Transcribing History with Tesseract: A Monomodal OCR Approach in the PastReader 2025 Shared Task

Jaione Macicior-Mitxelena[1,*,†]

[1]Universidad Nacional de Educación a Distancia (UNED), Madrid, Spain

Abstract

This paper presents the results of our participation in the PastReader 2025 shared task, which focused on the automatic transcription of historical Spanish documents. We explored a monomodal OCR approach using the open-source Tesseract engine, evaluating both its baseline performance and the effects of domain-specific fine-tuning. Experiments were conducted on a dataset of digitized newspaper pages provided by the Biblioteca Nacional de España (BNE). While Tesseract offered a solid baseline, fine-tuning with task-specific data did not yield consistent improvements, revealing challenges related to data heterogeneity, layout complexity, and OCR model generalization. Our findings highlight the limitations of monomodal approaches for noisy historical documents and suggest future directions involving layout-aware and multimodal methods.

Keywords

OCR, Historical Documents, Digital Humanities, Tesseract, Finetuning

1. Introduction

Access to the contents of digitized historical documentation in Spanish is usually achieved through transcriptions made using applications not designed for the characteristics of this type of texts, such as the use of old Spanish, decorations or obsolete fonts, poor scan quality, blots, manual notes in the margins and others, since they do not normally appear in most of the current texts. With the rise of the Digital Humanities [1] and increasing recognition of the importance of digitizing and preserving analog materials, Optical Character Recognition (OCR) has long been a central area of research and innovation due to its broad practical applications, renewed relevance and urgency. For this reason, the Spanish Society for Natural Language Processing (SEPLN) has organized at the Iberian Languages Evaluation Forum (IberLEF 2025) [2] the PastReader shared task, *Transcribing Texts from the Past*.

The remainder of this paper is organized as follows. To begin with, Section 2 provides a short background on the shared task and the dataset, and also outlines the primary methodology employed. Subsequently, Section 3 presents a detailed analysis of the dataset, highlighting several challenges that adversely affected model performance. Following this, Section 4 offers an in-depth description of the two submitted runs, explaining their respective configurations and outcomes. Finally, Section 5 concludes the paper with a summary of the key findings and some final reflections on the overall approach and its implications.

2. Short Description of the Shared Task and Dataset

The PastReader 2025 shared task [3] focuses on the automatic transcription of historical printed texts with the aim of improving OCR outputs on digitized Spanish newspapers and documents. This task addresses a critical need within the Digital Humanities: the preservation and enhanced accessibility of historical documents that present significant challenges for modern text processing systems due to their age, print quality, and complex layout.

IberLEF 2025, September 2025, Zaragoza, Spain

*Corresponding author.

†These authors contributed equally.

jmacicior2@alumno.uned.es (J. Macicior-Mitxelena)

0009-0001-7392-4226 (J. Macicior-Mitxelena)

Even if the competition is divided into two main subtasks, due to GRESEL1 time constraints only applying for the second task *End-to-End Transcription* was considered. In this demanding subtask, participants are required to develop systems that process scanned images of historical newspaper pages and output accurate, structured transcriptions.

The dataset used in this shared task is derived from the Hemeroteca Digital of the Biblioteca Nacional de España (BNE) [4]. This open-access digital archive includes historical Spanish press publications in the public domain [1].

OCR quality varies considerably due to factors such as the condition of the original documents, the complexity of the layout, and the OCR technology used at the time of digitization. For the purposes of the shared task, the dataset was partitioned into training, development, and test sets, using stratified sampling to ensure representative coverage across the collection. The training set consists of 8,959 pages and includes scanned PDFs, the original OCR output, and manually corrected reference transcriptions. The development set comprises 500 pages with the same structure—PDF scans, OCR text, and corrected ground truth—allowing participants to fine-tune and validate their models prior to submission. The test set for Task *End-to-End Transcription*, contains 3,000 scanned PDF pages, without accompanying transcriptions.

3. Approach Description

The two experiments made use of the Tesseract OCR engine. Its open-source nature and adaptability make it a practical choice for initial OCR tasks on historical documents. Tesseract is an OCR engine developed by Hewlett-Packard and later maintained by Google that supports over 100 languages and includes features such as layout analysis and recognition of various image formats. Its accessibility and flexibility make it a popular choice for digitizing printed texts. However, Tesseract's performance can be challenged when dealing with historical documents that often contain noise, complex layouts, or uncommon fonts. A benchmarking study comparing Tesseract with Amazon Textract and Google Document AI found that server-based processors outperformed Tesseract, especially on noisy documents [5]. Despite these limitations, Tesseract remains a valuable tool for establishing baseline OCR performance and serves as a useful point of comparison when evaluating more advanced or specialized OCR systems.

