Ingénierie des Systèmes d'Information
Vol. 30, No. 9, September, 2025, pp. 2499-2509
Journal homepage: http://iieta.org/journals/isi

Accurate Detection of Handwriting Features Using Bounding Box Optimization

Muhammad Haviz Irfani[1,2], Samsuryadi[3*], Abdiansah[3], Rudi Heriansyah[2]

[1] Doctoral Program in Engineering Faculty of Engineering, Universitas Sriwijaya, Palembang 30139, Indonesia
[2] Faculty of Computer and Natural Science, Universitas Indo Global Mandiri, Palembang 30129, Indonesia
[3] Faculty of Computer Science, Universitas Sriwijaya, Palembang 30139, Indonesia

Corresponding Author Email: samsuryadi@unsri.ac.id

https://doi.org/10.18280/isi.300923

Received: 17 July 2025
Revised: 20 August 2025
Accepted: 15 September 2025
Available online: 30 September 2025

Keywords:
handwriting detection, Inter-word spacing, baseline variation, bounding box optimization, YOLOv8, personality analysis, hyperparameter tuning, deep learning

ABSTRACT

Handwriting is a unique human expression that provides insights into personality traits and behavioral tendencies. Automatic detection of handwriting features remains challenging due to variations in size, shape, writing style, and inconsistencies caused by environmental, physical, and psychological factors. These challenges are even greater in multi-class datasets. This study employed 150 handwriting images annotated with feature classes, including inter-word spacing (narrow-SKS, medium-SKN) and baseline variations (up-BN, down-BT, wavy-BG), representing key traits of written expression. The research highlights the role of parameter optimization through hyperparameter tuning, covering epochs, batch size, learning rate, momentum, image resolution, and weight decay. Such optimization is crucial to enhance detection accuracy and improve the robustness of handwriting feature recognition models. Among the ten experimental configurations evaluated, the proposed model in the fifth run achieved a recall of 0.98, demonstrating strong sensitivity in detecting handwriting features. However, this high recall was accompanied by relatively low precision, as reflected in the F1-Score of 0.55, indicating the presence of false positives. This trade-off highlights both the effectiveness and limitations of the current approach and underlines the importance of hyperparameter optimization, bounding box configuration, and dataset structuring in improving handwriting feature detection outcomes.

1. INTRODUCTION

Deep learning-based object detection techniques have developed rapidly and are now widely applied in various computer vision fields, including facial recognition, medical diagnostics, and intelligent transportation systems. Among these methods, the You Only Look Once (YOLO) family of models has emerged as a leading approach, offering advantages in real-time performance, computational efficiency, and detection accuracy. However, while YOLO has been extensively applied for general object detection tasks, its application to handwriting analysis remains relatively underexplored, particularly in capturing structural characteristics such as inter-word spacing, baseline orientation, and consistency of handwritten forms [1, 2].

Handwriting pattern recognition to determine individual personality remains an important field, as handwriting often reflects distinctive cognitive and psychological traits. It provides valuable information that can be used in contexts such as employee selection, student admission, and performance evaluation. Structural handwriting features such as spacing between words, spacing between lines, baselines, and slant constitute important indicators of individuality. These characteristics can reveal underlying aspects of personality and behavior, making their accurate detection crucial for both academic research and practical applications [3].

Feature extraction plays a central role in handwriting recognition, with overall accuracy highly dependent on the quality of extracted features and classification methods. For instance, the Celled Projection (CP) method divides Bangla handwritten images into smaller cells and computes pixel projections as feature vectors, thereby simplifying preprocessing and improving speed. Experiments with k-Nearest Neighbors, Probabilistic Neural Network, and Feed-Forward Backpropagation Neural Network demonstrated that CP (4 horizontal & 4 vertical cells) achieved an accuracy of 94.12%, outperforming or matching other conventional techniques such as crossing, moments, zoning, and projection histograms [4]. Despite these advances, such handcrafted methods struggle to handle diverse handwriting variations, highlighting the need for adaptive deep learning solutions.

Deep learning-based approaches, particularly Convolutional Neural Networks (CNNs), have set a new standard in character recognition due to their ability to automatically extract features without manual design. CNNs excel at learning complex spatial patterns in handwriting through convolution and pooling layers, and they adapt well to variations in size, shape, and environmental conditions. Recent studies have shown that YOLO and its derivative architectures can deliver superior performance in detecting and classifying handwritten characters, words, and spatial

features compared to traditional methods [5]. Yet, previous work often focused on character-level recognition, leaving structural handwriting features insufficiently explored. In addition, hyperparameter tuning and dataset expansion remain important aspects that can significantly improve detection accuracy, but existing research often lacks a comprehensive treatment of these factors [6-8].

Recent works have begun to highlight the potential of YOLO for handwriting-related tasks. For example, the YOLOv5-HDR model [9] improved detection speed and accuracy compared to earlier versions, while YOLOv8 has been fine-tuned for forensic hand image identification, outperforming DETR and other variants [10]. Similarly, YOLOv7-PDM was designed to capture stroke-level details in calligraphy, surpassing YOLOv6 and YOLOv8, while another study combined YOLOv8 with EfficientNet-b4 for Bangla OCR to achieve robust performance on complex scripts [11]. Nevertheless, the use of YOLO to capture fine-grained structural handwriting features-such as inter-word spacing, baseline variation, and slant-remains insufficiently addressed. To fill this gap, the present study adapts YOLOv8 to extract spatial handwriting features from a multi-class annotated dataset. This approach not only extends YOLO's utility in handwriting analysis but also provides a flexible framework for comprehensive evaluation of writing patterns, with implications for forensic science, psychological assessment, and handwriting-based biometric applications.

2. MATERIAL AND METHOD

2.1 Research framework

This study aims to develop a handwriting image detection model that accommodates two offline writing formats. The proposed approach focuses on generating bounding box annotations for key handwriting feature classes, including inter-word spacing, baseline variations, and inter-line spacing. To achieve this, a Convolutional Neural Network (CNN)-based object detection method was employed, utilizing the YOLOv8 framework for efficient and accurate feature extraction (Figure 1).

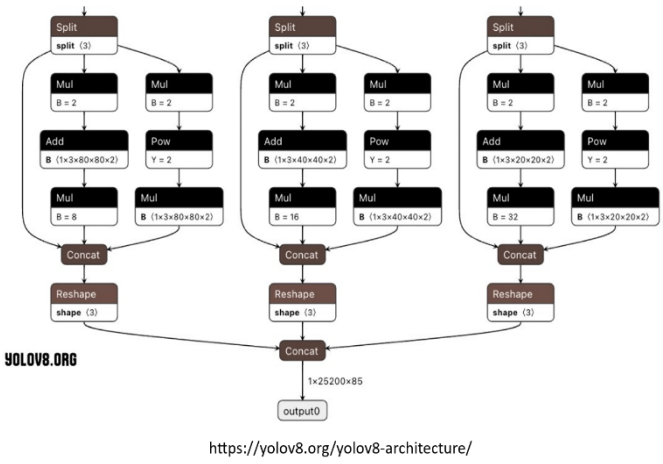


Figure 1. The three main components of the YOLOv8 architecture

The following is a research framework that can be used in the sustainability of the results of this study later (Figure 2).

