Scientific Paper / Artículo Científico

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR
INGENIUS
Revista de Ciencia y Tecnología
https://doi.org/10.17163/ings.n25.2021.05
pISSN: 1390-650X / eISSN: 1390-860X

Algorithms for Table Structure Recognition

Algoritmos para el reconocimiento de estructuras de tablas

Yosveni Escalona Escalona[1,*]

Received: 03-08-2020, Reviewed: 22-09-2020, Accepted after review: 09-10-2020

Abstract

Tables are widely adopted to organize and publish data. For example, the Web has an enormous number of tables, published in HTML, embedded in PDF documents, or that can be simply downloaded from Web pages. However, tables are not always easy to interpret due to the variety of features and formats used. Indeed, a large number of methods and tools have been developed to interpreted tables. This work presents the implementation of an algorithm, based on Conditional Random Fields (CRFs), to classify the rows of a table as header rows, data rows or metadata rows. The implementation is complemented by two algorithms for table recognition in a spreadsheet document, respectively based on rules and on region detection. Finally, the work describes the results and the benefits obtained by applying the implemented algorithm to HTML tables, obtained from the Web, and to spreadsheet tables, downloaded from the Brazilian National Petroleum Agency.

Keywords: Tabular Data, HTML Tables, Spreadsheets, Conditional Random Fields, Machine Learning, Algorithm.

Resumen

Las tablas son una manera muy común de organizar y publicar datos. Por ejemplo, en el Internet se halla un enorme número de tablas publicadas en HTML integradas en documentos PDF, o que pueden ser simplemente descargadas de páginas web. Sin embargo, las tablas no siempre son fáciles de interpretar pues poseen una gran variedad de características y son organizadas en diferentes formatos. De hecho, se han desarrollado muchos métodos y herramientas para la interpretación de tablas. Este trabajo presenta la implementación de un algoritmo, basado en campos aleatorios condicionales (CRF, Conditional Random Fields), para clasificar las filas de una tabla como fila de encabezado, fila de datos y fila metadatos. La implementación se complementa con dos algoritmos para reconocer tablas en hojas de cálculo, específicamente, basados en reglas y detección de regiones. Finalmente, el trabajo describe los resultados y beneficios obtenidos por la aplicación del algoritmo para tablas HTML, obtenidas desde la web y las tablas en forma de hojas de cálculo, descargadas desde el sitio de la Agencia Nacional de Petróleo de Brasil.

Palabras clave: datos tabulados, tablas HTML, hoja de cálculo, campos aleatorios condicionales, aprendizaje automático

[1,*]Research and Development Department, SOLINTEC, Araçatuba, Brazil
Corresponding author ✉: yosveni.escalona@solinftec.com.br. http://orcid.org/0000-0003-2992-0540

Suggested citation: Excalona Escalona, Y. (2021). «Algorithms for Table Structure Recognition». Ingenius. N.° 25, (january-june). pp. 50-61. doi: https://doi.org/10.17163/ings.n25.2021.05.

1. Introduction

The volume of data available on the Web has grown in a dizzying way, which makes the Web a vast repository of data that describe our environment and our interactions. The wealth and strength of these allow the development of today's economy and society.

The data found on the Web are related to product information, articles imparting encyclopedic knowledge, presentations of state-of-the-art scientific results or reports on current financial data. A great part of those data can be found in tables, which require a particular analysis since they may be expressed in HTML, embedded in PDF documents or made available as downloadable spreadsheets, among other formats. Usually, they are organized merely and compactly as rows and columns, but they can be much more complex, with metadata and additional information.

Tables proved to be valuable sources, but their use can be very diversified, ranging from Web search to data discovery in spreadsheets and knowledge base augmentation [1]. In the literature, one finds research about methods and tools for tabular data extraction from spreadsheets, HTML table, tables embedded in PDF documents, etc. The vast majority of these methods and tools follows strategies based on heuristic rules and machine learning algorithms. The strategy for tabular data extraction and for table row classification also depends on the document format. Exploring a large set of tables has been a challenge because, in general, table semantics is not known. In [2], a corpus of over 100 million tables is presented, but the meaning of each table is rarely explicit in the table itself. Another challenge is the structure of the table. For example, the tasks described in [3–6] focus on recovering table semantics and linking table data with external sources for tables classified as genuine, with considerable data loss. These works do not consider fundamental aspects, such as the orientation of the table, and discarded those tables that are classified as non-genuine.

Another aspect to consider is based in the kind of document, for example, Correa and Zander [7] analyzed a group of methods and tools focused on extracting tabular content from PDF based on two main characteristics: ease of use and output results and the categorization of the tools based on theoretical proposals, free of cost and commercial. In [8] were developed several heuristics, which together recognize and decompose tables in PDF files and store the extracted data in a structured data format (XML) for ease of use, these heuristics are divided into two groups: table recognition and table decomposition. Other techniques were presented in [9] for data tabular extraction from PDF documents with the goal of identified table boundary where the authors describe a methodology that applies two machine learning algorithms, CRFs and Support Vector Machine (SVM). Also, have been reviewed works based in the table boundaries identification process and designed for the semantic matching and annotation of numeric and time-varying attributes in Web tables as the presented in [10–12] which annotate Web tables effectively and efficiently and identify the boundaries between attributes name rows (or columns) and its corresponding value data rows (or columns) in the table.

Furthermore, we can do special mention to the works related to the recognition of HTML Table Structure and table header detection and classification described in [13, 14] suggesting some techniques based on heuristics rules and used a learning classification algorithm for delineating kinds of tables existing into a document and detecting the structure and header types.

