Information Extraction from Handwritten Tables in Historical Documents

José Andrés[1(✉)], Jose Ramón Prieto[1], Emilio Granell[1], Verónica Romero[2], Joan Andreu Sánchez[1], and Enrique Vidal[1]

[1] PRHLT Research Center, Universitat Politècnica de València, Valencia, Spain
{joanmo2,joprfon,emgraro,evidal}@prhlt.upv.es

[2] Departament d'Informàtica, Universitat de València, Valencia, Spain
veronica.romero@uv.es

Abstract. Recently, significant advances have been made in Document Understanding in structured historical documents. However, not much research has been done in information extraction from handwritten structured historical documents. In this paper, we compare two Machine Learning approaches and another approach that is based on heuristic rules to extract information in historical pre-printed forms with handwritten information. We analyze how each approach performs at each step of the extraction process. The proposed approaches improve the heuristic-rule baseline by up to 0.14 F-measure points throughout the information extraction pipeline.

Keywords: Structured handwritten documents · Information extraction · Neural networks

1 Introduction

In the last years, important advances have been achieved in Handwritten Text Recognition (HTR) of historical documents. The current prevalent techniques based on Deep and Recurrent Neural Networks (DRNN) [5] have largely contributed to these achievements. In this way, new problems related to HTR in large collections can now be explored. This paper researches the problem of information extraction from handwritten tables in historical documents.

Information Extraction from tables of historical documents is a very difficult problem given the laxity which the layout was usually considered in the past [8]. This problem is also present even for more recent printed documents for which some interesting solutions have been explored recently [10]. The interest in Information Extraction from historical handwritten tables is enormous since there exist large amounts of huge collections in which the information was registered in tabular format: border records, military registers, hospital

J. Andrés and J.R. Prieto have equally contributed to this paper.

records, records related to industrial processes, financial, population, forest, travel records, etc. Extracting the relevant information from these collections would allow researchers to study in more depth the past. Currently adopted solutions for extracting reliable information are mainly based on crowd-sourcing approaches. In these approaches, the crowd does not receive any automatic assistance.[1,2]

Two current limitations to get useful results in these collections are, on the one hand, the complex tabular layout that they have, and on the other hand, the current HTR results, which are not error-free.

The tabular layout can be very free in some situations with lines of running text from left side to right side of the image [19]. In this case, the line extraction process can be very good, and consequently, the HTR results and the extraction of relevant information can be good since they can rely on useful context [20]. The tabular layout can also be composed of pre-printed sheets [21]. Extracting row lines from side to side of the page image has been proven to be not very useful because the linguistic context along the columns in the same row may be not useful [21] and it seems better to extract lines at cell level. In such case, the line detection can be difficult since sometimes only quotes are written to indicate that the value of the preceding row is reproduced. In addition, the HTR results and the extraction of relevant information results cannot be very good since they cannot rely on the HTR context [21].

This paper researches on how to perform information extraction over a set of textlines given its geometric location. Three approaches have been considered for this task. The main idea in the two first ones is to use just the geometrical location to perform information extraction, while the last approach can also extract features from neighboring textlines [15].

The techniques that are researched in this paper are applied to the HisClima dataset [21] which is composed of logbooks of printed forms, and the tables are filled in with handwritten text. Although these forms have a fixed layout, the handwritten text does not strictly respect the cells. In addition, the tables contain difficult vocabulary related with winds, temperatures, atmospheric pressure, etc., which make the recognition process really difficult.

2 Related Work

In recent years, great advances have been made in Document Layout Analysis. Many of these advances have been possible thanks to neural networks, which have taken a leading role. Using a pre-trained encoder, [1] trained a generic deep learning architecture for various layout analysis tasks, such as text line detection and page extraction. In [4] they used Mask Convolutional and Recurrent Network (CRNN) detecting tables as an object detection problem; however, this work only focuses on detecting the table, not its structure. To deal with structure

[1] https://www.zooniverse.org/projects/krwood/old-weather-ww2.
[2] https://fromthepage.com/.

detection, Fully-Convolutional Neural Networks have been used to analyze the tables structures [22].

Graph Neural Networks (GNN) have also started to take on an important role in finding relationships in documents. [18] used a CNN to extract visual features, detecting rows, columns, and cells. Then, they classified the relationships among the different detected objects using GNNs.

However, none of these methods have been applied to handwritten images but assume printed images with high regularity and straightness.