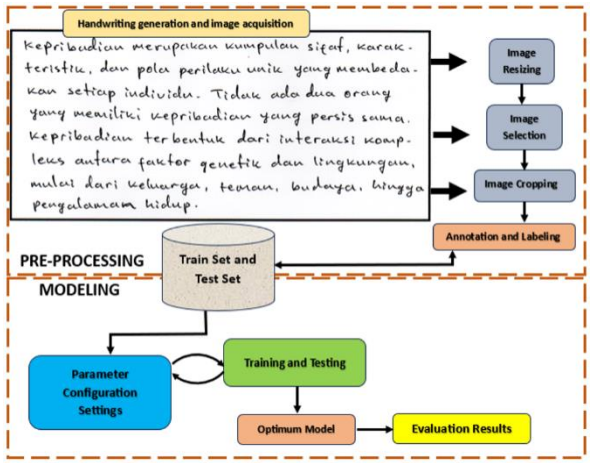


Figure 2. Workflow of handwriting image preprocessing, modeling, and evaluation

Figure 2 presents the overall research workflow, beginning with handwriting generation and image acquisition through the collection of handwritten samples in two distinct formats. The acquired images are subsequently subjected to selection and resizing to maintain consistent dimensions across the dataset. Following this step, image cropping is conducted-either at the paragraph or line level-to isolate relevant regions for further analysis. The cropped images are then spatially annotated and labeled according to predefined feature categories, forming the basis for the training phase.

In the next stage, the annotated data are organized into training and testing sets to support the learning and evaluation processes. Hyperparameter configurations-including learning rate, momentum, image resolution, and weight decay-are then defined to guide the training process. The model is iteratively trained and tested until an optimal configuration is achieved, ensuring the development of a reliable detection model.

Finally, model performance is evaluated using standard metrics such as mean Average Precision (mAP), Precision, Recall, and F1-Score. This evaluation is further complemented by visual tools, including convergence curves, confusion matrices, and error distribution analyses, to provide deeper insights into model performance. Collectively, this workflow represents a systematic and replicable approach that can be applied to handwriting detection and analysis research.

2.2 Data collection

This study employed primary handwritten image data collected under standardized conditions to ensure consistency and reliability. The writing sessions were conducted in a controlled indoor environment with stable room temperature (approximately 24 ± 1°C) and adequate illumination (around 400 lux), following recommended conditions for handwriting analysis [6, 7, 12]. All participants used identical writing materials, including A4 paper (80 g), a ballpoint pen with a 0.7 mm tip, and a flat writing surface. Such controls were applied to minimize external variability that could affect stroke clarity, line spacing, and baseline consistency.

Handwriting samples consisted of sentences and paragraphs written in Indonesian to capture natural writing flow and contextual variations. The format was standardized so that each subject produced continuous text paragraphs rather than isolated lines, allowing for the extraction of structural features such as inter-word spacing, line spacing, and baseline

orientation. This procedure follows previous studies that highlight the influence of writing format (Figure 3) on stroke direction, inter-letter connections, and fluency of hand movement [8]. The dataset collected under these conditions thus provides a reliable foundation for subsequent spatial annotation and feature detection.

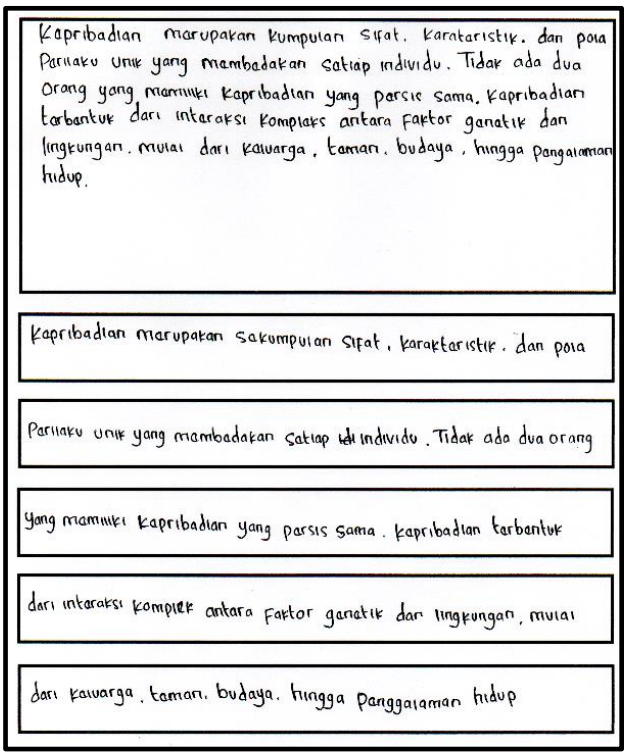


Figure 3. Examples of image data formats used

2.3 Cropping images after acquisition

The image of the text was acquired using an Epson 3250 printer scanner device with 300 dpi to obtain a PNG (Portable Network Graphics) unit image. Reducing the dimensions of an image can be done using a technique called cropping, which involves cutting the image at certain coordinates in a specific area of the image. In this process, a new object is obtained, which is the result of cutting the original image or part of the image to the desired size. Cropping is an effective method for resizing and focusing on certain parts of an image, allowing for adjusting the composition and taking only relevant or interesting parts [13, 14]. This process produces an object or part of an image with a predetermined size.

2.4 Annotation and labeling

Annotation is the process of marking or labeling certain parts of an image to indicate important objects or features. Labeling is the process of providing category labels to areas that have been annotated [15, 16]. The results of annotation and labeling are in the form of files in Json extension units (Figure 4).

Annotation is the process of marking or labeling certain parts of an image to indicate important objects or features. Labeling is the process of assigning category labels to areas that have been annotated [16, 17]. This stage uses the LabelMe application offline, which is done per image (with a .png extension); the results of the annotation and labeling are in the form of files with a JSON extension (Figure 5). In the original image, annotation is carried out using bounding boxes of the rectangle and polygon type, as many as five shapes, namely narrow interword-spacing (SKS), medium interword-spacing (SKN), baseline-up (BN), baseline-down (BT), and wavy-baseline (BG).