Recent research has explored enhancing Tesseract's capabilities for historical documents. For instance, a study focused on 19th-century printed serials employed a combined machine learning approach, fine-tuning Tesseract with synthesized and manually annotated data to improve OCR quality [6]. This approach demonstrated that integrating monomodal OCR techniques with comprehensive layout analysis can significantly enhance text recognition accuracy in historical documents. For instance, Fleischhacker et al. (2024) reported substantial reductions in Character Error Rate (CER) and Word Error Rate (WER) by combining structure detection with fine-tuned OCR processes. Similarly, the OCR4all framework provides a semi-automatic OCR workflow that incorporates layout analysis and continuous model training, leading to improved OCR performance in historical printings [7].

Despite these advancements in OCR methods that combine text recognition with layout analysis, our approach in this shared task focused solely on monomodal, text-based techniques. This decision was driven by the absence of preprocessed layout information in the dataset and time constraints that limited the feasibility of integrating comprehensive layout-aware processing. Consequently, our experiments prioritized transcription performance based only on textual content. Initially, Tesseract was applied in its baseline configuration to establish a performance benchmark. The model was then fine-tuned using the annotated training data provided by the task organizers, in an effort to improve accuracy by adapting to the dataset's specific characteristics.

These strategies, ranging from baseline to fine-tuned OCR models, represent a progression from general-purpose to more task-specific solutions. The following sections provide a detailed description of each approach. Future work will explore the incorporation of layout-aware techniques to further enhance OCR performance in complex historical documents.

[1] https://hemerotecadigital.bne.es/hd/es/fulltext-csv

Tesseract, as mentioned earlier, is an open-source OCR engine[2]. The installation was performed on a Windows-based system, but it can be done on other operating systems easily. Fine-tuning was implemented using the `tesstrain` training toolkit, which is compatible with the LSTM-based recognition architecture introduced in Tesseract 4.0 and maintained in later versions.

Two experimental runs were conducted to assess Tesseract's performance under different configurations. In the first run **GRESEL1_run2**, the default pre-trained Tesseract model was applied directly to the test set without any additional training or adaptation. This approach simulates a zero-shot or "off-the-shelf" application scenario, which is useful for establishing a baseline. The model was executed with standard parameters, and no preprocessing was applied beyond resizing to ensure compatibility with Tesseract's input expectations. In the second run **GRESEL1_run3**, the baseline was fine-tuned on the provided training dataset to tailor it more closely to the characteristics of the historical documents. The purpose of this fine-tuning was to determine whether training on similar materials could improve recognition accuracy—especially on degraded, low-contrast, or typographically irregular documents that typically reduce Tesseract's performance.

This two-step process was not only useful for benchmarking the Tesseract model but also served as a practical diagnostic tool for assessing the quality of the dataset itself. By observing the kinds of errors the model made—both before and after fine-tuning, it was possible to identify problematic samples, inconsistencies in labeling, and areas where document quality or transcription standards might affect downstream OCR performance. These insights were valuable for both improving model training and better understanding the dataset's challenges and limitations.

The fine-tuning was conducted using version 5.5.0 of Tesseract, installed with training support enabled (by selecting the "Install training tools" option during setup). The official Windows installer with training tools was downloaded from the Tesseract GitHub releases page. All scripts and auxiliary tools were developed and executed in a Windows environment. The tesstrain toolkit was cloned from the official GitHub repository [3] to provide the core infrastructure for training. A base Spanish model (spa.traineddata) was manually placed in a custom tessdata/ directory, and used as the initialization point for transfer learning. To streamline the process, a Python script (Finetune_tesseract.py) was developed to automate the full fine-tuning pipeline. The structure of the working directory before fine-tuning was organized as described in Figure 1 to the left.

