Pattern Recognition Letters 172 (2023) 128–136

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Information extraction in handwritten historical logbooks

Jose Ramón Prieto [a,*], José Andrés [a,b,c], Emilio Granell [a], Joan Andreu Sánchez [a], Enrique Vidal [a,b]

[a] Pattern Recognition and Human Language Technology Research Center, Universitat Politècnica de València, València 46022, Spain
[b] Valencian Graduate School and Research Network of Artificial Intelligence, Camí de Vera s/n, Valencia 46022, Spain
[c] tranSkriptorium AI, Valencia, Spain

ARTICLE INFO

Article history:
Received 1 August 2022
Revised 4 May 2023
Accepted 7 June 2023
Available online 10 June 2023

Edited by: Maria De Marsico

Keywords:
Structured handwritten documents
Information extraction
Neural networks

ABSTRACT

Document Image Understanding is a demanding Pattern Recognition problem that requires complex recognition models. This problem is even more difficult for document images with complicated layouts like tables, where the reading order is often intrinsically ambiguous, and consequently, the context is generally ambiguous as well. In this paper, we compare two machine learning approaches for extracting information in pre-printed historical tables with handwritten information. We analyze the performance of each approach at each step of the extraction process over different corpora, up to a realistic scenario where documents with different table layouts written by different hands are used. The results are good in general and show that a model based on Multilayer Perceptrons yields better results on more homogeneous documents, while another model based on Graph Neural Networks generalizes better on heterogeneous corpora.

1. Introduction

Nowadays, archives all over the world store vast amounts of historical tabular documents that remain untranscribed. These documents contain relevant information of all kinds, such as marriage, birth and death registers, notarial data, border records, travel records, logbooks, etc. These documents are interesting if they are processed as a whole since they can provide relevant information about migration movements, social changes, trade movements, the evolution of climate, economic changes, etc. Therefore, the task of extracting all or most information from these documents becomes of great interest. A remarkable characteristic of the above mentioned documents is that they use to be heavily structured as tabular information.

Currently adopted solutions to extract large amounts of reliable information from tabular document images are mainly based on crowd-sourcing. In this approach, the crowd volunteers do not receive any automatic assistance and the amount of extracted information is often limited to specific fields of the text images[1]. Nevertheless, given the large number of collections of this kind that exist, it is relevant to research on how to locate and extract automatically information from tabular images. This paper researches on automatic information extraction from tabular images using neural networks both for Document Layout Analysis and Handwritten Text Recognition.

Information extraction from tabular images has been considered in recent papers [1–3]. Another study on this problem was presented in [4], where the performance of three different information extraction techniques was assessed over a realistic scenario, that is, when using automatically extracted textlines transcribed with Handwritten Text Recognition (HTR) techniques. In that study, it was shown that machine learning techniques clearly outperformed the heuristic approach. Those researches make clear the need of using structured automatic methods that take into account the context to decrease both the layout analysis errors and the text recognition errors.

The main challenge we address is the automatic extraction of information from historical handwritten tabular documents with different (yet similar) layouts and writing styles along the structured methods mentioned above. A preliminary version of this research, that exhibited promising results, was introduced in [4]. Nevertheless, we observed several limitations in that research, namely: handling multi-span cells, different column header representations, and generalization to different corpora. This paper ex-

* Corresponding author.
E-mail addresses: joprfon@prhlt.upv.es (J.R. Prieto), joanmo2@prhlt.upv.es (J. Andrés), emgraro@prhlt.upv.es (E. Granell), jandreu@prhlt.upv.es (J.A. Sánchez), evidal@prhlt.upv.es (E. Vidal).

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

https://doi.org/10.1016/j.patrec.2023.06.008

tends [4] by addressing the mentioned limitations. We introduce an effective way to deal with multi-span cells based on a simple classification task, and also a graph neural network is presented to deal with multi-span cells. In addition, to handle the different column header representations, a text classifier is used.

We also extend the experiments to a larger dataset in which the document images look similar but there are several table types that make the layout analysis more complex since column headers, column width, or the number of columns can be different. We focus on two corpora from the HisClima dataset: Jeannette and Albatross [5]. These corpora consist of page images from Arctic logbooks dated from 1880 to 1920, containing both tabular pages and descriptive text. The tables, characterized by printed headers and handwritten content cells, detail weather and navigation conditions. A key aspect of this dataset, and a reason why this dataset is chosen, is the various layout challenges it presents, such as multi-span column headers, different table layouts, use of quotation marks, data spanning multiple cells, crossed-out column names, vertical texts, and handwritten headers.

The rest of this paper is organized as follows: Sec. 2 reviews relevant papers that consider the task at hand. Next, Sec. 3 provides details about the considered corpora and the challenges they pose. Then, our approaches to extract information from these documents are explained in Sec. 4, followed by an introduction of our evaluation criteria and metrics in Sec. 5. Next, the experimental framework and the obtained results are presented in Sec. 6, and a discussion of them is found in Sec. 7. Finally, conclusions are drawn in Sec. 8.

2. Related work

In recent years, significant advances have been made in Document Layout Analysis, most of them due to the irruption of neural networks, which have adopted a predominant role.

In [6] Mask Convolutional and Recurrent Networks were used to detect tables as an object detection problem; however, that work only focused on detecting the tables, not their structure. To deal with structure recognition, fully-connected Convolutional Neural Networks (CNN) were used in [7].

In order to find relationships in structured or semi-structured documents, Graph Neural Networks (GNN) have also started to take on an important role. In [8] a CNN was adopted to extract visual features, detecting rows, columns, and cells. Then, GNNs were used to classify the relationships among the different detected objects.

However, none of these methods have been applied to handwritten images and all assume printed images with high regularity and straightness. Regardless, progress has been made in the field of *historical handwritten* Table Understanding.

A competition was set up in track B of [2] for detecting the structure of handwritten tables. One of the teams used a fully-connected CNN and then constructed an adjacency matrix from the detected objects. In [9] a graph was created from the rows and each edge was then classified to find rows and columns through connected component analysis. Edges were classified as nodes in a conjugate graph, created from an initial graph of textlines. While this method is powerful, it has certain weaknesses as it heavily depends on the initial graph. In [10], the initial graph was improved, making the method more robust.

In the field of Information Extraction (IE) from handwritten historical tables, interesting works have been also published. In [3,11] geometry was used to select the rows and columns of already known headers. However, it only presents results on ground truth lines.