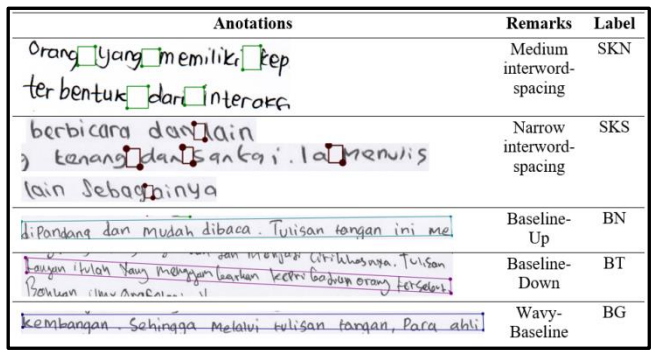


Figure 4. Image output results for 5 labels in two text formats

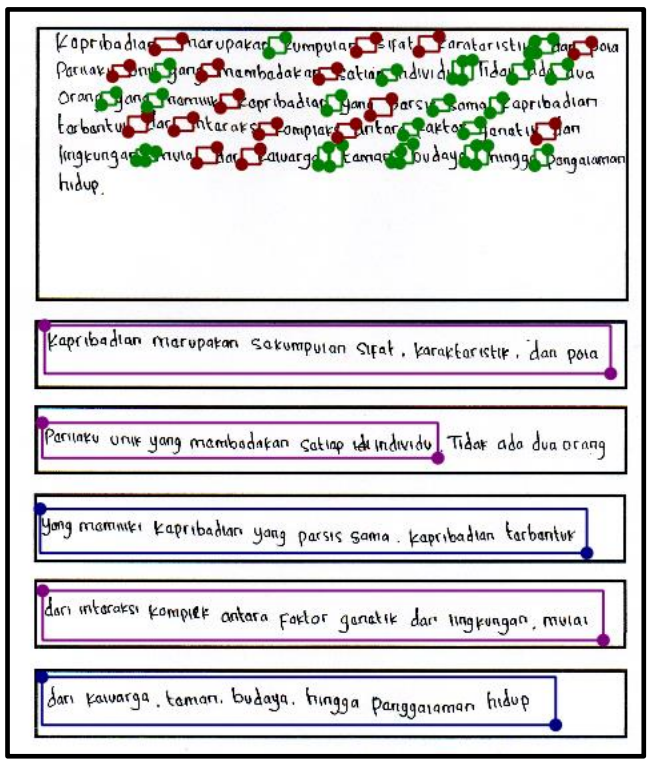


Figure 5. Image after annotation

Interword-spacing distance, as defined in Eq. (1).

$$S_i = L_{i+1} - (x_i + w_i) \quad (1)$$

The variable x_i denotes the x-coordinate of the left edge of the i-th (i = 1, 2, ..., N-1) bounding box, while $w_i$represents its width. Consequently, the right edge of the i-th bounding box is expressed as $R_i = x_i + w_i$. The left edge of the subsequent bounding box, namely the (i+1)-th box, is denoted as $L_{i+1} = x_{i+1}$. Based on these definitions, the horizontal spacing between two consecutive bounding boxes is represented as S_i. Finally, N indicates the total number of bounding boxes contained within a single handwriting line. These notations collectively serve as the foundation for quantifying and analyzing inter-word spacing patterns in handwritten text.

The space bounding box is formulated as defined in Eq. (2) [14].

$$SpaceBox_i = [R_i, y_i, S_i, max(h_i, h_{i+1})] \quad (2)$$

where R_i is the right edge coordinate, y_i the vertical position, and S_i the computed spacing width. The heights h_i *and* h_{i+1} represent adjacent bounding boxes, with $\max(h_i, h_{i+1})$ ensuring alignment to the taller box. This compact formulation effectively represents inter-word gaps as bounding boxes.

Spacing Classification Based on Thresholds as Eq. (3).

$$label(S_i) = \begin{cases} Narrow, S_i \leq 24 \\ Medium, S_i \geq 32 \end{cases} \quad (3)$$

Mathematical Formulation for Baseline Extraction as defined in Eqs. (4) to (9).

Horizontal Projection Profile:

$$P(y) = \sum_{x=1}^{W} BW(x, y) \quad (4)$$

The equation computes the horizontal projection profile by summing binary pixel values across each row y. The resulting projection profile *P(y)* is used to detect text regions along the horizontal axis.

Thresholding for Line Segmentation:

$$Line(y) = \begin{cases} 1, if P(y) > \theta \\ 0, otherwise \end{cases} \quad (5)$$

where $\theta = 0.1 \times \max(P(y))$. This criterion determines whether a row contains handwriting. A row is considered part of a text line if its projection value *P(y)* exceeds the threshold θ which is set adaptively to 10% of the maximum projection value.

Baseline Point Extraction:

$$P_x = x, P_y = maxy \mid BW(x, y) = 1 \quad (6)$$

This formula extracts baseline points by identifying the lowest handwritten pixel in each column x. The set of points (P_x, P_y) collectively represents the baseline structure of the handwriting.

Polynomial Regression:

$$y(x) \approx p_1.x^2 + p_2.x + p_3 \quad (7)$$

The baseline is approximated using second-degree polynomial regression. The coefficients p_1, p_2, p_3 are estimated using the least squares method, producing a smooth curve that models the baseline trajectory of the handwriting.

Baseline Classification:

$$A = \max(y(x)) - \min(y(x)) \quad (8)$$

$$\Delta = y(x_{end}) - y(x_{start}) \quad (9)$$

A denotes the amplitude, which represents the maximum curvature of the baseline, while Δ denotes the vertical displacement of the baseline from the beginning to the end of the text.

Based on the values of A and Δ in Eqs. (8) and (9) respectively, the baseline is classified into three categories:

1. Wavy if A > 15
2. Rising if Δ < −3
3. Falling if (Δ ≥ −3) and (A ≤ 15).

2.5 YOLOv8n architecture for handwriting detection

The detection process in this study for lines of text in handwritten paragraphs consists of two main stages: training and testing using the YOLOv8n framework [5, 18-20]. The training dataset undergoes preprocessing, augmentation, and normalization, while the testing stage evaluates the model's performance using unseen data. The model was trained using a system equipped with an NVIDIA Tesla T4 GPU (CUDA:0, 15095MiB), running Python 3.11.13 with PyTorch 2.6.0+cu124. The YOLOv8n model was trained with hyperparameters optimized to balance detection accuracy and inference speed.

2.6 Metrics

To evaluate the overall effectiveness of the proposed model, the Mean Average Precision (mAP) metric is employed. mAP is widely recognized as a standard measure for assessing the quality of object detection systems, as it reflects the model's capability to accurately identify and localize objects across different classes [5, 9, 20, 21]. Specifically, mAP is obtained by averaging the precision scores computed for each class, thereby providing a comprehensive view of detection performance in a given task. The computation of mAP begins with the calculation of Average Precision (AP), which integrates both precision and recall values across multiple threshold levels [22]. The mathematical formulations for AP and mAP are presented in Eqs. (10) and (11).

$$AP = \sum_{k} (precision\ at\ recall\ point\ k * \Delta recall_k) \quad (10)$$

where Precision at recall point *k*, which the precision value at a specific recall.

mAP50 (*Mean Average Precision*) in Eq. (11) [22]:

$$mAP = \frac{1}{N} . \sum_{i=1}^{N} AP_i \quad (11)$$

where *mAP* is the mean of the average precision values across all classes, and AP_i is the average precision for the *i*-th class. To calculate precision and recall, use Eqs. (12) and (13).

The *precision* is calculated by the following Eq. (12):

$$P = \frac{TP}{TP + FP} \quad (12)$$

where *P* denotes precision, calculated as the ratio of true positives (*TP*) to the sum of true positives and false positives (*TP*+*FP*).

As for the *recall* is solved by:

$$R = \frac{TP}{TP + FN} \quad (13)$$

where, *R* denotes recall, calculated as the ratio of true positives to the sum of true positives and false negatives (TP+FN).

