Jurnal Ilmiah Teknik Elektro Komputer dan Informatika (JITEKI)
Vol. 11, No. 2, June 2025, pp. 346-356
ISSN: 2338-3070, DOI: 10.26555/jiteki.v11i2.30238

Handwritten Digits Detection Using Convolutional Neural Network

Doni Oktavian Ibnu Effendi, Sofia Saidah, Yusnita Putri
School of Electrical Engineering, Telkom University, 1 Telekomunikasi St., Terusan Buahbatu, Bandung, West Java, Indonesia 40257

ARTICLE INFO

Article history:

Received October 14, 2024
Revised May 24, 2025
Published July 01, 2025

Keywords:

Convolutional Neural Network;
Digits Number;
Handwritten;
YOLO;
YOLOv8

ABSTRACT

Numbers are a collection of many lines and curves and play a vital role in everyday life. Each person has unique characteristics in handwriting, making handwritten digit detection a challenging task. This paper presents an approach for detecting handwritten digits using deep learning algorithms, particularly the Convolutional Neural Network (CNN)-based YOLOv8 family models. The main objective is to compare various YOLOv8 variants (YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x) and determine the most optimal one in detecting handwritten digits. Experimental results show that the YOLOv8x variant achieves the highest performance, with a mean Average Precision (mAP) of 96.9%, a recall of 100%, a precision of 99.8%, and an F1-score of 99.9%. The research contributions are achieving high accuracy in handwritten digit detection using the YOLOv8x model and utilizing a custom primary dataset of 3,000 handwritten digits for training and evaluation, which adds novelty and real-world relevance to the study.

Corresponding Author:

Sofia Saidah, Telkom University, 1 Telekomunikasi St., Terusan Buahbatu,Bandung, West Java, Indonesia 40257
Email: sofiasaidahsfi@telkomuniversity.ac.id

1. INTRODUCTION

In the digital era, handwriting remains significant in daily life, and online handwriting recognition is an important pattern recognition task that extracts spatial and temporal features from handwritten input [1]-[6]. Handwritten character recognition remains a significant focus of research, driven by both academic curiosity and commercial applications [2], [7]. In addition to letters, numerical digits also play a vital role. In the paperless era, converting documents into editable formats is essential, including both letters and crucial numerical digits [8], [9], [10]. Each individual has a unique handwriting style, particularly in writing numbers, which gives each digit distinct visual characteristics. This variability presents challenges in recognizing handwritten digits, especially when they become difficult to distinguish by the human eye. As a result, many researchers are exploring how computers can accurately recognize and classify digit images under these complex conditions [7], [11].

Handwritten character recognition poses significant challenges, especially in low-resource scenarios involving rare scripts or unconventional writing styles. The difficulty is further compounded by the unique patterns, strokes, and character sets found in each script, which influence the system's ability to generalize across different handwriting styles [12]-[16]. In many cases, particularly in the recognition of handwritten digits, deviations from standard shapes can lead to misclassification and increased detection time. Such inefficiencies become critical in real-time and industrial applications, where fast and accurate interpretation of handwritten input is essential to maintain operational flow. Delays caused by recognition errors can negatively impact productivity and disrupt time-sensitive processes [17].

One of the technologies that can be utilized to perform automatic detection and classification is deep learning architecture, which is a subset of machine learning [9], [18]. Deep learning offers powerful capabilities in learning complex patterns from large amounts of data, making it suitable for tasks involving visual recognition. Various deep learning techniques in machine learning have been advanced to enhance the

recognition of handwritten characters [18], [19], [20]. Among its various architectures**,** one of the most widely used for image-related tasks is the Convolutional Neural Network (CNN) [21]. CNN, inspired by human visual perception, is a specialized neural network widely used in image, speech, and language recognition, making it a promising solution for detecting and classifying handwritten numbers with diverse shapes [22]. CNN consists of several specialized layers, including convolution layers for feature extraction, activation layers such as ReLU for non-linearity, pooling layers for dimensionality reduction, and finally, the fully connected layers that perform the final classification [23], [24]. The combination of these components enables CNNs to learn spatial hierarchies of features effectively and robustly [25], [26]. However, despite the effectiveness of CNNs, challenges remain when recognizing small-scale handwritten digits that occupy few pixels in an image. Feature information for these digits can easily be lost during convolution operations, particularly when background noise and clutter are present. Therefore, incorporating multi-scale detection strategies that emphasize low-level feature extraction is essential to improve precision in such cases [27].

CNN consists of two main parts: the hidden layer and the fully connected layer. The hidden layer is responsible for converting the input image into feature maps, and then the feature maps are flattened into a single line vector in the fully connected layer to perform the final classification. Within the hidden layer, there are two essential sublayers: the convolutional layer and the pooling layer. Feature learning, also known as feature extraction, is the process of retrieving unique characteristics from the processed data. This stage aims to extract important information from input images, reduce data dimensionality, and enhance precision in data processing. Feature learning generally involves two stages: the convolution layer and the pooling layer. The convolution layer performs a convolution operation between the image matrix and the filter matrix. The filter slides across the image surface in a process known as stride, which determines the distance the filter moves while extracting information from the image. The pooling layer, also known as the subsampling layer, usually follows the convolution layer. Its main purpose is down-sampling, which reduces the matrix size without changing the filter size. The most commonly used types of pooling are Average Pooling and Max Pooling. Max Pooling captures the maximum value in the pooling window, while Average Pooling calculates the average value [23], [24], [25] .