In [12], a pipeline of information extraction in handwritten tables was presented. However, it relied on the fact that the used corpus only has one type of table, being very homogeneous and without variations in layout. This made it possible to learn a different language model per column, as well as to detect an entire line per row and cut it regularly. Unfortunately, no results on information extraction performance are reported.

Another pipeline for extracting information from handwritten tables was presented in [4]. In that pipeline, three different information extraction techniques were assessed. These techniques did not depend on always having the same table layout but could work with very different table layouts. However, it expected the textual contents of the headers to be known in advance in order to perform information extraction. Moreover, these systems were only assessed on a single corpus.

Table 1
Basic statistics of the Jeannette and Albatross corpus.

Corpus	Jeannette			Albatross		
Partition	Train	Val-	Test	Train	Val-	Test
Pages	143	15	50	52	7	25
Lines	23 614	2 282	7 838	19 871	2 538	9 138
Rel. Information	10 923	1 014	3 561	14 123	1 764	6 420

3. The hisclima dataset

As previously mentioned, we focus on the HisClima dataset, which is derived from the logbooks of two ships: Jeannette and Albatross. The tables of these corpora are divided into upper (AM) and bottom (PM) parts, featuring printed headers and handwritten content cells. They mainly contain weather and navigation conditions, such as wind directions and atmospheric pressures. A more comprehensive description can be found in [5], and here we detail some aspects related to this paper.

These documents exhibit several layout difficulties: use of quotation marks to avoid repeated writing the contents of precedent cells in the same column; data of a cell that is actually written in vertically or horizontally adjacent cells; crossed out column names; different number of rows completed in every table; texts in vertical; headers with handwritten contents; multi-span column headers; and different table layouts. Some examples of these challenges can be seen in Fig. 2.

The Jeannette corpus uses a single table layout and was written by a single writer. In this logbook, climatological attributes were typically written every three hours, leaving therefore several table rows empty. In addition, it is typical to find multi-line cells in this corpus (see Figs. 2 c) and 5). The Albatross set is formed by pages from seven different logbooks. Furthermore, we would like to remark that the climatological attributes were typically annotated every hour in this corpus, and in contrast with Jeannette, most of the tables are completely filled up. Also, we would like to note that despite there are multi-line cells in this corpus, they are less common than in Jeannette. Some examples of tables from both datasets can be found in Fig. 1.

The main figures for both corpora are shown in Tab. 1. The last row, *Relevant Information* (*Rel. Information*), accounts for the number of triplets $(\hat{y}_c, r, v)$ that we aim to extract (as will be explained in Sec. 5).

4. Proposed approaches

In this section, we discuss two different approaches developed for extracting textual information from hybrid printed-handwritten table images. We assume that tables are organized into orthogonal rows and *columns*. Each column has a *column header*, and each row has a *row header*. *Cells of interest* are the (generally handwritten)

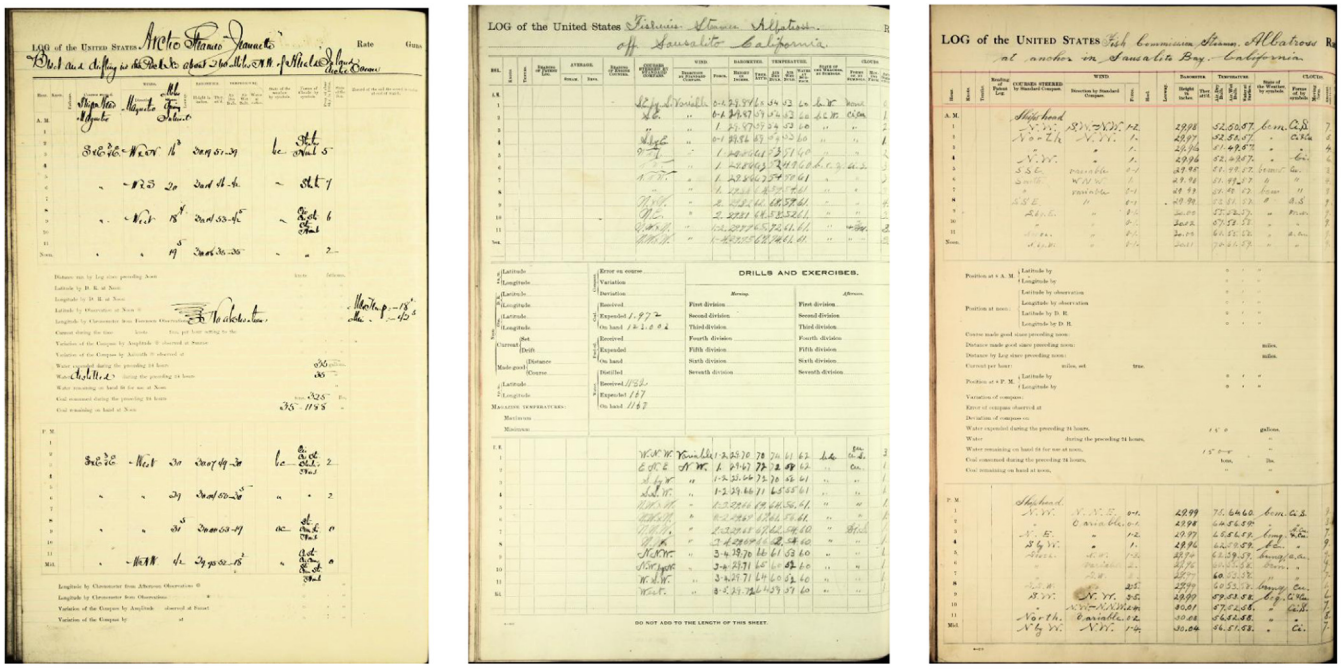

Fig. 1. Three table examples, Jeannette on the left and Albatross on the center and right. The number of columns is different in every image, being 17 in the first one, 19 on the second one and 18 on the third one. Moreover, note that while in the pages of Albatross all the rows are filled, in Jeannette it is only one row out of three. Additionally, it can be observed that multi-line cells are more common and tend to be larger in Jeannette, due to the fact that most rows of the tables are empty.