True positives (TP) are the number of objects in an image that the model correctly identifies. False positives (FP) occur when the network detects an object in an image when it is not present. False negatives (FN) occur when the network fails to detect an object in an image. Precision measures the number of correctly identified positive cases out of all predicted positive cases, and its value decreases as the number of false positives increases. Recall measures the number of correctly identified positive cases out of all actual positive cases and indicates the extent to which false negatives impact model performance.

3. RESULTS AND DISCUSSION

3.1 Experiment configuration before training dataset

This research produces a handwriting line distance detection model using Convolutional Neural Network (CNN) [17, 23, 24], which is implemented in the YOLO framework version 8 [18, 25-27]. The data used are 150 images of student handwriting with various writing styles, with a train and test data division of 80:20 or 120 for the training dataset and 30 for the test/validation dataset. Meanwhile, the experiment was carried out repeatedly using reading instances of each class in the range of 435 to 934 with the number of epochs of 80, the optimizer using AdamW, a learning rate of 0.001, and the number of batches equal to 4. Experiments were conducted using GPUs as an important component in various computationally intensive applications such as machine learning, scientific computing, and AI-based model inference. This research is GPU-based to ensure optimal stability and performance, with GPU conditions before the experiment in this study, nvidia-smi showed the system using the following configuration: NVIDIA driver version 550.54.15, CUDA version 12.4, the detected GPU is Tesla T4, a server-class GPU with a Turing architecture designed for AI inference and parallel computing.

Experiments were conducted on an NVIDIA Tesla T4 GPU (CUDA 12.4, driver version 550.54.15), a server-class GPU optimized for AI inference and parallel computing. The GPU was idle and ready before training, ensuring stable performance and no interference from other processes. This configuration provided sufficient computational power for model training and evaluation while maintaining consistency across all experimental runs [28].

3.2 Spacing between words and sentence baseline

Figure 5 shows the annotation results using bounding box rectangles and polygons to label handwriting images. Lines with different colors indicate different labels for each color. This annotation serves two purposes: (1) line segmentation to facilitate horizontal text separation, thereby improving OCR performance [19-25] and (2) individual line extraction for detailed textual analysis. In addition, examination of interword spacing and sentence baselines provides fundamental knowledge for further studies, including personality classification. As seen in Figure 6, interword spacing in handwriting images is categorized into three classes: narrow, medium, based on the spatial characteristics between adjacent words. Meanwhile, sentence baselines are also categorized into 3 classes: rising, falling, and wavy.

(a) Width of the spacing limit between words

(b) Degree of slope of sentence lines

Figure 6. Output of word spacing and baseline with MATLAB

3.3 Model detection

This study uses a backbone convolutional network based on the C2f module (one-stage detector) to detect bounding boxes and predict classes directly from handwriting component feature maps, by fine-tuning the model on a relevant handwriting dataset, with smaller entities being crucial to improve the accuracy of handwriting feature detection. Various configurations have been performed for experiments and running results in obtaining the previous detection model, and in this section, only the best configuration of the metric value search results from the various configurations that have been performed.

Table 1. Hyperparameter setting

No.	Configurations	Hyperparameter
1	learning rate	0.001
2	image size	640
3	momentum	[0.72, 0.73]
4	weight decay	[0.0006, 0.0007, 0.0008, 0.0009, 0.0010]

Table 1 presents the hyperparameter configurations used in the model training process. The parameters shown include the learning rate, image size, batch size, momentum value, and weight decay. The learning rate was set at 0.001 to control the speed of weight updates during the training process. The image size used was 640 pixels, a common resolution for object detection models, to maintain a balance between image detail and computational speed. The momentum value used ranged from 0.72 to 0.73, accelerating convergence by preserving the direction of the gradient.

Furthermore, the weight decay used varied between 0.0006 and 0.0010, aiming to reduce overfitting through regularization of the model weights. The selection of a range of weight decay values allows for the search for optimal values that improve model generalization on test data. This variation in hyperparameters indicates that the training process employed a hyperparameter tuning approach to find the most

effective combination for improving model performance. This approach is crucial in machine learning-based research because it can significantly impact the accuracy and stability of training results.

3.4 Model optimization for handwriting detection

Table 2 shows the experimental results for ten model training trials at epoch 80, using an image size of 640 pixels, a batch size of 4, the AdamW optimizer, and a learning rate of 0.001. The parameters measured included processing speed (Preprocess Speed and Postprocess Speed), mAP50 values per class (BT, BN, BG, SKS, SKN), average mAP50 (All), and the momentum and weight decay hyperparameters used in each trial.

Table 2. Experimental results for some good models epoch 80, imgsz 640px, batch=4, optimizer AdamW, and lr= 0.001

run	Speed Pre-Process	Speed Post-Process	mAP50					Momentum	WD
			BT	BN	BG	SKS	SKN		
1	0.2	4.0	0.42	0.42	0.5	0.54	0.64	0.51	0.72
2	0.5	4.7	0.45	0.32	0.48	0.59	0.57	0.51	0.72
3	0.2	4.1	0.39	0.40	0.47	0.56	0.68	0.50	0.72
4	0.2	4.6	0.43	0.35	0.47	0.58	0.67	0.53	0.72
5	0.9	4.6	0.41	0.50	0.48	0.62	0.71	0.54	0.72
6	0.3	4.0	0.41	0.49	0.47	0.54	0.73	0.53	0.73
7	0.2	4.7	0.42	0.43	0.52	0.54	0.67	0.52	0.73
8	0.3	2.4	0.43	0.47	0.48	0.55	0.72	0.53	0.73
9	0.7	3.4	0.41	0.50	0.55	0.56	0.69	0.54	0.73
10	0.2	2.4	0.46	0.46	0.48	0.57	0.69	0.53	0.73

Table 2 shows that the highest average mAP50 value (All) was obtained in the fifth trial, at 0.544, with a weight decay (WD) of 0.0010 and a momentum of 0.72. This trial also had relatively high mAP50 values per class, particularly for the SKS (0.62) and SKN (0.71) classes. Furthermore, the preprocessing speed in trial 5 was the highest (0.9 seconds), while the postprocessing remained in the general range of 2.4-4.7 seconds. Scientifically, these results show that variations in weight decay and momentum can significantly affect detection performance, and that the model with the highest mAP50 value does not always have the fastest processing time.

Table 3 lists down the performance evaluation results of ten models based on four metrics. Overall, recall values showed high consistency, ranging from 0.97 to 0.98, indicating that all models were able to detect most relevant objects. Precision values varied significantly, with the highest value in model 5 (0.922) and the lowest in model 4 (0.715), which impacted the F1-Score and mAP50 values for each model. Model 5 stood out with the highest mAP50 (0.544), the highest Precision (0.922), and the highest F1-Score (0.55), making it the most balanced model among all models. Conversely, models 2 and 4 showed relatively low Precision, although Recall remained high, indicating a tendency to produce more false positive predictions. This data can be used as a reference for selecting the best model and identifying areas for further optimization.

