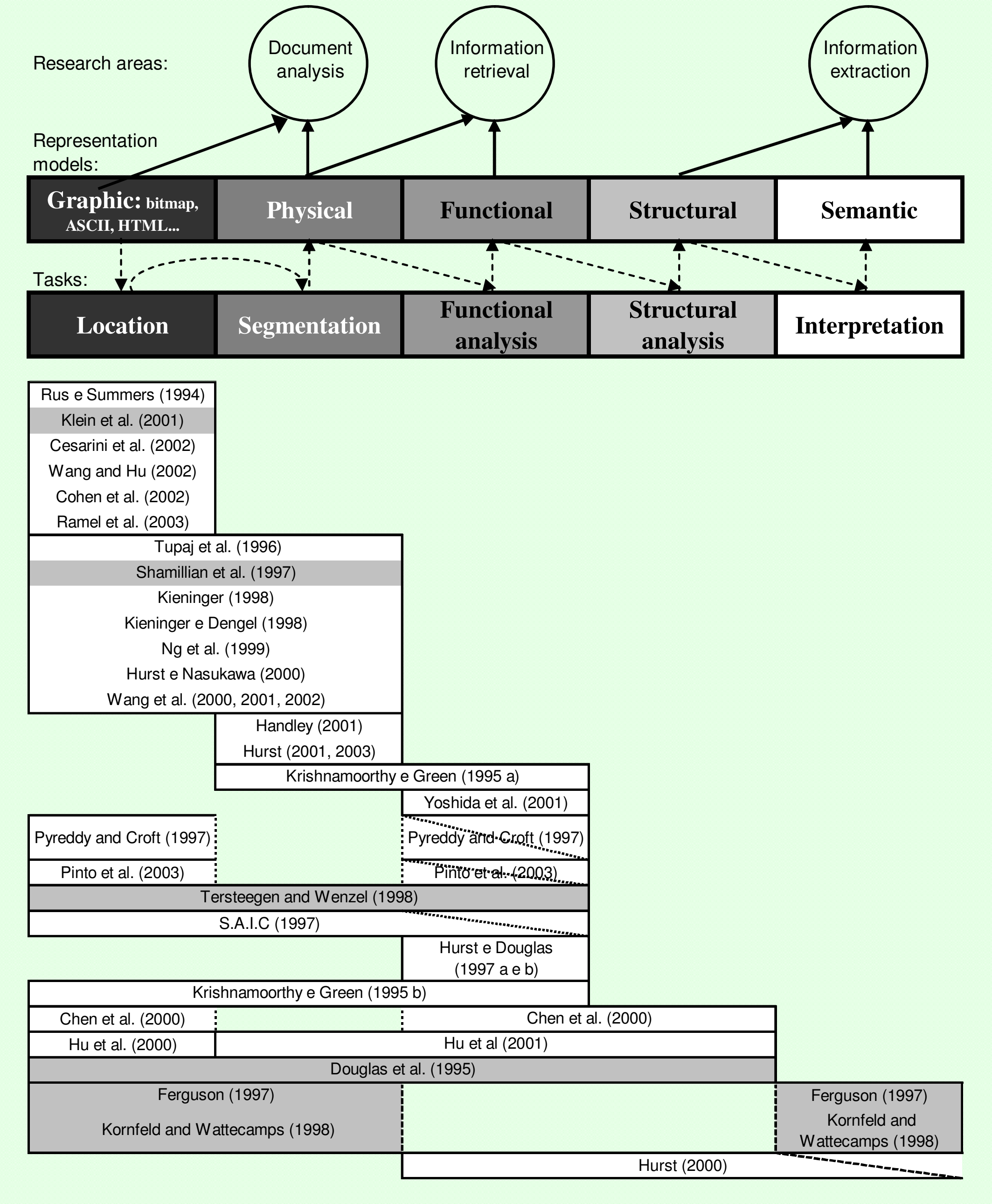


Figure 6. ***Input-output*** **relations between table-related tasks and representation models; Positioning selected authors (in grey, context specific works)**

Most researchers, even in papers that deal with more than one task, treat table-related tasks independently and sequentially. Exceptions occur mostly in approaches to location and segmentation: while most authors begin by finding the whole table, then search for its lines and columns and finally assign the cells to the formats thus built - a *top-down approach*, a few follow *bottom-up approaches*, i.e. expressions (or in image files word bounding boxes, *wbb* for short) are joined into columns and lines and these, when appropriate, into tables. Some authors also follow a *mixed approach*, e.g. begin by finding columns, then gather them in tables and finally search for cells.

There are nonetheless other table related tasks we will not approach in this paper, e.g. table compression (see Buchsbaum et al. (2000)), table clustering (identifying tables that present related information, usually by analysing their attribute cells) and table merging ("representing related tables as one large table") (see Yoshida et al. (2001)).

In the remainder of this section we will characterize how selected authors have dealt with the five tasks identified above: restricted as each was by the input available, we will see what sort of approach they followed, the features they recurred to and the classifiers utilized.

3.1. Location

"*Locating a table on page involves differentiating the table from other elements such as body text, headings, titles, bibliographies, lists, author listings, abstracts, line drawings and bitmap graphics*". Tupaj (1996)

In the location task, a decision must be made as to whether a given area on a page is a table or not. This task is relevant in image (I) or simple text (T) but also in poorly or unevenly tagged formats such as many HTML (H) or XML (X) can be. Below, we present in chronological order how several authors have dealt with the problem for the four types of input considered (T, I, H, X). In Table 1, we will classify each author under a set of characteristics according to the input they work with, the features on which a decision is based, the classifier chosen and whether or not their method recurs to pre-defined or user defined parameters. We will not reproduce their evaluation results, since these are affected by the homogeneity or variety of the documents the authors tested their methods on. For example, "tables from scientific journals were easier to locate than financial tables […] (which) layout is substantially more varied and harder to detect with current algorithms", Tupaj (1996).

(Format of input documents is T) Rus and Summers (1994). A block of text is characterised as a table if it has columns separated by white space (blank characters), "tolerating irregularities within (user) specified error bounds", and if the contents of the cells in each column are lexically stable.

(T) Douglas et al (1995). Each block of characters that is surrounded by rivers of white space is tested for table-like characteristics: table column blocks will be thin and surrounded by comparatively large white space corridors. Adjacent blocks are then grouped into one table if to the non-empty cells in one correspond non-empty cells in the other. A heuristic determines if these areas are tables.

(I) Green and Krishnamoorthy (1995). Users provide a model of the relevant tables, which includes a description of the type of line or space gap that separates tables from the rest of the document. The areas surrounded by these delimiters are marked as tables.

(I) Tupaj et al. (1996). A similar heuristic to Douglas et al. (1995) determines whether text blocks are possible table columns; such blocks are presented to an optical character recognition software (OCR) and each of their lines is characterised on the number of words it contains (0, 1, >1); then the neighbourhood of a set of keywords (e.g. *table*) is searched for predefined sequences of table-like lines, which are marked as tables; four empty lines heuristically signal the end of each table.

(I) Shamillian et al. (1997). Again a model based system, where the black pixel distribution of each line in the document is compared against the user-supplied table line. Unlike Green and Krishnamoorthy's method, which allows detection of any table surrounded by the specified delimiter, this method only finds tables with a fixed number of columns and predetermined field length.

(T) Pyreddi and Croft (1997). Taking a group of *k* lines, a search is conducted for gaps where the number of vertically aligned non-space characters is below a threshold; groups with 3 or more such gaps are table lines.

(I) Science Application International Corporation (1997). Segmentation is performed before location. A group of ten heuristics is classifies lines as tables. However, spanning cells or table lines where only one table column is filled in.will not be detected

(X) Ferguson (1997) and Kornfeld and Wattecamps (1998). Working in the context of SEC filings, candidate tables are identified on the basis of simple heuristics (e.g. number of columns); the areas above the tables are skimmed (in 1998 a *LR*(k) parser was used) for titles that are sufficiently close to those of the tables they want to extract.

(I) Tersteegen and Wenzel (1998) present an interesting exception to the traditional independence of the five table related tasks – functional analysis precedes the location algorithm to locate table headers within the document (see section 3.3). This is an important advantage they derive from operating within the specific context of German business letters. Their method is based on a set of models of tables that serve as reference. The similarity between the document's table headers and those of the reference tables is measured.

(T/I) Kieninger (1998). More interested in segmentation than location, this author groups the words (or, in image files, wbb) into blocks based on their vertical alignment. Those blocks that contain only one word (or wbb) per line, are said to be table columns.

(T) Ng et al (1999). Using a decision tree induction algorithm together with a back propagation algorithm for neural networks and on the basis of 27 features that classify each line of the document according to its own characteristics and those of its precedent and subsequent lines, each line is classified as belonging to a table or not. The characteristics are related with space distribution and the presence of specific characters.

(T/I) Hu et al. (2000). Each group of lines is tested for its *merit* as a table, by measuring the resemblance among them as the weighted average of the number of white spaces with the same position and the number of overlapping words (or wbb) with similar sizes. The group of lines that maximize merit above a certain limit is a table. However, the input must be a single column document.

(H) Chen et al. (2000). After excluding table tagged areas with non-table characteristics, three resemblance measures are taken between each cell and its surroundings: the number of common characters, if they hold he same content types and if they both hold mostly numbers. Tables will have a number of similar cells is above given a threshold.

(T) Klein et al. (2001). Working within the context of medical liquidations of health insurance companies, tables are identified through at least one of three features: a) there are a set of known table headers, b) there are layout structures which resemble parts of columns, or c) there are groupings of similar lines.

(I) Cesarini et al. (2002) created a table location method that finds tables within a common output of document image analysis systems, an X-Y tree, specifically they work from a Modified X-Y tree they had developed in previous research. The tree is analysed to identify sub-images that correspond to tables. Firstly, regions surrounded by lines are identified; then further parallel lines or rivers of white space are searched. Sub-tables are identified that satisfy a set of geometrical properties and these are then merged to form tables. The method relies on the definition of five thresholds that are chosen to maximize performance, as measured by the *Table Evaluation Index* (see section 4) along a training set of images; as such the method is trainable.

(I) Wang et al. (2000, 2001, and 2002). After detection of large blank blocks that delimit document zones, table candidates are identified by a binary decision tree on the basis of specific features (2000). To these table candidates, segmentation is applied (see next section) and the probability of the table being consistent is estimated, based on the percentage of the area occupied by the non-blank parts of the table, its width, and the justification of cells within their columns. When the probability is above 50%, we have a table (2001). Finally, table location is perfected by measuring the improvement that would occur in that probability if the table's upper and lower neighbouring zones were merged with it and all were considered as table or as text (2002).

(H) Cohen et al. (2002). Tables are located with a trained classifier that labels each occurrence of an HTML table tag as a relevant table or not. Each table is described with general attributes such as the number of rows and columns, the numbers of cells holding strings and the number of non-spanning cells. Different learning algorithms were tested, namely Naive-Bayes, Maximum Entropy Classifier, a decision tree built by the C45 algorithm and the Winnow classifier, the latest having achieved better performance.

(H) Wang and Hu (2002). Each occurrence of a table tag in a HTML file is classified as a genuine or a non-genuine table. Decision trees and SVM classifiers were trained on a large set of tables obtained from diverse web pages. Features include three types of features: table layout (e.g. length consistency of the contents by column/row/whole table); content type (e.g. alphabetic, hyperlink, image); word group (which provided little performance improvement over the remaining features).

(T) Pinto et al. (2003) begin by assigning a label to each line that indicates its function as data, header or caption; they also characterise it on the presence of white space, content type and on the presence of certain separator characters. Conditional Random Fields (CRF), a type of graphical model, is then used to estimate the probability of a sequence of labels given a sequence of observed document lines. As such, they get a partial functional analysis. They claim, and give experimental support, that CRF perform better than Hidden Markov Models and a Maximum Entropy classifier (a log-linear model).

(H) Ramel et al. (2003) agree with Klein et at. (2001) that “different perceived regularities complete each other to let the reader detect the presence of a table and identify its structure”. Their method is sensitive to rectangular gridlines that may delimit all the cells of the table or only a few; they believe more sparsely delimited tables tend to have more regular text blocks, to which their method is also sensitive. In analysing both gridlines as text blocks, regularity is searched for: minimal size rectangles formed with the existing gridlines should not overlap; alignment and relative proximity are considered when joining words into lines, and then columns, and then blocks, and finally tables. Cells will only be detected if surrounded by gridlines; in text it is unclear that spanning cells will be detected.

<u>In summary</u>: as can be seen in the table below: most approaches to location are top-down; analysis of poorly tagged HTML/XML files are reasonably recent; few methods require knowledge from outside the system (such as some form of user supplied table model), and these are already in a more distant past; space distribution is almost always analysed but it is assessed through different means, although character alignment charts are more out-dated, probably because they are less appropriate for non-Manhattan like table columns; similarity/cohesion among different cells/lines/columns is a common form of features; few methods perform semantic analysis; the use of some sort of parameter is common, but most are pre-defined; rule or heuristic based classifiers were predominant until 2000; application of data induced or probabilistic models is quite recent but most of the more recent works apply this sort of automatic classifiers.

		Rus and Summers	Douglas et al.	Green and Krishnamoorth	Tupaj et al.	Shamillian	Pyreddi and Croft	SAIC	Ferguson	Kornfeld and Wattecamps	Tersteegen and Wenzel	Kieninger	Ng et al.	Hu et al	Chen et al.	Klein at al.	Cesarini et al.	Wang et al.	Cohen et al.	Wang and Hu	Pinto et al.	Ramel et al.
		1994	1995	1995	1996	1997	1997	1997	1997	1998	1998	1998	1999	2000	2000	2001	2002	2000-2002	2002	2002	2003	2003
Input	Text file	x	x				x					x	x	x		x					x	
	Image file			x	x	x		x			x	x		x			x	x				
	Poorly-tagged file								x	x					x				x	x		x
Approach	Top-down	x		x		x	x	x	x	x			x	x	x	x	x	x	x	x	x	
	Mixed		x		x																	
	Bottom-up											x										x
Features	Existing mark-up								x	x					x				x	x		x
	User defined model			x		x					x											
	Grid lines			x													x					x
	Space distribution	x	x	x	x	x	x	x				x	x	x		x	x	x			x	
	- Detection of rivers of white space	x	x	x	x		x										x	x				
	- Relative word (wbb) position					x		x				x		x		x		x				x
	- Existence of consecutive space characters																				x	
	Similarity or consistency measures		x		x	x					x			x	x	x		x		x		x
	Character features												x		x						x	
	Content type features	x									x				x					x	x	
	Keywords				x				x	x	x					x						
	Semantic analysis										x									x		
Use of parameters	Predefined		x		x		x	x						x	x						x	x
	User defined	x					x															
	Trainable parameters																x					
Classifier	Heuristics	x	x		x		x	x	x	x		x				x						x
	Similarity-based clustering		x		x							x		x	x							x
	Comparison with a table model			x		x					x											
	Probabilistic approach	x																x			x	
	Data induced model												x					x	x	x		

Table 1: Strategies for locating tables in documents, summary table.