WORKING DIRECTORY STRUCTURE

Before Fine-Tuning

```
PastReader/
|-- train/
|   |-- pdf/     <- Original PDF documents ->
|   |__ ocr/     <- Transcripts (.txt) ->
|
|
|-- finetuning/
|   |-- spa.traineddata              <- Base model ->
|   |__ spa_custom.traineddata <- Fine-tuned ->
```

After Fine-Tuning

```
PastReader/
|-- train/
|   |-- pdf/
|   |-- ocr/
|   |-- tiff/       <- tiff images (.tiff) and box files
|   |__ lstmf/  <- Intermediate LSTM training files
|-- finetuning/
|   |-- spa.traineddata
|   |__ finetuned_model.traineddata
```

Figure 1: Comparison of the working directory structure before and after applying fine-tuning.

The fine-tuning process began with transcription alignment, where all corrected text transcriptions were renamed to adhere to the .gt.txt naming convention required by Tesseract for supervised learning. This step ensures that each transcription is correctly paired with its corresponding image during

[2]https://github.com/tesseract-ocr/tesseract

[3]https://github.com/tesseract-ocr/tesstrain.git)

training. Next, image conversion was carried out. The original PDF files were transformed into TIFF format since TIFF images are preferable for Tesseract OCR as they provide a rasterised representation at approximately 300 DPI, which is optimal for recognition and compatible with Tesseract's training requirements. Following this, training file generation was performed. Bounding box annotation files (.box) were created using Tesseract in training mode, capturing the spatial alignment between characters and their positions in the image. Subsequently, LSTM-compatible training data files (.lstmf) were generated by running Tesseract with the –psm 6 (assumes a block of text) and –oem 1 (uses the LSTM OCR engine) settings. These .lstmf files encapsulate the image-text pairings needed for training the recurrent neural network.

For dataset compilation, an index file named list.txt was automatically generated. This file contained the absolute paths to all .lstmf files and served as an input manifest required by the lstmtraining binary to locate and load the training data. The model training phase was then initiated using the lstmtraining tool. The process began from an extracted LSTM file (spa.lstm), derived from the base model spa.traineddata. Training was conducted iteratively, with model checkpoints saved at regular intervals to monitor progress and facilitate recovery if needed. Finally, in the model finalisation step, the trained model weights were packaged into a usable .traineddata format using the combine_tessdata utility. The resulting file (finetuned_model.traineddata) represents the final fine-tuned model, ready for inference on historical document images. With this process, the updated structure of the project after fine-tuning should be as described above, in Figure 1 to the right.

To study the effect of dataset size on model performance, multiple fine-tuning experiments were conducted using incrementally larger subsets of the training data: 100, 1,000, 2,000 samples, and the complete dataset. For each configuration, a separate model was fine-tuned following the same procedure described earlier. The resulting models were then evaluated on a held-out development set to measure changes in recognition accuracy and to analyze the trade-off between increased training time and performance improvements.

Regarding the inference process, it was applied uniformly across all models—both the baseline and each fine-tuned variant—to ensure a fair comparison. Since Tesseract requires image input, all evaluation data originally in PDF format was converted to high-resolution TIFF images, which are known to yield better OCR accuracy. Again, TIFF files were used, and they served as the input for the inference stage. Tesseract was executed via command-line interface using the same syntax for all models. The following generic command was applied: *tesseract <input_image.tiff> <output_file> -l <model_name> –psm 6*. This command runs Tesseract using the specified language model (the model's name without the extension .traineddata), with page segmentation mode 6 (–psm 6), which assumes a uniform block of text. This setting was selected based on empirical recommendations and its suitability for the layout of historical documents used in the dataset.

Turning now to a comparison between the performance of both Tesseract runs, Figure 2 reveals a surprising trend: the baseline model (i.e., the default pre-trained model without any fine-tuning) consistently achieved the best performance across key evaluation metrics. It should be noted that these results do not correspond to the test dataset, as the training partition was used for fine-tuning and the development set for evaluation. All fine-tuned models—regardless of the number of training samples used—produced nearly identical results, none of which surpassed the baseline we established with the non-fine-tuned model. This outcome suggests that the fine-tuning process, as implemented, did not lead to meaningful performance gains and may point to limitations in the training data quality, the amount of domain-specific signal, or the need for more aggressive preprocessing or augmentation strategies. Further investigation would be required to identify the cause of this plateau and improve the efficacy of the fine-tuning approach.

