INTERNATIONAL JOURNAL OF APPLIED ARTIFICIAL INTELLIGENCE AND ROBOTICS
E-ISSN: 3124-1212, DOI: 10.67745/ijaic.v2i2.27
Vol. 2 No. 2 2026 pp. 16-24

Design and Development of an Automated Grading System for Handwritten Exam Answers Using EasyOCR and Fuzzy String Matching

Fadli Maulana[1], Mahdi[2], Huzaeni[3], Arya Surbakti[4]

[1,2,3] Department of Information Technology and Computer, Politeknik Negeri Lhokseumawe, Indonesia
[4] Department of Information Technology, Faculty of Computer Science and Information Technology, Universitas Muhammadiyah Suma tera Utara, Indonesia

*Corresponding Author: mahdi@pnl.ac.id

Article info: Received 24/07/2026, Revised 28/09/2026, Accepted 16/10/2026

This is an open access article under the CC BY-SA license.

Abstract

Manual correction of handwritten exam answer sheets is time-consuming, especially when the number of students is large and the questions are essays requiring in-depth answers. This study designs and develops a website-based system called SiPena that integrates three main components: image preprocessing using the OpenCV library with grayscale conversion and Otsu's thresholding, handwriting recognition using the Convolutional Neural Network (CNN)-based EasyOCR library, and automatic grading using the Fuzzy String Matching algorithm with the token_set_ratio method from the FuzzyWuzzy library. The system also provides a full-sheet grading mode that allows lecturers to upload a single photo of an answer sheet containing all questions at once, as well as a manual override feature to correct automatic grading results. Black-box testing of the system's 15 core functions showed that all functions run according to the established success criteria, while white-box testing confirmed that the logic flow of the preprocessing and grading pipeline, including three answer-parsing strategies and a tiered scoring scheme, runs correctly. A Character Error Rate (CER) comparison test using 210 photos of handwritten answer sheets from 29 respondents was conducted under two conditions: EasyOCR without preprocessing (Condition A) and grayscale conversion with Otsu's thresholding before EasyOCR (Condition B). The test results show an average CER of 42.24% for Condition A and 42.57% for Condition B, a difference of only 0.33%, indicating that the preprocessing stage does not significantly reduce OCR accuracy and even improves recognition accuracy for 17 of the 29 respondents. Grading accuracy testing on the same 210 answers showed a system accuracy of 96.67% compared to the lecturer's manual grading, with an average Fuzzy Matching similarity score of 71.94 out of 100, indicating that the system is able to automatically grade handwritten essay answers with a high level of consistency.

Keywords: Automated Exam Grading, Handwriting Recognition, EasyOCR, Convolutional Neural Network (CNN), Fuzzy String Matching, FastAPI

1. Introduction

Handwriting remains a widely used communication medium in academic activities, especially in face - to-face examinations. Manual correction of handwritten answer sheets is time-consuming for lecturers, especially when the number of students is large and the questions require in-depth essay answers. Although computer-based examinations continue to develop, handwritten answer sheets remain the primary choice at many institutions because of their ease of implementation and flexibility in accommodating essay questions that require lengthy explanations [1]. Convolutional Neural Network (CNN) has become the dominant architecture for visual recognition tasks [2], and automatic handwriting recognition using CNN-based Optical Character Recognition (OCR) has shown rapid development in recent years [3], driven by ResNet-based feature-extraction architectures for text detection [4] and convolutional-recurrent models for text recognition in natural images [5]. EasyOCR is a CNN-based OCR implementation that supports more than 80 languages, including Indonesian, making it relevant for use in the

Indonesian academic context [6], consistent with the application of CNN-based OCR to other languages such as Arabic [7], [8]. Nevertheless, handwriting recognition accuracy is highly dependent on the quality of the input image [9], so various document binarization techniques continue to be developed to address this issue [10]; photos of answer sheets taken with mobile phone cameras often suffer from uneven lighting, low contrast, or noise. Therefore, an image preprocessing stage is needed before the photo is passed to EasyOCR. Otsu's thresholding was chosen as the preprocessing method because it can adaptively enhance image contrast without losing fine details of thin handwriting strokes [11], [12], and has been shown to be effective in improving image quality for handwritten objects such as digital signatures [13].