Unsolved issue: most methods have difficulty in distinguishing two tables that are similar in terms of number and appearance of columns/rows, if they are vertically/horizontally aligned and reasonably close (see section 5.4). In these cases merging errors will occur. Such tables can only be adequately split if some sort of functional analysis is performed that allows the identification of header rows in the middle of the table and poses the question of whether these header rows mark the beginning of a new table. So using the results of functional analysis to improve location is a lane of opportunity.

3.2. Segmentation

The purpose of segmentation is basically to give a physical description of the table, i.e. identify its cells and their relative positions, as well as its rows and columns. We will summarize the approaches of different authors to this task.

(Format of input documents is I) Green and Krishnamoorthy (1995). The user provides a table model with the characteristics of the relevant tables. This model includes the characterization of the delimiters that separate the table's different areas, down to its columns, rows and cells. Accordingly, the table is sequentially subdivided into the smaller levels of the model; and each subdivision is named as the concatenation of its own name with that of each of the previous divisions. The output is an X-Y tree.

(T) Douglas et al. (1995). Having followed a mixed approach in the location task, columns have already been delimited, so only rows are necessary. Empty lines delimit rows; if no empty lines exist, physical lines may have to be aggregated to form rows; how this is done is not explained.

(T) Tupaj et al (1996). Vertical corridors of white space traversing the table are identified; those lines interrupting a significant number of such corridors are excluded from the table; in the line where the corridors are narrower, the column delimiters are is found as the central point of the corridor. However, cells spanning several columns are not appropriately delimited; and columns with too few cells (ghost columns) are found.

(I) Shamillian et al. (1997). In the user supplied model table-row, column delimiters are found in the white space gaps; these delimiters are then applied to the table.

(I) Science Application International Corporation (1997). Tabular data processing utilizes OCR results with positional data, font information, horizontal lines and heuristic rules to determine cell boundaries and contents. If none of these features are present, horizontal and vertical white space will be used.

(X) Ferguson (1997) analyses column header rows to determine approximate column boundaries, thereafter extracting each line until the end of the table is detected. Kornfeld and Wattecamps (1998) cut table lines into cells such that each line will have at least one cell and no more cells than there are columns in the table. An algorithm will then merge vertically aligned cells which contents would be too large for the horizontal space available in the column.

(I) Tersteegen and Wenzel (1998). After identifying the table under analysis as one referenced by their table models, they use the information in their model to segment columns and rows, the intersections of which reveal cells. Contents are then attributed to these cells, a tolerance range being allowed.

(T) Kieninger (1998). At the end of his location algorithm, this author is left with the similar problems as Tupaj et al's (1996) leave unsolved in their segmentation approach: A) columns under spanning cells are not separated; to solve it, those expressions that are vertically aligned with only one expression in the row below (and vice versa) are isolated into a single column; B) cells that span more than one line are not aggregated; to solve this, each cell is characterised on the number of tokens it contains (0, 1, >1); 0 and 1 cells will be vertically isolated; the segmentation of >1 cells will follow that of 1 (as long as the first word in the following line does not fit the available room in the same column on the previous line, a care similar to Kornfeld and Wattecamps (1998)).

(T) Ng et al (1999). Column separators are expected to occur when there is a change from an alphanumeric character to a non-alphanumeric character. Each vertical 1-character wide column is compared against the previous and the following and the changes (or non-change) between character types are counted. C.4.5 decision tree induction algorithm together with a back propagation algorithm for neural networks was used to derive a classifier that decides whether each vertical line of the document is a column separator or not. To find line separators, the same algorithms generate a classifier based on the idea that different table vertical lines look similar. For features, the position of the first non-space character in the line is kept and then each line is compared character-by-character against the previous and the next to count the changes between space and non-space characters.

(I) Handley (2000) begins by detecting whether the cells in the table are fully delimited by line-art. If not, the location of eventual line-art is combined with the relative position of the word bounding boxes (wbb) and white spaces surrounding them. A first estimate of lines is made taking the histogram of the horizontal projection of the wbb in the table; then wbb with significant vertical overlap that are close to each other are merged into cells; a histogram of the vertical projection of the cells is made to estimate columns, but cells close to the top of the table weigh less, since these are more likely to be spanning. A set of heuristics is used to spanning cells and rows. With these, a more precise estimation of rows and columns is made using the same histogram approach.

(T) Hurst (2001 and 2003). The method implies having a model of the language used in the document; this contains the number of times two words appear together in a training set. The document is searched for points where a decision has to be made about what the follow up of the text should be: whether the next word in the same line or any words to its left in the next line. The system chooses the most coherent follow-up among the candidates given the model of language at hand. The output for tables is a segmentation into cells (2001), which includes spanning cells. In (2003) the author aims at aggregating these cells into columns and rows. This is done by linking together cells that have any horizontal or vertical overlap with each other. A set of constraints must be met when establishing these links, which may result in the identification and correction of mistakes made in the cell segmentation phase and which guarantees some robustness to cell misalignment (which is a consequence of editing environments).

(T) Hu et al. (2001). A hierarchical clustering algorithm is applied to group the words in the table according to the closeness of their beginning and end positions; a set of heuristics then determine the best grouping levels that correspond to a column; after taking the leftmost column for header, an approach similar (but less cautious) than Kieninger's is used to identify table rows - those lines with no data are joined with the previous (with no care for resulting cell length). This simple heuristic causes cells containing titles for lower attributes to be joined with them; and does not account for data cells that span several lines.

(I) Wang et al. (2001). A vertical projection of the word bounding boxes is made and columns are defined as the space between the end of a valley and the beginning of another. In (2002) the authors mention the intention of building a probabilistic approach to this task.

In summary: the table below presents a classification of the segmentation methods outlined above in terms of the type of input each author works on, the features taken into account, the method used to reach that decision and the output produced. As can be seen: few methods require knowledge from outside the system, such as some form of table model, and these are already in a farther past; few methods use semantic information; space distribution is very commonly applied and is assessed through different means, although relative word position has become the most common, probably because they are the most appropriate for non-Manhattan tables; use of some sort of parameter is still common, but all are pre defined; application of data induced or probabilistic models is still rare, most methods are still highly heuristic-based.

		Green e Krishnamoort	Douglas et al.	Tupaj et al.	Shamillian	SAIC	Ferguson	Kornfeld and Wattecamps	Tersteegen and Wenzel	Kieninger	Ng et al.	Handley	Hu et al.	Wang et al.	Hurst
		1995	1995	1996	1997	1997	1997	1998	1998	1998	1999	2000	2001	2001	2001-2003
Input	**Text file**		x	x						x	x		x	x	x
	Image file	x			x	x			x	x		x			
	Poorly-tagged file						x	x							
Features	**Existing mark-up**						x	x							
	User defined model	x			x				x						
	Grid lines	x				x						x			
	Space distribution	x	x	x	x	x				x		x	x	x	x
	- Detection of rivers of white space	x	x	x	x	x						x		x	x
	- Relative word (wbb) position					x	x	x		x			x		x
	Character features					x					x				
	Language model														x
Use of predefined parameters			x	x		x						x			
Classifier	**Heuristics**		x	x		x	x	x		x		x	x	x	x
	Comparison with a table model	x			x				x						
	Data induced models										x		x		
	Maximization of a measure														
Output	**Physical description**		x	x	x	x	x	x	x	x	x	x	x	x	x
	Hierarchy detection	x													

Table 2: Strategies for segmenting tables, summary table.

Segmentation has become a less fashionable problem since more recent papers have focused their attention on poorly marked-up documents types (present on the web), which typically hold a good cell/column/row segmentation. However, two main issues are still unsolved.

Unsolved issues: Most approaches are still not generalist enough to derive the physical model of Table 3, if no gridlines are available and cell (1,1) contains more than one distinguishable word. An exception would be Hurst (2001, 2003). Rather than using relative word position as a first step to column identification (as would Tupaj (1996) or Kieninger (1998)), this author begins by using a language model to concatenate different words into cells, with which columns and rows are searched for. This makes his method less likely to mistake casual space character alignment with a column delimiter, making it less permeable to ghost columns. Notice that Handley (2001) works with cell estimates rather than words to find columns (and for that matter also Rammel (2003) in the location task), but his heuristic approach to spanning cells identification makes his method less general.

********** ******** ************		*******	*******
*******	*******		*******
*********	******		*******

Table 3: A table most segmentation algorithms would not process accurately: cell (1,1) would probably be split in two or three cells, given the casuistic alignment of the space that separates two of its words with the space that separates two columns in the lower rows. This can be avoided if cells are located before rows and columns are formed.

But more importantly, all systems seen so far, which do not count on the presence of line-art or existing mark-up, fail to attribute to their rightful columns those cells which "extent only spans a subset of the values that its interpretation must be applied to" (Hurst (2003)). However, this is quite likely to happen in attribute rows. We propose a possible solution to this problem in section 5.5.2.

3.3. Functional analysis

Functional analysis aims at classifying a table area, be it a line, column or individual cell, according to the function it performs in the table: it either holds the *data* the table is conveying or the *attributes* that describe that data. We believe other elements, such as the table's titles and footnotes are not a part of a table but rather are attributes of it, even if they may facilitate the understanding of the information it contains.

Douglas et al. (1995). Operating within a specific context, the function of each cell is determined by a set of heuristics based on the coordinates of each cell, the semantic type of its content within the context's specific language, and a Boolean for whether there is a semantic connection between the content of each cell and the table's title.

Green and Krishnamoorthy (1995). A user-built model of the table maps relationship between each relevant table area (column, row, or cell) and its function.

Pyreddy and Croft (1997). Aiming at information retrieval, a set of heuristics classifies each line as attribute or data by comparing it with average characteristics taken from all table line. The comparison is based on the line sizes, alignment and gap structure. Whether the line contains mostly numbers or strings, and the number of columns it holds is also taken into account.

Science Application International Corporation (1997). A heuristic classifies as attribute the first line which has all (or all but the first) column filled in.

Hurst and Douglas (1997). Based on simple table models, they measure the cohesion among different cells within rectangular areas of the table, to evaluate whether each area could be an instantiation of one of the table models. With those areas that have enough cohesion, a tiling is built that covers the whole table with no overlaps. Evaluation revealed poor results.

Tersteegen and Wenzel (1998) begin by segmenting German business letters into non-text and text regions and these are presented to an OCR. The text is skimmed for context specific keywords (which are related in a *look-up dictionary*) and for content type characterization; keywords' relative position is noted. An area of up to three lines is drawn around each keyword. If an area holds too few keywords, it is disregarded; otherwise, the area with the maximum number of keywords is classified as table header.

Hurst (2000) tries three different approaches: a set of heuristics (considering the physical model of the table, the semantic types of its content, and a comparison of the content with the text in the document), a Naïve-Bayes classification (with physical and content-related attributes derived from comparison with surrounding text and titles) and a pattern based classification (which implies a definition of physical patterns for each cell on the basis of its surrounding cells, a mechanism to compare new examples against these predefined patterns and a voting mechanism to choose a classification for the cell based on all the patterns that apply to it); he also makes several combinations of the three methods. Heuristic approaches showed rather low recall but good precision.

Chen et al. (2000). The same similarity measure used in the location task (see section 3.1) is now applied to compare each line against the last table line (that is clearly identifiable since the input is HTML); if the number of similar lines is above a threshold and the first line is dissimilar from the rest, then it is classified as an attribute line. The same is applied to columns. In tables with spanning cells, these are first used to cut the table into sub-tables and an independent classification is made in each sub-table. More than one row/column of attributes will not be found if there are no spanning cells.

Yoshida et al. (2000). Assuming that "table structures can be categorized into nine types", the Expectation Maximization Algorithm is used to calculate the probability of a table following one of the nine types, given an ontological analysis of its strings.

Hu et al. (2001) assume the left most column in the table is a header. Then up to five rows will be compared with the remaining for content type consistency – inconsistent rows will be classified as headers; the presence of spanning cells serves to detect header hierarchy.

Pinto et al. (2003). As described in Section 3.1, Conditional Random Fields were used to automatically label each line of a document according to the role of the line (header, sub-header, data, separator, etc.).

In summary: Table 4 presents a classification of the functional analysis methods outlined above in terms of: the type of features taken into account and the method used to reach a decision. Although a good number of methods still rely on space distribution and consistency measures as features, it is interesting to notice that the relative weight of some sort of user defined table models is more common for this task than for any of the two before, which is indicative of the complexity this task involves. It is also noticeable that machine-learning methods have not yet been applied to this task. We believe there are two main reasons for this: a) many of the authors that have applied data induced models to tables are still dealing with the first tasks; and b) the classification of each unit is potentially dependent on the classification of each of the elements that surround it in all directions, which makes it an intricate problem of interdependent classification.

		Douglas et al.	Green e Krishnamoort	Pyreddi e Croft	SAIC	Hurst e Douglas	Tersteegen and Wenzel	Hurst	Chen	Yoshida et al.	Hu et al.	Pinto et al.
		1995	1995	1997	1997	1997	1998	2000	2000	2000	2001	2003
Indicia	Space distribution	x		x	x		x	x				x
	- Relative position of cells	x		x	x		x	x				
	- Existence of consecutive space characters			x								x
	User-defined table model		x			x		x		x		
	Character features			x					x			x
	Content type features			x			x	x	x		x	x
	Semantic analysis	x					x	x		x		
	Comparison against surrounding text	x						x				
	Similarity or consistency measures			x		x	x	x	x		x	
Use of predefined parameters				x		x	x	x			x	x
Classifier	Heuristics	x		x	x		x	x	x		x	
	Comparison with a table model		x			x				x		
	Pattern detection			x		x		x				
	Probabilistic approach							x		x		x

Table 4: Strategies for functionally analysing tables, summary table.

3.4. Structural analysis

"*[T]able mark-up contains a great deal of information about what a table looks like... but very little about how the table relates the entries. ...[T]his prevents me from doing automated context-based data retrieval or extraction*".

Thompson (1996)

The purpose of this task is to connect each data cell to all attribute cells that characterize it, thus grouping those cells that have to be read conjointly. The groups formed constitute each data cell's reading path. After each has been identified, the output can be stored in a relational table, which can be utilized for information extraction.

Hurst (2000). Each data cell is connected with the first attribute cell above and next to it. Semantic webs are built on the content of attribute cells. Heuristics attempt the connection of each attribute cell to those above or aside it.

Chen (2000). Pairs of attribute/data cells are found independently within each sub-table identified; each data cell is connected to the attribute cells above and/or next to it; if no attribute line or column exist, each cell is connected to the first cell in its column. In the result, one data cell can have more than one attribute and one cell can be attribute and data in different pairs; in this case, transitivity properties apply.

Hu et al. (2001). For each data cell, simply find the attribute cells in its line and column.

In summary: as demonstrated in these works, a simple heuristic approach is sufficient to guarantee very good results in this task.

3.5. Interpretation

"*Interpreting and understanding a table, once categories have been established, can be characterised as finding missing information. This information can be discovered in the document itself, in models of the domain which the document's content comes from or is about, or from world knowledge*". Hurst (2000)

After the completion of the tasks outlined above, one knows how to read the information in the table, but one does not yet know what is being said (in Figure 1, for instance, we already know that "life", "wild buckwheat" and "annual" have to be read conjointly, but we do not yet know that "wild buckwheat lives for a year", much less do we know that wild buckwheat is a plant. This information is added through interpretation. A deep interpretation of the table will almost always require context specific knowledge.