Table 3. Table evaluation metrics

Model	mAP50	Precision	F1-Score	Recall
1	0.512	0.895	0.51	0.97
2	0.511	0.733	0.53	0.97
3	0.524	0.895	0.51	0.97
4	0.531	0.715	0.53	0.97
5	0.544	0.922	0.55	0.98
6	0.526	0.849	0.51	0.98
7	0.515	0.812	0.51	0.97
8	0.530	0.845	0.51	0.98
9	0.542	0.906	0.52	0.97
10	0.532	0.810	0.51	0.97

As shown in Tables 2 and 3, the best performing configuration was obtained in run 5, which yielded the highest mAP50 (0.544), Precision (0.922), F1-Score (0.55), and Recall (0.98). To statistically validate these differences, we performed one-way ANOVA across all model runs followed by pairwise Welch's t-tests with Bonferroni correction, using multiple repetitions of each configuration. The ANOVA results indicated significant differences among models for all evaluation metrics (mAP50: $F = 81.49$, $p < 0.001$; Precision: $F = 69.64$, $p < 0.001$; F1-Score: $F = 11.14$, $p < 0.001$; Recall: $F = 7.34$, $p < 0.001$). Post-hoc analysis confirmed that run 5 was significantly superior ($p < 0.05$) compared to most other configurations in terms of mAP50, Precision, and F1-Score, while no significant difference was found when compared with run 9. These statistical results strengthen the reliability of our conclusion that run 5 provides the most stable and effective configuration for handwriting line-distance detection.

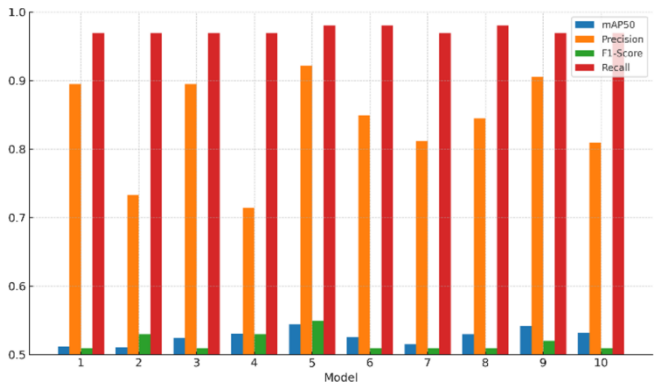


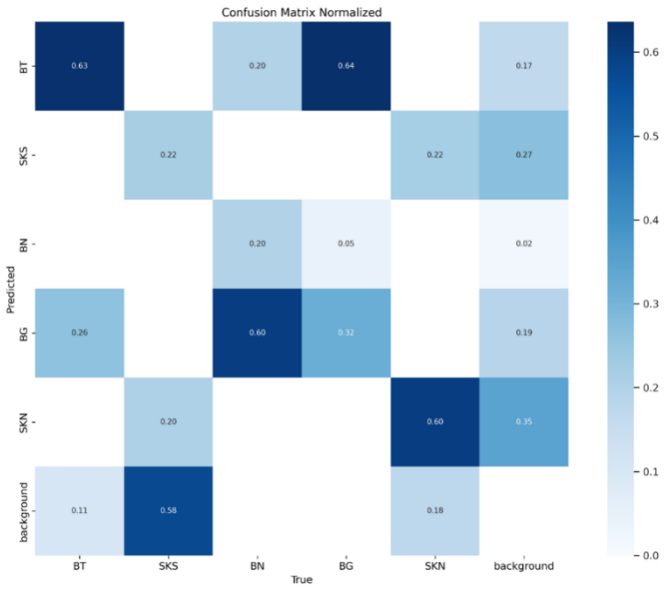
Figure 7. Performance analysis of 10 models

Figure 7 shows a bar graph comparing the values of four evaluation metrics: mAP50, Precision, F1-Score, and Recall, across the ten tested models. This visualization shows that Recall values range from 0.97 to 0.98 across all models, indicating a consistent ability to detect relevant objects. Precision values show greater variation, with a peak at 0.922 in the fifth model and a low of 0.715 in the fourth model. Meanwhile, mAP50 and F1-Score tend to fall within a

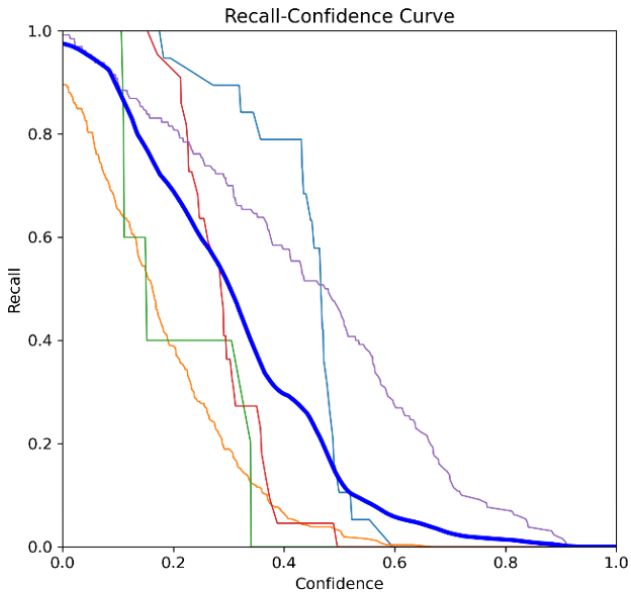
relatively narrow range (around 0.51 - 0.55), indicating stable performance in terms of precision and detection coverage.

In general, this graph pattern indicates that most models perform well in Recall, but there are significant differences in Precision, which impacts the F1-Score and mAP50 values. Model 5 stands out with its combination of high Precision, optimal Recall, and the highest F1-Score among all models, making it a candidate for the best model in this test. Meanwhile, the fourth and second models exhibit relatively low precision, which may result in an increased number of false positive predictions. This analysis suggests that optimizing hyperparameters, particularly those affecting precision, may be key to improving the model's overall performance.

3.5 Best model selection based on multi-parameter evaluation



(a) Confusion matrix normalized



(b) Confidence curve

Figure 8. Training result

Figure 8(a) presents the confusion matrix used to evaluate the performance of the YOLOv8 model in detecting different handwriting categories. Darker colors on the main diagonal indicate a high number of true positives for each class. Red circles mark off-diagonal cells, which represent misclassifications between classes, indicating that the model still has difficulty distinguishing between categories with similar visual features. The relatively high values in the diagonal cells compared to the off-diagonal values indicate that the model has good generalization ability, although there is still room for improvement in certain classes.

Figure 8(b) shows the Recall-Confidence curve, which shows the relationship between recall and confidence thresholds for each class. This curve illustrates that at low confidence levels, the model maintains high recall, but experiences a significant decline as the confidence threshold is increased. The thick blue line represents the combined performance across all classes, with an area under the curve (AUC) of 0.98 at a confidence threshold of 0.000, indicating excellent detection performance at maximum sensitivity.