Over the years, CNN-based architectures have evolved significantly to support more complex and real-time applications. One of the most notable advancements is the YOLO (You Only Look Once) family of models, which leverages CNN principles for efficient object detection. The latest version, YOLOv8, improves upon previous models by integrating advanced techniques such as detection without reliance on predefined anchors, strengthened feature pyramid networks for better multi-scale representation, and refined loss functions. These enhancements collectively boost its accuracy and speed in object detection tasks [28]. This research presents a thorough evaluation of five different YOLOv8 model variants using a manually constructed dataset of handwritten digits. The dataset was collected from a variety of individuals, ensuring diversity in handwriting styles, thickness, and spacing to reflect real-world variability. Such diversity in data is essential where careful attention to participant variety and image preprocessing (segmentation, binarization, and inversion) significantly improved data usability for recognition models [11]. Each YOLOv8 variant was trained and tested using identical conditions to maintain consistency in performance comparison [29]. The research contributions are as follows:

(1) Providing a comprehensive comparative analysis of YOLOv8 model variants—including YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x within the context of handwritten digit detection using CNN-based object detection architecture.
(2) Introducing a novel handwritten digit dataset that captures a wide range of human writing patterns, enabling robust benchmarking of object detection models in scenarios where digits may appear distorted, unclear, or overlapping.
(3) Evaluating detection accuracy, precision, recall, and inference speed of each YOLOv8 variant to identify the most optimal model for handwritten digit recognition applications, particularly in environments with limited computational resources.
(4) Demonstrating the potential application of this approach for digital document automation, form scanning, and real-time digit input in industry workflows where handwriting recognition plays a crucial role.

By combining CNN architecture and YOLOv8's capability for real-time object detection, this research offers valuable insights into model selection strategies and opens up opportunities for the deployment of accurate, efficient handwritten digit recognition systems in practical applications.

2. METHODS

2.1. Dataset

The dataset used in this study was created by the authors and consists of 10 classes representing handwritten digits: "zero," "one," "two," "three," "four," "five," "six," "seven," "eight," and "nine." The total

dataset contains 3,000 samples, which are divided into three parts: 2,100 samples for training, 600 samples for validation, and 300 samples for testing. These samples are handwritten digits written on white paper using black ballpoint pens by several random individuals. An example of the dataset images is shown in Fig. 1.

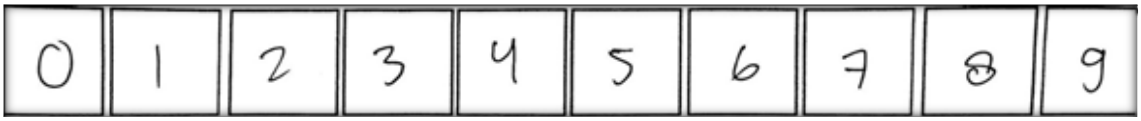

Fig. 1. Example Dataset

2.2. System Design

This study utilizes the CNN method with the YOLOv8 algorithm for handwritten digit detection. The input data are images of handwritten digits from the custom dataset described above. The images first undergo preprocessing, including manual cropping and resizing. YOLOv8 has five variants: YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x. Experiments are conducted with training over four different epoch values: 40, 120, 160, and 200 epochs. To provide a clearer overview of the research workflow, a research flowchart is presented in Figure 2. It outlines the main stages of the study, from data collection and preprocessing to model training, fine-tuning, and performance evaluation. Fig. 2 shows the research flowchart of the YOLOv8-based handwritten digit detection system.

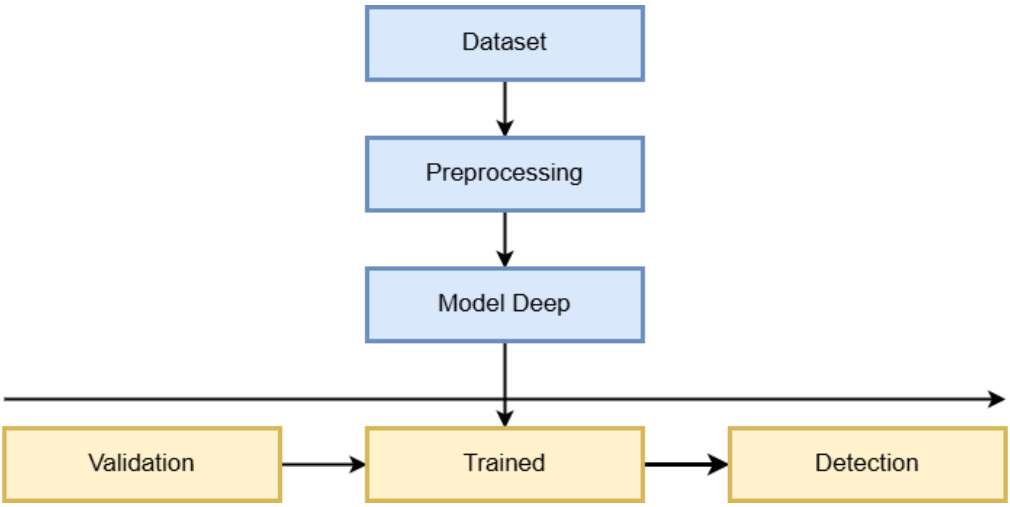

Fig. 2. Research Flowchart of the YOLOv8-based Handwritten Digit Detection

Following preprocessing steps such as cropping and resizing the images, the dataset was split into 2,100 training samples, 600 validation samples, and 300 testing samples. The evaluation was carried out using five different YOLOv8 model variants, with the specific hyperparameter settings for each model summarized in Table 1.