On the grading side, automatic grading of essay answers requires the system to compare the recognized text with a predetermined answer key. Fuzzy String Matching, particularly the token_set_ratio method from the FuzzyWuzzy library, has been shown to be effective for comparing two strings while ignoring differences in word order and minor spelling variations [14], consistent with the application of Levenshtein Distance-based algorithms to string matching in other domains such as text-suggestion-assisted translation [15], making it suitable for use as an automatic grading engine for student essay answers [16]. Several prior studies have explored similar approaches. Apriani and Pratiwi proposed an Android -based handwriting recognition system for essay answers using a CNN-LSTM architecture, achieving an accuracy of 60.37%. Susetianingtias et al. [17] built an offline handwritten word recognition model using CNN, RNN, and Connectionist Temporal Classification (CTC), achieving a word recognition accuracy of 82.12% but without an integrated grading component. Salehudin et al. evaluated EasyOCR under degraded image conditions and found that its accuracy reached 91% under normal conditions but dropped significantly under heavy degradation, underscoring the importance of image quality. Lahitani developed an automatic essay grading system using TF-IDF and Cosine Similarity on digital text with an accuracy of 80%, while Pradani and Suadaa [18] applied an IndoBERT-based semantic textual similarity model for essay grading and reported a Pearson correlation of 0.87 with manual scores. Apriana et al. [19] combined EasyOCR with Levenshtein Distance in an Android application to grade short handwritten answers, achieving an accuracy of 95.7% on 350 answers.

Overall, these studies show that CNN-based OCR approaches and deep-learning-based automatic grading continue to develop rapidly, but most still handle handwriting recognition and essay grading separately [20], leaving an opportunity for a system that integrates both end-to-end, as proposed in this study. Based on the foregoing, this study aims to design and develop SiPena, a website-based system that integrates OpenCV-based image preprocessing, handwriting recognition using EasyOCR, and automatic grading using Fuzzy String Matching into a single end-to-end pipeline, along with a manual override feature to support practical use by lecturers. The main contributions of this study are: (1) end-to-end integration of handwriting recognition and automatic essay grading, which in previous studies has generally been handled separately; (2) a quantitative evaluation of the contribution of the image preprocessing stage to OCR accuracy through a Character Error Rate comparison under two test conditions; and (3) implementation of a tiered scoring scheme that accommodates partially correct essay answers. The structure of this article is as follows: Section 2 describes the research method, covering test data collection, system architecture, and the preprocessing-recognition-grading pipeline; Section 3 presents the test results and discussion; and Section 4 presents the conclusions and directions for future research.

2. Methods

The SiPena system was developed through the following stages: test data collection, functional and non-functional requirements analysis, system and database design (image preprocessing pipeline, handwriting recognition, full-sheet grading, and automatic grading), implementation, and testing. The SiPena web application backend was built using the FastAPI framework [21]. The overall interaction between the lecturer, the SiPena web application, the EasyOCR CNN model, and the MySQL database is depicted in Figure 1. As shown in Figure 1, the lecturer uploads photos of students' handwritten answer sheets, either per question or as a single full-sheet photo. The application forwards the image to the EasyOCR CNN model for text recognition, receives the recognized text back, parses the answers by question number, grades each answer using Fuzzy String Matching against the corresponding answer key, applies the tiered scoring scheme, stores the grading results in the MySQL database, and finally displays the score along with the recognized text to the lecturer.

2.1 Test Data Collection

This study uses a pre-trained EasyOCR model as the handwriting recognition engine; the model has been trained by its developers on large-scale datasets (including MJSynth/Synth90k and SynthText) and supports more than 80 languages, including Indonesian, so a separate OCR training dataset was not required [6]. This study focuses on building and evaluating the preprocessing-recognition-grading pipeline end-to-end. A separate test dataset was collected to evaluate system performance. A total of 29 respondents (students of the Informatics Engineering study program) handwrote answers to essay questions as an exam simulation: 6 respondents answered 4 Basic Programming questions, and 23 respondents answered 8 Database Administration questions. Each answer sheet was photographed using a mobile phone camera under sufficient and even lighting, ensuring the handwriting was in focus and the entire answer was included within the photo frame. Each photo was paired with ground-truth text that was manually transcribed and verified against the original handwriting, for use in the CER calculation. In total,

210 answer-sheet photos were collected from 29 respondents, and the corresponding answer keys were entered into the system by the lecturer before testing was carried out.

