International Journal of Informatics and Data Science
ISSN 3026-7315 (media online)
Volume 3, No 2, June 2026, Page: 89–99
DOI: doi.org/10.64366/ijids.v3i2.604
https://journals.adaresearch.or.id/ijids/index

Comparative Evaluation of YOLOv8 Variants for Static BISINDO Alphabet Detection: Accuracy and Computational Cost Analysis

Yopy Tri Buana[1,*], Yonky Pernando[2], Raymond Erz Saragih[2], Mohammad Fadhol[3], Agus Suwandi[2]

[1] Faculty Computer, Department of Software Engineering, Universitas Universal, Batam, Indonesia
[2] Faculty Computer, Department of Informatics Engineering, Universitas Universal, Batam, Indonesia
[3] Faculty Computer, Department of Information Systems, Universitas Universal, Batam, Indonesia
Email: [1]yopytribuana@uvers.ac.id, [2]yongkyfernando194@gmail.com, [3]raymondes@uvers.ac.id, [4]mohammadfadhol@uvers.ac.id, [5]agus.suwandi@uvers.ac.id
Correspondence Author Email: yopytribuana@uvers.ac.id
Received: 18/05/2026, Revised: 10/06/2026, Accepted: 20/06/2026, Published: 30/06/2026

Abstract–Indonesian Sign Language (BISINDO) plays an important role in communication for deaf communities in Indonesia. Automatic BISINDO alphabet recognition requires accurate detection while maintaining reasonable computational cost. This study evaluates the effect of model capacity on static BISINDO alphabet detection by comparing YOLOv8n, YOLOv8s, YOLOv8m, and YOLOv8l under consistent experimental conditions. A dataset of 11,468 images representing 26 alphabet classes was divided into 9,168 training, 1,155 validation, and 1,145 held-out test images. All models were trained for 50 epochs using 416 × 416 pixel images on an NVIDIA RTX 2060 with 6 GB memory. Performance was evaluated using precision, recall, mAP@0.5, mAP@0.5:0.95, and training time. The validation results achieved mAP@0.5 values of 99.42%, 99.46%, 99.38%, and 99.40% for YOLOv8n, YOLOv8s, YOLOv8m, and YOLOv8l, respectively, while training time increased from 0.93 to 3.59 hours. Held-out test evaluation remained consistently high, with the main errors involving the visually similar M and N gestures. Because the subsets originate from the same public dataset source, the results provide within-dataset evidence and do not establish cross-dataset or real-world generalization. Under the evaluated conditions, YOLOv8n provides the most efficient balance between detection performance and computational cost.

Keywords: BISINDO; YOLOv8; Object Detection; Sign Language Recognition; Model Comparison

1. INTRODUCTION

Indonesian Sign Language (BISINDO) is widely used by the deaf community in Indonesia and represents an important medium for communication and accessibility. Recent studies have explored computer-vision-based approaches for BISINDO recognition, including image classification and static alphabet recognition using deep learning [1], [2], [3]. However, automatic recognition of sign-language alphabets remains challenging because different signs can involve subtle variations in handshape, finger configuration, and spatial arrangement. These characteristics make accurate representation of hand features an important consideration when selecting a detection architecture.

Recent advances in deep learning have made one-stage object detectors, particularly the YOLO family, increasingly applicable to sign-language recognition. YOLOv8 provides multiple model variants with different scales and architectural capacities, allowing a model to be selected according to the accuracy and computational requirements of a particular application [4], [5]. A larger model may provide richer feature representations and potentially capture subtle differences in handshape and finger configuration more effectively. However, increasing model capacity also increases computational requirements. Therefore, it remains unclear whether increasing YOLOv8 model capacity provides meaningful accuracy improvements for static BISINDO alphabet detection.

Previous studies have demonstrated the applicability of deep learning and object detection to BISINDO recognition. Sari et al. investigated Indonesian sign-language recognition using LSTM [6], while Josef and Kusuma applied deep learning with Bayesian optimization for sign-language alphabet recognition [7]. Fresmanda et al. evaluated YOLOv8 for BISINDO alphabet detection and reported a precision of 93.35%, recall of 70.25%, mAP@0.5 of 64.05%, and mAP@0.5:0.95 of 45.88% [8]. Ningsih et al. subsequently investigated YOLOv5-NAS-S for BISINDO detection and reported a mAP of 97.2% and recall of 99.6% [9]. These studies demonstrate the feasibility of deep-learning and YOLO-based approaches for BISINDO recognition, but they primarily evaluate specific architectures or configurations rather than systematically examining the effect of model capacity within the same YOLOv8 family.

Recent studies have also highlighted challenges related to BISINDO recognition under different visual conditions. Saputra and Rakun investigated BISINDO recognition in complex backgrounds and emphasized the importance of robustness to background variation and computational efficiency [10]. In addition, Pramasa et al. conducted a comparative evaluation of several deep-learning architectures for BISINDO recognition, including InceptionV3, MobileNetV2, and ResNet50 [3]. These studies provide evidence that architecture selection and environmental conditions can affect recognition performance. However, they do not specifically isolate the effect of model capacity within the YOLOv8 family under controlled experimental conditions.

YOLO-based approaches have also been investigated for sign-language recognition in other linguistic contexts. Alaftekin et al. developed a real-time sign-language recognition system using a YOLO-based architecture for Turkish Sign Language [11]. Alsharif et al. applied YOLOv8 to American Sign Language alphabet recognition and reported high detection performance [12], while Al Ahmadi et al. investigated an efficient YOLO-based model

International Journal of Informatics and Data Science
ISSN 3026-7315 (media online)
Volume 3, No 2, June 2026, Page: 89–99
DOI: doi.org/10.64366/ijids.v3i2.604
https://journals.adaresearch.or.id/ijids/index

for Arabic Sign Language recognition [13]. These studies demonstrate the applicability of YOLO architectures to sign-language detection across different languages, but they do not establish whether increasing model capacity within YOLOv8 necessarily provides proportional accuracy improvements.

Based on these studies, a research gap remains regarding whether increasing model capacity within the YOLOv8 family provides a meaningful performance improvement for static BISINDO alphabet detection when other experimental factors are controlled. This question is particularly relevant because a larger model may offer greater representational capacity while simultaneously imposing higher computational costs. A controlled comparison is therefore required to determine whether the additional capacity of larger YOLOv8 variants translates into measurable improvements in detection performance.