Finally, we make emphasis on the approach proposed in [15] that was based on machine learning techniques that cover two fundamental tasks of the table extraction process: localization of the table and identification of the row positions and types. This work is focused on the implementation of two algorithms to table recognition in Spreadsheets as well as other algorithm based on Conditional Random Fields (CRFs), to classify the kinds of rows into tables. The datasets were created with HTML tables downloaded from the Web and spreadsheet tables were downloaded from the Web site of the National Oil Agency of Brazil (ANP).

1.1. Background

Tables are frequently found in printed documents, such as books or newspapers, as well as digital documents, such as Web pages or presentation slides. But they also represent an essential concept in relational databases and spreadsheets. They may be distinguished according to their structure and orientation. A relational or horizontal [8], as illustrated in Table 1, has rows, which provide data about specific objects, called entities, and columns, which represent attributes that describe the entities.

Table 1. Example of a relational table

ID	Name	Age	Country	Job
1	Bob Smith	35	USA	Programmer
2	Jane Smith	31	USA	Teacher
3	Robert White	24	UK	Engineer

More complex types of tables exist, such as those where the attributes that describe the entities are laid vertically and the entities in a horizontal way or other kinds of structures as show Table 2 and Table 3.

Table 2. Example of a non-relational table

	Obj 1	Obj 2	Obj 3
Name	V1	V1	V1
Age	V2	V2	V2
Height	V3	V3	V3

Table 3. Example of a non-relational table with additional information

Patent Applications by Residents		
Data source: worldbanc.org (show countries top in each continent)		
Country	**Residents**	**Applications**
North America		
Unites States	307,700,000	224,912
Canada	33,739,900	5,067
Asia		
Japan	127,557,958	295,315
China	1,331,380,000	229,096

De manera más precisa, una tabla se define como:

Definition 1. A table is a pair T = (H,D) consisting of an optional header H and data D, where:

- The header $H = \{h_1, h_2, \ldots, h_n\}$ is an n-tuple of header elements h_i; if the set of header elements exists, it might be represented either as a row or as a column.
- The data nodes are organized as an (n, m) matrix consisting of n rows and m columns:

$$D = \begin{bmatrix} C_{11} & \therefore \cdots & C_{1m} \\ \vdots & \ddots & \vdots \\ C_{n1} & \cdots & C_{nm} \end{bmatrix}$$

The table row classification process consists of identifying each of the elements of a table. The general idea is based on locating the header and data in the table. It is also relevant to identify the layout elements and metadata in a table. Figure 1 shows the table row classification process denoting in different colors some of the elements present in the table: red indicates the elements that represent the titles; yellow the row header; blue the data rows; and green the additional metadata.

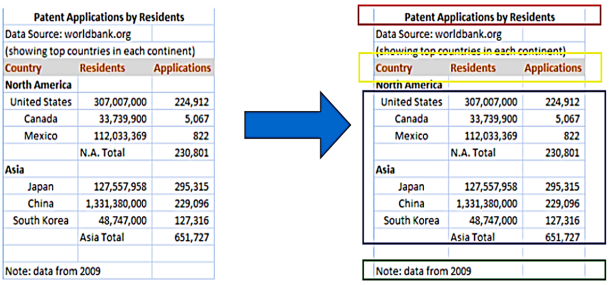


Figure 1. Tabular extraction process

The rest of the paper is organized as follows. Section 2 covers details of the implementations of the algorithms for table recognition and for table row classification. Finally, Section 3 describes experiments and results.

2. Materials and Methods

2.1. Algorithms for Table Recognition and for Table Row Classification

This section describes implementations of algorithms to recognize tables in spreadsheets and an algorithm, based on Conditional Random Fields, to classify table rows.

2.1.1. A Rule-based Algorithm to Detect Tables in Spreadsheets

Several Rule-based algorithms detect tables in spreadsheets making use of cell attributes, such as border, format, and data type. Where each cell attribute in the spreadsheet has a specific value associated with that cell. In turn, the cell border has the attributes direction, style, and color. The border may surround the cell in 4 different directions: top, bottom, left, and right.

A cell format is the visual formatting applied to the data of the cell, such as, number format, font style name, font name, font size, font bold, font italic, and font color. The detection of multiple tables in the same spreadsheet is performed by finding a separator between two tables (usually a set of empty rows), as explained in what follows [16].

Given a table T, with rn rows and cn columns, the following layout features are computed:

- Average number of columns, computed as the average number of cells per row.

$$c = \frac{1}{rn} \sum_{i=1}^{rn} c_i \quad (1)$$

where c_i is the number of cells in row i, $i = 1, \ldots rn$.

- Average number of rows, computed as the average number of cells per columns.

$$r = \frac{1}{cn} \sum_{i=1}^{cn} r_i \tag{2}$$

where r_i is the number of cells in column i, $i = 1, \ldots, cn$.

Figure 2 shows the algorithm that identifies the number of tables into a document and captures the range of the rows that represent the tables.