Table 1. Model Hyperparameter Value

Model	Depth Multiple	Width Multiple	Max Channels
YOLOv8n	0.33	0.25	1024
YOLOv8s	0.33	0.50	1024
YOLOv8m	0.67	0.75	768
YOLOvl	1.00	1.00	512
YOLOvx	1.00	1.25	512

Following training, a fine-tuning process is performed to determine the optimal number of epochs for each model. The models are then tested on the test dataset to evaluate their ability to detect handwritten numeric objects. The performance metrics used to assess the models include mAP, recall, precision, and F1-score.

2.3. YOLOv8 Algorithm

You Only Look Once (YOLO) is a deep learning model widely used for object detection that processes an entire image through a single neural network. This approach enables simultaneous classification and

localization in one step by analyzing images across multiple positions and scales, with the highest scoring region identified as the detected object. YOLOv8, the latest version developed by Ultralytics, supports a range of vision tasks including object detection, image classification, and instance segmentation. As a one-stage detector, YOLOv8 comprises two main components: the backbone network, which extracts features using convolutional layers, and the head network, which performs detection. The backbone is based on the CSPDarknet53 architecture, consisting of 53 convolutional layers. Additionally, YOLOv8 offers advanced training features such as mosaic augmentation, applied during training but disabled in the final 10 epochs to improve performance. The model is available in five scaled variants: nano (YOLOv8n), small (YOLOv8s), medium (YOLOv8m), large (YOLOv8l), and extra-large (YOLOv8x), enabling flexible use across diverse applications [30]. The architecture of YOLOv8 is illustrated in Fig. 3.

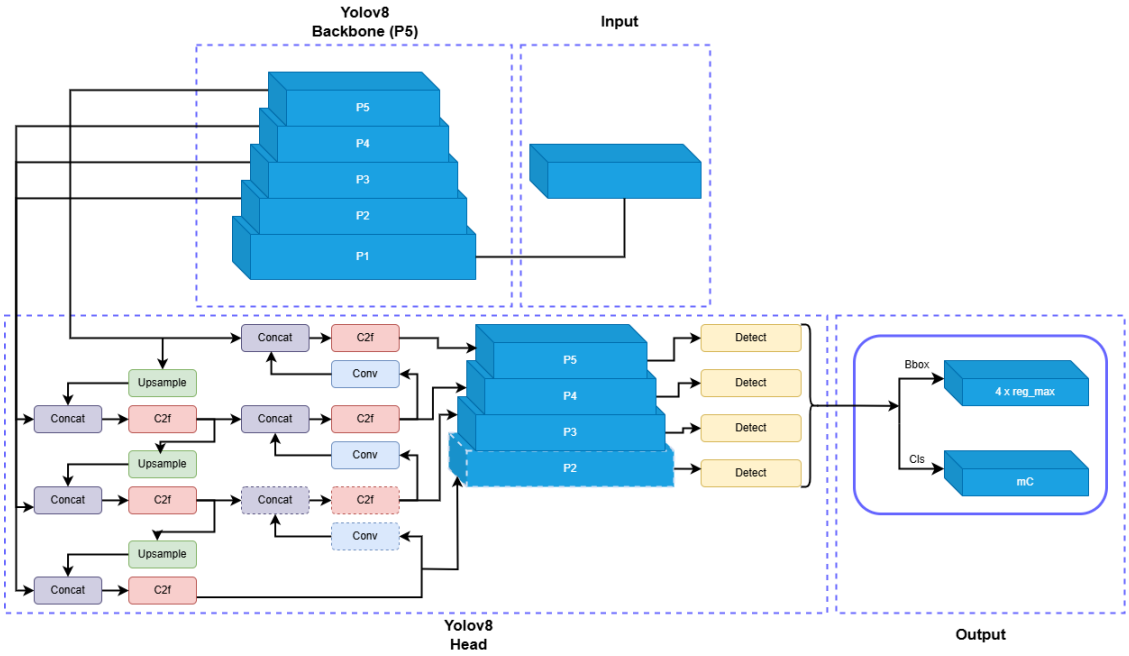

Fig. 3. YOLOv8 Architecture

The YOLOv8 architecture comprises a backbone network with multiple convolutional layers that extract hierarchical features, a Feature Pyramid Network (FPN) to improve detection across various scales, and a prediction head that utilizes convolutional and upsampling blocks to refine features and generate final detection results [31].

2.4. System Performance

This study uses four parameters to measure system performance: mAP, recall, precision, and F1-score. The formulas for these performance metrics are presented in Equations (1)-(4).

$$mAP = \frac{1}{n}\sum_{k=1}^{k=n} AP_k \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \times 100\% \tag{2}$$

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$F1 - Score = 2 \times \frac{Recall \times Precision}{Recall + Precision} \tag{4}$$

TP (True Positive) is the number of data points for which the prediction model is positive and the actual data is also positive, indicating correct classification. FP (False Positive) is the number of negative data points incorrectly predicted as positive by the model. FN (False Negative) is the number of positive data points incorrectly predicted as negative. TN (True Negative) is the number of negative data points correctly predicted as negative. The conclusion is that the classification is correct. N is the total number of classes, k is the class index, and AP_k represents the average precision for class k.

Based on the information above, several metrics are used to evaluate the performance of the YOLOv8 model. Precision measures the accuracy of the model in correctly classifying positive samples out of all positive predictions made by the model. Recall measures the ability of the model to identify all actual positive samples among the total positive data available. F1-Score is the harmonic mean of precision and recall, used when a balance between these two metrics is desired**.** The mAP is a specialized metric used to evaluate object detection performance, combining precision and recall across various detection thresholds. The formulas above clarify that N is the number of classes, k refers to the specific class, and AP_k is the average precision of class k.