To address this gap, this study comparatively evaluates YOLOv8n, YOLOv8s, YOLOv8m, and YOLOv8l under consistent experimental conditions for static BISINDO alphabet detection. The evaluation considers precision, recall, mAP@0.5, mAP@0.5:0.95, and training time, supported by three repeated deterministic executions and held-out test evaluation. The contributions of this study are threefold: (1) a controlled comparison of YOLOv8 model capacities for static BISINDO alphabet detection; (2) an empirical analysis of detection accuracy relative to training cost; and (3) additional evidence of reproducibility and within-dataset generalization, while acknowledging that same-source testing cannot establish cross-dataset generalization or completely exclude collection-specific bias.

2. RESEARCH METHODOLOGY

2.1 Research Stages

This study employed a controlled experimental design to investigate the effect of YOLOv8 model capacity on static BISINDO alphabet detection. The research procedure consisted of four main stages: data collection, preprocessing, model training, and evaluation. The same experimental procedure was applied to YOLOv8n, YOLOv8s, YOLOv8m, and YOLOv8l. Model capacity was therefore treated as the primary independent variable, while the dataset, preprocessing procedure, training configuration, computational environment, and evaluation protocol were kept consistent across all model variants. The overall experimental procedure is illustrated in Figure 1.

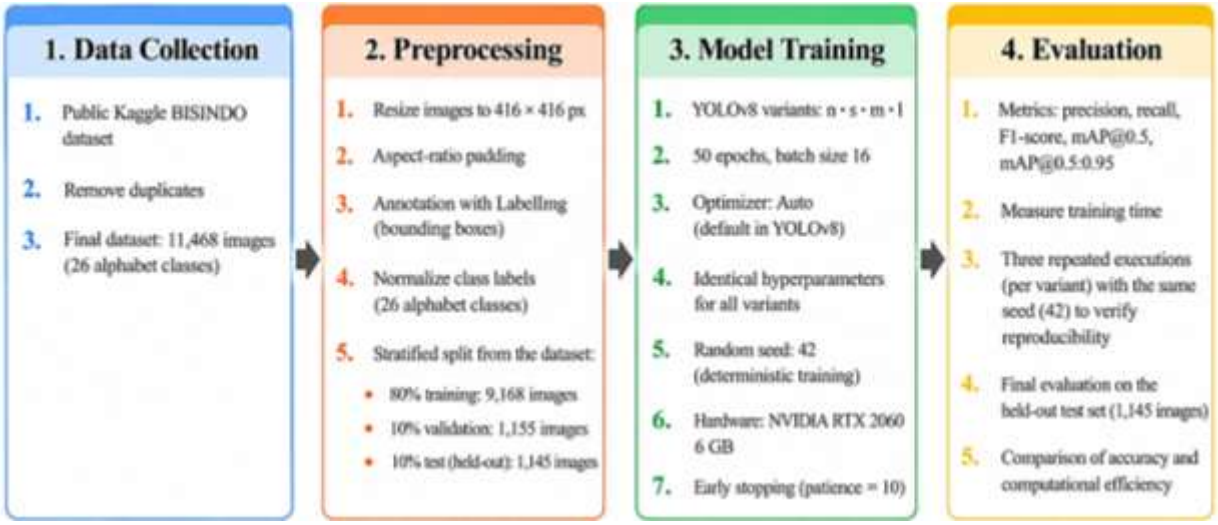


Figure 1. Experimental pipeline. The same four stages are applied to every YOLOv8 variant, with model capacity as the primary independent variable.

2.2 Dataset Collection and Preparation

The dataset used in this study was obtained from a publicly available BISINDO dataset hosted on Kaggle [14]. The dataset contains images representing the 26 alphabet classes of BISINDO, with each class corresponding to a specific hand configuration. Before the dataset was used in the experiment, duplicate images were removed to prevent identical samples from being assigned to different subsets and to reduce the possibility of direct data leakage.

After the data preparation process, the final dataset consisted of 11,468 images representing 26 BISINDO alphabet classes. The images were prepared for object detection using bounding-box annotations, with the same 26 class definitions maintained throughout the preprocessing, training, validation, and testing processes.

The visual characteristics of the dataset are illustrated through representative samples in Figure 2, which show variations in hand position, illumination, and background across the collected images. These variations provide representations of signing conditions and support robust evaluation across different image characteristics.

International Journal of Informatics and Data Science
ISSN 3026-7315 (media online)
Volume 3, No 2, June 2026, Page: 89–99
DOI: doi.org/10.64366/ijids.v3i2.604
https://journals.adaresearch.or.id/ijids/index

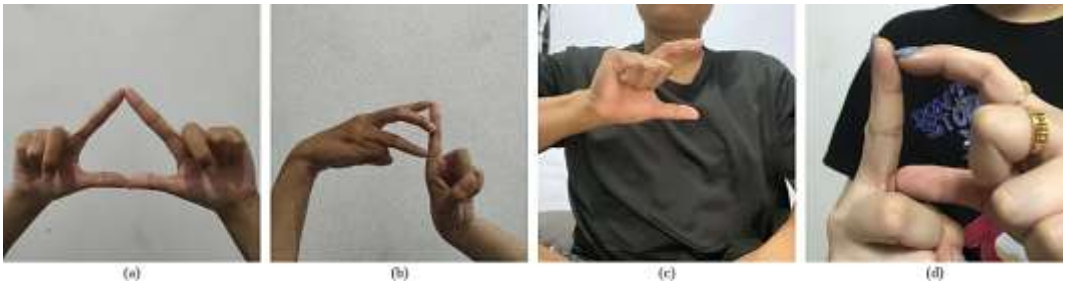


Figure 2. Representative images from the dataset, showing variation in hand position, illumination, and background across (a) to (d).

2.3 Preprocessing and Dataset Splitting

The images were prepared using an input resolution of 416 × 416 pixels. Aspect-ratio padding was applied to fit the images to the target resolution while maintaining their original proportions. The object annotations were represented using bounding boxes, and the class labels were normalized according to the 26 BISINDO alphabet classes.