Algorithm 1 Table Detection and Recognition

Input: A spreadsheet document
Output: Number of tables into the document (ct)

$U \leftarrow$ The threshold to empty rows
$H \leftarrow$ Set header cells
$D \leftarrow$ Set data cells
$T \leftarrow$ Set title cells
$E \leftarrow$ Set empty rows
$R_i \leftarrow$ Numbers of cells tagged as header in row i
$X \leftarrow$ Numbers of Empty cells between a header row and title row
for $i \leftarrow 0$ to rn **do** ▷ Number of rows into the spreadsheet document
  for $j \leftarrow 1$ to cn **do** ▷ Number of columns into the spreadsheet
    if $C[i-1,j]$ **is** $Header$ **and** $C[i-1,j].type = String$ **then**
      if $C[i,j].format = C[i,j-1].format = C[i,j+1].format$ **and** $C[i-1,j].format = C[i-1,j].format = C[i,j].font = C[i-1,j].font$ **then**
        if $R_i > 0$ **then**
          $C[i,j] \in Header$
          if $C[i-1,j] \neq Empty$ **and** $C[i,j].format \neq C[i,j].format$ **then**
            $H \leftarrow H \cup \{c\}$ ▷ Adds c to the set of header cells
          end if
        end if
      end if
    else if $C[i,j]$ **and** $C[i-1,j]$ **and** $C[i+1,j] \in \{type, format\}$ **then**
      $D \leftarrow D \cup \{c\}$ ▷ Adds c to the set of data cells
    else if $C[i,j].type = String$ **and** $E \geq X$ **then**
      if $C[i,j-1]$ **and** $C[i,j+1] \in E$ **and** j *is first column* **then**
        $T \leftarrow T \cup \{c\}$ ▷ Adds c to the set of title cells
      else
        $E \leftarrow E \cup \{c\}$ ▷ Adds c to the set of empty cells
      end if
    end if
  end for
  if $len(E) = U$ **then** ▷ if the count of empty cells equals to threshold
    $ct \leftarrow ct + 1$
  end if
end for
return ct

Figure 2. Table Detection and Recognition Algorithm

Region Detection

Region Detection is computed through a graphp-based algorithm that detects tables in spreadsheets, called Remove and Conquer [17], it uses a comprehensive set of rules and heuristics based on a graph representation of a spreadsheet. The spreadsheet files contain one or more sheets and where each sheet is comprised of a collection of cells organized in rows and columns are defined as some useful definitions that help in the region detection process.

Definition 2. Let W denote the set containing all the cells of a worksheet.

Region detection consists in scanning the spreadsheet from the first cell in the left top corner until the last non-empty cell in the right bottom corner to check cells with similar formatting and to detect separators, such as empty rows, different cells formatting or different kinds of borders, such as different cell value type. A region is defined as follows. More precisely, a region is defined as follows.

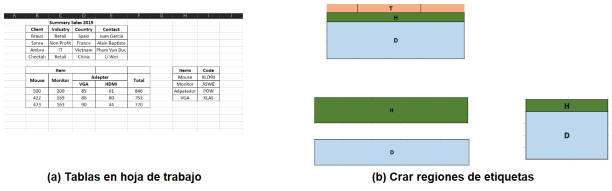
Definition 3. A region is a maximal collection $R \subseteq W$ of cells from a rectagular area of the worksheet.

It also is inferred the layout role of non-empty cells in the worksheet where each non-empty cell is assigned one of the following roles: Header(H), Data(D), Title(T), Metadata or non-relational (N). This cell role is defined as following.

Definition 4. Let label: $W \rightarrow$ Labels, where Labels = {Header, Data, Title, Metadata}, be a function that maps cells to their assigned layout role. For empty cells label is undefined. We identify them using empty: $W \rightarrow \{0, 1\}$. It returns 1 for empty cells, otherwise 0.

Sea la *label*: Etiquetas, donde Etiquetas = {Encabezado, Datos, Título, Metadatos}, una función que relaciona a las celdas su rol de diseño asignado. Para celdas vacías, la etiqueta no está definida; estas celdas se identifican utilizando *empty*: , que retorna 1 para celdas vacías y 0 en otro caso.

The cells of a spreadsheet are grouped together so that adjacent cells have the same layout role (label) or form larger structures. These groups are called label regions, as shown in [17], as shown in Figure 3



(a) Tablas en hoja de trabajo (b) Crar regiones de etiquetas

Figure 3. Creation label Regions Process

Formally, a label region is defined as follows:

Definition 5. A label region is a region LR of a spreadsheet such that, for any two cells c and c' in LR, $label(c) = label(c')$ and $empty(c) \neq 1$ and $empty(c') \neq 1$.

Figure 4(a) shows tables in a spreadsheet and Figure 4(b) indicates the regions corresponding to the table structures. The label region detection process clusters cells based on their label. It is iterated over each row to create sequences of cells having the same label. These form the base LRs. Subsequently, it merges LRs from consecutive rows, if their labels, minimum column, and maximum column match.

Table Representation through Graphs

Regions allow constructing graphs that captures the spatial interrelations of label regions. Figure 4 shows the representation of tables as a graph.

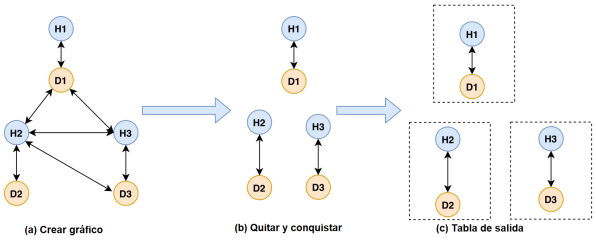


Figure 4. Table Representation through Graphs

The graph construction process consists of identifying spatial relations, such as top, bottom, left, and right, based on locating the nearest neighboring regions for each direction and identifying all vertices whose maximum row is less than the minimum row of another vertex. For each direction, a distance function is defined where all the nearest vertices are identified:

$$ND_v = \{n \in D_v \vee ddist(v,n) = \underset{u \in DV}{minddist}(u,v)\} \quad (3)$$

where D_v is the direction for the vertex v; directed edges (v,n) are created for every $n \in ND_v$.