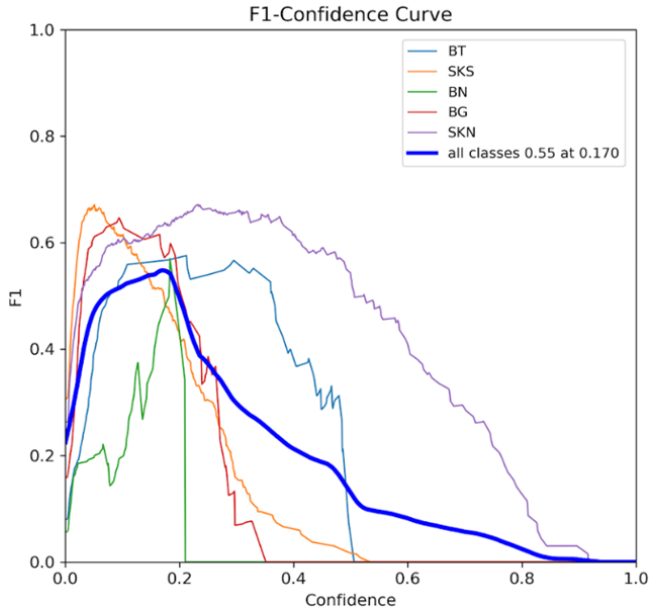


Figure 9. F1-Confidence curve for handwriting feature classification

Figure 9 presents the F1-Confidence curve, which assesses the classification model's performance across different confidence thresholds for both individual classes and the overall dataset. Each class demonstrates a distinct curve, highlighting variations in peak F1 scores and corresponding optimal confidence values. The aggregated curve for all classes reaches its maximum F1 score of approximately 0.55 at a confidence threshold of 0.170. This evaluation provides valuable insights into selecting an appropriate confidence threshold to achieve an optimal trade-off between precision and recall in handwriting feature detection.

Figure 10 illustrates the ground truth annotations generated for validation batch 0, showing the visual labeling of handwritten survey document images. Each textual element is enclosed by a color-coded bounding box that corresponds to a specific label class. The annotated objects include handwritten text in paragraph sections, input fields, and respondent signatures. These labels are categorized into distinct classes such as SKS, SKN, BN, BT, and BG, with bounding box colors indicating class differentiation. The annotated dataset serves as the reference ground truth for model evaluation, enabling performance comparison against predictions and

facilitating analysis of spatial distributions, dimensions, and structural characteristics of handwritten elements within the documents.

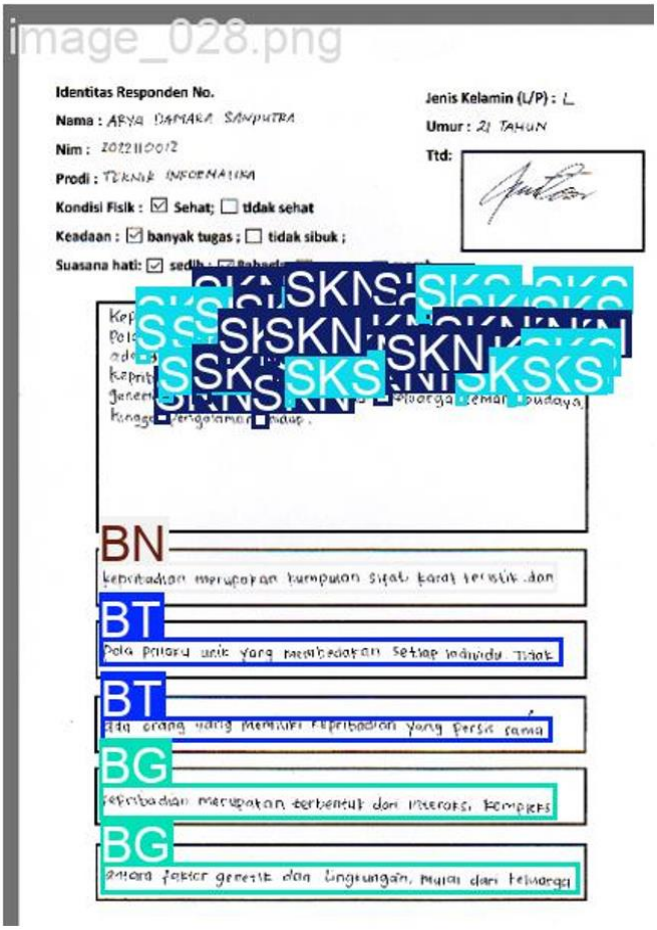


Figure 10. Ground truth annotations of handwritten text

Based on these two visualizations, it can be concluded that the YOLOv8 model used in this study shows promising handwriting detection performance, with high accuracy on most classes and the ability to maintain good recall over a wide range of confidence values.

This annotation process is performed consistently across the entire dataset to ensure uniformity and accuracy in training and validating deep learning-based object detection models. Labels are placed directly within the bounding boxes to facilitate visual identification during model performance evaluation. This stage is a crucial part of data preprocessing and validation, particularly in handwriting analysis research, which involves extracting information from physical documents and classifying text elements. The existence of accurate ground truth allows for precise measurement of evaluation metrics such as precision, recall, and F1-Score, thus helping to ensure that the model is able to recognize and classify text elements according to predefined categories.

Figure 11 specifically shows the prediction results of the YOLO model trained for handwritten object detection. The numbers in the figure indicate how confident the YOLO model is in identifying objects within the boxes, such as SKN, SKS, BT, and BG. The batch validation process is a stage in the machine learning model development cycle where the model is tested on a new data set (batch) that it has never seen during training to evaluate its predictive performance.

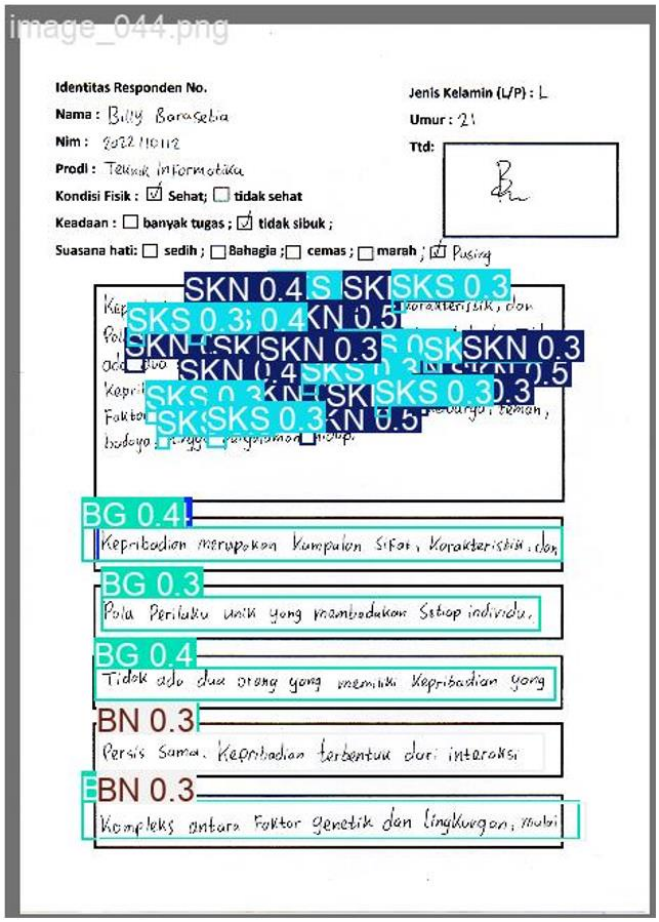


Figure 11. Validation results for batch 0 predictions

Furthermore, the model successfully detected object classes with varying degrees of confidence. The figures show how the model identified the location (bounding box) and type (class label) of each text element analyzed in a validation batch.