Several factors may help explain this outcome. First, the training data itself may have been too heterogeneous, encompassing a wide range of fonts, layouts, languages, and document qualities (e.g., scanned pages with noise, low contrast, or skewed alignment). Such variability can make it difficult for the model to generalise effectively, especially when trained on small or inconsistent subsets. In addition, historical documents often contain typography and page structures that deviate significantly from the patterns learned by Tesseract's pre-trained LSTM model, which is optimised for contemporary

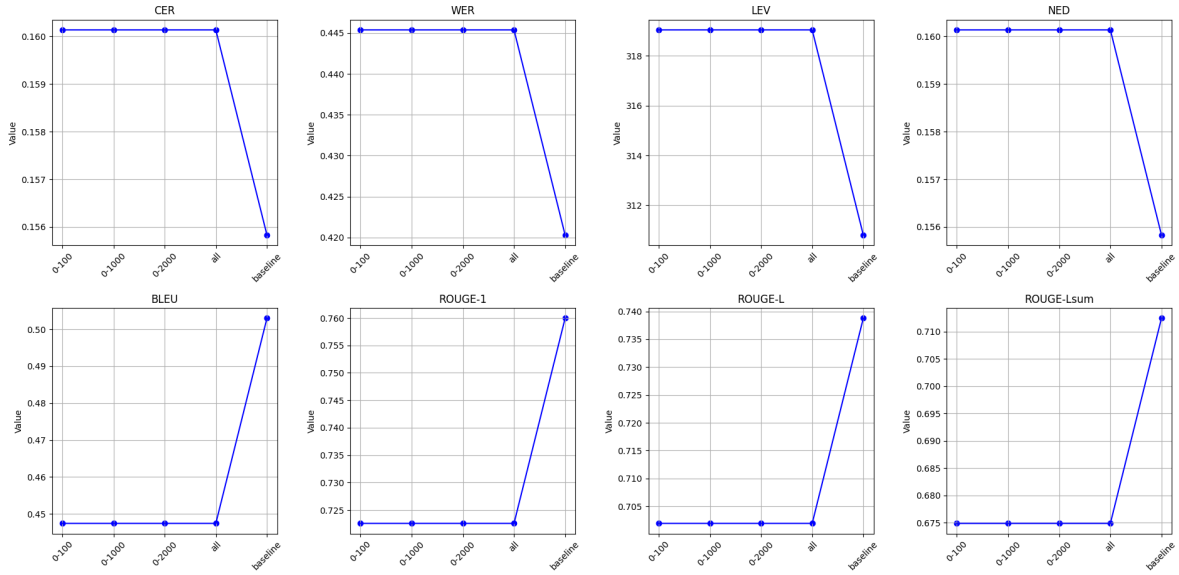

Figure 2: Tesseract charts comparing performance between fine-tuned and non-fine-tuned model.

printed text.

Another possible limitation is the alignment quality of the training pairs. Even small mismatches between the text transcriptions and the visual data can introduce noise during training, weakening the model's ability to learn accurate character-level associations. Furthermore, Tesseract's architecture is sensitive to input resolution and layout assumptions; deviations from its expected formats can lead to suboptimal performance unless carefully corrected during preprocessing. While fine-tuning Tesseract theoretically allows for domain-specific adaptation, the lack of performance gains in this study indicates that the effectiveness of this approach is heavily dependent on the consistency and quality of the training data. Future work may benefit from more targeted preprocessing, tighter control over dataset variability, or the use of alternative OCR models better suited to historical or visually degraded texts.

4. Analysis of results

4.1. Validation set: qualitative insight

To gain deeper insight into the behavior of the Tesseract models, an initial qualitative analysis was conducted based on the results of applying inference to the validation set. As a first step, it was necessary to identify the most representative samples—specifically, those for which the transcriptions yielded the poorest results. Given that the evaluation involved seven distinct metrics, the ten lowest-scoring files for each metric were initially selected. Upon comparison, it was observed that certain files appeared in multiple "top 10" worst-performing lists. Since this analysis was performed manually, attention was focused on the files that occurred in more than four of these lists. Because these files consistently showed poor performance across several metrics, they are assumed to be particularly informative for identifying common failure patterns and understanding the limitations of the models. In the figures 3 and 4 we can observe some of the most problematic cases for the baseline model and fine-tuned model (respectively).