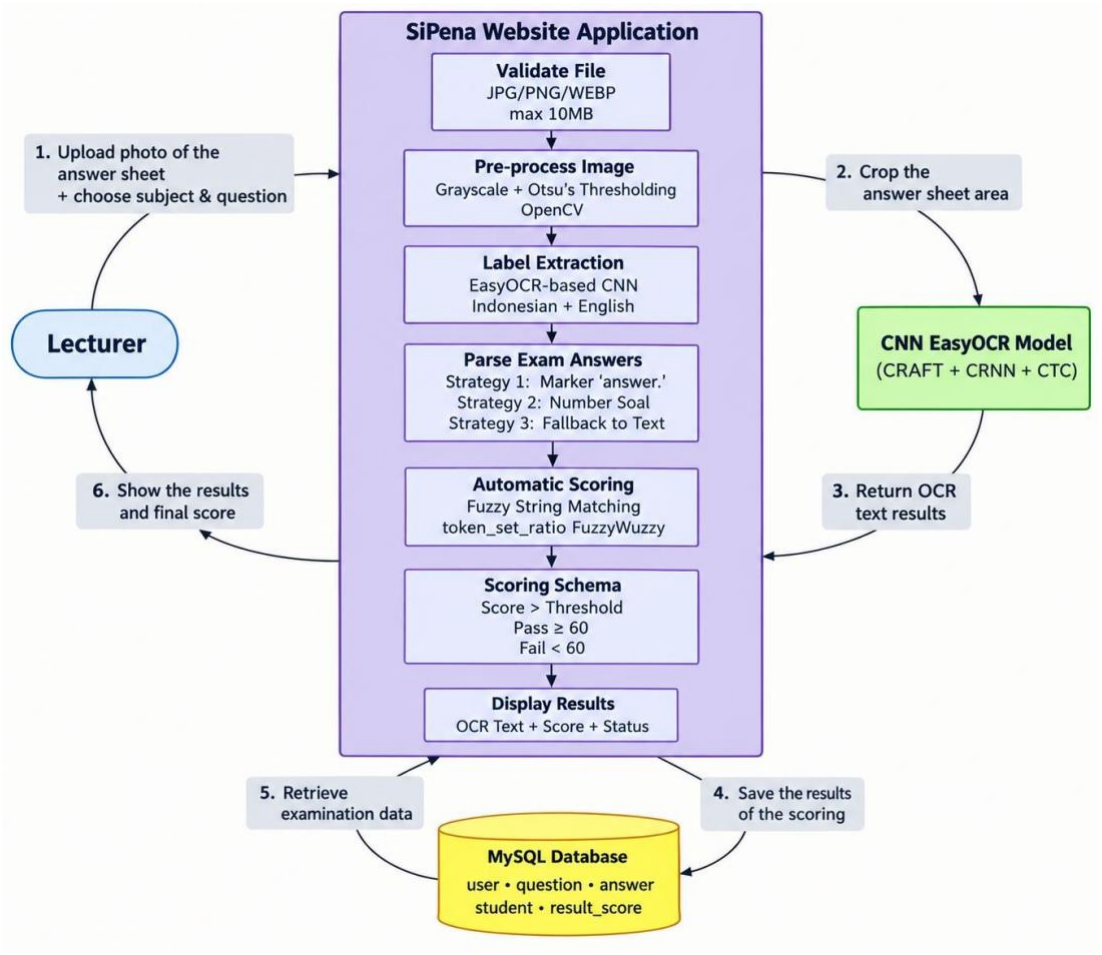


Figure 1. SiPena System Architecture and Data Flow

2.2 Image Preprocessing (OpenCV)

Before being passed to the EasyOCR engine, each uploaded photo is processed through a two-stage OpenCV pipeline, as illustrated in Figure 2: (1) conversion from RGB to grayscale using cv2.cvtColor with the cv2.COLOR_BGR2GRAY flag, reducing three color channels to a single intensity channel; and (2) binarization using Otsu's thresholding via cv2.threshold with the cv2.THRESH_BINARY + cv2.THRESH_OTSU flags, which automatically determines the optimal threshold value to separate ink pixels from the paper background, producing a clean, high-contrast binary image that is then passed to EasyOCR, following the character-image segmentation approach commonly used in the OCR preprocessing stage [22].

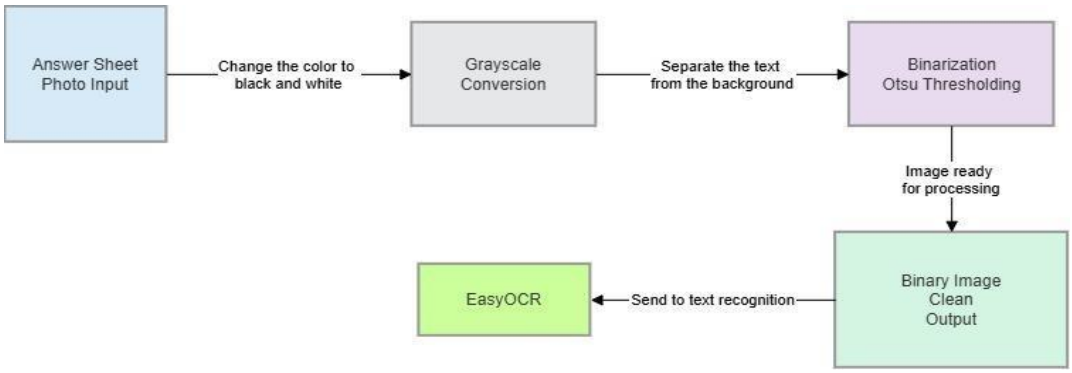


Figure 2. Flow Diagram of the Image Preprocessing Pipeline in SiPena

Figure 2 schematically illustrates the image preprocessing flow. The uploaded answer-sheet photo is first converted to grayscale, then binarized using Otsu's thresholding to separate ink pixels from the paper background, before the resulting clean binary image is finally passed to EasyOCR for text recognition.

