Journal of Electronics, Electromedical Engineering, and Medical Informatics
Homepage: jeeemi.org; Vol. 8, No. 4, October 2026, pp: 1769-1795 e-ISSN: 2656-8632

RESEARCH ARTICLE OPEN ACCESS

A Graph-Assisted Post-OCR Framework with Damerau-Levenshtein Spelling Correction for Drug Name Recognition from Pharmaceutical Packaging

Fito Satrio[1], Muhammad Raafi'u Firmansyah[2], and Diah Septiani[1]

[1] Data Science Study Program, Telkom University, Purwokerto Campus, Indonesia
[2] Informatics Engineering Study Program, Telkom University, Purwokerto Campus, Indonesia

Corresponding author: Muhammad Raafi'u Firmansyah. (e-mail: raafiu@telkomuniversity.ac.id), **Author(s) Email**: Fito Satrio (e-mail: fitoosatrio@gmail.com), Diah Septiani (e-mail: diahseptiani@telkomuniversity.ac.id)

Abstract **Medication errors remain a persistent threat to patient safety, with a global prevalence of approximately 3%, rising to 11% in elderly care and 7% in intensive care units. In Indonesia, the large number of BPOM-registered pharmaceutical products increases the risk of drug misidentification from look-alike, sound-alike (LASA) names. At the same time, conventional OCR pipelines remain limited by heterogeneous layouts, font variation, and brand–generic name ambiguity. This study develops an integrated, lightweight post-OCR framework resolving both character-level OCR errors and brand–generic naming ambiguity without costly large language models. The main contributions are a Damerau-Levenshtein Spelling Correction (DLSC) module correcting character-level errors against a validated drug dictionary, and a Graph Traversal Correction (GTC) mechanism modeling brand–generic relationships as a directed graph for disambiguation. The framework was evaluated on a self-collected, web-scraped dataset of 9,500 Indonesian drug packaging images spanning 3,678 brand labels, split 80:20 at the brand-label level; three pretrained OCR engines (Tesseract, EasyOCR, PaddleOCR) were benchmarked without domain-specific fine-tuning under both CPU and GPU configurations, with the training subset serving only as a consistency check rather than fitting data, since DLSC and GTC are rule-based and undergo no parameter training. PaddleOCR + DLSC + GTC achieved the best performance, raising accuracy from 50.94% (Raw OCR) to 96.13% on training and 96.69% on testing data, with WER reduced to 3.82%/3.44% and CER to 3.64%/3.76%. Paired McNemar statistical testing confirmed GTC's improvement over DLSC is significant across all three engines, and baseline comparison confirmed this gain stems specifically from explicit transposition modeling rather than generic dictionary correction. GTC added only 0.05–0.06 ms beyond DLSC regardless of CPU or GPU configuration. Findings indicate that combining lightweight edit-distance correction with graph-based disambiguation substantially improves drug-name recognition accuracy under resource-constrained, pretrained-model deployment conditions.**

Keywords **Post-OCR Correction; Damerau-Levenshtein; Graph Traversal; Drug Name Recognition**

I. Introduction

Patient safety remains a central priority in modern healthcare delivery, and medication errors (MEs) continue to represent one of the most persistent threats to this goal, with a global prevalence of about 3%, rising to 11% in elderly care and 7% in intensive care units, and imposing an economic burden estimated at billions of dollars annually [1], [2]. Among the contributing factors, drug misidentification remains a critical and recurring failure point, as an incorrectly identified drug can directly translate into a wrong-drug or wrong-label error [1], [2], underscoring the importance of accurate drug-name recognition as a foundational safeguard against medication errors.

This problem is equally pressing in Indonesia, where BPOM has authorized thousands of pharmaceutical products, from generic drugs and branded products to biological preparations [3], compounded by the persistent challenge of look-alike, sound-alike (LASA) medication names and packaging. A retrospective study at a major public hospital in South Borneo documented 338 medication errors out of 251,491 prescriptions over one year, with wrong-label and wrong-drug preparation accounting for nearly one-third of cases, and noted error rates elsewhere in Indonesian hospitals reaching as high as 20.4% [4]; a similar pattern was reported in Thai hospitals, where

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

confusion from similar drug names accounted for the majority of LASA-related errors [5].

Automating drug-label reading using optical character recognition (OCR) can reduce reliance on manual visual identification, but two distinct failure modes limit its reliability for pharmaceutical packaging. The first is a recognition problem: heterogeneous layouts, diverse fonts, geometric distortion, and inconsistent illumination degrade raw OCR output, producing character-level substitution, insertion, deletion, and transposition errors; Previous pharmacy-related OCR studies have reported an accuracy of 83.78% [6], while an end-to-end drug-label recognition framework reported a CER of 21.6% and a WER of 23.7% [7], with information-retrieval accuracy dropping as low as 55.6% for specific fields such as expiry dates [8]. Post-OCR correction methods, including deep-learning-based approaches [9] and similarity-search frameworks [10], substantially reduce this error class, while dictionary-based edit-distance methods such as Damerau-Levenshtein offer a lighter-weight alternative that explicitly handles all four canonical error types without requiring model training [11].

The second is a disambiguation problem, largely independent of recognition accuracy: brand and generic names frequently co-occur on a single package, and even when both are correctly extracted, a system must still determine which one is the product's registered label. Prior work addressing recognition [9], [10] does not model this relational ambiguity, while prior work addressing brand–generic disambiguation, through dual-engine OCR fusion [12] or LLM-based classification [13], does not incorporate lightweight character-level correction. Other studies on cylindrically distorted packaging [14], web-based drug information retrieval [15], and morphology-based brand identification [16] likewise leave this joint problem unaddressed.

A graph-based traversal approach offers a promising solution to this unresolved problem, systematically traversing connections between a detected candidate and its pharmacologically equivalent counterparts for more context-aware disambiguation than isolated string-matching. Unlike dictionary lookup, which resolves each token independently, the proposed Graph Traversal Correction (GTC) exploits relations between co-occurring labels to arbitrate competing brand–generic interpretations; unlike embedding-based knowledge-graph linking [17], it performs a training-free one-hop traversal rather than learning entity embeddings; and unlike existing rule-based brand–generic matching, it enforces token-distinctness and position-aware verification rather than accepting a co-occurring counterpart on similarity alone. The principal methodological advancement is thus not Damerau-Levenshtein Spelling Correction (DLSC) or graph traversal individually, but their integration as a cascade where GTC resolves DLSC's residual brand–generic errors under a length-adaptive threshold. To the best of our knowledge, limited studies have explored this specific integration of character-level OCR correction with graph-based relational disambiguation for brand–generic drug name ambiguity.

Despite these advances, limited attention has been given to jointly addressing the recognition and disambiguation problems within a single, lightweight framework, motivating two guiding research questions: whether dictionary-based edit-distance correction and graph-based relational disambiguation can be integrated into a single cascade outperforming either stage alone, and whether this can be achieved without the computational overhead of large language models or learned graph embeddings, remaining suitable for resource-constrained healthcare deployment. This gap is particularly relevant to drug-name recognition using an Indonesian pharmaceutical reference database such as data derived from BPOM.

To address these questions, this study proposes a graph-assisted post-OCR framework for drug name recognition from pharmaceutical packaging, integrating OCR-based text extraction, Damerau-Levenshtein Spelling Correction (DLSC) for character-level recognition errors, and Graph Traversal Correction (GTC) for brand–generic disambiguation. The main contributions of this study are summarized as follows:

1. Implementation of a Damerau-Levenshtein Spelling Correction module that corrects character-level OCR errors by matching extracted text against a validated drug-name reference database.
2. Design of a graph-based traversal mechanism that models brand–generic relationships as an interconnected graph, resolving ambiguity and correctly identifying the product's registered drug label when both names co-occur on a single package.
3. Measurement and comparison of the performance achieved by integrating OCR, Damerau-Levenshtein Spelling Correction, and graph traversal against baseline OCR without additional post-processing.
4. Comparative benchmarking of three OCR engines under a unified preprocessing pipeline to identify the most suitable baseline engine for pharmaceutical packaging recognition.

This paper is structured as follows: Section II reviews related work on OCR-based drug-label recognition, post-OCR correction, and graph-based entity disambiguation. Section III describes the proposed method. Section IV presents the experimental results. Section V discusses the findings.

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

Table 1. Comparative analysis of existing OCR-based drug-label recognition methods, highlighting post-OCR correction techniques, brand–generic name disambiguation, and methodological limitations.

Ref	Study	Post-OCR Correction	Brand–Generic Handling	Main Limitation
[6]	OCR + CNN	None	Not addressed	No character-error correction
[7]	YOLOv8 + CRNN-gMLP	None	Not addressed	High detection precision but no correction stage
[9]	Generic OCR + RoBERTa	Deep-learning language model	Not addressed	High computational cost; not domain-specific to drug packaging
[10]	CRNN (VietOCR) + Aho–Corasick	Similarity-based, not edit-distance	Not addressed	Does not explicitly model the four canonical OCR error types
[12]	Dual-engine OCR fusion	Rule-based parsing and deep learning correction	Not addressed	Does not resolve brand–generic ambiguity; heavier computation.
This Study	Single-engine OCR (benchmarked independently)	Damerau-Levenshtein (DLSC)	Brand–generic relational graph (GTC)	High accuracy, low error and computation

Section VI concludes the paper and outlines future work.

II. Related Works

Research on automated drug-label identification has progressed along several complementary directions, ranging from off-the-shelf OCR engines applied directly to pharmaceutical packaging [6], [7] to hybrid pipelines combining deep learning detection with rule-based correction [8], [14], with the shared objective of improving drug-name recognition reliability in real-world healthcare settings.

Several studies have benchmarked general-purpose OCR engines directly on packaging-style text to assess their baseline suitability for pharmaceutical applications. An evaluation of Tesseract, EasyOCR, PaddleOCR, and TrOCR on food-packaging labels showed Tesseract achieving the lowest Character Error Rate, while EasyOCR offered the best accuracy–multilingual balance under uncontrolled retail imaging conditions [18]. Confidence-score comparisons further indicate PaddleOCR consistently achieves the highest recognition confidence at approximately 0.93, compared to 0.89 for Tesseract and 0.85 for EasyOCR [19]. However, engine-ablation studies applied to visual text-extraction tasks note that PaddleOCR's higher precision often comes at the cost of lower recall and slower inference relative to Tesseract, which instead offers the fastest processing time among the three [20]. These engines also differ substantially in underlying architecture, reflecting the distinct design philosophies and continued technical evolution of each framework [21], [22]. Despite this progress, none of these general-purpose engines were designed specifically for pharmaceutical packaging, where dense multilingual text, embossed surfaces, and cylindrical distortion remain common failure modes.

Beyond engine-level benchmarking, error correction research targeting edit-distance methods has continued to mature in recent years. A Damerau-Levenshtein spell-checker combined with an N-gram model demonstrated strong correction performance for a low-resource North African language, confirming the algorithm's continued relevance for morphologically rich, non-English text [23]. Applied to Persian medical records, a Damerau-Levenshtein-based correction system substantially improved downstream text quality by explicitly modeling all four canonical OCR error types [24]. Closer to the Indonesian context, Damerau-Levenshtein matching refined with cosine similarity for candidate ranking reported measurable accuracy improvement over the unmodified metric while

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

maintaining low inference latency suitable for dictionary-scale lookup [25]. Combining Damerau-Levenshtein distance with an N-gram representation and a Random Forest classifier further improved robustness against noisy input compared to edit distance alone [26]. Collectively, these studies indicate that edit-distance-based correction remains an active and effective research direction, particularly for domain-specific, morphologically complex text where LLM-based correction is computationally prohibitive.

Parallel to these correction-focused efforts, graph-based representations have increasingly been applied to resolve entity-level ambiguity in specialized domains. A knowledge-graph entity-disambiguation method combining entity-relationship embeddings with graph-structure embeddings demonstrated that incorporating topological connections surrounding a candidate entity substantially improves disambiguation precision over relying on contextual similarity alone [17]. In the healthcare domain, a comprehensive review of knowledge-graph applications highlighted their growing role in structuring unstructured clinical information, noting that graph-based entity linking consistently outperforms isolated lookup approaches when candidate entities share overlapping identifiers or synonymous naming [27]. A knowledge-graph framework supporting pharmacists' clinical information needs further demonstrated the feasibility of representing product-equivalence relationships between pharmaceutical entities as an explicitly queryable graph structure, directly addressing naming inconsistencies across brand and generic product labeling [28]. Broader reviews of drug-related knowledge graphs have similarly emphasized their utility for relational inference tasks such as therapeutic substitution and drug repurposing [29]. However, these knowledge-graph studies do not formalize a lightweight, training-free traversal mechanism downstream of edit-distance correction with explicit token-distinctness and position constraints, the specific gap this study's GTC design addresses.

Based on Table 1, previous studies fall into two categories: those relying solely on raw OCR output without correction [6], [7] and those applying post-processing without addressing brand–generic disambiguation [9], [10], [12]. The first suffers from uncorrected OCR errors, limiting accuracy for healthcare applications, while the second improves recognition through deep-learning-based correction, similarity search, or rule-based parsing, but remains computationally expensive or fails to explicitly handle the full range of canonical OCR error types. The dual-engine fusion approach in [12] combines two OCR engines to improve character-level recognition but does not resolve semantic relationships between co-occurring brand and generic names. Overall, none of the reviewed studies integrate lightweight OCR correction with relational disambiguation in a single

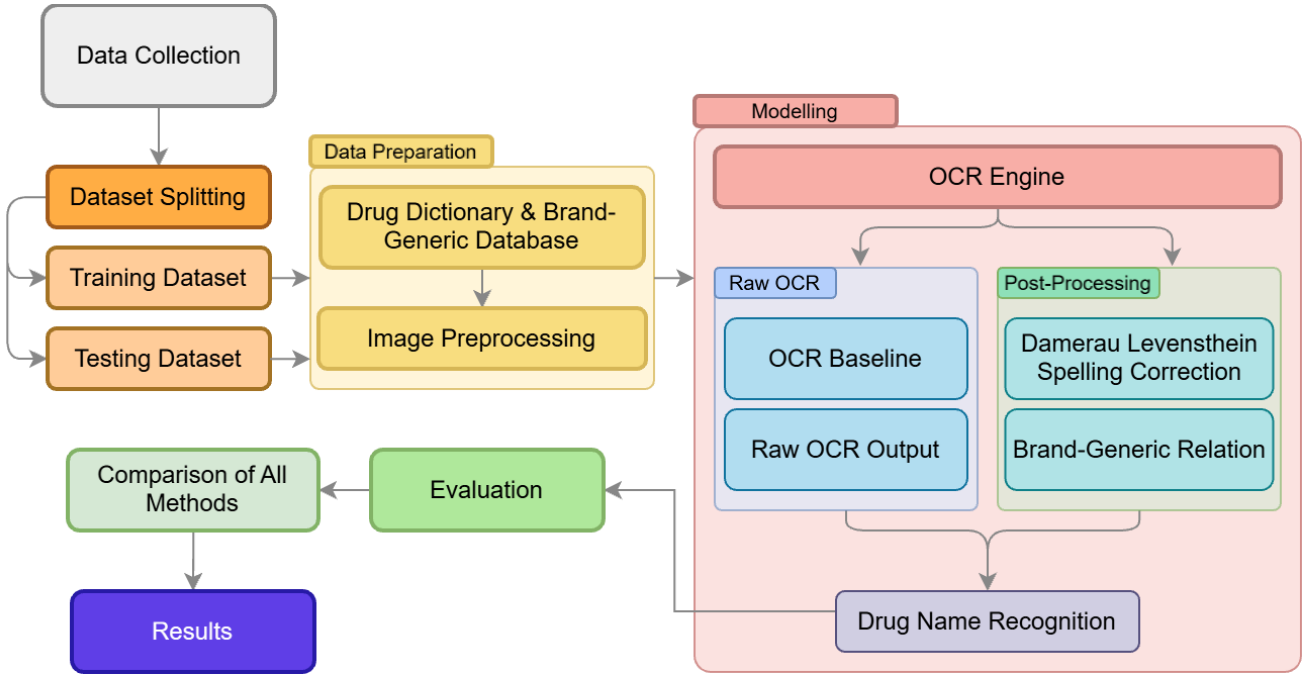

Fig. 1. Overall workflow of the proposed graph-assisted post-OCR drug-name recognition framework, illustrating data collection, dataset preparation, image preprocessing, OCR extraction, DLSC, GTC, and performance evaluation.