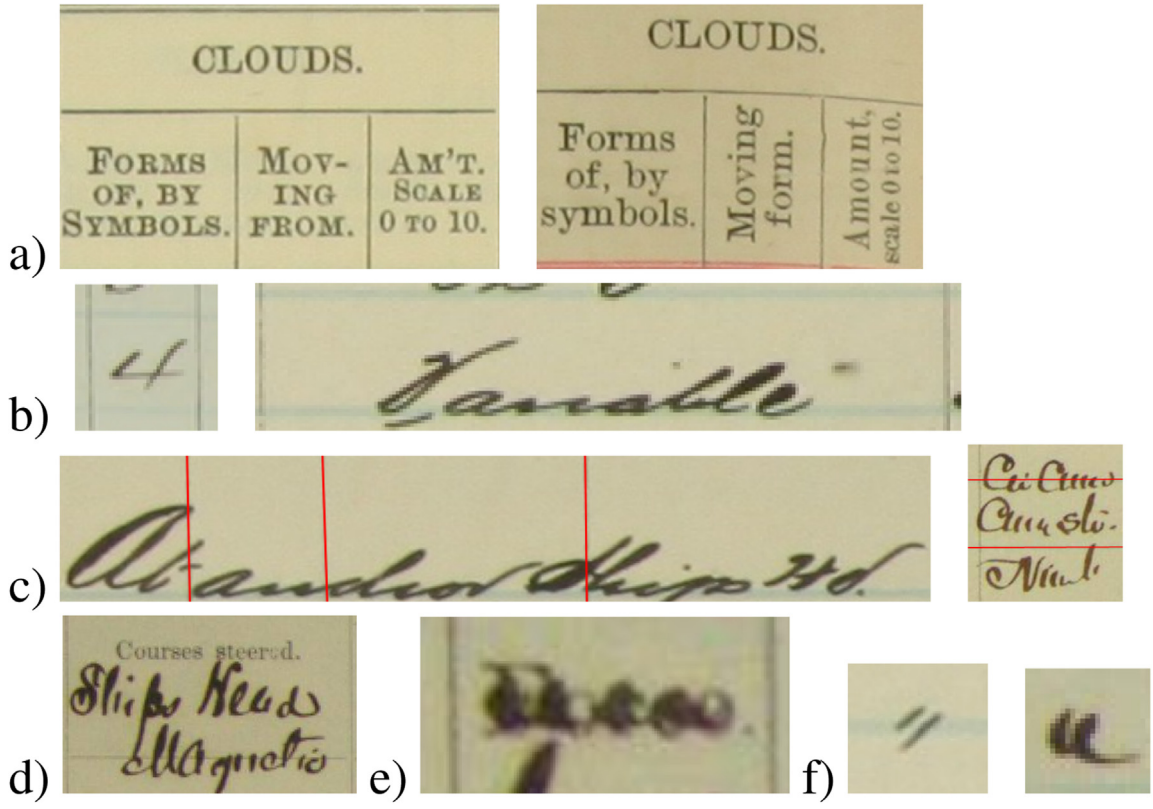

Fig. 2. Examples of challenges in Jeannette and Albatross datasets: (a) multi-span cell headers, differently written attributes depending on the table layout, and vertical text (right); (b) different cell widths; (c) cell contents exceeding boundaries (denoted in red); (d) column headers with handwritten contents; (e) crossed-out column headers; (f) quotation marks. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

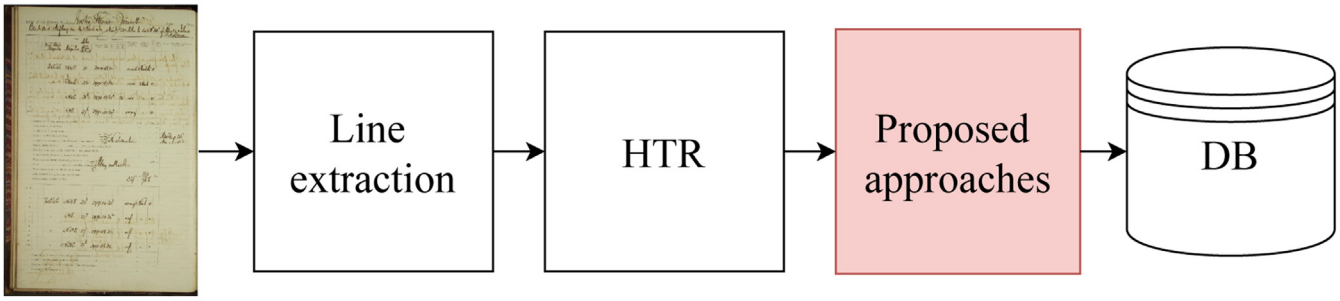

Fig. 3. Methodological pipeline. The block highlighted in red corresponds to the information extraction techniques described in this section. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

cells which loosely lay at the intersection of columns and rows and contain the information to be extracted.

The bounding boxes (BB) associated to these cells will be referred to as b_c, b_r and b_v, and the text in these cells as c, r and v, respectively,

It is important to note that the same columns can be arranged differently depending on the table layout. Moreover, column headers are not always written in the same ways; that is, different values of c may correspond to the same attribute to be extracted. So we will use y_c to denote a *normalized* column-header ID or attribute. The set of possible values of y_c has been semi-automatically determined from the ground-truth transcripts of the training table images.

The following approaches[2] take as input the extracted textlines and the corresponding automatic transcripts provided by an HTR system. As output, they return the textual contents of interest v associated with each column and row header tuple. Fig. 3 shows the phases going from the images to a database filled with the extracted information. Finally, we would like to remark that they do not explicitly make assumptions of the challenges shown in Fig. 2, nor do use templates.

4.1. Cremaet

First we describe the approach that we refer to as *Cremaet*[3]

4.1.1. A Previous Log-Linear model

Cremaet stems from the log-linear model proposed in [4]. Column headers, row headers and cells of interest were detected using four conditional probability distributions: $P(H_c \mid b)$, $P(H_r \mid b)$, $P(A_c \mid b_c, b_v)$, $P(A_r \mid b_r, b_v)$, implemented using Multilayer Perceptrons (MLP). The four random variables are binary. H_c or H_r are 1 *iff* b is a column or row header BB, respectively, and A_c or A_r are 1 *iff* the BB of a cell of interest, b_v, is vertically or horizontally aligned with b_c or b_r, respectively.

