OCRticle - a Structure-Aware OCR Application

Sofia G. Rodrigues dos Santos
Informatics Department, University of Minho, Braga, Portugal

J. João Dias de Almeida
ALGORITMI/LASI, University of Minho, Braga, Portugal

Abstract

While there are currently many applications and websites capable of performing Optical Character Recognition (OCR), none of the widely available options offer structured OCR, i.e., OCR that maintains the text's original structure. For example, if a document has a title, after performing OCR on it, the title should have a different formatting, in order to distinguish it from the rest of the text.

This paper covers the topic of structure-aware OCR, first by describing the current state of OCR tools, then by showcasing a prototype tool capable of retaining the structure of articles scanned from an image.

2012 ACM Subject Classification Applied computing → Optical character recognition

Keywords and phrases OCR, Optical Character Recognition, Data Structure, Data Parsing, Document Structure

Digital Object Identifier 10.4230/OASIcs.SLATE.2023.8

Supplementary Material *Software (Source Code)*: `https://github.com/RisingFisan/OCRticle` archived at `swh:1:dir:651451c61ae5fca1265a703ed38eab264bb82551`

Funding This work has been supported by FCT Fundação para a Ciência e Tecnologia within the R&D Units Project Scope: UIDB/00319/2020.

1 Introduction

Currently, most Optical Character Recognition (OCR) tools are limited to a textual output. In other words, if someone is scanning an image that contains text with varying font sizes, indentation or colors, for example, inputting this image into an OCR tool will generate a string of pure text without any of that "extra" information. When scanning documents like newspaper or magazine pages, it becomes important to retain that information, otherwise it becomes much harder to identify features of the text like titles or captions.

The main goal of this paper is to provide a critical assessment of OCR tools, their strengths and limitations, and to showcase a prototype tool capable of performing OCR while keeping the original text's structure as intact as possible.

2 State of the Art

This section covers the current state of OCR tools and some of their main features. It also mentions and describes Tesseract, one of the most used OCR engines nowadays.

2.1 OCR Tools

Most available OCR tools can be described as "basic", and a simple online search for the term "online OCR" will reveal hundreds of websites that perform this type of OCR. Using these tools, a user can upload a file, usually an image, and the tool will process it and return a string of text that it found on the image. These tools offer very little customization or

extra features, as they are meant to be used by anyone, regardless of previous knowledge in character recognition software, with most of them only allowing users to select the language in which the text is written and if the output file should be a text file, a PDF file or a Microsoft Word file, for example [12][14].

Table 1 contains a list of some popular OCR tools [10] and the main features supported by each. This list only includes tools that can be used for free. Paid tools tend to offer most, if not all of these features, but we did not find any additional feature offered by a paid tool that was not also part of a free tool. By default, all of these tools support textual output.

Table 1 Comparison of some OCR tools.

Tool	Recognize text in Portuguese	Auto-rotation	Table recognition	Restrict OCR to part of file/image	Create searchable PDF	Create DOCX file
OCRSpace [11]	X	X	X		X	
FreeOCR [3]						X
OnlineOCR [12]	X	X	X		X	X
Google Lens [4]	X	X		X		
Text Grab [19]				X		

One of these OCR tools that stands out from the rest is Google Lens [4]. This tool, available on mobile devices, allows the user to use their smartphone's camera to capture an image containing text. After taking this picture, the user can interactively select and copy the text recognized in the picture. Google Lens also excels at recognizing handwritten text and other kinds of text that traditional tools struggle with [20], mainly due to Google's investment in Machine Learning algorithms.

Some tools also offer the possibility to create a special PDF file containing the original scanned document, but with an invisible text layer above the images [5]. This way, users can still see the original document but also select the text found in it, as if it were a regular PDF file. This method is a possible solution to structure-aware OCR, and is used by search engines in order to find text in PDF files composed only of images [6]. However, the text itself is not structured, it only appears to be, thus it is not the best solution for structured OCR.

It's worth noting that different tools have different use cases, and therefore might not have the need for all of the features shown in Table 1. For example, Google Lens uses a smartphone's camera to capture and detect text, unlike the other tools, and Text Grab allows the user to take a screenshot of their computer and immediately recognize text from the captured image, so it makes sense that these tools wouldn't focus on features like creating searchable PDF files.

2.2 OCR Engines

Although the aforementioned tools perform optical character recognition, they are merely wrappers for OCR engines. An OCR engine is the software responsible for recognizing the characters in an image and converting them to text [1]. Applications like Text Grab use these engines to bring OCR functionality to their tools, and are meant to be more user-friendly and intuitive than purely using an engine.

One of the most widely used OCR engines is called Tesseract. It can be used directly via command line, through a 3rd party tool or by using an API written for a programming language [15]. Examples of Tesseract APIs include Python-tesseract [13] or Tesseract.js [17], which can be used by programmers to create applications with OCR functionalities.

Tesseract supports dozens of languages [18] and offers a wide assortment of options [15]. One of these is related to page segmentation, and affects how Tesseract detects text in an image. By default, Tesseract tries to divide the original picture into segments, which can be titles or columns of text, for example. Then, it performs OCR on each of those segments. If the output format is a textual format, this will mean that the final text will be split into segments. However, each one of these segments has the same font and size, making it impossible to distinguish between a title and a regular sentence, for example, without additional context. This default behavior can be modified or disabled, if one desires.

Additional options include output formatting, which can be a PDF file, a text file or an hOCR file (HTML compatible file, with additional information about the original text's structure), for example, or an option to let Tesseract detect words from a user-provided list.

