A Comparative Study on Automated Expiry Date Extraction from Official Documents Using OCR and Image Preprocessing

Alaeddin Türkmen, Barış Bayram, Ahmet Çay, Zehra Hafızoğlu Gökdağ
Data Science Team
Hepsijet
Istanbul, Turkey
e-mail: {alaeddin.turkmen | baris.bayram| ahmet.cay| zehra.gokdag}@hepsijet.com

***Abstract*—Extracting expiry dates from official documents is a critical task in numerous administrative and compliance workflows. Traditionally performed manually, this process is time-consuming, error-prone, and costly at scale. In this study, we present a comparative evaluation of multiple optical character recognition (OCR) engines combined with a diverse set of image preprocessing techniques to automate expiry date extraction from scanned and photographed documents, including insurance policies, identity cards, licenses, and inspection reports. A dataset of manually annotated portable document format (PDF) and joint photographic experts group (JPEG) files was used for benchmarking. Each image was processed using various transformations. Extracted texts were parsed using comprehensive regular expression patterns to identify date candidates, from which the latest valid date was selected as the predicted expiry. Our findings indicate that SuryaOCR, particularly when applied to unprocessed raw images, consistently outperformed other configurations, substantially reducing the need for manual intervention.**

***Keywords-OCR; Automated Document Processing; Image Preprocessing*.**

I. INTRODUCTION

Extracting expiry dates from official documents is a fundamental task in numerous administrative, regulatory, and compliance workflows. Documents, such as driver's licenses, identity cards, insurance policies, and inspection certificates often contain expiry periods that must be accurately recorded and tracked. In many organizations, this information is still collected manually, a process that is time-consuming, error-prone, and difficult to scale.

Optical Character Recognition (OCR) technologies have emerged as a practical solution for automating the extraction of textual content from scanned or photographed documents. While OCR has proven effective in many domains, its performance can vary significantly depending on document structure, image quality, and the specific recognition model employed [1]. Furthermore, expiry dates are not always consistently formatted or positioned and may appear in multiple languages, be partially obscured by stamps, seals or other occlusions, making the extraction task particularly sensitive to errors in both recognition and post-processing.

To improve accuracy and robustness, OCR pipelines are often combined with image preprocessing techniques aimed at enhancing textual features. However, there is a common but largely untested assumption that preprocessing universally improves OCR performance. As in Björkman's study [2], especially in structured and high-quality documents, preprocessing may sometimes distort visual features rather than enhance them.

This study addresses these challenges by proposing a comparative framework for automated expiry date extraction, focusing on multilingual and partially occluded real-world administrative documents. The study was conducted to determine the expiry dates of the employment documents of couriers who started working at Hepsijet, a logistics company. We evaluate the performance of three OCR engines—SuryaOCR, EasyOCR, and Pytesseract—across ten different image preprocessing techniques. The experiments are conducted on a diverse dataset of real-world administrative documents, each manually annotated with ground-truth expiry dates. Extracted texts are parsed using carefully crafted, language-aware regular expression patterns, and the latest valid date is selected as the predicted output.

Our results show that SuryaOCR, particularly when applied to raw (unprocessed) images, significantly outperforms other configurations. In contrast, many preprocessing techniques negatively impact accuracy, challenging the assumption that preprocessing is always beneficial. This paper contributes an empirical evaluation of OCR-preprocessing combinations for expiry date extraction under multilingual and occlusion conditions, offers practical insights into building more reliable automated document processing pipelines for real-world deployments.

The remainder of this paper is organized as follows. Section 2 reviews related work on OCR-based date extraction and image preprocessing techniques. Section 3 describes the dataset, including multilingual and occluded document samples, as well as the preprocessing methods, OCR engines evaluated and evaluation procedure. Section 4 as Results and Discussion, presents performance comparison across OCR engines, impact of preprocessing, error analysis and limitations & observations. Section 5 concludes the study with key findings, practical recommendations for real-world deployment, and directions for future research.

