https://doi.org/10.2352/issn.2168-3204.2025.22.1.26

Tweaking Mainstream Open-Source OCR Engine for Minority Languages, How To?

Tuomo Räisänen, Anssi Jääskeläinen, South-Eastern Finland University of Applied Sciences, Mikkeli, Finland; Atte Föhr, National Archives of Finland, Helsinki, Finland

Abstract

The digitization of historical documents is vital for preserving cultural heritage, yet mainstream OCR (Optical Character Recognition) systems often fail to support minority languages due to limited training data and language-specific models. This study explores how open-source OCR frameworks can be adapted to overcome these limitations, focusing on Finnish and Swedish as case studies. We present a practical methodology for fine-tuning PaddleOCR using a combination of manually annotated and synthetically generated data, supported by high-performance computing infrastructure. Our enhanced model significantly outperforms both Tesseract and baseline PaddleOCR, particularly in recognizing handwritten and domain-specific texts. The results highlight the importance of domain adaptation, GPU acceleration, and open-source flexibility in building OCR systems tailored for under-resourced languages. This work offers a replicable blueprint for cultural institutions seeking locally deployable OCR solution.

Motivation

Optical Character Recognition technology plays an important role in the preservation, accessibility, and analysis of printed and handwritten texts within historical archives. OCR and its handwritten counterpart, HTR (Handwritten Text Recognition), have become essential tools for unlocking the contents of physical documents and integrating them into searchable, analyzable digital ecosystems.

Agarwal and Anastasopoulos [1] emphasize the importance of data creation efforts and data-efficient algorithms. They highlight the challenges of digitizing image-based non-machine-readable documents, such as scanned dictionaries and linguistic field notes. Their recommendations include focusing on accurate layout detection and post-processing to make the extracted text usable for downstream NLP (Natural Language Processing) tasks. Compared to our approach, their study underscores the need for extensive data creation and preprocessing, which aligns with our methodology of using synthetic data generation and annotation [1].

Another study by Ignat & all [4] introduces a novel dataset for evaluating OCR systems on low-resource languages, enriched with noise to simulate real-world conditions. The study evaluates state-of-the-art OCR systems and analyzes common errors. Our approach goes further by fine-tuning PaddleOCR specifically for the Finnish and Swedish languages and leveraging high-performance hardware to optimize training time. There are also studies of enhancing other OCR engines to better meet minority languages. Microsoft Research [5] for example have introduced TrOCR, an end-to-end text recognition approach using pre-trained Transformer models, which also leverages large-scale synthetic data for pre-training and human-labeled datasets for fine-tuning. Our approach is quite similar but uses PaddleOCR as the base OCR engine.

The selection of PaddleOCR was guided by empirical experimentation using a set of various documents. While the choice reflects the current state of OCR technology, it is important to acknowledge the rapid advancement of AI. It is entirely possible that a different engine may prove more suitable in the near future as new models and frameworks emerge.

The primary objective of this work is not to advocate for a single tool, but rather to investigate whether deep learning-based OCR models can meaningfully improve recognition accuracy, especially in case of underrepresented languages.

Problem

Despite the advancements in OCR and HTR, the mainstream OCR engines such as Tesseract remain heavily skewed toward high-resource languages, where abundant training data and commercial interest have driven rapid enhancements. In contrast, minority and under-resourced languages such as Finnish and Swedish continue to be marginalized in this evolution. The main challenge lies in the lack of annotated training material and language-specific models, which are prerequisites for an accurate recognition system.

The second challenge lies within the diverse structure of documents, even within the same group. This poses a challenge in generating sufficient training and ground truth data for AI model fine-tuning. Variations in fonts, backgrounds, and image quality further complicate the OCR process, requiring sophisticated processing techniques.

Thirdly, training an OCR model demands significant computational capacity, necessitating appropriate hardware to ensure timely fine-tuning. Additionally, multilingual support, especially for minority languages, requires extensive resources.