Regardless, progress has been made in the field of Table Understanding *historical handwritten*. A competition was developed for detecting the structure of handwritten tables, [3] track B. One of the teams used a fully CNN and then constructed an adjacency matrix from the detected objects. [13] created a graph from the rows and classified each edge to find rows and columns, doing a subsequent connected component analysis on the initial pruned graph. Edges were classified as nodes in a conjugate graph, created as a pre-process of the initial graph of textlines. However, although the method is powerful, it has certain weaknesses as it depends on the initial graph. In [15], the initial graph was improved, making the method more robust. In this paper, GNN-based neural networks modify the latter research by making it less costly and complex, eliminating conjugate graphs for edge classification.

In the field of Information Extraction (IE) for handwritten historical tables, interesting researches have been published so far. Thus, [8,21] use geometry to select the rows and columns of already known headers. However, it only presents results on ground truth lines. This paper tests methods with automatically detected lines and compares them with those based on neural networks.

3 HisClima Dataset

The HisClima database is a freely available handwritten text database [14] compiled from logbooks of ships that sailed the Arctic ocean from July of 1880 until February of 1881. In this paper we only used one of the logbooks belonging to the Jeannette ship.

These logbooks documents are composed by two different kind of pages, some pages containing tables and other ones containing descriptive text. There is preprinted text in both type of pages. The Jeannette logbook follows this structure and it is composed of 419 pages, 208 correspond to table pages and the other 211 correspond to descriptive text.

Given that the most relevant information for researchers of this kind of documents is included in the tables, that contain a lot of numerical data, we focus only on table pages. These pages are divided into two parts: the upper part is for registering the information in the AM period of each day and the bottom part registers the information referred to the PM period of each day (see Fig. 1). The registered information is related to weather navigation conditions and is plenty of the particular jargon related to this problem: wind type, atmospheric pressure, etc.

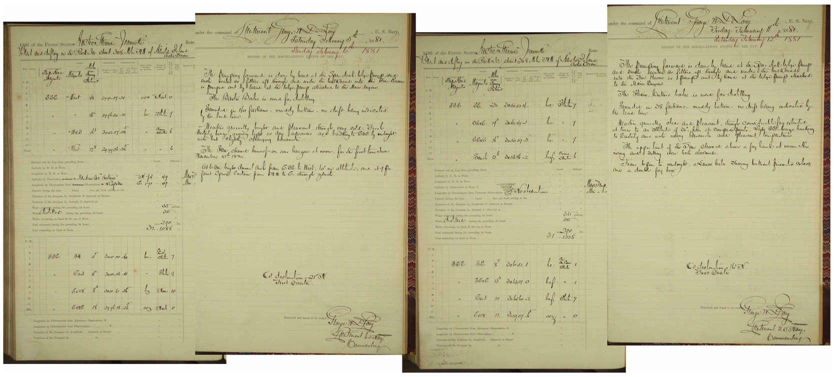

Fig. 1. Example of ship logs with annotation about the weather conditions.

These documents entail some challenges related with different areas such as layout analysis, handwritten recognition and information extraction. One of the main difficulties for automatic layout analysis comes from the fact that the information included in a cell sometimes is replaced with quotation marks (") when the data is the same as the data in the same column in the previous row (or some row above). These quotation marks are very short and sometimes they are difficult to automatically detect. Other layout difficulties that can be found are: data related with a cell that is really written in the upper and lower cells, crossed out column names, words written between cells, and different number of rows completed in every table.

The HisClima database has been endowed with two different types of annotations. First, the layout of each page was annotated to indicate blocks, columns, rows, and baselines. Second, the text was completely transcribed by an expert paleographer, including relevant semantic information. The lines in the table regions were marked at cell level, resulting in a total of 3 525 lines. In addition, the printed and the handwritten text have been labeled with different tags to differentiate between both types of text in the recognition process. Note that the

Table 1. Basic statistics of the HisClima database and average values for the three partitions.

Number of	Train	Validation	Test	Total
Pages	143	15	50	208
Lines	23 617	2 284	7 838	33 739
Running words	46 599	4 604	15 611	66 814
Lexicon	1 287	491	924	1 483
Character set size	76	76	76	76
Rel. information	10 917	1 021	3 533	15 471

printed text is more regular and easy to learn and therefore, it is important to provide separated results for both types of text.

In this paper, the partitions defined in [21] have been used. The main figures are shown Table 1. The last row, *Rel. Information*, shows the number of cells in the tables that contain relevant information.

4 Proposed Approaches

In this section, we will discuss the different approaches that have been employed to face the information extraction task. We consider the problem of extracting textual information in hybrid printed/handwritten tables. We wish to be agnostic with respect to possibly predefined table layouts, but we assume tables to be organized into orthogonal rows and columns. Each column typically has at least one column header h_c, and each row has a row header h_r. The textual content of interest v is contained in cells which are the intersection of columns and rows.