Cell BBs were built by possibly grouping some text lines and, if a cell contained a quotation mark, it was recursively replaced by the most likely textual information it referred to.

Finally, the relevant information was actually extracted from all sufficiently likely BB triplets (b_c, b_r, b_v). To compute a likelihood score, the probabilities of the four predictors were log-linearly combined, using four weights optimized on training.

4.1.2. A New Formulation

In *Cremaet*, we unify the four predictors of our previous approach into a single probability distribution, $P(R \mid b_c, b_r, b_v)$, where R is now a binary random variable whose value is 1 *iff* b_v is aligned vertically with b_c and horizontally with b_r. This distribution is directly estimated using a MLP trained from the GT annotations of the training tables and thereby the log-linear combination is not needed any longer. Moreover, while in our previous work a specific, hand-crafted representation was used for each type of BB, now a common, simple geometrical representation is adopted; namely, the BB center, width and height.

In addition, two new components are added to our previous pipeline: *cell classification* and *column header normalization*.

Therefore, as shown in Fig. 4, the new approach entails five phases: First, textlines are grouped into cells. This phase is needed to deal with multi-line cells. Next, cells are classified into column headers, row headers, cells of interest and out-of-the-table cells. Third, quotation marks in the cells of interest are replaced by the contents they represent. Fourth, in parallel with the previous phase, column header cells are normalized. Finally, information is extracted.

4.1.3. Group textlines

This process encompasses two steps. First, for any two textlines $b_v, b_{v'}$, a MLP is used to estimate the conditional probability, $P(C \mid b_v, b_{v'})$, where C is a binary random variable that denotes if b_v and $b_{v'}$ belong to the same cell or not. Then, b_v and $b_{v'}$ are grouped *iff* $P(C{=}1 \mid b_v, b_{v'}) \geq P(C{=}0 \mid b_v, b_{v'})$. Note that this procedure may result in textlines appearing in multiple cells simultaneously. Therefore, in a second step we simply merge any two cells that share at least one textline. An example is shown in Fig. 5.

4.1.4. Classify cells

Here each cell is classified as a column header, a row header, a cell of interest, or an out-of-the-table cell. The aim is to reduce the computational cost and hopefully improve the accuracy in other steps of the pipeline. To this end, again, a MLP is used to estimate the four-way class-posterior probability of a cell, represented by the geometric features of its BB. Then, each cell is assigned to the max-posterior class.

4.1.5. Replace quotation marks

Quotation marks need to be expanded to the (unique) textual contents they represent. Let b_q be a BB whose textual content is a quotation mark. In our probabilistic formulation, this content is replaced by the textual content of another cell b_v which is most likely referred to by b_q. Therefore, we define the conditional probability $P(Q \mid b_q, b_v)$, where Q is a binary random variable that denotes if b_v is or is not the cell to which b_q is referencing. Again, this probability is estimated by yet another MLP, using as input the geometric features of b_q and b_v.

4.1.6. Normalize column headers

Two normalization processes are needed to handle the column header challenges discussed in Sec. 3.

[2] These approaches only require light computing resources. An NVIDIA RTX 2060 GPU with 6GB of memory took less than two hours to train the required models.

[3] Name chosen just because we like this word – not an acronym of anything.

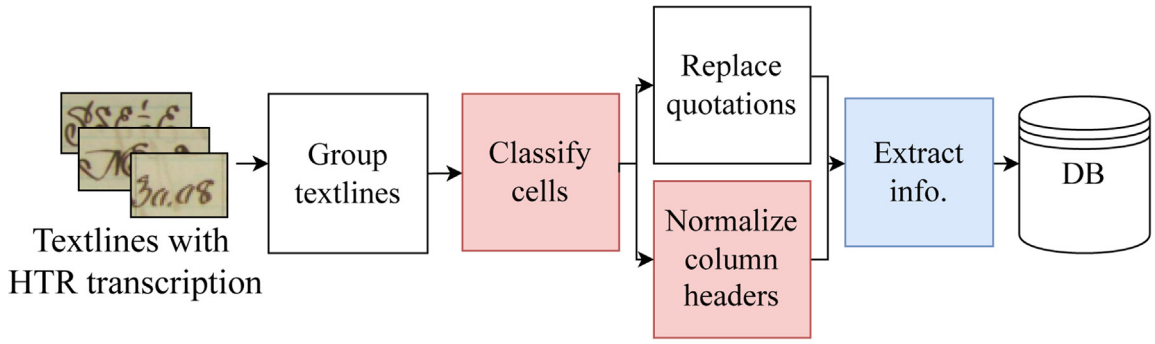

Fig. 4. General diagram of the *Cremaet* approach. The new or newly reformulated components with respect of our previous approach are highlighted in red or blue color, respectively. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

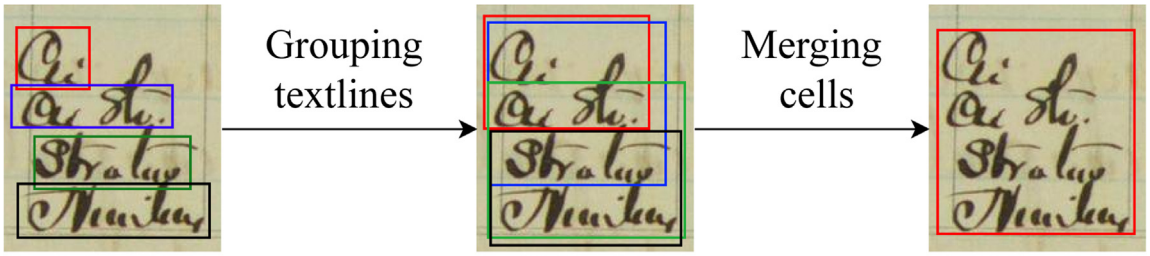

Fig. 5. Example showing how four textlines (leftmost image) are grouped into a single cell in two steps. In the first step, the textlines are grouped into tentative cells (middle), but it failed to group the most distant textlines. In the second step, the tentative cells that share at least one textline are merged, leading to the cell shown in the right image.

First, to deal with *multi-span headers*, the textual contents of two column header cells b_c and $b_{c'}$ are joined if it is likely enough that b_c is a multi-span cell on top of $b_{c'}$. Formally speaking, the textual contents of b_c and $b_{c'}$ are merged if $P(M=1 \mid b_c, b_{c'}) \geq P(M=0 \mid b_c, b_{c'})$, where M is a binary random variable that indicates if b_c is or is not a multi-span cell on top of $b_{c'}$. This probability is again estimated through another MLP, using as input the geometric features of b_c and $b_{c'}$.