3. RESULTS AND DISCUSSION

The digit number dataset used in this study has undergone a preprocessing stage and consists of 3,000 images of digits ranging from 0 to 9. The dataset is split into 2,100 training images, 600 validation images, and 300 test images. System performance is then evaluated using four key metrics: mAP, recall, precision, and F1-score. In this experiment, the training was conducted using a learning rate of 0.01 and the SGD optimizer. Table 2 shows the comparison of mAP values across five YOLOv8 model variants at different training epochs.

Table 2. Comparison of mAP Result Each Epoch

Model	Epoch (40)	Epoch (80)	Epoch (120)	Epoch (160)	Epoch (200)
YOLOv8n	95.6%	96.2% (0.6%↑)	96.5% (0.3%↑)	96.6% (0.1%↑)	96.8% (0.2%↑)
YOLOv8s	95.9%	96.8% (0.9%↑)	96.7% (0.1%↓)	96.8% (0.1%↑)	97% (0.2%↑)
YOLOv8m	95.6%	96.5% (0.9%↑)	96.6% (0.1%↑)	96.8% (0.2%↑)	96.6% (0.2%↓)
YOLOv8l	95.9%	96.8% (0.9%↑)	96.8% (0%)	96.8% (0%)	96.9% (0.1%↑)
YOLOv8x	96%	96.5% (0.5%↑)	96.6% (0.1%↑)	96.9% (0.3%↑)	95.8% (1.1%↓)

Based on the results in Table 2, it can be observed that all model variants generally experienced an increase in mAP values as the number of epochs increased, especially in the range of 40 to 160 epochs. The YOLOv8s model consistently demonstrated strong performance and achieved the highest mAP of 97.0% at 200 epochs. Meanwhile, YOLOv8x, despite achieving a high mAP of 96.9% at 160 epochs, showed a significant performance drop to 95.8% at 200 epochs, indicating potential overfitting. The YOLOv8l model exhibited stable performance across all epochs, with minimal fluctuation in mAP, making it a reliable option. YOLOv8n and YOLOv8m also showed steady improvement, though YOLOv8m experienced a slight decline at the last epoch. These trends suggest that while deeper and larger models like YOLOv8x can offer higher early performance, they may be more sensitive to overtraining. In conclusion, YOLOv8s offers a favorable trade-off between model size and accuracy, making it suitable for real-time digit classification tasks on moderately constrained systems.

In order to further understand the architectural differences that may influence model performance, Table 3 presents a comparison of YOLOv8 variants based on their structural parameters, including depth multiple, width multiple, and maximum number of channels used. These architectural settings play a crucial role in determining model capacity, speed, and risk of overfitting.

Table 3. Comparison of YOLOv8 Variants Based on Structural Complexity

Model	Depth Multiple	Width Multiple	Max Channels
YOLOv8n	0.33	0.25	1024
YOLOv8s	0.33	0.50	1024
YOLOv8m	0.67	0.75	768
YOLOv8l	1.00	1.00	512
YOLOv8x	1.00	1.25	512

As shown in Table 2, the most optimal performance was achieved at 160 epochs, where all YOLOv8 model variants demonstrated an increase in mAP that was linearly aligned with precision and recall. In contrast, training up to 200 epochs led to overfitting, particularly in the YOLOv8x variant, which ceased to improve after 176 epochs. Table 3 highlights that YOLOv8x has the highest width multiple (1.25) and a relatively high depth multiple (1.00), contributing to a more complex model architecture with increased computational load and susceptibility to overfitting. Despite this, YOLOv8x achieved the best performance at 160 epochs, indicating that its advanced capacity allowed it to effectively learn intricate digit patterns when provided with sufficient but not excessive training. Meanwhile, smaller models such as YOLOv8n and YOLOv8s, with lower structural complexity, did not overfit but also failed to reach high performance, likely due to their limited ability to capture diverse handwriting features.

Table 4 presents the performance results of the YOLOv8x model across all digit classes after 160 training epochs. This model was selected due to its optimal balance of precision and recall as observed in the previous evaluation.

Table 4. YOLOv8x All Class Model Testing Result at 160 Epochs

Class	Precision	Recall	F1-Score	mAP
All	99.8%	100%	99.9%	96.9%
Zero	99.7%	100%	99.85%	98.6%
One	99.8%	100%	99.9%	84.8%
Two	**100%**	99.6%	99.8%	97.3%
Three	99.8%	100%	99.9%	98.5%
Four	99.5%	100%	99.75%	98.7%
Five	99.9%	100%	**99.95%**	95.9%
Six	99.8%	100%	99.9%	98.8%
Seven	99.7%	100%	99.85%	98.1%
Eight	99.7%	100%	99.85%	98.9%
Nine	99.7%	100%	99.85%	**99.1%**

From Table 4, it can be observed that the highest precision value is achieved by class "two" at 100%. Meanwhile, recall scores are consistently at 100% across all classes except class "two", which recorded a slightly lower recall of 99.6%. The class with the highest F1-score is "five", reaching 99.95%, indicating a perfect balance between precision and recall for that class. In terms of mean Average Precision (mAP), class "nine" records the highest score at 99.1%, whereas class "one" has the lowest mAP at 84.8%. Despite this, the model achieves an overall mAP of 96.9% across all classes, demonstrating strong generalization and robustness in digit classification at this epoch level.