Hurst (2000) identifies two major interpretation steps that can be performed before interpretation becomes context-specific.

States		E	B
q	Sequence	P	QP
	Probability	1.0	.2
r	Sequence	R	QR
	Probability	0.0	.1

Table 5: Deriving categories (presented in Hurst (2000)

One step is joining together those attribute cells that are members of one same category. For example, in the adjacent table, the categories would be {States.q, States.r}, {Sequence, Probability}, {E,B}. The author details an algorithm for deriving these categories; however different outcomes, eventually all valid, can be obtained depending on the options taken along the process.

The other step is classifying some of the relationships between cells. This author proposes several different types of relationships:

- nominal super-type, e.g. [Car, Ford]; or qualitative super-type, e.g. [Car, Red]
- partitive, e.g. [Car, Wheel]
- units of measure [Turnover, EUR]
- quantitative [Turnover, 1 million]

Some steps are suggested to achieve a tentative classification of relationships, based on the semantic types of the contents (dates, numbers, units of measure, years), as well as the comparison of the table's contents with the rest of the text in the document and the presence of certain keywords.

Ferguson (1997) and Kornfeld and Wattecamps (1998) developed context specific solutions for the problem of extracting information from American companies' financial statements published on the internet in a poor XML-based format (the SEC filings). In this Accounting context, there are several constraints the information must fulfil, for example subtotals are often presented that must equal the addition of its parcels. If these constraints are not satisfied either the table contained a mistake to begin with (which was measured to happen in only 2% of the cases), or the information was wrongly extracted. Therefore, these authors rely on the satisfaction of these arithmetic rules and, in the case of Kornfeld and Wattecamps (1998), also on indentation to identify partitive relationships between the contents of cells. With these, a hierarchical tree can be built with the attributes of the table. This tree is then compared word by word with a manually built collection of all possible alternatives to name the concepts they wish to extract. The top of the tree is compared first and thus restricts the search space of the hierarchically lower members.

Ferguson (1997) further guarantees the accuracy of the information extracted by: comparing the result of the extraction of items that appear in different tables (e.g. profit or loss for the financial year); verifying that certain ratios built with the information extracted have reasonable values; inferring the unit of measure of a table from comparison with the scale of the items presented in other tables where the unit was identified. By ranking mechanism of the different tables in the document according to their likelihood of being one of those they wish to extract, the method can proceed to analyse a second best choice if the first choice failed to comply with the restrictions of the context. Awareness to titles is also an important asset.

In summary: few authors have actually targeted the interpretation task. However, those who have: have either avoided committing themselves to a specific context and hence not exploited the full potential of interpretation (Hurst (2000)); or have taken interpretation to deep levels (including in terms of the guarantee it can give of the quality of the results achieved) but to do so have made their systems dysfunctional in any context other than their own limited one (to use Ferguson's (1997) or Kornfeld and Wattecamps (1998) method on financial statements that do not abide North American accounting disclosure practices, great adaptations would be necessary!).

Unsolved issue: no author has yet found a way of making interpretation general, i.e. fully interpreting a table and being able to do so in any or many contexts.

4. Evaluating table processing systems

As seen in section 3, table-processing methods suggested by different authors have had different targets. One of the main difficulties in comparing their performance is that they deal with so many subtasks that one sole measure cannot provide a full-length evaluation. As an example, consider an end-to-end table processing method that aims at extracting information from tables (e.g. our method). Obvious performance measure would be whether the extracted information actually is the required (this measurement was actually used by Ferguson (1997) and Kornfeld and Wattecamps (1998) and how much time the process takes. Such *end-result evaluation* is fundamental as overall performance often depends on the interaction between the different steps rather than on the performance at each individual step.

However, we need to evaluate the method's quality at each of its intermediate tasks as well. This allows us to refine each of the steps independently and to compare their achievements with those of other authors. There are nonetheless difficulties in comparing the results of different authors, which we encapsulate under the heading Diversity.

- *Diverse document types*: Take for instance the location task, there are different issues involved in locating tables within images, text or poorly tagged files, as different features with different efficacies are available. For example an ASCII input does not hold any tags pointing to potential tables or differences in font size to hint concept hierarchy; and even if it is OCR error free, it is subject to other ambiguities such as misalignment (Hurst 2003); as such, it can be considered a poorer input type. On the other hand, a table location competition that does not allow several different types of outputs, e.g. the coordinates of the four edges of the table to be expressed in terms of number of ASCII characters of a given type to the edges of the page as well as the number of pixels, would probably exclude some otherwise interested competitors.

- *Diverse test domains*: Domains exist where tables follow relatively stable styles and where pages are less likely to contain diverse document elements such as graphics or organizational charts. This would be the case of scientific journals and newspaper databases, which often follow predefined layout conventions, or law books. Systems reaching good performances in such stable environments are often feeble outside of their protected atmospheres (as Tupaj et al. (1996) well noticed). This is closely connected to the fact that a representation of the heterogeneity of the tables that exist in the Universe and throughout history and how likely each type is has not been obtained (yet). This is a common issue when the data under analysis is a human creation (closely connected with human imagination and art and even history). Against such a representation authors could try their methods and know for which type or types they are (and are not) more appropriate. Until then, we are bounded to academic competitions.

- *Diverse table definitions*: There are examples of table-like portions of documents that different users might classify differently, as we saw in Section 2; Hu et al. (2001b) provide several cases where "there is significant disagreement between ground-truthers" and explore the reasons for it. To address this problem, in Hu et al (2000) ground truth has been generated by all four authors: "A line was classified as table or non-table if three or four votes (out of four) classified it in that way".

- *Diverse measurement levels*: Recall (degree of correct identification with respect to the total number of target elements), precision (degree of correct identification with respect to the total number of identified elements), and their geometric mean (the F-measure) have been the most common performance measures used by different authors. These measures have nonetheless been taken at different levels of the table's physical model: Pyreddi and Croft's (1997) measured them in terms of the number of lines correctly tagged; Wang et al. (2000-2002) took the same measurements on cells; Tupaj et al (1996) and Chen et al (2000) consider full tables as well; and Ng et al (1999) take full table, column and row.

4.1. Metrics for segmentation

Segmentation is a particularly intricate task to evaluate, because unlike location and functional analysis, which basically involve classifying a given element into 2 or more categories, segmentation involves “cutting” or “gluing” a level of elements to form another, so the number and type of the elements at the output of the task can be quite different from what you started with.

Hurst (2003) proposes new measurement levels for evaluating the ability of the segmentation task in deriving the table’s columns and lines. In the ground truth generation stage, each cell is linked with those with which it has vertical and horizontal adjacency, while respecting some constraints, to form columns and row. Recall and precision are measured on the system’s ability to accurately detect those cell-to-cell links. However we believe the use of this metric alone may be misleading of a method’s performance since an error made in the assignment of one cell to its column may compromise the interpretation of the entire column and ultimately of the table, particularly if the mistake separates data cells from their attributes.

Hurst proposes another complementary measurement for evaluating segmentation, which seems to aim at this gap. The “table recognition evaluation” considers as an “omission […] those cases where there is a logical connection between two cells but they are not aligned”. However, the method seems to require the ground-truthing of the functional and structural models of the table to evaluate the accuracy of its physical model. Moreover, it also takes the measurement too close to cell level, which has the effect of diluting the errors made in one or two among many correctly identified cells.

We believe that errors made on cell-to-cell links should propagate to their entire row or column and precision and recall should be measured on the accurately detected link threads that traverse the whole table vertically or horizontally, i.e. its columns and rows. Basically, we believe that in the gluing and splitting of different elements to form others two main types of errors can occur: either two (or more) different elements are glued together so the detected element is *impure*, or one individual element is split in two (or more), and thus the detected elements are *incomplete*. This reasoning can be applied to both column, row and cell detection.

Below we propose some metrics that are complementary to Hurst’s “proto-link evaluation”; one can calculate these measurements with the sheer knowledge of the table’s physical model (that any HTML provides). To illustrate our proposed metrics, we will use the example in Figure 7 that was presented in Hurst (2003). There are 16 rows and 6 columns.

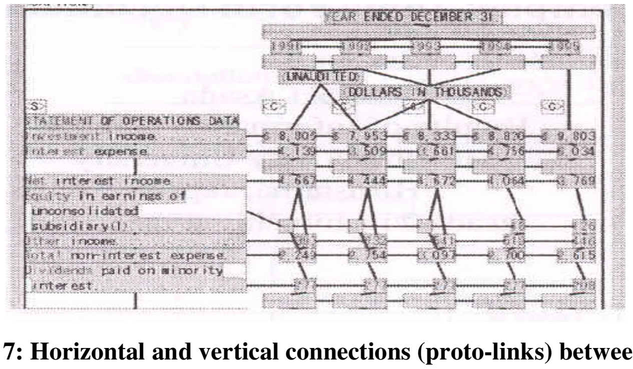


Figure 7: Horizontal and vertical connections (proto-links) between cells.

In Table 6 we present the resulting contingency table. It is somewhat particular: the total number of elements being qualified (25) differs from the total number of real elements in the table (22); this number in particular does not show on the contingency table, but still should be presented *pro memoria*. The difference between the total number of detected elements and the total number of real elements, if positive, means ghost elements were generated (in Hurt's example there were 3 ghost columns), although exactly how many is tricky to count; if negative a greater number of elements were glued together than ghosts were created.

In the example, apart from the 16 rows, only 2 columns were completely and correctly identified, the first and the fifth, which contain all the cells they are supposed to and none they are not supposed to. 7 columns did not have all the cells that belonged to one sole true column in the real table, of which 3 were ghost columns (25 detected columns – 22 real columns) and 1 column merely lost the cell "Dollars in thousands". We call this sort of mistake a *splitting error*; the opposing, when the cells in different columns intertwine, is a *gluing error*. Gluing errors cause the resulting element to be *impure*; splitting errors cause the resulting element to be *incomplete*.

		Completeness		
		Complete	Incomplete	Total
Purity	Pure	CP = 18	PI = 7	25
	Impure	CI = 0	II = 0	0
	Total	18	7	25

Table 6: Evaluating the performance of the segmentation task (presented in Hurst (2003)). *Pro memoria*: the number of real cells is 22.

While *recall* gives as an account of a method's ability to correctly identify all the elements (and is measured in relation to the total number of existing elements), *precision* indicates how accurate the given result is (and is measured in relation to the total number of detected elements). In the example, *recall* equals 18/22=82% and *precision* 18/25=72% (smaller but more realistic than the "table recognition evaluation" and the "proto-link evaluation"). However, with these two measurements, we do not know which sort of problems occurred. Besides, because this is not a classification but rather a detection problem, the trade-off between precision and recall is lost.

We consider *completeness*, defined as the proportion of completely identified elements w.r.t the total number of real elements, to be a relevant measure. Notice that in order to be completely identified, a line (or a column) must contain all of its cells. In this case, completeness is 18/22=82% (it happens to equal recall because there are no impure columns or rows).

High completeness may be obtained at the cost of the *purity* of the detected elements, where a pure detected element is one whose cells belong to only one original element. Therefore, purity is defined as the proportion of pure detected elements w.r.t. all the detected elements. For the above example, purity is 25/25=100%. This value of 100% indicates that the error of having cells from different elements mixed on the same detected element did not occur.

If we wanted a table's columns and rows to be totally pure and cared not about completeness, we could place each cell in its own column and row, creating a large number of ghost columns and rows; inversely, if we cared only about completeness, we could throw all the cells in a single column/row, but purity would be 0%. Clearly there is a trade-off between both measurements, which is a positive aspect in evaluation measures; different costs can be attributed to the two types of mistakes; to compare different methods, an efficiency border can be defined and different methods can be preferred by users who have different utility functions. For information extraction, we consider *incompleteness* to be a far less serious problem than *impurity*.

The same measurement of errors may be applied to other examples of splitting/gluing different elements to form new elements. For example, in the task of splitting lines into cells, pure cells only have the characters of a single true cell; and complete cells hold all the elements of a given true cell. We will see an example of these metrics in action in section 5.2.1.

4.2. Metrics for other tasks

Hurst (2000) measures the results of *functional analysis* by computing performance and recall on the classification of cells as attributes or data; because attribute cells tend to be less frequent in tables but mistakes in their recognition are more serious, the author suggests normalizing the two using the relative weight of attribute and data cells in the test set. Aiming at information extraction, Ferguson (1998) takes those measurements on the number of tables that are properly handled by their extraction system; and Silva et al. (2003) on the number of lines that belong to those tables holding the information to be extracted.

Apart from these more standard measurements to evaluate *location* Hu et al. (2000) and Cesarini et al. (2002) take "a measure of the similarity of two documents (the recognised document and its ground truth) in terms of their table structure". Both measures are positively correlated with the total areas that are correctly identified as tables and negatively correlated with the total area of non-tables that is identified. However, being similar to Hu et al.'s *teval*, Cesarini et al's Table Evaluation Index does not associate costs to the different possible errors in table detection; therefore, when used in the table location competition proposed in ICDAR 2003 it was complemented with the number of found, missed, split, merged and false tables detected.

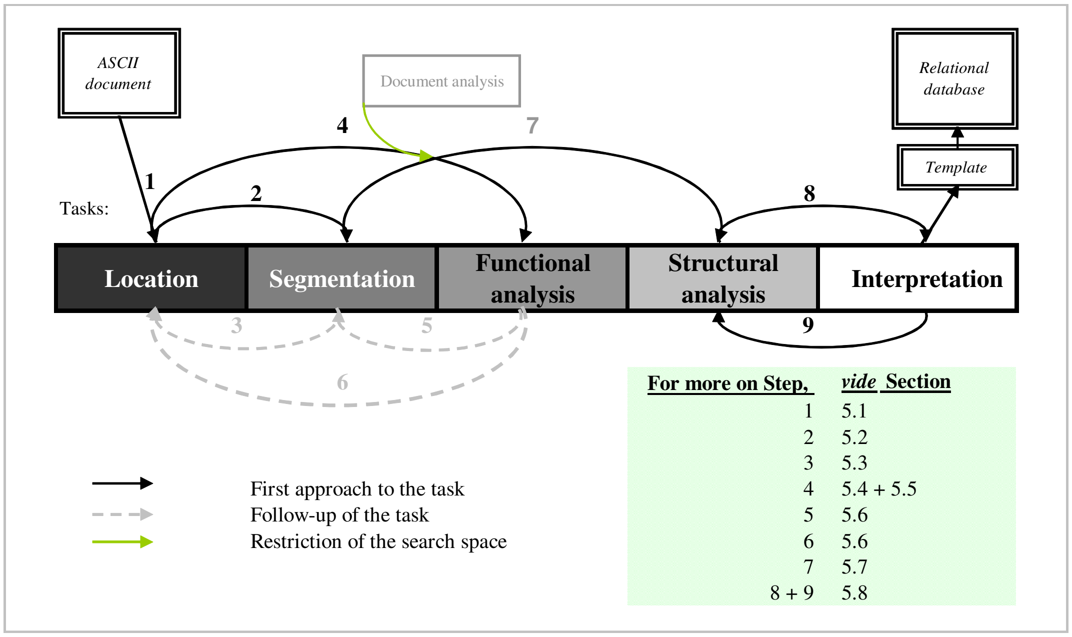
Hu et al. (2001) also proposed an evaluation model that is resistant to the difficulties in ground-truthing tables and that can be used on other table related tasks. A graph model of the table is the output of their table analysis method; in parallel, a ground truth graph model of each table is manually created. Three classes of questions are asked to both graphs and the percentage of agreement is measured. The first class of questions aims at evaluating the quality of the *physical model* derived by the method with questions like "how many columns does the table contain?"; the second set evaluates the tables *functional model*, e.g. "how many attributes cells contain the word *Open*?"; the third set aims at the *structural model* and mimics database-type queries. Wang et al. (2002) propose to combine three measures that are similar to Hu et al's by weighing them by the costs of the errors made in the graphical, functional and physical models of the table. Hurst (2003) calls this sort of approach "functional" in opposition to the "absolute" approach followed in section 4.1.