II. RELATED WORK

OCR has long evolved from rule-based systems like Tesseract to deep-learning approaches tailored for complex, real-world scenarios. Kshetry et al. introduced a modified adaptive-thresholding method that uses the dominant pixel intensity within text regions to enhance contrast, showing measurable

ISBN: 978-1-68558-303-3

gains in PyTesseract accuracy on photographs [3]. El Harraj and Raissouni designed a nonparametric pipeline—combining local brightness normalization, grayscale conversion, unsharp masking, and global binarization—that significantly improved OCR performance on mobile-captured documents [4]. Dias and Lopes applied multi-objective parameter tuning of preprocessing filters (adaptive thresholding, bilateral filtering, morphological opening) on typewritten heritage documents, demonstrating that the effectiveness of preprocessing depends on document typology [5]. Kavin and Shirley focus on automating the extraction of batch numbers and expiration dates from pharmaceutical packaging using OCR. To improve recognition performance, the authors employ preprocessing techniques and validate extracted data through rule-based checks. Their system, tested on a diverse medical image dataset, demonstrates potential for reducing manual entry errors and enhancing patient safety in healthcare workflows [6].

More recently, Tavares systematically evaluated preprocessing techniques—grayscale conversion, contrast limited adaptive histogram equalization (CLAHE), and bilateral filtering—on license plate recognition pipelines and reported that combining CLAHE with bilateral filtering yielded the highest OCR accuracy under varied lighting [7]. Complementing this, a study in a controlled refrigerator-monitoring context tested multiple open-source OCR engines, integrating on-device preprocessing to improve robustness in suboptimal lighting, and found that tailored preprocessing was essential for capturing reliable expiry date text [8].

Beyond classical preprocessing, deep-learning approaches specifically addressing expiry date recognition are emerging. Florea and Rebedea developed a convolutional neural network (CNN) based model combined with synthetic data to improve recognition of expiry dates on food packaging, reporting a 9.4% accuracy boost over text-only baselines [9].

Emel, Terzioğlu, and Özkan address the variability in invoice structures by introducing a three-stage framework for date extraction comprising custom object detection, OCR, and regular expressions. Leveraging the YOLOv8 model for object detection and PaddleOCR for text recognition, the study presents a robust pipeline that adapts to diverse invoice formats. The authors emphasize the effectiveness of combining modern detection models with traditional parsing techniques to enhance accuracy in document processing [10].

Other research leverages scene-text detection networks (e.g., maximally stable extremal regions (MSER) detectors, TextBoxes++) to isolate date regions before OCR, often combined with deep sequence decoders like CRNN, yielding more robust extraction in cluttered environments [11]. While these studies explore individual OCR engines, preprocessing pipelines, or date detection networks, few perform a controlled comparison across multiple OCR engines *and* preprocessing methods within a unified framework—especially for structured expiry dates in administrative documents. Our work closes this gap by empirically evaluating combinations of SuryaOCR, EasyOCR, and PyTesseract with ten established preprocessing techniques on a real-world annotated dataset, thus providing practical insights for optimizing end-to-end expiry-date extraction workflows.

III. Methodology

This study proposes a comparative framework to identify the optimal combination of OCR engine and image preprocessing technique to extract the latest valid date from scanned document images. The aim is to simulate real-world conditions in administrative and regulatory domains, where expiry dates, such as expiration or renewal deadlines must be accurately identified from heterogeneous document formats.

A dataset of real documents with manually labeled ground-truth expiry dates is processed through multiple OCR pipelines, each involving a different preprocessing technique. The extracted textual content is parsed using carefully designed regular expression patterns to identify and retrieve date entities. The maximum (latest) date extracted from each OCR output is compared to the ground truth to assess accuracy. The evaluation results across combinations enable the selection of the most effective OCR + preprocessing strategy. The general flow is shown in Figure 1.

