JITET (Jurnal Informatika dan Teknik Elektro Terapan)
Vol. 14 No. 2, pISSN: 2303-0577 eISSN: 2830-7062 http://dx.doi.org/10.23960/jitet.v14i2.9348

PERFORMANCE EVALUATION AND COMPARISON OF YOLO11 MODEL VARIANTS FOR INDONESIAN SIGN LANGUAGE (BISINDO)

Dimas Eka Putra Sahtio1*, Made Adi Paramartha Putra2, Ida Bagus Kresna Sudiatmika3

[1,2,3]Informatics, Faculty of Information Technology and Design, Primakara University

Keywords:
BISINDO;
Deep Learning;
Hyperparameter Tuning;
Object Detection;
YOLO11

Corespondent Email:
adi@primakara.ac.id

Abstrak. Penelitian ini bertujuan untuk mengevaluasi dan membandingkan performa berbagai varian model You Only Look Once (YOLO) versi 11 dalam mendeteksi kata pada Bahasa Isyarat Indonesia (BISINDO). Komunikasi merupakan kebutuhan mendasar, namun individu dengan gangguan pendengaran sering menghadapi tantangan karena kurangnya pemahaman masyarakat terhadap bahasa isyarat. Penggunaan Artificial Intelligence (AI) dengan pendekatan Deep Learning, khususnya algoritma YOLO, memberikan solusi untuk menerjemahkan bahasa isyarat secara otomatis. Meskipun YOLOv8 telah digunakan sebelumnya, pembaruan ke YOLO11 menjanjikan peningkatan efisiensi dan akurasi. Dalam studi ini, hyperparameter tuning dilakukan untuk mendapatkan konfigurasi terbaik (batch size 16, lr0=0.001, lrf=0.1), yang kemudian digunakan untuk melatih dan membandingkan varian YOLO11 (YOLO11n, YOLO11s, YOLO11m) pada dataset berisi 36 kelas BISINDO. Hasilnya menunjukkan bahwa model YOLO11n memberikan performa paling optimal dalam hal keseimbangan antara akurasi dan efisiensi waktu, dengan nilai mAP50-95 sebesar 88,7%, durasi pelatihan 3,771 jam, dan kecepatan inferensi 3,6 ms. Varian YOLO11s memiliki akurasi deteksi yang sedikit lebih tinggi dengan mAP50-95 sebesar 89,2%, dengan waktu inferensi 5,351.

Abstract. *This study aims to evaluate and compare the performance of various versions of the You Only Look Once (YOLO) versionx 11 model in detecting words in Indonesian Sign Language (BISINDO). Communication is a fundamental need, but individuals with hearing impairments often face challenges due to the public's lack of understanding of sign language. The use of Artificial Intelligence (AI) with a Deep Learning approach, specifically the YOLO algorithm, provides a solution for automatically translating sign language. Although YOLOv8 has been used previously, the update to YOLO11 promises improvements in efficiency and accuracy. In this study, hyperparameter tuning was conducted to obtain the best configuration (batch size 16, lr0=0.001, lrf=0.1), which was then used to train and compare YOLO11 variants (YOLO11n, YOLO11s, YOLO11m) on a dataset of 36 BISINDO classes. The results showed that the YOLO11n model provided the most optimal performance in terms of balancing accuracy and time efficiency, with an mAP50-95 value of 88.7%, a training duration of 3.771 hours, and an inference speed of 3.6 ms. The YOLO11s variant had a slightly higher detection accuracy with an mAP50-95 of 89.2%, with 5.351 inference time.*

1. INTRODUCTION

Communication is a fundamental human need that facilitates the exchange of

information, emotional expression, and the building of harmonious social relationships. However, individuals with disabilities, particularly those who are deaf or hard of hearing, frequently encounter substantial challenges in daily communication due to the limitations of verbal language. Consequently, non-verbal communication, primarily sign language, serves as their primary medium of interaction. In Indonesia, two distinct sign language systems exist: the Indonesian Sign System (SIBI) and Indonesian Sign Language (BISINDO). BISINDO is significantly more prevalent within the deaf community as it developed naturally from their daily communicative habits, offering greater flexibility and expressiveness compared to the highly formalized, text-structured SIBI. Despite its widespread use among the deaf community, public comprehension of BISINDO remains critically low. Previous studies highlight that approximately 85% of the general population struggles to communicate effectively with deaf individuals, leading to frequent misunderstandings and social friction.