2.1.2. The Remove and Conquer Algorithm

Remove and Conquer (RAC) is a rule-based algorithm whose objective is to separate the edges that are farther to the left and to the right direction of the graph that was created from each worksheet in a spreadsheet as shows Figure 5. The algorithm processes the strongly connected components of the graph to pair all groups formed and to detect valid tables.

The vertices are sorted in descending order of their maximum row, followed by the ascending order of their minimum row, thus the tables are searched in order inverse, from the bottom to top. Each header h is individually processed to identify vertices with minimum row greater or equal to h.

The algorithm that verifies the valid header is shown in Figure 6. All valid headers are stored in Q that represent the vertices set, including h; these vertices set is called potential tables.

Algorithm 2 Remove and Conquer
Input: Graph representation of a worksheet
Output: Tables into document
$P \leftarrow 0$
$E_l \leftarrow \{e \in E / dir(e) = Left$ **and** $ldist(e) > 1\}$
$E_r \leftarrow \{e \in E / dir(e) = Right$ **and** $rdist(e) > 1\}$
$E \leftarrow (E_l \cup E_r)$
for $G^s \in getSCC(G)$ **do**
  $LQ \leftarrow Null$
  $S^{'} \leftarrow \{v \leftarrow S^{'} / lbl(v) = Header\}$
  if $|S^{'}_H| > 0$ **then**
    for $h \leftarrow S^{'}_H$ **do**
      if $h \leftarrow LQ$ **then**
        $Q \leftarrow \{s \in S / rmin(s) \geq rmin(h)$ **and** $hasPath(s,h,E_s)\}$
      end if
      if $isValid(h,Q,0.5)$ **then**
        $P \leftarrow P \cup \{LQ\}$
        $S^{'} \leftarrow S^{'} / Q$
      else if $LQ = Null$ **then**
        if $|Q| = 1$ **and** $isAligned(h,LQ)$ **then**
          $LQ \leftarrow LQ \cup \{h\}$
        end if
      end if
    end for
  end if
  $P \leftarrow P \cup \{LQ\}$
end for
$U \leftarrow U \cup \{S\}$ ▷ Remaining unpaired
$P, U \leftarrow handleOverlapping(P,U)$
for $u \in U$ **do** ▷ Find nearest table left or right
  $N, dist \leftarrow getNearestVertices(u, (E_l \cup E_r))$
  $P^{'} \leftarrow \{P \leftarrow P / 0 < |N \cap P|\}$
  if $|P^{'}| = 1$ **and** $dist \leq 3$ **then**
    $P \leftarrow P \cup \{u\}, where\ P \in P^{'}$
  end if
end for
return P, U

Figure 5. Remove and Conquer Algorithm

Input: h: a Header vertex, Q: vertices to form table with, th: threshold for alignment ratio
Output: $True$ if h is valid, $False$ otherwise
if $|\{q \in Q | rmin(q) > rmax(h)\}| > 0$ **then**
  $Q_{\mathcal{H}} \leftarrow \{q \in Q | lbl(q) = Header$ **and** $rmin(q) \leq rmax(h)$ **and** $rmin(q) \geq rmin(h)\}$
  $X \leftarrow \emptyset$; $X' \leftarrow \emptyset$
  for all $u \in Q_{\mathcal{H}}$ **do**
    $X \leftarrow X \cup \{x \in \mathbb{N} | cmin(u) \leq x \leq cmax(u)\}$
  for all $v \in Q \setminus Q_{\mathcal{H}}$ **do**
    $X' \leftarrow X' \cup \{x \in \mathbb{N} | cmin(v) \leq x \leq cmax(v)\}$
  return $\frac{|X \cap X'|}{|X'|} \geq th$ **and** $|X| > 1$
else
  return $False$

Figure 6. Header Validity Check

The algorithm ensures that other vertices connected to h are not left isolated. Those vertices paired with a valid header are subtracted from the vertices set and then ordered to create set S'. The valid headers are appended to the set of valid headers, called LQ. The vertices that represent potential tables, called Q, are not appended directly to the tables set P because the algorithm needs to check that h is not connected to other vertices. The tables that cannot be formed are stored in U. Then, in the last step the algorithm, it attempts to pair the tables in U with the nearest table on their left or right.

2.1.3. A Machine Learning Algorithm for Table Row Classification

One important contribution in this work is the identifying and classification of the kinds of rows that compose a table through the implementation of a machine learning algorithm, this case, Conditional Random Fields(CRFs) which is based on features set, values of the cells, as well the classes that represent the table structure.

CRFs are undirected graph models, introduced by Lafferty *et al* [18], that can act as classifiers for sequence labeling tasks. They are frequently used for natural language processing, such as part-of-speech tagging. The CRF algorithm defines X as a random variable over data sequences to be labeled, and Y as a random variable over the corresponding label sequences. Figure 7 shows a structure of a linear Conditional Random Field.

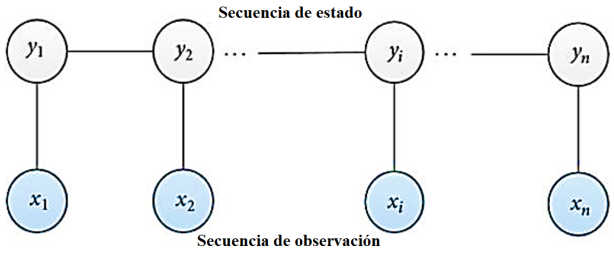


Figure 7. Structure of a linear chain Conditional Random Field

In our problem of classifying table rows, the input sequence x corresponds to a series of rows of a given table, while the label sequence y is the series of labels assigned to the observed rows. Each row x in is assigned exactly one label in y.