These results indicate that the YOLOv8x model is most optimal at the 160th epoch, as it is able to maintain high precision and recall values across nearly all digit classes without significant signs of overfitting. The lower mAP in class "one" may suggest a higher intra-class variability or less distinctive features compared to other digits, which could be further investigated in future work. Compared to other variants, YOLOv8x stands out in this experiment due to its higher depth and width multipliers, which allow it to learn richer and more detailed features. However, this comes with the trade-off of increased computational complexity. Thus, while YOLOv8x offers the best performance, its deployment in resource-limited environments may require further optimization or pruning strategies.

Table 5 presents a comparative analysis of the proposed method using YOLOv8s with various state-of-the-art approaches in handwritten digit recognition. These methods range from traditional machine learning techniques to advanced deep learning architectures and ensemble models. The comparison considers datasets used, performance metrics, and observed limitations to provide a holistic overview of progress in the field.

Table 5 provides a comparative overview of various handwritten digit detection methods, ranging from traditional machine learning models to advanced deep learning and ensemble-based approaches. Although several prior studies achieved high accuracy, such as EfficientDet-D4 and various CNN-based models, the proposed method using YOLOv8x stands out for its real-time detection capability and balanced performance across multiple evaluation metrics. In particular, the YOLOv8x model achieved a mean Average Precision (mAP) of 96.9% during the 160-epoch training phase, with a recall of 100%, precision of 99.8%, and an F1-score of 99.9%, indicating strong generalization and robust feature learning. While some models reported higher accuracy on specific datasets, they often lacked adaptability across varying data conditions or required high computational resources. Compared to these, YOLOv8x offers a practical trade-off between accuracy and efficiency, making it highly suitable for real-time handwritten digit detection tasks.

Table 5. Comparison of Existing Approaches of Handwritten Digit Detection

Reference	Year	Method/ Model	Dataset	Results/Accuracy/Detection Rate	Limitation/Notes
This Study	2025	YOLOv8x (best variant)	3,000 digit images (preprocessed)	mAP 96.9% (160 epochs)	High accuracy, real-time capable, efficient size
Sohail *et al.* [32]	2023	EfficientDet-D4 (EfficientNet-B4 backbone)	MNIST, USPS	Accuracy = 99.83% (MNIST), 99.10% (USPS)	High accuracy across datasets, robust to noise, blurring, chrominance, position, and style variations
Adhikary *et al.* [33]	2023	Support Vector Machine (SVM)	Dzongkha handwritten digits (custom)	Accuracy = 98.29%	Low-resource language, newly collected dataset, promising result in a complex script
Kusetogullari *et al.* [34]	2021	DIGITNET-dect	DIDA (historical digit dataset)	75.96% detection rate	Lower performance due to historical digit variability
Gope *et al.* [35]	2021	SVM Classifier	MNIST	Accuracy = 95.88%	Traditional ML with competitive accuracy and low time
Ahlawat *et al.* [16]	2020	Pure CNN architecture with optimized learning parameters	MNIST	Accuracy = 99.87%	Outperforms ensemble methods while reducing computational cost and testing complexity
Boukharouba and Bennia [36]	2017	Handcrafted features using chain code histogram with SVM classifier	Handwritten Farsi digit dataset (Hoda)	Accuracy = 98.55%	Unable to distinguish between similar classes
Govindarajan [37]	2013	Ensemble classifier using bagged-SVM, bagged-RBF, and RBF-SVM	MNIST	Accuracy = 98.0%	Lack of robustness to unseen samples
Cruz *et al.* [38]	2010	Gradient directional features with MLP classifier	MNIST	Accuracy = 99.68%	Performance degrades on distorted samples
Dine *et al.* [39]	2017	Structural and statistical features with KNN classifier	MNIST	Accuracy = 95%	Computationally inefficient due to large vector size
Hou and Zhao [40]	2017	Hybrid feature using Gabor filter and CNN with ANN classifier	MNIST	Accuracy = 99.23%	Performance degrades for samples with intense rotation variation
Pham *et al.* [41]	2014	RNN	RIMES, IAM, OpenHaRT	Character error rate = 9.17% (OpenHaRT best)	Needs performance improvement
Shamim *et al.* [42]	2018	Several ML algorithms: MLP, Naive-Bayes, Bayes-Net, SVM, J48, RF, Random tree	Digits-A dataset	Accuracy = 90.37%	Needs evaluation on other complex datasets

Shi *et al.* [43]	2017	CRNN (CNN + RNN)	IIIT-5k, Street View Text, ICDAR	Accuracy = 98.7%	High inference time limits practical applications
Arbain *et al.* [44]	2018	Handcrafted features (multizoning) with SVM and MLP classifier	MNIST	Accuracy = 95.35%	Performance degrades on shape and size variations
Assegie and Nair [45]	2019	Spatial features with decision tree classifier	MNIST	Accuracy = 83.4%	Performance needs improvement
Ali *et al.* [46]	2020	CNN with ELM classifier	MNIST	Accuracy = 99.8%	Requires generalization on unseen data
Aly and Almotairi [47]	2020	Deep convolutional self-organizing map	MNIST	Error rate = 0.57%	Performance degrades on geometric variations like rotation
Jain *et al.* [48]	2018	Modified LeNet-CNN	MNIST	Accuracy = 99.5%	Low performance on intense rotational variations
Malik and Roy [49]	2019	Spatial and spectral features with ANN and ELM classifier	MNIST	Accuracy = 98.4% (ELM)	Performance needs improvement
Albahli *et al.* [50]	2021	DenseNet + faster RCNN	MNIST	Accuracy = 99.78%	Computationally expensive

4. CONCLUSION

This study demonstrates that handwritten digit detection can be effectively performed using the Convolutional Neural Network (CNN) approach with the YOLOv8 algorithm, which consists of five model variants: YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x. Among these, the YOLOv8x variant achieved the most optimal performance during the 160-epoch training iteration. It recorded a mean Average Precision (mAP) of 96.9%, with corresponding recall, precision, and F1-score values of 100%, 99.8%, and 99.9%, respectively. These results highlight YOLOv8x's strong capability in feature extraction and classification accuracy for handwritten digit recognition, albeit with a higher computational cost due to its larger architecture.