Moreover, to perform information extraction over a tabular image x, a set of possible column headers (one per column) and row headers (one per row) are (semi-automatically) extracted from the ground-truth transcripts of the training table images. In addition, we assume that the regions of x matching h_c, h_r and v are small bounding boxes (BBs), b_c, b_r and b_v respectively, each tightly containing the corresponding textual contents.

The following approaches take as input the extracted textlines and their 1-best transcripts. As output, they return the textual contents of interest v associated to each tuple (h_c, h_r). A diagram showing the necessary steps to go from images to a database filled with the extracted information can be seen in Fig. 2.

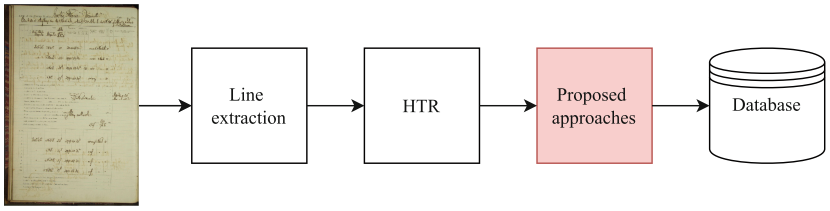

Fig. 2. Overall diagram of the methodological pipeline. It is highlighted in red the phase which accounts for the information extraction techniques described in this section. (Color figure online)

4.1 Heuristic Geometric Information

This approach follows the information retrieval method presented in [21]. This method is based on the geometrical information of a structured multi-word query and the 1-best transcript of the detected lines in a table page.

The retrieval process is carried out in three steps for each table page. First, every column header (h_c) word is retrieved. Second, the row header (h_r) words are retrieved by looking for all the lines that appear below the first column region, delimited by the horizontal span of the column-heading word line. Then, every cell-content word is searched by the combination of the corresponding column and row regions. Finally, for those cells whose content is a double quotation marks, it is replaced by the content of the first previous cell different to the double quotation marks.

4.2 Log-Linear Model

As an alternative technique to solve this problem and building on the foundations of the previous approach, we propose making use of the geometric information employing a combination of robust machine learning models.

In this method, we assume that the structure of a table is given by the positions of its column headers, row headers and cells of interest. Nevertheless, two problems should be addressed before starting to perform information extraction: the multi-line cells and the quotation marks. As discussed before in Sect. 3, there are cells whose content is spread across multiple lines. This is a problem since, when we perform information extraction, we are interested in extracting the whole cell. Similarly, we aim at extracting the textual content that a quotation marks symbol represents, not the quotations themselves. Therefore, this approach entails three consecutive steps: grouping the textlines that belong to the same semantic cell, substituting the quotation marks for the content they represent, and finally, performing information extraction over the resulting groups of textlines. This pipeline can be seen in Fig. 3.

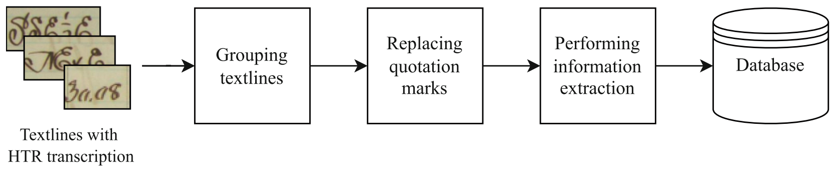
Textlines with HTR transcription
Grouping textlines
Replacing quotation marks
Performing information extraction
Database

Fig. 3. Pipeline of the log-linear model approach. In the first phase, the textlines are grouped into semantic cells. Then, the quotation marks are replaced by the textual content they represent. Finally, information extraction is performed over the resulting set of semantic cells.

Grouping Textlines. This phase can be divided in two steps. In the first step, we group two textlines b_v and $b_{v'}$ when their probability of being in the same cell is greater than a given threshold t_1. To express this formally, we introduce the binary random variable C, which denotes if b_v and $b_{v'}$ belong to the same semantic cell. Therefore, we will group b_v and $b_{v'}$ when $P(C = 1 \mid b_v, b_{v'}) \geq t_1$.

To learn this probability, an MLP has been employed. As input features we have considered the x coordinate of b_v, the absolute distance between b_v and

$b_{v'}$ in the y-axis and in the x-axis. With the use of the two first attributes, the system learns which is the expected distance in the y-axis between two textlines that belong to the same cell while taking into account their position in the x-axis. Moreover, with the last considered attribute, the system learns which deviation can be expected in the x-axis. Note that the multi-line cells appear only in a concrete image region in this collection. Therefore, the allowed distance in the y-axis in that image region will be larger than in other regions.