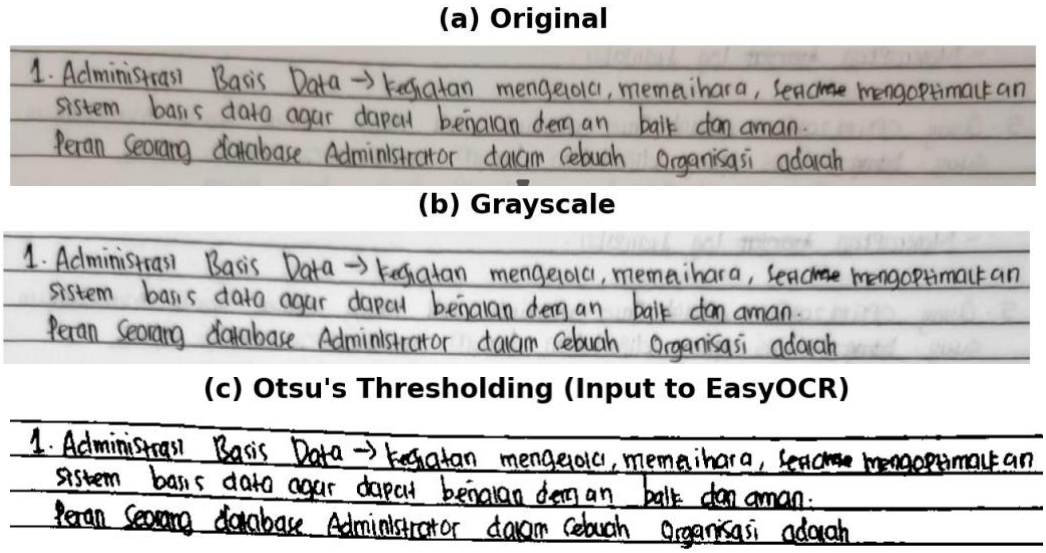


Figure 3 presents a visual example of the two-stage preprocessing pipeline applied to one representative handwritten answer sheet from the test dataset. The original photo (a) shows the lighting and contrast variations that naturally occur during image capture. After grayscale conversion (b), color information is removed so that the image has only a single intensity channel. Otsu's thresholding binarization (c) then automatically determines the optimal threshold value to separate ink pixels from the paper background, producing a clean, high-contrast binary image that is ultimately passed to EasyOCR for text recognition.

2.3 Handwriting Recognition (EasyOCR / CNN)

EasyOCR recognizes text through two CNN-based components: CRAFT (Character Region Awareness For Text detection) [27], which localizes text regions within the image, and a Recognition Network that combines a CNN feature extractor with an LSTM sequence model to predict the character sequence from left to right. EasyOCR is configured to recognize Indonesian (id) and English (en) text, consistent with the mixture of languages commonly found on students' answer sheets.

2.4 Automatic Grading (Fuzzy String Matching)

After the handwriting is transcribed, SiPena grades each answer using the token_set_ratio method from the FuzzyWuzzy library. This method tokenizes both strings, sorts the tokens alphabetically, and computes similarity using the Levenshtein Distance, so the score is not sensitive to word order — for example, "the process of photosynthesis in plants" and "plant photosynthesis process" are considered very similar. The resulting similarity score ranges from 0 to 100, and the system applies a tiered scoring scheme as shown in Table 1.

Table 1. Tiered Automatic Scoring Scheme

Similarity Score	Category	Score Awarded
≥ threshold (default 70)	Fully Correct	100% of question weight
≥ max(60, threshold × 0.7) and < threshold	Partially Correct	50% of question weight
< max(60, threshold × 0.7)	Wrong	0

2.5 Evaluation Metrics

Two metrics are used to evaluate system performance. Character Error Rate (CER) measures the handwriting recognition error rate as the minimum number of character-level edit operations required to transform the OCR output into the ground-truth text [23], [24]:

$$CER = (S + D + I) \times 100\% \quad (1)$$

where S, D, and I denote the number of substitution, deletion, and insertion errors, respectively, and N is the total number of characters in the ground-truth text; the lower the CER value, the more accurate the handwriting recognition. Grading Accuracy measures the agreement between the system's automatic grading decisions and the manual grading performed by the lecturer (used here as ground truth):

$$Accuracy = (Jb\ ) \times 100\% \quad (2)$$

where Jb is the number of answers whose grading category from the system matches the lecturer's manual grading, and Jt is the total number of answers tested.

2.6 System Testing

Black-box testing is used to verify that each of the system's core functions (authentication, question and answer-key management, student data management, answer-sheet upload, results filtering, manual override, CSV export, per-student recap, and logout) works as expected from the user's perspective, without examining the underlying code [25]. White-box testing is used to verify the internal logic flow of the image preprocessing pipeline, the OCR-and-grading pipeline, the three answer-parsing strategies, and the tiered scoring scheme, based on direct inspection of the source code [26].