Cloud based solutions, which are commonly offered by commercial providers, are suitable solutions for some users. However, the solution intended for the GLAM (Galleries, Libraries, Archives, Museums) sector, must often be locally installed to ensure data security and comply with regulations like GDPR [7]. Furthermore, this sector often needs to customize the solution and alter the output formats, language support etc. to fit into the existing workflows. These are aspects that can rarely be achieved with commercial or closed cloud-based solutions.

Finally, LLMs (Large Language Models) have revolutionized NLP by enabling machines to understand, generate, and manipulate human language at an unprecedented scale. However, LLMs operate on digital text, limiting their ability to process information from non-digital sources such as scanned documents, handwritten notes, or images containing text. OCR bridges this gap by converting visual text into structured, machine-readable input, thereby

expanding the applicability of LLMs. The effectiveness of this transition is highly dependent on the quality of OCR output. Recognition errors propagate through downstream tasks such as information retrieval, summarization, and question answering, this could lead to degraded performance or even misinformation.

Approach

The ground truth was produced by the NAF (National Archives of Finland) and Central Archives for Finnish Business Records' (ELKA) within an AIDA project which had limited number of human resources available. This manual work was conducted with an open-source Label Studio tool [2]. Due to limited resources, also synthetic data was generated to provide more training material for the finetuning process. There are various tools available for creating synthetic data but for this case, TRDG (TextRecognitionDataGenerator) was chosen without any specific approval process. The tool was utilized to vary fonts, text form and the background of texts. Figure 1 shows an example of the synthetic content created for the Finnish language. Texts for the synthetic data generation were harvested from Project Gutenberg free eBooks.

Figure 1. *Sample of the generated Finnish synthetic data*

Tesseract was chosen as a baseline reference for this study, as it continues to serve as a de facto standard among open-source OCR engines. Table 1 presents a comparative overview of Tesseract and the selected PaddleOCR framework and deeper differences are presented under the table. While Tesseract offers broad language support and a long-standing development history, it was ultimately not chosen for fine-tuning due to some limitations.

Table 1. Comparison of Tesseract and PaddleOCR

Feature	Tesseract	PaddleOCR
Easy of use	Simple CLI, Python API	More setup required
Accuracy (printed text)	Good	Excellent
Accuracy (handwritten)	Poor	Better with fine-tuning
Fine-Tuning Difficulty	Harder	Easier

In addition, Tesseract lacks native support for GPU acceleration, which is a significant drawback given our access to one of Finland's most powerful GPU-enabled supercomputing environments. Also, the fine-tuning process for Tesseract is considerably more complex and less flexible compared to modern deep learning-based OCR frameworks. Its architecture, while robust for general-purpose OCR, is not optimized for rapid experimentation or adaptation to new languages and scripts. In contrast, PaddleOCR emerged as a more suitable candidate for our objectives. It offers built-in GPU support, modular architecture, and out-of-the-box compatibility with over 80 languages. These features significantly reduce the overhead associated with model customization and training. After evaluating open-source OCR engines through a non-scientific test, PaddleOCR was selected for fine-tuning due to its performance, extensibility, and ease of integration into high-performance computing workflows.

Previously we have explored the potential impact of image preprocessing on OCR accuracy. The goal of this experiment was to determine whether image preprocessing techniques such as binarization, contrast adjustment or noise reductions could improve recognition performance. The results of this investigation, which are detailed in [3], indicated that preprocessing had no statistically significant effect on overall OCR accuracy across our test datasets. This is assumably due to inbuild pre-processing that is conducted within the OCR engines. Therefore, we opted not to incorporate preprocessing steps in the current study. Instead, our focus was in enhancing the OCR model itself, as this approach showed greater potential for improving recognition quality, particularly in the context of under-resourced languages.