Since Tesseract is mainly an OCR engine, its image processing capabilities [16] may not be adequate for all use cases, which leads many users to apply pre-processing effects to their images before inputting them in Tesseract, in order to improve the text detection [9]. These may include changes in contrast, brightness, size or even converting the image to black-and-white, to avoid issues with colored text/backgrounds.

3 Problem Definition

This paper proposes the creation of a program capable of performing Optical Character Recognition on a file, while maintaining its original structure. For this purpose, a prototype called **OCRticle** was developed, in order to showcase the benefits of this approach to OCR. Unlike already existing solutions that convert a file to a PDF file with invisible text, this tool creates a purely textual file. In order to have a structured textual file, this application uses Markdown [2], a markup language [7], in the generated files.

3.1 Program Specifications

OCRticle is a desktop application developed in the Python programming language. This decision stems from the authors' experience with Python and the ability to easily and rapidly create a command-line or graphical application with Python. Tesseract is used for the OCR component of the application, along with `pytesseract`, Python's Tesseract API [13].

The application focuses primarily on detecting text from newspapers or magazine pages, essentially pages with one or more articles, as its name implies. In practice, it can scan any type of document, but it might not be able to preserve its structure entirely.

OCRticle has a Graphical User Interface (GUI), developed using the Kivy framework [8], where a user can select a source image containing the article(s) they want to convert. Then, they can select each article's position within the image. This allows the tool to focus less on discerning between different articles and more on formatting each one independently. After this step, the tool performs OCR on each image section and display the results to the user, who can then save the detected text in a Markdown file or perform some minor adjustments to the final document's structure.

3.2 System Architecture

The system follows an architecture similar to the one shown in Figure 1.

OCRticle is composed of a front-end and a back-end. The front-end is responsible for user interaction and uses the Kivy module to render a GUI. After receiving user input, the front-end sends data to the back-end, the "brain" of the program. The back-end processes

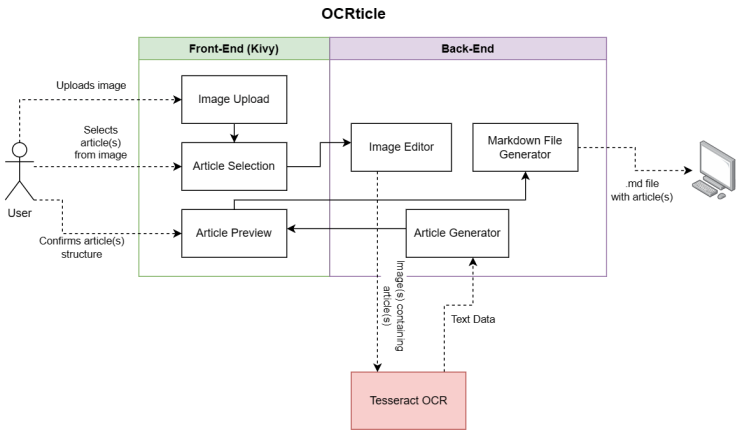

Figure 1 System architecture diagram. Interactions within OCRticle are represented with solid lines and interactions with the "outside" are represented with dotted lines.

data and generates results, in this case, a formatted article or group of articles. It also communicates with Tesseract, sending it images and receiving data about the text present in them.

4 Development

This section covers the development steps of OCRticle.

The first step in development consisted of creating a simple program capable of converting an image into an intermediate representation (IR). This IR holds information about the text and additional context about the OCR process, for example, the dimension of the bounding box where the text was located. The pytesseract module already has a method `image_to_data` which can convert an image into a data structure (e.g., TSV file, dictionary or `pandas` dataframe). However, this data structure contains more data than necessary, so it became necessary to simplify it and make it easier to navigate and access. Hence, this information is filtered and converted into instances of Python classes.

4.1 Classes

Four classes were created for this purpose: Article, Block, Paragraph and Line. The last three classes simply mirror the information returned by pytesseract, which splits text into pages, blocks, paragraphs, lines and words. Since this tool will only work with individual pages, the first category is not needed, and the fifth category, words, is represented as a list in the Line class, since there is no need to store extra information about each word. Besides a list of words, the Line class also contains information about its height. This information is useful because Tesseract might sometimes classify two lines as being part of different paragraphs, or two paragraphs as being part of different blocks, which is not always the case. If we assume that lines of the same paragraph and paragraphs of the same block should have text of the same size, by storing each line's height, we can compare the height of different lines and paragraphs, and infer if two lines/paragraphs should be part of the same paragraph/block. For this purpose, the Paraphaph class contains a `get_line_height` method, which returns the average height of all its lines. Similarly, the Block class contains an equal method, but one which returns the average height of its paragraphs' lines.

Blocks have an additional attribute, `type`, which can be one of four different values: TITLE, TEXT, QUOTE, or CODE. These types should match the role of the text inside the block in the original article. When saving the final file generated by OCRticle, each block type has a different representation, according to the Markdown syntax [2].

Figure 2 illustrates the internal representation of an article.

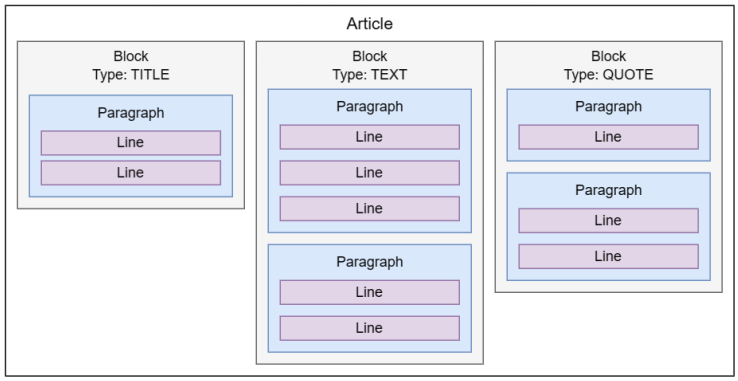

Figure 2 Article internal structure - example.