Once this first step is finished, we will have the textlines grouped in semantic cells. However, this procedure could leave us with textlines appearing in multiple cells simultaneously, as the network might have trouble grouping the most distant textlines. To overcome this problem, the second step of this phase consists in merging two groups of textlines if they share at least a textline. An example is shown in Fig. 4. The output of this second step is the final set of semantic cells.

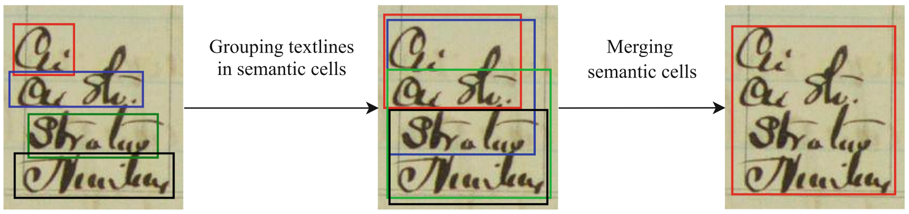

Fig. 4. Example showing the two steps that conform the grouping textlines procedure. At the beginning, we have four different textlines. Then, we apply the first step and group the textlines into semantic cells. As can be seen in the middle figure, the first step might fail to group the most distant textlines. However, when we perform the second step, the semantic cells that share at least one textline are merged, leading us to the semantic cell shown in the last figure.

Replacing Quotation Marks. In this phase, our objective is to substitute the quotation marks for the textual contents they represent. Note that every quotation mark has a unique textual content it refers to. Therefore, we replace the textual content of each quotation marks BB, b_q, with the textual content of the semantic cell, b_v, with largest probability of being the precedent cell of b_q (and consequently, the cell b_q is referring to). This probability is denoted as $P(S = 1 \mid b_v, b_q)$, where S is a binary random variable that denotes if b_v is the semantic cell to which b_q is referencing.

This probability has been approximated by means of an MLP, where the distance in the y-axis and the absolute distance in the x-axis between b_v and b_q have been adopted as input features. By doing this, the model learns at which distance in both axes the referenced cell b_v is expected to be found. It is worth mentioning that the textual contents of b_v might also be another quotation marks. In that case, we would repeat the replacing quotation marks process but employing the previous b_v as current b_q, until achieving a b_v whose textual contents are not a quotation marks. The final output of this phase is the set of semantic cells without quotations.

Performing Information Extraction. Finally, after grouping the textlines and replacing the quotation marks by the textual content they represent, we can perform information extraction over the resulting semantic cells.

In this task, we extract the textual content of interest v when its probability of being relevant to a header tuple $(h_c,\ h_r)$ is larger than a given threshold t_2. This is formally expressed as extracting v when $P(\mathcal{R} = 1 \mid h_c, h_r, v) \geq t_2$, where $\mathcal{R}$ is a binary random variable which indicates if v is relevant to $(h_c,\ h_r)$. Note that, following the ideas presented in [24], $P(\mathcal{R} = 1 \mid h_c, h_r, v)$ can be approximated as:

$$P(\mathcal{R} = 1 \mid h_c, h_r, v) \approx \max P(\mathcal{R} = 1 \mid h_c, h_r, v, b_c, b_r, b_v) \quad (1)$$

To model this distribution we have utilized a combination of probabilities, where each of them accounts for a different structural component of a table. They are the following:

The first probability is $P(H_c = 1 \mid b_c)$, where H_c is a binary random variable that denotes if a semantic cell b_c is classified as a column header. The second considered probability is $P(H_r = 1 \mid b_r)$, where H_r is a binary random variable that denotes if b_r is classified as a row header. The third probability that has been used is $P(A_c = 1 \mid b_c, b_v)$, where A_c is a binary random variable that denotes if cell b_v is aligned with a given column header b_c. Finally, the last probability distribution that has been considered is $P(A_r = 1 \mid b_r, b_v)$, where A_r is a binary random variable that denotes if a semantic cell b_v is aligned with respect to a given row header b_r.

With the purpose of modeling these probabilities, different attributes have been considered. For $P(H_c \mid b_c)$, the x-coordinate of b_c has been employed. Analogously, the y-coordinate of b_r has been considered for $P(H_r \mid b_r)$. This choice of attributes is due to the fact that column headers are expected to appear at the top of the page, while row headers are expected to appear at the left side of the table. Moreover, the absolute distance in the x-axis between b_c and b_v has been chosen as attribute for $P(A_c \mid b_c, b_v)$, while the homologous distance in the y-axis has been utilized for $P(A_r \mid b_r, b_v)$. When employing these attributes, $P(A_c \mid b_c, b_v)$ learns which is the expected deviation in the x-axis between two BBs which are considered to be in the same column, while $P(A_r \mid b_r, b_v)$ learns the expected deviation in the y-axis when two BBs are considered to be in the same row. To learn each of these probabilities, a Perceptron has been used.