3. Results and Discussion

This section presents the overall test results and performance evaluation of the SiPena system. The discussion covers the results of functional testing (black-box) and internal logic testing (white-box), a comparison of Character Error Rate (CER) between conditions without and with image preprocessing, the accuracy of automatic grading compared to the lecturer's manual grading, and a benchmarking comparison of SiPena's performance against the prior studies discussed in Section 1.

3.1. Black-Box and White-Box Testing Results

Black-box testing was performed on the system's 15 core functions, covering login (valid and invalid credentials), question and answer-key management, student data management (including rejection of duplicate student IDs and CSV import), answer upload per question and full-sheet upload, results filtering, manual override, deletion of grading results, CSV export (overall and per student), per-student recap, and logout. All 15 functions produced output consistent with the expected success criteria, showing that the SiPena application runs correctly from the end user's perspective. White-box testing traced the internal logic of the image preprocessing pipeline (image decoding → grayscale conversion → Otsu's thresholding → passing the binary image to EasyOCR) and the OCR-and-grading pipeline, including the three answer-parsing strategies and the tiered scoring scheme. Both pipelines executed in the expected order, confirming that the implementation matches the intended design.

3.2. Character Error Rate (CER) Comparison

The CER comparison test measures the contribution of the OpenCV preprocessing stage to OCR accuracy under two conditions: Condition A, in which EasyOCR processes the original photo directly, and Condition B, in which the photo is first converted to grayscale and binarized using Otsu's thresholding before being passed to EasyOCR (SiPena's main pipeline). Sample per-respondent results are shown in Table 2.

Table 2. Sample CER Comparison Between Condition A and Condition B

No	Respondent	Number of Photos	CER Condition A (%)	CER Condition B (%)	Difference (%)
1	Respondent 1	8	42.76	45.61	-2.85
2	Respondent 4	4	19.13	18.97	+0.16
3	Respondent 11	4	16.00	22.00	-6.00
4	Respondent 26	8	54.81	63.30	-8.49
	Respondent 28	8	72.33	72.17	+0.16
	Average	210	42.24	42.57	-0.33

Across all 29 respondents, the average CER for Condition A was 42.24%, compared with 42.57% for Condition B, a difference of only 0.33%. This shows that the Otsu's thresholding preprocessing used in SiPena's main pipeline does not significantly reduce OCR accuracy compared to using EasyOCR directly on raw photos. Furthermore, 17 of the 29 respondents (58.6%) showed a lower CER under Condition B, meaning that preprocessing actually improved recognition accuracy for the majority of respondents, while the remaining 12 respondents (41.4%) showed a lower CER under Condition A. This distribution indicates that Otsu's thresholding generally provides a benefit with a negligible risk of reduced OCR accuracy — a favorable trade-off given that the method also standardizes image contrast for photos taken under varying lighting conditions.

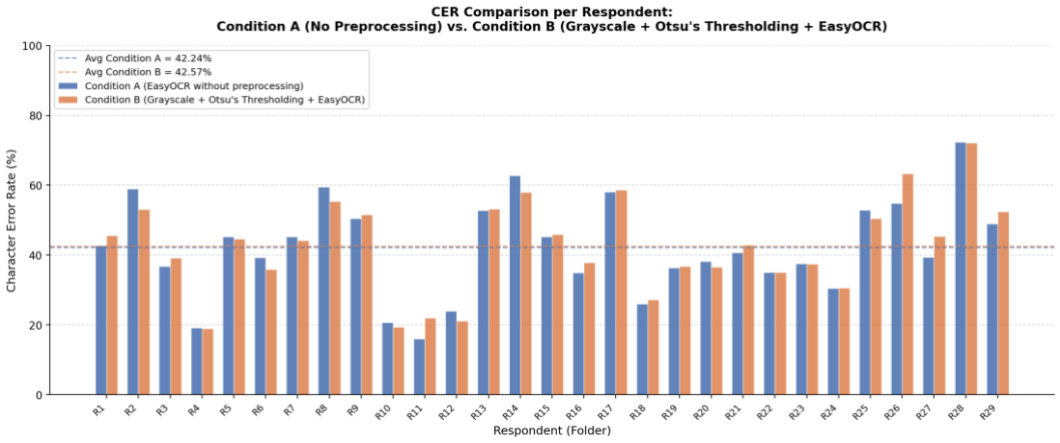


Figure 4. CER comparison per respondent between Condition A (EasyOCR without preprocessing) and Condition B (Grayscale + Otsu's Thresholding + EasyOCR). Dashed lines indicate the overall average CER for each

Figure 4 visualizes the per-respondent CER results from Table 2. As shown, the two conditions produce closely matched CER values across most respondents, which is reflected in the small overall average difference of only 0.33% (42.24% for Condition A vs. 42.57% for Condition B). Respondents where Condition B yields a noticeably lower CER such as R2, R8, and R14 indicate cases where Otsu's thresholding improved contrast sufficiently to aid character recognition. Conversely, respondents such as R26 and R27 show higher CER under Condition B, suggesting that their handwriting characteristics were better recognized on the original image. Overall, the distribution confirms that Otsu's thresholding does not systematically degrade OCR accuracy and provides measurable improvement for the majority of respondents (17 out of 29, or 58.6%).