This particular article is composed of three blocks, one of which is the article's title, another a block of text, and the third one a quote.

4.2 Graphical User Interface

After these classes were developed, the main step in creating OCRticle arose, building the GUI.

4.2.1 Image selection and preprocessing

When opening the application, the interface contains a window where the user can select an image to be analyzed. This step can be skipped if OCRticle is opened through the command line and given a file path as an argument.

After selecting an image, the next window of the application allows the user to select the articles present in the image by drawing rectangles over them. Optionally, users can select areas for the program to ignore. For example, if the original articles contain images with text, one could select the image as an area to be excluded, so that Tesseract won't detect the text in the image. Additionally, if an image has many articles and the user only wants to run OCRticle on some of them, they can just select the articles that they want from the image, instead of having to perform OCR on the entire document or manually cropping the image.

There are also options to control the image's brightness, contrast, and saturation. This is particularly useful if the original image is not well-lit or has a lot of colors in it.

4.2.2 Text detection and formatting

After the user selects the articles in the image, OCRticle divides the image into segments, based on the drawn rectangles, and feeds those segments to Tesseract, which proceeds to detect the text within those images. Then, the tool creates instances of the Article class, each containing the text from a different article. When creating these instances, the program perform an optimization step, where it tries to group together different blocks or paragraphs,

based on how they end. For example, if a paragraph ends with a hyphen and the next one begins with a lowercase letter and both paragraphs have the same line height, it probably means that Tesseract failed to detect both lines as part of the same paragraph, and the program will try to automatically fix that mistake.

In addition, OCRticle also tries to detect an article's title. To do this, it analyses every block from the scanned text and finds the one with the biggest font size, which it then tags as the article's title. In the final file, this is represented with a pound sign (#) before the block, which is used in Markdown to define a heading. If every block has a similar font size, OCRticle doesn't do anything, to avoid mislabeling a block, although this can still happen if the block with the biggest font size is not the article's title.

In case of mislabeling, OCRticle allows the user to manually label each detected block before saving the final file. These labels correspond to the block types mentioned previously. OCRticle also joins blocks that it believes are similar enough to be of the same type. For example, if an article has two columns, each with some paragraphs, and Tesseract returns one block for each column, OCRticle will try to figure out if those blocks have the same structure and should therefore be merged. If OCRticle doesn't merge two blocks, for some reason, the user can perform that merge automatically, though a button in the article preview screen.

Another feature offered by OCRticle, which is not present in most OCR tools, is the possibility to include or exclude line breaks inside each paragraph. Typical OCR tools, when scanning a document, will preserve each line in the original image as a separate line in the final text. However, in the original document, the text is only split into multiple lines due to the limited size of the paper. On a computer screen, which is typically wider than a piece of paper, it might not make sense to preserve the original line breaks. Therefore, after scanning a picture, there's an option in the application to remove the original line breaks, and keep each paragraph as a single continuous line. This behavior mimics computer text editors, like Microsoft Word, where the line breaks are artificially created by the software in order to make the text fit in the screen, while the text remains stored as a single line. Figure 3 illustrates this difference, with the newline characters highlighted in red.

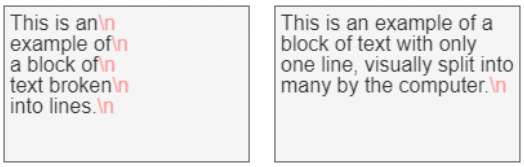

Figure 3 Distinct options for line breaks in output file. The first option mimics the original image.

Line breaks will always be removed in titles because headings in Markdown must consist of a single line.

4.2.3 File saving

After a user has confirmed the labels for each articles' blocks, OCRticle allows them to save the formatted text as a Markdown file. This file can then be opened in a Markdown viewer and it will be displayed according to the preferences set by the user before saving.

5 Usage Example

This section illustrates a usage scenario for OCRticle.

In this scenario, a user called Sam wants to convert an image with an article to text while keeping the article's structure, so Sam uses OCRticle for this purpose. Sam's original image can be seen in Figure 4.

Figure 4 Image that Sam wants to convert to structured text.

Sam starts by opening OCRticle and selecting the image from their computer. This particular image is stored as a PNG file, but Tesseract accepts any kind of image format.

Then, OCRticle asks Sam to select the article or articles from the image. Since this image only has one article, Sam can simply press "Submit" and OCRticle assumes that the entire image contains just one article. However, since the image also contains an advertisement, Sam uses the "Exclude from article" drawing mode to draw a red rectangle over the banner, thus excluding it from being scanned by Tesseract.

In addition, since the image has a purely white background and black text, Sam has no need to use the brightness, contrast or saturation sliders. However, these options would be useful if the image did not have a white background, or if it wasn't bright enough for the text to be easily read.

After pressing "Submit" on the article selection screen, Sam is taken to the next window, the article preview screen.

Here, the text from the image is divided into logical blocks, each representing a section of the original article. OCRticle correctly identified the article's title, but it didn't identify the quote block as being a quote, instead labeling it as "TEXT". Thus, Sam proceeds to click on the "TEXT" button besides the corresponding block, which opens a drop-down menu where they can select the "QUOTE" option, changing the block into a quote block.