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

framework, motivating the proposed method, which combines Damerau-Levenshtein spelling correction with graph-traversal-based disambiguation for efficient and accurate drug name recognition.

III. Method

This study proposes a post-OCR processing framework integrating three components in cascade: text extraction via Optical Character Recognition (OCR), character-level error correction using the Damerau-Levenshtein distance algorithm, and brand–generic drug name disambiguation through graph traversal. The output of each stage feeds into the next, allowing errors introduced by the OCR engine to be progressively reduced until an accurate drug-name identification is obtained at the final stage.

The overall workflow of the proposed framework is illustrated in Fig. 1. The research process begins with Data Collection, followed by Data Preparation, which includes dataset splitting into training and testing subsets, construction of a drug dictionary and a brand–generic relational database, and image preprocessing. The prepared data then enters the modeling stage, comprising a Raw OCR sub-process, where the OCR engine generates baseline text extraction, and a post-processing sub-process, where the raw OCR output is corrected through Damerau-Levenshtein spelling correction and resolved through the brand–generic relational graph. The framework concludes with an evaluation stage in which the performance of all tested configurations is compared. This staged pipeline aligns with recent trends in post-OCR correction research, showing that combining a lightweight correction mechanism with an explicit relational representation can improve entity-recognition accuracy without costly large language models [9], [23], [17].

A. Data Collection

The dataset consists of self-collected images of Indonesian drug packaging. A custom dataset was necessary because existing public OCR benchmarks do not adequately capture pharmaceutical-specific characteristics such as mixed Indonesian–international terminology, varied fonts, and co-occurring brand–generic names. Since no large-scale public dataset for Indonesian drug packaging exists, images were collected via Google Images search using drug-name keywords (A–Z) sourced from the BPOM registration database and Alodokter, following practices used for other Indonesia-specific datasets such as IndoHerb [30] and addressing the Western-market bias noted in prior pharmaceutical dataset studies [31].

Collected images span multiple packaging types (bottles, blister strips, boxes) with mixed Indonesian and English/Latin text, ranging from 119 × 75 to 11,174 × 4,887 pixels (mean 760 × 699, median 650 × 550). Images were included if they showed legible packaging text and excluded if duplicate or non-packaging, with duplicates removed manually. As the images originate from web search rather than field photography, the sample likely skews toward products with higher online visibility and toward favorable studio-style imaging conditions rather than the degraded lighting and handling typical of real dispensing settings. Ground-truth labels were manually annotated from each search query and cross-checked against BPOM and Alodokter [31], [32], serving jointly as the study's quality-control procedure. This yielded 9,500 validated images spanning 3,678 brand labels, larger than most prior pharmaceutical image datasets [31], [32].

Table 2. Distribution of the Indonesian pharmaceutical packaging image dataset across training and testing subsets, including image counts and unique brand labels.

Data Subset	Number of Images	Number of Brand Labels
Training Set (80%)	7,600	2,926
Testing Set (20%)	1,900	752
Total	**9,500**	**3,678**

The dataset was split 80:20 at the brand-label level, as shown in Table 2, so all images of a given brand belong to exactly one subset. The drug dictionary and brand–generic graph were partitioned accordingly: images from the training subset were matched only against the dictionary and graph built from the 2,926 training brand labels, and images from the testing subset were matched only against those built from the 752 testing brand labels. No brand label therefore appears in both the training and testing reference resources. The terms "training" and "testing" describe only this predefined data partition, used respectively for consistency analysis and final evaluation; no parameter estimation or model fitting is performed on either subset. Consequently, testing evaluates recognition of previously unseen packaging images, all belonging to labels not represented in the training set.

B. Image Preprocessing

Referring to the Image Preprocessing stage in Fig. 1, each collected image undergoes a sequence of enhancement steps prior to text extraction, aimed at mitigating common degradations found in drug packaging photographs such as non-uniform illumination, geometric skew, motion blur, and inconsistent background–text contrast. Four preprocessing techniques are applied in sequence: Contrast Limited Adaptive Histogram Equalization (CLAHE), skew correction, image deblurring, and image binarization.

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

1. Contrast Enhancement (CLAHE)

CLAHE is applied selectively to images identified as having insufficient contrast, based on statistical measures of mean brightness and intensity standard deviation computed from the grayscale representation, thereby avoiding unnecessary noise amplification on already well-lit images, a limitation widely reported for uniform histogram equalization approaches [33]. For each image requiring enhancement, the image is partitioned into 8×8 contextual tiles, and the local histogram of each tile is redistributed according to the transformation function as defined in Eq. (1) [33]:

$$T(r_k) = (L-1) \times \sum_{j=0}^{k} p_r\left(r_j\right) \quad (1)$$

where L is the number of grayscale levels and $pr(rj)$ is the probability of intensity j within the local histogram. To prevent over-enhancement, each histogram is clipped at a limit as defined in Eq. (2) [34]:

$$C_{max} = \frac{\beta \times (m \times n)}{L} \quad (2)$$

with β = 2.0 and $m \times n$ denoting the tile size, following adaptive clip-limit configurations shown to balance contrast gain and noise suppression in recent CLAHE-based enhancement studies [34].

2. Skew Correction

Packaging images captured at non-perpendicular angles are corrected using a Canny edge map followed by Hough Transform-based line detection, expressed in polar coordinates as Eq. (3) [35]:

$$\rho = x\cos\theta + y\sin\theta \quad (3)$$

The dominant orientation angle $\theta skew$ is estimated from the accumulation peak corresponding to the most frequent horizontal line orientation, an approach demonstrated to be effective and computationally efficient for correcting skew in scanned and photographed documents prior to OCR [35]. The affine rotation matrix used to rotate the image is defined in Eq. (4) [35]:

$$R = \begin{bmatrix} \cos(-\theta_{skew}) & -\sin(-\theta_{skew}) \\ \sin(-\theta_{skew}) & \cos(-\theta_{skew}) \end{bmatrix} \quad (4)$$

with bilinear interpolation applied to preserve pixel quality during rotation.

3. Image Deblurring

Blurred images, primarily caused by camera motion during data collection, are restored using Wiener filtering in the frequency domain. The degradation model is defined as Eq. (5) [36]:

$$g(x,y) = f(x,y) * h(x,y) + n(x,y) \quad (5)$$

where g is the observed blurred image, f is the latent sharp image, h is the point spread function (PSF), and n is additive noise. The corresponding Wiener restoration filter is given by Eq. (6) [36]:

$$\hat{H}(u,v) = \left[\frac{H^*(u,v)}{|H(u,v)|^2 + K}\right] \times G(u,v) \quad (6)$$

where K denotes the noise-to-signal ratio. This formulation is consistent with recent Wiener-based deconvolution frameworks shown to effectively suppress ringing artifacts while restoring edge sharpness under motion-blur conditions [36].

4. Image Binarization

The final preprocessing step converts each enhanced grayscale image into a binary representation to maximize text-background separability prior to OCR, a step widely recognized as a critical determinant of downstream OCR accuracy across both global and adaptive thresholding approaches. Global thresholding via Otsu's method is applied first, maximizing the between-class variance as in Eq. (7) [37]:

$$\sigma_b^2(t) = \omega_0(t) \times \omega_1(t) \times [\mu_0(t) - \mu_1(t)]^2 \quad (7)$$

with the optimal threshold selected as in Eq. (8) [37]:

$$t^* = \underset{0 \le t \le L-1}{\arg\max}\, \sigma_b^2(t) \quad (8)$$

For images with non-uniform illumination, where global thresholding is insufficient, the adaptive Sauvola method is applied instead, computing a local threshold based on the mean and standard deviation within a 25×25 pixel neighborhood as defined in Eq. (9) [37]:

$$T(x,y) = \mu(x,y) \times \left[1 + k \times \left(\frac{\sigma(x,y)}{R} - 1\right)\right] \quad (9)$$

where k = 0.3 and R = 128. The Sauvola approach has been shown to outperform global thresholding on documents with complex, non-uniform backgrounds, consistent with the packaging designs encountered in this study's dataset.



Fig. 2. Representative pharmaceutical packaging images before and after preprocessing, illustrating the enhancement of text visibility through the proposed image preparation pipeline.

The four-stage preprocessing pipeline progressively enhances drug packaging image quality, as shown in Fig. 2. Each stage improves text visibility and strengthens the separation between textual content

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

and background, producing clearer character boundaries. In the final preprocessed image, the printed brand name, generic composition, and dosage information appear as well-defined foreground elements against a more uniform background, with reduced reflections and uneven illumination enabling more precise character delineation. These cleaner visual features facilitate more accurate text recognition during the subsequent OCR stage.

C. Optical Character Recognition

The general OCR pipeline followed in this study is illustrated in Fig. 3, which depicts the conventional four-stage process of document capture, image file creation, OCR processing, and generation of editable text output. This conceptual pipeline forms the basis for the Raw OCR stage shown earlier in Fig. 1, in which the preprocessed drug packaging images serve as the

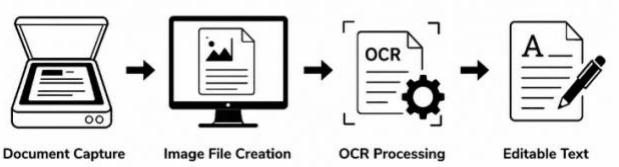

Fig. 3. General optical character recognition workflow comprising document acquisition, image generation, OCR processing, and text extraction.

features processed by a sequence-modelling encoder to capture contextual dependencies between characters within a text line [39]. The encoder output is then passed to a decoder consisting of a fully connected layer and a softmax activation, producing a per-timestep probability distribution over the character vocabulary, which is subsequently decoded into the

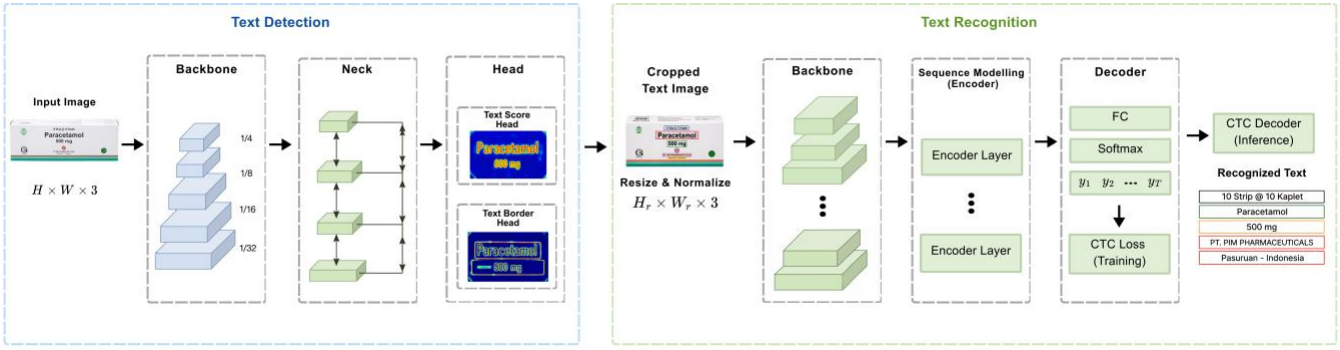

Fig. 4. Architecture of a generic two-stage OCR system illustrating text-region detection, feature extraction, sequence modeling, and CTC-based text recognition.

document capture input, and the corresponding raw text output is subsequently passed to the post-processing stage.To capture the internal mechanism underlying modern OCR systems, Fig. 4 presents the generic two-stage paradigm of Text Detection followed by Text Recognition, which underlies deep-learning-based OCR architectures. In the detection stage, the input image of dimension $H \times W \times 3$ is passed through a backbone network to extract multi-scale feature maps, aggregated by a neck module and passed to detection heads that produce a text score map and border map, identifying candidate text regions such as the brand name and dosage information on the packaging. Segmentation-based detection architectures, such as the Differentiable Binarization (DB) module adopted by PaddleOCR, have been shown to maintain high accuracy and inference speed on text of arbitrary shape and orientation, particularly relevant for the curved and embossed text frequently found on cylindrical pharmaceutical packaging [38].

Once candidate regions are detected, each is cropped, resized, and normalized to a fixed height $Hr \times Wr \times 3$ before being passed into the recognition stage, where a backbone extracts sequential visual final character sequence using a Connectionist Temporal Classification (CTC) decoder [40].

$$\text{Conf} = \frac{1}{T}\sum_{t=1}^{T} \max_{c} p(c \mid t) \quad (10)$$

For each recognized text region, an overall confidence score is computed as the average of the maximum per-timestep character probabilities produced by the decoder as defined in Eq. (10) [40]:

where T is the length of the recognized character sequence and $p(c \mid t)$ is the probability assigned to character c at timestep t. This confidence score reflects the decoder's certainty in the recognized character sequence and is later used to support filtering of low-confidence OCR predictions.

The three OCR engines benchmarked in this study instantiate this generic paradigm to varying degrees. PaddleOCR most closely follows the full architecture depicted in Fig. 4, employing a DB-based detector to localize text regions, followed by an SVTR-based recognizer with a CTC decoding head, and additionally incorporating a text-orientation classification module to handle rotated or cylindrically distorted labels.

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

EasyOCR follows the same two-stage structure but with different constituent modules, using a CRAFT-based detector for text-region localization and an attention-augmented CRNN recognizer for sequence decoding. Tesseract OCR, in contrast, does not perform an explicit, separately trained detection stage; instead, it relies on a page-layout analysis step configured in PSM 11 mode to treat each packaging label as a single uniform text block, which is then passed directly to an LSTM-based recognition backbone.