Formally, Conditional Random Fields are defined as follows:

Definition 6. Let $G = (V, E)$ be a graph and let $Y = (Y_v)_{v \in V}$ be a sequence of random variables indexed by the vertices of G. A conditional random field is a pair (X, Y) such that, when conditioned on X, the random variables Y_v obey the Markov property with respect to the graph.

$$P(Y_v|X, Y_w, w \neq v = P(Y_v \vee X, Y_w \approx v) \quad (4)$$

where $w \approx v$ means that w and v are neighbors in G.

The probability $P(X \vee Y)$ of a state sequence Y, given an observation sequence X, is:

$$P(X \vee Y) = \frac{1}{Z(x)} exp(\sum_j \lambda_j f_j(Y_{i-1}, Y_i, X, i) + \\ + \sum_k \mu_k g_k(Y_i, X, i)) \quad (5)$$

where $f_j(Y_{i-1}, Y_i, X, i)$ is a transition feature function of the observation sequence and of the labels at positions i and i-1 in the label sequence; $g_k(Y_i, X, i)$ is a state feature function of the label at position i and the observation sequence; and λ_j and μ_k are parameters to be estimated from training data.

In the data table scenario, X represents the list of rows in the table, and Y represents the corresponding row classes. Each relational data table has a schema, which, in the context of data tables, consists of attribute names, values, and types, where attribute names are column titles, attribute types are the types of values in the column, and attribute values correspond to data values in the column's cells. Column names are stored in a special row or rows, usually near the head of the table, called header rows, while the data is stored in rows referred to as data rows.

The data table may also contain descriptions of data, which it refers to the metadata. In corresponding to the criteria addressed above are identified each type of row according to the properties of each cell in a data table. Then, we focused our problem in assign one label each row where each row consists of constituent cells, which can exhibit different sets of attributes. The feature selection process involves the extraction of a collection of attributes for individual cells and combining attributes from all cells in the row, in order to construct a set of row features. Consider the ideas addressed above and an example of a simple table with header and data as showed in Figure 8.

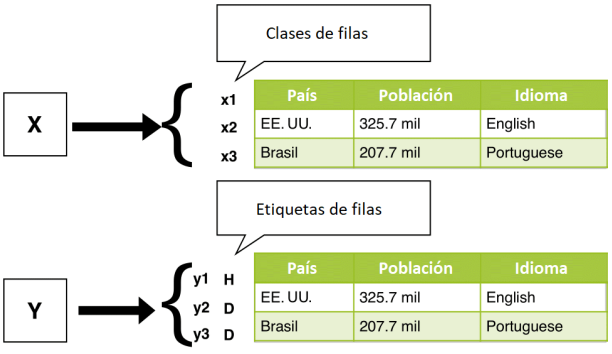


Figure 8. Example of a labelled table

The X represents a vector with the rows of the table and Y represents another vector with the tags of each row x of the table.

2.1.4. Row Classes

According to the structure of tables and the Definition 6 was defined the kinds of classes as shown 4.

Table 4. Row classes

Label	Description
H	Represents the header row in the table
D	Data rows contains data records
N	Non-relational metadata

2.1.5. Feature Set

In any machine learning algorithm, a feature is an individual measurable property or characteristic of a phenomenon being observed [19]. So, each feature was partitioned into three categories considering aspects related to the layout, styles, and values which we call layout attributes.

Layout attributes are the cells that are commonly found in header rows, which usually contain merged table cells with centered text.

Style attributes are various properties derived from stylesheets, such as font type, font color, font weight or underlined text.

Value attributes are those that represent cells where the stored information is exclusively linked to the data rows. Header rows often contain relatively short textual values, rather than numbers or dates.

Let see an example like the CRF algorithm works in our table classification problem given the transition feature $f_j(Y_{i-1}, Y_i, X, i)$ and features function $g_k(Y_i, X, i)$:

> x is a row into the data table.
>
> j is a position-row in the table (each feature is associated with a position); more than one feature associated with the same position.
>
> y_{j-1}, y_j are the tags (classes) assigned to rows j and $j-1$ de x

Then, the feature function and the state function are as follows:

$$f_1(Y_{i-1}, Y_i, X, i) = \begin{cases} 1 if x_j \in heady_j = H \\ 0, otherwise \end{cases} \quad (6)$$

$$f_2(Y_{i-1}, Y_i, X, i) = \begin{cases} 1 if x_j \in daty_j = D \\ 0, otherwise \end{cases} \quad (7)$$

$$g_1(Y_i, X, i) = \begin{cases} 1 if (x_j isacell \in x) \wedge (x_j \in rowfeatures) \wedge y_j \\ 0, otherwise \end{cases} \quad (8)$$

$$g_2(Y_i, X, i) = \begin{cases} 1 if (x_j isacell \in x) \wedge (x_j \in rowfeatures) \wedge y_j = D \\ 0, otherwise \end{cases} \quad (9)$$

The full list of individual cell attributes is given in Table 5. The features are divided according to the kind of attributes that they represent.

Table 5. Atributos de celdas

Layout	Style	Value	Spatial
IsMerged	IsBold	IsEmpty	RowNumber
Aligment	IsItalic	IsText	ColNumber
	IsUnderlined	IsNumber	NumNeighbor
	IsColored	IsDate	MatchStyle
	Font	IsAlpNum	MatchType
	Format	IsCapital	
	Border	TotalWord	

2.1.6. Similarity between rows

Another characteristic that was taken into account to generalize the training data was the row similarity [20] where a unique feature is assigned to each unique combination (c, r) where c is the number of cells exhibiting an attribute and r is the number of cells in the row. Then, two rows R_x and R_y are considered similar with respect to a certain cell attribute α if the logarithm of their widths are equal and the logarithm of the number of cells exhibiting or lacking attribute α. This approach is known as feature binnig and can be defined as follow.