The first two paragraphs of the text were split into different blocks because of the empty space in the image caused by the removal of the advertisement. Therefore, Sam presses the "Merge above" button on the second text block, in order to merge it with the first.

Sam also selects the "Keep line breaks" option, since they want the final text to match the original image as best as possible.

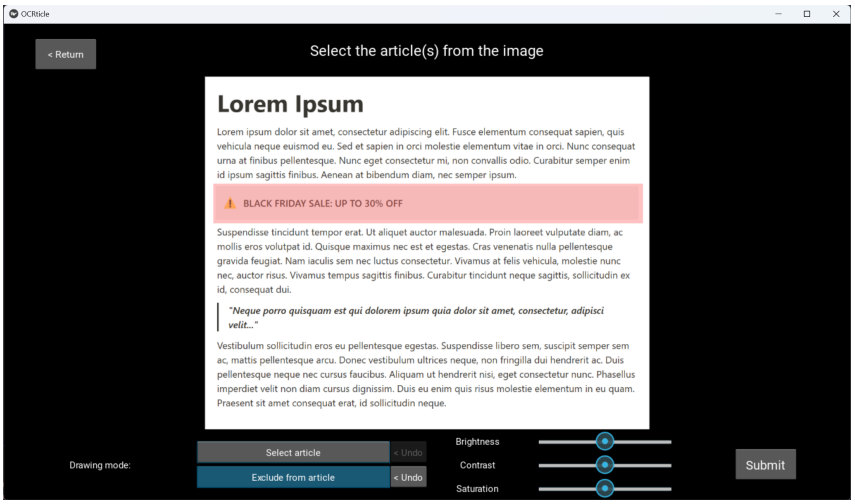

Figure 5 Article selection screen. The red rectangle represents an "exclusion zone" that won't be considered by OCRticle.

Figure 7 shows the same screen as Figure 6, but now with Sam's changes applied to the article.

Finally, Sam presses the "Save article(s)" button, which takes them into the final screen of the application, where Sam is asked to save a file containing the text from the previous screen, correctly formatted. For this example, the generated file can be seen in Figure 8.

Both the article's title and the quote are correctly identified, according to Markdown syntax.

6 Case study

Throughout OCRticle's development, several tests were conducted with all kinds of images. In order to verify how OCRticle fared against real pictures, instead of just computer screenshots, we used pictures of newspaper clippings or pages. One of those tests, which was performed with a Portuguese newspaper from 1928, is described below.

Figure 9 shows the newspaper page in question, which was given to both Tesseract and OCRticle.

Due to the color of the paper and to the fact that Tesseract's Portuguese dictionary does not contain some of the old words used in these articles, the results from just using Tesseract for OCR are rather poor.

The full output was too large to be included in this paper, but this snippet, which corresponds to the top left article from Figure 9 shows clearly that plain Tesseract does a bad job at detecting text from this image:

```
Instalação da Comissão Administra-
realisada no dia à do corrente.

| Snr. Presidente da C. A. da
"pelo Snr. Governador Ci-
rmos do $ 3.º do art. 2.º
eto de 31 de Dezembro
"nomeado Administrador
ncelho, cargo que actual-
erce e assim nos termos
```

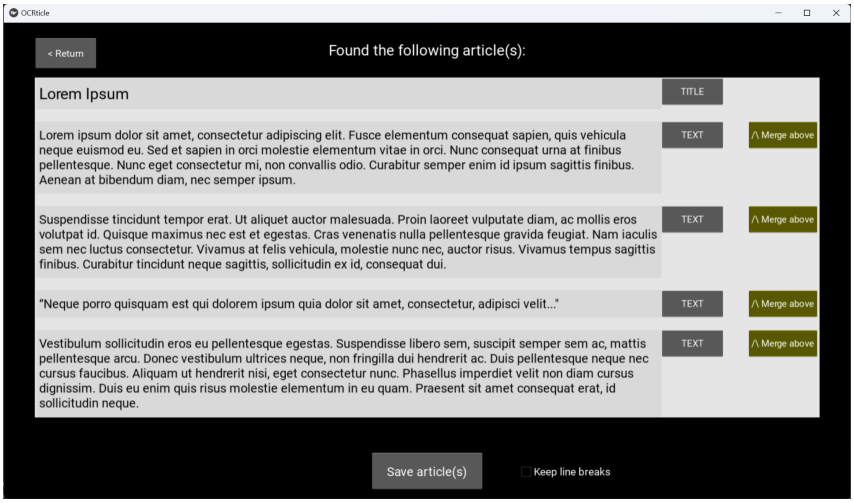

Figure 6 Article preview screen.