Six files—9100, 9113, 9171, 9382, 9330, and 9088—were identified as among the worst-performing cases for both the baseline and fine-tuned models. Analysis of these files reveals several consistent failure patterns. The models tend to perform poorly on images that,

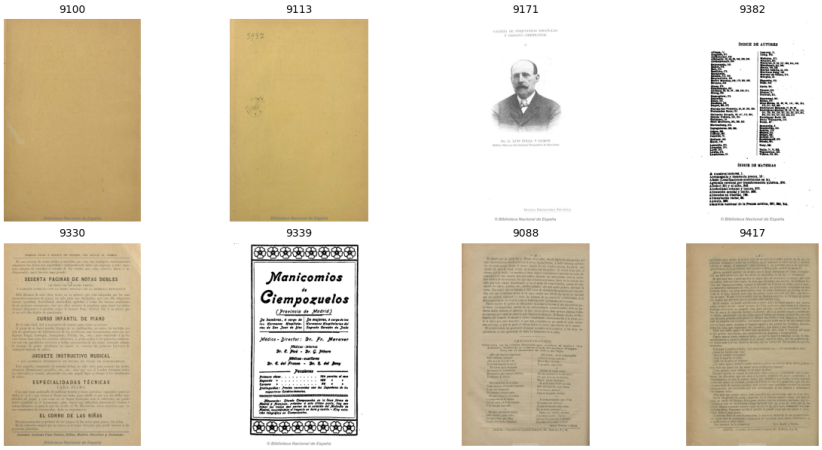

Figure 3: Images corresponding to the worst-performing files with the baseline model

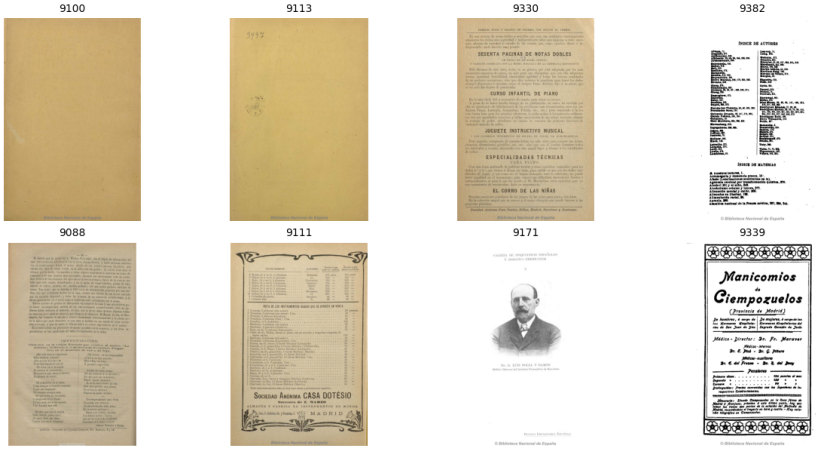

Figure 4: Images corresponding to the worst-performing files with the fine-tuned model

- contain little or no text (e.g., 9100, 9113, 9171),
- exhibit low contrast (e.g., 9330, 9088),
- feature non-traditional layouts resembling diagrams or schematics (e.g., 9382).
- include decorative elements, as seen in file 9111.

Notably, while the baseline model already struggles with such inputs, the fine-tuned model additionally fails to accurately transcribe pages with decorative elements. Probably mixing somehow the information from both images and text will benefit transcription (e.g., 9100, 9113, 9171) [8], [9], but this work is out of the scope of this paper at this moment.

Although this analysis could be extended and refined through the examination of a larger sample, the present findings offer a preliminary understanding of the models' limitations. Such qualitative assessment is essential for informing future improvements to model performance.

4.2. Quantitative analysis

In order to understand Table 2, a short description of each of our runs is provided below in Table 1.

Table 2 presents the evaluation outcomes of the shared task, as provided by the organising committee. Although the initial plan involved six evaluation metrics—Word Error Rate (WER), Sentence Error

Run ID	Description
GRESEL1_run2	Using Tesseract without fine-tuning; serves as a baseline.
GRESEL1_run3	Using the fine-tuned version of Tesseract for inference.

Table 1
Description of the submitted runs

Rate (SER), Levenshtein Distance, Normalised Edit Distance (NED), BLEU, and ROUGE—not all were ultimately adopted as expected. SER appears to have been excluded without explanation, whereas ROUGE was expanded into four distinct variants: ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-LSum. WER and Levenshtein Distance were retained, bringing the total number of metrics used to eight. This expanded metric set enables a multifaceted assessment of OCR output quality.