Formally, for a row R_i of length r in which c cells exhibit a specific cell attribute α, is assigned feature $R_\alpha = (a, b)$ to $R_j(a, b)$, where a and b are the bin and computed as follows.

$$a = \begin{cases} 0, & if c = 0 \\ \lfloor log_2(c) + 1 \rfloor, & if 0 < c \leq r/2 \\ \lfloor log_2(r - c) + 1 \rfloor^-, & if\ r/2 < c < r \\ 0^- & if c = r \end{cases} \quad (10)$$

$$b = \lfloor log_2(r) \rfloor \quad (11)$$

The aim of the bins is given by:

1. Differentiate Between Table Widths.
2. Aggregate Wide Tables.
3. Highlight Uniform Rows.

3. Results and discussion

This section presents the experiments performed to test the accuracy of the implementation of the table row classifier as well as the experiments with table recognition in spreadsheet documents.

3.1. Preprocessing

The preprocessing task focuses on two table scenarios, HTML tables and spreadsheets tables, with the goal of removing irrelevant content or content that will not provide information for our table row classifier. Other aspects that were considered were the structure of the tables and the information present in both kinds of tables. This work does not cover in details whole the preprocessing of tables so we only emphasize those elements that we consider most important. In the case of the spreadsheet tables we highlight that the dataset had a predefined annotation, but with many errors related to the identification of ranges of data rows and head rows.

3.2. Main Characteristics of the Datasets used for Testing

Table 6 shows the statistics of the annotation process in both datasets. Each row of each table was annotated with the label corresponding to its class: "H" for header, "D" for data, etc.

Table 6. Annotated tables

	HTML	Spreadsheet
Annotated tables	105	252
Annotated rows	13,025	227,638
Header rows	105(<1 %)	252(<1 %)
Data rows	12,920(99 %)	227,254(98 %)
Other row classes	0(0 %)	132(<1 %)

The above table indicates that a critical aspect of both HTML and spreadsheets tables is that the percentage of header rows is very low, due to the fact that the tables obtained were simple tables with simple schemes (tables with a single header row followed by one or more data rows).

3.3. Table Classification Experiments

This section presents the experiments to evaluate the table classification solution proposed. In a first stage, the algorithm was trained with 80% of the data and tested with 20% of the data, randomly selected. The algorithm used L-BFGS as the optimization method and regularizations parameters L1 and L2 set to 0.1 and 0.01. The experiments with HTML and spreadsheet tables were performed separately to expose the differences between the two table formats. The performance metrics adopted were precision, recall, f1-score, support.

3.3.1. Results

Shows the results obtained. We observe that the precision value for spreadsheet tables was higher than that for HTML tables due to two main factors: (1) the features of the spreadsheet tables have a better definition; (2) we guarantee a correct definition for data rows. The recall was similar for both kinds of tables, as well as the F1-score. An important point in this analysis relates to the number of rows classified as non-relational in the spreadsheet dataset, due to the fact that we annotated spreadsheets tables manually, as opposed to the HTML tables, where some rows could have been identified as data rows or header rows, being in fact non- relational rows (Table 7).

Table 7. Results for HTML and Spreadsheets tables

Row class	Precision	Recall	F1-Score	Support
HTML				
D	0.966	0.982	0.970	2,496
H	0.955	0.992	0.970	17
N	0.980	0.980	0.970	92
Spreadsheets				
D	0.997	0.985	0.994	39,08
H	0.969	0.993	0.983	49
N	0.985	0.965	0.974	5

Note: row labels are as in Table 4:
D: Data rows
H: Header rows
N: Non-relational metadata (a note, clarification, etc.)

3.3.2. Cross Validation

Validation is the process of deciding whether the numerical results quantifying hypothesize between variables are acceptable as descriptions of the data this is a helpful process when there is not enough data to train a model and there is a large imbalance in the number of objects in each class. Then, was applied a k-fold strategy known as stratified k-fold, which is a slight variation in the k-fold cross-validation technique, such that fold contains approximately the same percentage of samples of each target class as the complete set.

Table 8 shows the results obtained for both datasets. We observe that for case of HTML tables the best results were achieved for k=2 and k=3 and that the average accuracy was 0.958 and for spreadsheets tables each k=1...,5 is similar and that the average accuracy was 0.997.

Table 8. Accuracy of cross-validate method for HTML and spreadsheets tables

HTML					
Stage	**K = 1**	**K = 2**	**K = 3**	**K = 4**	**K = 5**
Test	0,92	0,98	0,98	0,94	0,97
spreadsheets					
Stage	**K = 1**	**K = 2**	**K = 3**	**K = 4**	**K = 5**
Test	0,997	0,998	0,996	0,998	0,996

3.3.3. Confusion matrix

As in any classification problem, there are aspects that may be improved. In our experiments, we have to examine the rows in each class that were confused with rows in another class. We then used a confusion matrix, as shown Figure 9 and Figure 10. Each cell of the matrix shows the percentage of all classified rows that were actually of the class with the label shown in the first column, but which the classifier assigned the row label shown in the first row. The shaded cells with blue color stronger in the diagonal show correct row classifications, while the remainders show incorrect classification.