Following preprocessing, the dataset was divided using a stratified splitting procedure to maintain a relatively consistent class distribution across the resulting subsets. The final dataset consisted of 9,168 training images, 1,155 validation images, and 1,145 held-out test images, corresponding approximately to 80%, 10%, and 10% of the complete dataset, respectively. The resulting class distribution across the three subsets is presented in Figure 3.

Figure 3. Class distribution across the training, validation, and held-out test subsets.

As shown in Figure 3, the class distribution remains relatively consistent across the three subsets, indicating that the stratified splitting procedure preserved the representation of the 26 alphabet classes. The detailed distribution of the dataset subsets is summarized in Table 1.

Table 1. Dataset composition and partitioning

Dataset Subset	Number of Images	Proportion
Training	9,168	≈80%
Validation	1,155	≈10%
Held-out test	1,145	≈10%
Total	11,468	100%

As summarized in Table 1, the training subset was used to optimize the model parameters, while the validation subset was used to monitor model performance during training. The 1,145 image held-out test subset was excluded from the training and validation processes and was reserved for the final evaluation.

Because all three subsets originated from the same public dataset source, the test subset is referred to as a held-out test subset rather than an independent external test dataset. Consequently, the test evaluation measures generalization to previously unseen samples from the same dataset source but does not establish cross-dataset generalization. No separate offline data augmentation stage was introduced in the preprocessing pipeline; therefore, all four YOLOv8 variants were trained and evaluated under the same dataset preparation and splitting conditions.

International Journal of Informatics and Data Science
ISSN 3026-7315 (media online)
Volume 3, No 2, June 2026, Page: 89–99
DOI: doi.org/10.64366/ijids.v3i2.604
https://journals.adaresearch.or.id/ijids/index

2.4 YOLOv8 Model Variants and Training Configuration

Four variants within the YOLOv8 family were evaluated: YOLOv8n, YOLOv8s, YOLOv8m, and YOLOv8l. These variants represent progressively increasing model capacity within the same YOLOv8 architecture family. The comparison was designed to determine whether increasing model capacity provides a meaningful improvement in static BISINDO alphabet detection. All four variants were trained under the same experimental configuration. Each model was trained for 50 epochs with a batch size of 16 and an input resolution of 416 × 416 pixels. The Auto optimizer configuration provided by YOLOv8 was used consistently across all model variants. The random seed was fixed at 42, and deterministic execution was enabled. The experiments were conducted on the same NVIDIA RTX 2060 GPU with 6 GB of VRAM. The hardware and principal training conditions were kept unchanged across the four variants so that differences in detection and computational performance could be evaluated under the same experimental environment. The complete training configuration is summarized in Table 2.

Table 2. Experimental Configuration

Parameter	Configuration
Model variants	YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l
Input resolution	416 × 416 pixels
Batch size	16
Training epochs	50
Optimizer	Auto
Random seed	42
Deterministic execution	Enabled
Hardware	NVIDIA RTX 2060 6 GB
Early stopping	Patience = 10

As shown in Table 2, the same principal training configuration was applied to all four model variants. This consistency was maintained to ensure that observed differences in detection performance and computational requirements could be primarily associated with differences in model capacity.

2.5 Experimental Execution and Evaluation

Each YOLOv8 variant was executed three times using the same experimental configuration, including the random seed of 42 and deterministic execution setting. The purpose of these repeated executions was to verify the reproducibility of the implemented training and evaluation pipeline. The three executions produced identical evaluation results for each model variant. Because the same random seed was used for all executions, these repetitions are interpreted as deterministic reproducibility checks rather than statistically independent trials with different random initializations. Therefore, they are not used to establish statistical significance across random seeds.

During training, the validation subset was used to monitor model performance and training behavior. Training and validation metrics were recorded throughout the training process. After training was completed, the final trained models were evaluated using the 1,145-image held-out test subset, which was not used for model optimization or model selection.

The detection performance of each model was assessed using precision, recall, F1-score, mAP@0.5, and mAP@0.5:0.95. Precision measures the proportion of predicted detections that are correct, while recall measures the proportion of actual objects that are successfully detected. The F1-score represents the harmonic mean of precision and recall and is defined as

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (1)$$

The mAP@0.5 metric evaluates detection performance at an Intersection over Union (IoU) threshold of 0.5. In contrast, mAP@0.5:0.95 provides a stricter evaluation by averaging Average Precision across IoU thresholds from 0.50 to 0.95 at intervals of 0.05. This metric provides additional information regarding bounding-box localization quality.

The held-out test evaluation provides an assessment of model performance on previously unseen samples from the same dataset source. However, because the test subset originates from the same public dataset as the training and validation subsets, high test performance cannot by itself establish cross-dataset generalization or completely exclude source-specific dataset bias.

2.6 Computational Analysis

In addition to detection performance, the computational characteristics of each YOLOv8 variant were analyzed using training time. This measurement was used to examine the computational implications of increasing model capacity.

The comparison focused on whether the additional capacity of YOLOv8s, YOLOv8m, and YOLOv8l produced measurable improvements in detection performance relative to YOLOv8n and how those changes affected

International Journal of Informatics and Data Science
ISSN 3026-7315 (media online)
Volume 3, No 2, June 2026, Page: 89–99
DOI: doi.org/10.64366/ijids.v3i2.604
https://journals.adaresearch.or.id/ijids/index

computational requirements. Rather than assuming that a larger model is inherently superior, the analysis evaluates the relationship between model capacity, detection accuracy, and computational efficiency under the same experimental conditions. The resulting analysis provides the basis for determining the most appropriate YOLOv8 variant for the static BISINDO alphabet detection task investigated in this study.

3. RESULTS AND DISCUSSION

This section presents the comparative performance of the four YOLOv8 variants, their training cost and convergence behaviour, localization and class-level errors, and the implications of the results for model selection in BISINDO alphabet detection. All models were trained under identical experimental conditions, and the reported results were obtained from the retained training logs and evaluation results.

3.1 Detection Accuracy

Table 3 summarizes the final validation performance of YOLOv8n, YOLOv8s, YOLOv8m, and YOLOv8l after 50 training epochs.