Then, for *header text normalization*, the textual contents c of column header cell b_c are classified into a max-posterior column-header class according to:

$$\hat{y}_c = arg\,max_{y_c} P(c \mid Y = y_c)\ P(Y = y_c) \tag{1}$$

where c is the textual contents of b_c and the random variable Y takes values in the set of attributes we want to extract. To estimate $P(c \mid Y = y_c)$, a character n-gram language model is trained for each attribute y_c and the prior $P(Y = y_c)$ is straightforwardly estimated by maximum likelihood.

4.1.7. Information extraction

Finally, all the BB triplets composed of a column header cell b_c, a row header cell b_r and a cell of interest b_v are considered. However, information is only extracted from those for which b_v is likely enough to be column-wise and row-wise aligned with b_c and b_r, respectively. This likelihood is formalized by the probability $P(R \mid b_c, b_r, b_v)$ introduced in Sec. 4.1.2, above, and a triplet is selected if:

$$P(R=1 \mid b_c, b_r, b_v) \geq P(R=0 \mid b_c, b_r, b_v) \tag{2}$$

The textual information finally extracted from this triplet is ($\hat{y}_c$, r, v), where $\hat{y}_c$ is the column header class of b_c, r is the textual contents of the row header cell b_r and v is the textual contents of the cell of interest b_v.

4.2. Graph neural network

First, we describe the approach we refer to as GNN, as outlined in [4]. Subsequently, we explain how we extend the model to deal with multi-span and classify the headers by textual content.

4.2.1. Our previous approach

Techniques based on graph neural network (GNN) have been used to extract sub-structures of each table (rows and columns) from previously detected textlines. In this paper, the GNN-based techniques of [10] have been slightly modified following [4]. Instead of using the conjugate graph, the normal graph is used directly to classify edges using Eq. 3. This results in a more memory-efficient method, with no pre-processing (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 research, the number of nodes is kept the same, growing the number of edges but avoiding conjugation.

In a first step an *initial graph* is created which connects textlines by their "line of sight" (a fairly trivial process already used in other works [9,10]), and applying the improvements proposed in [10]. This yields a graph, G, depending on the sub-structure considered. Then, for each edge (s, d) of G the probability that it belongs to the actual graph which defines the sub-structure aimed at is estimated using a GNN, trained on pairs of initial graphs and corresponding correct GT graphs. This probability can be written as $P(Z=z_{s,d} \mid G, s, d)$, where Z is a binary random variable that takes the value $z_{s,d} = 1$ if the edge (s, d) should be in G and 0 if not.

$P(Z = z_{s,d} \mid G, s, d)$ is estimated using a GNN which includes an MLP as its output stack. The MLP has a sigmoid output for each edge s, d of G. The inputs are $\left|\boldsymbol{x}'_s - \boldsymbol{x}'_d\right|$ and $\boldsymbol{e}_{s,d}$, where $\boldsymbol{x}'_s$ and $\boldsymbol{x}'_d$ are the embeddings calculated by the GNN for the origin and destination nodes, respectively, and $\boldsymbol{e}_{s,d}$ are additional (hand-crafted) edge properties which are calculated at the time of creating G and remaining unchanged. The weights of the MLP layers are trained along with all the other layers of the GNN stack as a whole net-

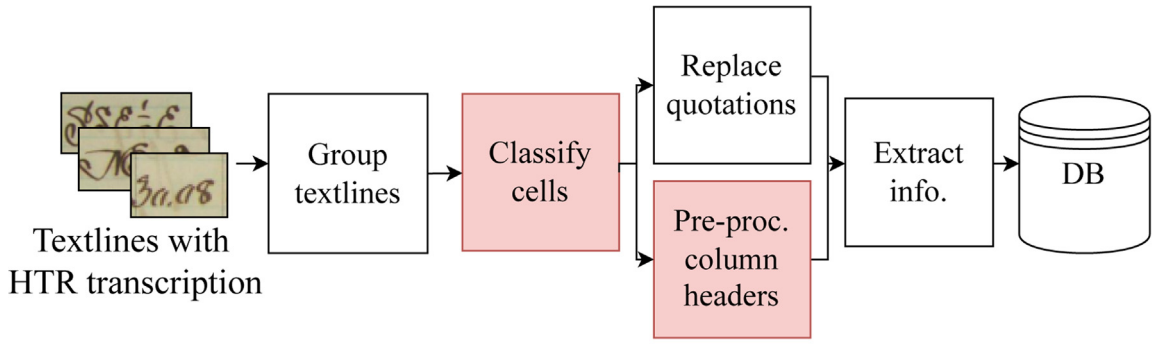

Fig. 6. Example of a directed graph solving the problem of a multi-span column header. In this case, following the directed paths, three headers would be extracted: "Clouds Forms of by symbol", "Clouds moving form", and "Clouds amount scale 0 to 10".

work, using the binary cross-entropy loss applied to all the MLP outputs.

Once the GNN has been trained, it can be used to prune from G those edges (s, d) such that

$$P(Z = 1 \mid G, s, d) \geq t_G \tag{3}$$

where t_G is a threshold to be empirically tuned. Finally the connected components of the pruned graph are extracted and each component is assumed to correspond to one element (row or column) of the sub-structure considered.

A GNN is also used to detect the textlines headers. 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 (for row, column and header detection), for each existing cell, the row and the column it belongs to are detected, 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.

4.2.2. Dealing with multi-span

To handle the multi-span column headers, a directed graph has been created with the previously classified headers. The direction of the edges within the cell corresponds to the reading order, and when two textlines from different cells are connected, it corresponds to the multi-span, as shown in Fig. 6. Unlike the previous graphs, this one is much smaller as it only has the nodes classified as headers. The only difference between this model and the other GNN-based models is that the absolute value is not applied to $\boldsymbol{x}'_s - \boldsymbol{x}'_d$. This allows having a different value for the tuples $(\boldsymbol{x}'_s, \boldsymbol{x}'_d)$ and $(\boldsymbol{x}'_d, \boldsymbol{x}'_s)$.