The relative ranking among these three engines varies with dataset composition, font size, and imaging conditions [18], [41]. Since these prior findings originate from packaging domains distinct from the Indonesian drug packaging dataset used in this study, all three engines are evaluated independently under an identical preprocessing pipeline, rather than selecting a single engine based on findings from a different domain. Each engine produces raw text output, bounding-box coordinates, and confidence scores that serve as the baseline input for the Damerau-Levenshtein spelling correction stage.

D. Baseline Post-OCR Correction Methods

To assess whether the reported performance gains originate specifically from Damerau-Levenshtein correction and graph-based disambiguation, rather than from dictionary-based correction in general, four widely used post-OCR correction baselines were implemented for comparison. All baselines are matched against the same drug reference database constructed during data collection, ensuring that any performance difference reflects the correction method itself rather than dictionary coverage. (1) Standard Levenshtein Matching**.** This baseline computes edit distance using only insertion, deletion, and substitution, omitting the transposition term in Eq. (17). The distance is computed following the same recursive formulation as Eqs. (13)-(16), with the transposition case removed. (2) Fuzzy Matching. This baseline applies RapidFuzz's token-based ratio scoring, which computes similarity from the length of the longest common alignment between two strings normalized by their combined length, without explicitly modeling individual character-level operations as DLSC does. (3) N-gram Retrieval. Each token and dictionary entry are represented as a set of character trigrams (n = 3). Candidates are ranked by the Jaccard similarity between token and reference trigram sets, as defined in Eq. (11) [26]:

$$\text{Sim_ngram}(S,T) = \frac{|G_s \cap G_t|}{|G_s \cup G_t|} \tag{11}$$

where G_s and G_t denote the trigram sets of the OCR-extracted token and candidate reference string, respectively. (4) Cosine Similarity Matching. Each token and dictionary entry is represented as a character-frequency vector, and candidates are ranked by cosine similarity between the OCR token vector and each reference vector, as defined in Eq. (12) [25]:

$$\text{Sim_cos}(S,T) = \frac{v_s \cdot v_t}{\|v_s\|\|v_t\|} \tag{12}$$

where v_s and v_t denote the character-frequency vectors of the OCR-extracted token and candidate reference string, respectively.

E. Damerau-Levenshtein Spelling Correction

The raw text from each OCR engine is processed using a Damerau-Levenshtein Distance-based spelling correction module before matching against the drug dictionary constructed during data collection. This step addresses character-level OCR errors from packaging noise, low contrast, and curved surfaces, commonly including insertion, deletion, substitution, and transposition. Unlike standard Levenshtein Distance, which considers only insertion, deletion, and substitution, Damerau-Levenshtein also accounts for adjacent-character transposition, an error not handled by conventional edit-distance correction [23].

For two strings S and T (a candidate reference entry), the distance is computed recursively as the minimum cost among four character-level operations, defined as follows:

1. Insertion.

This operation adds one character to string S to match a character in string T, taking the value from the cell to the left and adding unit cost as Eq. (13) [23]:

$$D_{ins}(i,j) = D(i,j-1) + 1 \tag{13}$$

2. Deletion.

This operation removes a character from string S so that it moves closer to string T, taking the value from the upper cell plus a unit cost as Eq. (14) [23]:

$$D_{del}(i,j) = D(i-1,j) \; + \; 1 \tag{14}$$

3. Substitution.

This operation replaces the character at position i with the character at position j when the two differ, taking the value from the diagonal cell and adding a cost determined by the character-mismatch function as defined in Eq. (15) [23]:

$$D_{sub}(i,j) = D(i-1,j-1) + t(S_i \neq T_j) \tag{15}$$

with the mismatch cost defined as Eq. (16):

$$t(S_i \neq T_j) = \begin{cases} 0, & if S_i = T_j \\ 1, & if S_i \neq T_j \end{cases} \tag{16}$$

4. Transposition.

This operation corrects the swapped position of two adjacent characters, applicable only when $S_i = T_{j-1}$ and $S_{i-1} = T_j$, taking the value from the cell two positions diagonally back and adding a unit cost as defined in Eq. (17) [23]:

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

$$D_{trans}(i,j) = D(i-2, j-2) + 1 \quad (17)$$

The computation is initialized using a $(m + 1) \times (n + 1)$ dynamic-programming matrix, where m and n denote the lengths of S and T. Each cell is filled iteratively by selecting the minimum among the four operations in Eqs. (13)-(17), and the bottom-right cell represents the final edit distance between the two strings. This formulation runs in $O(mn)$ time; recent work has proposed faster bounded-distance algorithms that restrict the search to a threshold edit distance, allowing large drug dictionaries to be searched more efficiently without filling the full matrix for every candidate [42].

This dictionary-lookup formulation, comparing each token exhaustively against a reference vocabulary using edit distance, has recently been shown to achieve high correction accuracy for non-standard Indonesian and English terms when combined with frequency-based candidate ranking [43]. To support candidate ranking independent of string length, a normalized similarity score is additionally computed by Eq. (18):

$$\text{Sim}(S,T) = 1 - \frac{D(S,T)}{\max(|S|,|T|)} \quad (18)$$

where $|S|$ and $|T|$ denote the lengths of the OCR-extracted token and the candidate reference string, respectively. Normalized edit-distance scoring has similarly been shown to improve precision and recall in string-based record matching for noisy, real-world identifier fields [44], while weighted variants of the Levenshtein family have proven effective for transliteration and correction tasks in other morphologically distinct, low-resource languages [45], reinforcing the suitability of edit-distance-based matching for ranking candidate drug names of varying length.

F. Graph Traversal Brand-Generic Relation

Commercial drug packaging frequently displays both a proprietary brand name and its active pharmaceutical ingredient or generic counterpart within the same visual field. This co-occurrence introduces ambiguity into post-OCR string matching, as character-level errors in either token can cause the system to misclassify a brand-name product as a generic equivalent, or vice versa. To resolve this ambiguity without introducing computationally heavy language models, a lightweight graph-based traversal module is integrated downstream of the Damerau-Levenshtein spelling correction stage. This ambiguity reflects a naming inconsistency widely reported among registered pharmaceutical brands sharing the same active ingredient [46]. The relational structure is modeled as a directed graph $G = (V, E)$, where the vertex set $V = Vbrand \cup Vgeneric$ represents the validated drug vocabulary extracted from the database. The edge set $E \subseteq Vbrand \times Vgeneric$ consists of directed edges (b, g) representing the pharmacological equivalence relation is_brand_of(b, g), where brand drug $b \in Vbrand$ contains the generic active ingredient $g \in Vgeneric$. This explicit relational structure allows the disambiguation module to exploit the topological connections surrounding a candidate entity, rather than relying on isolated string similarity alone. Relational structure of this kind has been shown to improve entity matching beyond similarity search alone [47].

Given the set of validated labels present on a single packaging image after Damerau-Levenshtein correction, $L = l_1, l_2, \ldots, l_k \subset V$ the module performs a one-hop graph traversal by visiting each label $l \in L$ against its generic neighbor set $Ng(l) = g \in Vgeneric \mid (l,g) \in E$. A neighbor is accepted as a verified companion only if it also appears in L and its similarity score meets a length-adaptive threshold, since shorter strings are more susceptible to spurious high similarity than longer ones:

$$\tau_{comp}(g) = \begin{cases} 0.80, & |g| \le 6 \\ 0.85, & 6 < |g| \le 10 \\ 0.90, & |g| > 10 \end{cases} \quad (19)$$

A label l is treated as a brand candidate if a generic neighbor g shared between $Ng(l)$ and L achieves a similarity score meeting or exceeding $\tau comp(g)$, where the OCR tokens xi and xj matched to l and g must be distinct; this distinctness constraint prevents a single token from being counted as evidence for both entities. This condition confirms that a detected brand and its corresponding generic composition originate from the same package and therefore constitute a valid brand–generic relationship. The traversal also operates in the reverse direction: if the top-ranked candidate is a generic entity, its associated brands $Nb(g) = b \in Vbrand \mid (b,g) \in E$ are checked against L under the same threshold in Eq. (19). A resolved brand b must have a matched token that is not a substring of the generic token's match and must appear at or before the generic token's position in the OCR text, $pos(b) \le pos(g)$ unless its own similarity score already satisfies Eq. (18) independently. When multiple such candidates qualify, ties are resolved by similarity score, then by token position and character length:

$$b^* = \arg\max_{b \in B} (\text{Sim}(x,b), -pos(x), |x|) \quad (20)$$

where $B \subseteq L$ denotes the set of qualifying brand candidates. Because the traversal only queries the direct one-hop neighbor set of each candidate label, its cost scales with $O(\mid Ng(l) \mid)$ rather than with the size of the entire reference graph, keeping the disambiguation step computationally lightweight and suitable for per-image post-processing.

G. Evaluation Metrics

The performance of each configuration is quantified using seven metrics: Accuracy, WER, CER, Latency,

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

Unresolved Rate, Correction Success Rate (CSR), and False-Correction Rate (FCR). Together, these capture prediction correctness, computational cost, and the specific behavior of the correction stages, relevant for a post-OCR pipeline intended for future use as a decision support tool in resource-constrained healthcare settings. Accuracy is the proportion of images whose final predicted label exactly matches the ground-truth label, with unresolved predictions treated as incorrect, computed as Eq. (21) [48]:

$$\text{Accuracy} = \frac{N_{correct}}{N} \times 100\% \quad (21)$$

where $Ncorrect$ is the number of correctly predicted images out of N evaluated images. WER measures word-level differences between predicted and reference labels, defined as Eq. (22) [48]:

$$\text{WER} = \frac{S_w + D_w + I_w}{N_w} \quad (22)$$

where Sw, Dw, and Iw are the word-level substitutions, deletions, and insertions, and Nw is the number of words in the ground-truth label. Unresolved predictions are assigned WER = 1.0. CER applies the same concept at the character level, as defined in Eq. (23) [48]:

$$\text{CER} = \frac{S_c + D_c + I_c}{N_c} \quad (23)$$

where Sc, Dc, and Ic are character-level substitutions, deletions, and insertions, and Nc is the number of characters in the ground-truth label. Unresolved predictions are assigned CER = 1.0. Latency measures the average post-OCR processing time, including DLSC and GTC, excluding OCR inference since OCR outputs are pre-computed and cached [49], as defined in Eq. (24) [49]:

$$\text{Latency (ms)} = \frac{T_{end} - T_{start}}{N} \times 1000 \quad (24)$$

where $Tstart$ and $Tend$ denote the wall-clock times at the start and end of processing all N images for a given configuration. Unresolved Rate, CSR, and FCR are added to characterize the specific behavior of the correction stages beyond aggregate accuracy: the first captures cases where the pipeline returns no candidate label, while the latter two capture how often correction helps versus harms an already-correct raw prediction, as defined in Eqs. (25)-(27):

$$\text{Unresolved Rate} = \frac{N_{unresolved}}{N} \times 100\% \quad (25)$$

$$\text{CSR} = \frac{N_{fixed}}{N_{wrong_raw}} \times 100\% \quad (26)$$

$$\text{FCR} = \frac{N_{broken}}{N_{correct_raw}} \times 100\% \quad (27)$$

Algorithm 1: Proposed Graph-Assisted Post-OCR Drug Name Recognition Using DLSC and GTC.

Input: Packaging Image I, Drug Dictionary D, Brand–Generic Graph G

Output: Final Predicted Drug Label $\hat{y}$

Function $DNR(I)$:

- $T \leftarrow OCR(I)$; normalize tokens and retain positions
- $L \leftarrow \emptyset$; $B \leftarrow \emptyset$
- $\tau(s) \leftarrow$ 0.80 if $|s| \leftarrow$ 6; 0.85 if $|s| \leftarrow$ 10; otherwise 0.90
- **for each** token $xi \leftarrow T$ **do**
  - Compute normalized DL scores against D
  - Apply prefix/suffix scoring for $|d| \geq 6$
  - Store best label, score, token, and position in L
- **end**
- **if** $L = \emptyset$ **then return** Unresolved
- $C \leftarrow$ top 10 ranked labels in L
- **for each** brand $b \in C$ **do**
  - **for each** generic $g \in N^{g}(b) \cap L$ **do**
    - **if** $token(b) \neq token(g)$ **and** $score(g) \geq \tau(g)$ **then**
      - $B \leftarrow B \cup b$
    - **end**
  - **end**
- **end**
- $g \leftarrow$ highest-ranked label in L
- **if** g is generic **then**
  - **for each** brand $b \in N^{b}(g) \cap C$ **do**
    - **if** $token(b) \neq token(g)$ **and** $score(b) \geq \tau(b)$
      - **and** $matchedToken(b)$ is not a substring of $matchedToken(g)$
      - **and** [$pos(b) \leq pos(g)$ or $score(b) \geq \tau_{strong}$] **then**
      - $B \leftarrow B \cup b$
    - **end**
  - **end**
- **end**
- **if** $B \neq \emptyset$ **then**
  - Return brand ranked by score, position, token, length
- **else**
  - Return highest-ranked DLSC label in L
- **End**
- Return Final Prediction $\hat{y}$

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

where $N_{unresolved}$ is the number of images with no resolved prediction; and N_{fixed} are the number of images incorrect at Raw OCR and the subset subsequently corrected; N_{wrong_raw} and $N_{correct_raw}$ and N_{broken} are the number of images correct at Raw OCR and the subset subsequently changed to incorrect.

IV. Result

This section presents and empirical evaluation of the proposed post-OCR framework across three OCR engines and their correction configurations, covering the experimental setup, qualitative recognition behavior, and quantitative performance on training and testing datasets, assessing the contribution of DLSC and GTC alongside the framework's generalization to unseen packaging images. Additionally, component-wise processing latency is evaluated across CPU and GPU environments to assess practical deployment feasibility. DLSC is further benchmarked against standard post-OCR correction baselines, GTC's design components are isolated through ablation and sensitivity analysis, the statistical significance and effect size of GTC's improvement are established, and correction behavior is examined at the class and label level to characterize residual error sources.

A. Experimental Setup

All experiments were conducted using Python on a single workstation equipped with an NVIDIA GeForce RTX 4090 GPU with 24 GB of VRAM. To assess performance across deployment scenarios relevant to healthcare institutions with varying computational infrastructure, each OCR engine was evaluated under both CPU-only and GPU-accelerated configurations. This dual-configuration setup reflects realistic deployment conditions ranging from resource-constrained facilities relying on CPU inference to mid-to-high-end institutions adopting GPU acceleration, while avoiding the added latency and data-privacy concerns associated with cloud-based inference for pharmaceutical packaging images. Exact software and library versions, including the CUDA toolkit used for GPU acceleration, are reported in Table 3.