Table 3. Final detection performance of the four YOLOv8 variants

Variant	Precision	Recall	mAP@0.5	mAP@0.5:0.95	F1-score	Training time (h)
YOLOv8n	0.9905	0.9966	0.9942	0.9360	0.9935	0.93
YOLOv8s	0.9904	0.9971	0.9946	0.9356	0.9937	1.25
YOLOv8m	0.9892	0.9908	0.9938	0.9368	0.9900	2.32
YOLOv8l	0.9874	0.9931	0.9940	0.9355	0.9902	3.59

The results show that all four variants achieved highly similar detection performance. Precision ranges from 0.9874 to 0.9905, recall from 0.9908 to 0.9971, and mAP@0.5 from 0.9938 to 0.9946. The spread in mAP@0.5 is therefore less than 0.001, indicating that increasing the backbone capacity does not produce a measurable improvement in detection accuracy under the evaluated conditions.

The ranking also changes across evaluation metrics. YOLOv8s achieves the highest mAP@0.5 of 0.9946 and the highest F1-score of 0.9937, whereas YOLOv8m achieves the highest mAP@0.5:0.95 of 0.9368. YOLOv8n, despite being the smallest variant, obtains the highest precision at 0.9905. These results indicate that no single larger model consistently dominates the smaller variants across all accuracy measures.

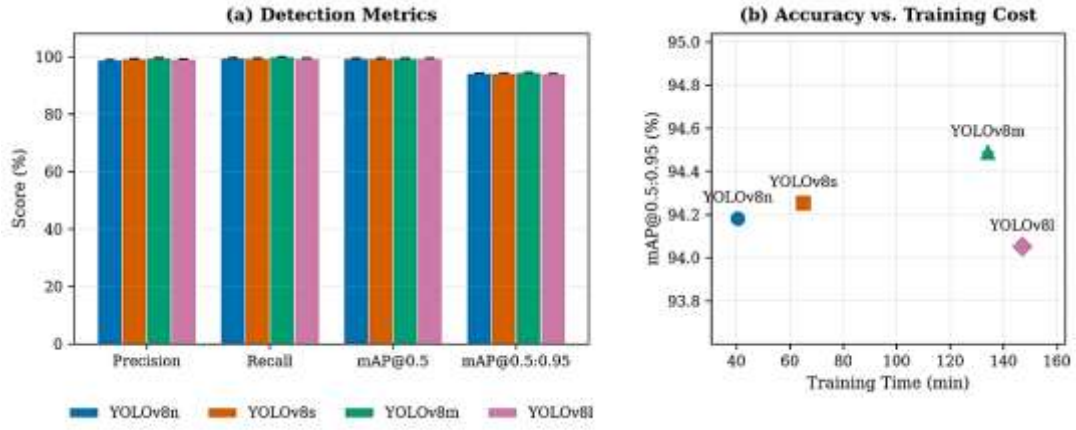


Figure 4. Comparative detection performance of the four YOLOv8 variants.

Figure 4 illustrates the small differences among the four architectures. The nearly overlapping performance values suggest that the BISINDO alphabet detection task does not require substantially larger backbone capacity to achieve high detection accuracy.

The held-out test partition provides an additional evaluation of the model's generalization performance. The test set contains 1,145 evaluated images in the reported run. YOLOv8n, YOLOv8s, YOLOv8m, and YOLOv8l achieved mAP@0.5 values of 0.99430, 0.99398, 0.99464, and 0.99489, respectively. Their corresponding mAP@0.5:0.95 values were 0.94287, 0.94468, 0.94593, and 0.94342. The validation-to-test differences in mAP@0.5 were only 0.00034, 0.00059, 0.00036, and 0.00044 in absolute value for YOLOv8n, YOLOv8s, YOLOv8m, and YOLOv8l, respectively. For mAP@0.5:0.95, the absolute differences were 0.00105, 0.00215, 0.00105, and 0.00291, respectively.

The small validation-to-test differences provide evidence that the high validation performance is not accompanied by a substantial performance collapse on the held-out test partition. However, these results should still be interpreted within the scope of the present dataset and experimental conditions rather than as evidence of universal generalization to all BISINDO signing environments.

International Journal of Informatics and Data Science
ISSN 3026-7315 (media online)
Volume 3, No 2, June 2026, Page: 89–99
DOI: doi.org/10.64366/ijids.v3i2.604
https://journals.adaresearch.or.id/ijids/index

3.2 Training Cost and Efficiency

Although the four variants exhibit almost identical detection accuracy, their computational costs differ substantially. YOLOv8n completed the 50-epoch training process in 0.93 h, whereas YOLOv8s, YOLOv8m, and YOLOv8l required 1.25, 2.32, and 3.59 h, respectively. The difference becomes more apparent when training time is considered relative to the smallest model. The per-epoch training times were 66.8 s, 90.2 s, 167.3 s, and 258.7 s for YOLOv8n, YOLOv8s, YOLOv8m, and YOLOv8l, corresponding to relative computational costs of 1.00×, 1.35×, 2.50×, and 3.87×, respectively.

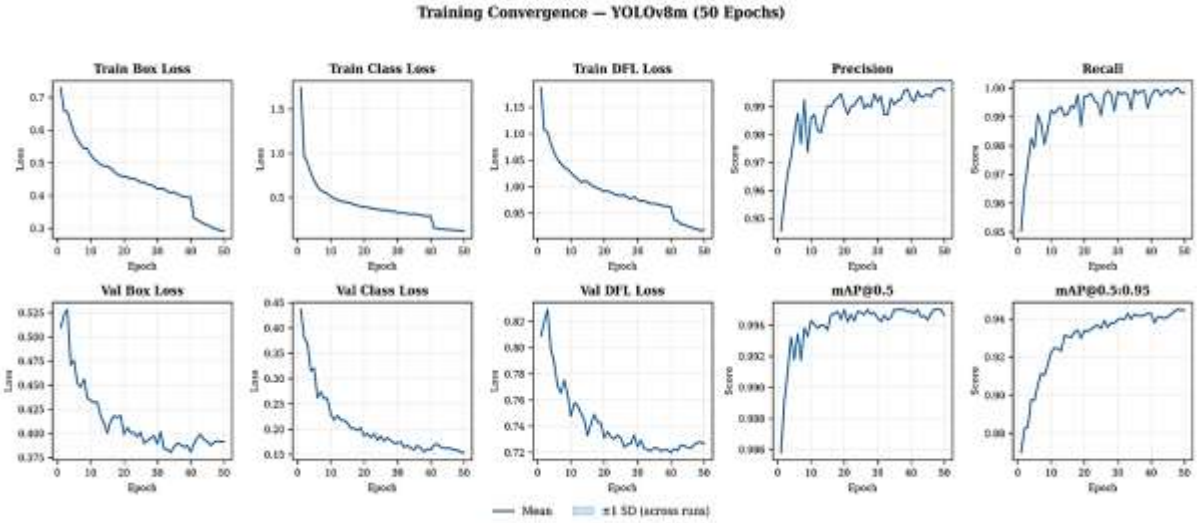


Figure 5. Training convergence of the YOLOv8 variants across 50 epochs.