After classifying the edges with the GNN explained above and starting from each end-node of the graph, a path to a parent node of each column multi-span is searched for. Each of these paths will be a column header. Fig. 6 shows an example of what the output of the directed graph of the GNN looks like after classifying and pruning the edges. In this case, three column headers would be extracted with the word "Clouds" in common.

Finally, we classify the textual contents of each column header following the same method explained in Sec 4.1.6.

5. Evaluation criteria and metrics

The HTR transcription quality is assessed using the Character and Word Error Rates (CER and WER), while line extraction is evaluated utilizing the F_1^b as defined in [13].

The performance of the information extraction approaches has been assessed employing the F_1 score. To evaluate it, the extracted triplets by our systems have been compared to a list of extracted triplets generated from the GT transcripts. Therefore, an extracted triplet $(\hat{y}_c, r, v)$ is considered a *true positive* (TP) when it matches one triplet of the list of GT triplets that are found in the same page. Otherwise, it is considered a *false positive* (FP). Moreover, when a triplet found in the GT list of triplets is not extracted, this is considered as a *false negative* (FN). Please note that, as we are evaluating only employing the textual contents, we do not need to check if the BBs of the extracted triplets match with the ones of the list of GT triplets.

Finally, we would like to remark that 95% confidence intervals ($\alpha = 0.025$) have been calculated using the bootstrap method with 10 000 repetitions [14].

6. Experimental framework and results

Different experiments have been performed to evaluate the text recognition and the proposed information extraction (IE) approaches. In the following sections, we will describe the experimental framework that has been followed and the achieved results. In order to make the experiments reproducible, the code and the corpora used are available online.[4] [5]

6.1. Experimental settings

Both for line detection and HTR, the systems have been trained putting together all the training images of Jeannette and Albatross. However, to assess IE, the models have been trained separately with data of each corpus and, finally, again with both corpora together without distinction.

6.1.1. Line detection

Lines were automatically detected using MaskRCNN, as implemented in the Detectron2 tool [15]. The same model has been used for all the images, with ResNet50 as backbone.

To help the HTR processing, lines have been detected in two classes: vertical and horizontal. This way, lines can be rotated before they are processed by the HTR system.

6.1.2. Handwriting text recognition (HTR)

The PyLaia [16] toolkit was used for text recognition as described in [17]. Extracted lines were pre-processed to reduce basic geometry variabilities, such as skew and slant.

Optical models are based on Convolutional and Recurrent Neural Networks. This model consists of five convolutional layers with

[4] https://github.com/JoseRPrietoF/tableIE
[5] https://doi.org/10.5281/zenodo.6937607

filters composed of different maps of characteristics (16, 32, 48, 64 and 64) with kernel sizes of 3×3 pixels. As an activation function, *LeakyReLU* has been used without any reduction in image size overall the process. After that, the recurrent block is made up of three recurrent layers composed of 128 bidirectional long-short term memory units. All the hyper-parameters, such as the number of convolutional and recurrent layers, were configured in the validation set. It is important to remark that the same optical models were used both for the printed and handwritten text.

A 10-gram character language model was estimated directly from the transcripts of the training and validation text lines using the SRILM [18] toolkit. A single language model was used for both the handwritten a printed text.

As input for the two IE approaches, the 1-best HTR transcripts obtained using the trained optical and language models, have been employed.

6.1.3. Cremaet

The MLPs adopted in this approach encompass two hidden layers of 512 units plus the final softmax classification layer. The standard cross-entropy loss function was adopted in all the cases, including all the binary classifiers, for which the two underlying classes were explicitly assumed as such. The Adam solver [19] was used to train each MLP for 250 training epochs, with a learning rate of 0.01.

The input features for the different MLPs are the center, width and height of the cells or textlines considered in each case. As training samples, all the possible combinations of cells or textlines in each phase have been considered. In the grouping textlines phase, only the pairs where two textlines are consecutive and belong to the same cell are considered positive samples. Otherwise, they are considered negative samples.

To model $P(c \mid Y = y_c)$, a character bigram language model has been estimated for each class of column header y_c, using the NLTK [20] toolkit.

6.1.4. Graph neural network

The same configuration has been used for the four cases (rows, columns, headers and multi-span), with four layers of 64 filters each in the first steps of the GNN. Finally, an MLP with four hidden layers of 64 neurons each, plus a binary classification layer has been employed.

Only geometric characteristics have been calculated for nodes and edges, up to 12 for each node and 9 for edges. These are, for example, the size of the node, the position in the image or the length of the edge, among others. All of them can be found in the code.

The Adam solver [19] was used to optimize the GNNs, with a minibatch SGD and a learning rate of 0.01. The networks were trained for 4000 epochs. To reduce false positives, which have a very detrimental effect, a weighted cross-entropy has been used as a loss function. 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, α has been set to 5 in all experiments. To consider an edge as positive, the threshold t_G of Eq. 3 was just set to 0.5 in all the cases, without any further tuning.

To improve the initial graph as proposed in [10], σ_1 was set to 1 and σ_2 to 10 for columns and the other way round for rows. To detect headers we used the original graph. In the multi-span case, the original graph has been used but filtered only by textlines classified as headers and with directed edges.

Finally, $P(c \mid Y = y_c)$ as trained and used in Sec. 6.1.3.

6.2. Textline detection and text recognition results

Although Albatross has more lines per page than Jeannette, line extraction performance remained stable on both corpora, achieving an F_1^b of 0.93.

With respect to text recognition, Table 2 reports the results obtained for the test pages of each corpus. These results correspond to the detected lines given in the GT, which means that all annotated lines were used to compute these values.

The overall error rates (O, handwritten + printed text) are very low in general; for both corpora together (J+A), we achieve CER=2.60% and WER=5.39%). However, as pointed out in previous sections, it is relevant to distinguish between errors in the printed (P) and handwritten (M) text. As expected, the recognition of the handwritten text represents a greater challenge than the printed text, resulting in much higher CER and WER for handwritten text in all the cases.

Regarding the results by individual corpus, results for Jeannette are always better than for Albatross; significantly so for handwritten text. This is due to two main reasons: first, the writing in Jeannette is more uniform, whereas in Albatross there are different writing styles. Second, Albatross tables tend to have more cells filled with content, which means that the text is more dense and more challenging to detect and/or recognize.