A step-by-step guide in our case is as follows:

1. Collect data: Gather text images in target language
2. Preprocess data: Annotate images and use synthetic data generation. Separate training and evaluation sets. Prepare unseen data, to measure results.
3. Prepare data for fine-tuning: Set the data to the correct format
4. Fine-tune model
5. Evaluate and validate model performance
6. Integrate the model into the application

This general workflow is model- and language-agnostic, making it adaptable to a wide range of OCR scenarios. While the fastest way to initiate fine-tuning is by leveraging only synthetic data, this yields only preliminary results due to the domain gap between synthetic and authentic documents. Our work began with manually annotated real-world data, which provided a more accurate foundation for training and evaluation. This choice was driven by the availability of high-quality labeled datasets which were created within earlier projects [3, 8].

Results

Collect data: The data was sourced from ELKA's archive, ensuring a diverse range of materials. These included letters, company reports, and forms from the 20th century. Additionally, we annotated materials from the National Archives of Finland, which comprised of public documents collected from various sources, primarily governmental and archival records.

Preprocess data: Annotation was performed using Label Studio software. Every line of text was annotated, though not all were used in training. Especially, handwritten texts had some lines that were difficult to read and were therefore excluded from training. After

annotation, we had approximately 30,000 annotated line images for our model training. With synthetic generation, we ended up with roughly 160,000 images used for training, of which only about 8,300 were handwritten.

Prepare data for fine tuning: To prepare the dataset for fine-tuning, we developed custom Python script to convert annotated line images from the Label Studio format into the format required by PaddleOCR. This transformation step was essential to ensure compatibility with the training pipeline and to preserve the integrity of the annotations. The dataset was then partitioned into training, validation, and test subsets using a 70% / 15% / 15% split. For the primary test set, we used an unseen dataset comprising approximately 4,300 line images, on which we computed the CER (Character Error Rate) to evaluate model performance. In addition to this internal test set, two external datasets provided by the NAF were also used to further assess the generalizability of the fine-tuned model across different document types and sources.

Fine-tune model: We used PaddleOCR's Latin-PPOCR-v3 model as our base, which provided all the necessary fonts directly. Our hardware setup included an NVIDIA DGX A100 with 8x A100 GPUs, 640 GB GPU memory, and 2x AMD EPYC 7742 CPUs. We conducted performance comparisons using this hardware. Initially, we created a virtual machine like a standard laptop and found that the estimated training time would have been about 240 days (Paddle provides an approximation of training time). In a virtual machine with two A100 GPUs, full training took about one day. However, using all eight GPUs on the bare metal DGX system, the training time was reduced to approximately two hours. This highlights the importance of hardware and its utilization. Memory usage was around 40 GB per card, and average GPU performance was about 80%, demonstrating that PaddleOCR can efficiently utilize available GPU capacity.

In PaddleOCR training it is also possible to adjust the hyperparameters to reach the optimum performance with the available hardware. A hyperparameter search for number of epochs and learning rate was conducted with grid search. Our search indicated that the PaddleOCR's latin-PPOCR-v3 model's original hyperparameters were optimal, but the best results could be achieved with 25 epochs of training. Additionally, data augmentation techniques were searched in separate search. Four augmentations were selected out of the PaddleOCR's augmentation techniques. These include reverse, noise, blur and hsv_aug. Reverse inverts all the colors in the image, noise adds gaussian noise to the image, blur applies blur to the image and hsv_aug changes the colors of the image slightly. Every one of these augmentations has a 25 % change of being applied.

Evaluate and validate model: We used datasets from two different organizations, NAF and ELKA, as test data to evaluate the retrained PaddleOCR. These are the same datasets that were originally introduced in [3,8]. Despite the versatility of our training material, we wanted to ensure that the test data was entirely unseen during training. Once the model is trained, its validation is lightweight and can be performed on a standard laptop, unless it deals with a large volume of data. Character Error Rate was used as the evaluation metric, as it provides a rigorous and widely accepted measure of OCR accuracy. The CER results are summarized in Table 2 and Table 3.