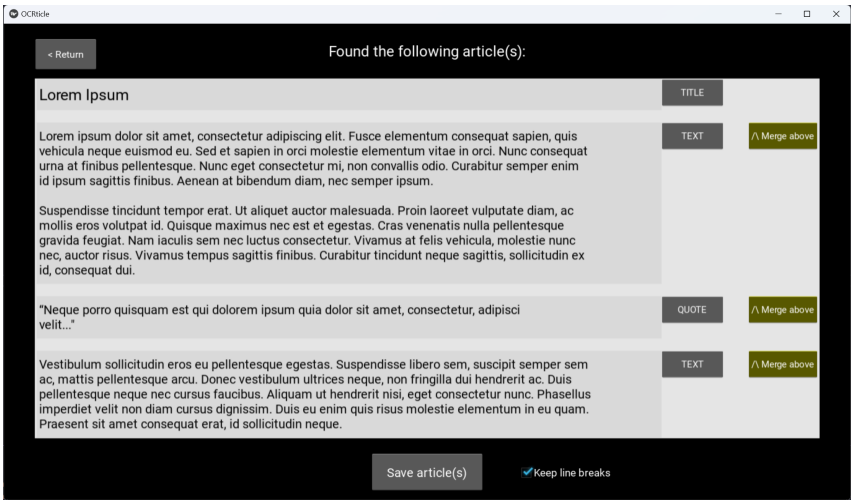

Figure 7 Article preview screen after Sam's changes.

```
1 presidiu a esta sessão.

Designou o dia para as ses-
uartas feiras pelas 15

gusto Barreira.

expostos; Dr.

jaldio:
ndes

juzir esta secção.

o - Mesa da Camara:

te Dr. Gonçalo Monteiro
; Vice-presidente Dr.
aquim Machado Guima-
```

```
# Lorem Ipsum



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce elementum consequat sapien, quis
vehicula neque euismod eu. Sed et sapien in orci molestie elementum vitae in orci. Nunc consequat
urna at finibus pellentesque. Nunc eget consectetur mi, non convallis odio. Curabitur semper enim
id ipsum sagittis finibus. Aenean at bibendum diam, nec semper ipsum.

Suspendisse tincidunt tempor erat. Ut aliquet auctor malesuada. Proin laoreet vulputate diam, ac
mollis eros volutpat id. Quisque maximus nec est et egestas. Cras venenatis nulla pellentesque
gravida feugiat. Nam iaculis sem nec luctus consectetur. Vivamus at felis vehicula, molestie nunc
nec, auctor risus. Vivamus tempus sagittis finibus. Curabitur tincidunt neque sagittis, sollicitudin ex
id, consequat dui.



> "Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur, adipisci
> velit..."



Vestibulum sollicitudin eros eu pellentesque egestas. Suspendisse libero sem, suscipit semper sem
ac, mattis pellentesque arcu. Donec vestibulum ultrices neque, non fringilla dui hendrerit ac. Duis
pellentesque neque nec cursus faucibus. Aliquam ut hendrerit nisi, eget consectetur nunc. Phasellus
imperdiet velit non diam cursus dignissim. Duis eu enim quis risus molestie elementum in eu quam.
Praesent sit amet consequat erat, id sollicitudin neque.

```

Figure 8 File containing Sam's article.

```
Secretario Juão Rodrigues.
ice-secretario Guilher-

ibuição de plouros: Presi-
Instrução, Fazenda, Po-
José J.
ado Guimarães - Taipas e
e; João Rodrigues Lourei-
Ss, Aguas e Incendios ;
Ribeiro Guima-
Pevidem; Guilhermino Au-
Barreira - Obras, Viação,
ro, Limpeza e Cemiterio;
os Pereira Mendes - Im-
Feiras, Mercados, e Luz;
o Alves - Vizela.
```

On the other hand, with OCRticle, text detection improves a fair amount, substantially in some cases. The last two articles from the original image were detected and processed by OCRticle as follows (consecutive blank lines have been suppressed in order to preserve space):

```
rto, 3

ate da a Cates.

Maior,

tos

tie.

to Meão de 5 comr Dr. o Rigusto

as

gos com mais renc Por is: Comissão a sua ati para con verba n.º AE to ordina sob a ileg
des» put oficial da proximo | na «Desp rubrica «] económic:

Que los e aprovac plementar começo a:

# Viagem de estudo
```

4 O CONQUISTADOR

DOMUS MUNICIPALIS

Sessão da Instalação da Comissão Administrativa da Camara, realisada no dia 2 do corrente.

O Snr. Presidente da C. A. da C. foi pelo Snr. Governador Civil nos termos do § 3.º do art. 2.º do Decreto de 31 de Dezembro de 1927, nomeado Administrador do Concelho, cargo que actualmente exerce e assim nos termos da Lei presidiu a esta sessão.

— Designou o dia para as sessões ás quartas feiras pelas 15 horas.

Eleição — Mesa da Camara: Presidente Dr. Gonçalo Monteiro de Meira, Vice-presidente Dr. José Joaquim Machado Guimarães, Secretario João Rodrigues Loureiro, Vice-secretario Guilhermino Augusto Barreira.

Distribuição de pelouros: Presidencia — Instrucão, Fazenda, Policia e expostos: Dr. José J. Machado Guimarães — Taipas e higiene; João Rodrigues Loureiro — Baldios, Aguas e Incendios; José Mendes Ribeiro Guimarães — Pevidem; Guilhermino Augusto Barreira — Obras, Viacão, Matadouro, Limpeza e Cemiterio; Domingos Pereira Mendes — Impostos, Feiras, Mercados, e Luz; Francisco Alves — Vizela.

Sessão no dia 8

N. da R. E' nosso proposito dar um relato quanto possivel completo das reuniões da C. A. da C. M. de G. No presente numero, porém, porque desejamos arquivar, com merecido destaque e elogio a proposta apresentada ontem, pelo Presidente da C. A. da C. Snr. Dr. Gonçalo Monteiro de Meira, o que fazemos em primeira mão, vêmo-nos forçados a reduzir esta secção.

Sob a Presidencia do Snr. Dr. Gonçalo Meira e tendo comparecido os vereadores Snrs. José Machado, João Rodrigues Loureiro, Guilhermino Augusto Barreira, Domingos Pereira Mendes e Francisco Alves realisou-se a sessão pelas 20,30 horas.

Lida pelo chefe da Secretaria, Sr. Gomes Alves, a acta da sessão anterior, que foi aprovada, e do expediente, foi tomado conhecimento de varios oficios que tiveram despacho.