3.6 Comparative review with previous studies

Figure 12 presents a comparative analysis of evaluation metrics reported in recent YOLO-based handwriting detection studies. While previous works have shown that YOLO and its variants [11, 19, 20, 29] - particularly YOLOv8- are widely applied across diverse handwriting contexts, their performance varies substantially depending on dataset complexity, object scale, and annotation quality. For example, Zhang and Shi [19] applied YOLOv8 for handwritten exam sheet detection but obtained relatively low scores (P = 4.3%, R = 3%, mAP@0.5 = 4.6%). These results suggest that heterogeneous exam sheets with cluttered backgrounds and variable handwriting styles increased the detection difficulty, while limited hyperparameter tuning may have further constrained performance. In contrast, Maung et al. [11] achieved very high precision (P = 0.97) in Bangla handwriting detection. This can be attributed to consistent dataset preparation and the structured grapheme-rich script, which allowed the model to leverage stroke-level distinctions. However, the lack of reported mAP limits a full comparison. Similarly, Schreurs [29] demonstrated balanced performance on the MNIST dataset (Precision, Recall, and F1-Score ≈ 0.96). The superior results are likely due to MNIST's controlled environment-clean, centered, and low-noise digit samples-where object detection is inherently less challenging

compared to natural handwriting with irregular spacing and baselines.

Meanwhile, Guo et al. [20] focused on small-scale signatures embedded in book pages and reported low performance (P = 2.2%, R = 8.6%, mAP@0.5 = 3.9%). The difficulty stemmed from detecting dense and tiny objects in complex page layouts, where overlapping strokes and background clutter reduced the effectiveness of anchor-based YOLO detection. Similarly, Singh and Khare [30] encountered performance degradation when applying CNN-based feature extraction on spiral drawings for Parkinson's disease detection, emphasizing how fine-grained handwriting-like structures challenge conventional deep-learning models. Comparable challenges have also been noted in other image-based detection domains, such as medical imaging, where deep learning architectures are used to identify subtle lesion patterns in complex visual backgrounds [31].

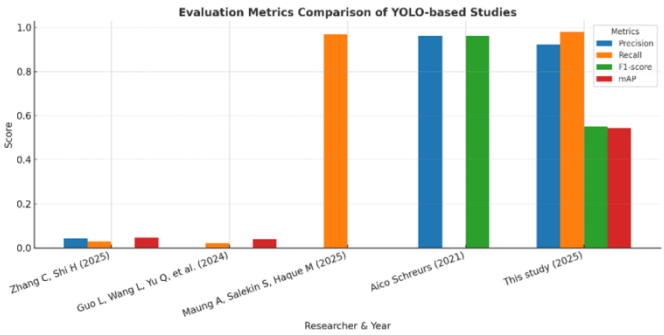


Figure 12. Comparative evaluation metrics of YOLO-based for previous handwriting detection research

By comparison, this study (2025) achieved strong overall performance (Precision = 0.922, Recall = 0.98, F1-Score = 0.55, mAP@0.5 = 0.544), despite addressing a more challenging task-interword spacing and baseline variation detection in unconstrained handwriting. The improvement can be linked to (i) careful annotation protocols, (ii) multi-class structuring of handwriting features, and (iii) systematic hyperparameter optimization. These factors reduced noise in feature boundaries and enhanced generalization. Importantly, the originality of this study lies in extending YOLOv8 beyond character recognition toward handwriting structural features for personality analysis, bridging technical object detection with behavioral interpretation. Such integration highlights the broader applicability of YOLO-based models in forensic document analysis and psychological handwriting studies.

4. CONCLUSIONS

This study successfully applied the YOLOv8 model for handwritten text detection, enabling accurate identification and labeling of inter-word spacing and baseline classes. The optimized model demonstrated reliable performance, as reflected in the mean Average Precision (mAP) values across all categories. Specifically, the classes of narrow inter-word spacing, normal/wide inter-word spacing, rising baseline, falling baseline, and wavy baseline were detected with high confidence, underscoring the robustness of the trained model. The systematic approach involving dataset preparation, manual annotation, hyperparameter tuning, and model training contributed significantly to these outcomes.

Among the ten experimental configurations, the fifth run achieved a recall of 0.98, an F1-Score of 0.55, and an mAP@0.50 of 0.544, indicating consistent and reliable detection of inter-word spacing and baseline variations in handwritten text. In addition, the precision-recall curves confirmed performance stability across different confidence thresholds, validating the model's capacity to maintain detection quality under varying conditions. These findings advance handwriting analysis research, providing valuable implications for personality identification, writing habit assessment, and cognitive status evaluation. Future work should focus on expanding to more diverse and multiclass handwriting datasets, leveraging advanced model architectures, and incorporating real-time processing capabilities to further enhance evaluation metrics and practical applicability.

However, several limitations should be acknowledged. First, the relatively low F1-Score indicates that the model, while highly sensitive (high recall), still produces false positives, reducing overall precision. Second, the model shows constraints in generalizing across complex handwriting conditions, such as cursive, overlapping strokes, and inconsistent line structures, which are common in natural writing scenarios. Third, cross-language applicability remains limited since this study focused only on Indonesian handwriting; applying the model to scripts with different grapheme structures (e.g., Arabic, Chinese, Bangla) may require substantial adaptation and retraining.

Future work should address these limitations by exploring advanced architectures that better balance recall and precision, expanding datasets to include multilingual and more heterogeneous handwriting samples, and integrating post-processing strategies to minimize false detections. These steps will help strengthen the model's generalizability and practical relevance in real-world applications, such as forensic handwriting analysis, psychological assessment, and document authentication.

ACKNOWLEDGMENT

This work was funded by the Directorate of Research, Technology and Community Service (DRTPM) of the Ministry of Education, Culture, Research and Technology of the Republic of Indonesia through the 2024 Doctoral Dissertation Research Scheme, with grant funding from decision letter number 0459/E5/PG.02.00/2024 and contract agreement number 090/E5/PG.02.00.PL/2024.