5. Our method

"*Table recognition […] may be understood as sequences of decisions (inferences) supported by observations and transformations of available data. These decisions are made relative to a table model*".

Zanibbi et al. (2003)

Overview: We are currently developing an integrated method to allow extracting information from tables contained in ASCII documents. We are applying it within the context of companies' financial statements. While some financial statements are rather plain, others are characterized by high heterogeneity of the layout styles used by authors to display information on page and can contain several diverse elements, such as graphics and organizational charts. For examples, please refer to Auto Industrial (2000) and EDP (2000). Our aim is to extract from them key financial information that is usually presented in tables. This information obeys certain predefined arithmetic rules imposed by its accounting context, which is often country-specific, that we want to integrate in order to assert and improve the quality of the extracted information.



For more on Step,	*vide* Section
1	5.1
2	5.2
3	5.3
4	5.4 + 5.5
5	5.6
6	5.6
7	5.7
8 + 9	5.8

Figure 8: Method's schema (the numbers represent steps)

The method, which was first presented in Silva (2003) and which schema is shown in Figure 8, takes as input an ASCII document and locates within it the existing tables (Silva et al. 2003), segments these into cells, distinguishes the cells that describe the content of the table and relates them to those that contain the data; finally it interprets the results to identify the relevant items and extracts these to a database. As we said in section 3.5, full interpretation is almost always context-specific. However, we aim at "contextualising" the method in a "generalist" way, by letting the user plug-in and out the relevant knowledge base. For accounting, XBRL (eXtensible Business Reporting Language) jurisdictional taxonomies are an excellent encoding of each area's national accounting environment (Silva et al. 2004).

Input: we want our method to be as input free as possible, so we allow the treatment of any sort of untagged document, that we simply convert to ASCII[1], although in the future we would like to exploit the extra information image files contain, namely font information.

Process: the method does not work linearly, the different tasks being inter-connected such that the information gathered at each step is used to improve the decision of the previous steps. This contributes to: a) increasing the confidence of the decisions taken, all the way to the final result; b) revisiting previous decisions under the light of new information, allowing the method to correct its own mistakes; c) reducing the search space of subsequent (eventually more complex) tasks by restricting them to portions of the document that are most likely to contain the information to be extracted. To facilitate the improvement of the decisions, flags are left behind every time a decision seems risky. Each flag signals a decision that could be the subject of especially crafted data induced classification methods; for each a likelihood of accuracy can be measured.

The knowledge base is supplied, it can be custom made or adapted from existing ontologies, and will be a plug in to the method, such that: if absent, it will still be able to obtain the output, eventually with smaller confidence levels; if changed, it will interpret the results under the light of the new ontology. To this end, we will construct the different steps of the process in the most general way possible. If the method finds an item that does not abide the constraints that apply to it under the knowledge base, it will move on to the next best candidate; the output the best result found.

The method should also be user-friendly: if the confidence in a result is too low, the user shall be alerted and shown the location of the best result in the original document; if no result is obtained at all, the user will be given a choice to supply the location of the table that contains the item, if any; however, if confidence is high enough, the method will not require user intervention. Any parameters the system encloses should be accessible to the user, so that he can personalize the performance of the method, which is even more important when no knowledge base is available.

Output: the output of the method will be a set of (eventually interrelated) items chosen by the user and extracted from the document. These will satisfy, to an adequate extent, the constraints the knowledge base imposes in terms of its relationship to other items in the document. The information can then be extracted to a database (for diverse analysis purposes as detailed in Silva and Ramos (2004)); or converted to the terms of the ontology (in the accounting context, items can, for example, be converted to XBRL, for a more user-friendly downloadable transmission through the internet).

In the remainder of this section we will describe the main steps of the schema above. We will not describe in detail the plans for the five basic table-related tasks, to which we will apply a combination of the methods described in the bibliographical analysis.

[1] One of the reasons for starting with ASCII files is that, although electronic filing of financial statements is a reality that has already reached companies – in USA public companies make their information available to the Security Exchange Commission (SEC) in ASCII, HTML or a shallow XML and in Europe electronic filing will have to be accepted by public entities from 2007 onwards, under EPC (2003), and although sooner or later it will reach the majority of companies around the Globe, for the time being the format those filings will take is not known. Only one thing is certain, they will contain text, numbers and tables, and all three are preserved in ASCII documents.

Rather we will focus on the possible inter-connections between these tasks, trying to show how we can benefit from the information gathered in each to a) improve the quality of the remaining, and b) work on the unsolved issues identified in sections 3.1, 3.2, 3.3, and 3.5.

5.1. Location I

Goal: skim the document to identify those lines that are more likely to belong to tables, so as to considerably reduce the search space for the subsequent more computer intensive tasks, while keeping the large majority of table lines.

Input: the ASCII is imported into a relational table where each record is a full line of the original document; in the experimental phase for this step, we have worked with 19 pdf files downloaded from the web and converted into ASCII using a pdftotxt utility; our resulting training set had 87,843 lines.

Process: the process followed in location I has been detailed in Silva et al. (2003b). It begins by analysing, through some basic descriptive statistical measurements, the distribution within the document of the number of interior spaces per line. From these, it estimates a threshold for pre-classifying each line as being in a table or not. In those pages with a high enough percentage of above-threshold lines, the presence of *n* close enough such lines determines their neighbourhood to be inspected. The size of the neighbourhood is measured in terms of the number of non-empty lines between each two lines (in this context it is customary to have large gaps of empty lines in the middle of a table) and is a function of the percentage of above-threshold lines in each page; the function however can be manipulated by the user. Table lines are simply those with any white space or that are narrow. Each individual table area is assigned an ID number (the existence for instance of a relatively long text line with no interior spaces causes the identifier to change).

Overview: through this process, the decision on each line depends on the perceived specific characteristics of its source document, of its page, and of the lines that surround it; it is this dependency that makes the method capable of adapting its performance in those documents with more varied error-prone layouts (as EDP 2001 is).

Output: individualized table candidate areas, where each record is a line of the document.

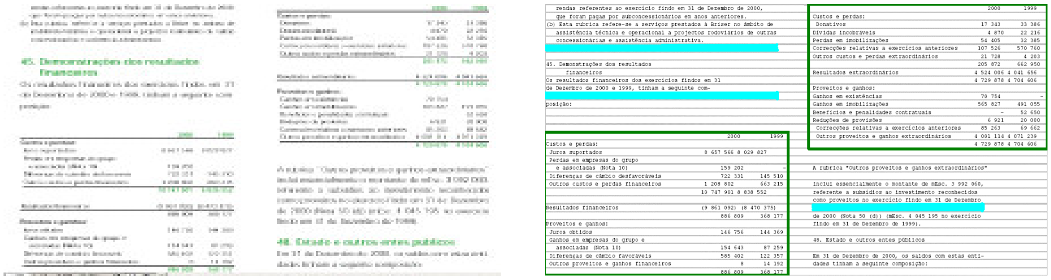
Evaluation: we have conducted a two-fold evaluation. On the one hand, setting the parameters to aim at capturing all sorts of tables, we have measured the total number of document lines discarded (*economy*) *versus* the total number of lines accurately kept (*recall*), both presented as percentages of the total number of lines in the document. On the other hand, because our final goal is to extract information, we have taken those same measurements on the total number of relevant lines, after adjusting the parameters to the tables we wish to extract, i.e. on the total number of lines that belong to tables that contain information we wish to extract.

Results: empirical evaluation was made on the training data and indicates that more than 95% of all table lines (and 99% of those table lines that contain information to be extracted) are selected after discarding on average 62% (70%) of them. All relevant tables are at least partially detected. User manipulation of parameters was necessary for one table containing one column with very long vertically spanning cells. Evaluation was also taken on the unseen data, a set of 3 financial statements with 9,470 lines; *recall* was always above 99% and *economy* was between 70% and 79% in the context specific evaluation and 58%, 67, 68% in the generic evaluation.

Errors: in an end-to-end method, it is often not as important to know how many records you got right but rather whether the records you got wrong can be corrected later on. To this end, we try to typify the mistakes made and plan the solution for them.

False positives; the end of paragraphs was sometimes mistaken for the beginning of a subsequent table and headings were sometimes mistaken with the end of previous tables - we will solve this in section 5.3; two columns of text separated by an usually large (for the document) column separator organizational charts or graphics with legends (on the dots or axis) were also mistaken for a table- to be solved in section 5.3;

True negatives; the last lines of some tables and the first header lines of others can be missed as well portions of the table which have too narrowly filled-in - we will solve this in section 5.3; very narrow tables or tables with less than 3 lines can be totally missed - this sort of table is not commonly relevant in the accounting context, for now we will live with this disadvantage; tables with very different font sizes may be problematic (to be solved in future work, when image files are integrated into the method), small differences in the fonts used on a page however are handled, especially if all the table itself has similar or the same font – see Figure 9.



```
   rendas referentes ao exercício findo em 31 de Dezembro de 2000,
   que foram pagas por subconcessionários em anos anteriores.
(b) Esta rubrica refere-se a serviços prestados à Briser no âmbito de
   assistência técnica e operacional a projectos rodoviários de outras
   concessionárias e assistência administrativa.


45. Demonstrações dos resultados
      financeiros
Os resultados financeiros dos exercícios findos em 31
de Dezembro de 2000 e 1999, tinham a seguinte com-

posição:



                                               2000          1999
Custos e perdas:
 Juros suportados                         8 657 566 8 029 827
 Perdas em empresas do grupo
   e associadas (Nota 10)                      159 202            -
 Diferenças de câmbio desfavoráveis            722 331      145 510
 Outros custos e perdas financeiros       1 208 802         663 215
                                         10 747 901 8 838 552

Resultados financeiros                   (9 861 092) (8 470 375)
                                               886 809      368 177
Proveitos e ganhos:
 Juros obtidos                                 146 756      144 369
 Ganhos em empresas do grupo e
   associadas (Nota 10)                        154 643       87 259
 Diferenças de câmbio favoráveis               585 402      122 357
 Outros proveitos e ganhos financeiros               8       14 192
                                               886 809      368 177
```

```
                                                       2000         1999
Custos e perdas:
 Donativos                                           17 343       33 386
Dívidas incobráveis                                   4 870       22 216
Perdas em imobilizações                              54 405       32 385
Correcções relativas a exercícios anteriores        107 526      570 760
Outros custos e perdas extraordinários               21 728        4 203
                                                    205 872      662 950
Resultados extraordinários                        4 524 006 4 041 656
                                                  4 729 878 4 704 606
Proveitos e ganhos:
Ganhos em existências                                70 754            -
Ganhos em imobilizações                             565 827      491 055
Benefícios e penalidades contratuais                      -       52 650
Reduções de provisões                                 6 921       20 000
 Correcções relativas a exercícios anteriores        85 263       69 662
 Outros proveitos e ganhos extraordinários        4 001 114 4 071 239
                                                  4 729 878 4 704 606


A rubrica "Outros proveitos e ganhos extraordinários"

inclui essencialmente o montante de mEsc. 3 992 060,
referente a subsídios ao investimento reconhecidos
como proveitos no exercício findo em 31 de Dezembro

de 2000 (Nota 50 (d)) (mEsc. 4 045 195 no exercício
findo em 31 de Dezembro de 1999).

48. Estado e outros entes públicos

Em 31 de Dezembro de 2000, os saldos com estas enti-
dades tinham a seguinte composição:
```

Figure 9: A page with different font size. The image on the left is an extract of a pdf financial statement; on the right is the same after its conversion to ASCII and insertion into a relational table. As can be seen, the tables have smaller font sizes than the text. During the conversion to ASCII, the horizontal alignment of the portions in smaller font is preserved and in the remaining extra lines, that we paint in blue above, are added to make up for the extra space. All lines of the two tables were detected.

Other aspects; since this step only classifies lines and not columns as likely tables, non-table areas that are in horizontal alignment with tables are still present; this aspect will be solved in step 5.3. No more can be obtained without transforming the data, i.e., moving to a different table model. On the other hand, tables that are not separated by at least one long line of text will have been merged, which we will solve in section 5.3 or 5.4.

5.2. Segmentation I

Goal: the purpose of this step is to identify and delimit the cells in the tables areas obtained in section 5.1 and assign them to their correct beginning and end columns. We believe we should find cells before we try to find columns so as to avoid an accidental space alignment resulting in the detection of false columns. Therefore, we will divide this step in two sub-steps.

5.2.1. Segmenting lines into cells

Goal: cut each line into its respective cells.

Input: individualized table candidate areas, where each record is a line of the document.

Process: in our approach, we have adapted tokenising techniques: we search for space characters and delimit words when there are three or more spaces between them; in the case of numbers however, the presence of a single space between two digits is analysed to decide whether it is a cell delimiter or a thousands/decimal separator; in these cases, or when a single space separates a number from text (or *vice versa*), a flag is left to signal the riskier decision, so that the method can reanalyse it at posterior stages.

Overview: although we generally agree that using a language model to decide whether or not to group two words with a particular graphical arrangement into one cell is a more general approach (Hurst (2001)), we suspect it is computationally more expensive than simply taking a graphical approach. This aspect is relevant to us since our final aim is to extract information from specific tables in long documents that contain several other tables, most of which hold not the information sought for; we want to skim the tables before scan them. On the other hand, a language model would not help much when segmenting numbers into cells and numeric tables not only account for the majority of tables in the accounting context but also for a good part of tables in general. Luckily, finding a number is a rather deterministic problem, which a carefully crafted set of rules can solve quite well.

We intend to measure the benefit of applying now Hurst's (2001) approach to cell segmentation, properly adjusted to handle numbers, against saving it for the text cells in section 5.5 and the few problematic cases identified in section 5.2.2 and using now a more simplistic approach.

Output: each cell is horizontally segmented the start and end positions of its content have been determined, tentative column placement may be added if we simply accumulate the number of cells found per line.

Evaluation: we will measure purity and completeness, the evaluation metrics presented in section 4.1, on our ability to clip lines into cells. For completeness, we will also show precision and recall. We will again make a two-fold evaluation; we will measure the percentage of cells that were correctly segmented in tables that hold numeric data and the same in tables that also hold text.