The convergence analysis shown in Figure 5 indicates that all four variants reached 99.5% of their final mAP@0.5 before the end of the training schedule. YOLOv8s reached this level at epoch 24, YOLOv8n at epoch 29, and YOLOv8m and YOLOv8l at epoch 30. Thus, the difference between the fastest and slowest models in reaching this performance level is only six epochs, despite the much larger difference in total training time.

The validation losses continued to decrease through the end of training, with the minimum box and classification validation losses occurring around epochs 49–50. No sustained increase in validation loss was observed, providing no clear evidence of overfitting within the adopted 50-epoch training schedule. The standard deviation of mAP@0.5 during epochs 40–50 was 0.00088 for YOLOv8n, 0.00056 for YOLOv8s, 0.00094 for YOLOv8m, and 0.00063 for YOLOv8l, indicating stable late-stage behaviour.

A notable change occurs between epochs 40 and 41 across all variants. Training box loss decreases by approximately 17.7–18.8% at this point. This discontinuity corresponds to the scheduled deactivation of mosaic augmentation during the final ten epochs rather than a sudden convergence event. Following this change, mAP@0.5:0.95 increases by approximately 0.0146–0.0201 during the final ten epochs. This observation is important because the improvement during the final training stage is larger than the performance difference among the four backbone variants. It suggests that the training strategy and augmentation schedule can have a greater influence on localization quality than simply increasing model capacity.

3.3 Localization and Class-Level Error Analysis

Despite the very high mAP@0.5 values, a consistent difference is observed between mAP@0.5 and mAP@0.5:0.95. Averaged across the four variants, mAP@0.5 reaches 0.9942, whereas mAP@0.5:0.95 reaches 0.9360, producing an absolute difference of 0.0582. The corresponding gaps are 0.0582 for YOLOv8n, 0.0590 for YOLOv8s, 0.0570 for YOLOv8m, and 0.0585 for YOLOv8l.

This relatively stable gap indicates that the remaining performance limitation is not strongly dependent on backbone size. mAP@0.5 evaluates detections using a relatively permissive IoU threshold, whereas mAP@0.5:0.95 evaluates localization under progressively stricter IoU thresholds. Consequently, a detection can be classified correctly while still receiving a lower score when the predicted bounding box does not closely match the ground-truth boundary. The result therefore suggests that classification is substantially easier than precise localization in the present task. This interpretation is further supported by the class-level recall results. Moreover, the persistent difference between these metrics suggests that future improvements should prioritize accurate bounding box regression and annotation quality for visually similar hand configurations that may remain difficult to localize.

The confusion matrix on the held-out test set indicates that 22 of the 26 BISINDO alphabet classes achieve a recall of 1.00. The lowest recall values are observed for the N and M classes, with recall values of 0.94 and 0.97, respectively. The most prominent inter-class confusion occurs between these two visually similar configurations. Approximately 6% of true N instances are classified as M, whereas approximately 3% of true M instances are

International Journal of Informatics and Data Science
ISSN 3026-7315 (media online)
Volume 3, No 2, June 2026, Page: 89–99
DOI: doi.org/10.64366/ijids.v3i2.604
https://journals.adaresearch.or.id/ijids/index

classified as N. To further examine this error pattern, representative misclassification cases were analyzed from the experimental predictions.

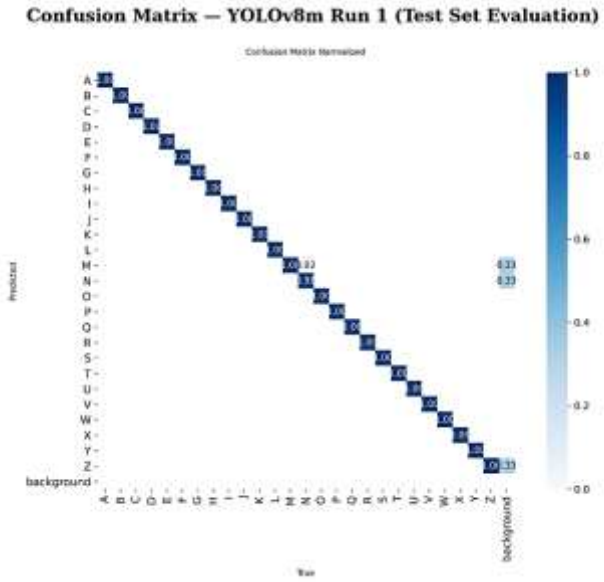
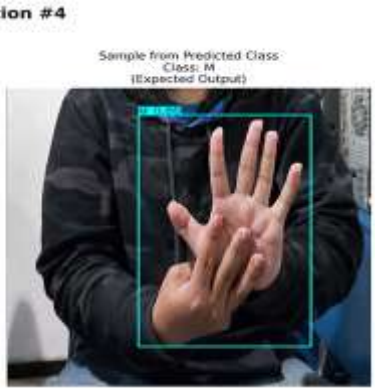


Figure 6. Normalized confusion matrix of YOLOv8m on the held-out test set.