REFERENCES

[1] Abioye, S.O., Oyedele, L.O., Akanbi, L., Ajayi, A., et al. (2021). Artificial intelligence in the construction industry: A review of present status, opportunities and future challenges. Journal of Building Engineering, 44: 103299. https://doi.org/10.1016/j.jobe.2021.103299

[2] Kamran, I., Naz, S., Razzak, I., Imran, M. (2021). Handwriting dynamics assessment using deep neural network for early identification of Parkinson's disease. Future Generation Computer Systems, 117: 234-244. https://doi.org/10.1016/j.future.2020.11.020

[3] Samsuryadi, R.K., Mohamad, F.S. (2021). Automated handwriting analysis based on pattern recognition: A

survey. Indonesian Journal of Electrical Engineering and Computer Science, 22(1): 196-206. https://doi.org/10.11591/ijeecs.v22.i1

[4] Hossain, M.Z., Amin, M.A., Yan, H. (2012). Rapid feature extraction for optical character recognition. arXiv preprint arXiv:1206.0238. https://doi.org/10.48550/arXiv.1206.0238

[5] Wang, H., Wang, L., Chen, H., Li, X., Zhang, X., Zhou, Y. (2023). Waste-YOLO: Towards high accuracy real-time abnormal waste detection in waste-to-energy power plant for production safety. Measurement Science and Technology, 35(1): 016001. https://doi.org/10.1088/1361-6501/ad042a

[6] Ali, S., Abdulrazzaq, M. (2023). A comprehensive overview of handwritten recognition techniques: A survey. Journal of Computer Science, 19: 569-587.

[7] Qi, H., Zhang, R., Wei, Z., Zhang, C., et al. (2023). A study of auxiliary screening for Alzheimer's disease based on handwriting characteristics. Frontiers in Aging Neuroscience, 15: 1117250.

[8] Fitjar, C.L., Rønneberg, V., Torrance, M. (2024). Assessing handwriting: A method for detailed analysis of letter-formation accuracy and fluency. Reading and Writing, 37(2): 291-327. https://doi.org/10.1007/s11145-022-10308-z

[9] Moustapha, M., Tasyurek, M., Ozturk, C. (2023). A novel yolov5 deep learning model for handwriting detection and recognition. International Journal on Artificial Intelligence Tools, 32(4): 2350016. https://doi.org/10.1142/S0218213023500161

[10] Nguyen, T.T., Wilson, C., Khan, I., Dalins, J. (2024). Object detection approaches to identifying hand images with high forensic values. In 2024 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Kuching, Malaysia, pp. 2727-2734. https://doi.org/10.1109/SMC54092.2024.10831640

[11] Maung, A.T., Salekin, S., Haque, M.A. (2025). A hybrid approach to Bangla handwritten OCR: Combining YOLO and an advanced CNN. Discover Artificial Intelligence, 5(1): 119. https://doi.org/10.1007/s44163-025-00251-7

[12] Ahmed, R.M., Rashid, T.A., Fattah, P., Alsadoon, A., et al. (2022). Kurdish handwritten character recognition using deep learning techniques. Gene Expression Patterns, 46: 119278. https://doi.org/10.1016/j.gep.2022.119278

[13] Doermann, D., Tombre, K. (2014). Handbook of Document Image Processing and Recognition. Springer Publishing Company, Incorporated.

[14] Gonzales, R.C., Woods, R.E. (2018). Digital Image Processing, 4th edition. Pearson Education.

[15] Guruprasad, P., Sujith Kumar, S., Vigneswaran, C., Chakravarthy, V.S. (2021). An end-to-end, interactive deep learning based annotation system for cursive and print English handwritten text. In ICDSMLA 2020: Proceedings of the 2nd International Conference on Data Science, Machine Learning and Applications, pp. 567-583. https://doi.org/10.1007/978-981-16-3690-5_50

[16] Kölsch, A., Mishra, A., Varshneya, S., Afzal, M.Z., Liwicki, M. (2018). Recognizing challenging handwritten annotations with fully convolutional networks. In 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), Niagara Falls, NY, USA, pp. 25-31. https://doi.org/10.1109/ICFHR-2018.2018.00014

[17] Alamsyah, D., Widhiarso, W., Hasan, S. (2022). Handwriting analysis for personality trait features identification using CNN. In 2022 International Conference on Data Science and Its Applications (ICoDSA), Bandung, Indonesia, pp. 232-238. https://doi.org/10.1109/ICoDSA55874.2022.9862910

[18] Hussain, M. (2024). Yolov5, yolov8 and yolov10: The go-to detectors for real-time vision. arXiv preprint arXiv:2407.02988. https://doi.org/10.48550/arXiv.2407.02988

[19] Zhang, C., Shi, H. (2025). YOLO-Handwritten: Improved YOLOv8 for handwritten text detection in examination papers. In Eighth International Conference on Computer Graphics and Virtuality (ICCGV 2025), pp. 89-99. https://doi.org/10.1117/12.3065327

[20] Guo, L., Wang, L., Yu, Q., Xie, X. (2024). BSMD-YOLOv8: Enhancing YOLOv8 for book signature marks detection. Applied Sciences, 14(23): 10829. https://doi.org/10.3390/app142310829

[21] Saikumari, V., Kirthika, S.V., Jayakrishna, A.R., Lokeshwaran, K. (2021). A study on effectiveness of competency mapping through training and development. Turkish Journal of Computer and Mathematics Education, 12(11): 411-416.

[22] Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., Savarese, S. (2019). Generalized intersection over union: A metric and a loss for bounding box regression. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, pp. 658-666. https://doi.org/10.1109/CVPR.2019.00075

[23] Haridas, A., NR, M., Muralidharan, R. (2021). Personality Prediction based on Handwriting using CNN MLP. International Journal of Engineering Research Technology, 9(7): 81-84.

[24] ShanWei, C., LiWang, S., Foo, N.T., Ramli, D.A. (2021). A CNN based handwritten numeral recognition model for four arithmetic operations. Procedia Computer Science, 192: 4416-4424. https://doi.org/10.1016/j.procs.2021.09.218

[25] Alqoyyum, M.A., Wibowo, A., Sarwoko, E.A. (2023). YOLOv4 algorithm implementation based on darknet and optical character recognition on vehicle license plate detection. AIP Conference Proceedings, 2683(1). https://doi.org/10.1063/5.0124890

[26] Zhu, C., Liang, J., Zhou, F. (2023). Transfer learning-based YOLOv3 model for road dense object detection. Information, 14(10): 560. https://doi.org/10.3390/info14100560

[27] Yang, Z., Feng, H., Ruan, Y., Weng, X. (2023). Tea tree pest detection algorithm based on improved Yolov7-Tiny. Agriculture, 13(5): 1031. https://doi.org/10.3390/agriculture13051031

[28] Yu, S., Ma, P.L., Singh, B., Silva, S., Pritchard, M. (2024). Two-step hyperparameter optimization method: Accelerating hyperparameter search by using a fraction of a training dataset. Artificial Intelligence for the Earth Systems, 3(1): e230013. https://doi.org/10.1175/AIES-D-23-0013.1

[29] Schreurs, A. (2022). Automatic school handwriting detection and classification based on YOLO and Vision Transformers models. Master's thesis. Utrecht University, Utrecht, Netherlands.

[30] Singh, M., Khare, V. (2022). Detection of Parkinson's

disease using the spiral diagram and convolutional neural network. Ingénierie des Systèmes d'Information, 27(6): 991-997. https://doi.org/10.18280/isi.270616
[31] Sivasankaran, P., Dhanaraj, K.R. (2024). Lung cancer detection using image processing technique through deep learning algorithm. Revue d'Intelligence Artificielle, 38(1): 297-302. https://doi.org/10.18280/ria.380131