The overall inference procedure is summarized in Algorithm 1, consolidating the OCR extraction, DLSC, and GTC stages from Section III into a single function, DNR(I). Given packaging image I, drug dictionary D, and brand–generic graph G, the algorithm extracts and normalizes OCR tokens, then matches each token against D using the length-adaptive threshold $\tau(s)$ and prefix/suffix scoring to form a corrected label set L; if no token meets this threshold, the image is marked Unresolved. Graph verification is restricted to the top-10 ranked labels in L. The traversal first checks each brand candidate's generic neighbors for a token-distinct, threshold-satisfying match; if the top-ranked label is instead generic, the reverse direction is checked, requiring each candidate brand's matched token to be non-substring and positioned at or before the generic

Table 3. Software versions, OCR engine configurations, and computational libraries used in the experimental implementation.

Library	Version	Parameter/Configuration
pytesseract	0.3.13	Python wrapper for Tesseract
Tesseract OCR	5.5.0.2	(OEM 3 = LSTM engine; PSM 11 = sparse text)
easyocr	1.7.2	Language: [‘en’]
paddleocr	2.10.0	Language: en; use_angle_cls=True;
rapidfuzz	3.14.3	Used for DLSC similarity computation
CUDA	13.0	GPU acceleration

token unless its own score already exceeds the stricter threshold τ_strong. When multiple brand candidates qualify, ties are resolved by score, then token position, then length. The prediction $\hat{y}$ resolves to the single qualifying brand; with no valid brand candidate, the highest-scoring DLSC label is returned as a fallback.

Each engine was deployed using its publicly released pretrained model without domain-specific fine-tuning on

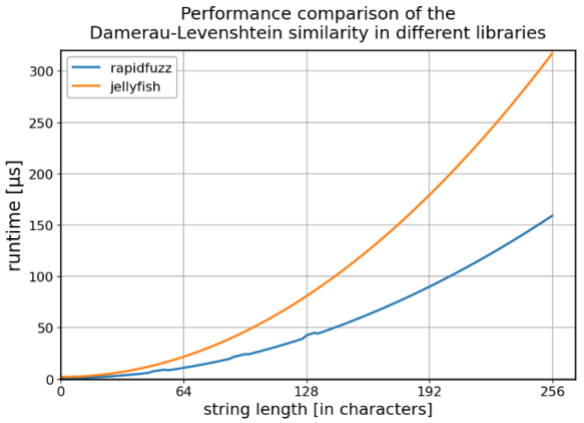

Fig. 5. Computational runtime comparison between RapidFuzz and Jellyfish implementations of DLSC calculation across increasing input string lengths.

the drug packaging dataset, consistent with the framework's objective of evaluating off-the-shelf recognition performance prior to post-processing correction and evaluated under both CPU and GPU

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

configurations as described above to reflect realistic deployment settings. Exact library versions and initialization parameters for each engine are reported in Table 3. PaddleOCR's automatic text-angle classification allows correction of rotated text regions on curved packaging surfaces. Tesseract's Page Segmentation Mode 11 detects sparse, non-sequential text blocks without assuming a fixed reading order, better suited to the irregular spatial arrangement of brand names, dosage information, and regulatory text on pharmaceutical packaging than Tesseract's default page-layout assumption.

For the DLSC computation, the framework adopts the RapidFuzz library, with the version used reported in Table 3, over alternatives such as jellyfish, supported by RapidFuzz's published benchmark comparing runtime across increasing string lengths as shown in Fig. 5. RapidFuzz consistently outperforms jellyfish, with the gap widening as string length increases; at 256 characters, it completes in approximately 160 µs versus over 315 µs for jellyfish, roughly half the runtime. Since the pipeline computes similarity scores between each OCR-extracted token and every candidate in the drug reference database, this efficiency advantage compounds at scale, directly contributing to the low per-image latency reported in Table 4.

Three additional configuration values govern the pipeline's runtime behavior, each determined empirically from recurring OCR failure patterns observed during pipeline development rather than tuned against accuracy on either subset; because the split is performed at the brand-label level, with no label shared between training and testing, these patterns could not reflect characteristics specific to the testing labels. First, graph verification is restricted to the top-10 DLSC candidates, since the correct label fell within this range in most cases. Second, to handle tokens fused with adjacent words or dosage information (e.g., "paracetamol500" extracted as a single token against the reference "paracetamol"), a candidate is accepted as a strong match when it is a prefix or suffix of a longer token sharing at least six characters. Standard Damerau-Levenshtein similarity penalizes such tokens by their full-length difference and often pushes correct matches below the 0.90 threshold in Eq. (19), producing a false negative. The affix-match score is therefore computed as 0.78 plus 0.22 times the ratio of the shorter token's length to the longer token's length, and the six-character minimum prevents coincidental substrings between unrelated drug names from being accepted. Third, multi-word labels are matched against their full string rather than the first word alone, to avoid confusion with unrelated products sharing the same leading term.

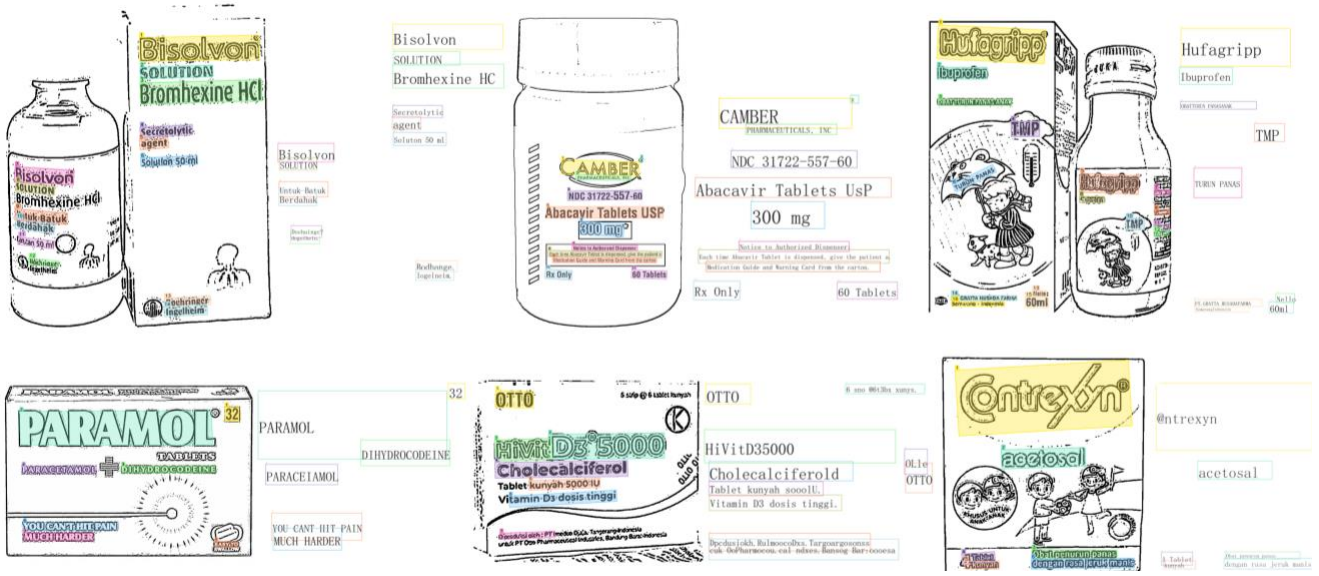

Fig. 6. Representative drug packaging samples and OCR detection and recognition outputs, illustrating the effects of typography, background complexity, and packaging characteristics on text extraction

B. Qualitative OCR Engine Performance

To complement the quantitative results presented in the following subsections, Fig. 6 illustrates representative raw OCR outputs across six drug packaging samples, spanning both favorable and adverse recognition conditions. For each image, the OCR engine localizes candidate text regions with a bounding box and

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

Table 4. Comparative training and testing performance of three OCR engines with DLSC and GTC, evaluated using accuracy, word error rate (WER), character error rate (CER), unresolved rate, correction success rate (CSR), and false-correction rate (FCR).

Model	Accuracy (Train/Test)	WER (Train/Test)	CER (Train/Test)	Unresolved Rate (Train/Test)	CSR (Train/Test)	FCR (Train/Test)
Tesseract OCR	25.52% / 26.59%	74.44% / 73.36%	76.34% / 76.37%	50.73% / 49.61%	0%	0%
Easy OCR	49.90% / 49.08%	50.57% / 51.50%	55.81% / 57.73%	8.48% / 7.3%	0%	0%
PaddleOCR	50.94% / 48.16%	49.61% / 52.52%	56.01% / 60.05%	2.97% / 2.89%	0%	0%
Tesseract OCR + DLSC	44.58% / 44.56%	56.60% / 56.33%	51.20% / 50.90%	0%	25.82% / 24.84%	0.67% / 0.99%
Easy OCR + DLSC	86.11% / 86.23%	14.04% / 13.85%	14.16% / 15.11%	0%	73.16% / 73.79%	0.9% / 0.86%
PaddleOCR + DLSC	90.15% / 89.80%	9.88% / 10.38%	10.44% / 11.60%	0%	81.32% / 81.74%	1.34% / 1.53%
Tesseract OCR + DLSC + GTC	47.69% / 47.19%	53.46% / 53.70%	47.72% / 48.00%	0%	30.04% / 28.35%	0.93% / 0.79%
Easy OCR + DLSC + GTC	92.40% / 92.96%	7.66% / 7.12%	7.05% / 7.28%	0%	85.73% / 87.00%	0.9% / 0.86%
PaddleOCR + DLSC + GTC	**96.13% / 96.69%**	**3.82% / 3.44%**	**3.64% / 3.76%**	0%	**93.54% / 94.52%**	**1.39% / 0.98%**

recognizes the text within them, displaying the recognized string alongside its box for direct comparison with the original packaging. This layout distinguishes detection errors, where a region is missed, merged, or incorrectly bounded, from recognition errors, where the region is correctly localized, but the extracted characters are wrong.

For packaging featuring high-contrast, sans-serif typography and minimal background clutter, the OCR engine performed reliably across both stages of the pipeline: bounding boxes were drawn precisely around each text region, and the recognized output correctly captured the brand name together with its corresponding generic composition. Only minor formatting artifacts appeared, typically involving a single trailing character being dropped rather than misread as a different character. Packaging with decorative illustrations around the printed text was handled with reasonable fidelity, with both the brand name and generic label extracted correctly despite the cluttered background, indicating that moderate clutter alone does not substantially degrade recognition when the typography remains clear. In contrast, packaging with stylized or overlapping fonts introduced more severe recognition errors, even when the bounding boxes were reasonably well placed. Several samples exhibited character-level substitutions consistent with the OCR error taxonomy addressed by the Damerau-Levenshtein correction stage. More pronounced failures involved bounding two adjacent text regions as one, fusing the brand name with dosage information, alongside additional character-level errors affecting the generic name. The most severe cases arose from cursive or heavily stylized brand typography, where the leading character was misread entirely, while smaller, well-defined text regions elsewhere on the same packaging were still recognized correctly despite their reduced print size. These qualitative observations show that recognition failures cluster around labels with decorative or non-standard typography, word-adjacent fusion, and low text-background contrast, motivating the character-level correction and graph-based relational disambiguation mechanisms.

C. Quantitative Performance on Training and Testing Datasets

Table 4 reports Accuracy, WER, CER, Unresolved Rate, Correction Success Rate (CSR), and False-Correction Rate (FCR) for all nine configurations evaluated on both the training and testing datasets, spanning three OCR engines (Tesseract, EasyOCR, PaddleOCR) across three processing stages (Raw OCR, +DLSC, +DLSC+GTC). Because no component undergoes parameter training, the training subset serves only as a consistency check against testing, not

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

Table 5. Component-wise processing latency and total inference time of OCR–DLSC–GTC configurations under CPU and GPU execution environments.

Model	Configuration	OCR Latency (Train/Test)	DLSC Latency (Train/Test)	GTC Latency (Train/Test)	Total Latency (Train/Test)
Tesseract OCR + DLSC + GTC	CPU	1584.34 ms / 1574.46 ms	309.37 ms / 451.56 ms	0.05 ms / 0.05 ms	1893.76 ms / 2026.07 ms
Easy OCR + DLSC + GTC	CPU	59142.57 ms / 49952.83 ms	307.07 ms / 306.08 ms	0.05 ms / 0.05 ms	59449.69 ms / 50258.96 ms
	GPU	1276.15 ms / 1197.95 ms			1583.27 ms / 1504.08 ms
PaddleOCR + DLSC + GTC	CPU	2356.07 ms / 2306.12 ms	261.80 ms / 340.93 ms	0.06 ms / 0.06 ms	2617.93 ms / 2647.11 ms
	GPU	310.99 ms / 285.48 ms			572.85 ms / 626.47 ms

as fitting data. CSR and FCR are computed cumulatively against the Raw OCR baseline at both post-processing stages, so the +DLSC+GTC row reflects the end-to-end correction behavior of the full pipeline rather than GTC's marginal contribution alone

At the Raw OCR stage, all three engines exhibit limited recognition accuracy, ranging from 25.52% for Tesseract to 50.94% for PaddleOCR on the training set, with a comparable pattern on testing (26.59%–48.16%). Tesseract's markedly lower accuracy is accompanied by a substantially higher Unresolved Rate (50.73%/49.61%) than EasyOCR (8.48%/7.3%) or PaddleOCR (2.97%/2.89%), indicating that roughly half of Tesseract's raw predictions fail to produce any token similar enough to a dictionary entry to be resolved at all, consistent with its reliance on page-layout segmentation rather than a dedicated text-detection stage. WER and CER remain correspondingly high across all engines (49.61%–76.37%).

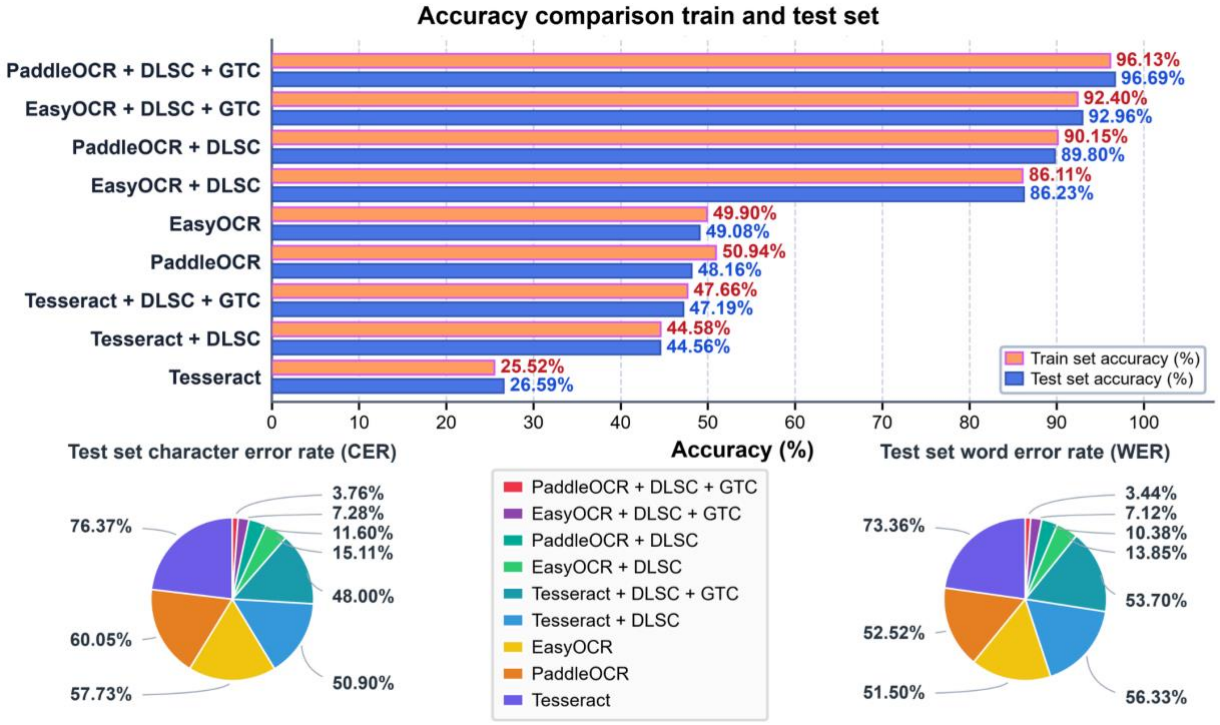

Fig. 7. Performance comparison of nine OCR and post-OCR correction configurations: (a) training and testing accuracy, (b) testing character error rate, and (c) testing word error rate.