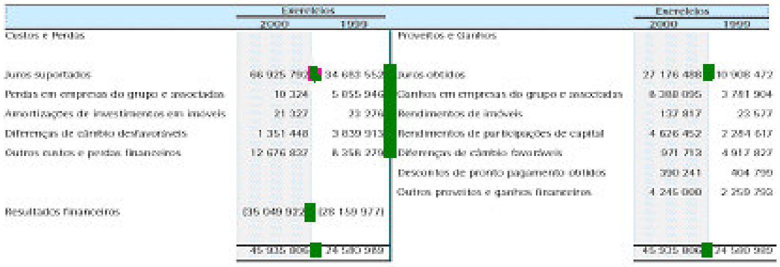
Results: due to time limitations, we have taken a small sample of examples, all deriving from the document we have had more trouble treating during Location I. The document contained 104 tables, of which 84 presented mostly numbers. After demarking the vertical delimiters of the table by hand, we have applied our segmentation method to it and then corrected all mistakes by hand.

		Completeness						
		Textual tables		Numeric tables		All tables		
		Complete	Incomplete	Complete	Incomplete	Complete	Incomplete	Total
Purity	Pure	CP = 2896	PI = 0	CP = 4642	PI = 38	7538	38	7576
	Impure	CI = 205	II = 0	CI = 10	II = 0	215	0	215
	Total	3101	0	4642	38	7753	38	7791
Purity		93.4%		99.8%		97.2%		
Completeness		100.0%		99.6%		99.8%		
Precision		82.0%		98.2%		91.3%		
Recall		93.4%		99.0%		96.8%		

Table 7: Evaluating the performance of the segmentation task – detecting cells. ***Pro memoria*: the number of real cells is 8259, of which 3530 in textual tables and 4729 in numeric tables.**

Splitting errors in numeric tables occurred when polynomials were presented, of the sorts of "15 + 19"; and one happened when by mistake the author left out the thousand delimiter dot in 1 584.786,95 (under the continental number representation convention).

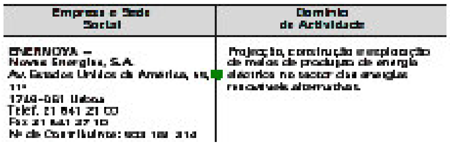
<u>Errors</u>: this rule based procedure has been able to work well in examples of tricky tables, such as Figure 10, where the pdftotxt converter turned to one space the separators between the numbers in the first and last two lines of data and between columns 3 and 4. Although space is also used as the thousands delimiter, the rule managed to accurately segment the cells horizontally. The divider between the third and the forth columns was obviously not identified, but a flag signalled it.



	Exercícios			Exercícios	
	2000	1999		2000	1999
Custos e Perdas			Proveitos e Ganhos		
Juros suportados	66 925 792	34 683 552	Juros obtidos	27 176 488	10 908 472
Perdas em empresas do grupo e associadas	10 324	5 055 946	Ganhos em empresas do grupo e associadas	8 388 095	3 781 904
Amortizações de investimentos em imóveis	21 327	33 276	Rendimentos de imóveis	137 817	23 677
Diferenças de câmbio desfavoráveis	1 351 448	3 839 913	Rendimentos de participações de capital	4 626 452	2 284 617
Outros custos e perdas financeiros	12 676 837	8 356 279	Diferenças de câmbio favoráveis	971 713	4 917 827
			Descontos de pronto pagamento obtidos	390 241	404 799
			Outros proveitos e ganhos financeiros	4 245 000	2 250 792
Resultados financeiros	(35 049 922)	(28 159 977)			
	45 935 806	24 580 989		45 935 806	24 580 989

Figure 10: A table segmentation problem - a single space serves as a thousands delimiter, a word delimiter and a column delimiter (in the places painted in dark green) in one same table. 46 cells were detected, all cells had all the characters of each real cell; 5 cells had characters of 2 different real cells each. Purity: 41/46=89%. Completeness: 46/46=100%; Precision=41/51; Recall=41/46.

Notice that it is arguable whether a transition between different content types should result in the identification of a new cell or merely a signalling flag. The first alternative would actually solve the only segmentation errors in Figure 10 and would probably be more prudent in the accounting context, but it would produce bad results in the first column of Figure 11. Obviously, a careful evaluation of this aspect will be made, but the answer may differ from context to context. In our opinion however the issue is often not how many mistakes are made, but which and whether the method can in later stages solve them, for which flagging is the first step. Figure 11's only segmentation error was that the cell ending with "55" was merged together with the one starting with "eléctrica". This aspect will be solved in section 5.2.2.



Empresa e Sede Social	Domínio de Actividade
ENERNOVA – Novas Energias, S.A. Av. Estados Unidos da América, 55, 11º 1749-061 Lisboa Telef. 21 841 21 00 Fax 21 841 27 10 Nº de Contribuinte: 503 180 314	Projecção, construção e exploração de meios de produção de energia eléctrica no sector das energias renováveis alternativas.

Figure 11: A segmentation problem - an extract of a textual table. Of the 15 detected cells, of which one holds characters of more than one true cell. Purity=14/15=93%. Completeness=100%, since no splitting errors occur; Precision=14/16; Recall=14/15.

5.2.2. Grouping cells into columns

Goal: attribute each cell to its respective beginning and end column while correcting eventual cell segmentation errors.

Overview: our first idea for identifying the beginning and end positions of table columns was to assume that the maximum number of cells per line indicates the number of columns in the table; then we would take the lines with most cells, and estimate the column delimiters from the position of the contents of their cells. The chart below shows in dark blue the minimum, maximum and median start position of the cells in the lines with a maximum number of cells and in light grey the same measurements taken from hand-marking all the cells in a small sample of examples.

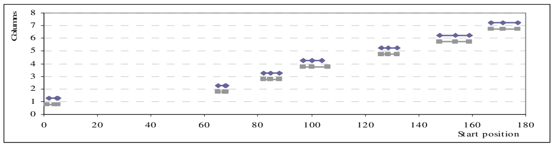


Figure 12: One idea for identifying columns. The start positions of all the cells in the table are similar to the start positions of the cells in the lines with the maximum number of cells.

This simple solution assumes that at least a few lines of the table have all cells filled in; a few heuristics can be used to correct some of the possible mistakes of this assumption. It can work well in contexts with uniform tables, such as that of journals or books and some industrial applications (Baum et al. (2003)), as it would in Figure 11 and Figure 12. However, these are only a portion of existing tables. It is true that the portion is not small and it is also true the approach is general enough to accommodate spanning cells, but in the accounting context several examples can be found where it is not general enough, as it is not for Figure 11 or Table 3: (since no lines have four filled-in columns).

Process: after_segmenting the table into cells we will apply a method similar to Hurst's (2003) except we will only be linking together those cells that are vertically aligned, not horizontally. We will only segment rows in section 5.5.1.

As such, cells that overlap vertically will be assigned to the same column, but the respect for some constraints will be guaranteed here when dealing with ambiguous overlaps, which may lead to the disregard of a vertical overlap, the merging of cells together but also the splitting of a cell that was flagged in the previous step. Thus, in Figure 10 after noticing that the third column (where the numbers got merged with the text in the fourth column) is quite wide and spans over the contents of the header columns which in turn are not vertically aligned with each other; and/or when noticing that there is an accumulation of flags alerting to the change from number to text, and that all these flags were left at approximately the same position; all of these aspects will lead to questioning the decision made before, eventually to splitting the cells into two columns and to validating the constraints again. In Figure 11, the knowing that these are two textual columns and that the method for segmenting textual cells into columns is more error-prone and the knowing that there is only one spanning cell while all the others are well segmented apart will lead to wondering whether these two should also be split. We may consider validating this decision in the light of a language model. For each of these more ambiguous decisions, a flag will be signalled so that a cell that is attributed to a given column but could also belong in another is properly identified.

We will also adapt this step to characterize column alignment by measuring the percentage of cells that share (or have close) beginning, middle or end position (using a metric similar to the one applied by Tersteegen and Wenzel (1998)).

Output: the potential table areas have been segmented into cells and these were attributed to their lines and columns, i.e. their contents have been situated.

Evaluation: we will measure our ability to group cells in one same column using precision and recall both at cell level, as suggested by Hurst (2003), and at column level, by measuring recall and precision on the total number of proto-links crossing the whole table that were correctly identified (section 4). We consider more costly to assign a cell to a column it does not belong to than to split columns, creating ghost columns that can be rejoined later. Cells spanning several columns have been partially detected.

Errors: cells which content reaches less far than their interpretation applies to will be dealt with in section 5.5.2.

Other aspects; Ii areas where two vertically adjacent tables or for instance a table and a chart were not separated yet, the number of columns for the area will equal or be bigger than the total number of columns of the area with more real columns; and in the area with less real columns, more spanning cells will be found. This aspect will be solved in 5.3.2.

5.3. Location II

Goal: with the current partial physical model of the table, we want to remove the remaining non-table cells/columns/rows from the process and recapture any lines that were improperly lost after Location I.

5.3.1. Improving the identification of table areas' vertical delimiters

Goal: improve the identification of the table's vertical delimiters.

Overview: after Location I, sometimes false positives, i.e. non-table lines that were wrongly detected, were simply narrowly filled-in lines adjacent to real table lines, such as headings or a page's header or footer (Chao (2003) has an interesting approach for identification of recurrent headers and footers and other background patterns, which we will consider in future work).

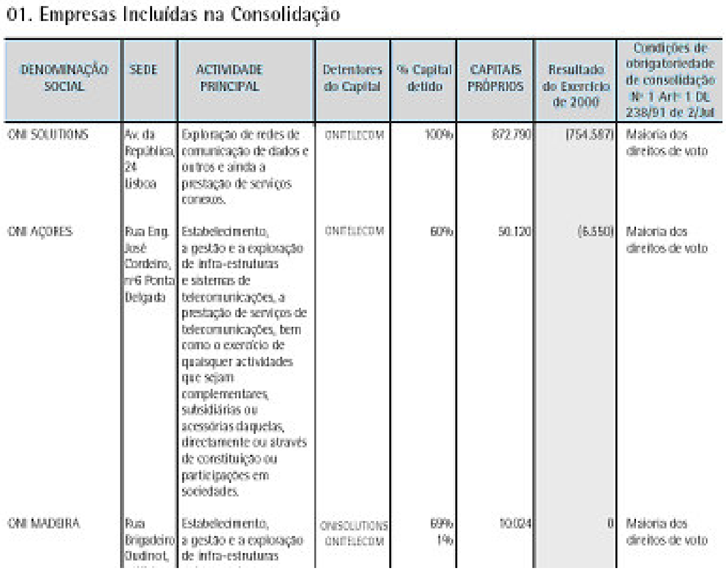
On the other hand, false negatives, i.e. true but undetected table lines, belonged to tables that were only partially detected because they had a set of relatively narrow lines. This can happen in header rows, which are known for not having the first and often wider column filled in, but also in the middle of true tables (as is the case in Figure 13); in this later case, splitting errors are likely to occur. The consequences of these mistakes may be rather serious in what interpretation is concerned.

Authors like Klein et al. (2001) and Rammel (2002) have noted that good table locators should be sensitive to a wide array of aspects, since tables make themselves noticeable to readers in more ways than one and not all occur simultaneously. In section 5.1, we chose as table candidates those lines with more contiguous interior spaces than an adaptable threshold; around these candidates we defined a neighbourhood in which we applied to each line a simple classification rule; the lines in this neighbourhood that were not candidates originally are considered more error prone than the remainder of the table. Also, because of the simplicity of the rule, the neighbourhood did not expand to include other lines, even if all the lines in the neighbourhood were classified affirmatively. Lines around detected areas are also considered more error-prone.

The idea of revisiting the lines around detected tables is not new, authors like Hu et al.(2000) and Wang et al. (2002) follow a similar approach.

Process: we will inspect the first and last lines of each detected table area to see if they obey the regular column structure detected for the table (in section 5.2). If these validate positively, then we will expand the neighbourhood and continue to validate new lines until the alignment configuration is lost.

Do notice however that header rows are more likely to have spanning cells. We may develop a machine learning approach to help us decide what to do when normal table alignment is only partially respected; the approach would consider how many column delimiters are lost and the position of the potential table line within the detected table areas.



01. Empresas Incluídas na Consolidação

DENOMINAÇÃO SOCIAL	SEDE	ACTIVIDADE PRINCIPAL	Detentores do Capital	% Capital detido	CAPITAIS PRÓPRIOS	Resultado do Exercício de 2000	Condições de obrigatoriedade de consolidação Nº 1 Art.º 1 DL 238/91 de 2/Jul
ONI SOLUTIONS	Av. da República, 24 Lisboa	Exploração de redes de comunicação de dados e outros e ainda a prestação de serviços conexos.	ONITELECOM	100%	872.790	(754.587)	Maioria dos direitos de voto
ONI AÇORES	Rua Eng. José Cordeiro, nº6 Ponta Delgada	Estabelecimento, a gestão e a exploração de infra-estruturas e sistemas de telecomunicações, a prestação de serviços de telecomunicações, bem como o exercício de quaisquer actividades que sejam complementares, subsidiárias ou acessórias daquelas, directamente ou através de constituição ou participações em sociedades.	ONITELECOM	60%	50.120	(6.550)	Maioria dos direitos de voto
ONI MADEIRA	Rua Brigadeiro Oudinot,	Estabelecimento, a gestão e a exploração de infra-estruturas	ONISOLUTIONS ONITELECOM	69% 1%	10.024	0	Maioria dos direitos de voto

Figure 13: This is an example of a table that would require user intervention in the parameterisation of Location 1 if that were an isolated step. Even if we can expect this type of table to have a small weight in the spectrum of existing real tables, we want the method to be as general as possible. Here only a few lines are simultaneously sufficiently close and wide, so splitting errors are likely. Through alignment analysis of each of the table candidates, the problem can be solved.

5.3.2. Identifying false tables

Goal: use graphical characteristics to identify areas that were mistaken for tables in section 5.1.

These include graphs with textual captions, text columns with unusually large space separators for their document. These elements may or may not have been vertically or horizontally merged with real tables.

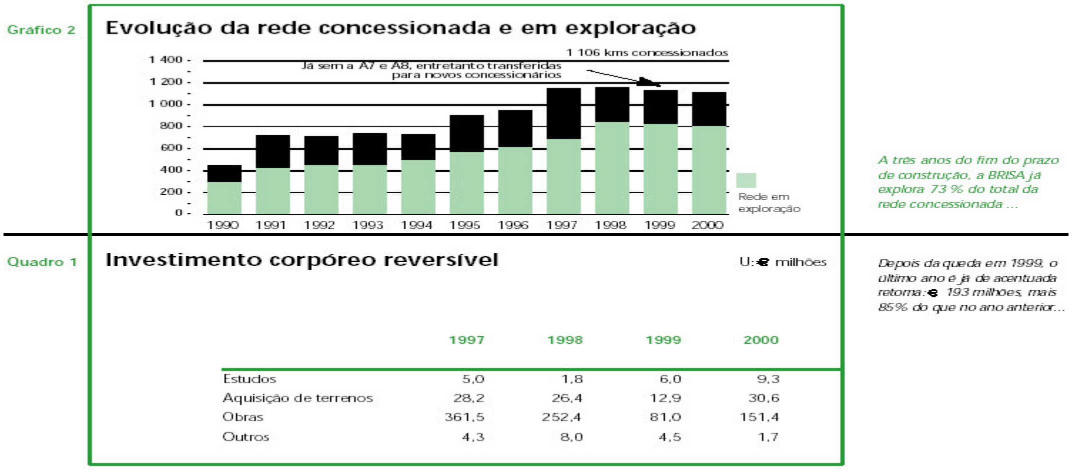


Gráfico 2

Evolução da rede concessionada e em exploração

A três anos do fim do prazo de construção, a BRISA já explora 73 % do total da rede concessionada ...