To bridge this communication gap, alternative technological methods powered by Artificial Intelligence (AI) and Deep Learning (DL) are essential. Deep Learning, specifically Convolutional Neural Networks (CNNs), has proven highly capable of processing visual data to recognize complex hand gestures. Among various DL architectures, the You Only Look Once (YOLO) algorithm stands out as a state-of-the-art object detection model that processes bounding box predictions and class probabilities simultaneously, enabling rapid and efficient single-stage detection [1].

While previous research has successfully utilized older iterations like YOLOv8 to detect BISINDO, Ultralytics' release of YOLO11 introduces remarkable advancements in feature extraction and processing efficiency. YOLO11 incorporates an upgraded backbone utilizing C3k2 blocks, which replace the older C2f blocks to capture more complex features across varying object scales and an advanced neck component featuring the C2PSA (Partial Self Attention) block to enhance global modeling efficiency without increasing computational time. Therefore, this research focuses on implementing and evaluating YOLO11 model variants for BISINDO word detection. By conducting comprehensive hyperparameter tuning and benchmarking YOLO11 against both its predecessors (YOLOv8) and successors (YOLO12), this study aims to identify the most optimal model that balances high detection accuracy with computational efficiency.

2. LITERATURE REVIEW

The application of Deep Learning for sign language recognition has seen significant exploration in recent years. A study by [2] evaluated the performance of various models for recognizing the SIBI alphabet. Their findings demonstrated that the YOLO architecture achieved the highest accuracy at 97%, outperforming SSD MobileNetV2 (86%) and Faster R-CNN (84%). The specific application of the YOLOv8 algorithm for BISINDO word detection was explored by [3] and [4], reported promising mAP50-90 scores of 93.1% and 88.4%, respectively, solidifying YOLOv8's viability for complex gesture recognition.

More recently, the introduction of YOLO11 has prompted comparative studies across different international sign languages. Researcher in [5] evaluated YOLO11 against YOLOv8, v9, v10, and 12 for Bangladeshi Sign Language (BdSL) and American Sign Language (ASL), finding that YOLO11 achieved a superior mAP50 of 99.4%, outperforming all other versions. Similarly, researcher in [6] compared multiple YOLO versions for Indian Sign Language (ISL), concluding that YOLO11 provided the best performance with an mAP50 of 97.8%. Despite these advancements, specific evaluations of YOLO11 variants (nano, small, medium) tailored to the BISINDO dataset, particularly involving rigorous hyperparameter tuning to optimize training stability and resource efficiency, have not yet been thoroughly explored. This gap forms the foundation of the current study.

3. METHOD

This research adopted the Cross-Industry Standard Process for Data Mining (CRISP-DM) methodology, encompassing Business Understanding, Data Understanding, Data Preparation, Modeling, and Evaluation phases.

3.1. Data Understanding

The dataset utilized in this study combined primary and secondary data sources. The primary dataset consisted of 348 videos (1920x1080 resolution) recorded directly by the researchers, capturing various BISINDO word gestures lasting between 3 to 7 seconds. The secondary dataset was sourced from a public Kaggle repository titled "BISINDO 40 Kata" by [7], containing 2,000 videos (400x300 resolution) with an average duration of 3 seconds. To prevent model confusion, classes with visually identical or highly overlapping gestures (e.g., 'Dimana' vs. 'Dia', 'Apa' vs. 'Membaca') were filtered. Following this adjustment, the dataset was refined to 36 distinct BISINDO word classes, resulting in 318 primary videos and 280 secondary videos. The frame extraction process successfully obtained 17,852 data by extracting each video by 8 frame per seconds to obtain optimal movement data [8]. The frame extraction from primary and secondary video detailed in Tabel 1.

3.2. Data Preparation

The dataset utilized in this study combined primary and secondary data sources. The primary dataset consisted of 348 videos (1920x1080 resolution) recorded directly by the

3.3. Modeling

In this work, we adopt You Only Look Once (YOLO) which known as a deep learning-based object detection model designed to detect and recognize objects in a single stage rapidly and efficiently. This model frames object detection as a regression problem, enabling the simultaneous processing of bounding box predictions and class probabilities [9]. The YOLO architecture consists of a backbone for feature extraction from the input image, a neck for combining and upsampling multi-scale feature maps, and a head for generating bounding box predictions and class probabilities. This architecture allows YOLO to perform fast and accurate object detection [10]. YOLO11, the latest model from Ultralytics released in September 2024, introduces exceptional capabilities in object detection, feature extraction, segmentation, pose estimation, tracking, and classification. Compares with YOLO8 [11], the performance of YOLO11 is significantly improved. By integrating new components, YOLO11 successfully enhances feature extraction capabilities, object detection accuracy, and real-time object detection compared to previous YOLO models. The architecture of the YOLO11 models depicted in Figure 1.