Ideally, our classifier would result in zeroes for the values off the diagonal. However, our model indeed misclassified rows. In the case of spreadsheets tables, we observed that, for both data rows and header rows, erroneous results were obtained with respect to the non-relational rows, that is, a considerable number of data rows and header rows were identified as non-relational rows. In the HTML tables, the erroneous results for non-relational rows was larger than for spreadsheet tables, being 7.9% and 6.6% for data rows and header rows, respectively.

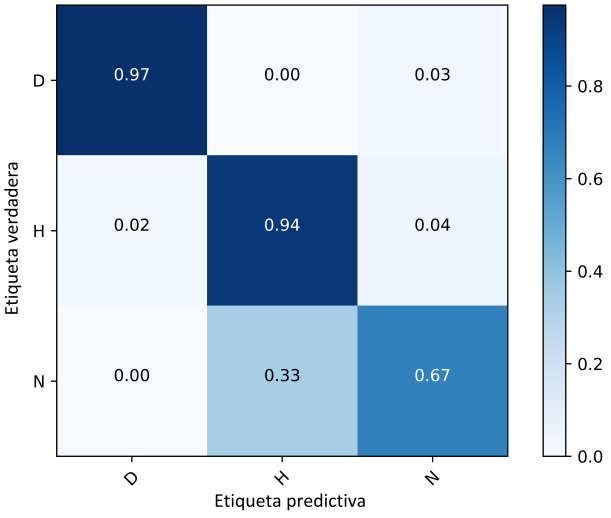


Figure 9. Confusion matrix for spreadsheets tables

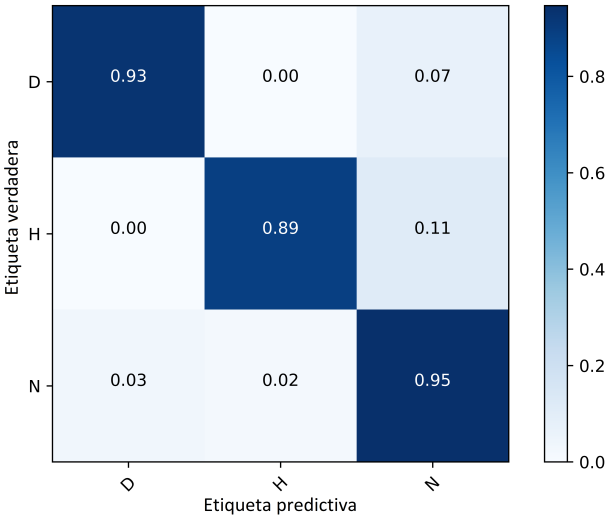


Figure 10. Confusion matrix for HTML tables

This deserves some explanation: (1) the difference between the average number of rows of the HTML tables and the average number of rows of spreadsheets tables; (2) in our classification process, a given row is classified as "non-relational metadata" when the row cannot be identified as data or header; (3) the spreadsheets tables have a better definition in term of features, for example, tables depend on properties encapsulated into CSS files.

3.4. Rule-based Table Detection Algorithm

The rule-based detection algorithm was applied to worksheets in a sample set that contained tables with different layouts and embedded charts. Table 9 summarizes the results obtained, which analyzed a total of 1,000 spreadsheet documents, detected 1,481 tables and misclassified 141.

Table 9. Results for the detection rule-based algorithm

Spreadsheet document	**1000**
Tables	1481
Tables misclassified	141
Single Table	700
Multi Table	158

The algorithm failed for multi-tables with internal separators that are less than the thresholds defined. In that case, the algorithm would consider the two tables as a single table. Also, would not recognize tables correctly when the table cells do not have attributes or separators (e.g, a table with no borders, not font formatting, no background colors, and no empty rows that separate headers and table title) and did not discover tables where the number of empty cells to the right and left is extremely large.

3.4.1. Experiments with the Remove and Conquer Table Detection Algorithm

The Remove and Conquer algorithm were applied to the same dataset. This algorithm detected tables that could not be recognized by the rule-based algorithm and maximized the match between a proposed table P and a true table T, which is equivalent to maximizing the number of cells that they have in common and minimizing the number of cells by which they differ. Table 10 shows the results if we compare with Algorithm 1, we can observe that the number of tables misclassified decreased and the number of multi-tables detected increased.

Table 10. Table recognized through of RAC

Spreadsheet document	1000
Tables	1481
Tables misclassified	141
Single Table	650
Multi Table	230

3.5. Results of the algorithms and environment description

Before entering details about the execution times of the algorithms let's explain the characteristics main of the environment: Portable Computer (PC) Lenovo 80YH model with 8 Gigabytes of Random Memory. Processor Intel(R) Core i7-7500 with 2.70 GHz. Board Graphic Intel(R) 620 with 128 of Memory. Operating System Windows 10 Home of 64 Bits. Table **??** shows the execution times each of the Algorithms.

Table 11. Execution Time to the Tables Recognition Algorithms

Algorithm	Execution time (s)	CPU (%)	Memory(%)
Remove and Conquer	114,28	4,3	1,3
Rule-based	69,19	3,7	1
Conditional Random Field	376,57	11,5	2,5

4. Conclusions

In this work we have described the implementation of three algorithms to classify rows of a table and recognize tables in spreadsheet documents respectively. We performed experiments to test the performance of the table row classifier using HTML and Spreadsheet tables. The experiments show that the classifier obtained excellent results for both kinds of tables. Also was applied a k-fold cross-validation where were obtained results similar to the other experiments reported in [20].

To summarize, the contributions of this work were:

- A table row classifier, applicable to both HTML and spreadsheet tables.
- Experiments to validate the classifier.
- Two datasets containing annotated HTML and Spreadsheets tables to train and validate table row classifiers.
- The implementation of two algorithms for table recognition in spreadsheet documents.