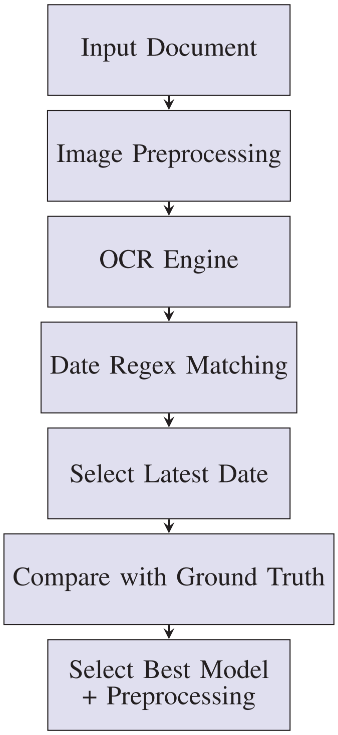


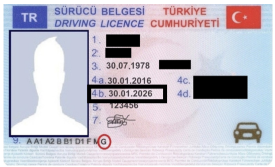
Figure 1. Overview of the processing pipeline: from input documents to model evaluation and selection.

A. Dataset Description

The dataset comprises 1,000 real-world documents in PDF and JPEG formats, sourced exclusively from a single logistics company's internal records. These documents include photos taken with varying lighting, contrast, noise, and distance, screenshots, rotated documents. The seven documents demonstrating the couriers' ability to perform their jobs when they first begin work:

- Vehicle insurance policies
- Driver's licenses (Shown in Fig. 2)
- Identity cards
- Health and liability insurance documents
- Psychotechnical evaluation reports
- Vehicle inspection certificates
- Commercial cargo operating certificate

Each document is manually annotated with its true expiry date (e.g., expiration or renewal date), allowing for quantitative evaluation of OCR-based extraction.



TR SÜRÜCÜ BELGESİ TÜRKİYE
DRIVING LICENCE CUMHURİYETİ

1. [redacted]
2. [redacted]
3. 30.07.1978 [redacted]
4a. 30.01.2016 4c. [redacted]
4b. 30.01.2026 4d.
5. 123456
7.
9. A A1 A2 B B1 D1 F M [✓ G]

Figure 2. One of the sample employment documents

B. OCR Engines

Three open source and free OCR engines were selected to represent a diverse range of architectures and recognition methodologies:

1) SuryaOCR: SuryaOCR [12] is an open-source document layout analysis and OCR framework built on top of deep learning-based detection and recognition pipelines in more than 90 languages. It utilizes transformer-based models for layout-aware text detection and recognition. It consists of the EfficientViT [13] based detection model and the Donut [14] based recognition model. SuryaOCR is particularly well-suited for structured document processing tasks where field localization improves extraction accuracy.

2) EasyOCR: EasyOCR [15] is a deep learning-based OCR engine that supports over 80 languages and is widely used for both scene text and scanned document recognition. The model operates through three main stages: feature extraction, sequence labeling, and decoding. In the first stage, visual features are extracted from the input image using convolutional neural networks, such as Visual Geometry Group (VGG) and residual network (ResNet). These extracted features are then processed through Long Short-Term Memory (LSTM) networks, which are capable of modeling sequential dependencies and capturing contextual information across character sequences. Finally, the output is decoded using the Connectionist Temporal Classification (CTC) algorithm, which is well-suited for sequence-to-sequence tasks with unknown alignments between input and target sequences. The model architecture is implemented on top of the deep-text-recognition-benchmark framework, allowing for robust training across various datasets and enhancing EasyOCR's adaptability to diverse document types and layouts.

3) Pytesseract: Pytesseract is a Python wrapper for the Tesseract OCR engine [16], originally developed by HP and now maintained by Google. Tesseract operates using a classical OCR pipeline with optional LSTM support in newer versions. It supports a wide variety of configurations and preprocessing steps but is more sensitive to noise and skew compared to modern deep learning models.

Each OCR engine was tested independently with every image processing variant described in the following section.

C. Image Preprocessing Techniques

To investigate the influence of image quality and structure on OCR performance, a variety of preprocessing techniques were applied. These transformations aim to enhance features relevant for text recognition while reducing noise and artifacts. Each technique was implemented using either OpenCV or PIL (Pillow), depending on availability.

The preprocessing methods used are:

1) Raw: The original image is passed to the OCR engine without modification.

2) Grayscale: The image is converted to a single-channel grayscale format to remove color variations.

3) Binary Thresholding: Fixed thresholding is applied (default threshold = 150), converting pixels below the threshold to black and others to white. The value of 150 was selected based on preliminary experiments on a small validation subset, where it yielded the highest average OCR character accuracy compared to alternative values (100, 128, 180).