REFERENCES

[1] S. Singh, A. Sharma, and V. K. Chauhan, "Online handwritten Gurmukhi word recognition using fine-tuned Deep Convolutional Neural Network on offline features," *Machine Learning with Applications*, vol. 5, p. 100037, Sep. 2021, https://doi.org/10.1016/j.mlwa.2021.100037.

[2] B. N. Van and V. T. Hoang, "A Short Review for Handwritten Math Expression Recognition Techniques," in *Procedia Computer Science*, pp. 231–239, 2024, https://doi.org/10.1016/j.procs.2024.04.025.

[3] A. Chiney *et al.*, "Handwritten Data Digitization Using an Anchor based Multi-Channel CNN (MCCNN) Trained on a Hybrid Dataset (h-EH)," in *Procedia CIRP*, pp. 175–182, 2021, https://doi.org/10.1016/j.procs.2021.05.095.

[4] R. C. Karpagalakshmi, S. Hegde, R. S. R, S. Mahapatra, E. S. Leni, and A. Sungheetha, "Deep Learning-Based Recognition of Handwritten English Characters," *Procedia Comput Sci*, vol. 258, pp. 1783–1792, 2025, https://doi.org/10.1016/j.procs.2025.04.430.

[5] H. Deshpande, D. Chaudhari, T. Sarode, A. Kamath, and D. Khan, "LeanConv-SENet: A novel lightweight Neural Network Architecture for Handwritten character recognition," *Procedia Comput Sci*, vol. 258, pp. 2826–2836, 2025, https://doi.org/10.1016/j.procs.2025.04.543.

[6] R. Barati, "Incorporating locally linear embedding and multi-layer perceptron in handwritten digit recognition," *e-Prime - Advances in Electrical Engineering, Electronics and Energy*, vol. 2, Jan. 2022, https://doi.org/10.1016/j.prime.2022.100081.

[7] J. O. Bappi, M. A. T. Rony, and M. S. Islam, "BNVGLENET: Hypercomplex Bangla handwriting character recognition with hierarchical class expansion using Convolutional Neural Networks," *Natural Language Processing Journal*, vol. 7, p. 100068, Jun. 2024, https://doi.org/10.1016/j.nlp.2024.100068.

[8] J. Pareek, D. Singhania, R. R. Kumari, and S. Purohit, "Gujarati Handwritten Character Recognition from Text Images," in *Procedia Computer Science*, pp. 514–523, 2020, https://doi.org/10.1016/j.procs.2020.04.055.

[9] R. Ameri, A. Alameer, S. Ferdowsi, K. Nazarpour, and V. Abolghasemi, "Labeled projective dictionary pair learning: application to handwritten numbers recognition," *Inf Sci (N Y)*, vol. 609, pp. 489–506, Sep. 2022, https://doi.org/10.1016/j.ins.2022.07.070.