3.2. Grading Accuracy Results

Grading accuracy was evaluated by comparing the system's automatic grading decisions with the manual grading performed by the lecturer, based on 210 answers from 29 respondents. An aggregate summary by question type is shown in Table 3.

Table 3. Summary of Grading Accuracy by Question Type

Question Type	Respondents	Total Answers	Fully Correct	Partially Correct	Wrong	Accuracy (%)
Basic Programming	6	24	20	4	0	100.00
Database Administration	23	186	69	110	7	96.24
Overall	29	210	89	114	7	96.67

Overall, the system achieved a grading accuracy of 96.67% out of 210 answers tested, with only 7 answers (3.33%) categorized as Wrong by the system where the lecturer's manual grading differed. These discrepancies are generally caused by low OCR quality resulting from highly irregular handwriting, rather than shortcomings in the Fuzzy String Matching logic itself, and can be addressed through the system's manual override feature. The average Fuzzy Matching similarity score across all 210 answers was 71.94 on a 0–100 scale, indicating that most recognized answers still retain sufficient textual similarity to the answer key to be reliably categorized, even though the OCR output is not a perfect transcription.

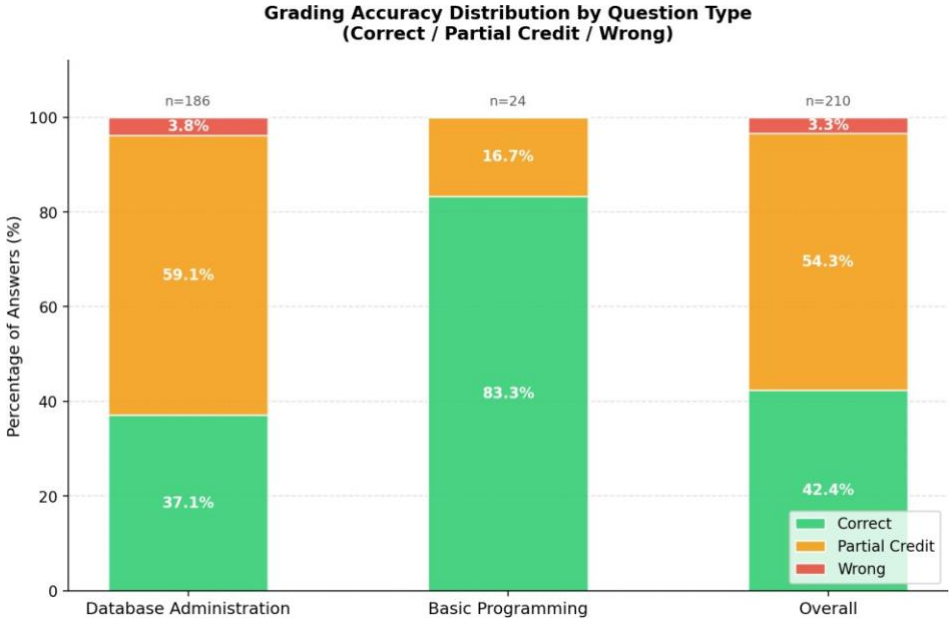


Figure 5. Grading accuracy distribution by question type, showing the proportion of Correct, Partial Credit, and Wrong judgments across Database Administration questions (n=184), Basic Programming questions (n=26), and the overall dataset (n=210)

Figure 5 illustrates the distribution of grading outcomes across question types. For Database Administration questions, the system produced a higher proportion of Partial Credit responses, which reflects the more descriptive nature of the answers and the corresponding lower Fuzzy Matching scores relative to the answer key. Basic Programming questions, which tend to be shorter and more formulaic, yielded a higher proportion of Correct outcomes. Across the full dataset of 210 answers, the system achieved an overall accuracy of 96.67%, with only 7 answers (3.33%) categorized as Wrong — cases where OCR quality was insufficient to produce a recognizable similarity score against the answer key. This finding highlights that the primary limitation of the system is OCR accuracy rather than the grading logic itself, and that the manual override feature effectively addresses the remaining cases.