With the purpose of combining these probabilities, the log-linear framework has been followed. This framework has been used with great success in diverse fields such as statistical machine translation [11] and automatic speech recognition [6] among others. When using it in this context, we find a set of 4 feature functions, $g_1, ..., g_4$, which we want to combine taking into account their corresponding weight, $\lambda_1, ..., \lambda_4$. Note that as feature functions, we have utilized the four previously described probabilities. It is worth mentioning that, in this task we are interested in the textual content of interest v when $P(\mathcal{R} \mid h_c, h_r, v) \geq t_2$, and not necessarily in its actual probability. Therefore, we could get rid of the normalization term of a log-linear model, leading us to:

$$f(\mathcal{R}, h_c, h_r, v, b_c, b_r, b_v) = \exp(\sum_{m=1}^{4} \lambda_m g_m(\mathcal{R}, h_c, h_r, v, b_c, b_r, b_v))$$

Finally, the whole information extraction process described in this approach could be summarized as follows: Firstly, we group the textlines into semantic cells. Secondly, we substitute the quotation marks with the content they represent. Thirdly, employing the list of predefined column headers and row headers described in Sect. 4, we calculate the score of each textual content of interest v for each possible tuple (h_c, h_r). When the achieved score of v for a tuple is greater than a given threshold t_2, we extract v.

4.3 Graph Neural Network

Techniques based on neural network graphs (GNN) have been used to extract sub-structures of each table (rows and columns) from previously detected textlines. The first step is to create a graph by connecting the textlines by their line of sight (LoS) and, in addition, using the improvements on the graph of [15]. These techniques result in a graph depending on the sub-structure to be detected. Then, a score in $[0, 1]$ is obtained for each edge, which can be interpreted as an estimate of the probability that the corresponding edge joins two elements of the same sub-structure.

In this work, the GNN-based techniques of [15] have been slightly modified. Instead of using the conjugate graph, the raw graph is used directly to classify edges using Eq. 2. This result in a more memory-efficient method, with no preprocessing (conjugation) and faster to train since conjugation treats each edge as a node, and this makes it much more expensive when the number of edges grows. In this work, the number of nodes is kept the same, growing the number of edges but avoiding conjugation.

In order to obtain the classification of the edges between the origin (s) and destination (d), Eq. 2 is used, being $\boldsymbol{x}'_s$ and $\boldsymbol{x}'_d$ the embeddings associated to the nodes calculated by the GNN, from origin and destination respectively, and $\boldsymbol{e}_{s,d}$ edge properties given these node embeddings. From $\boldsymbol{x}'_s$ and $\boldsymbol{x}'_d$ another vector is obtained from the result of the absolute value of the subtraction of each component, expressed as $|\boldsymbol{x}'_s - \boldsymbol{x}'_d|$ in the equation. $\boldsymbol{e}_{s,d}$ is the value calculated at the time of creating the graph and remains unchanged. The vector function $Q : \mathbb{R}^{\phi} \times \mathbb{R}^{\gamma} \rightarrow \mathbb{R}$ is implemented as an MLP with a sigmoid activation function, therefore the results can be interpreted as a probability of the edge between s and d nodes, and trained together with the GNN as a whole network.

$$\hat{y}_{s,d} = Q(|\boldsymbol{x}'_s - \boldsymbol{x}'_d|, \boldsymbol{e}_{s,d}) \tag{2}$$

At the same time, with another GNN the textlines headers are detected. In this case, each textline, which is equivalent to a node in the original network, is classified as a binary header detection problem.

With all the information from the three trained GNNs (row, column and header detection), a process is performed to obtain, for each existing cell, the row and the column to which it belongs, as well as the header of that column.

Note that with these GNN-based methods, multi-line cells are naturally detected by intersecting rows and columns, resulting in a cell with all the lines in it.

5 Evaluation Criteria and Metrics

The transcripts quality obtained by the HTR system is assessed using the Levenshtein edit distance [9] with respect to the reference text, at both the word and the character levels. The Character Error Rate (CER) is the Levenshtein edit distance at character level and it can be defined as the minimum number of substitutions, deletions and insertions needed to transform the transcription into the reference text, divided by the number of characters in the reference text. Similarly, Word Error Rate (WER) is this edit distance calculated at word level.