Table 4. Representative misclassification cases in which the ground-truth BISINDO letter N is predicted as M

Sample Misclassification	Expected output
Incorrect Prediction #1	
	
Incorrect Prediction #2	
	
Incorrect Prediction #3	
	

International Journal of Informatics and Data Science
ISSN 3026-7315 (media online)
Volume 3, No 2, June 2026, Page: 89–99
DOI: doi.org/10.64366/ijids.v3i2.604
https://journals.adaresearch.or.id/ijids/index

Sample Misclassification	Expected output
	

The representative cases in Table 4 show that the model consistently predicts several N gestures as M, with confidence values of 0.856, 0.828, 0.690, and 0.687. Although the samples differ in signer appearance, hand position, and capture conditions, the hand configurations exhibit overlapping fingers and similar spatial arrangements. These characteristics may reduce the visual separability between the N and M classes and contribute to the observed confusion. The confusion is also asymmetric, with the proportion of N instances classified as M being approximately twice that of M instances classified as N. This pattern suggests that some visual representations of the N gesture are more likely to be interpreted as the M configuration. The observation is supported by both the quantitative confusion matrix and the representative misclassification cases. Other class-level errors are comparatively limited, with background false positives concentrated primarily on M (0.29), Z (0.25), and N (0.21).

These findings provide a more detailed interpretation of the high overall detection scores. The models are highly effective at distinguishing most BISINDO alphabet classes, but the remaining errors are concentrated around specific visually similar configurations. Therefore, further improvement may be obtained more effectively through improved representation of difficult hand configurations, annotation consistency, and targeted data collection than through simply increasing backbone capacity.

3.4 Accuracy, Computational Cost, and Model Selection

The accuracy and computational results reveal a clear difference between model capacity and practical training efficiency. YOLOv8n requires only 0.93 h to complete the training schedule, compared with 3.59 h for YOLOv8l. This represents approximately a 3.9-fold increase in training time, while the difference in mAP@0.5 between the two models is less than 0.0002.

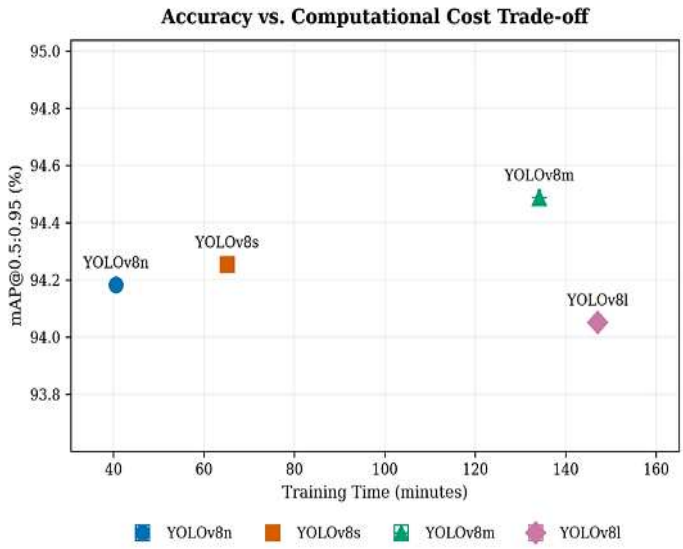


Figure 7. Accuracy–computational cost trade-off.

The same relationship can be expressed using training efficiency. YOLOv8n obtains an mAP@0.5-per-training-hour ratio of 1.069, whereas YOLOv8l obtains only 0.277, giving YOLOv8n approximately 3.9 times higher training efficiency under the experimental configuration. The results therefore do not support the assumption that a larger YOLOv8 backbone necessarily provides better BISINDO alphabet detection. YOLOv8m and YOLOv8l require substantially more computational resources while providing no consistent improvement in the primary

ISSN 3026-7315 (media online)
Volume 3, No 2, June 2026, Page: 89–99
DOI: doi.org/10.64366/ijids.v3i2.604
https://journals.adaresearch.or.id/ijids/index

detection metrics. YOLOv8s provides the highest mAP@0.5 at 0.9946, but the improvement over YOLOv8n at 0.9942 is only 0.0004, while its training time increases from 0.93 h to 1.25 h.

Therefore, under the evaluated dataset and hardware configuration, YOLOv8n provides the most favourable balance between detection performance and computational cost. YOLOv8s may be considered when the marginal improvement in mAP@0.5 is preferred, but the gain is small relative to the additional computational cost. The evidence does not provide a strong accuracy-based justification for selecting YOLOv8m or YOLOv8l. It is important to clarify that this conclusion concerns training cost and detection accuracy, not inference latency. The experiment did not directly measure end-to-end inference latency or deployment memory consumption. Consequently, real-time deployment performance should be evaluated separately on the intended target device.

Overall, the experiments demonstrate that increasing YOLOv8 model capacity from nano to large does not yield a proportional improvement in BISINDO alphabet detection accuracy. The performance differences remain very small, while the computational cost increases substantially. The error analysis further shows that the remaining classification errors are concentrated mainly around the visually similar N and M configurations, while the persistent difference between mAP@0.5 and mAP@0.5:0.95 indicates that precise bounding-box localization remains an important area for improvement.

3.5 Discussion

Previous studies have demonstrated the feasibility of YOLO-based object detection for Indonesian Sign Language (BISINDO) recognition. Setiawan et al. applied YOLOv8 for BISINDO alphabet detection [15]. Munandar et al. investigated YOLOv5 for BISINDO alphabet detection, while Silalahi et al. developed a mobile BISINDO translation application using YOLOv5 [16], [17]. Farhana et al. subsequently investigated a BISINDO sign-language interpreter using YOLOv8 combined with CNN [18]. These studies establish the applicability of YOLO-based architectures for BISINDO recognition, but they primarily focus on a specific architecture or application rather than systematically examining model capacity within the same YOLOv8 family.

BISINDO detection has also been investigated in video-based and alternative YOLO configurations. Renaningtias et al. evaluated YOLOv7 for BISINDO alphabet detection in video using 26 alphabet classes [19], while Kinanti et al. investigated BISINDO alphabet object detection using a YOLO-based deep-learning architecture [20]. More recently, Sahtio et al. evaluated and compared different YOLO11 variants for BISINDO detection, demonstrating the growing interest in systematic evaluation of YOLO model variants for Indonesian Sign Language recognition [21].