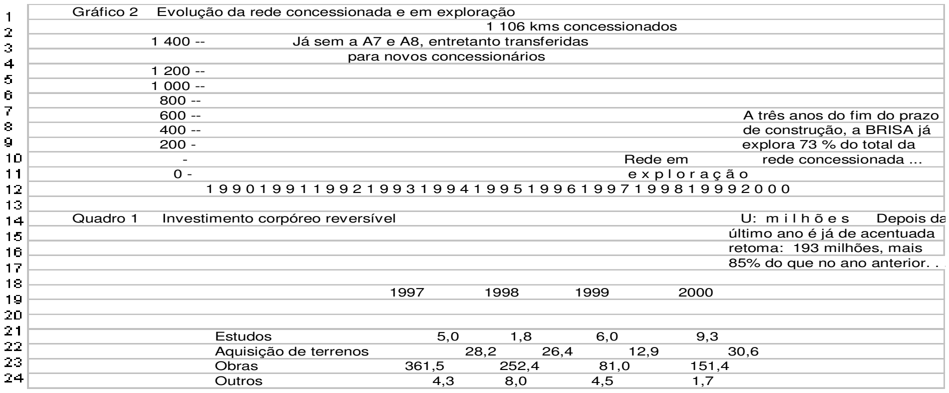
Quadro 1

Investimento corpóreo reversível

U: € milhões

Depois da queda em 1999, o último ano é já de acentuada retoma: € 193 milhões, mais 85% do que no ano anterior...

	1997	1998	1999	2000
Estudos	5,0	1,8	6,0	9,3
Aquisição de terrenos	28,2	26,4	12,9	30,6
Obras	361,5	252,4	81,0	151,4
Outros	4,3	8,0	4,5	1,7



```
1        Gráfico 2   Evolução da rede concessionada e em exploração
2                                                    1 106 kms concessionados
3                1 400 --           Já sem a A7 e A8, entretanto transferidas
4                                          para novos concessionários
5                1 200 --
6                1 000 --
7                  800 --
8                  600 --                                                                   A três anos do fim do prazo
9                  400 --                                                                   de construção, a BRISA já
10                 200 -                                                                    explora 73 % do total da
11                   -                                                      Rede em              rede concessionada ...
12                  0 -                                                      exploração
13                      19901991199219931994199519961997199819992000
14
15       Quadro 1    Investimento corpóreo reversível                                       U:  milhões     Depois da
16                                                                                         último ano é já de acentuada
17                                                                                         retoma:  193 milhões, mais
18                                                                                         85% do que no ano anterior. .
19                                          1997        1998        1999         2000
20
21
22                        Estudos                     5,0       1,8         6,0         9,3
23                        Aquisição de terrenos            28,2       26,4        12,9         30,6
24                        Obras                     361,5       252,4        81,0        151,4
                          Outros                      4,3       8,0         4,5         1,7
```

Figure 14: Two different document elements joined together as one table area during Location I; after segmentation Line 12 was considered as one big spanning cell; the graph area will have less cells filled in than the real table but more spanning cells.

Process: For this task, a combination of the methods outlined in 3.1 will be used. Wang and Hu (2002) measured the consistency of the columns and rows of each table in terms of: the content type of each cell (numerical, date, alphabetical, or other) and the content length; they also considered the average and standard deviation of the number of filled in cells per row and column, to decide whether a candidate table area was a table or not. Even if we appreciate these measures, we cannot take them over the overall detected table (this risk is mostly absent in Wang and Hu (2002) HTML inputs).

So we will derive blocks of text that are sufficiently close, much in the way that is done in Rammel (2003). We will then analyse these blocks individually to determine their likelihood of being a table block. Douglas et al (1995) do this by analysing the width of the space between two blocks *versus* each block's width – we expect that the false columns of graphics tend to have narrower "columns" and wider gaps than tables and false text columns just the opposite. We will also analyse intra block consistency in terms of relative location and alignment, as Rammel (2003) does, but also the content-type and content length as Wang and Hu (2002) do. These intra-consistency measurements will be presented to a classifier that will determine for each block how likely it is that it is a true table block.

The next step will be to join intra-consistent blocks together into tables, by analysing their row/column mapping. For Douglas et al. (1995), two blocks should be joined together if the non-empty cells in one correspond to non-empty cells in the other. Also, if a block has fewer cells filled in than those around it, it might not belong to the table all together or it might belong to a different table. We may also consider average and standard deviation of the number of columns and rows that are filled in each block and the degree of horizontal or vertical overlapping between them. Inter-consistent blocks will be joined together to form inter-consistent tables; the result may be the division of an original table area into two or more tables or the removal of non-table areas from the focus of the method.

Overview: there are many features on which decisions in Location II will be based. Balancing them and incorporating them into an appropriate classification scheme or more than one is important. We may recur to different classifiers ensembles of classifiers

Output: a set of cells assigned to their lines and columns. Virtually all non-table lines and columns have been removed. Most adjacent tables have been unmerged (when two or more tables had been mistaken for one) or un-split (when one table had been mistaken for two or more).

Evaluation: precision and recall measured on the basis of the number of cells that actually belong to tables.

Errors: two tables with no lines of text in between them and with inter consistent columns would not have been separated (e.g.Figure 15). This aspect will be solved in section 5.4.

5.4. Functional analysis

Goal: distinguish those rows/columns that contain data from those that describe data. We will assume that functional areas are rectangular.

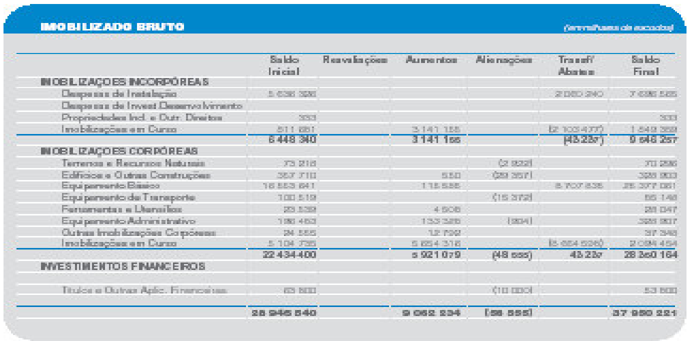
Process: a combination of the methods outlined in section 3.3 will be used. One important feature we will take into account will again be consistency measured for each of the table's columns and rows. We intend to use different classifiers and combine their results through voting or another committee based strategy. We consider it important to study data induced models for this task. We believe this step to be one of the most crucial in the entire process, since errors made in identifying table attributes will be difficult to detect in posterior phases but their consequences may condemn an entire table to be undetected or misinterpreted in the later stages.

Overview: it is only after functional analysis that the last step of location will be fulfilled. Until now, two tables with no lines of text in between them and eventually the same number of columns would not have been separated. The same would have happened with similar horizontally adjacent tables. By identifying a header line/column in the middle of such a table area, we will consider separating it into two or more tables, possibly not before (lexically) comparing the headers.

On the other hand, a table that had been inadvertently split into two vertically adjacent tables after Location I (for holding one long text-like line with no inner spaces) can be remerged, since the two parts will be considered physically similar, their distance will be small and the second part will not have a header row.

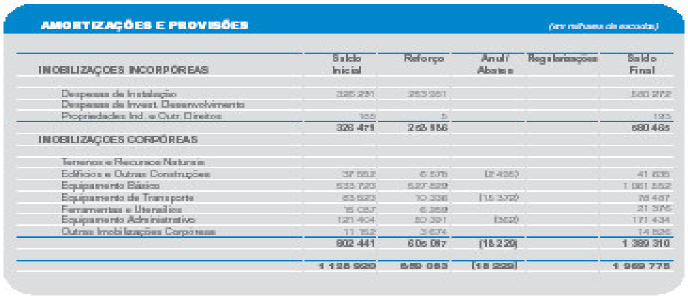
These are a common source of Location errors for many systems (as we discussed in section 3.1) that we believe functional analysis can help solve.

Output: each cell of the input is characterized in terms of its function in the table as data or attribute. Similar but independent tables have been unmerged and dependent areas remerged.



IMOBILIZADO BRUTO *(em milhares de escudos)*

	Saldo Inicial	Reavaliações	Aumentos	Alienações	Transf/ Abates	Saldo Final
IMOBILIZAÇÕES INCORPÓREAS						
Despesas de Instalação	5 636 326				2 060 240	7 696 565
Despesas de Invest.Desenvolvimento						
Propriedades Ind. e Outr. Direitos	333					333
Imobilizações em Curso	811 681		3 141 155		(2 103 477)	1 849 359
	6 448 340		**3 141 155**		**(43 237)**	**9 546 257**
IMOBILIZAÇÕES CORPÓREAS						
Terrenos e Recursos Naturais	73 218			(2 922)		70 296
Edifícios e Outras Construções	357 710		550	(29 357)		328 903
Equipamento Básico	16 553 641		115 585		8 707 835	25 377 061
Equipamento de Transporte	100 519			(15 372)		85 148
Ferramentas e Utensílios	23 539		4 508			28 047
Equipamento Administrativo	196 463		133 328	(884)		328 907
Outras Imobilizações Corpóreas	24 555		12 792			37 348
Imobilizações em Curso	5 104 735		5 654 316		(8 664 598)	2 094 454
	22 434 400		**5 921 079**	**(48 535)**	**43 237**	**28 350 164**
INVESTIMENTOS FINANCEIROS						
Títulos e Outras Aplic. Financeiras	63 500			(10 000)		53 500
	28 946 840		**9 062 234**	**(58 535)**		**37 950 221**



AMORTIZAÇÕES E PROVISÕES *(em milhares de escudos)*

	Saldo Inicial	Reforço	Anul/ Abates	Regularizações	Saldo Final
IMOBILIZAÇÕES INCORPÓREAS					
Despesas de Instalação	326 291	253 981			580 272
Despesas de Invest. Desenvolvimento					
Propriedades Ind. e Outr. Direitos	180	5			185
	326 471	**253 986**			**580 457**
IMOBILIZAÇÕES CORPÓREAS					
Terrenos e Recursos Naturais					
Edifícios e Outras Construções	37 552	6 576	(2 493)		41 635
Equipamento Básico	533 723	527 829			1 061 552
Equipamento de Transporte	83 523	10 336	(15 372)		78 487
Ferramentas e Utensílios	15 087	6 289			21 376
Equipamento Administrativo	121 404	50 391	(362)		171 434
Outras Imobilizações Corpóreas	11 152	3 674			14 826
	802 441	**605 097**	**(18 229)**		**1 389 310**
	1 128 920	**859 083**	**(18 229)**		**1 969 775**

Figure 15: Unmerging similar tables. In ASCII, how else could we know that these are really two tables without adding the results of functional analysis to the vertical distance between the two tables (since vertical distance alone is common in the interior of accounting tables)?

Evaluation: precision and recall in terms of the total number of attribute cells and data cells accurately identified; these two measures may be normalised as suggested by Hurst (2000).

On the other hand, as it is only after this step we can say Location has been complete, it is now appropriate to use either Hu et al.'s *teval*, or Cesarini et al's *Table Evaluation Index* complemented by the number of found, missed, split, merged and false tables detected, to evaluate the full-length of the location task.

5.5. Segmentation II

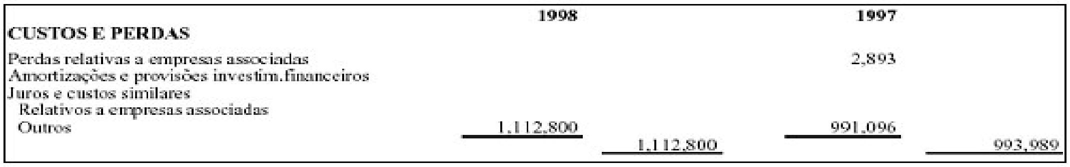
Goal: delimit spanning cells. We consider that this problem deserves different approaches for cells spanning several lines or cells spanning several columns.

5.5.1. Cells that span several lines

We distinguish two different situations in treating cells that span several lines: A) detection within data lines and B) detection within attribute lines.

Process:
A) Hu et al's (2001) simply join those lines with no data with the previous line to form the table rows. Applied in Figure 16, this simple heuristic would cause the line containing "Perdas relatives a empresas associadas" to be wrongfully joined with the following two. Besides, data cells can also span several lines. We consider Kieninger's (1998) approach to be more general: the segmentation of the table into rows will obey

the row structure of those columns which only contain one string per cell, special care being taken to first inspect "the blocks of interest […] for words that might as well have fitted at the end of a preceding line […] since modern word processors try to fill up lines as much as possible". Other cares would also be taken, attention to character type and punctuation is important. As such, the three lines in Figure 16 would not be joined together because if we added the word "Amortizações" to the previous cell, its total length would still be below the length of the widest cell in the column. In Figure 17, apart from that indicator there is also that the contents of the first two end in ":", b) they are not all written in the same type of letter.



	1998		1997	
CUSTOS E PERDAS				
Perdas relativas a empresas associadas			2,893	
Amortizações e provisões investim.financeiros				
Juros e custos similares				
Relativos a empresas associadas				
Outros	1,112,800		991,096	
		1,112,800		993,989

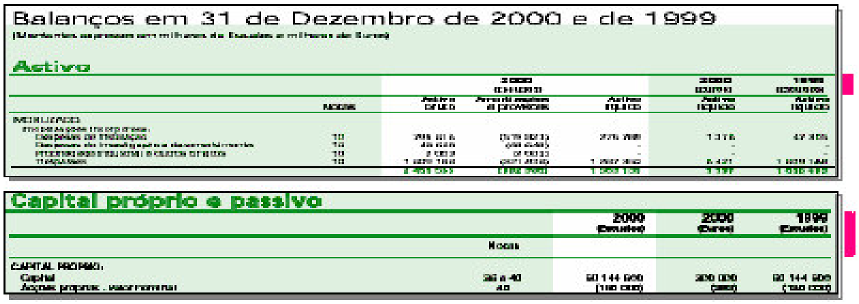
Figure 16: An interesting row segmentation problem, Auto Industrial (2000).

We will measure the results of applying this simplistic approach or using Hurst's (2001) language model to row segmentation instead. For textual tables and more generally for all tables where no columns contain only one string per cell, we will also revert to Hurst's language model.

B) In attribute lines, however we believe it is essential to develop even a simple language model of the sort of that described in Hurst's (2001) and as such combine physical characteristics with linguistic ones. Only such an approach will lead us to recognise that "(Escudos)" and "Amortizações" belong in different cells but "Amortizações e provisões" (i.e. depreciation, amortization and provision") go together.