4) Adaptive Thresholding: A Gaussian-based adaptive threshold is used to binarize the image in varying lighting conditions.

5) Contrast Enhancement: Global contrast is increased using a scaling factor (default = 2.0). A factor of 2.0 consistently enhanced text visibility without introducing excessive noise or halo effects around characters, thereby improving OCR readability for most document types in the dataset.

6) Sharpening: Image sharpness is enhanced to accentuate text edges.

7) Noise Reduction: Gaussian blur is applied to reduce background noise.

8) Otsu Thresholding: An automatic threshold value is calculated based on Otsu's method for optimal binarization.

9) Morphological Operations: Operations, such as opening and dilation are applied to remove noise and enhance text connectivity.

10) Deskewing: The image is deskewed by calculating the rotation angle of the largest text contour and rotating accordingly.

Each technique was applied in isolation, yielding multiple versions of each document image. These versions were fed independently into the OCR engines, resulting in a combinatorial evaluation framework. If the date is not extracted after each recognition process, the image is rotated 90, 180 and 270 degrees in order to prevent the documents from being loaded at different angles and then fed back to the model.

D. Date Pattern Matching and Extraction

Following OCR-based text extraction, a rule-based pattern matching module was applied to identify all occurrences of

ISBN: 978-1-68558-303-3

dates within the recognized text. This is because the latest date recorded in the company documents represented the expiry date of the document. To ensure broad coverage across diverse document formats and languages, a comprehensive set of regular expression (regex) patterns was implemented. These patterns were designed to capture both numeric and textual date formats commonly found in official documents.

The date formats targeted include, but are not limited to:

- **Day/Month/Year (with slashes):**
  Examples: `01/01/2023`, `15/12/2021`
  Pattern: `\d{1,2}/\d{1,2}/\d{4}`
- **Day.Month.Year (with dots):**
  Examples: `01.01.2023`, `5.6.2020`
  Pattern: `\d{1,2}\.\d{1,2}\.\d{4}`
- **Year-Month-Day (ISO format):**
  Examples: `2023-06-19`, `2020-01-01`
  Pattern: `\d{4}-\d{2}-\d{2}`
- **Day-Month-Year (with dashes or commas):**
  Examples: `01-01-2023`, `15,12,2021`
  Pattern: `\d{1,2}[-,\.]\d{1,2}[-,\.]\d{4}`
- **Written month names (Turkish or English):**
  Examples: `15 Mart 2022`, `3 July 2021`
  Patterns:
  `\d{1,2} (Ocak|Şubat|...|Aralık) \d{4}`
  `\d{1,2} (January|...|December) \d{4}`

Additionally, expanded patterns were included to accommodate common inconsistencies:

- Omission of leading zeros (e.g., `1/1/2023` vs. `01/01/2023`)
- Use of mixed or nonstandard separators (e.g., `12.03/2023`, `15-02,2021`)
- Interchangeable date orderings (e.g., `YYYY/MM/DD` and `DD/MM/YYYY`)

In total, over 15 regular expression rules were crafted to support a wide variety of layout and formatting inconsistencies often observed in scanned or photographed documents.

After identifying all candidate date strings, each is parsed into a standardized date object. The latest (chronologically maximum) valid date is then selected as the document's predicted expiry date.

This prediction is subsequently compared to the manually annotated ground truth date to assess correctness for each combination of OCR engine and preprocessing method.

E. Evaluation Procedure

The evaluation of system performance was based on the accuracy of the predicted expiry date compared to a manually annotated ground truth. For each document processed through a specific combination of OCR engine and image preprocessing technique, the latest date identified via regular expression matching was interpreted as the predicted expiration date.

To quantify performance, this predicted date was directly compared to the annotated ground truth date. A prediction was considered correct only if the extracted date exactly matched the reference value, including day, month, and year components. No partial matches were accepted in order to maintain a strict and interpretable evaluation criterion. This is important to clearly determine the official expiry date of employment documents. Therefore, performance was measured with a strict accuracy metric focused on exact matching. This binary assessment (correct/incorrect) enabled the computation of accuracy for each OCR-preprocessing pair as the ratio of correctly predicted dates to the total number of documents evaluated.