The performance of the different information extraction approaches has been assessed employing the precision (P), recall (R) and its harmonic mean (F_1). In this context, the precision denotes the percentage of the retrieved cells that are correct according to the GT, while the recall denotes the proportion of relevant cells that has been retrieved from all the relevant cells. Finally, the F_1 accounts for the balance between both metrics.

It is worth noting that this task has been evaluated at cell level. Therefore, a match is considered a *true positive* (TP) when the content retrieved by the system is exactly the same as the content found in the GT for that cell. Otherwise, a match is considered a *false positive* (FP). Moreover, when the contents of a cell in the GT are not retrieved, this is considered as a *false negative* (FN).

Finally, we would like to remark that 95% confidence intervals ($\alpha = 0.025$) have been calculated using the bootstrap method with 10 000 repetitions [2].

6 Experimental Framework and Results

Different experiments have been performed to assess the text recognition and the proposed information extraction approaches. In the following sections, we will describe the experimental framework that has been employed and the results that have been achieved.

6.1 Experimental Settings

Handwriting Text Recognition. The text recognition system used in this work is based on the technology described in [16] and is implemented by using the PyLaia [17] toolkit. The input images were pre-processed to correct the slope and inclination.

Optical models are based on Convolutional and Recurrent Neural Networks (CRNN), consisting of a convolutional block and a recurrent block. This model consists of three convolutional layers with filters composed of different maps of characteristics (16, 32 and 48) with kernel sizes of 3×3 pixels and horizontal and vertical dilations of 1 pixel. *LeakyReLU* is used as activation function, and the output of the convolutional layers is fed to a layer of *max pooling* with non-overlapping of 1×1 pixels, only at the output of the first two layers. After that, the recurrent block is made up of three recurrent layers composed of 256 bidirectional long-short term memory units. Finally, a fully connected linear layer is used after the recurrent block. All hyper-parameters, such as the number of convolutional and recurring layers were configured in the validation set. It is important to remark that the optical models for the printed characters and for the handwritten characters were the same.

A language model was estimated as 3-gram of characters directly from the transcripts of the included lines of text on the training and validation partitions using the SRILM [23] toolkit. Just a language model was used, both for the printed part and for the handwritten part.

As input for the three different approaches that are described below, the 1-best transcript obtained using the previously described HTR system has been employed.

Heuristic Geometric Information. In the case of the heuristic geometric information method, the geometric position of every detected line in the page was used.

Log-Linear Model. MLPs and Perceptrons have been employed to tackle this problem. Concretely, the employed MLPs are formed by one hidden layer of 128 units plus the final layer for binary classification.

To optimize every network, we have employed Adam [7] solver with a learning rate of 0.01. To handle class imbalance, a version of weighted cross-entropy has been used as a loss function. In this version, weights per class were computed as $w_k = \frac{s_m}{s_k}$, where s_k denotes the number of samples of the class k and s_m denotes the number of samples of the majority class. As batch size, we have employed 256. As metric for choosing the best model of each network during training, we have employed the F_1 score obtained over the validation dataset.

Finally, the weights of the log-linear model and the employed threshold in the grouping textlines phase (t_1) have been estimated jointly employing the Powell Search algorithm [12], while the threshold used to associate semantic information to a textual content of interest (t_2) has been estimated as the optimal threshold obtained when calculating the F_1 score over the validation dataset.

Graph Neural Network. We have used the same configuration for the three cases (rows, columns and headers), with four layers of 64 filters each in the first steps of the GNN and, finally, the MLP composed of four layers of 64 neurons each, plus the final layer of binary classification.

To optimize the GNNs we used minibatch SGD and Adam solver [7] with a learning rate of 0.01. To reduce false positives, which have a very detrimental effect, a version of weighted cross-entropy has been used as a loss function. In this version, cross-entropy values of the negative class are multiplied by a weight $w_0 = \frac{\alpha}{log \epsilon + p_0}$, $\epsilon \geq 0$, where p_0 is the prior-probability of the negative class, after some tests, the hyper-parameter α has been set to 5 in all experiments. The networks were trained for 6000 epochs. To consider an edge as positive, a threshold of 0.95 has been used on $\hat{y}_{s,d}$, minimizing false positives.

In order to improve the created graph we used, in the case of columns σ_1 were set to 1 and σ_2 to 10 and the other way round in the case of columns. To detect headers we used the original graph.

Line Detection. Line detection was carried out with MaskRCNN implemented with Detectron2 tool [25].