Entre estes damos publicidade ao do Comandante da 1.ª Região Militar Sr. Coronel Craveiro Lopes:

Ex.mo Sr.— Sua Ex.a o Comandante da Região encarrega-me de transmitir a V. Ex.a os seus agradecimentos pelas saudações com que V. Ex.a se dignou distinguilo após o acto de posse da Comissão a que tão dignamente preside e de assegurar que dentro dos limites das suas atribuições, sempre procurará defender os legitimos enteresses duma terra merecedora de tôda a consideração, não só pelas suas tradições historicas como ainda pela feição especial do seu povo ordeiro, patriotico e trabalhador.

Saude e fraternidade.

Quartel General no Porto, 3 de Fevereiro de 1928.

Ao Ex.mo Senhor Presidente da Comissão Administrativa da Camara Municipal de Guimarães.

O Chefe do Estado Maior, interino
Daniel de Matos
major.

Pôsto Médico e Repartição de Saúde.

Deliberou encerrar o Posto Medico e nomear uma Comissão de remodelação destes serviços composta pelos Srs. Vereador Dr. José Machado, Dr. Alberto Ribeiro de Faria e Dr. Augusto Cunha.

Casas Económicas

A proposta apresentada pelo Sr. Dr. Gonçalo Meira.

«A reconhecida falta de habitações necessita de ser urgentemente solucionada. O chamado problema do inquilinato, não se resolve, com as publicadas leis do inquilinato ou outras que sigam o mesmo critério; não se resolve, como não se resolveu, somente garantindo aqueles que conseguiram uma habitação a sua permanência nela. Do facto de haver mais inquilinos que casas e destas se encontrarem, na maioria dos casos, em péssimas condições higienicas, é que decorre a gravidade da situação presente, situação de que é preciso sair sem delongas.

Faltam casas porque as não há e porque muitas das que existem se acham em deploravel estado de aceio e conforto que é urgente modificar. Assim encarado o problema a sua solução consista simplesmente na construção de casas habitaveis.

Não pode esperar se que o capital procure a sua remuneração aplicando-se na construção de casas para aluguer pois que êle tem presentemente outros empregos com menos riscos e de muito mais rendimento.

Por isso proponho: que esta Comissão Administrativa dê toda a sua atenção a êste assunto e para começar, determine que a verba n.º 118 de 12 000$00 do art.º 1.º capitulo 12 do orcamento ordinario que lá se encontra sob a ilegal rubrica de «festividades» publicas e representação oficial da camara seja inscrita no proximo orçamento suplementar na «Despesa facultativa» sob a rúbrica «Edificação de habitações económicas» e

Que logo que esteja organisado e aprovado aquele orçamento suplementar imediatamente se dê começo aos trabalhos necessarios para que dentro de um breve praso seja construido e acabado o maior numero possivel de casas economicas.

Desnecessário è encarecer o alcance social que terá a efectivação dum tal empreendimento porque não há hoje quem ignore que um grande numero de doenças, entre elas a terrivel tuberculose, se propagam com extrema facilidade em habitações acanhadas, faltas d'ar e luz onde a população se comprime numa percentagem enorme tal como entre nós sucede, concorrendo em muito a promiscuidade de sexos e de edades no mesmo compartimento para o baixo nivel moral, a que, em bastantes casos, se tem descido».

Esta proposta, à qual muito em breve faremos as merecidas e louvaveis referencia que nos sugere, foi unanimemente aprovada.

Viagem de estudo

Acompanhados de dois Professores, os Snrs. Drs. Vieira Brito e Correia Cardoso, chegaram ante-ontem aqui, no comboio das 19,30, 47 alunos do Liceu Central José Falcão, de Coimbra.

Visitaram o nosso Liceu, Castelo, Sociedade Martins Sarmento e vários Monumentos que muito apreciaram, lastimando, porem, o estado de ruina em que se encontram os Claustros da Oliveira.

Os nossos simpaticos hospedes deram hontem um espectaculo em o nosso *S. Carlos*,

O programa anunciado foi cumprido rigorosamente deixando nos espectadores agradavel impressão.

Houve alegria e graça, sendo os simpaticos rapazes muito aplaudidos. «Pepita Graiêra», a mais linda *hespanhola* que até hoje nos tem visitado, e que mereceu as honras da noite, na exibição dos seus apreciadissimos bailados, ficou muito penhorada ao ser destinguida com os camarins do palco, graca que a empresa exploradora do D. Afonso, só costuma conceder às grandes celebridades, que raro veem a Guimarães.

A formosíssima *hespanhola*, das margens do Mondego, impreciosissima com tanta gentileza por tão luxuoso aposento, teve expressões amaveis para o «arquitecto» amador Snr. Luiz do Souto a quem se deve aquela *obra d'arte*. E num requinte de galanteria, que muito sensibilisou o Snr. Souto, ofereceu-lhe, como singela prova de gratidão, uma almotolia de azeite para desinferrujar a velha e arrelienta engrenagem do pano de boca.

O Snr. Souto ultra sensibilisado, corou... sorriu... e osculou a mão graciosa da apreciada «tonadillera».

—No espectaculo dos estudantes do Liceu de Coimbra presenceamos um facto que a autoridade tem de reprimir.

Jogou-se ali o carnaval, como é proprio da quadra que atravessamos, mas o que é impróprio, o que é mesmo imperdoavel, é que as serpentinas se apanhem, em rôlos, do chão e se joguem para os camarotes ou para os espectadores da plateia.

Em terra alguma que se diga civilizada tal abuso é imediatamente castigado.

Além de ser anti-higienico, denota pelintrice.