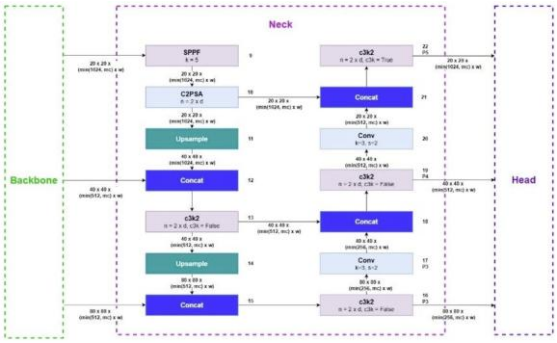

Figure 1. The architecture of the YOLO11 model architecture with Backbone, Neck and Head.

First, the backbone is a fundamental component within the YOLO architecture responsible for feature extraction from the input image. This process is achieved by stacking multiple convolutional blocks alongside specialized modules, such as the C3k2 block. The backbone performs downsampling, which progressively reduces the spatial resolution. The C3k2 block is utilized to facilitate feature extraction across multiple hierarchical levels.

The neck serves as a continuation of the backbone component. Within the neck, the Spatial Pyramid Pooling Fast (SPPF) block is utilized to capture spatial information across multiple scales, alongside the C2PSA block, which employs a spatial attention mechanism to enhance global modeling efficiency. This C2PSA block is exclusively applied at the stage with the lowest spatial resolution [12]. Furthermore, the neck incorporates several upsampling and concatenation operations, enabling the fusion of features from various resolution levels. This multi-scale feature fusion effectively supports the detection of objects of varying sizes.

Lastly, The head constitutes the final component responsible for generating predictions following the processing stages within the backbone and neck. This module processes the feature maps derived from the

neck to yield bounding box predictions and class probabilities [13].

In this work, the model training was conducted in a Kaggle Notebook environment powered by multi-GPU (T4 x2) support to allow parallel computation [14]. The Ultralytics framework was utilized to train the YOLO models. Training was standardized across 100 epochs using a batch size of 16 and the AdamW optimizer. The input image size was set to 800, and a multi-scale learning approach was enabled to allow the models to adapt to randomly varying image sizes during training. Additionally, a cosine learning rate scheduler (cos_lr=True) was implemented to dynamically and stably decay the learning rate from the initial to the final epoch.

4. RESULTS AND DISCUSSIONS

This section presents the comprehensive findings from the training and evaluation phases of the YOLO models applied to the BISINDO dataset. To systematically determine the most optimal architecture for sign language word detection, the experimental process was structured into several primary stages. First, a baseline evaluation was conducted without hyperparameter tuning to establish the initial performance metrics of the model. Second, rigorous hyperparameter tuning was applied to the baseline model to maximize learning stability, enhance accuracy, and reduce computational load. Finally, an intra-generational analysis was carried out across different YOLO11 scale variants (nano, small, and medium) to carefully evaluate the trade-offs between detection accuracy, resource consumption, and inference speed.

4.1. Baseline Model

Before executing the comprehensive hyperparameter tuning, an initial baseline evaluation was conducted to establish the model's default performance capabilities. In this unoptimized scenario, the YOLO11n (nano) model was trained for 100 epochs using a standard configuration: a batch size of 8, an initial learning rate (lr0) of 0.01, and a final learning rate (lrf) of 0.01. The results of this baseline training phase yielded a mean Average Precision (mAP50-95) score of 86.2%. Furthermore, the default configuration proved to be computationally demanding, requiring a total training duration of 4.953 hours. This baseline performance established the necessary benchmark to measure the effectiveness of the subsequent optimization efforts.

4.2. Optimized Performance

To enhance the model's accuracy and computational efficiency, rigorous hyperparameter tuning [15] was subsequently applied to the baseline YOLO11n model. Five different experimental scenarios were tested by manipulating the batch size (8 and 16) alongside the learning rates (lr0 and lrf). The evaluation results detailed in Table 3.

