Journal of Soft Computing Exploration
Homepage: htttps://shmpublisher.com/index.php/joscex
p-ISSN: 2746-7686
e-ISSN: 2746-0991

.

Indonesian sign language (BISINDO) gesture detection using Yolov11-Pose

Achmad Dany Alfansyah[1], Ardian Yusuf Wicaksono[2], Pima Hani Safitri[3]
[1,2,3]Department of Informatics, Telkom University Surabaya, Indonesia

Article Info

Article history:

Received April 26, 2026
Revised May 28, 2026
Accepted July 04, 2026

Keywords:

BISINDO
Gesture detection
Pose estimation
Sign language recognition
YOLOv11-Pose

ABSTRACT

Sign language serves as the primary means of communication for deaf individuals, yet existing recognition systems for *Bahasa Isyarat Indonesia* (BISINDO) have not fully exploited keypoint-based pose estimation, particularly for dynamic alphabetic gestures. This study developed a BISINDO gesture detection system using the YOLOv11m-Pose algorithm integrated with MediaPipe hand landmark extraction, Letterbox image preprocessing, and a multi-phase class representation strategy that divided dynamic letters into two movement-phase classes. The main novelty of this study lies in the integration of YOLOv11-Pose with MediaPipe hand keypoint extraction and a multi-phase representation strategy for dynamic letters, which has not been previously applied to BISINDO alphabet detection. A dataset of 1.450 images across 29 classes was collected, augmented to 8.700 samples, and split into training, validation, and test sets. Evaluation on 870 test images yielded an overall accuracy of 99,43%, a macro precision of 99,44%, macro recall of 99,43%, and a mAP50 of 99,11%. All six dynamic letter classes achieved perfect prediction scores, confirming the effectiveness of the multi-phase representation approach. These results demonstrated that the proposed system was capable of reliable BISINDO alphabet detection and provided a solid foundation for further development toward full support for sign language communication.

Corresponding Author:

Achmad Dany Alfansyah,
Department of Informatics,
Telkom University Surabaya,
Jl. Ketintang 156, Gayungan, Surabaya, 60231, Indonesia.
Email: achdany14@gmail.com
https://doi.org/10.52465/joscex.v7i3.69

1. INTRODUCTION

Sign language is a specialized form of communication used by people with disabilities, particularly those who are deaf [1]. In everyday communication, sign language plays a vital role in non-verbal communication, especially for individuals with speech impairments such as deafness [2]. Therefore, sign language is the primary means of communication for deaf individuals for conveying ideas and concepts, as well as for receiving information from others [3]. In Indonesia, there are two widely used sign language systems, Sistem Isyarat Bahasa Indonesia (SIBI), developed by the government, and Bahasa Isyarat Indonesia (BISINDO), which emerged and developed naturally within the deaf community [4].

BISINDO has been in use since at least the 1950s and was officially named in 2006 by Gerakan untuk Kesejahteraan Tunarungu Indonesia (GERKATIN), the Indonesian Association for the Welfare of the Deaf [5]. Over time, BISINDO has served as the primary means of communication for the deaf community throughout Indonesia [6]. Furthermore, BISINDO exhibits variations across different regions of Indonesia as it has evolved naturally in accordance with the customs and culture of local deaf communities [7]. This diversity in the forms and movements of BISINDO necessitates a computer vision-based approach capable of accurately detecting and recognizing hand gestures [8].

Several previous studies have sought to develop BISINDO recognition systems using various approaches. Wiliam et al. [6] utilised a combination of SVM and MediaPipe to recognise the BISINDO alphabet, achieving an accuracy of 98,82%, although some classes, such as the letter 'Y', still had lower scores. Kautsar et al. [9] compared LSTM and 1D CNN+Transformer models for BISINDO word recognition at the word level, where the LSTM model had weaknesses in classifying classes with similar keypoints. Setiawan et al. [10] applied YOLOv8 for BISINDO alphabet detection, including letters with dynamic movements such as G, J, and R, with 99% accuracy however, the model remains susceptible to light intensity, hand position, and movement speed during sign language.