The evaluation framework combines both form-based and content-based approaches. On the one hand, character-level accuracy is addressed through metrics such as Levenshtein Distance and NED, which account for the number and nature of required edits, alongside WER, which reflects deviations at the lexical level. These measures prioritise exact textual correspondence, with lower scores indicating better alignment with the reference. On the other hand, semantic similarity and fluency are captured through BLEU and the various ROUGE metrics. BLEU evaluates precision in n-gram overlap, while ROUGE-1 and ROUGE-2 measure recall of unigrams and bigrams, respectively. ROUGE-L and ROUGE-LSum focus on the longest common subsequence and overall summary-level coherence. In contrast to form-based metrics, higher scores in these semantic metrics reflect improved retention of meaning and structural integrity.

Collectively, this comprehensive metric suite enables an in-depth analysis of OCR system performance, encompassing both strict textual fidelity and broader linguistic adequacy. Figure 5 complements the table by offering a visual representation of these results.

TEAM	LEVENSHTEIN	WER	NED	BLEU	ROUGE1	ROUGE2	ROUGEL	ROUGELSUM
OCRTITS_run1	56.3023	0.2344	0.0191	0.8035	0.8849	0.8065	0.8834	0.8843
GRESEL1_run3	89.1427	0.3846	0.0302	0.6220	0.8232	0.6907	0.8180	0.8226
GRESEL1_run2	93.3819	0.3650	0.0316	0.6229	0.8306	0.7114	0.8256	0.8299
BASELINE	98.4497	0.3725	0.0334	0.6083	0.8244	0.7014	0.8190	0.8237

Table 2
Official leaderboard

Table 2 presents a generally strong performance across all submitted runs, with most configurations surpassing the baseline across various evaluation metrics. This indicates a solid level of robustness and effectiveness in the approaches tested, suggesting that even the least performing configurations meet a reasonable threshold of OCR quality.

The Tesseract-based configurations—**GRESEL1_run2** and **GRESEL1_run3**—exhibit performance patterns that align with prior results obtained during training-phase experiments (refer to Figure 2). Notably, the non-fine-tuned model (GRESEL1_run2) outperformed the fine-tuned counterpart in six out of eight evaluated metrics, showing broader robustness across both surface-level and semantic evaluations. In contrast, the fine-tuned model (GRESEL1_run3) attained higher scores in only two precision-focused metrics: Levenshtein distance, where it secured the second-best overall score, and normalized edit distance (NED). These results point toward a potential overfitting effect during the fine-tuning process, wherein the model may have adapted too closely to specific features of the training data and failed to generalize effectively to unseen test samples. Nonetheless, performance gaps in semantic metrics such as ROUGE-L (0.8180 for GRESEL1_run3 vs. 0.8256 for GRESEL1_run2) and BLEU (0.6220 vs. 0.6229) remain relatively minor, indicating that fine-tuning did not drastically compromise the model's semantic coherence. Rather, it maintained competitive performance, albeit without delivering consistent improvements.

In parallel, our investigation into the environmental impact of model deployment revealed a modest