In contrast to these studies, the present work performs a controlled comparison of four YOLOv8 variants, namely YOLOv8n, YOLOv8s, YOLOv8m, and YOLOv8l. All variants were evaluated using the same dataset, training procedure, input configuration, and evaluation protocol. The results show that the four models achieve highly similar detection performance, with mAP@0.5 values of 0.9942, 0.9946, 0.9938, and 0.9940, respectively, while mAP@0.5:0.95 values are 0.9360, 0.9356, 0.9368, and 0.9355. The difference in mAP@0.5 between the highest and lowest performing variants is therefore only 0.0008, indicating that increasing backbone capacity does not necessarily provide a substantial accuracy improvement under the evaluated conditions.

The computational results provide an additional perspective for model selection. Training time increases from 0.93 h for YOLOv8n to 1.25 h, 2.32 h, and 3.59 h for YOLOv8s, YOLOv8m, and YOLOv8l, respectively. The corresponding average training times per epoch are 66.8 s, 90.2 s, 167.3 s, and 258.7 s. Thus, YOLOv8l requires approximately 3.9 times the training time of YOLOv8n, while providing no substantial improvement in detection performance. These findings indicate that increasing model capacity alone is not an efficient strategy for improving BISINDO alphabet detection under the evaluated conditions.

Finally, the class-level analysis provides additional insight beyond the aggregate performance metrics. The confusion matrix shows that the principal classification errors occur between the M and N classes, with approximately 6% of N instances predicted as M and 3% of M instances predicted as N. Representative incorrect predictions further indicate that these errors are associated with visually similar hand configurations, particularly in finger arrangement and overlap. Therefore, future improvement may be more effectively pursued through additional representative samples and improved feature discrimination for visually similar gestures rather than simply increasing model capacity. Overall, this study extends previous BISINDO research by providing a controlled within-family comparison of YOLOv8 variants and examining the relationship between model capacity, detection performance, computational cost, and class-specific errors.

4. CONCLUSION

This study evaluated four YOLOv8 variants, namely YOLOv8n, YOLOv8s, YOLOv8m, and YOLOv8l, for static BISINDO alphabet detection under a controlled experimental configuration. The results demonstrate that all four variants achieved consistently high detection performance, with validation mAP@0.5 values of 0.9942, 0.9946, 0.9938, and 0.9940, respectively, and mAP@0.5:0.95 values of 0.9360, 0.9356, 0.9368, and 0.9355. The small differences among these values indicate that increasing YOLOv8 model capacity does not provide a substantial improvement in detection accuracy for the evaluated BISINDO dataset. The computational analysis further shows

ISSN 3026-7315 (media online)
Volume 3, No 2, June 2026, Page: 89–99
DOI: doi.org/10.64366/ijids.v3i2.604
https://journals.adaresearch.or.id/ijids/index

that the increase in model capacity results in a considerably higher training cost. Training time increased from 0.93 h for YOLOv8n to 3.59 h for YOLOv8l, while the corresponding per-epoch training time increased from 66.8 s to 258.7 s. Considering the small differences in detection performance, the results indicate that YOLOv8n provides the most favourable accuracy–computational cost balance among the evaluated variants. Therefore, a larger backbone should not be selected solely on the assumption that greater model capacity will produce better BISINDO alphabet detection performance. The class-level analysis provides an additional finding regarding the remaining detection errors. Although most alphabet classes were detected with high recall, the main classification errors were concentrated between the M and N classes. Approximately 6% of N instances were predicted as M, while approximately 3% of M instances were predicted as N. Representative incorrect predictions showed similar visual characteristics in the hand configurations, particularly in finger arrangement and overlap. This indicates that further improvement may be more effectively achieved through additional representative samples and improved discrimination of visually similar hand configurations rather than simply increasing model capacity. Overall, this study contributes a controlled within-family evaluation of YOLOv8 variants for BISINDO alphabet detection and provides empirical evidence regarding the relationship between model capacity, detection accuracy, computational cost, and class-specific errors. The findings are limited to the evaluated dataset and experimental configuration; therefore, they do not establish cross-dataset or real-world generalization. Future research should investigate more diverse BISINDO datasets, additional signers and acquisition environments, targeted data collection for difficult classes such as M and N, and direct evaluation of inference latency and memory consumption on the intended deployment hardware.

5. DECLARATIONS

5.1 Author Contributions

Conceptualization: Y.T.B;
Methodology: Y.T.B;
Software: Y.T.B;
Validation: Y.P;
Formal Analysis: Y.P, R.E.S;
Investigation: Y.T.B, M.F, A.S;
Resources: M.F, A.S;
Data Curation: M.F;
Writing Original Draft Preparation: Y.T.B;
Writing Review and Editing: R.E.S;
Visualization: A.S;
All authors have read and agreed to the published version of the manuscript.

5.2 Data Availability

The dataset used in this study is derived from a publicly available Indonesian Sign Language (BISINDO) dataset hosted on Kaggle. The dataset contains images representing 26 BISINDO alphabet classes. The dataset used in the experiments consisted of 11,468 images after duplicate removal and preprocessing. The dataset is publicly accessible, while the processed data and experimental materials used in this study are available from the corresponding author upon reasonable request.

5.3 Funding

This research received no external funding.

5.4 Institutional Review Board Statement

Not applicable.

5.5 Informed Consent Statement

Not applicable.

5.6 Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

5.7 Generative AI and AI-assisted Technologies in The Writing Process

The authors used Generative AI to improve the writing clarity of this paper. They reviewed and edited the AI-assisted content and took full responsibility for the final publication. Thankyou to those who have supported the implementation of this research.

ISSN 3026-7315 (media online)
Volume 3, No 2, June 2026, Page: 89−99
DOI: doi.org/10.64366/ijids.v3i2.604
https://journals.adaresearch.or.id/ijids/index

REFERENCES

[1] K. K. Candra and K. Kusrini, "Klasifikasi Gambar Bahasa Isyarat Indonesia (BISINDO) pada Komunitas Tuli Menggunakan Machine Learning," *e-Jurnal JUSITI (Jurnal Sistem Informasi dan Teknologi Informasi)*, vol. 14, no. 1, pp. 56–63, 2025, doi: 10.36774/jusiti.v14i1.1649.