6.2 Text Recognition Results

Table 2 presents the obtained text recognition results for the test partition. As it can be observed, the error in the overall results is quite low (CER = 2.0% and WER = 4.4%). But as noted in previous sections, in this type of experiments is quite relevant to distinguish between the errors in the printed part and in the handwritten part. The GT was annotated with this information. We can observe that the recognition of the handwritten text (CER = 5.7% and WER = 10.4%) represents a greater challenge than the printed text (CER = 1.5% and WER = 1.8%). These results are reported with the GT, which means that all annotated lines were used to compute these values.

Table 2. Results of text recognition. 95% confidence intervals are never larger than 1.1% for manuscript text, 0.4% for printed text and 0.5 % overall.

Text type	Manuscript	Printed	Overall
CER	5.7%	1.5%	2.0%
WER	10.4%	1.8%	4.4%

6.3 Information Extraction Results

Table 3 reports the results obtained when extracting information from the GT lines and from the automatically extracted lines, using the HTR system previously described in both cases. Note that the former experiment simulates the scenario in which a perfect line detector is used. Therefore it can be considered optimistic. The latter experiment simulates a real scenario.

In order to qualify the results obtained by our approaches, we are interested in knowing which is the maximum performance that could be achieved if our systems made no errors obtaining the structure of columns, rows and headers. These results are denoted as oracle in Table 3. On the one hand, when employing GT lines, the only errors present in the oracle are the ones corresponding to HTR. On the other hand, another source of errors is added when employing automatic lines, which is a possible bad line detection. Oracle/Automatic cells are indirectly measuring the error detection line.

Table 3. Information extraction results. 95% confidence intervals are never larger than 0.01 when using GT lines and 0.02 when employing automatic lines.

Lines	Ground thruth			Automatic		
Metric	P	R	F_1	P	R	F_1
Heuristic geometric information	0.79	0.78	0.78	0.64	0.55	0.59
Log-linear model	0.87	0.79	0.83	0.77	**0.69**	**0.73**
Graph neural network	**0.88**	**0.83**	**0.85**	**0.78**	0.67	0.72
Oracle	0.89	0.88	0.89	0.79	0.72	0.76

Results in Table 3 show that the best performance using GT lines has been achieved by the GNN, reaching 0.85 out of the 0.89 possible F_1 points, while the best results with automatic lines have been achieved by the log-linear model, which has obtained 0.73 out of the 0.76 possible F_1 points. Nevertheless, it is worth noting that the difference in F_1 between both approaches is very narrow and therefore, a similar performance in practice is expected.

Moreover, it can be seen that the worst performance has been achieved by the heuristic geometric information approach, achieving 0.78 F_1 points and 0.59 F_1 points when employing GT and automatic lines respectively.

Finally, we would like to remark that if we compare the oracle results with GT and automatic lines, the automatic extraction leads us to a loss of 0.13 F_1 points.

7 Discussion

First of all, it is clear that there is a significant difference in performance between the heuristic approach and the other two employed techniques. This difference is due to the fact that this method is not able to group textlines, and therefore, it is not able of retrieving correctly any multi-line cell. The neural networks methods are able of joining textlines when they belong to the same cell, and therefore have obtained significantly better results. As mentioned before, a similar performance has been achieved by these methods.

The log-linear model approach is much less computationally expensive than the GNN method, as it is formed by simple neural networks, such as MLPs and Perceptrons. However, it presents a major drawback: it needs to choose the appropriate attributes to model the different distributions that have been employed, and therefore, a study should be carried out to choose the appropriate attributes for each new dataset.

On the other hand, the GNN method is much more computationally expensive than the other approaches. Nevertheless, it presents a much wider generalization capability, allowing this approach to be employed in other layout understanding tasks. Moreover, this generalization ability lets the GNN to automatically learn the appropriate attributes without having to manually select them.

8 Reproducibility

We hope this work will help to boost research on how to perform information extraction over handwritten historical tables. For this reason, the source code used in the experiments presented in this paper is freely available on https://github.com/PRHLT/table-ie-das2022, along with the subsets used as training, validation, test[3].

9 Conclusions

This paper has shown a research on information extraction on handwritten historical tabular documents. Two of the employed approaches are based on Machine Learning techniques while the other is based on heuristic rules. Empirical results show that the best results are obtained by the Machine Learning approaches, achieving results close to the maximum reachable considering the HTR and line detection errors.

Taking this information into account, future efforts should be focused on improving the automatic line extraction and the HTR transcriptions. A possible solution to overcome HTR errors is to use *probabilistic indices* [24] instead of the 1-best transcripts given that, where the HTR fails, the *probabilistic indices* might have other accurate hypotheses. As another possible solution to face this issue, an end-to-end system which performs the line detection, HTR transcription and information extraction could be developed.