Quem não tem dinheiro pastante para se divertir deixa-se estar quêdo, como está a grande maioria.

Paro o caso chamamos a atenção da Autoridade, e da propria Direcção do Teatro.

Ou não teremos razão?!

Ultima notícia

As 3 horas

Foi hoje notificado o chefe da Repartição Municipal de Saúde do encerramento do Posto Médico, cessando as gratificações e salários ao pessoal-

Figure 9 Image of a 1928 Portuguese newspaper used for testing.

Acompanhados de dois Professores, os Snrs. Drs. Vieira Brito e Correia Cardoso, chegaram ante- -ontem aqui, no comboio das 19,30, 47 alunos do Liceu Central José Falcão, de Coimbra.

Visitaram o nosso Liceu, Castelo, Sociedade Martins Sarmento e vários Monumentos que muito apreciaram, lastimando, porem, o estado de ruina em que se encontram os Claustros da Oliveira.

Os nossos simpaticos hospedes deram hontem um espectaculo em o nosso S. Carlos,

O programa anunciado foi cumprido rigorosamente deixando nos espectadores agradavel impressão.

Houve alegria e graça, sendo os simpaticos rapazes muito aplaudidos. «Pepita Graiêra», a mais linda hespanhola que até hoje nos tem visitado, e que mereceu as honras da noite, na exibição dos seus apreciadissimos bailados, ficou muito penhorada ao ser destinguida com os camarins do palco, graca que a empresa exploradora do D. Afonso, só costuma conceder às grandes celebridades, que raro veem a Guimarães.

A formosíssima hespanhola, das

Ultima notícia

As 3 horas

Foi hoje notificado o chefe da Repartição Municipal de Saúde do encerramento do Posto Médico, cessando as gratificações e salários ao pessoal:

On the last article, OCRticle was able to maintain 100% of the original text and structure. However, the other article has many problems on the first half. This is mostly due to the fact that this article's title, on the original image, is below some of the text, which causes Tesseract to become confused.

Figure 10 shows how the articles were selected in OCRticle. Moreover, the image was converted to black-and-white and its brightness and contrast were increased using OCRticle's sliders, which also improved text detection.

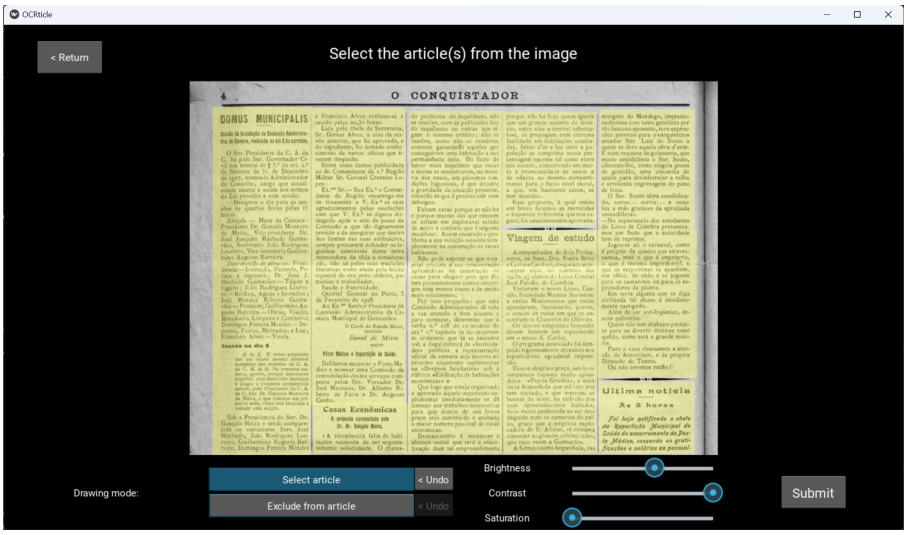

Figure 10 Article selection screen for the 1928 newspaper image.

The issue with the penultimate article can be alleviated by selecting the two columns as separate articles (i.e., as non-intersecting rectangles) in OCRticle. In Figure 10, since the two rectangles over the article are intersecting, OCRticle considers them to be part of the same article. When using two different rectangles, the following output is obtained:

```
# Viagem de estudo


Acompanhados de dois Professores, os Snrs. Drs. Vieira Brito e Correia Cardoso, chegaram
ante- -ontem aqui, no comboio das 19,30, 47 alunos do Liceu Central | José Falcão, de
Coimbra. -

Visitaram o nosso Liceu, Cas- | telo, Sociedade Martins Sarmento e vários Monumentos que
 muito apreciaram, lastimando, porem, o estado de ruina em que se encontram os Claustros
 da Oliveira.

Os nossos simpaticos hospedes deram hontem um espectaculo em o nosso S. Carlos,

O programa anunciado foi cumprido rigorosamente deixando nos espectadores agradavel
impressão.

Houve alegria e graça, sendo os simpaticos rapazes muito aplaudidos. «Pepita Graiêra», a
mais linda hespanhola que até hoje nos tem visitado, e que mereceu as honras da noite, na
exibição dos seus apreciadissimos  bailados, ficou muito penhorada ao ser destinguida com
os camarins do palco, graca que a empresa exploradora do D. Afonso, só costuma conceder
às grandes celebridades, que raro veem a Guimarães.

A formosíssima hespanhola, das
```

```
---

margens do Mondego, imprecionadíssima com tanta gentileza por tão luxuoso aposento, teve
expressões amaveis para o «arquitecto» amador Snr. Luiz do Souto a quem se deve aquela
obra d'arte. E num requinte de galanteria, que muito sensibilisou o Snr. Souto,
ofereceu-lhe, como singela prova de gratidão, uma almotolia de azeite para desinferrujar
a velha e arrelienta engrenagem do pano de boca.

O Snr. Souto ultra sensibilisado, corou... sorriu... e osculou a mão graciosa da
apreciada «tonadillera». -No espectaculo dos estudantes do Liceu de Coimbra presenceamos
um facto que a autoridade tem de reprimir.

Jogou-se ali o carnaval, como é proprio da quadra que atravessamos, mas o que é impróprio,
o que é mesmo imperdoavel, é que as serpentinas se apanhem, em rôlos, do chão e se joguem
para os camarotes ou para os espectadores da plateia.

Em terra alguma que se diga civilizada tal abuso é imediatamente castigado.

Além de ser anti-higienico, denota pelintrice.

Quem não tem dinheiro pastante para se divertir deixa-se estar quêdo, como está a grande
maioria.

Paro o caso chamamos a atencão da Autoridade, e da propria Direcção do Teatro.

Ou não teremos razão?!

---
```