To ensure robustness, the evaluation was conducted across all documents for each unique (OCR engine + preprocessing method) configuration. The resulting accuracy scores were then analyzed to identify performance patterns and rank the combinations according to their effectiveness in reliable date extraction. In addition to overall accuracy, qualitative observations regarding systematic failure cases—such as common misread characters, formatting inconsistencies, or model-specific artifacts—were documented to support deeper insights into the limitations and sensitivities of each approach.

IV. Results and Discussion

The evaluations and discussions about the results of the study are as follows.

A. Performance Comparison Across OCR Engines

The results demonstrate considerable variation in performance among the three evaluated OCR engines. The output of all model+preprocessing combinations is in Table 1. SuryaOCR yielded the highest overall accuracy, reaching 0.661 when applied to raw document images. EasyOCR followed with a maximum of 0.533, while Pytesseract remained below 0.37 across all configurations.

These differences can be attributed to the architectural choices and modeling paradigms employed by each engine. SuryaOCR is based on recent transformer vision architectures, such as EfficientViT and Donut, both of which are optimized for visually structured documents and capable of capturing long-range spatial dependencies. These models have been shown to perform well even when textual information is embedded in dense layouts or accompanied by background noise.

EasyOCR, in contrast, utilizes a convolutional neural network for visual feature extraction, followed by a recurrent LSTM layer for sequence modeling, and a CTC decoder for transcribing sequences. While this approach is well-suited to natural scene text and moderately structured inputs, it may be more sensitive to the kinds of layout variation and dense text regions observed in certain administrative documents.

Pytesseract, which serves as a Python wrapper for the classical Tesseract OCR engine, does not rely on deep learning-based visual reasoning. Instead, it applies rule-based heuristics and pattern matching for layout detection and character recognition. While effective in clearly scanned and uniformly formatted documents, its limitations become apparent under inconsistent lighting, noise, or structural distortion. Its comparatively low performance across all preprocessing techniques likely stems from this lack of adaptive learning mechanisms.

Overall, the superior performance of SuryaOCR suggests that modern vision transformers provide a robust foundation for text extraction tasks where document formatting is dense but consistent, as in the case of many official forms.

TABLE I
ACCURACY OF OCR MODELS WITH DIFFERENT PREPROCESSING TECHNIQUES

Model + Preprocessing	Accuracy
SuryaOCR	
Raw	**0.661**
Grayscale	0.644
Sharpening	0.619
Denoise	0.606
Contrast	0.558
Deskewing	0.528
Otsu	0.431
Morphological	0.424
Binary	0.424
Adaptive	0.253
EasyOCR	
Raw	0.533
Grayscale	0.518
Sharpening	0.510
Contrast	0.479
Deskewing	0.442
Otsu	0.343
Binary	0.343
Morphological	0.343
Denoise	0.377
Adaptive	0.132
Pytesseract	
Raw	0.365
Grayscale	0.348
Sharpening	0.328
Contrast	0.317
Deskewing	0.312
Denoise	0.307
Otsu	0.283
Binary	0.274
Morphological	0.274
Adaptive	0.066

B. Impact of Preprocessing

The influence of image preprocessing was found to be mixed and often detrimental. For all three OCR engines, raw images consistently resulted in better performance than their preprocessed counterparts. In SuryaOCR, grayscale and sharpening filters yielded only marginal drops in accuracy, while other operations, such as adaptive thresholding, morphological transformations, and deskewing led to substantial degradation in recognition quality.

This trend was even more pronounced in EasyOCR and Pytesseract, where most preprocessing techniques reduced accuracy significantly. Adaptive thresholding, for instance, decreased EasyOCR's performance to as low as 0.132, and even further in Pytesseract. These findings indicate that for structured and high-quality printed documents, preprocessing may distort textual features rather than enhance them, potentially disrupting the models' learned representations or rule-based heuristics.

It is likely that the preprocessing operations, particularly those designed for low-quality or noisy inputs, interfere with the sharp edges and uniform backgrounds typically found in scanned or photographed administrative forms. Therefore, the assumption that preprocessing universally improves OCR quality does not hold in this context, where the majority of documents are already visually clean and consistently formatted.