Table 2. CER results from the OCR test

Model	ELKA test (4273 row pictures)	NAF1 test (3475 row pictures)
Tesseract	4,6 %	2,7%
PaddleOCR	6,7 %	3,9%
Enhanced PaddleOCR	2,0 %	1,2 %

Table 3. CER results from the OCR test

Model	NAF2 test (3247 row pictures)	ELKA test handwritten (714 row pictures)
Tesseract	4,4 %	79,3%
PaddleOCR	6,8 %	50,2%
Enhanced PaddleOCR	2,3 %	20,7%

As demonstrated by the results, fine-tuning the PaddleOCR model led to a substantial improvement in recognition performance across both datasets, highlighting the effectiveness of domain-specific adaptation.

While CER is the most used metric in OCR evaluation, other useful alternatives exist depending on the specific goals of the project. WER (Word Error Rate) is often preferred for applications where word-level accuracy affects usability, such as search or language processing. Exact match accuracy provides a stricter measure, counting only fully correct lines or words, making it suitable for structured data or form recognition. Edit distance (Levenshtein distance) can also be reported as a raw error count for simpler comparisons. Additionally, some evaluations use precision, recall, and F1 scores, especially in post-OCR correction or entity recognition contexts—to capture how well meaningful content is recovered. Choosing the right metric depends on the intended use of the OCR output and the granularity of required accuracy. In this work our focus is on developing OCR engine itself.

Figure 2 presents an example PoC (Proof of Concept) with a simple Gradio UI. This same version is up and running on https://memorylab.fi/AIDA/extended-paddle-demo/. The UI is in Finnish, but the basic utilization should be clear enough. Upload an image, press the button and wait for the results to appear. The training material used for this task is shared openly via https://huggingface.co/datasets/Kansallisarkisto/AIDA_ocr_training_data and the trained model can be found https://github.com/project-AIDA/Finnish_PaddleOCR/tree/main. Finally, the dockerized codes behind the public demo can be

accessed via GitHub https://github.com/xamkfi/digitalia-aida-extended-paddle-demo.

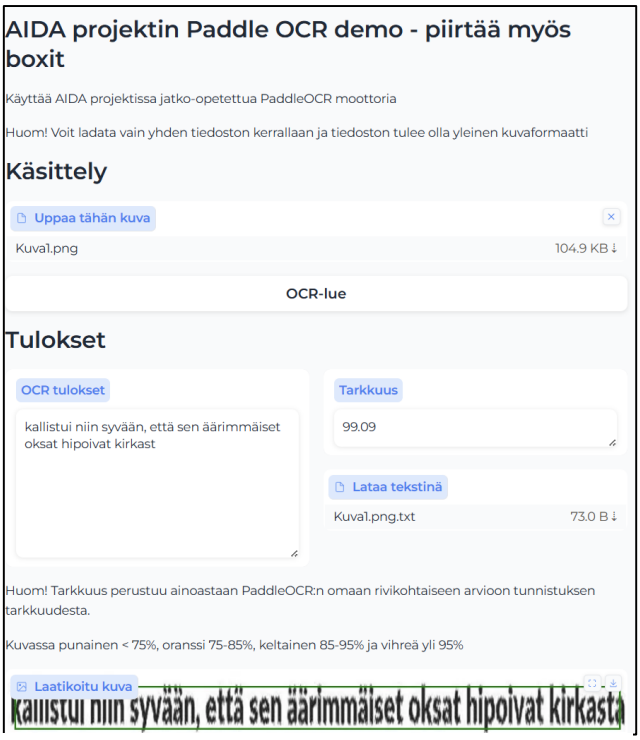

***Figure 2**. PaddleOCR running on Gradio UI*

Conclusions

This study demonstrates that mainstream open-source OCR engines can be effectively adapted to support minority languages through targeted fine-tuning, synthetic data generation, and the use of high-performance computing resources. By leveraging PaddleOCR and enhancing it with domain-specific training data, both real and synthetic, we achieved significant improvements in recognition accuracy, particularly for Finnish and Swedish texts. The enhanced model outperformed both the baseline PaddleOCR and Tesseract, especially in challenging handwritten scenarios.

Our results underscore the importance of localized, open-source solutions for the GLAM sector, where data privacy, customization, and integration into existing workflows are critical. The approach outlined in this work is scalable and adaptable, offering a replicable framework for other under-resourced languages and archival contexts.