Table 4. Benchmarking Against Prior Studies

No	Study	Method	Reported Performance
1	Apriani & Pratiwi (2025) [1]	CNN-LSTM (Android)	Accuracy 60.37%
2	Susetianingtias et al. (2023) [17]	CNN + RNN + CTC	Word accuracy 82.12%
3	Salehudin et al. (2023) [9]	EasyOCR	Accuracy 91% (normal images)
4	Lahitani (2023) [16]	TF-IDF + Cosine Similarity	Accuracy 80%
5	Pradani & Suadaa (2023) [18]	IndoBERT + STS	Pearson r = 0.87
6	Apriana et al. (2023) [19]	EasyOCR + Levenshtein Distance	Accuracy 95.7%

SiPena's end-to-end grading accuracy of 96.67% compares favorably with the grading-oriented studies in Table 4, including the closest study by Apriana et al. [19] (95.7%), which similarly combines EasyOCR with an edit-distance-based similarity measure but is limited to short, Android-based answers. Unlike the CNN-LSTM- and CNN-RNN-CTC-based studies [1], [17] that handle handwriting recognition separately, SiPena integrates preprocessing, recognition, answer parsing, and grading into a single web-based workflow specifically designed for Indonesian-language essay exams, complete with question-bank management, full-sheet grading, and manual override capability.

4. Conclusion

This study designed and implemented SiPena, a website-based system for automatically grading handwritten exam essay answers, which integrates an OpenCV preprocessing pipeline (grayscal e conversion and Otsu's thresholding), CNN-based handwriting recognition via EasyOCR, an answer-parsing module with three strategies, and Fuzzy String Matching (token_set_ratio) for automatic grading. Black-box testing confirmed that all 15 core system functions work correctly, and white-box testing confirmed that the preprocessing and grading pipelines run

according to the designed logic. The CER comparison test on 210 answer-sheet photos from 29 respondents showed an average CER of 42.24% without preprocessing compared to 42.57% with preprocessing, a difference of only 0.33%, indicating that Otsu's thresholding does not significantly reduce OCR accuracy and even improves recognition accuracy for the majority (58.6%) of respondents. The grading accuracy evaluation showed that SiPena's automatic grading decisions matched the lecturer's manual grading on 96.67% of the 210 answers tested, with an average Fuzzy Matching similarity score of 71.94 out of 100. These results indicate that the integration of CNN-based OCR and Fuzzy String Matching, supported by a manual override feature, can serve as a practical and sufficiently accurate tool for reducing the manual workload of grading handwritten essay answers. Future research could explore fine-tuning EasyOCR using a locally collected Indonesian handwriting dataset, automatic segmentation of answer regions, semantic-similarity-based grading (e.g., IndoBERT or sentence-transformers) as an alternative to Fuzzy String Matching, integration with existing Learning Management Systems, support for answer types other than essays, and asynchronous task processing to improve system responsiveness.

REFERENCES

[1] E. Apriani and N. Pratiwi, "Perancangan sistem pengenalan tulisan tangan pada jawaban esai menggunakan metode CNN-LSTM berbasis Android," METIK Jurnal, vol. 9, no. 2, pp. 385 –396, 2025, doi: 10.47002/metik.v9i2.1094.

[2] K. Sharada, W. Alghamdi, K. Karthika, A. H. Alawadi, N. Gulomova, and V. Vijayan, "Deep learning techniques for image recognition and object detection," E3S Web of Conferences, vol. 399, p. 04032, 2023, doi: 10.1051/e3sconf/202339904032.

[3] A. Younesi, M. Ansari, M. Fazli, A. Ejlali, M. Shafique, and J. Henkel, "A comprehensive survey of convolutions in deep learning: Applications, challenges, and future trends," IEEE Access, vol. 12, pp. 41180–41218, 2024, doi: 10.1109/ACCESS.2024.3376441.

[4] L. Huang, H. Tseng, C. Hsieh, and C. Yang, "Deep learning based text detection using ResNet for feature extraction," Multimedia Tools and Applications, vol. 82, pp. 46871 –46903, 2023, doi: 10.1007/s11042-023-15449-z.

[5] Y. Liu, Y. Wang, and H. Shi, "A convolutional recurrent neural-network-based machine learning for scene text recognition application," Symmetry, vol. 15, no. 4, p. 849, 2023, doi: 10.3390/sym15040849.

[6] JaidedAI, "EasyOCR: Ready-to-use OCR with 80+ supported languages" (Version 1.7.2) [Software], GitHub, 2024. https://github.com/JaidedAI/EasyOCR

[7] S. Faizullah, M. S. Ayub, S. Hussain, and M. A. Khan, "A survey of OCR in Arabic language: Applications, techniques, and challenges," Applied Sciences, vol. 13, no. 7, p. 4584, 2023, doi: 10.3390/app13074584.

[8] K. M. Nahar, I. Alsmadi, R. E. Al Mamlook, A. Nasayreh, H. Gharaibeh, A. S. Almuflih, and F. Alasim, "Recognition of Arabic air-written letters: Machine learning, convolutional neural networks, and optical character recognition (OCR) techniques," Sensors, vol. 23, no. 23, p. 9475, 2023, doi: 10.3390/s23239475.

[9] M. A. M. Salehudin et al., "Analysis of optical character recognition using EasyOCR under image degradation," Journal of Physics: Conference Series, vol. 2641, no. 1, 2023, doi: 10.1088/1742 -6596/2641/1/012001.