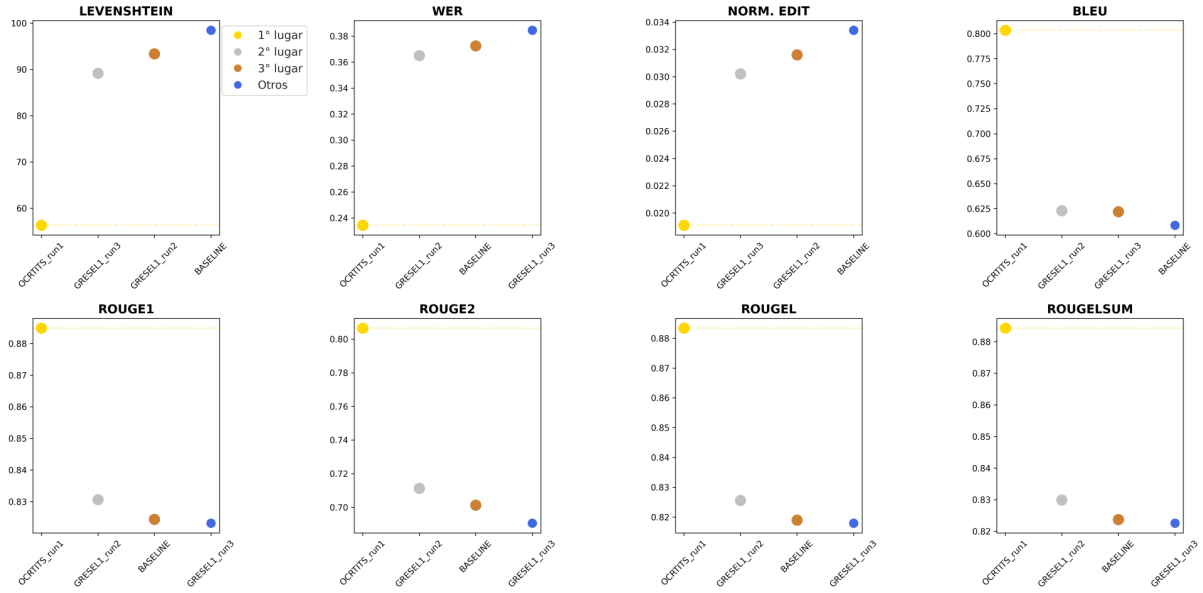

Figure 5: Scattered plots for each metric sorted by best scores.

increase in emissions associated with the fine-tuned Tesseract model, as depicted in Figure 6. Several plausible factors could underlie this discrepancy. Fine-tuning typically introduces increased computational overhead, whether due to subtle internal architectural shifts, longer inference times per input, or heightened memory requirements. Additionally, variations in preprocessing pipelines—such as the use of higher-resolution inputs or more complex image transformations—can amplify the computational burden. Although the observed increase in emissions was relatively small, it nonetheless reflects a broader trade-off inherent in machine learning systems: the tension between maximizing model accuracy and maintaining energy efficiency. These findings emphasize the importance of evaluating sustainability metrics alongside traditional performance measures, particularly when deploying models at scale or in resource-constrained environments.

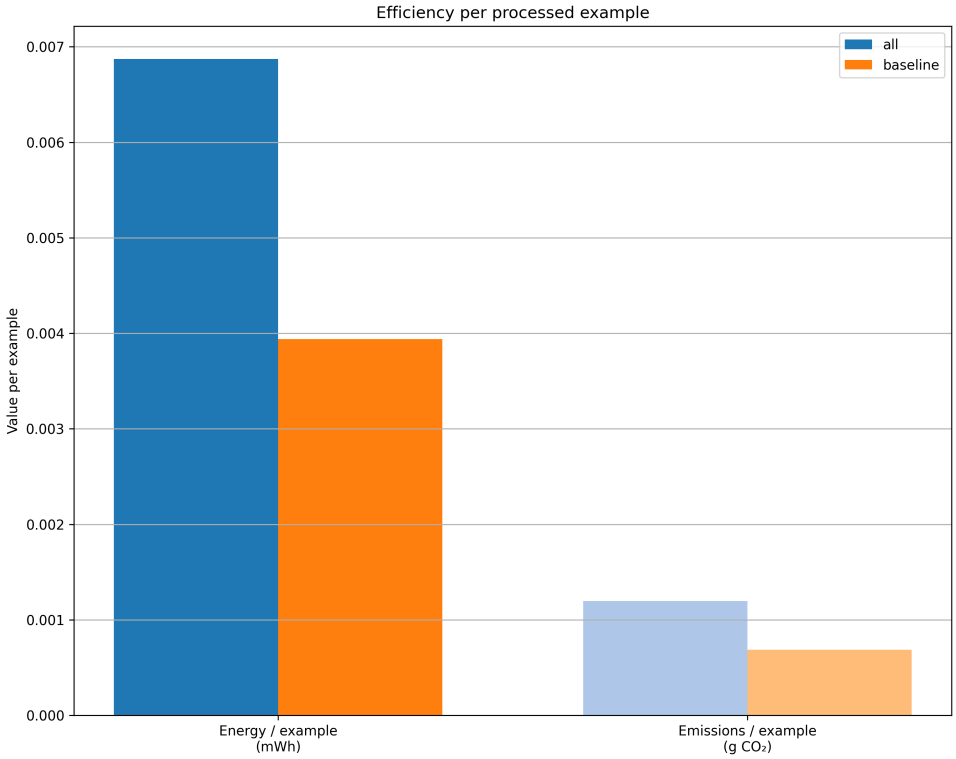

Figure 6: Energy and emission estimations

5. Conclusions

This paper presented our participation in a challenging OCR shared task using a monomodal approach. Despite the complexities posed by the dataset's diverse and heterogeneous nature, all our submissions yielded satisfactory results when compared to the baseline.