6.3. Information extraction results

Table 3 reports the IE performance achieved in three scenarios: a) both lines are perfectly detected and transcripts (contents) are perfect (GT); b) lines are perfect but transcripts are HTR hypotheses and c) both lines and transcripts are automatically obtained hypotheses. Note that the latter is the real-world scenario we would encounter in practice. Obviously, it is also the most pessimistic case because all the possible errors of the whole pipeline are accumulated.

In each case, results are shown for Jeannette, Albatross, and both corpora jointly (J+A). On the other hand, both IE techniques, *Cremaet* and GNN, have been trained for each of individual corpus separately or for both together in the J+A case. The joined J+A dataset would correspond to a third corpus with larger layout and writing style variabilities. This was aimed to challenge the generalizing capabilities of the systems.

To assess how far is each of our approaches from perfect IE behavior, we report the maximum performance that could be achieved if our systems did not make errors in obtaining the structure of columns, rows, and headers. These results are denoted as an “Oracle” in Table 3. In the “Oracle” row we can see that there would be no error when we have GT lines and content. Then, we observe the maximum performance that could be achieved when employing the HTR transcription hypotheses. It is worth mentioning that it does not perfectly correlate with the WER reported in Table 2. This is mainly due to the quotation marks: when they are correctly transcribed, they will not increase the WER. However, they could still harm the IE performance if the contents that they refer to are incorrectly transcribed. Finally, we can see the best results reachable when using HTR and the line detection together, where we would get at most 0.86, 0.66, and 0.73 F_1 points in Jeannette, Albatross, and both corpora (J+A) respectively.

On the one hand, the results of Table 3 show that when it comes to only one corpus, that is, only Jeannette or Albatross, *Cremaet* generally obtains better or similar results than the GNN model. On the other hand, when we employ both corpora together, the GNN model achieves superior performance in all cases. In the most realistic scenario the GNN obtains 0.70 F_1 points compared to 0.65 points from *Cremaet*.

Table 2
Results of text recognition for Jeannette (J) and Albatross (A). M, P and O refers to manuscript, printed and overall text respectively. 95% confidence intervals are never larger than 0.9% for manuscript text, 0.4% for printed text and 0.3% overall. All numbers are percentages.

Corpus	Jeannette			Albatross			J+A		
Test type	M	P	O	M	P	O	M	P	O
CER	4.14	1.21	1.72	14.19	1.48	5.56	6.92	1.27	2.60
WER	6.82	1.60	3.45	18.83	1.92	10.48	11.20	1.67	5.39

Table 3
Information extraction F_1 results. 95% confidence intervals are never larger than 0.01. J+A accounts for Jeannette plus Albatross.

Corpus	Jeannette			Albatross			J+A		
Lines	GT	GT	Hyp	GT	GT	Hyp	GT	GT	Hyp
Content	GT	Hyp	Hyp	GT	Hyp	Hyp	GT	Hyp	Hyp
Cremaet	0.98	0.90	0.85	0.93	0.70	0.64	0.88	0.72	0.65
GNN	0.95	0.88	0.81	0.90	0.68	0.65	0.93	0.76	0.70
Oracle	1.00	0.92	0.86	1.00	0.75	0.66	1.00	0.80	0.73

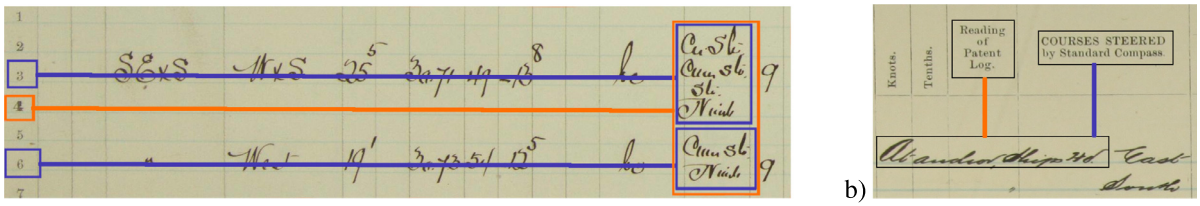

Fig. 7. Example of system failures. The orange edges correspond to the Cremaet output, while the blue ones correspond to the GNN output. For simplicity, only the bounding boxes of the affected textlines are shown. In a), we see how Cremaet has failed to group textlines and has joined two cells together. Moreover, it attributes this cell to row 4, whereas there should be two cells in rows 3 and 6, respectively. In blue, we can observe how GNN has got it right. In b) we can see how the GNN has joined a line that does not respect the layout with the header that does not belong to. This GNN failure probably affects the whole column. In orange, we see how Cremaet has got it right. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Furthermore, Table 3 also shows how the entire pipeline distorts the results at each successive phase. While in Jeannette the difference is less noticeable (only about 0.07 points are lost to HTR content errors and another 0.07 more when employing not automatic lines and HTR), in Albatross it becomes more apparent. In this corpus, which internally exhibits significant differences in types of layout and writing styles, 0.25 points are lost when using the HTR system and 0.09 more points are lost in "Oracle" when employing automatically extracted lines.

7. Discussion

On the one hand, we have the *Cremaet* model, which is more straightforward than the GNN-based model, as it does not require the creation of an initial graph. Such a graph can limit the results due to possible misclassification of edges. As demonstrated in [10], just a single false positive may lead to join two rows or two columns, and all of the triplets involved in those rows or columns become wrong. This significantly harms the results. Moreover, the viewpoint of *Cremaet* differs from the one of the GNN-based model. While the GNN-based model tries to find substructures (rows, columns and multi-span) by looking at its neighboring textlines, *Cremaet* relies on the column and row headers to unravel the structure of the table.

On the other hand, the GNNs-based model is more general as it can be applied, with practically no model changes, to tables or any other type of structures (e.g., paragraphs, acts, etc). Although the GNN gives a probability per edge in the case of substructures, this probability has been truncated in a binary way to extract connected components (i.e., rows and columns). If the probability of each edge is used correctly in the future, the results will likely be improved.

Comparing both systems with the results obtained, we see that *Cremaet* performs better when the tables are more uniform, while the model based on GNNs obtains better results than *Cremaet* when the corpora is more heterogeneous (different number of filled rows, table layouts, etc). Moreover, the fact that it relies on the neighboring textlines to determine the substructure makes this technique more robust to pages with severe skew. Fig. 7 shows one error of each method. The GNN error happened because the physical layout was not respected, while a bad cell joining caused several errors in the Cremaet result.