[10] A. U. Islam, M. J. Khan, M. Asad, H. A. Khan, and K. Khurshid, "iVision HHID: Handwritten hyperspectral images dataset for benchmarking hyperspectral imaging-based document forensic analysis," *Data Brief*, vol. 41, Apr. 2022, https://doi.org/10.1016/j.dib.2022.107964.
[11] P. A. Abdalla, A. M. Qadir, M. Y. Shakor, A. M. Saeed, and A. T. Jabar, "A vast dataset for Kurdish handwritten digits and isolated characters recognition," *Data in Brief*, 2023, https://doi.org/10.17632/zb66pp7vjh.1.
[12] K. K. Podder *et al.*, "Self-ChakmaNet: A deep learning framework for indigenous language learning using handwritten characters," *Egyptian Informatics Journal*, vol. 24, no. 4, Dec. 2023, https://doi.org/10.1016/j.eij.2023.100413.
[13] Jyoti, K. Solanki, K. Kaushik, Sudhir, and S. Dalal, "Handwriting Recognition: Unravelling Performance Diversity and Practical Implications," *Procedia Comput Sci*, vol. 259, pp. 1387–1397, 2025, https://doi.org/10.1016/j.procs.2025.04.093.
[14] V. Agrawal, J. Jagtap, and M. V. V. P. Kantipudi, "Exploration of advancements in handwritten document recognition techniques," *Elsevier B.V.*, 2024, https://doi.org/10.1016/j.iswa.2024.200358.
[15] F. Haghighi and H. Omranpour, "Stacking ensemble model of deep learning and its application to Persian/Arabic handwritten digits recognition," *Knowl Based Syst*, vol. 220, May 2021, https://doi.org/10.1016/j.knosys.2021.106940.
[16] S. Ahlawat, A. Choudhary, A. Nayyar, S. Singh, and B. Yoon, "Improved handwritten digit recognition using convolutional neural networks (Cnn)," *Sensors (Switzerland)*, vol. 20, no. 12, pp. 1–18, Jun. 2020, https://doi.org/10.3390/s20123344.
[17] M. A. Souibgui, A. Fornés, Y. Kessentini, and B. Megyesi, "Few shots are all you need: A progressive learning approach for low resource handwritten text recognition," *Pattern Recognit Lett*, vol. 160, pp. 43–49, Aug. 2022, https://doi.org/10.1016/j.patrec.2022.06.003.
[18] M. B. Bora, D. Daimary, K. Amitab, and D. Kandar, "Handwritten Character Recognition from Images using CNN-ECOC," in *Procedia Computer Science*, pp. 2403–2409, 2020, https://doi.org/10.1016/j.procs.2020.03.293.
[19] J. Ravi, "Handwritten alphabet classification in Tamil language using convolution neural network," *International Journal of Cognitive Computing in Engineering*, vol. 5, pp. 132–139, Jan. 2024, https://doi.org/10.1016/j.ijcce.2024.03.001.
[20] A. A. A. Ali and S. Mallaiah, "Intelligent handwritten recognition using hybrid CNN architectures based-SVM classifier with dropout," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 6, pp. 3294–3300, Jun. 2022, https://doi.org/10.1016/j.jksuci.2021.01.012.
[21] Z. Kayumov, D. Tumakov, and S. Mosin, "Hierarchical Convolutional Neural Network for Handwritten Digits Recognition," in *Procedia Computer Science*, pp. 1927–1934, 2020, https://doi.org/10.1016/j.procs.2020.04.206.
[22] G. Yan, "Basketball motion recognition and tracking method based on improved convolutional neural network," *Systems and Soft Computing*, vol. 7, Dec. 2025, https://doi.org/10.1016/j.sasc.2025.200272.
[23] J. Niu *et al.*, "Research Note: Application of Convolutional Neural Networks for Feather Classification in Chickens," *Poult Sci*, p. 105254, May 2025, https://doi.org/10.1016/j.psj.2025.105254.
[24] Aanchal, Nidhi, Preeti, and Gurpratap, "Automatic Cropping of Handwritten Scanned Documents with Object Detection Algorithm," in *Procedia Computer Science*, pp. 1733–1741, 2022, https://doi.org/10.1016/j.procs.2023.01.151.
[25] M. Kohler, A. Krzyżak, and B. Walter, "Analysis of the rate of convergence of an over-parametrized convolutional neural network image classifier learned by gradient descent," *J Stat Plan Inference*, vol. 239, p. 106291, Dec. 2025, https://doi.org/10.1016/j.jspi.2025.106291.
[26] M. Raquib, M. A. Hossain, M. K. Islam, and M. S. Miah, "VashaNet: An automated system for recognizing handwritten Bangla basic characters using deep convolutional neural network," *Machine Learning with Applications*, vol. 17, p. 100568, Sep. 2024, https://doi.org/10.1016/j.mlwa.2024.100568.
[27] X. Chai, M. Zhao, J. Li, and J. Li, "Image small target detection in complex traffic scenes based on Yolov8 multiscale feature fusion," *Alexandria Engineering Journal*, vol. 126, pp. 578–590, Jul. 2025, https://doi.org/10.1016/j.aej.2025.04.105.
[28] A. T. Khan, S. M. Jensen, and A. R. Khan, "Advancing precision agriculture: A comparative analysis of YOLOv8 for multi-class weed detection in cotton cultivation," *Artificial Intelligence in Agriculture*, vol. 15, no. 2, pp. 182–191, Jun. 2025, https://doi.org/10.1016/j.aiia.2025.01.013.
[29] A. Nayfeh, S. Al-Azani, and H. Samma, "A Two-Stage YOLOv8 Approach for Waste Detection and Classification in Cognitive Cities," in *Transportation Research Procedia*, pp. 579–586, 2025, https://doi.org/10.1016/j.trpro.2025.03.111.
[30] P. Zhu and W. Zhou, "Identification of glass eel capture equipment in the Yangtze River estuary based on high-spatial -resolution imagery and an improved YOLOv8 model," *Ecol Inform*, vol. 89, Nov. 2025, https://doi.org/10.1016/j.ecoinf.2025.103188.
[31] N. Jegham, C. Y. Koh, M. Abdelatti, and A. Hendawi, "YOLO Evolution: A Comprehensive Benchmark and Architectural Review of YOLOv12, YOLO11, and Their Previous Versions," *Arxiv,* 2024, [Online]. Available: http://arxiv.org/abs/2411.00201.
[32] S. S. Ahmed, Z. Mehmood, I. A. Awan, and R. M. Yousaf, "A Novel Technique for Handwritten Digit Recognition Using Deep Learning," *J Sens*, vol. 2023, 2023, https://doi.org/10.1155/2023/2753941.