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

Applying DLSC yields the single largest improvement across all engines: PaddleOCR rises from 50.94%/48.16% to 90.15%/89.80% (+39.21/+41.64 points), EasyOCR from 49.90%/49.08% to 86.11%/86.23% (+36.21/+37.15 points), while Tesseract improves more modestly from 25.52%/26.59% to 44.58%/44.56% (+19.06/+17.97 points). This gain is accompanied by Unresolved Rate dropping to 0% for all three engines, since the DLSC stage always returns the highest-scoring dictionary match as a fallback rather than leaving a prediction unresolved. CSR against the Raw OCR baseline shows how much of each engine's raw error DLSC alone recovers: PaddleOCR corrects 81.32%/81.74% of its previously wrong predictions, EasyOCR 73.16%/73.79%, while Tesseract recovers only 25.82%/24.84%, indicating that DLSC's correction capacity depends on the underlying OCR output already retaining a reasonable character-level approximation of the true label. FCR remains low across all engines (0.67%–1.53%), showing that DLSC rarely overturns an already-correct raw prediction.

Adding Graph Traversal Correction (GTC) produces a further, smaller but consistent gain on top of DLSC: PaddleOCR reaches 96.13%/96.69% (+5.98/+6.89 points over DLSC alone), EasyOCR reaches 92.40%/92.96% (+6.29/+6.73 points), and Tesseract reaches 47.69%/47.19% (+3.11/+2.63 points). Corresponding reductions are observed in WER and CER, following the same trend in reverse, with PaddleOCR + DLSC + GTC achieving the lowest error rates overall (3.82%/3.44% WER, 3.64%/3.76% CER). CSR rises further at the full pipeline stage, reaching 93.54%/94.52% for PaddleOCR and 85.73%/87.00% for EasyOCR. FCR remains comparably low end-to-end (0.98%–1.39% for PaddleOCR), indicating that the full pipeline rarely overturns an originally correct raw prediction despite the additional relational verification step. Tesseract's smaller CSR gain (30.04%/28.35%, only modestly higher than DLSC's 25.82%/24.84%) reflects its weaker raw baseline, which limits the pool of recoverable brand–generic errors GTC can act on.

Table 6. Quantitative comparison of conventional post-OCR correction methods and the proposed DLSC approach using recognition accuracy, WER, CER, and processing latency.

Method	Accuracy (Train/Test)	WER (Train/Test)	CER (Train/Test)	Latency (Train/Test)
Standard Levenshtein	87.38% / 86.7%	12.72% / 13.59%	13.42% / 14.68%	**14.77 ms / 11.59 ms**
Fuzzy Matching	89.3% / 89.17%	10.98% / 11.23%	11.38% / 12.21%	706.48 ms / 708.42 ms
N-gram Retrieval	87.35% / 86.75%	12.78% / 13.59%	13.38% / 14.77%	16.08 ms / 10.81 ms
Cosine Similarity Matching	87.34 % / 87.01%	12.86% / 13.38%	13.41% / 14.46%	21.39 ms / 11.02 ms
DLSC (Proposed)	**90.15% / 89.80%**	**9.88% / 10.38%**	**10.44% / 11.60%**	261.80 ms / 340.93 ms

Across all configurations, accuracy, WER, and CER values on the testing dataset remain within approximately one percentage point of their training counterparts, as shown in Fig. 7, with PaddleOCR + DLSC + GTC in fact performing marginally better on testing data (96.69% vs. 96.13%). This close train–test agreement is consistent with the rule-based, database-lookup nature of DLSC and GTC, neither of which involves parameter fitting to the training set, so performance depends primarily on whether a queried label exists in the reference database and graph rather than on prior exposure to a specific packaging image.

Table 5 reports end-to-end latency, decomposed into OCR, DLSC, and GTC components, under CPU and GPU configurations; Tesseract is CPU-only, consistent with its native execution environment. OCR inference dominates total latency for all three engines, particularly on CPU, where EasyOCR requires 59,142.57 ms/49,952.83 ms per image, roughly an order of magnitude slower than Tesseract (1,584.34 ms/1,574.46 ms) and PaddleOCR (2,356.07 ms/2,306.12 ms); this gap narrows sharply under GPU acceleration, where EasyOCR and PaddleOCR OCR latency drops to 1,276.15 ms/1,197.95 ms and 310.99 ms/285.48 ms respectively, confirming that EasyOCR's architecture benefits disproportionately from GPU acceleration relative to CPU-only deployment. DLSC latency remains comparatively stable across engines and hardware configurations (261.80–451.56 ms), reflecting its fixed dictionary-search cost independent of the upstream OCR engine or hardware backend. GTC latency is negligible across all configurations (0.05–0.06 ms), consistent with the one-hop, top-K-restricted traversal design in Section III.F. Total latency

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

for PaddleOCR + DLSC + GTC thus ranges from 572.85/626.47 ms (GPU) to 2,617.93/2,647.11 ms (CPU), with hardware backend, not GTC, serving as the primary determinant of overall pipeline latency.

D. Comparison with Baseline Post-OCR Methods

Table 6 compares DLSC against four widely used post-OCR correction baselines, Standard Levenshtein Matching, Fuzzy Matching, N-gram Retrieval, and Cosine Similarity Matching, each evaluated standalone (without GTC) under the same drug reference database and OCR input, isolating the contribution of the correction method itself. Accuracy, WER, CER, and latency are reported for both the training and testing sets, using PaddleOCR as the backend.

The three baselines lacking explicit transposition handling, Standard Levenshtein (87.38%/86.7%), N-gram Retrieval (87.35%/86.75%), and Cosine Similarity Matching (87.34%/87.01%), achieve near-identical accuracy, clustering roughly three percentage points below DLSC (90.15%/89.80%). This convergence suggests these methods reach a similar performance ceiling when restricted to character- or substring-level similarity without accounting for adjacent-character transposition, a common OCR error type. Fuzzy Matching achieves the closest accuracy to DLSC among the baselines (89.3%/89.17%), consistent with RapidFuzz's token-based ratio scoring implicitly tolerating minor positional shifts, though it still falls short of DLSC's explicit transposition modeling. Latency reveals a sharper distinction. Standard Levenshtein, N-gram Retrieval, and Cosine Similarity Matching are substantially faster than DLSC (10.81–21.39 ms versus 261.80 ms/340.93 ms), reflecting their simpler distance computations. Fuzzy Matching, despite its competitive accuracy, incurs the highest latency among the four baselines (706.48 ms/708.42 ms), slower than DLSC, indicating that RapidFuzz's alignment-based ratio scoring is computationally more expensive than dynamic-programming edit distance at the dictionary scale used in this pipeline. These results indicate that DLSC's advantage over generic dictionary-based correction does not stem merely from performing correction at all, but specifically from explicitly modeling the transposition operation absent in the other edit-distance and similarity-based baselines, while remaining substantially more efficient than the only baseline (Fuzzy Matching) that approaches comparable accuracy. This supports the methodological claim that the reported performance gains originate from the specific combination of Damerau-Levenshtein correction and graph-based disambiguation, rather than from post-OCR correction in general.

Table 7. Ablation analysis of graph traversal constraints and image preprocessing components on PaddleOCR-based drug-name recognition performance.

Model Variant	Acc	WER	CER
		(Train/Test)	
Full Model	**96.13% /96.69%**	**3.82% / 3.44%**	**3.64% / 3.76%**
GTC w/o position constraint	96.13% /96.69%	3.82% / 3.44%	3.64% / 3.76%
GTC w/o candidate size	96.12% /96.69%	3.83% /3.44%	3.65%/ 3.76%
GTC w/o Tie-breaking rule	94.58% /96.00%	5.40% /4.21%	5.39% /4.33%
GTC w/o Substring restriction	95.66% /95.53%	4.29% /4.55%	3.84% /4.38%
GTC w/o token-distinctness	95.74% /95.69%	4.21% /4.39%	3.79% /4.16%
GTC w/o reverse traversal	95.32% /95.43%	4.64% /4.65%	4.55% /4.99%
No Threshold	95.73% /96.58%	4.23% /3.55%	3.61% /3.58%
No GTC (DLSC only)	90.15% /89.80%	9.88% / 10.38%	10.44%/ 11.60%
No Image Preprocessing	94.22% /93.32%	5.78% /6.81%	5.66% /6.99%

E. Ablation Studies and Sensitivity Analysis

Table 7 isolates the contribution of individual GTC design components by selectively disabling each rule from the full pipeline, while Fig. 8 presents a complementary sensitivity analysis across four configuration axes governing the pipeline's runtime behavior: the top-K candidate restriction, the minimum substring length for prefix/suffix matching, the acceptance threshold strategy, and the joint base–weight parameterization of the substring scoring function. Removing GTC entirely (DLSC only) produces the largest single drop in performance, from 96.13%/96.69% to 90.15%/89.80% (Table 4), confirming GTC's substantial contribution beyond DLSC alone. Disabling image preprocessing produces a comparably large drop, to 94.22%/93.32%, with WER and CER rising to 5.78%/6.81% and 5.66%/6.99%, respectively, indicating that preprocessing remains a necessary precondition for both DLSC and GTC to operate on sufficiently clean OCR tokens. Among the individual graph-resolution rules, removing the reverse traversal direction causes the largest degradation (95.32%/95.43%, WER 4.64%/4.65%, CER 4.55%/4.99%), indicating that a meaningful share of brand–generic corrections originate from cases where

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

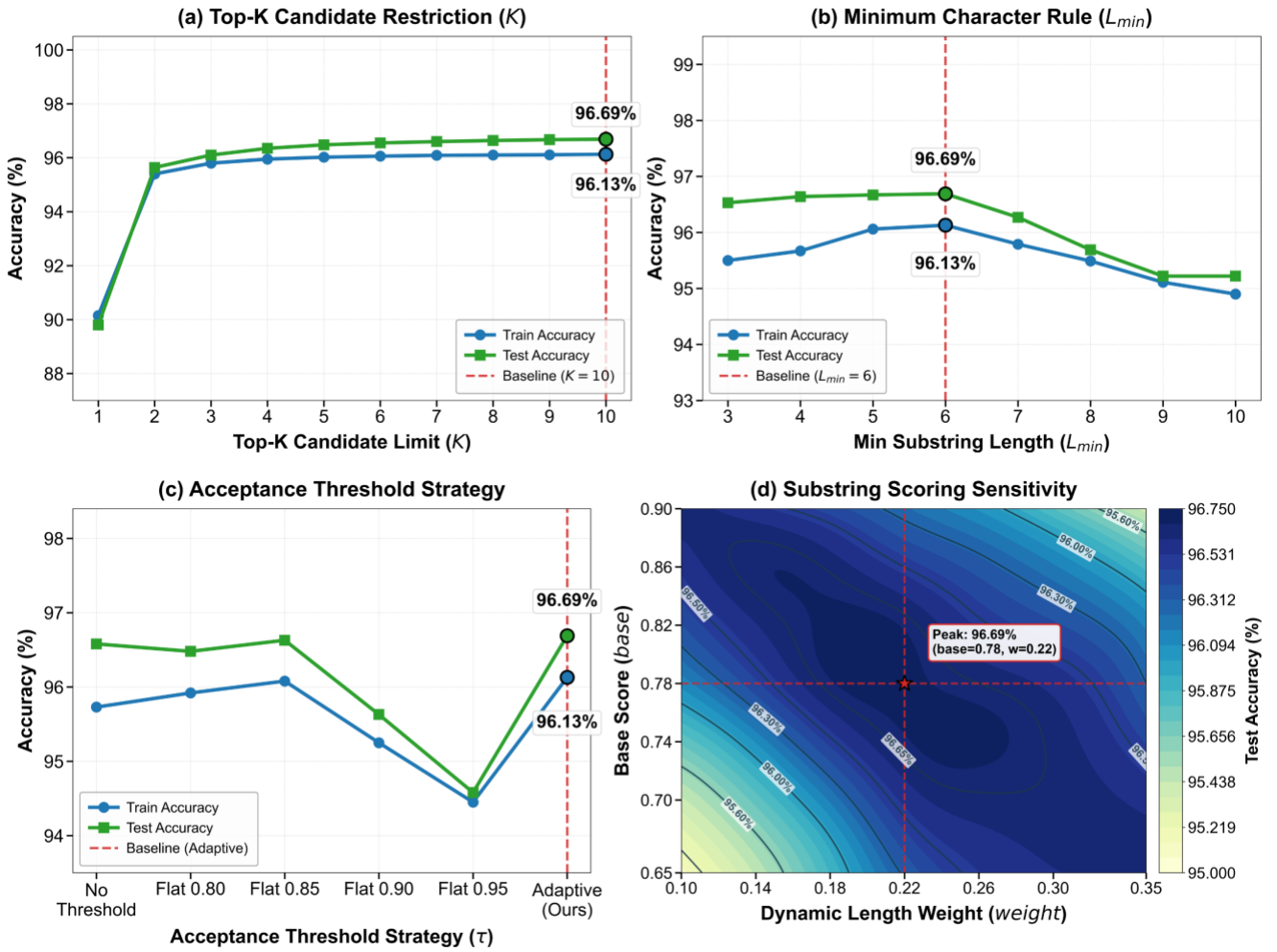

Fig. 8. Sensitivity analysis of GTC parameters: (a) top-K candidate restriction, (b) minimum substring length, (c) acceptance threshold strategy, (d) joint base–weight substring scoring.

the top-ranked DLSC candidate is itself generic rather than brand, which only the reverse-direction check can resolve. Removing the tie-breaking rule produces a similarly large drop on the training set (94.58%, WER 5.40%, CER 5.39%), though its effect is comparatively smaller on testing (96.00%, WER 4.21%, CER 4.33%), suggesting that ambiguous cases with multiple qualifying brand candidates, while relatively rare, are resolved inconsistently without an explicit priority ordering. Disabling the token-distinctness constraint (95.74%/95.69%) and the substring restriction (95.66%/95.53%) each produce moderate accuracy reductions of approximately 0.5–1 point, confirming that both constraints meaningfully reduce false-positive brand–generic matches, even though neither is individually as consequential as the reverse traversal or tie-breaking mechanisms. In contrast, removing the position constraint (96.12%/96.69%) or restricting verification to the top-10 candidate set (96.12%/96.69%, labeled "candidate size" in Table 7) leaves performance virtually unchanged from the full model, indicating these two conditions function primarily as efficiency safeguards against rare edge cases rather than as substantial contributors to accuracy.