Acknowledgement. Work partially supported by the Universitat Politècnica de València under grant FPI-I/SP20190010 (Spain), by Generalitat Valenciana under project DeepPattern (PROMETEO/2019/121), by grant PID2020-116813RB-I00a funded by MCIN/AEI/10.13039/501100011033, by the Generalitat Valenciana under the project GV/2021/072, by grant RTI2018-095645-B-C22 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe".

References

1. Ares Oliveira, S., Seguin, B., Kaplan, F.: DhSegment: a generic deep-learning approach for document segmentation. In: Proceedings of International Conference on Frontiers in Handwriting Recognition, ICFHR, pp. 7–12 (2018)
2. Bisani, M., Ney, H.: Bootstrap estimates for confidence intervals in ASR performance evaluation. In: ICASSP, vol. 1, pp. 409–412 (2004)
3. Gao, L., et al.: ICDAR 2019 competition on table detection and recognition (cTDaR). In: Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, pp. 1510–1515 (2019)
4. Gilani, A., Qasim, S.R., Malik, I., Shafait, F.: Table detection using deep learning. Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, vol. 1, pp. 771–776 (2017)
5. Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., Schmidhuber, J.: A novel connectionist system for unconstrained handwriting recognition. IEEE TPAMI **31**(5), 855–868 (2009)

[3] https://doi.org/10.5281/zenodo.4106887.

6. Heigold, G.: A log-linear discriminative modeling framework for speech recognition (2010)
7. Kingma, D.P., Ba, J.L.: Adam: a method for stochastic optimization. In: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings (2015)
8. Lang, E., Puigcerver, J., Toselli, A.H., Vidal, E.: Probabilistic indexing and search for information extraction on handwritten German parish records. In: 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), pp. 44–49 (2018). https://doi.org/10.1109/ICFHR-2018.2018.00017
9. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. Sov. Phys. Dokl. **10**(8), 707–710 (1966)
10. Lin, W., et al.: ViBERTgrid: a jointly trained multi-modal 2D document representation for key information extraction from documents. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12821, pp. 548–563. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86549-8_35
11. Och, F.J., Ney, H.: Discriminative training and maximum entropy models for statistical machine translation. In: Proceedings of the 40th Annual meeting of the Association for Computational Linguistics, pp. 295–302 (2002)
12. Powell, M.J.: An efficient method for finding the minimum of a function of several variables without calculating derivatives. Comput. J. **7**(2), 155–162 (1964)
13. Prasad, A., Dejean, H., Meunier, J.L.: Versatile layout understanding via conjugate graph. In: Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, pp. 287–294 (2019)
14. PRHLT: HisClima dataset, October 2020. https://doi.org/10.5281/zenodo.4106887
15. Prieto, J.R., Vidal, E.: Improved graph methods for table layout understanding. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12822, pp. 507–522. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86331-9_33
16. Puigcerver, J.: Are multidimensional recurrent layers really necessary for handwritten text recognition? In: ICDAR, vol. 01, pp. 67–72, November 2017
17. Puigcerver, J., Mocholí, C.: Pylaia (2018). https://github.com/jpuigcerver/PyLaia
18. Qasim, S.R., Mahmood, H., Shafait, F.: Rethinking table recognition using graph neural networks. In: Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, pp. 142–147 (2019)
19. Romero, V., et al.: The ESPOSALLES database: an ancient marriage license corpus for off-line handwriting recognition. Pattern Recogn. **46**(6), 1658–1669 (2013)
20. Romero, V., Fornés, A., Granell, E., Vidal, E., Sánchez, J.A.: Information extraction in handwritten marriage licenses books. In: Proceedings of the 5th International Workshop on Historical Document Imaging and Processing, HIP 2019, pp. 66–71 (2019)
21. Romero, V., Sánchez, J.A.: The HisClima database: historical weather logs for automatic transcription and information extraction. In: ICPR (2020)
22. Siddiqui, S.A., Fateh, I.A., Rizvi, S.T.R., Dengel, A., Ahmed, S.: DeepTabStR: deep learning based table structure recognition. In: Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, pp. 1403–1409 (2019)
23. Stolcke, A.: SRILM-an extensible language modeling toolkit. In: Proceedings of the 3rd Annual Conference of the International Speech Communication Association (Interspeech), pp. 901–904 (2002)
24. Vidal, E., Toselli, A.H., Puigcerver, J.: A probabilistic framework for lexicon-based keyword spotting in handwritten text images. arXiv preprint arXiv:2104.04556 (2021)
25. Wu, Y., Kirillov, A., Massa, F., Lo, W.Y., Girshick, R.: Detectron2 (2019). https://github.com/facebookresearch/detectron2