You Only Look Once (YOLO) is an object detection method based on Convolutional Neural Networks (CNNs) that is renowned for its speed and accuracy in processing visual data [11]. YOLOv11-Pose is a single-stage network based on YOLOv11, with an end-to-end architecture that simultaneously integrates object detection and keypoint estimation via backbone, neck, and head components, thereby enhancing focus on key features during pose estimation [12]. This keypoint-based approach has the potential to address the limitations of sign language recognition systems by more precisely mapping hand movements through the analysis of position and keypoints [13]. Previous studies using YOLOv11-Pose, Wu et al. [14] on a monocular ship distance measurement system, reported a Pose mAP50 of 89,8%, demonstrating the model's capability for detailed keypoint extraction.

Previous research on BISINDO detection has been limited to traditional classification models or earlier versions of YOLO that did not integrate pose estimation consequently, the resulting representations of hand movements have not fully utilized keypoint information to its full potential [8]. No research has specifically applied YOLOv11-Pose to BISINDO alphabet detection via keypoint extraction [15]. The keypoint-based approach used in this study offers significant theoretical advantages over conventional bounding box detection methods [16]. While bounding box approaches only provide coarse localization of the hand, keypoints represent the geometric structure of the hand skeleton, enabling more precise mapping of finger configurations, joint positions, and movement dynamics [12]. This capability is particularly well-suited for BISINDO recognition, where the primary distinctions between letter gestures lie in subtle differences in finger angles and joint relationships rather than the overall position or size of the hand [17].

Given these advantages and the limitations of previous studies that relied on standard object detection without pose estimation, this study develops a BISINDO alphabetic gesture detection system using the YOLOv11m-Pose algorithm integrated with MediaPipe hand landmark extraction. The performance of the proposed system is evaluated using mAP50, accuracy, precision, and recall.

2. METHOD

The research pipeline in this study comprises four main stages: dataset collection, preprocessing, model training, and evaluation as illustrated in Figure 1. Each stage is described in detail below.

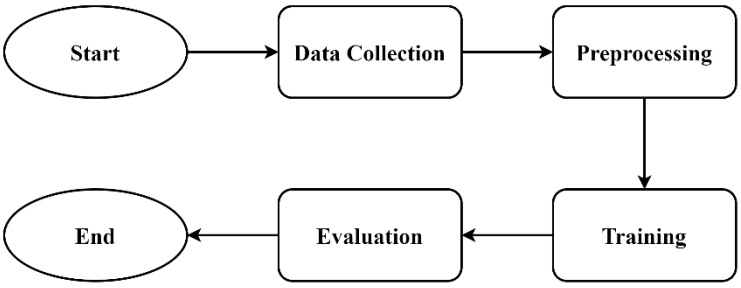

Figure 1. System process flow diagram

Data Collection

The dataset used consists of 1.450 images divided into 29 classes, with each class containing 50 data points, as shown in Figure 2, which displays examples of primary data collected independently.

Figure 2. Sample of self-collected BISINDO images dataset

Figure 3 displays examples of secondary data from various sources, including GitHub [18], Kaggle [19], and Roboflow [20]–[22], with diverse characteristics.

Figure 3. Sample BISINDO alphabet image datasets sourced from github, kaggle and roboflow

These data consist of 20 primary data points collected independently and 30 secondary data points obtained from various sources. Specifically for the dynamic letters G, J, and R, all data are primary, with no secondary data used to maintain consistency in the representation of movement. This study divides dynamic letters into two classes for each letter, namely G1, G2, J1, J2, R1, and R2. For the 23 static letter classes, each class contains 20 primary images and 30 secondary images sourced from GitHub, Kaggle, and Roboflow, while the 6 dynamic classes each contain 50 primary images, as presented in Table 1.

Table 1. Primary and secondary data of the bisindo alphabet

Primary Data	Total	Secondary Data	Total
A	20	A	30
B	20	B	30
C	20	C	30
D	20	D	30
E	20	E	30
F	20	F	30
G1	50	G1	0
G2	50	G2	0
H	20	H	30
I	20	I	30
J1	50	J1	0
J2	50	J2	0
K	20	K	30
L	20	L	30
M	20	M	30
N	20	N	30
O	20	O	30
P	20	P	30
Q	20	Q	30
R1	50	R1	0
R2	50	R2	0
S	20	S	30
T	20	T	30
U	20	U	30
V	20	V	30
W	20	W	30
X	20	X	30
Y	20	Y	30
Z	20	Z	30

This study collects data on dynamic letters by representing two main movement stages, the initial and final positions. Each dynamic class captures the differences across these movement phases, enabling the model to learn hand pose changes more clearly. The secondary data sourced from GitHub, Kaggle, and Roboflow inherently provides diverse visual conditions, including varied lighting, background clutter, skin tones, and

camera distances. The augmentation pipeline further enhances this diversity through brightness adjustment, Gaussian noise, and Gaussian blur, improving the model's robustness to real-world conditions.