Looking forward, further improvements in handwritten text recognition, integration with language models for post-processing, and semi-automated annotation pipelines could push the boundaries of OCR performance even further.

Further development

While the enhanced PaddleOCR model demonstrated significant improvements in recognizing printed and some handwritten texts in Finnish and Swedish, several avenues remain for future enhancement:

Improving Handwritten Text Recognition: Despite notable gains, the model's performance on handwritten documents, especially those with cursive or degraded writing, remains limited. Future work could focus on expanding the volume and diversity of handwritten training data, particularly from regional archives. Exploring hybrid models or integrating architectures like TrOCR, known for their superior handwritten text recognition, could also be beneficial, albeit with higher computational costs.

Post-OCR Structuring and Enrichment: OCR outputs are currently unstructured, consisting mainly of text and bounding boxes. A promising direction is to enrich these outputs by identifying and linking key-value pairs, especially in structured documents like forms. Leveraging large language models (LLMs) for semantic understanding and layout-aware parsing could significantly enhance the usability of OCR results in downstream applications.

Semi-Automated Annotation Pipelines: Manual annotation is resource intensive. Implementing active learning strategies, where the model suggests uncertain samples for human review, could reduce annotation effort while maintaining quality. This would accelerate dataset expansion and model refinement.

User-Friendly Deployment and Integration: To support adoption by smaller institutions, future work should focus on creating lightweight, containerized deployment packages with intuitive user interfaces. Enhancing the current Gradio-based demo into a more robust, multilingual platform could broaden accessibility and impact.

Continuous Benchmarking and Model Updating: As new documents and OCR technologies emerge, periodic retraining and benchmarking will be essential. Establishing a continuous evaluation pipeline using diverse datasets will ensure the model remains effective and adaptable over time.

References

[1] M. Agarwal and A. Anastasopoulos, "A Concise Survey of OCR for Low-Resource Languages," Journal of Computational Linguistics, vol. 40, no. 2, pp. 123-145, 2024.

[2] Label Studio. "Label Studio: Open Source Data Labeling Tool," Label Studio. [Online]. Available: https://labelstud.io/. [Accessed: Feb. 24, 2025].

[3] A. Jääskeläinen, M. Lipsanen, A. Föhr, and T. Räisänen, "OCR Quality: Key to Enhanced Data Mining," in Proceedings of the 2023 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), 2023, pp. 123-130.

[4] O. Ignat, J. Maillard, V. Chaudhary, and F. Guzmán, "OCR Improves Machine Translation for Low-Resource Languages," in Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, 2022, pp. 1164-1174. Available: https://aclanthology.org/2022.findings-acl.92/. [Accessed: May 6, 2025]

[5] M. Li, T. Lv, J. Chen, L. Cui, Y. Lu, D. Florencio, C. Zhang, Z. Li, and F. Wei, "TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models," arXiv, Available: https://arxiv.org/abs/2109.10282. [Accessed: May 6, 2025]

[6] Li, Minghao, et al. "Trocr: Transformer-based optical character recognition with pre-trained models." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 37. No. 11. 2023.

[7] P. Voigt and A. von dem Bussche, The EU General Data Protection Regulation (GDPR): A Practical Guide, 2nd ed., Springer, 2024.

[8] AIDA dataset. “AIDA OCR Training data” Huggingface [Online]. Available

https://huggingface.co/datasets/Kansallisarkisto/AIDA_ocr_training_data [Accessed: May 12, 2025]

Author Biography

Tuomo Räisänen has a PhD (2014) from the University of Jyväskylä, Finland. His current interests are in AI, large scale computing and usability, using open source tools.

Atte Föhr has a M.Sc. (Tech.) (2022) from Aalto University, Finland. Currently, he is working on extracting data from documents using machine learning approaches.

PhD Anssi Jääskeläinen works as a research manager at Xamk university He has an extensive knowledge of Open Source and AI development and he is eager to implement demo and PoC solutions with Python and Containers.