While Tesseract—particularly in its default configuration—offers a reasonable benchmark for OCR tasks involving historical documents, it lacks the accuracy and robustness demonstrated by more recent, domain-specific systems. Nevertheless, the comparison between Tesseract's baseline and fine-tuned versions proved valuable for assessing the benefits of domain adaptation and for understanding the dataset's inherent challenges.

This endeavor provided meaningful insights and learning opportunities through our experiments. Looking ahead, we plan to evaluate similarly scaled multimodal models to benchmark performance across different approaches. We aim to further improve our results in OCR tasks by continuing to explore novel strategies and methodologies.

Declaration on Generative AI

During the preparation of this work, the author utilized ChatGPT-4o for grammar and spelling correction, paraphrasing, and rewording. Additionally, Microsoft Copilot was employed for formatting support, including LaTeX commands, image labeling, and table creation. ChatGPT-4o was also used to generate the charts shown in Figures 5 and 6. All content produced with the assistance of these tools was subsequently reviewed and edited by the author, who takes full responsibility for the final publication.

References

[1] A. Garcia Serrano, A. Menta Garuz, La inteligencia artificial en las humanidades digitales: dos experiencias con corpus digitales, Revista de Humanidades Digitales 7 (2022) 19–39. URL: https://revistas.uned.es/index.php/RHD/article/view/30928. doi:10.5944/rhd.vol.7.2022.30928.

[2] J. Á. González-Barba, L. Chiruzzo, S. M. Jiménez-Zafra, Overview of IberLEF 2025: Natural Language Processing Challenges for Spanish and other Iberian Languages, in: Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2025), co-located with the 41st Conference of the Spanish Society for Natural Language Processing (SEPLN 2025), CEUR-WS. org, 2025.

[3] A. Montejo-Ráez, E. Sánchez-Nogales, G. Expósito-Álvarez, L. A. Ureña-López, M. T. Martín-Valdivia, J. Collado-Montañez, I. Cabrera-de Castro, M. V. Cantero-Romero, R. Ortuño-Casanova, Overview of pastreader shared task in iberlef 2025: Transcribing texts from the past, Procesamiento del Lenguaje Natural 75 (2025).

[4] A. Montejo-Ráez, E. Sánchez Nogales, G. Expósito Álvarez, A. Ureña López, M. T. Martín-Valdivia, J. Collado-Montañez, I. Cabrera de Castro, M. V. Cantero Romero, A. García Serrano, R. Ortuño Casanova, Y. A. Torterolo Orta, Pastreader 2025, https://doi.org/10.5281/zenodo.15084265, 2025. [Data set].

[5] T. Hegghammer, Ocr with tesseract, amazon textract, and google document ai: a benchmarking experiment, Journal of Computational Social Science 5 (2022) 861–882.

[6] D. Fleischhacker, W. Goederle, R. Kern, Improving ocr quality in 19th century historical documents using a combined machine learning based approach, arXiv preprint arXiv:2401.07787 (2024).

[7] C. Reul, D. Christ, A. Hartelt, N. Balbach, M. Wehner, U. Springmann, C. Wick, C. Grundig, A. Büttner, F. Puppe, Ocr4all—an open-source tool providing a (semi-) automatic ocr workflow for historical printings, Applied Sciences 9 (2019) 4853.

[8] E. Garcia-Arias, A. Garcia-Serrano, Creación de un modelo de descripciones de imágenes especializado en arqueología griega (en prensa), Procesamiento del Lenguaje Natural 75 (2025).

[9] A. García-Serrano, X. Benavent, R. Granados, J. M. Goñi-Menoyo, Some results using different approaches to merge visual and text-based features in clef'08 photo collection, in: C. Peters, T. Deselaers, N. Ferro, J. Gonzalo, G. J. F. Jones, M. Kurimo, T. Mandl, A. Peñas, V. Petras (Eds.), Evaluating Systems for Multilingual and Multimodal Information Access, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 568–571. doi:10.1007/978-3-642-04447-2_69.