Preprocessing

Preprocessing is the initial stage that prepares image data to meet the requirements of the model used in the training process [23]. In this study, preprocessing comprises three main stages, keypoint extraction using the MediaPipe Hand Landmark, image resizing using the Letterbox technique, and data augmentation to increase dataset diversity. MediaPipe is a cross-platform framework developed by Google for building machine learning pipelines. MediaPipe Hands can detect 21 landmarks per hand, representing keypoints at specific locations on the fingers, palm, and wrist [24], as shown in Figure 4.

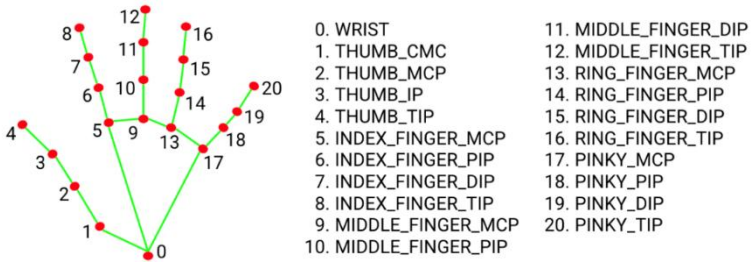

Figure 4. Hand landmark by mediapipe [25]

In the implementation of the YOLO-Pose-based system, the 21 landmarks detected by MediaPipe Hands are converted to 16 keypoints to match the YOLO-Pose format used during training. The five omitted landmarks are the intermediate PIP joints of the index, middle, ring, and little fingers and the intermediate joint of the thumb. These joints are excluded because they can be approximated from the surrounding tip and MCP landmarks and contribute less distinctive information for gesture differentiation, as seen in Figure 5. The resulting 16 keypoints still effectively represent the shape and movement of the hand for BISINDO alphabet detection.

Figure 4. Example of 16 keypoints conversion results

Letterboxing is an image pre-processing technique in the YOLO model that resizes the input image without altering its aspect ratio [26]. Data augmentation is a key technique in deep learning for increasing the size and diversity of training datasets without collecting new data, thereby reducing overfitting and improving the model's generalisation ability [27], [28]. In this study, each image produces 1 original and 5 augmentation variants selected randomly, increasing the dataset from 1.450 to 8.700 images. The augmentation parameters applied are horizontal flip (for two-handed letters only), rotation at +45° and -45°, Gaussian noise (std=12), Gaussian blur (kernel size=5), brightness increase (factor=1,35), and brightness decrease (factor=0,65) [29]. For flip and rotation augmentations, keypoint coordinates are transformed consistently to match the augmented image [30].

Training

The training stage used the YOLOv11m-Pose model with the dataset divided into 70% training, 20% validation, and 10% test [31], the details can be seen in Table 2 below.

Table 2. Data distribution after preprocessing and augmentation

Subset	Number of Data	Percentage
Training	6.090	70%
Validation	1.740	20%
Testing	870	10%
Total	8.700	100%

Training ran for 100 epochs with an image size of 640×640 pixels, a batch size of 16, the AdamW optimizer, an initial learning rate of 0,0003, and a pose loss weight of 20,0 to emphasise keypoint accuracy. The close mosaic strategy was applied during the final 20 epochs, disabling mosaic augmentation so the model could learn on cleaner data distributions in the final phase.

Evaluation

After training, evaluation was performed on the 870 test images that were withheld from both training and validation. Performance was measured using mAP50 to assess detection and keypoint accuracy at an IoU threshold of 0.5 [32], along with a confusion matrix covering precision, recall, and accuracy to evaluate the alignment between predicted and actual class labels [33], [34].

3. RESULTS AND DISCUSSIONS

The model used in this study is YOLO11m-pose, selected based on a performance comparison of all available YOLOv11-Pose variants. Table 3 below presents the comparison of each variant's performance according to the official YOLOv11 benchmark [35].