The sensitivity analysis in Fig. 8 corroborates these findings from a complementary angle. Fig. 8 (a) shows accuracy plateauing from K=3 onward, with the baseline K=10 (96.13%/96.69%) positioned well within this stable region, consistent with Table 7's finding that top-K restriction contributes negligibly to accuracy once a sufficient candidate pool is retained. Fig. 8 (b) shows a clear peak at the chosen minimum substring length of six characters (96.13%/96.69%), with accuracy declining on either side, most sharply as the threshold approaches ten characters, confirming six as a locally optimal rather than arbitrary choice. Fig. 8 (c) shows the adaptive threshold strategy outperforming all flat-threshold alternatives and the no-threshold baseline, reinforcing the ablation results in Table 7. Fig. 8 (d), a joint contour analysis of the substring-scoring base and weight parameters, confirms that the selected

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

Table 8. Paired statistical comparison of DLSC and DLSC–GTC across three OCR engines, reporting accuracy confidence intervals, percentage-point improvements, McNemar test results, Holm-adjusted p-values, and relative error reduction.

Engine	DLSC Acc. [95% CI]	DLSC+GTC Acc. [95% CI]	Δ pp [95% CI]	McNemar p	Holm-adjusted p	Error Reduction
PaddleOCR	[88.43, 91.17]	[95.90, 97.48]	[5.73, 8.04]	2.461×10^{-38}	7.384×10^{-38}	67.53%
EasyOCR	[84.66, 87.76]	[91.80, 94.06]	[5.57, 7.88]	3.684×10^{-35}	7.368×10^{-35}	48.85%
Tesseract OCR	[42.35, 46.82]	[44.98, 49.50]	[1.94, 3.36]	1.776×10^{-15}	1.776×10^{-15}	4.74%

configuration (base=0.78, weight=0.22) lies at the empirical peak of 96.69% test accuracy, with performance degrading gradually toward both corners of the explored parameter space rather than dropping sharply, indicating the scoring formula is reasonably robust to small perturbations around the chosen values.

Collectively, these results indicate that GTC's accuracy gains are not attributable to a single design choice but arise from the combined effect of several constraints of varying individual weight, reverse traversal and tie-breaking contributing most substantially, token-distinctness and substring restriction offering moderate additional protection against false matches, and the top-K and position constraints serving primarily as computational safeguards rather than accuracy-critical components.

F. Statistical Significance Analysis

To determine whether the accuracy gain from adding GTC to DLSC reflects a genuine improvement rather than sampling variation, two complementary analyses were conducted on the testing set predictions: bootstrap confidence intervals for each configuration's accuracy, and a paired significance test comparing DLSC against DLSC+GTC on the same images. Because each image is evaluated under both configurations, the outcomes are paired rather than independent, and because the per-image outcome is binary (correct/incorrect), the exact McNemar test [50] was adopted, as it directly models discordant pairs of binary outcomes without requiring a normality assumption or a separate omnibus test across groups.

For each image i, let $y_i = 1$ if the prediction is correct and 0 otherwise. Accuracy is the proportion of correctly recognized images among all N evaluated images, including unresolved predictions in the denominator, as defined in Eq. (21). A 95% confidence interval for accuracy was obtained via image-level bootstrap resampling [51]. N image indices are resampled with replacement, accuracy is recomputed on the resample, and this is repeated 10,000 times (seed = 42), with the 2.5th and 97.5th percentiles of the resulting distribution forming the interval. On the PaddleOCR testing set, accuracy rose from 89.80% [88.43%, 91.17%] with DLSC to 96.69% [95.90%, 97.48%] with DLSC+GTC, non-overlapping intervals consistent with a genuine improvement.

Because DLSC and DLSC+GTC are evaluated on the same images, each image falls into one of four outcome pairs. On PaddleOCR, GTC corrected 132 previously wrong predictions while introducing only 1 new error, with 1,707 images remaining correct and 62 remaining wrong under both configurations. The exact McNemar test evaluates only the discordant pairs (b = 132 corrected, c = 1 broken), testing the null hypothesis that correction and breakage are equally likely, as defined in Eq. (28) [50]:

$$p = \min\left(1, 2\sum_{k=0}^{\min(b,c)} \binom{b+c}{k} (0.5)^{b+c}\right) \quad (28)$$

This yielded $p = 2.461 \times 10^{-38}$, far below the conventional significance threshold of α = 0.05, leading to rejection of the null hypothesis; in other words, a result this asymmetric between corrections and breakages would be extremely unlikely to occur by chance alone if GTC provided no real benefit over DLSC.

Beyond statistical significance, two effect-size measures quantify the magnitude of improvement: the absolute percentage-point difference (6.89 points, 95% paired-bootstrap CI: 5.73–8.04) and the relative error reduction, defined as the proportion of DLSC's errors eliminated by GTC, yielding 67.53% for PaddleOCR, indicating that roughly two-thirds of DLSC's residual errors are resolved by GTC. The same procedure was applied to EasyOCR and Tesseract. Because three per-engine tests were conducted, testing multiple hypotheses increases the chance of a false positive by chance alone, so Holm's sequentially rejective procedure [52] was applied to control the family-wise error rate at α = 0.05: p-values are ranked from smallest to largest, and each rank-i p-value is compared against a progressively stricter adjusted threshold, as defined in Eq. (29) [52]:

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

$$p_{(i)}^{\text{Holm}} = \min\left[1, \max_{j \le i}\{(m-j+1)p_{(j)}\}\right] \quad (29)$$

A comparison remains significant only if its Holm-adjusted p-value stays below 0.05. Table 8 summarizes the results. All three Holm-adjusted p-values remain far below 0.05, so the null hypothesis of no difference between DLSC and DLSC+GTC is rejected for every engine, confirming that GTC's benefit is not confined to the best-performing engine. The magnitude, however, differs substantially: PaddleOCR shows the largest effect size (67.53% relative error reduction), EasyOCR a comparable direction of benefit (48.85%), while Tesseract, despite reaching significance, shows a comparatively small reduction (4.74%), consistent with its weaker raw baseline limiting the pool of brand–generic errors GTC can resolve. Statistical significance in the Tesseract case should accordingly be interpreted alongside its substantially lower absolute accuracy (47.19%) rather than as evidence of practically meaningful performance, since a statistically significant result does not by itself imply the improvement is large enough to matter in practice. These results should be interpreted within the scope of the evaluated testing set: the bootstrap and McNemar procedures assume image-level independence, and generalization beyond the present packaging distribution, imaging conditions, and drug population has not been separately validated.

Table 9. Representative examples of drug-name recognition showing raw OCR outputs, ground-truth labels, DLSC predictions, similarity scores, and final GTC-corrected predictions.

OCR Raw	True Label	Predicted		
		DLSC	Sim. Score	GTC
[“RANITIDIN”]	Ranitidine	Ranitidine	0.978	Ranitidine
[“Doxezosin”]	Doxazosin	Doxazosin	0.889	Doxazosin
[“Dolutenravir”]	Dolutegravir	Dolutegravir	0.917	Dolutegravir
[“OCOR”, “Simvastatin”]	Zocor	Simvastatin	1.00	Zocor
[“Dulcelax”, “Bisacodyl”]	Dulcolax	Bisacodyl	1.00	Dulcolax
[“ROSUPI”, “Rosuvastatin”]	Rosupid	Rosuvastatin	1.00	Rosupid

G. Label-Level Accuracy and Correction Analysis

To examine correction behavior beyond the aggregate metrics in Sections IV.C–IV.F, Table 9 traces representative cases through the PaddleOCR + DLSC + GTC pipeline, from raw OCR output to final prediction. These cases reveal two correction behaviors. In most instances, DLSC alone recovers the correct label from a single misread token, with GTC only confirming the prediction. In a smaller subset, the generic token is recognized with little error while the co-occurring brand token contains typographical errors; since DLSC ranks candidates purely by similarity score, the cleanly recognized generic term outranks the imperfectly matched brand term. GTC then traverses the brand–generic graph to identify the valid brand candidate and overrides this generic-biased prediction, showing that its contribution concentrates where OCR quality asymmetry between a brand and its generic counterpart would otherwise cause DLSC to select the wrong entity, even when a valid relational counterpart is present in the same image.

Fig. 9 extends this analysis to the class level. Per-class accuracy for ten representative test classes spanning the observed accuracy range is reported with 95% Wilson confidence intervals in Fig. 9 (a), since the standard normal approximation is unreliable at small sample sizes. Two classes with larger support (Acetazolamide, 17/17; Amikacin, 12/12) achieve 100% accuracy with comparatively tight intervals, while classes with minimal support exhibit substantially wider uncertainty: Apolar (1/2, 50%) spans a Wilson interval of roughly 9–91%, and Hi-D (0/3) and Nutrimax Cmax (0/2) show zero observed accuracy without their intervals excluding non-trivial success rates. This indicates that point estimates alone are insufficient to characterize small-sample classes, motivating the interval-based reporting adopted here rather than relying on raw accuracy figures as in the earlier version of this analysis.

This sensitivity to sample size is systemic rather than isolated to the examples above. Fig. 9 (b) reports the full distribution of class sample sizes across the dataset: the majority of classes are supported by very few images, with 970 of 2,926 training classes (33.2%) and 255 of 752 testing classes (33.9%) containing only a single image, and a further 1,728 training and 450 testing classes falling in the 2–5 image range. Consequently, 705 of 752 testing classes (93.75%) are supported by five or fewer images, meaning that per-class accuracy for most drug classes can shift by 20–50 percentage points from a single prediction changing the outcome. This underscores why class-level rankings based on raw accuracy alone, as previously

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

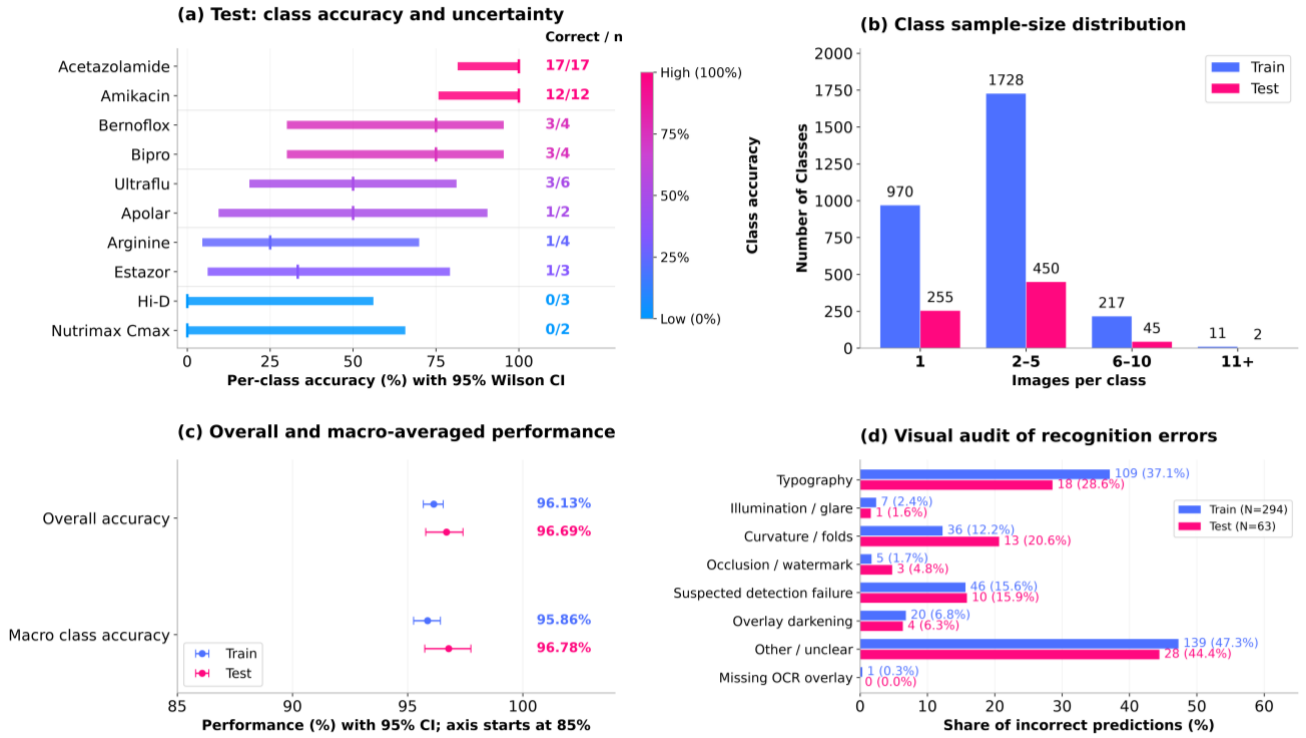

Fig. 9. Class-level recognition performance and residual error analysis: (a) per-class accuracy with 95% Wilson confidence intervals, (b) class sample-size distribution, (c) overall versus macro-averaged accuracy with confidence intervals, and (d) visual categorization of recognition errors.

reported, should be interpreted cautiously rather than as a stable measure of per-drug reliability.

Given this instability at the class level, aggregate accuracy metrics warrant closer scrutiny as to whether they are driven disproportionately by high-support classes. Fig. 9 (c) compares overall accuracy, in which each image contributes equally, against macro-averaged class accuracy, in which each class contributes equally regardless of support, both with 95% bootstrap confidence intervals (10,000 image-level resamples for overall; 4,000 class-level resamples for macro, seed = 42). On the testing set, overall accuracy reaches 96.69% [95.90%, 97.48%] and macro class accuracy reaches 96.78% [95.74%, 97.74%], closely matching training values of 96.13% [95.69%, 96.55%] and 95.86% [95.24%, 96.40%] respectively. The close agreement between overall and macro accuracy indicates that performance remains high even when small classes are weighted equally with larger ones, though this aggregate closeness does not preclude individual low-support classes from showing poor accuracy, as seen in Fig. 9 (a).

Beyond sample-size effects, the visual factors underlying incorrect predictions were manually audited across all 294 training and 63 testing errors, as summarized in Fig. 9 (d). Each misclassified image was visually inspected and assigned to the single most likely contributing factor. Typography-related issues are the most frequent (37.1% training, 28.6% testing), followed closely by curvature or folds on cylindrical packaging (12.2%/20.6%) and suspected detection failure, where no OCR bounding box was produced (15.6%/15.9%). A substantial share of errors (47.3%/44.4%) could not be attributed to a clear visual factor and are categorized as "Other/unclear." Because categories are not mutually exclusive, these proportions should be read as descriptive rather than as formal causal attributions. Taken together, these class-level and visual findings indicate that the residual errors remaining after DLSC and GTC correction are not attributable to a single dominant failure mode. Rather, they arise from the interaction of limited per-class sample support, which inflates the apparent variability of class-level accuracy, and genuine packaging-level visual degradation that constrains what the upstream OCR stage can extract before any correction mechanism is applied.