While not a perfect solution, it substantially improves text detection. One would just need to delete the three dashes between both text blocks (they are introduced by OCRticle to separate distinct articles within the same file) in order to manually merge both articles.

The only remaining issue with OCRticle's output is the occasional appearance of vertical bars, colons or hyphens which don't appear in the original text. This happens because the paper on the original image has some imperfections, like folds or signs of age, which confuse Tesseract. While converting the image to black-and-white fixes some of these issues, it's impossible to fully remove these imperfections without manually editing the image. Another method to fix these errors would be to develop an algorithm to detect these "intruders" and remove them. However, this introduces a new problem, i.e., how do we detect if a vertical bar or other erroneous text element is supposed to be in the text or not. With OCRticle, since the main focus was text detection and not necessarily text correction, we followed a more conservative approach, and decided to let each individual user deal with these imperfections, instead of trying to automatically fix them. However, a future version of OCRticle could possess such a feature, even if optional or togglable.

7 Conclusion

This paper showcased OCRticle, a prototype software capable of performing Optical Character Recognition on articles while maintaining the article's original structure. OCRticle is able to successfully receive an image with its articles highlighted and extract them in textual format. It's able to automatically detect some types of text blocks, like titles, marking them as such in the generated Markdown file. The case studies shown in the previous section demonstrated that, although not perfect, OCRticle is a much more valuable solution than just using Tesseract when performing OCR on an image with one or more articles, even more so if one wishes to keep the text's original structure.

7.1 Release

OCRticle is publicly available for download and installation on PyPI, at `https://pypi.org/project/ocrticle/`, and on GitHub, at `https://github.com/RisingFisan/OCRticle`.

On Unix systems, it can be installed by first installing Tesseract and then running `pip install ocrticle`. On Windows systems, due to external dependencies, installation using `pip` may fail. Therefore, an executable file is available for download on the project's GitHub page. Tesseract installation is still required for Windows users.

References

1 What is ocr (optical character recognition)? - aws. URL: `https://aws.amazon.com/what-is/ocr/`.
2 Matt Cone. Markdown guide. URL: `https://www.markdownguide.org/`.
3 Freeocr. URL: `http://www.paperfile.net/`.
4 Search what you see. URL: `https://lens.google/`.
5 Trey Harris. Converting a scanned document into a compressed, searchable pdf with redactions, September 2022. URL: `https://medium.com/@treyharris/converting-a-scanned-document-into-a-compressed-searchable-pdf-with-redactions-63f61c34fe4c`.
6 Google answers whether it's better to ocr text in pdfs or not, August 2022. URL: `https://iloveseo.com/seo/google-answers-whether-its-better-to-ocr-text-in-pdfs-or-not/`.
7 An introduction to markup. URL: `https://port.sas.ac.uk/mod/book/view.php?id=568&chapterid=336`.
8 Kivy: Cross-platform python framework for gui apps development. URL: `https://kivy.org/`.
9 Kaan Kuguoglu. How to use image preprocessing to improve the accuracy of tesseract, July 2021. URL: `https://towardsdatascience.com/getting-started-with-tesseract-part-ii-f7f9a0899b3f`.
10 12+ best free ocr software for windows [2022 updated list], September 2022. URL: `https://www.softwaretestinghelp.com/ocr-software-for-pc/`.
11 Ocrspace. URL: `https://ocr.space/`.
12 Image to text converter using ocr online. URL: `https://www.onlineocr.net/`.
13 Pytesseract. URL: `https://pypi.org/project/pytesseract/`.
14 Online ocr - free ocr pdf document scanner & converter. URL: `https://www.sodapdf.com/ocr-pdf/`.
15 Tesseract user manual. URL: `https://tesseract-ocr.github.io/tessdoc/`.
16 Improving the quality of the output - tesseract documentation. URL: `https://tesseract-ocr.github.io/tessdoc/ImproveQuality.html`.
17 Tesseract.js: Pure javascript ocr for 100 languages! URL: `https://tesseract.projectnaptha.com/`.
18 Languages supported in different versions of tesseract. URL: `https://tesseract-ocr.github.io/tessdoc/Data-Files-in-different-versions.html`.
19 TheJoeFin. Thejoefin/text-grab: Use ocr in windows 10 quickly and easily with text grab. with optional background process and popups. URL: `https://github.com/TheJoeFin/Text-Grab`.
20 James Vincent. Google lens can now copy and paste handwritten notes to your computer, May 2020. URL: `https://www.theverge.com/2020/5/7/21250556/google-lens-copy-paste-handwritten-notes-computer-phone-ios-android`.