Table 3. Optimized Performance Results with Hyperparameter Tunning

Model	Hyperparameter				Results	
	Epoch	Batch	lr0	lrf	Train Time (h)	mAP5 0-95
Bisindo-v0-1 (yolo11 n)	100	8	0.01	0.01	4.953	86.2%
Bisindo-v0-2 (yolo11 n)	100	16	0.01	0.01	3.761	87%
Bisindo-v0-3 (yolo11 n)	100	16	0.001	0.01	3.658	88.3%
Bisindo-v0-4 (yolo11 n)	100	16	0.001	0.1	3.771	88.7%
Bisindo-v0-5 (yolo11 n)	100	16	0.01	0.1	3.659	86.9%

The initial tuning step increased the batch size from 8 to 16 while keeping the learning rates at their default values (lr0=0.01, lrf=0.01). This single adjustment not only reduced the training duration to 3.761 hours but also improved the mAP50-95 score to 87%. Following the establishment of 16 as the optimal batch size, variations in learning rates were systematically tested. Adjusting the rates to lr0=0.001 and lrf=0.01 further improved the accuracy score to 88.3%. Ultimately, the peak performance was discovered in the scenario where lr0=0.001 and lrf=0.1 were applied. This configuration successfully boosted the mAP50-95 to 88.7% while maintaining a highly efficient training duration of 3.771 hours. Compared to the baseline, the hyperparameter tuning process successfully increased overall accuracy by 2.5% while reducing the training

time by over an hour. This optimized configuration was established as the standard for all subsequent model evaluations.

4.3. Evalution Across YOLO11 Models

To further drill down into the capabilities of the YOLO11 architecture, three scale variants were evaluated: YOLO11n (nano), YOLO11s (small), and YOLO11m (medium). Larger models (large and extra-large) were excluded as their parameter weight exceeded the memory limitations of the Kaggle environment.

Table 4. Optimized Performance Results with Hyperparameter Tunning

Model	Hyperparameter				Results	
	Epoch	Batch	lr0	lrf	Train Time (h)	mAP5 0-95
Bisindo-v0-4 (yolo11n)	100	16	0.001	0.1	3.771	88.7%
Bisindo-v1-1 (yolo11s)	100	16	0.001	0.1	5.351	89.2%
Bisindo-v2-1 (yolo11m)	100	16	0.001	0.1	10.509	89.1%

Table 4 shows the results of various YOLO11 models. YOLO11n achieved an mAP50-95 of 88.7% with a highly efficient 3.771-hour training time and 3.6 ms inference speed. YOLO11s recorded a slight dip in Recall (99.5%) but achieved the highest overall mAP50-95 of 89.2%. This indicates superior precision across strict Intersection over Union (IoU) thresholds from 0.50 to 0.95. The trade-off was an increased training duration of 5.351 hours and an inference speed of 6.6 ms. Lastly, YOLO11m demonstrated the highest ability to detect all positive instances, scoring a peak Recall of 99.7% and an F1-Score of 99.7%. However, its mAP50-95 settled at 89.1%. The primary drawback of the medium variant was its massive resource consumption, requiring 10.509 hours to train and 16.7 ms per image for inference.

5. CONCLUSION

In this work, we address the fundamental communication barriers faced by individuals with hearing and speech impairments due to the public's limited understanding of Indonesian Sign Language (BISINDO). To bridge this communication gap, automated sign language detection systems utilizing Deep Learning offer a promising solution. However, successfully deploying these systems in real-world scenarios requires a delicate balance between high detection accuracy and computational efficiency. We solve this challenge by systematically optimizing and evaluating scale variants of the state-of-the-art YOLO11 architecture to identify the most effective and practical model for BISINDO word detection.

Based on the experimental results, this study demonstrates that rigorous hyperparameter optimization is critical for developing a robust detection system. The tuning process proved highly successful; by adjusting the batch size to 16 and learning rates to lr0=0.001 and lrf=0.1, the baseline YOLO11n model's overall accuracy (mAP50-95) increased by 2.5% while simultaneously reducing the training duration by over an hour. This optimized configuration established a strong and stable foundation for further architectural comparisons.

Furthermore, the intra-generational analysis of the YOLO11 variants revealed a clear trade-off between computational efficiency and strict detection accuracy. The YOLO11n (nano) variant emerged as the most structurally efficient model, achieving an mAP50-95 of 88.7% with a training time of 3.771 hours and a rapid inference speed of just 3.6 ms, making it the optimal choice for fast, resource-constrained applications. Conversely, if strict detection precision is the primary goal, the YOLO11s (small) variant serves as the recommended alternative. It achieved the highest overall mAP50-95 of 89.2%, though this comes at the cost of moderately increased training and inference times. In contrast, while the YOLO11m (medium) variant excelled in overall recall at 89.1%, its massive computational demands, requiring over 10 hours to train and 10.5 ms for inference, yielded diminishing returns, rendering it inefficient for practical deployment. Ultimately, the optimized YOLO11n and YOLO11s models present highly capable and practical solutions for BISINDO translation, allowing developers to prioritize either maximum processing speed or peak precision depending on their specific deployment constraints.