V. Discussion

A. Interpretation of Main Findings

The substantially larger accuracy gains from DLSC (up to +41.64 points) compared with GTC (up to +6.89 points) reflects their differing scope: DLSC corrects the broad class of character-level OCR errors dominating misclassifications across all engines, while GTC targets a narrower, structurally specific error, brand–

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

Table 10. Comparative evaluation of the proposed framework and existing drug-label OCR studies in terms of dataset characteristics, computational requirements, validation strategies, recognition accuracy, WER, and CER.

Study	Dataset	Sample Size	Computational Requirement	Validation Setting	Acc (%)	WER (%)	CER (%)
[6]	Internal drug packaging	18 test conditions	PC + webcam; hardware NR	Three lighting levels: 5, 35, 182 lux	83.78	-	-
[7]	ADL dataset	6,491	N/A	Train / test	-	23.7	21.6
[9]	NHS reports + MiBio pages	100 + 211	Hardware NR	Before/after OCR correction	-	13.67 / 18.62	12.48 / 17.17
[10]	Vietnamese prescription	4,000	Core i7-1165G7; 16 GB RAM	Train / validation / test	91.27	2.0	2.0
[12]	ICDAR SROIE + SynthText	1,000	Hardware NR	600 train-val / 400 test	96.5 / 97.1	-	-
This Study	Indonesian drug packaging	9,500	RTX 4090 GPU and CPU	Brand-label-level 80:20 split	96.69	3.44	3.76

generic co-occurrence ambiguity, arising only when both tokens are correctly extracted but assigned to the wrong entity. The baseline comparison in Section IV.D reinforces that this gain is specific to DLSC's explicit transposition modeling rather than dictionary-based correction in general: three baselines lacking transposition handling converge to a similar, lower accuracy ceiling, while the only baseline approaching DLSC's accuracy (Fuzzy Matching) does so at substantially higher latency. The statistical analysis in Section IV.F further confirms that GTC's contribution is systematic rather than incidental: all three engines show a statistically significant improvement after Holm correction, with the ablation results in Section IV.E showing that this improvement is not attributable to any single design choice but to the combined effect of several constraints, reverse traversal and tie-breaking contributing most substantially, while the position and candidate-size constraints function primarily as computational safeguards.

The disparity in DLSC's effectiveness across engines (+41.64 points for PaddleOCR versus +17.97 points for Tesseract on the testing set) further indicates that post-OCR correction is not a substitute for adequate detection quality; rather, DLSC's dictionary-lookup correction can only recover a label when the underlying character sequence retains sufficient similarity to it. This dependency implies that the choice of base OCR engine remains a first-order determinant of overall pipeline accuracy, with post-processing serving to amplify rather than replace a reasonably capable detector. The end-to-end latency results in Table 5 further clarify where computational cost in the full pipeline originates. For the best-performing configuration, PaddleOCR + DLSC + GTC, total latency is dominated by OCR inference itself (2,356–2,306 ms on CPU versus 311–285 ms on GPU) rather than by DLSC (262–341 ms) or GTC (0.06 ms), confirming that GPU acceleration of the OCR stage, not optimization of the post-processing cascade, is the primary lever for reducing deployment latency. This effect is most pronounced for EasyOCR, whose CPU inference time (59,143–49,953 ms) is roughly 37 times its GPU counterpart (1,276–1,198 ms), while Tesseract, which lacks GPU support entirely, instead derives its comparatively low total latency (1,894–2,026 ms) from a lightweight CPU-only backbone rather than from any advantage in detection quality. Within this cost structure, GTC's near-zero overhead (0.05–0.06 ms across all engines) confirms that brand–generic disambiguation adds negligible cost regardless of deployment hardware, in contrast to the post-processing-only latency reported earlier for DLSC+GTC alone, which reflects the correction stages in isolation and should not be read as representative of total deployment latency.

B. Comparison with Previous Studies

Table 10 situates these results against prior drug-label OCR studies that report quantitative outcomes. At 9,500 images spanning 3,678 brand labels, the present dataset is substantially larger than [6] (18 test conditions), [9] (311 documents), [10] (4,000 prescriptions), and [12] (1,000 receipts), and is the only one evaluated under a brand-label-level train/test split rather than purely in-domain testing; [7] 's ADL dataset (6,491 images) is closer in scale but reports neither a comparable split nor an accuracy figure. Direct accuracy comparison is further limited by differing task definitions: [6] reports field-matching success on six known drug names under controlled lighting, and [12]

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

reports character- and field-level accuracy on receipt layouts rather than full drug-name identification, so their 83.78% and 96.5–97.1% figures are not measuring the same task as this study's 96.69%.

Among studies reporting comparable WER/CER, the proposed framework (3.44% WER, 3.76% CER) outperforms the YOLOv8+CRNN-gMLP pipeline in [7] (23.7%/21.6%) and the RoBERTa-based post-correction in [9] (13.67–18.62%/12.48–17.17%), both of which lack an explicit brand–generic disambiguation mechanism. The Vietnamese prescription system in [10] reports lower error rates still (2.0%/2.0%), but on a narrower, single-language extraction task with substantially less brand–generic co-occurrence than Indonesian pharmaceutical packaging. Given these differences in dataset composition, task scope, and metric definition, the comparison is indicative of relative standing rather than a controlled benchmark conducted under matched experimental conditions.

C. Limitations and Implications

Despite the overall accuracy improvement achieved by the proposed framework, several limitations should be acknowledged. First, DLSC's similarity-only matching can, in isolated cases, select an unrelated dictionary entry when a substring coincidentally achieves a high similarity score, as illustrated in Table 9; this indicates that character-level correction alone cannot always guarantee correct candidate selection prior to graph verification, underscoring the complementary role GTC plays in the pipeline. Second, all three OCR engines were evaluated using their publicly released pretrained weights without domain-specific fine-tuning on drug packaging images, which may not reflect the highest achievable recognition accuracy for this domain. Third, because the dataset was assembled via keyword-driven web search rather than field photography, it likely skews toward products with higher online visibility and toward favorable studio-style imaging conditions, which may not fully represent the degraded lighting, handling, and packaging diversity typical of real dispensing environments or other regional markets. Fourth, GTC's similarity thresholds were determined empirically from observed candidate-matching behavior rather than exhaustively tuned; the sensitivity analysis in Section IV.E indicates the chosen configuration lies near a locally optimal region rather than at an arbitrary point, though this does not preclude a more optimal configuration outside the explored parameter range. Finally, the visual error audit presented in Section IV.G was conducted as a single-reviewer, provisional assessment rather than an independently validated, blinded annotation process; the reported error-factor proportions should accordingly be treated as exploratory characterization rather than confirmatory evidence of causal failure sources.

The framework's reliance on lightweight, rule-based correction stages, rather than large language models, positions it as a feasible decision-support candidate for resource-constrained healthcare facilities where computational infrastructure and internet connectivity may be limited, particularly given that GTC itself adds negligible latency regardless of hardware configuration. The demonstrated separation between DLSC-correctable and GTC-correctable error types, supported by the component-level ablation in Section IV.E, also suggests that future development effort is best allocated toward refining candidate-selection criteria in DLSC and extending domain-specific fine-tuning of the underlying OCR engines, rather than redesigning the graph traversal mechanism itself. More broadly, the substantial gains in OCR-based drug-name recognition accuracy achieved by this framework indicate potential value as a decision-support or verification aid within pharmacy dispensing workflows or mobile applications. However, these results reflect recognition performance on static packaging images rather than demonstrated reductions in real-world medication errors; prospective evaluation in actual dispensing settings is needed before claims regarding look-alike, sound-alike (LASA) error mitigation or medication-safety effectiveness can be substantiated.

VI. Conclusion

This study proposed a graph-assisted post-OCR framework combining Damerau-Levenshtein Spelling Correction (DLSC) and Graph Traversal Correction (GTC), confirming that both can be cascaded to outperform either stage alone without relying on large language models or learned graph embeddings, thereby resolving character-level OCR errors and brand–generic naming ambiguity in pharmaceutical packaging recognition. PaddleOCR + DLSC + GTC achieved the best performance, raising accuracy from 50.94% (Raw OCR) to 96.13%/96.69% (training/testing), with WER at 3.82%/3.44%, CER at 3.64%/3.76%, and GTC adding only 0.05–0.06 ms beyond DLSC. DLSC's gain stems from explicit transposition modeling rather than generic dictionary correction, while GTC's improvement over DLSC is statistically significant across all three engines, driven mainly by reverse traversal and tie-breaking, with correction concentrated where DLSC favors a generic name despite a valid co-occurring brand candidate. The principal contribution lies in this lightweight DLSC+GTC cascade, which, to the best of our knowledge, is among the few frameworks to jointly address character-level correction and relational brand–generic disambiguation without training, thereby offering a practical decision-support candidate for resource-constrained healthcare settings; residual errors stem from limited per-class sample support and packaging-level visual degradation rather than a single failure mode. Future work may

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

expand the brand–generic reference graph, validate the framework in real dispensing settings, and explore cross-engine complementarity across broader packaging conditions.

Acknowledgment

The authors acknowledge the Data Science Study Program and Informatics Engineering Study Program at Telkom University, Purwokerto Campus, Indonesia, for providing academic and institutional support during this research.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Data Availability

The dataset was collected from publicly accessible pharmaceutical packaging images and cross-validated using the Indonesian National Agency of Drug and Food Control (BPOM) database and the Alodokter platform. The dataset comprises 9,500 images covering 3,678 drug brand labels. Supporting research materials, including annotations, dataset partitions, drug dictionaries, and brand–generic relationships, are available from the corresponding author upon reasonable request, subject to applicable copyright and licensing restrictions.

Code Availability

All source code used in this study is available from the corresponding author upon reasonable request.

Author Contribution

Fito Satrio: Conceptualization, data collection, data curation, formal analysis, and investigation. Muhammad Raafi'u Firmansyah: Methodology, software development, implementation of DLSC and GTC, and writing-review and editing. Diah Septiani: Formal analysis, validation, interpretation of results, and writing, review and editing. All authors reviewed and approved the final manuscript and agreed to be accountable for the integrity and accuracy of the work.

Declarations

Ethical Approval

study involved computational analysis of publicly available pharmaceutical packaging images and did not involve human participants, animal experiments, or identifiable personal health information. Therefore, formal ethical approval and informed consent were not considered applicable. The authors should confirm compliance with the relevant institutional research ethics requirements.

Consent for Publication Participants.

Not applicable. This study did not involve any individual participants, and therefore no consent for publication was required.

Competing Interests

The authors declare that they have no competing financial or non-financial interests that could have influenced the research findings or interpretation.

Generative AI and AI-Assisted Technologies

The authors declare that AI-assisted technology was used solely for language editing. The authors reviewed and revised the content and take full responsibility for the publication. No AI-assisted technology was used for code generation, image preparation, or data analysis.

References

[1] A. Z. Al Meslamani, “Medication errors during a pandemic: what have we learnt?,” *Expert Opin. Drug Saf.*, vol. 22, no. 2, pp. 115–118, 2023, doi: 10.1080/14740338.2023.2181954.

[2] R. A. Elliott, E. Camacho, D. Jankovic, M. J. Sculpher, and R. Faria, “Economic analysis of the prevalence and clinical and economic burden of medication error in England,” *BMJ Qual. Saf.*, vol. 30, no. 2, pp. 96–105, Feb. 2021, doi: 10.1136/bmjqs-2019-010206.

[3] Badan Pengawas Obat dan Makanan Republik Indonesia, “Cek Produk - Badan Pengawas Obat dan Makanan RI,” https://cekbpom.pom.go.id/.

[4] D. S. Atmaja, R. Y. Saksono, Yulistiani, Suharjono, and E. Zairina, “Evaluation of medication errors in one of the largest public hospital: A retrospective study,” *Clin. Epidemiol. Glob. Health*, vol. 28, pp. 1–5, Jul. 2024, doi: 10.1016/j.cegh.2024.101640.

[5] T. S. Supapaan, J. Napaporn, R. Ruthirakanok, S. Srichathum, A. Songmuang, and C. Pitchayajittipong, “Look-alike sound-alike (LASA) medication errors in Thai hospitals,” *Pharm. Pract. (Granada).*, vol. 22, no. 1, Jan. 2024, doi: 10.18549/PharmPract.2024.1.2886.

[6] W. P. Putri, I. Munadhif, M. K. Hasin, N. Rinanto, and R. Y. Adhitya, “Implementasi Optical Character Recognition untuk klasifikasi penempatan obat berdasarkan kelas terapi di apotek,” *Jurnal Elektronika dan Otomasi Industri*, vol. 11, no. 2, pp. 420–432, Jul. 2024, doi: 10.33795/elkolind.v11i2.5341.

[7] M. Bensouilah, M. Taffar, and E. Boulham, “Efficient End-to-End Deep Learning Model for Algerian Drug Label Recognition,” in *Lecture Notes in Networks and Systems*, 2024, pp. 93–103. doi: 10.1007/978-3-031-71848-9_9.

[8] P. Rimdusit, P. Aphiromyarnont, S. Phimoltares, and S. Panthuwadeethorn, “Extracting Information from Drug Label Using Image and Text Processing,” in *Proceedings* -

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

21st International Joint Conference on Computer Science and Software Engineering, JCSSE 2024, Institute of Electrical and Electronics Engineers Inc., 2024, pp. 377–382. doi: 10.1109/JCSSE61278.2024.10613666.

[9] S. Karthikeyan, A. G. S. De Herrera, F. Doctor, and A. Mirza, "An OCR Post-Correction Approach Using Deep Learning for Processing Medical Reports," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 5, pp. 2574–2581, May 2022, doi: 10.1109/TCSVT.2021.3087641.

[10] K. Le Bao, T. T. Hoang Cong, T. N. Hoang Anh, M. V. Duc, T. H. Le Hai, and N. Thoai, "MediFind - A Medicine Detection Framework for Vietnamese Medical Prescription," in *Proceedings - 2023 International Conference on Advanced Computing and Analytics, ACOMPA 2023*, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 38–45. doi: 10.1109/ACOMPA61072.2023.00016.

[11] N. Mukazhanov, Z. Alibiyeva, A. Yerimbetova, A. Kassymova, and N. Alibiyeva, "Development of an augmented Damerau-Levenshtein method for correcting spelling errors in Kazakh texts," *Eastern-European Journal of Enterprise Technologies*, vol. 5, no. 2(125), pp. 23–33, 2023, doi: 10.15587/1729-4061.2023.289187.