Table 3. Comparison of YOLOv11-Pose variant performance [35]

Model	Size	mAP50	Speed CPU (ms)	Speed T4 (ms)	Params (M)	FLOPs (B)
YOLO11n-pose	640	83.3	40.3 ± 0.5	1.8 ± 0.0	2.9	7.5
YOLO11s-pose	640	86.6	85.3 ± 0.9	2.7 ± 0.0	10.4	23.9
YOLO11m-pose	**640**	**89.6**	**218.0 ± 1.5**	**5.0 ± 0.1**	**21.5**	**73.1**
YOLO11l-pose	640	90.5	275.4 ± 2.4	6.5 ± 0.1	25.9	91.3
YOLO11x-pose	640	91.6	565.4 ± 3.0	12.2 ± 0.2	57.6	201.7

Based on the comparison in Table 3, the YOLO11m-pose variant was selected as the model used in this study because it offers the best balance between accuracy and computational efficiency. The model has a sufficient number of parameters to extract keypoint features optimally without excessive computational complexity. Compared to smaller variants, YOLO11m-pose achieves a higher mAP50 score of 89.6%, while its CPU inference speed of 218.0 ms and T4 speed of 5.0 ms. Compared to the larger variants (l and x), which reach CPU speeds of 275.4 ms and 565.4 ms respectively, the medium variant is more efficient while maintaining competitive accuracy. Therefore, YOLO11m-pose is considered the most appropriate choice for the BISINDO alphabet detection system based on YOLOv11-Pose [14].

The study conducted testing of the BISINDO gesture detection system based on YOLOv11m-Pose across three consecutive evaluation stages training evaluation, validation data evaluation, and final evaluation on test data. These three stages ensure that the model not only performs well on known data but also generalises reliably to new data, previously unseen data. Table 4 summarises the development of pose metrics at key epochs during training.

Table 3. Evolution of pose metrics across key epochs

Epoch	Pose Loss (Train)	Pose Loss (Val)	mAP50(P)	Precision(P)	Recall(P)
1	2,2826	10,1628	0,0363	0,1839	0,1054
5	1,2941	5,3390	0,7209	0,7347	0,6824
10	1,1459	4,2733	0,8327	0,8288	0,8199
80	0,5455	0,9987	0,9931	0,9916	0,9916
90	0,1539	0,8453	0,9931	0,9937	0,9879
98	**0,1446**	**0,7787**	**0,9939**	**0,9938**	**0,9897**
100	0,1458	0,7723	0,9939	0,9937	0,9900

As shown in Table 4, the training process proceeded with very rapid and stable convergence. Within the first 5 epochs, mAP50(P) had already reached 72,09%, and by the 20th epoch, it had exceeded 94,58%. The training Pose Loss decreased from 2,283 in the first epoch to 0,146 in the 100th epoch, representing a reduction of 93,6%. The results show an interesting phenomenon during the transition from epoch 80 to 81, with a drastic drop in the training Pose Loss from 0,545 to 0,167. This was not an anomaly, but rather the effect of activating the close mosaic strategy, which deactivated mosaic augmentation during the final 20 epochs, allowing the model to train on a cleaner data distribution.

This training strategy is a standard approach in the YOLO training pipeline, designed to stabilise the model at the end of training by removing the complex scene compositions introduced by mosaic augmentation [36]. Wang et al. [36] similarly applied this approach in YOLOv9, turning off mosaic augmentation for the final 15 epochs to improve convergence stability. This strategy drives the loss to its lowest level without compromising generalisation, as can also be observed visually in the training examples before and after mosaic deactivation. The absence of any significant spikes across the entire loss curve over the 100 epochs confirms the overall stability of the training. The training process automatically saves the best model at epoch 98 based on the highest validation mAP50(P) value. Table 5 presents the full evaluation results on the validation data using this best model.

Table 4. Evaluation results on validation data

Metric	Value
mAP50(P)	99,39%
Precision(P)	99,38%
Recall(P)	98,97%
Pose Loss (Val)	0,779

Table 5 shows that on 1.741 validation images that the model never saw during training, the best model achieved an mAP50(P) of 99,39%. Precision (P) of 99,38% and Recall (P) of 98,97% were both very high, indicating that the model produced almost no false positives or false negatives. Most importantly, the validation Pose Loss of 0,779 continued to decrease consistently in line with the training Pose Loss over 100 epochs, confirming the absence of overfitting. This consistent performance on the validation data ensures that the model is ready to be tested on entirely new data. Augmentation was applied to the full dataset prior to splitting, so augmented variants of the same original image may appear across subsets. However, since each technique introduces visually distinct transformations rather than near-identical copies, the risk of inflating test performance is considered minimal.