5.5.2. Cells that span several columns

All methods mentioned in section 3.2 which cannot count on the presence of line-art or existing mark-up fail at attributing to their columns those cells which "extent only spans a subset of the values that its interpretation must be applied to" (Hurst (2003)). This problem is quite unlikely to happen in data rows but is particularly likely in attribute rows. Lets focus on Figure 17: the cells containing "2000" and "Escudos" actually also span its two adjacent columns; on the other hand, the expression "Notas" is contained in a unitary cell. A human user does not need special accounting knowledge to know this. We believe a distinction can be accomplished by combining information on column alignment and function.



Balanços em 31 de Dezembro de 2000 e de 1999

[illegible]

Activo

		2000 (Escudos)			2000 (Euros)	1999 (Escudos)
	Notas	Activo bruto	Amortizações e provisões	Activo líquido	Activo líquido	Activo líquido
IMOBILIZADO:						
Imobilizações incorpóreas:						
Despesas de instalação	10	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Despesas de investigação e desenvolvimento	10	[illegible]	[illegible]	-	-	-
Propriedade industrial e outros direitos	10	[illegible]	[illegible]	-	-	-
Trespasses	10	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
		[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Capital próprio e passivo

		2000 (Escudos)	2000 (Euros)	1999 (Escudos)
	Notas			
CAPITAL PRÓPRIO:				
Capital	36 a 40	60 144 600	300 000	60 144 600
Acções próprias - valor nominal	40	[illegible]	[illegible]	[illegible]

Figure 17: The spanning cell problem – cells that span over several columns.

Process:
1. take the attribute lines with less columns filled in than the number of data columns in the table (in Figure 17 we signal these with red);
2. For each cell in those lines, identify the content's alignment in relation to the columns its content extends over: centred, aligned on the right, aligned on the left ("2000" and "Escudos" and "Notas" are centred).
3. Expand the cells, i.e., join the filled-in cells with adjacent empty cells.
Observe that centred cells will expand symmetrically, while cells aligned to the right may span over more cells on the right than on the left and vice versa.
Stop the expansion (on one or both sides) once: a) one of the adjacent cells is filled-in, or b) belongs to an attribute column, or c) the edge of the table has been reached ("2000" and "Escudos" in the first table will both expand horizontally to include one empty cell on either sides; "Notas" in the second table would not be expanded at all, because it is centred and next to an attribute line; "2000" and "Escudos" in the second table will not be expanded because they are aligned to the left and they are next to a filled in cell).
In case one empty cell can be merged with two growing cells on either side, context knowledge would be required to reach a decision, e.g. in the case of dates, more recent dates are likely to span over more columns than more distant dates; and a cell containing "Notas" or "Nota" will most likely only span over itself. Hurst's (2001) language model may also be useful in these situations, as it captures context specific expressions and can be trained.

Output: the full physical and functional models of each table have been derived.

Evaluation: the total number of spanning cells accurately delimited in terms of the columns/rows they span. This is an interesting moment to reapply Hurst's (2003) physical model evaluation metric at cell and column/row levels (see section 4) and see how it has improved over the obtained after segmentation I.

5.6. Structural analysis I

Goal: the purpose of this task is to connect each data cell to all attribute cells that characterize it, thus grouping those cells that have to be read conjointly.

Process: in structural analysis, we use the same sort of heuristic approach that was described in section 3.4.

Output: data - attribute groupings, i.e. the reading paths of each data cell has been defined. The reading paths of those data cells with empty values in the header column may be incomplete.

Evaluation: to assert the quality of the derived structural model of the table, we will use recall and precision based on the database-type queries similar to proposed Hu et al. (2001). This evaluation however will only be conducted after Structural analysis II (section 5.7.5). From this step onwards we will divert to an end result evaluation, at the method's end.

Error: the simple heuristics used will not allow us to infer that the number 2.455.535 appearing in the last line of the balance sheet in Figure 17 really refers to the concept "Imobilizações incorpóreas" (intangible assets) shown in the second cell of the header column. We will have to come back to this during the interpretation stage.

5.7. Interpretation

Goal: pair the Items we want to extract with the respective Data extracted from the document.

Overview: only so much can be done in terms of table interpretation without reaching for world or context specific knowledge. This is also valid for human users: one has to have some knowledge of accounting to fully understand the information contained in a financial report and the intrinsic links connecting the items displayed.

Our aim is to create methods that allow an in-depth interpretation of the information presented in the tables, taking full advantage of the set of constraints the data must obey within its context, while maintaining the method's operability in other contexts. This goal can be obtained if:
- the system can function with or without a knowledge source;
- the knowledge source is an independent component, that can be replaced or altered by the user or the system itself; the information contained in the new knowledge source will be queried by the method at each relevant stage.

5.7.1. Opportunities and risks to information extraction within accounting

Accounting information holds a large set of partitive relations, where the addition of several elements must equal other elements, which in turn can be added together to form yet other elements placed higher in the hierarchical tree of accounting concepts. In accounting, most tables are numerical and in this sort of tables, whenever the leaves of a concept are presented, their root is also displayed. On the other hand, some items (e.g. profit and loss of the accounting year) appear with the same value in different tables of the statement. This sort of constraints provides a powerful tool to:
- guarantee the coherence of the information extracted; and
- improve risky decisions made in previous steps, because the best decision will be the one that leads to the satisfaction of the constraints (rather than not).

As risks, it should be stated that different accounting items may have the same or very similar names but appear in different positions in the table and in the tree of hierarchical concepts (e.g. Accruals and deferrals exist both on the Assets and Liabilities side of the Balance sheet).

Nonetheless, it would be a lengthy task to capture this accounting knowledge into an ontology that can be used by the method, especially if we consider that accounting rules differ from country to country. Luckily, the work is already being done by professional accountants.

XBRL, eXtensible Business Reporting Language, is an extension of XML to the specific field of financial reporting. XBRL taxonomies that describe the national (or jurisdictional) specific accounting rules are or have been created in the United States, Germany, United Kingdom, Spain, Belgium, among other countries, the list is getting bigger. An XBRL taxomomy comprises:

XBRL Taxonomy components	e.g. IASCF & XBRL International (2004)
- a set of elements that univocally describe a set of publishable concepts; these are intended to be used when preparing XBRL instance documents;	IssuedCapital
- a set of linkbases which characterize each:	
▪ the definitions that apply to it, which may encompass non-arithmetic complex relations with different items (in the definition linkbase);	
▪ the arithmetic constraints that relate it to other items (the calculation linkbase);	Adds into EquityParentTotal
▪ the position within the statement the item is expected to appear in (presentation linkbase);	Appears as credit in the balance sheet, after EquityPresentation
▪ the label that describes the item (label linkbase); several labels in different languages or even dialects can be defined for a given item; this particularity allows each element to be automatically translated into different languages;	EN: Issued Capital PT: Capital
▪ the relevant legislation that applies to the item (the reference linkbase); it is this relevant legislation that defines the valuation rules that apply to the element, which may differ from GAAP to GAAP.	IAS 1§73e, IAS 1§72

Table 8. Components of XBRL taxonomies

The XBRL Consortium is supervising the creation of these taxonomies to insure they comply with a minimum set of best practices. Taxonomies encompass a clear view of accounting concept hierarchy and permit identifying which items should be added up to compose others (calculation linkbase); they also permit a clear view of the order each item is expected to appear in the statement and items are grouped according to the tables that normally contain them (presentation linkbase). Furthermore, taxonomies encompass specifications of the relationships that hold amongst the items of different tables, by naming those items with the same element ID. As such, they are a powerful tool for supplying the method with knowledge on the context at hand.

5.7.2. Listing the items we wish to extract

The user will provide the method with a List of all the items it should extract, by selecting items from the taxonomy. The method will generate and append to the List each item in whichever number of ways it is likely to appear in a financial statement. It will do the same for all the items that are likely to appear in the same table. We hope to resort not only to the label linkbases but also to Wordnets for help in this area, although with a large enough set of examples Accounting Wordnets could be derived. An XBRL tag will univocally identify each item.

The items in the List will be ordered according to the Presentation linkbase and will contain a numerical column that identifies each item sequentially. We will henceforth refer to this attribute as Key.

Notice that this list will be built automatically in the accounting context, once a financial statement is presented to the method and the proper taxonomy is identified (eventually these two steps can be done by a crawler rather than the user himself). In other contexts, the user can still operate if he builds the List by hand, or adapts other ontologies for the purpose.

5.7.3. Reduction of the search space

Goal: At this point, we will use a probabilistic approach to sort the tables in the document according to their likelihood of containing the information we wish to extract. This will limit the application of the remaining computationally more expensive steps to those tables where the effort is more likely to be rewarded. We treat those tables with higher likelihood first and only handle less likely if the first do not to contain the desired information after all.

Overview: the precise location of this step within the overall method has to be carefully evaluated, eventually we would apply it right after Location I.

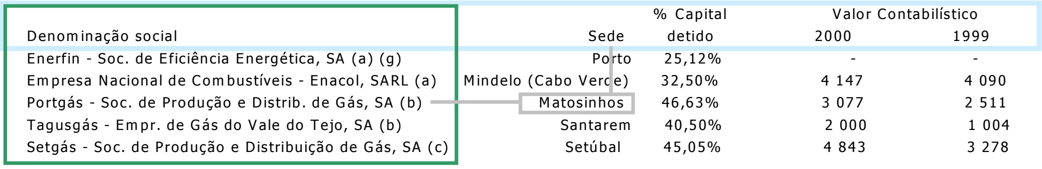
Process: we intend to derive the general document structure by identifying headings against any indexes the document may have and possibly the portion of the knowledge base that can identify the content of usual headings in the document; within the accounting context, we will recur to the main headings of the presentation linkbase. Measurement of the probability takes into account a number of features, such as the position of the identified tables within the document structure as indicated by heading identification; how precise we believe the document structure that has been derived is; the existence and content of any table titles; the presence of keywords; the approximate number of columns and rows (e.g. a balance sheet will always have more than five lines).

Evaluation: this is a classification problem with multiple categories; standard classification evaluation metrics will be applied.

5.7.4. Identifying relationships between cells

Goal: The main purpose of this step is to further assert that the table under analysis is indeed one of the tables that contain information for extraction. The method for doing this will depend on how certain the system is of having found one of the desired tables, in section 5.7.3.

Overview: There are two main behaviours of relationships types in the accounting context. One of the tables we would like to extract contains a list of characteristics of those companies that are held by the company the document is about. XBRL taxonomies do not cover this sort of table. In these tables, among the types of relationships identified in Hurst (2000) and seen in section 3.5, qualitative super-type and nominal super-type relationships are the most common for non-numeric data cells (e.g. in Figure 18, Matosinhos is the location of the headquarters, “Sede” (its nominal super-type) and a characteristic of the company Portgás (its qualitative super-type)).



		% Capital	Valor Contabilístico	
Denominação social	Sede	detido	2000	1999
Enerfin - Soc. de Eficiência Energética, SA (a) (g)	Porto	25,12%	-	-
Empresa Nacional de Combustíveis - Enacol, SARL (a)	Mindelo (Cabo Verde)	32,50%	4 147	4 090
Portgás - Soc. de Produção e Distrib. de Gás, SA (b)	Matosinhos	46,63%	3 077	2 511
Tagusgás - Empr. de Gás do Vale do Tejo, SA (b)	Santarem	40,50%	2 000	1 004
Setgás - Soc. de Produção e Distribuição de Gás, SA (c)	Setúbal	45,05%	4 843	3 278

Figure 18: Finding relationships in a table with textual data cells.

We believe that in our context and in this type of table (identifiable for example because it has a high concentration of company names) this simple heuristic will allow good results in distinguishing both types: for a given non-numeric data cell, the relationship with its column header will be nominal super-type and with its row header it will be qualitative super-type. And since relationships involving a number (other than a year) are either value (in Figure 18, the value of “% de Capital detido” is 46,63%) or measurement (“% de Capital detido” is measured as a percentage), if we properly identify measurement units, all quantitative relations will have been treated.

However, in most accounting tables but more generally in any table where a hierarchy can be detected in the main attribute column, we must also consider partitive relationships, e.g. intangible assets are a part of assets. The calculation linkbases of XBRL taxonomies address this sort of tables and allow a clear identification of partitive relationships in the calculation taxonomy.

Process for sufficiently high probabilities in section 5.7.3:
If the probability is high enough, we are quite sure of which information the table at hand contains. As such, instead of comparing the items of the table with the all the items in the List, which can be computationally expensive, we can make a greedy search:
1. Detect concept hierarchy within the contents of the header column, through several features – indentation, letter type (mostly capital or not), the presence of “:” or the presence of data in the row (higher hierarchy elements tend to have no data).
2. Improve the hierarchy by verifying arithmetic rules.
3. Follow a top-down approach to string match the table’s attributes that are higher in the hierarchy to the part of the List that holds the elements of the likely table. Use the matches found to restrict the search space of those elements lower in the hierarchy.

Process for insufficiently high probabilities in section 5.7.3:
1. Take the content of all the attributes of the most likely table. Compare them with the List of the information we wish to extract (in which all items will be named in whichever number of ways they are likely to appear in a financial statement). There are several methods to doing this string matching; as in the future we may wish to apply OCR on paper documents and spelling mistakes do occur, an error resistant method, of which Lopresti and Wilfong (1999) is an example, will be preferable. *If* the proportion

of similar items is not high enough, reject the table and take the next table with higher probability.
2. *Otherwise*, take those List items that maximize the similarity to the attributes in the table. Compute the relative distance between them, which can be approached by the variation coefficient of the Key of the similar items. *If* the absolute deviation is too high and its value cannot be attributed to a relatively small number of items, reject the table and take the next table with higher probability.
3. *Otherwise*, identify the table of the List to which more table attributes are similar; try to replace farther items with items from the same List-table, which are still similar. The result is a set of tentative pairs of extractable List item – Table attribute item.
4. Verify that the arithmetical rules that hold on List elements are present in the table.
5. Improve the pairs of extractable List item – Table attribute item in light of the relations found and hierarchies they imply.

Notice that it may be preferable to compare against the List not only the attributes in the attribute column but rather the concatenation of all the attributes of a given cell except the dates and measurement units, even because XBRL label linkbases often include such concatenations.

In both alternatives, the verification of arithmetical rules should degrade gracefully: if too few arithmetical rules are verified (the definition of “too few” will depend on the List-table at hand), the method should give up on the table and take the next most likely table; if not all rules are found, there is cause to doubt past decisions, namely the full location of tables (previously disregarded surrounding lines should be inspected), or segmentation of cells (the presence of flags to identify more doubtful decisions will be of help), or the attribution of cells to wrongful columns or lines (alternative arrangements should be tried). If these alternatives do not solve the problem and the arithmetic difference is of a material amount (within the accounting materiality is defined by standard auditing best practice), the user should be asked to intervene.

On the other hand, notice that the steps mentioned above are different from those outlined by Ferguson (1997) and Kornfeld and Wattecamps (1998). They created methods that are best applicable to reports produced under Anglo-Saxon accounting standards, where the labels used on each item are not stable. However, under Continental accounting standards a comparison between labels is much more reliable as labels are much more consistently applied by companies, so the first step to detecting partitive relations does not have to be the computationally consuming verification of mathematical relationships. The specific order of string matching/arithmetical relationship verification will be decided on an experimental basis, for those two types of accounting backgrounds and for other contexts.