[12] S. Wu and F. Chang, "A dual-engine fusion optical character recognition method for fast identification and key information extraction of drug labels," *Alexandria Engineering Journal*, vol. 128, pp. 1027–1036, Sep. 2025, doi: 10.1016/j.aej.2025.05.037.

[13] P. Thetbanthad, B. Sathanarugsawait, and P. Praneetpolgrang, "Automated Redaction of Personally Identifiable Information on Drug Labels Using Optical Character Recognition and Large Language Models for Compliance with Thailand's Personal Data Protection Act," *Applied Sciences (Switzerland)*, vol. 15, no. 9, pp. 1–13, May 2025, doi: 10.3390/app15094923.

[14] K. Gromova and V. Elangovan, "Automatic Extraction of Medication Information from Cylindrically Distorted Pill Bottle Labels," *Mach. Learn. Knowl. Extr.*, vol. 4, no. 4, pp. 852–864, Dec. 2022, doi: 10.3390/make4040043.

[15] A. Ghazaryan, J. Hovhanisyan, A. Kirakosyan, A. Aramyan, and S. Moosa Moghadam, "An Intelligent Web Application for Drug Information Retrieval Using Optical Character Recognition and Deep Learning," *Norwegian Journal of development of the International Science*, no. 152, pp. 86–96, Mar. 2025, doi: 10.5281/zenodo.15024999.

[16] K. Almash and A. Indriyanti, "Identifikasi Merek Obat Menggunakan Metode OCR dengan Perbandingan Morphological Opening dan Closing," *Journal of Informatics and Computer Science (JINACS)*, vol. 3, pp. 494–505, Jun. 2022, doi: 10.26740/jinacs.v3n04.p494-505.

[17] J. Ma, D. Li, Y. Chen, Y. Qiao, H. Zhu, and X. Zhang, "A Knowledge Graph Entity Disambiguation Method Based on Entity-Relationship Embedding and Graph Structure Embedding," *Comput. Intell. Neurosci.*, vol. 2021, pp. 1–11, 2021, doi: 10.1155/2021/2878189.

[18] M. Nagayi, A. Scaria Khan, T. Frank, R. Swart, and C. Nyirenda, "Evaluating OCR Performance on Food Packaging Labels in South Africa," in *Communications in Computer and Information Science*, 2025, pp. 127–143. doi: 10.1007/978-3-032-11733-5_8.

[19] V. K. Soni, V. Shukla, S. R. Tandan, A. Pimpalkar, N. K. Nema, and M. Naik, "Performance Evaluation of Efficient and Accurate Text Detection and Recognition in Natural Scenes Images Using EAST and OCR Fusion," *International Journal of Advanced Computer Science and Applications*, vol. 16, no. 1, pp. 445–453, 2025, doi: 10.14569/IJACSA.2025.0160144.

[20] N. Zhao, "WebPII: Benchmarking Visual PII Detection for Computer-Use Agents," *ICLR 2026 Conference Workshop Track*, Mar. 2026, doi: 10.48550/arXiv.2603.17357.

[21] J. Feng *et al.*, "Evaluating OCR performance for assistive technology: effects of walking speed, camera placement, and camera type," *Disabil. Rehabil. Assist. Technol.*, pp. 1–41, 2026, doi: 10.1080/17483107.2026.2673178.

[22] R. R. R and M. Nazeem, "Open-Source OCR Libraries: A Comprehensive Study for Low Resource Language," *Proceedings of the 21st International Conference on Natural Language Processing (ICON)*, pp. 416–421, Dec. 2024, [Online]. Available: https://aclanthology.org/2024.icon-1.48/

[23] Y. Chaabi and F. Ataa Allah, "Amazigh spell checker using Damerau-Levenshtein algorithm and N-gram," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, pp. 6116–6124, 2022, doi: 10.1016/j.jksuci.2021.07.015.

[24] S. M. S. Dashti and S. F. Dashti, "Improving the quality of Persian clinical text with a novel spelling correction system," *BMC Med. Inform. Decis. Mak.*, vol. 24, no. 1, pp. 1–21, 2024, doi: 10.1186/s12911-024-02613-0.

[25] Husni, Y. D. Pramudita, M. Syarief, A. Justitia, and I. O. Suzanti, "Improving Computational

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

Efficiency and Accuracy of Damerau-Levenshtein Distance for Indonesian Spelling Correction using Cosine Similarity," *Journal of Innovation Information Technology and Application (JINITA)*, vol. 7, no. 2, pp. 305–313, Dec. 2025, doi: 10.35970/jinita.v7i2.2893.

[26] D. Fatah, Y. Negara, N. Nasrulloh, and N. Utami, "Enhancing Sentiment Analysis Accuracy Through Intelligent Spelling Correction Using Damerau-Levenshtein Distance and N-Gram with Random Forest Classifier," *Ingénierie des systèmes d information*, vol. 30, no. 10, pp. 2609–2624, Oct. 2025, doi: 10.18280/isi.301008.

[27] H. Cui *et al.*, "A review on knowledge graphs for healthcare: Resources, applications, and promises," *J. Biomed. Inform.*, vol. 169, pp. 1–14, 2025, doi: 10.1016/j.jbi.2025.104861.

[28] L. Farrugia, L. M. Azzopardi, J. Debattista, and C. Abela, "medicX-KG: a knowledge graph for pharmacists' drug information needs," *J. Biomed. Semantics*, vol. 16, no. 1, pp. 1–27, 2025, doi: 10.1186/s13326-025-00332-7.

[29] P. Perdomo-Quinteiro and A. Belmonte-Hernández, "Knowledge Graphs for drug repurposing: a review of databases and methods," *Brief. Bioinform.*, vol. 25, no. 6, pp. 1–3, 2024, doi: 10.1093/bib/bbae461.

[30] M. S. Ikrar Musyaffa, N. Yudistira, M. A. Rahman, A. H. Basori, A. B. Firdausiah Mansur, and J. Batoro, "IndoHerb: Indonesia medicinal plants recognition using transfer learning and deep learning," *Heliyon*, vol. 10, no. 23, pp. 1–16, 2024, doi: 10.1016/j.heliyon.2024.e40606.

[31] Y.-S. You and Y.-S. Lin, "A Novel Two-Stage Induced Deep Learning System for Classifying Similar Drugs with Diverse Packaging," *Sensors*, vol. 23, no. 16, pp. 1–29, 2023, doi: 10.3390/s23167275.

[32] W. Wang, X. Jing, S. Jiang, and Y. Han, "Pill-MSVAE: A Multiscale Variational Autoencoder Based on Internet of Things for Accurate Automatic Drug Identification and Monitoring," *IEEE Internet Things J.*, vol. 13, no. 11, pp. 22680–22690, 2026, doi: 10.1109/JIOT.2026.3664411.

[33] I. Majid Mohammed and N. Ashidi Mat Isa, "Contrast Limited Adaptive Local Histogram Equalization Method for Poor Contrast Image Enhancement," *IEEE Access*, vol. 13, pp. 62600–62632, 2025, doi: 10.1109/ACCESS.2025.3558506.

[34] A. Fawzi, A. Achuthan, and B. Belaton, "Adaptive Clip Limit Tile Size Histogram Equalization for Non-Homogenized Intensity Images," *IEEE Access*, vol. 9, pp. 164466–164492, 2021, doi: 10.1109/ACCESS.2021.3134170.

[35] R. Ahmad, S. Naz, and I. Razzak, "Efficient skew detection and correction in scanned document images through clustering of probabilistic hough transforms," *Pattern Recognit. Lett.*, vol. 152, pp. 93–99, Dec. 2021, doi: 10.1016/j.patrec.2021.09.014.

[36] Z. Zhang, Y. Cheng, J. Suo, L. Bian, and Q. Dai, "INFWIDE: Image and Feature Space Wiener Deconvolution Network for Non-Blind Image Deblurring in Low-Light Conditions," *IEEE Transactions on Image Processing*, vol. 32, pp. 1390–1402, 2023, doi: 10.1109/TIP.2023.3244417.

[37] B. Bataineh *et al.*, "A Comprehensive Review on Document Image Binarization," *J. Imaging*, vol. 11, no. 5, pp. 1–30, 2025, doi: 10.3390/jimaging11050133.

[38] M. Liao, Z. Zou, Z. Wan, C. Yao, and X. Bai, "Real-Time Scene Text Detection with Differentiable Binarization and Adaptive Scale Fusion," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 919–931, 2023, doi: 10.1109/TPAMI.2022.3155612.

[39] Y. Liu, Y. Wang, and H. Shi, "A Convolutional Recurrent Neural-Network-Based Machine Learning for Scene Text Recognition Application," *Symmetry (Basel).*, vol. 15, no. 4, pp. 1–17, 2023, doi: 10.3390/sym15040849.

[40] R. Buoy, M. Iwamura, S. Srun, and K. Kise, "Explainable Connectionist-Temporal-Classification-Based Scene Text Recognition," *J. Imaging*, vol. 9, no. 11, pp. 1–20, 2023, doi: 10.3390/jimaging9110248.

[41] B. Mohamed Nadjib, B. Meriem, and N. Lamri, "An OCR system optimized for Algerian drug package labels," *Journal of Molecular and Pharmaceutical Sciences*, vol. 04, no. 02, pp. 61–70, 2025, [Online]. Available: https://jmps.crsp.dz/wp-content/uploads/2025/12/Article-6-An-ocr.pdf

[42] R. Yamano and T. Shibuya, "Faster Algorithm for Bounded Damerau–Levenshtein Distance," in *Lecture Notes in Computer Science*, 2025, pp. 291–303. doi: 10.1007/978-3-032-05228-5_23.

[43] D. Maheswari, D. Putri, and A. Wirdiani, "Implementation of Dictionary Lookup and Damerau-Levenshtein Distance for Correcting Non-Standard Indonesian and English Terms," *Jurnal Ilmiah Merpati (Menara Penelitian Akademika Teknologi Informasi)*, vol. 13, no. 1, pp. 63–73, May 2025, doi: 10.24843/JIM.2025.v13.i01.p07.

[44] A. Kaufman and A. Klevs, "Adaptive Fuzzy String Matching: How to Merge Datasets with

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

Only One (Messy) Identifying Field," *Political Analysis*, vol. 30, pp. 590–596, Oct. 2021, doi: 10.1017/pan.2021.38.

[45] S. Hajbi, O. Amezian, N. E. Moukhi, R. Korchiyne, and Y. Chihab, "Moroccan Arabizi-to-Arabic conversion using rule-based transliteration and weighted Levenshtein algorithm," *Sci. Afr.*, vol. 23, pp. 1–19, 2024, doi: 10.1016/j.sciaf.2024.e02073.

[46] J. Napaporn, P. Buakaew, P. Suksakornthanawat, S. Sripa, P. Jinatongthai, and T. Supapaan, "Analogous comparison of registered brand name drugs of tablets and capsules commercially available in Thailand: A retrospective study," *PLoS One*, vol. 17, no. 10, 2022, doi: 10.1371/journal.pone.0276321.

[47] D. Obraczka and E. Rahm, "Fast Hubness-Reduced Nearest Neighbor Search for Entity Alignment in Knowledge Graphs," *SN Comput. Sci.*, vol. 3, no. 6, pp. 1–19, 2022, doi: 10.1007/s42979-022-01417-1.

[48] H. Ullah, M. Tanveer, and A. Jan, "Enhancing Handwritten Prescription Recognition with AI-Driven OCR," *Journal of Computing and Biomedical Informatics*, vol. 9, no. 2, 2025, [Online]. Available: https://www.jcbi.org/index.php/Main/article/view/1054/734

[49] P. Li, X. Wang, K. Huang, Y. Huang, S. Li, and M. Iqbal, "Multi-Model Running Latency Optimization in an Edge Computing Paradigm," *Sensors*, vol. 22, no. 16, pp. 1–17, 2022, doi: 10.3390/s22166097.

[50] J. A. Roldán-Nofuentes, T. S. Sheth, and J. F. Vera-Vera, "Hypothesis Test to Compare Two Paired Binomial Proportions: Assessment of 24 Methods," *Mathematics*, vol. 12, no. 2, pp. 1–23, Jan. 2024, doi: 10.3390/math12020190.

[51] S. Mokhtar, Z. Yusof, and H. Sapiri, "Confidence Intervals by Bootstrapping Approach: A Significance Review," *Malaysian Journal of Fundamental and Applied Sciences*, vol. 19, no. 1, pp. 30–42, Feb. 2023, doi: 10.11113/mjfas.v19n1.2660.

[52] S. Holm, "Board of the Foundation of the Scandinavian Journal of Statistics A Simple Sequentially Rejective Multiple Test Procedure A Simple Sequentially Rejective Multiple Test Procedure," *Source: Scandinavian Journal of Statistics*, vol. 6, no. 2, pp. 65–70, 1979, [Online]. Available: https://www.jstor.org/stable/4615733

Author Biography



Fito Satrio was born in Jakarta, Indonesia, in 2005. He has been pursuing an undergraduate degree in Data Science at Telkom University Purwokerto, Indonesia, since 2023. His research interests include computer vision, deep learning, and natural language processing, with a particular focus on applying these techniques to healthcare informatics and medical text recognition problems. He has previously worked on several applied computer vision projects, including human fall motion analysis, driver drowsiness detection, and pothole detection, reflecting a broader interest in developing practical, real-time recognition systems for safety and healthcare applications. He is particularly interested in developing lightweight, computationally efficient methods that remain practical for deployment in resource-constrained settings, rather than relying solely on large-scale pretrained models. He actively engages in applied machine learning research aimed at improving the reliability and accessibility of automated systems in real-world healthcare settings.



Muhammad Raafi'u Firmansyah received his Master of Engineering degree from Gadjah Mada University, Yogyakarta, Indonesia, in 2022. His master's thesis focused on speaker identification using artificial neural networks, and he published part of his research in an IEEE publication in 2021. He is currently a lecturer at the Faculty of Informatics, Telkom University Purwokerto, Indonesia. His research interests include machine learning, speech recognition, and intelligent data analysis, with particular expertise in classification problems. He has also been involved in several research and development projects related to computer vision, the Internet of Things (IoT), and intelligent systems. His research aims to apply artificial intelligence techniques to solve practical problems using data-driven approaches.



Diah Septiani received her Master of Computer Science in Artificial Intelligence from Universitas Gadjah Mada, Indonesia. She is currently a Lecturer in the Bachelor of Data Science Study Program at Telkom University Purwokerto, Indonesia. Her research interests include Artificial Intelligence, Machine Learning, Data Science, Computer Vision, Image Processing,

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).

and Optical Character Recognition (OCR). Her research focuses on the development and application of intelligent and data-driven systems for addressing real-world problems, particularly in image analysis and information processing. She is also actively involved in research, academic development, technology innovation, and community engagement by applying Artificial Intelligence and Data Science across domains including healthcare, education, industry, and society.

Manuscript received 1 September 2026; Revised 23 October 2026; Accepted 26 October 2026; Available online 31 October 2026
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v8i4.2062
Copyright © 2026 by the authors. This work is an open-access article and licensed under a Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).