As future work, we propose to increment the number of instances and classes in our datasets and add more complex features. We expect that CRF can also be applied to other non-tabular classification tasks involving content of various formatting and layouts. In general, CRF may help constructing generic information extraction systems.

References

[1] M. Yakout, K. Ganjam, K. Chakrabarti, and S. Chaudhuri, "Infogather: Entity augmentation and attribute discovery by holistic matching with web tables," in *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '12. New York, NY, USA: Association for Computing Machinery, 2012, pp. 97–108. [Online]. Available: https://doi.org/10.1145/2213836.2213848

[2] M. J. Cafarella, A. Halevy, D. Z. Wang, E. Wu, and Y. Zhang, "Webtables: Exploring the power of tables on the web," *Proc. VLDB Endow.*, vol. 1, no. 1, pp. 538–549, Aug. 2008. [Online]. Available: https://doi.org/10.14778/1453856.1453916

[3] E. Koci, M. Thiele, O. Romero, and W. Lehner, "Table identification and reconstruction in spreadsheets," in *Advanced Information Systems Engineering*, E. Dubois and K. Pohl, Eds. Cham: Springer International Publishing, 2017, pp. 527–541.

[4] P. Venetis, A. Halevy, J. Madhavan, M. Paşca, W. Shen, F. Wu, G. Miao, and C. Wu, "Recovering semantics of tables on the web," *Proc. VLDB Endow.*, vol. 4, no. 9, pp. 528–538, Jun. 2011. [Online]. Available: https://doi.org/10.14778/2002938.2002939

[5] G. Limaye, S. Sarawagi, and S. Chakrabarti, "Annotating and searching web tables using entities, types and relationships," *Proc.*

VLDB Endow., vol. 3, no. 1–2, pp. 1338–1347, Sep. 2010. [Online]. Available: https://doi.org/10.14778/1920841.1921005

[6] T. F. Varish Mulwad and A. Joshi, "Generating Linked Data by Inferring the Semantics of Tables," in *Proceedings of the First International Workshop on Searching and Integrating New Web Data Sources*, September 2011, co-located with VLDB 2011. [Online]. Available: https://bit.ly/3p8s1q0

[7] A. S. Corrêa and P.-O. Zander, "Unleashing tabular content to open data: A survey on pdf table extraction methods and tools," in *Proceedings of the 18th Annual International Conference on Digital Government Research*, ser. dg.o '17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 54–63. [Online]. Available: https://doi.org/10.1145/3085228.3085278

[8] B. Yildiz, K. Kaiser, and S. Miksch, "pdf2table: A method to extract table information from pdf files." [Online]. Available: https://bit.ly/3k2ejBa

[9] Y. Liu, P. Mitra, and C. L. Giles, "Identifying table boundaries in digital documents via sparse line detection," in *CIKM '08*, 2008. [Online]. Available: https://bit.ly/369nWcm

[10] T. Kieninger, "Table structure recognition based on robust block segmentation," 1998, pp. 22–32. [Online]. Available: https://bit.ly/38k4YT9

[11] M. Zhang and K. Chakrabarti, "Infogather+: Semantic matching and annotation of numeric and time-varying attributes in web tables," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '13. New York, NY, USA: Association for Computing Machinery, 2013, pp. 145–156. [Online]. Available: https://doi.org/10.1145/2463676.2465276

[12] Z. Zhang, "Towards efficient and effective semantic table interpretation," in *The Semantic Web – ISWC 2014*, P. Mika, T. Tudorache, A. Bernstein, C. Welty, C. Knoblock, D. Vrandečić, P. Groth, N. Noy, K. Janowicz, and C. Goble, Eds. Cham: Springer International Publishing, 2014, pp. 487–502. [Online]. Available: https://doi.org/10.1007/978-3-319-11964-9_31

[13] H. Masuda and S. Tsukamoto, "Recognition of html table structure," 2004. [Online]. Available: https://bit.ly/3p8xL2Q

[14] J. Fang, P. Mitra, Z. Tang, and C. L. Giles, "Table header detection and classification," in *AAAI*, 2012. [Online]. Available: https://bit.ly/2IcT3vy

[15] D. Pinto, A. McCallum, X. Wei, and W. B. Croft, "Table extraction using conditional random fields," in *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, ser. SIGIR '03. New York, NY, USA: Association for Computing Machinery, 2003, pp. 235–242. [Online]. Available: https://doi.org/10.1145/860435.860479

[16] I. A. Doush and E. Pontelli, "Detecting and recognizing tables in spreadsheets," in *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, ser. DAS '10. New York, NY, USA: Association for Computing Machinery, 2010, pp. 471–478. [Online]. Available: https://doi.org/10.1145/1815330.1815391

[17] E. Koci, M. Thiele, W. Lehner, and O. Romero, "Table recognition in spreadsheets via a graph representation," in *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, 2018, pp. 139–144. [Online]. Available: https://doi.org/10.1109/DAS.2018.48

[18] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the Eighteenth International Conference on Machine Learning*, ser. ICML '01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 282–289. [Online]. Available: https://bit.ly/3lbW1yE

[19] J. L. Solé, *Book review: Pattern recognition and machine learning. Cristopher M. Bishop. Information Science and Statistics.* Springer, 2007. [Online]. Available: https://bit.ly/3l7doRq

[20] M. D. Adelfio and H. Samet, "Schema extraction for tabular data on the web," *Proc. VLDB Endow.*, vol. 6, no. 6, pp. 421–432, Apr. 2013. [Online]. Available: https://doi.org/10.14778/2536336.2536343