[2] E. L. Kelana, M. R. A. Prasetya, Mambang, and M. Zulfadhilah, "Integrating the CNN Model with the Web for Indonesian Sign Language (BISINDO) Recognition," *Journal of Applied Informatics and Computing*, vol. 9, no. 3, pp. 883–896, 2025, doi: 10.30871/jaic.v9i3.9345.

[3] A. T. Pramasa, N. P. Sutramiani, I. P. A. Bayupati, and I. W. A. S. Darma, "Extensive Deep Learning Models Evaluation for Indonesian Sign Language Recognition," *Lontar Komputer: Jurnal Ilmiah Teknologi Informasi*, vol. 16, no. 2, 2025, doi: 10.24843/LKJITI.2025.v16.i02.p04.

[4] J. Terven, D. M. Córdova-Esparza, and J. A. Romero-González, "A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS," *Mach. Learn. Knowl. Extr.*, vol. 5, no. 4, pp. 1680–1716, 2023, doi: 10.3390/make5040083.

[5] M. Hussain, "YOLOv1 to v8: Unveiling Each Variant-A Comprehensive Review of YOLO," *IEEE Access*, vol. 12, pp. 42816–42833, 2024, doi: 10.1109/ACCESS.2024.3378568.

[6] C. A. Sari, E. H. Rachmawanto, Z. Saifullah, C. Jatmoko, and D. Sinaga, "Real-time detection of indonesian sign language (ISL) gestures based on long short-term memory," *Journal of Soft Computing Exploration*, vol. 5, no. 3, pp. 251–262, 2024, doi: 10.52465/joscex.v5i3.452.

[7] A. Josef and G. P. Kusuma, "Alphabet Recognition in Sign Language Using Deep Learning Algorithm with Bayesian Optimization," *Revue d'Intelligence Artificielle*, vol. 38, no. 3, pp. 929–938, 2024, doi: 10.18280/ria.380319.

[8] M. Maheza Fresmanda, Istiadi, and Syahroni Wahyu Iriananda, "Deteksi Objek Video Bahasa Isyarat Untuk Anak Tuna Rungu dan Tuna Wicara Menggunakan YOLOv8," *Jurnal Komputer, Informasi dan Teknologi*, vol. 4, no. 2, pp. 1–9, 2024, doi: 10.53697/jkomitek.v4i2.1895.

[9] M. R. Ningsih *et al.*, "Sign Language Detection System Using YOLOv5 Algorithm to Promote Communication Equality People with Disabilities," *Scientific Journal of Informatics*, vol. 11, no. 2, pp. 549–558, 2024, doi: 10.15294/sji.v11i2.6007.

[10] M. A. Saputra and E. Rakun, "Recognizing Indonesian Sign Language (BISINDO) Gesture in Complex Backgrounds," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 36, no. 3, pp. 1583–1593, 2024, doi: 10.11591/ijeecs.v36.i3.pp1583-1593.

[11] M. Alaftekin, I. Pacal, and K. Cicek, "Real-time sign language recognition based on YOLO algorithm," *Neural Comput. Appl.*, vol. 36, no. 14, pp. 7609–7624, 2024, doi: 10.1007/s00521-024-09503-6.

[12] B. Alsharif, E. Alalwany, and M. Ilyas, "Transfer learning with YOLOV8 for real-time recognition system of American Sign Language Alphabet," *Franklin Open*, vol. 8, no. October, p. 100165, 2024, doi: 10.1016/j.fraope.2024.100165.

[13] S. Al Ahmadi, F. Mohammad, and H. Al Dawsari, "Efficient YOLO-Based Deep Learning Model for Arabic Sign Language Recognition," *Journal of Disability Research*, vol. 3, no. 4, pp. 1–15, 2024, doi: 10.57197/jdr-2024-0051.

[14] A. Ma'ruf, "Indonesian Sign Language - BISINDO," 2023, *Kaggle*. [Online]. Available: https://www.kaggle.com/datasets/agungmrf/indonesian-sign-language-bisindo

[15] S. R. Andra, Munandar, Zara Yunizar, "Indonesian Sign Language (BISINDO) Alphabet Detection Using the You Only Look Once (YOLO) Algorithm Version 8," *International Conference on Computer, Control, Informatics and its Applications, IC3INA*, vol. 00001, no. 2024, pp. 388–393, 2024, doi: 10.1109/IC3INA64086.2024.10732209.

[16] A. Munandar, Z. Yunizar, and S. Retno, "Indonesian Sign Language (BISINDO) Alphabet Detection System Using YOLO (You Only Look Once) Algorithm," in *Proceedings of Malikussaleh International Conference on Multidisciplinary Studies (MICoMS)*, 2024. doi: 10.29103/micoms.v4i.952.

[17] A. V. F. Silalahi *et al.*, "Aplikasi Penerjemah Bahasa Isyarat BISINDO Menggunakan Metode YOLOv5 Berbasis Mobile," *Jurnal SPEKTRUM*, vol. 11, no. 3, 2024, doi: 10.24843/SPEKTRUM.2024.v11.i03.p2.

[18] F. Farhana, A. R. E. Najaf, and R. Permatasari, "BISINDO Sign Language Interpreter System Using YOLOv8 and CNN," *bit-Tech*, vol. 8, no. 1, 2025, doi: 10.32877/bt.v8i1.2638.

[19] N. Renaningtias, F. P. Utama, and A. N. A. Sobri, "Detection System Indonesian Sign Language (BISINDO) in Video with YOLOv7," *JSAI (Journal Scientific and Applied Informatics)*, vol. 8, no. 1, pp. 1–8, 2025, doi: 10.36085/jsai.v8i1.7067.

[20] A. Kinanti, M. A. Afandi, I. Permatasari, and N. Y. Tarigan, "Deteksi Objek Bahasa Isyarat Alfabet BISINDO Menggunakan Deep Learning dan Arsitektur YOLO," *Techno.Com: Jurnal Teknologi Informasi*, vol. 23, no. 2, pp. 409–417, 2024, doi: 10.62411/tc.v23i2.9889.

[21] D. E. P. Sahtio, M. A. P. Putra, and I. B. K. Sudiatmika, "Performance Evaluation and Comparison of YOLO11 Model Variants for Indonesian Sign Language (BISINDO)," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 14, no. 2, 2026, doi: 10.23960/jitet.v14i2.9348.