Finally, it should be noted that the line detection and HTR phases are crucial, given that if a line is not correctly detected and/or its contents are incorrectly transcribed, it might lead to a series of failures that the IE methods will not be able to fix it.

8. Conclusions

This paper reports research carried out on information extraction from historical handwritten tabular documents using two machine learning models. Both models achieve results close to the maximum achievable given the pipeline followed, i.e., automatic line extraction and HTR transcripts. *Cremaet* obtains better results than the GNN model when tables are more uniform, while the GNN-based model generalizes better than *Cremaet* when the corpora is more heterogeneous.

Taking this information into account, improving the line detection and HTR subsystems would be a direct way to mitigate the errors that the IE module currently encounters. Moreover, the use of

textual information could be crucial to improve substructure systems and will be studied in the future.

As a replacement, or complement, to the direct improvement of HTR results, *Probabilistic Indexing* (PrIx)[21] looks like a promising alternative image representation that could largely overcome the impact of HTR errors. Rather than single, error-prone 1-best HTR line transcripts, PrIx provides rich probabilistic distributions of words and their geometry that could be advantageously used to avoid taking irreversible decisions along the processing pipeline.

Yet as another, more definitive solution to this issue, it would be interesting to explore approaches to develop an end-to-end system that performs the line detection, HTR transcription and information extraction in a fully holistic way.

Finally, we would like to remark that, beyond the application to ship logs of the information extraction pipeline proposed and assessed in this paper, it is straightforward to apply the very same approach to other types of tables or form-structured documents, such as records, questionnaires, etc.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgement

Work partially supported by : the Universitat Politécnica de Valéncia under grant FPI-I/SP20190010 (Spain), the SimancasSearch project as Grant PID2020-116813RB-I00a funded by MCIN/AEI/ 10.13039/501100011033, grant DIN2021-011820 funded by MCIN/AEI/ 10.13039/501100011033, the valgrAI - Valencian Graduate School and Research Network of Artificial Intelligence and the Generalitat Valenciana, and co-funded by the European Union.

References

[1] E. Lang, J. Puigcerver, A.H. Toselli, E. Vidal, Probabilistic indexing and search for information extraction on handwritten german parish records, in: ICFHR, 2018, pp. 44–49.
[2] L. Gao, Y. Huang, H. Dejean, J.L. Meunier, Q. Yan, Y. Fang, F. Kleber, E. Lang, ICDAR 2019 Competition on table detection and recognition (ctdar), Proc. Int. Conf. Document Anal. Recognit., ICDAR (2019) 1510–1515.
[3] V. Romero, J.A. Sánchez, The HisClima database: historical weather logs for automatic transcription and information extraction, in: 2020 25th International Conference on Pattern Recognition (ICPR), 2021, pp. 10141–10148.
[4] J. Andrés, J.R. Prieto, E. Granell, V. Romero, J.A. Sánchez, E. Vidal, Information extraction from handwritten tables in historical documents, in: S. Uchida, E. Barney, V. Eglin (Eds.), Document Analysis Systems, Springer International Publishing, Cham, 2022, pp. 184–198.
[5] E. Granell, V. Romero, J.R. Prieto, J. Andrés, L. Quirós, J.A. Sánchez, E. Vidal, Processing a large collection of historical tabular images, Pattern Recognit. Lett. 170 (2023) 9–16.
[6] A. Gilani, S.R. Qasim, I. Malik, F. Shafait, Table detection using deep learning, Proc. Int. Conf. Docum. Anal. Recognit., ICDAR 1 (2017) 771–776.
[7] S.A. Siddiqui, I.A. Fateh, S.T.R. Rizvi, A. Dengel, S. Ahmed, Deeptabstr: deep learning based table structure recognition, Proc. Int. Conf. Document Anal. Recognit., ICDAR (2019) 1403–1409.
[8] S.R. Qasim, H. Mahmood, F. Shafait, Rethinking table recognition using graph neural networks, Proc. Int. Conf. Document Anal. Recognit., ICDAR (2019) 142–147.
[9] A. Prasad, H. Dejean, J.L. Meunier, Versatile layout understanding via conjugate graph, Proc. Int. Conf. Document Anal. Recognit., ICDAR (2019) 287–294.
[10] J.R. Prieto, E. Vidal, Improved graph methods for table layout understanding, in: J. Lladós, D. Lopresti, S. Uchida (Eds.), Document Analysis and Recognition – ICDAR 2021, Springer International Publishing, 2021, pp. 507–522.
[11] E. Lang, J. Puigcerver, A.H. Toselli, E. Vidal, Probabilistic indexing and search for information extraction on handwritten german parish records, in: 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), 2018, pp. 44–49.
[12] T. Constum, N. Kempf, T. Paquet, P. Tranouez, C. Chatelain, S. Brée, F. Merveille, Recognition and information extraction in historical handwritten tables: toward understanding early 20th century paris census, in: Document Analysis Systems, Springer International Publishing, Cham, 2022, pp. 143–157.
[13] T. Grüning, R. Labahn, M. Diem, F. Kleber, S. Fiel, READ-BAD: A new dataset and evaluation scheme for baseline detection in archival documents, CoRR (2017).
[14] M. Bisani, H. Ney, Bootstrap estimates for confidence intervals in ASR performance evaluation, in: ICASSP, volume 1, 2004, pp. 409–412.
[15] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, R. Girshick, Detectron2, 2019.
[16] J. Puigcerver, C. Mocholí, Pylaia, 2018, (https://www.github.com/jpuigcerver/PyLaia).
[17] J. Puigcerver, Are multidimensional recurrent layers really necessary for handwritten text recognition?, volume 01, 2017, pp. 67–72.
[18] A. Stolcke, SRILM-an extensible language modeling toolkit, in: Proceedings of the 3rd Annual Conference of the International Speech Communication Association (Interspeech), 2002, pp. 901–904.
[19] D.P. Kingma, J.L. Ba, Adam: a method for stochastic optimization, in: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, 2015.
[20] S. Bird, E. Loper, E. Klein, Natural language processing with python o'reilly media inc (2009).
[21] E. Vidal, A.H. Toselli, J. Puigcerver, A probabilistic framework for lexicon-based keyword spotting in handwritten text images, arXiv preprint arXiv:2104.04556 (2021).