C. Error Analysis

A manual inspection of incorrect predictions revealed several common sources of error. In some cases, the OCR engine partially recognized the date field, extracting only the day and month, or misinterpreted the format due to spacing or font irregularities. In others, the algorithm erroneously selected unrelated date fields, such as print dates or issuance dates, rather than the intended expiry date. This issue was particularly evident in documents containing multiple date fields in similar visual prominence.

Additionally, the dataset consisted of 1000 randomly selected document images, which were acquired through a mix of scanning and mobile photography. Consequently, some samples exhibited poor visual quality due to factors, such as motion blur, shadow artifacts, low resolution, or uneven lighting. In several instances, the actual expiry date was illegible to both the OCR engine and human annotators, especially in heavily degraded or partially occluded regions of the document. These visual defects likely contributed to both false positives and false negatives.

Another noteworthy factor is the diversity of document types included in the dataset, such as identity cards, driver's licenses, insurance forms, and inspection reports. Each category possesses distinct structural patterns, font styles, and positional layouts of the expiry date. This heterogeneity may have introduced further complexity to the task, particularly for OCR engines that lack document-type-specific tuning. For example, dates on vehicle inspection reports are typically handwritten or stamped, whereas those on ID cards are printed in machine-readable zones, requiring different recognition strategies.

Together, these sources of error highlight the need for document-aware post-processing techniques, including context-driven date selection, region-of-interest filtering, or the integration of layout prediction models to disambiguate visually similar fields.

D. Limitations and Observations

The evaluation was conducted on a diverse and realistic corpus of administrative documents. However, several limitations must be acknowledged. Firstly, the ground truth for each document included only a single expiry date, whereas documents often contain multiple dates, any of which could be visually or semantically plausible. This binary evaluation framework may have underestimated the practical utility of certain OCR outputs that extracted valid but unintended date fields.

Secondly, while various preprocessing techniques were systematically tested, the choice of parameters, such as kernel sizes or contrast factors, was kept fixed across documents. A more adaptive preprocessing pipeline might yield improved results, particularly if guided by document layout classification or confidence estimation.

Finally, the evaluation focused solely on matching extracted date strings with ground truth labels, without incorporating additional semantic validation or natural language context. Future extensions could explore hybrid models that combine

OCR with named entity recognition or multimodal transformers trained to reason over both visual and textual cues.

These limitations notwithstanding, the results provide useful insights into the practical performance boundaries of common OCR frameworks and preprocessing strategies when applied to real-world administrative records.

V. Conclusion and Future Work

This study investigated the effectiveness of various OCR engines and image preprocessing techniques in extracting the expiry date from structured administrative documents, such as insurance certificates, identification cards, vehicle inspection reports, and similar official records. The proposed system demonstrated a significant automation potential, achieving up to 66.1

Among the evaluated models, transformer-based SuryaOCR exhibited superior performance, particularly when used on raw document images. Contrary to common expectations, most preprocessing operations, such as thresholding, noise reduction, and morphological filtering led to diminished accuracy. These findings suggest that for visually clean and structured documents, aggressive preprocessing may disrupt rather than enhance text recognizability. This insight can inform future system designs by reducing unnecessary computational overhead related to image enhancement stages.

Despite promising results, several avenues for improvement remain. First, model accuracy may be further increased through task-specific fine-tuning of OCR engines using domain-relevant training data. Especially in cases where expiry dates follow predictable patterns in fixed regions of the document, integrating visual layout modeling or region-based text filtering could improve both precision and recall. Moreover, hybrid approaches that combine rule-based post-processing with semantic filtering (e.g., recognizing keywords, such as “Valid Until”) may aid in disambiguating among multiple date fields.

Handling exceptional cases also remains a critical consideration. In particular, documents that are uploaded incorrectly—either by being outside the supported set of document types or by omitting the required expiry date field—can lead to failed or misleading outputs. To address this, a document classification step could be introduced prior to OCR processing to ensure compatibility. Furthermore, if no valid date is detected, the system can be configured to trigger a fallback workflow, such as alerting human reviewers or requesting resubmission. Confidence scoring and anomaly detection methods may also be leveraged to flag uncertain predictions, reducing the risk of silent failure in high-stakes applications.