Table 5. Evaluation results on testing data

Metric	Value
mAP50(P)	99,11%
Precision Macro	99,44%
Recall Macro	99,43%
Accuracy Overall	99,43%
Total Correct	865/870
Total Incorrect	5/870

Table 6 presents the final evaluation results on the test data. Evaluation on 870 test images, which were entirely separate from the training and validation sets, yielded an overall accuracy of 99,43%, with only 5 incorrect predictions out of a total of 870 samples. mAP50(P) reached 99,11%, whilst macro precision and macro recall stood at 99,44% and 99,43% respectively. This study surpasses previous studies on BISINDO gesture detection, as shown in Table 7. It is worth noting that this study classifies 29 classes, which is three more than the 26 classes used in the YOLOv8 and SVM-MediaPipe studies, because the three dynamic letters (G, J, R) are each divided into two movement-phase classes. This larger and more complex label space means the classification task is inherently more difficult, yet the proposed system still achieves higher accuracy and precision, demonstrating the effectiveness of the keypoint-based multi-phase approach.

Table 6. Comparison with previous studies on BISINDO gesture detection

Study	Method	Classes	Accuracy	Precision	Recall
Indonesian Sign Language (BISINDO) Alphabet Detection Using the You Only Look Once (YOLO) Algorithm Version 8 [10]	YOLOv8	26	99.00%	99.40%	98.40%
Recognitions of Bahasa Isyarat Indonesia (BISINDO) Alphabets Using SVM and Mediapipe [6]	SVM-MediaPipe	26	98.82%	97.34%	99.00%
This Study	YOLOv11m-Pose	29	99.43%	99.44%	99.43%

These overall figures require further analysis by class performance distribution. Figure 7 displays the training curve documenting the model's convergence, followed by Table 8, which details metrics per class on the test data.

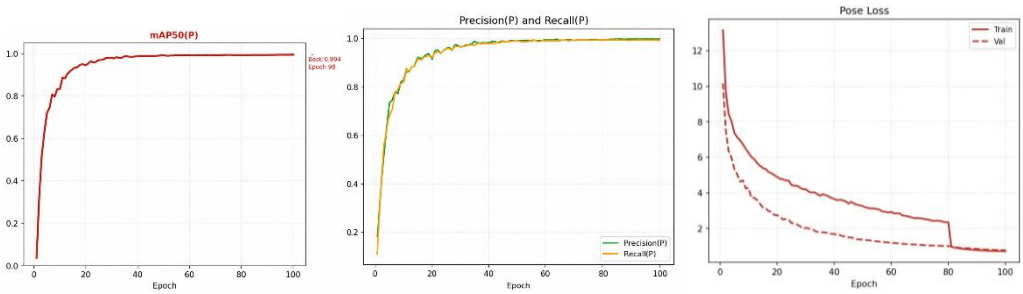

Figure 5. Training performance overview

The mAP50(P) graph shows a sharp rise during the first 20 epochs, followed by gradual, stable convergence, peaking at the 98th epoch. The Precision(P) and Recall(P) graphs show both metrics moving in very close tandem from the 50th epoch onward, indicating that the model has successfully balanced detection accuracy and completeness. The Pose Loss graph displays a parallel decrease in training and validation loss without divergence over 100 epochs, providing concrete evidence of the absence of overfitting. Given this positive convergence, it is now necessary to examine how this performance is distributed across each class individually.

Table 7. Class-wise evaluation metrics on testing data

Class	Type	Precision	Recall	Accuracy	Support
A	2 hands	1,0000	1,0000	1,0000	30
B	2 hands	1,0000	1,0000	1,0000	30
C	1 hand	1,0000	0,9333	0,9977	30
D	2 hands	1,0000	1,0000	1,0000	30
E	1 hand	1,0000	1,0000	1,0000	30
F	2 hands	1,0000	1,0000	1,0000	30
G1	moving	1,0000	1,0000	1,0000	30
G2	moving	1,0000	1,0000	1,0000	30
H	2 hands	0,9677	1,0000	0,9989	30
I	1 hand	1,0000	1,0000	1,0000	30
J1	moving	1,0000	1,0000	1,0000	30
J2	moving	1,0000	1,0000	1,0000	30
K	2 hands	1,0000	0,9667	0,9989	30
L	1 hand	1,0000	1,0000	1,0000	30
M	2 hands	1,0000	1,0000	1,0000	30
N	2 hands	1,0000	1,0000	1,0000	30
O	1 hand	1,0000	1,0000	1,0000	30
P	2 hands	1,0000	0,9667	0,9989	30
Q	2 hands	1,0000	1,0000	1,0000	30
R1	moving	1,0000	1,0000	1,0000	30
R2	moving	1,0000	1,0000	1,0000	30
S	2 hands	0,9677	1,0000	0,9989	30
T	2 hands	0,9355	0,9667	0,9966	30
U	1 hand	0,9677	1,0000	0,9989	30
V	1 hand	1,0000	1,0000	1,0000	30
W	2 hands	1,0000	1,0000	1,0000	30
X	2 hands	1,0000	1,0000	1,0000	30
Y	2 hands	1,0000	1,0000	1,0000	30
Z	1 hand	1,0000	1,0000	1,0000	30
Macro Avg		0,9944	0,9943	0,9986	870