REFERENCE

[1] N. B. A. Karna, M. A. P. Putra, S. M. Rachmawati, M. Abisado and G. A. Sampedro, "Toward Accurate Fused Deposition Modeling 3D Printer Fault Detection Using Improved YOLOv8 With Hyperparameter Optimization," in IEEE Access, vol. 11, pp. 74251-74262, 2023, doi: 10.1109/ACCESS.2023.3293056.

[2] N. S. W. Nugroho and M. P. K. Putra, "Leveraging deep learning approach for accurate alphabet recognition through hand gestures in sign language," Jurnal Teknik Informatika (JUTIF), vol. 6, 2025.

[3] D. S. Ariansyah, "Pendeteksi Kata dalam Bahasa Isyarat Menggunakan Algoritma YOLO Versi 8," Jurnal Informatika dan Teknik Elektro Terapan, vol. 12, no. 3, 2024, doi: 10.23960/jitet.v12i3.4904.

[4] A. Pangestu, M. Muttaqin, and M. Sunandar, "Sistem Deteksi Bahasa Isyarat Indonesia (Bisindo) Menggunakan Algoritma You Only Look Once (YOLO)v8," JATI (Jurnal Mahasiswa Teknik Informatika), vol. 8, no. 5, pp. 9891–9897, 2024.

[5] N. Navin et al., "Bilingual Sign Language Recognition: A YOLOv11-Based Model for Bangla and English Alphabets," Journal of Imaging, vol. 11, no. 5, p. 134, 2025, doi: 10.3390/jimaging11050134.

[6] R. Raj, R. Sreemathy, M. Turuk, J. Jagdale, and M. Anish, "Performance Comparison of Different Versions of YOLO for Indian Sign Language Captioning in Real Time of Multiple Signers," Procedia Computer Science, vol. 259, pp. 991–1000, 2025, doi: 10.1016/j.procs.2025.04.053.

[7] A. Y. Pardede, "BISINDO 40 Kata mp4," 2024. [Online]. Available: https://www.kaggle.com/datasets/anggiyohanespardede/bisindo-40-kata-mp4

[8] G. Wiriasto, A. Rizaldy, P. Wiguna, and I. Kinasih, "Drip Infusion Monitoring and Data Logging System Based on YOLOv5," JITK (Jurnal Ilmu Pengetahuan Dan Teknologi Komputer), vol. 11, pp. 171–179, 2025.

[9] T. Diwan, G. Anirudh, and J. V. Tembhurne, "Object detection using YOLO: challenges, architectural successors, datasets and applications," Multimedia Tools and Applications, vol. 82, no. 6, pp. 9243–9275, 2023, doi: 10.1007/s11042-022-13644-y.

[10] P. Hidayatullah, N. Syakrani, M. R. Sholahuddin, T. Gelar, and R. Tubagus, "YOLOv8 to YOLO11: A Comprehensive Architecture In-depth Comparative Review," 2025. [Online]. Available: https://arxiv.org/pdf/2501.13400.

[11] R. Supriyadi et al., "Pengembangan Aplikasi Estimasi Kalori Makanan Berbasis Citra dengan Pendekatan Deteksi Objek Menggunakan YOLO," Jurnal Informatika dan Teknik Elektro Terapan, vol. 14, no. 1, 2026, doi: 10.23960/jitet.v14i1.8545.

[12] K. Wang, J. Liu, and X. Cai, "C2PSA-enhanced YOLOv11 architecture: A novel approach for small target detection in cotton disease diagnosis," arXiv preprint arXiv:2508.12219, 2025.

[13] D.-L. Nguyen, X.-T. Vo, A. Priadana, J. Choi, and K.-H. Jo, "Improved YOLOv11 Based on Attention and Extra Head Modules for Dental Disease Detection," in 2025 International Workshop on Intelligent Systems (IWIS), Ulsan, South Korea, 2025, pp. 1–6, doi: 10.1109/IWIS66215.2025.11142416.

[14] L. Quaranta, F. Calefato, and F. Lanubile, "KGTorrent: A Dataset of Python Jupyter Notebooks from Kaggle," in 2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR), Madrid, Spain, 2021, pp. 550–554, doi: 10.1109/MSR52588.2021.00072.

[15] M. A. P. Putra, I. Utama, N. W. Utami, and I. G. E. J. Putra, "Enhancing Federated Learning Performance through Adaptive Client Optimization with Hyperparameter Tuning," Journal of Applied Data Sciences, vol. 5, no. 2, pp. 747–755, 2024, doi: 10.47738/jads.v5i2.251.