Due to the scope of this study, the types of documents used were limited, and the validity dates could be identified using regular expression patterns. In future work, we plan to expand the dataset to include a broader range of document types and languages, and to evaluate the inclusion of layout-aware deep learning models. Furthermore, we aim to enhance the system’s intelligence by enabling it to recognize the specific type of each document. Additionally, integrating Large Language Models (LLMs) or vision transformer models for semantic validation and context-based extraction of date fields represents a promising direction for improving robustness in real-world deployments.

References

[1] A. Alaei, V. Bui, D. Doermann, and U. Pal, “Document image quality assessment: A survey”, *ACM computing surveys*, vol. 56, no. 2, pp. 1–36, 2023.

[2] J. Björkman, *Evaluation of the effects of different preprocessing methods on ocr results from images with varying quality*, 2019.

[3] R. L. Kshetry, “Image preprocessing and modified adaptive thresholding for improving ocr”, *arXiv preprint arXiv:2111.14075*, 2021.

[4] A. El Harraj and N. Raissouni, “Ocr accuracy improvement on document images through a novel pre-processing approach”, *arXiv preprint arXiv:1509.03456*, 2015.

[5] M. Dias and C. T. Lopes, “Optimization of image processing algorithms for character recognition in cultural typewritten documents”, *arXiv preprint arXiv:2311.15740*, 2023.

[6] S. Kavin and C. Shirley, “Ocr-based extraction of expiry dates and batch numbers in medicine packaging for error-free data entry”, in *2024 7th International Conference on Circuit Power and Computing Technologies (ICCPCT)*, IEEE, vol. 1, 2024, pp. 278–283.

[7] R. A. Tavares, “Comparison of image preprocessing techniques for vehicle license plate recognition using ocr: Performance and accuracy evaluation”, *arXiv preprint arXiv:2410.13622*, 2024.

[8] K. Hosozawa, R. H. Wijaya, T. D. Linh, H. Seya, M. Arai, T. Maekawa, and K. Mizutani, “Recognition of expiration dates written on food packages with open source ocr”, *International Journal of Computer Theory and Engineering*, vol. 10, no. 5, pp. 170–174, 2018.

[9] V. Florea and T. Rebedea, “Expiry date recognition using deep neural networks”, *International Journal of User-System Interaction*, vol. 13, no. 1, pp. 1–17, 2020.

[10] M. H. Emel, M. Terzioğlu, and R. Özkan, “Efficient and accurate date extraction from invoices: A comprehensive three-step methodology integrating custom object detection, ocr, and refined regular expressions”, *Advances in Artificial Intelligence Research*, vol. 4, no. 1, pp. 10–17, 2024.

[11] L. Gong, M. Yu, W. Duan, X. Ye, K. Gudmundsson, and M. Swainson, “A novel camera based approach for automatic expiry date detection and recognition on food packages”, in *IFIP international conference on artificial intelligence applications and innovations*, Springer, 2018, pp. 133–142.

[12] V. Paruchuri and D. Team, “Surya: A lightweight document ocr and analysis toolkit”, GitHub repository, 2025, [Online]. Available: https://github.com/VikParuchuri/surya (visited on 08/18/2025).

[13] H. Cai, J. Li, M. Hu, C. Gan, and S. Han, “Efficientvit: Multi-scale linear attention for high-resolution dense prediction”, *arXiv preprint arXiv:2205.14756*, 2022.

[14] G. Kim, T. Hong, M. Yim, J. Nam, J. Park, J. Yim, W. Hwang, S. Yun, D. Han, and S. Park, “Ocr-free document understanding transformer”, in *European Conference on Computer Vision*, Springer, 2022, pp. 498–517.

[15] JaidedAI, “Easyocr”, GitHub repository, 2021, [Online]. Available: https://github.com/JaidedAI/EasyOCR (visited on 08/18/2025).

[16] R. Smith, “An overview of the tesseract ocr engine”, in *ICDAR ’07: Proceedings of the Ninth International Conference on Document Analysis and Recognition*, Washington, DC, USA: IEEE Computer Society, 2007, pp. 629–633, ISBN: 0-7695-2822-8.