Output: Pairs of extractable List item – Table attribute item.

5.7.5. Structural analysis II

Goal: The output of section 5.7.3 left the reading paths of those data cells with empty values in the header column incomplete. The detection of hierarchies done in the previous step allows the reading paths to finally be completed.

Process: Take all lines with data but no attribute in the attribute column. If there is more than one line in that situation, for example the first part of Figure 17 is such a table, knowing that the value 2.455.535 is the summation of the values in the same column in preceding rows, which form its hierarchical group, its attribute would be searched in the first attribute line that holds no data and is on the opposite end of the hierarchical group. If there is only one line in that situation, the attribute may be the table's heading itself.

Output: Pairs of extractable List item – Table attribute item – Data item.

Evaluation: recall and precision based on database-type queries aimed at asserting the quality of the derived structural model of the table, similar to proposed Hu et al. (2001).

5.7.6. Identifying units of measure

Goal: accurately identifying the unit of measurement of numerical data; in the case of currencies, a multiplier should be derived such that the data can be saved in a single database in a single currency.

Process: Two major types of units are likely to appear in an accounting context: percentages and currencies. In our data set, currencies were often stated in subtitles to each table, but sometimes they were only presented in the bigger tables and for smaller tables the unit had to be inferred from the previous (which is in fact what a human user would do). As such, the system will record the location of all expressions in the document that may refer to currencies and their multipliers (e.g. thousand, million). Each table will be classified under the currency that is stated closer to it, preferably above the table on the same page and not embedded in text. The currency thus obtained will be tested: arithmetic relations among items of different tables will be verified and the coherence of certain ratios will be accompanied (in a way similar to that outlined by Ferguson 1997).

In case the data is published for example in USD and the database is in EUR, the method may perform the conversion by including a table with the daily currency conversion ratios, which it updates over the internet.

Output: Pairs of extractable List item – Table attribute item – Data item – Unit of measure. This output can be saved in a relational database.

Evaluation: percentage of List items accurately extracted; the time the end-to-end process took. This is an end-result evaluation.

6. Conclusion

We have designed the basic characteristics of an end-to-end method for extracting information from tables. After delimiting our research area by defining what a table is, we have identified the major tasks such a method should address. We have conducted a thorough bibliographical analysis aimed at characterising the solutions found by different authors for those tasks. An important contribution of this paper is not only in drawing the path that research in this area has been taking, but also highlighting some key issues no author has accurately solved so far.

We have then proceeded to analyse table processing evaluation approaches and its difficulties. We propose two new complementary metrics to evaluate a method's the capacity of segmenting different table elements, be it columns, rows or cells; we believe these metrics fulfil one important gap in current evaluation approaches.

Finally, we have proposed our own end-to-end table processing method and presented its main components. It receives a simple input, ASCII, to which virtually any document type can be transformed. The fist step, table location, adapts its performance to the assessed difficulties of a given document, grounding decisions on a number of diverse features, which makes it more robust. Within a sample of 22 different documents with over 90,000 lines in total, it managed to reduce search space by between 50% and 70% while keeping between 95% and 99% of the relevant information. In general, the overall method does not work linearly: we use the output of later steps as input to earlier ones, thus taking advantage of the extra knowledge each stage provides to improve the quality of the previous. For example, the functional model of a table has the potential to: a) improve segmentation of spanning cells which content does not encompass all cells it is vertically aligned with; and b) unmerge similar vertically/horizontally aligned tables; so we return to location and segmentation after functional analysis. Throughout the method, flags are left behind to signal decisions that are considered to be more error-prone.

We have also explored new venues in interpretation. Specifically we have seen how increasingly common XML-based context-specific taxonomies can be adapted as a knowledge source to the method, functioning as an independent component that can be replaced. Because of this aspect and because the tasks before interpretation are conducted in the most general way possible, the method may be easily adapted to different contexts. The constraints that the context imposes on the results can be used to assert the accuracy of the extraction performed; more importantly, they provide the method with ways to detect its own mistakes and correct them by following the flags left behind in previous stages; finally, user intervention can be requested when the constraints are not met by a significant amount.

As future work, we will gradually implement several of the steps outlined and conduct a thorough evaluation of the results. We will pursue the search for better alternatives for each step but also for the integration of the different steps.

Acknowledgments

We would like to thank Josef MacDonald of the International Accounting Standards Board, for the interesting discussions on XBRL, and FEDER / Programa de Financiamento Plurianual de Unidades de I & D for financial support.

References

Auto Industrial (2000), "Relatório e contas consolidadas 2000", available at http://www.grupoindustrial.pt/pdf/rel2000.pdf

Baum, Larry, Lawrence S, John H. Boose, Molly Boose, Carey S. Chaplin, James Cheung, Ole B. Larsen, Monica Rosman Lafever, Ronald C. Provine, David Shema (2003), "Document layout problems facing the aerospace industry", *in Proceedings of the Third International Workshop in Document Analysis and its Applications, DLIA 2003*, Edinburgh, UK.

Buchsbaum, A.L., D. Caldwell, K.W. Church, G.S.Fowler, S. Muthukrishnan (2000), "Engineering the compression of massive tables: an experimental approach", in *Proceedings of the 11th ACM-SIAM Symposium on Discrete Algorithms*, pp. 175-184, Philadelphia, USA.

Cameron, James P. (1989), "A cognitive model for table editing", Technical report OSU-CISRC6/89-TR 26, Computer and Information Science Research Centre, Ohio State University, USA.

Cesarini, F., S. Marinai, L. Sarti, G. Soda, "Trainable table location in document images", International Conference on Pattern Recognition, ICPR 2002 Vol. 3, pag. 236-240, Quebec, Canada.

Chao, Hui (2003), "Background pattern recognition in multi-page PDF document", *in Proceedings of the Third International Workshop in Document Analysis and its Applications, DLIA 2003*, Edinburgh, UK.

Chen, Hsin-His, Shih-Chung Tsai, Jin-Hi Tsai (2000), "Mining Tables from Large Scale HTML Texts", in *18th International Conference on Computational Linguistics (COLING)*, pp. 166-172, Saarbrucken, Germany.

Cohen, W. W., Hurst, M., Jensen, L.S., (2002), A flexible learning system for wrapping tables and lists in HTML documents, in *Proceedings of the Eleventh International World Wide Web Conference (WWW2002)*, pp. 232-241, Hawaii, USA.

Douglas, Shona, Matthew Hurst, David Quinn (1995), "Using Natural Language Processing for *Identifying* and Interpreting Tables in Plain Text", in *Fourth Symposium on Document Analysis and Information Retrieval*, pp. 535-545, Nevada, USA.

EDP, Electricidade de Portugal, SA (2000), "Relatório e contas consolidadas 2000", available at http://www.edp.pt/download/EDP_RC.pdf

Ferguson, Don (1997), "Parsing Financial Statements Efficiently and Accurately Using C and Prolog", in *Practical Applications of Prolog Conference '97*, London, UK.

Green, Edward, M. Krishnamoorthy (1995), "Model-based of Printed Tables", in *Proceeding of International Conference of Document Analysis and Recognition 95 (ICDAR95)*, pp. 214-217, Montréal, Canada.

Handley, John C. (2001), "Table analysis for multiline cell identification", in *Document Recognition and Retrieval VIII*, *Proceedings of SPIE*, volume 4307-04, San Jose, USA.

Hu, J., R Kashi, D. Lopresti, G. Wilfong (1999), "Table detection across multiple media", in *International Workshop on Document Layout Interpretation and Its Applications,*, Bangalor, India.

Hu, J., R Kashi, D. Lopresti, G. Wilfong (2000), "Medium-Independent Table Detection", in *Document Recognition and Retrieval VII*, *Proceedings of SPIE*, volume 3967, pp. 291-302, USA.

Hu, J., R Kashi, D. Lopresti, G. Wilfong (2001), "Table structure recognition and its evaluation", in *Document Recognition and Retrieval VIII*, *Proceedings of SPIE*, volume 4307-05, San Jose, USA.

Hu, Jianying, Ramanujan Kashi, Daniel Lopresti, George Nagy, Gordon Wilfong (2001b), "Why Table Ground-Truthing is Hard", in *Proceedings of Sixth International Conference on Document Analysis and Recognition (ICDAR '01)*, Seattle, USA.

Hu, J., R Kashi, D. Lopresti, G. Wilfong (2002), "Evaluating the performance of table processing algorithms", in *International Journal of Document Analysis and Recognition*, USA.

Hurst, Mathew, Shona Douglas (1997), "Layout and Language: Preliminary investigations in recognizing the structure of tables", in *Proceedings of International Conference on Document Analysis and Recognition (ICDAR'97)*, pp. 1043-1047, Ulm, Germany.

Hurst (2000), "The interpretation of tables in texts", PhD. Thesis, School of Cognitive Science, Informatics, The University of Edinburgh, UK.

Hurst, Mathew, Tetsuya Nasukawa (2000), "Layout and Language: Integrating Spatial and Linguistic Knowledge for Layout Understanding Tasks", in *Proceedings of the 18th International Conference on Computational Linguistics*, *ICCL*, Saarbruecken, Germany.

Hurst, Mathew (2001), "Layout and Language: An Efficient Algorithm for Text Block Detection based on Spatial and Linguistic Evidence", in *Document Recognition and Retrieval VIII*, *Proceedings of SPIE*, Volume 4307, pp. 55-67, San Jose, USA.

Hurst, Mathew (2003), "A constraint-based approach to table structure derivation", in *Proceedings of International Conference on Document Analysis and Recognition (ICDAR'03)*, pp.911-915, Edinburgh, UK.

IASCF, XBRL International (2004), "International Financial Reporting Standards (IFRS), General Purpose Financial Reporting for Profit-Oriented Entities (GP), 2004-06-15, Exposure Draft", UK.

Kieninger, Thomas (1998), "Table Structure Recognition Based on Robust Block Segmentation", in *V Document Recognition*, *Proceedings of SPIE*, San Jose, USA.

Kieninger, Thomas, Andreas Dengel (1998), “A paper-to-html table converting system”, in *Proceedings of Document Analysis Systems, (DAS’98)*, Nagano, Japan.

Klein, Bertin, Serdar Gökkus, Thomas Kieninger, Andreas Dengel, “Three Approaches to "Industrial" Table Spotting” (2001), *Proceedings of Sixth International Conference on Document Analysis and Recognition (ICDAR '01)*, Seattle, USA.

Kornfeld, William, John Wattecamps (1998), “Automatically locating, extracting and analyzing tabular data”, in *Proceedings of the 21st annual international ACM SIGIR conference (SIGIR ’98)*, pp. 347-349, Melbourne, Australia.

Lopresti, Daniel, Gordon Wilfong (1999), “Cross-Domain Approximate String Matching”, in *Proceedings of the Sixth International Symposium on String Processing and Information Retrieval*, pp. 120-127, Cancun, Mexico.

Ng, Hwee Tou, Chung Yong Lim, Jessica Li Teng Koo (1999), “Learning to recognize tables in free text”, in *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pp.443-450, Maryland, USA.

Pinto, David, McCallum, A., Wei, X., Croft, W. B., (2003), “Table Extraction Using Conditional Random Fields”, *Proceedings of SIGIR 2003*, ACM.

Pyreddy, Pallavi, W. Bruce Croft (1997), “A System for Retrieval in Text Tables”, Technical report 105, Center for Intelligent Information Retrieval, Department of Computer Science, University of Massachusetts, Massachusetts, USA.

Rus, Daniela, Kristen Summers (1994), “Using White Space for Automated Document Structuring”, in *Workshop on the Principles of Document Processing*, Seeheim, Germany.

S.A.I.C., Science Applications International Corporation (1997), “An Automated Conversion of Structured Documents into SGML” Technical Report, Distributed Object Computation Testbed (DOCT), San Diego Supercomputer Center (available at http://www.sdsc.edu/DOCT/Publications/a3-3/a3-3.html).

Shamillian, John H., Henry S. Baird, Thomas L. Wood (1997), “A retargetable table reader”, in *Proceedings of the IAPR 97 International Conference on Document Analysis and Recognition 97*, pp.448-453, Ulm, Germany.

Silva, Ana Costa e (2003), “Extracting information from tables in text - an application to financial statements of Portuguese companies”, Thesis for Masters in Science in Data Analysis and Decision Support Systems, Faculty of Economics of the University of Oporto, Portugal.

Silva, Ana Costa e, Alípio Jorge, Luís Torgo (2003), “Selection of Table Areas for Information Extraction”, *in Proceedings of the Third International Workshop in Document Analysis and its Applications, DLIA 2003*, Edinburgh, UK.

Silva, Ana Costa e, Margarida Brites Ramos (2004), “Reporting standards for statistical purposes - the experience of Banco de Portugal”, Best paper award at the *Digital*

Accounting Research Conference 2004, Spain; Accepted for publication in the *International Journal of Digital Accounting Research*, vol 4, n.º 8, pp. 145-174.

EPC, European Parliament and Council (2003), "Directive 2003/58/EC of the European Parliament and Council of 15 July 2003 amending Council Directive 68/151/EEC, as regards disclosure requirements in respect of certain types of companies", Official Journal of the European Communities L 221, 04/09/2003, pp.13-16, Belgium.

Tersteegen, Wolfgang, Claudia Wenzel (1998), "ScanTab - Table Recognition by Reference Tables" in *Proceedings of Document Analysis Systems, (DAS'98)*, Nagano, Japan.

Thompson, Marcy (1996), "A tables manifesto", in *Proceedings of SGMK Europe*, pp. 151-153, Munich, Germany.

Tupaj, Scott, Zhongwen Shi, C. Hwa Chang, Hassan Alam (1996), "Extracting Tabular Information From Text Files", EECS Department, Tufts University, Medford, USA (available o-line at http://www.ee.tufts.edu/hchang/paper1.ps).

Yoshida, Minoru, Kentaro Torisawa, Jun'ichi Tsujii, "A method to integrate tables of theWorldWideWeb", in *First International Workshop on Web Document Analysis (WDA2001)*, Seattle, USA.

Wang, Yalin, Ihsin T. Phillips, Robert Haralick, (2000) "Improvements of zone content classification by using background analysis", in *Proceedings of Document Analysis Systems, (DAS'00)*, Rio de Janeiro, Brazil.

Wang, Yalin, Ihsin T. Phillips, Robert Haralick, (2001) "Automatic ground truth generation and A background-analysis-based table structure extraction method", in), *Sixth International Conference on Document Analysis and Recognition (ICDAR '01)*, Seattle, USA.

Wang, Yalin, Ihsin T. Phillips, Robert Haralick, (2002) "Table detection via probability optimization", in *Proceedings of Document Analysis Systems, (DAS'02)*, Princeton, NY, USA.

Wang, Yalin, J. Hu, (2002), "A machine learning based approach for table detection on the web", in *Proceedings of the Eleventh International World Wide Web Conference (WWW2002)*, pp. 242-250, Hawaii, USA.

Zanibbi, R., D. Blostein, J.R.Cordy (2003), "A survey of Table Recognition: models, observations, transformations, and inferences", in *International Journal of Document Analysis and Recognition.*