[33] P. K. Adhikary, P. Dadure, P. Saha, Tawmo, and P. Pakray, "Dzongkha Handwritten Digit Recognition using Machine Learning Techniques," in *Procedia Computer Science*, pp. 2350–2358, 2022, https://doi.org/10.1016/j.procs.2023.01.210.
[34] H. Kusetogullari, A. Yavariabdi, J. Hall, and N. Lavesson, "DIGITNET: A Deep Handwritten Digit Detection and Recognition Method Using a New Historical Handwritten Digit Dataset," *Big Data Research*, vol. 23, Feb. 2021, https://doi.org/10.1016/j.bdr.2020.100182.
[35] B. Gope, S. Pande, N. Karale, S. Dharmale, and P. Umekar, "Handwritten digits identification using mnist database via machine learning models," in *IOP Conference Series: Materials Science and Engineering*, Jan. 2021. https://doi.org/10.1088/1757-899X/1022/1/012108.
[36] A. Boukharouba and A. Bennia, "Novel feature extraction technique for the recognition of handwritten digits," *Applied Computing and Informatics*, vol. 13, no. 1, pp. 19–26, Jan. 2017, https://doi.org/10.1016/j.aci.2015.05.001.
[37] M. Govindarajan, "Evaluation of Ensemble Classifiers for Handwriting Recognition," *International Journal of Modern Education and Computer Science*, vol. 5, no. 11, pp. 11–20, Nov. 2013, https://doi.org/10.5815/ijmecs.2013.11.02.
[38] R. M. O. Cruz, G. D. C. Cavalcanti, and T. I. Ren, "Handwritten Digit Recognition Using Multiple Feature Extraction Techniques and Classifier Ensemble," in *IWSSIP 17th International Conference on Systems, Signals and Image Processing*, 2010. [Online]. Available: www.cin.ufpe.br/~viisarRecife,Brazil.
[39] K. Z. Dine, M. Nasri, M. Moussaoui, S. Benchaou, and F. Aouinti, "Digit recognition using different features extraction methods," in *Advances in Intelligent Systems and Computing*, Springer Verlag, pp. 167–175, 2017, https://doi.org/10.1007/978-3-319-46568-5_17.
[40] Y. Hou and H. Zhao, "Handwritten Digit Recognition Based on Depth Neural Network," in *ICIIBMS 2017, Track2: Artificial Intelligence, Robotics and Human-Computer Interaction*, pp. 35-38, 2017, https://doi.org/10.1109/ICIIBMS.2017.8279710.
[41] V. Pham, T. Bluche, C. Kermorvant, and J. Louradour, "Dropout Improves Recurrent Neural Networks for Handwriting Recognition," in *Proceedings of International Conference on Frontiers in Handwriting Recognition, ICFHR*, pp. 285–290, Dec. 2014, https://doi.org/10.1109/ICFHR.2014.55.
[42] S. M. Shamim, M. B. A. Miah, A. Sarker, M. Rana, and A. Al Jobair, "Handwritten digit recognition using machine learning algorithms," *Indonesian Journal of Science and Technology*, vol. 3, no. 1, pp. 29–39, 2018, https://doi.org/10.17509/ijost.v3i1.10795.
[43] B. Shi, X. Bai, and C. Yao, "An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition," *IEEE Trans Pattern Anal Mach Intell*, vol. 39, no. 11, pp. 2298–2304, Nov. 2017, https://doi.org/10.1109/TPAMI.2016.2646371.
[44] N. Atikah Arbain, M. S. Azmi, A. K. Muda, N. A. Muda, and A. R. Radzid, "Offline Handwritten Digit Recognition Using Triangle Geometry Properties," 2018. [Online]. Available: www.mirlabs.net/ijcisim/index.html.
[45] T. A. Assegie and P. S. Nair, "Handwritten digits recognition with decision tree classification: A machine learning approach," *International Journal of Electrical and Computer Engineering*, vol. 9, no. 5, pp. 4446–4451, Oct. 2019, https://doi.org/10.11591/ijece.v9i5.pp4446-4451.
[46] S. Ali, J. Li, Y. Pei, M. S. Aslam, Z. Shaukat, and M. Azeem, "An effective and improved cnn-elm classifier for handwritten digits recognition and classification," *Symmetry (Basel)*, vol. 12, no. 10, pp. 1–15, Oct. 2020, https://doi.org/10.3390/sym12101742.
[47] S. Aly and S. Almotairi, "Deep Convolutional Self-Organizing Map Network for Robust Handwritten Digit Recognition," *IEEE Access*, vol. 8, pp. 107035–107045, 2020, https://doi.org/10.1109/ACCESS.2020.3000829.
[48] A. Jain, G. R. K. Sai Subrahmanyam, and D. Mishra, "Rotation invariant digit recognition using convolutional neural network," in *Advances in Intelligent Systems and Computing*, pp. 91–102, 2018, https://doi.org/10.1007/978-981-10-7895-8_8.
[49] H. Malik and N. Roy, "Extreme learning machine-based image classification model using handwritten digit database," in *Advances in Intelligent Systems and Computing*, vol. 697, pp. 607–618, 2019, https://doi.org/10.1007/978-981-13-1822-1_57.
[50] S. Albahli, M. Nawaz, A. Javed, and A. Irtaza, "An improved faster-RCNN model for handwritten character recognition," *Arab J Sci Eng*, vol. 46, no. 9, pp. 8509–8523, Sep. 2021, https://doi.org/10.1007/s13369-021-05471-4.

BIOGRAPHY OF AUTHORS

Doni Oktavian Ibnu Effendi received a bachelor's degree in telecommunication engineering from Telkom University, Indonesia in 2023. He has an interest in deep learning, image processing and computer vision. He can be contacted at email: doni.effendi.2014@gmail.com.

Sofia Saidah received the B.S. and M.S. degree in telecommunication engineering from Telkom Institute of Technology, Bandung, Indonesia in 2012 and 2014 respectively. She is currently a lecturer in School of Electrical Engineering Telkom University. Her research interests include image processing, audio processing, biomedical engineering, steganography and watermarking. She can be contacted at email: sofiasaidahsfi@telkomuniversity.ac.id.

Yusnita Putri is pursuing a Bachelor's Degree in Telecommunication Engineering at the School of Electrical Engineering, Telkom University. Her research interests include image processing, signal processing, interdisciplinary applications of telecommunication and biology, also human-computer interaction and design. Email: putriyusnita@student.telkomuniversity.ac.id.