[10] Z. Yang, S. Zuo, Y. Zhou, J. He, and J. Shi, "A review of document binarization: Main techniques, new challenges, and trends," Electronics, vol. 13, no. 7, p. 1394, 2024, doi: 10.3390/electronics13071394.

[11] K. Maliński and K. Okarma, "Analysis of image preprocessing and binarization methods for OCR -based detection and classification of electronic integrated circuit labeling," Electronics, vol. 12, no. 11, p. 2449, 2023, doi: 10.3390/electronics12112449.

[12] OpenCV team, "Image thresholding," OpenCV 4.x Documentation, 2024. https://docs.opencv.org/4.x/d7/d4d/tutorial_py_thresholding.html

[13] D. T. Anggraeni and C. Wibawa, "Perbaikan citra tanda tangan digital menggunakan metode Otsu thresholding dan Sauvola," Jurnal Ilmiah Matrik, vol. 25, no. 1, pp. 28 –34, 2023, doi: 10.33557/jurnalmatrik.v25i1.2324.

[14] K. B. K. Malaga, K. L. Verdillo, and E. S. Pascual, "An enhancement of the Jaro -Winkler fuzzy searching algorithm applied in library search engine," Journal of Information Systems Engineering and Management, vol. 10, no. 28, pp. 649–660, 2025, doi: 10.52783/jisem.v10i28s.4369.

[15] N. A. Sekartaji and R. Arifudin, "Implementation of Raita algorithm in Manado -Indonesia translation application with text suggestion using Levenshtein distance algorithm," Recursive Journal of Informatics, vol. 2, no. 2, pp. 88–96, 2024, doi: 10.15294/rji.v2i2.73651.

[16] A. R. Lahitani, "Automated essay scoring menggunakan cosine similarity pada penilaian esai multi soal," Jurnal Kajian Ilmiah, vol. 22, no. 2, pp. 107–118, 2022, doi: 10.31599/jki.v22i2.1121.

[17] D. T. Susetianingtias, D. A. Prasetya, and S. Handayani, "Pembentukan model recurrent neural network dan connectionist temporal classification pada pengenalan kata tulisan tangan offline," DECODE: Jurnal Pendidikan Teknologi Informasi, vol. 3, no. 2, pp. 172–183, 2023, doi: 10.51454/decode.v3i2.151.

[18] K. A. Pradani and L. H. Suadaa, "Automated essay scoring menggunakan semantic textual similarity berbasis Transformer untuk penilaian ujian esai," Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK), vol. 10, no. 6, pp. 1177–1184, 2023, doi: 10.25126/jtiik.2023107338.
[19] K. Apriana, I. M. A. D. Suarjaya, and N. K. D. Rusjayanthi, "Esscore: An OCR-based Android app for scoring short handwritten answer using Levenshtein distance," Journal of Applied Informatics and Computing, vol. 9, no. 4, pp. 1383–1393, 2025, doi: 10.30871/jaic.v9i4.9708.
[20] H. Misgna, B.-W. On, I. Lee, and G. S. Choi, "A survey on deep learning-based automated essay scoring and feedback generation," Artificial Intelligence Review, vol. 58, no. 2, pp. 1 –40, 2025.
[21] S. Ramírez, FastAPI Documentation (Version 0.111.0), Tiangolo, 2024. https://fastapi.tiangolo.com/
[22] C. Wibawa and D. T. Anggraeni, "Comparison of image segmentation method in image character extraction preprocessing using optical character recognition," Jurnal Teknik Informatika (JUTIF), vol. 4, no. 3, pp. 583–589, 2023, doi: 10.52436/1.jutif.2023.4 .3.956.
[23] B. K. Pattanayak, A. K. Biswal, S. R. Laha, S. Pattnaik, B. B. Dash, and S. S. Patra, "A novel technique for handwritten text recognition using easy OCR," in Proc. 2023 Int. Conf. Self Sustainable Artificia l Intelligence Systems (ICSSAS), 2023, pp. 1115–1119, doi: 10.1109/ICSSAS57918.2023.10331704.
[24] T. D. K, J. James, D. P. Gopinath, and M. Ashraf K, "Advocating character error rate for multilingual ASR evaluation," in Findings of the Association for Computational Linguistics: NAACL 2025, 2025, pp. 4941–4950.
[25] S. Izzat and N. N. Saleem, "Software testing techniques and tools: A review," Journal of Education and Science, vol. 32, no. 2, pp. 30–44, 2023, doi: 10.33899/edusj.2023.137480.1305.
[26] R. S. Pressman and B. R. Maxim, Software Engineering: A Practitioner's Approach, 9th ed. McGraw-Hill Education, 2024. Available: https://www.mheducation.com/highered/product/software -engineering-a-practitioners-approach-pressman.html
[27] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character region awareness for text detection," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), 2019, pp. 9365 -9374, doi: 10.1109/CVPR.2019.00959.