Table 8 shows a very even distribution of performance across all 29 classes. A total of 22 classes achieved perfect scores across all metrics, including all 6 classes of moving letters, demonstrating that the multi-phase representation approach is highly effective at accurately distinguishing each movement phase. Classes that experienced a decline in performance include C, H, K, P, S, T, and U. Of all these classes, class T experienced the most significant decline because it became the target of incorrect predictions from several other classes.This error distribution pattern can be visualised in the confusion matrices for each letter group in Figures 8.

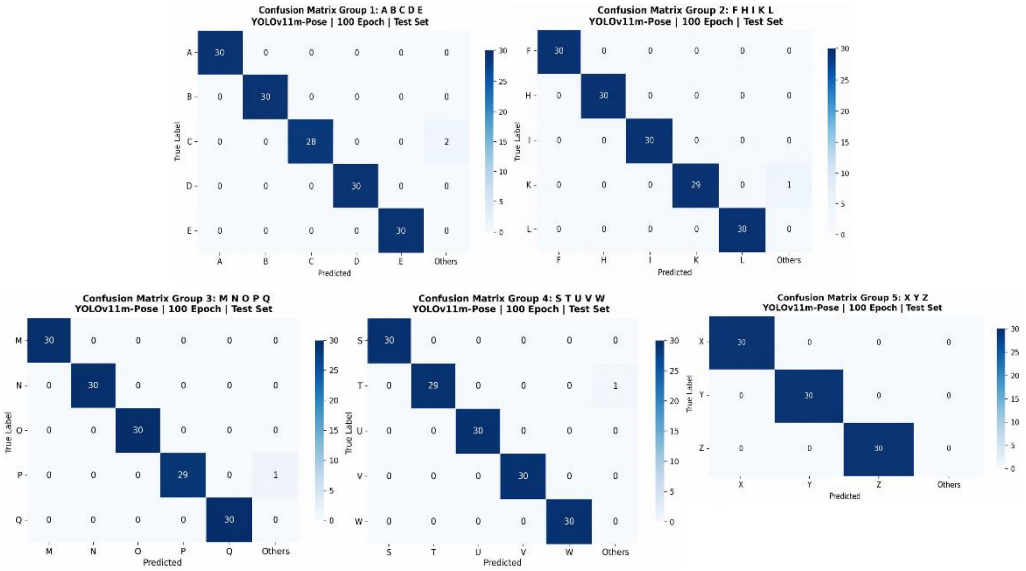

Figure 6. Confusion matrix for group 1-5

Figure 8 shows the confusion matrix for Group 1-5. Classes A, B, D and E all achieved perfect predictions. Class C had 2 errors in the others column, the model incorrectly predicted 1 sample as S and 1 as U. The similarity in the position of the thumb and the curved index finger in some samples of the letter caused both errors C, which, from certain angles, resembled the hand position components of the letters S and U. Classes F, H, I, and L achieved perfect results. Class K had 1 error in the other class (class T), as the upright position of the index finger in the letter K can resemble one of the keypoint configurations of the letter T under certain image conditions.

Classes M, N, O and Q all achieved perfect results. Class P made one error, misclassifying others as class T. Classes S, U, V and W achieved perfect results. Class T made 1 error, misclassifying a sample from the others class into class H. An interesting pattern is evident here, class T was the victim of misclassification by classes K and P, whilst also making an error regarding class H, indicating that the keypoint patterns of the letter T share representational similarities with certain other letters under specific conditions. Classes X, Y, and Z all classes achieved perfect prediction

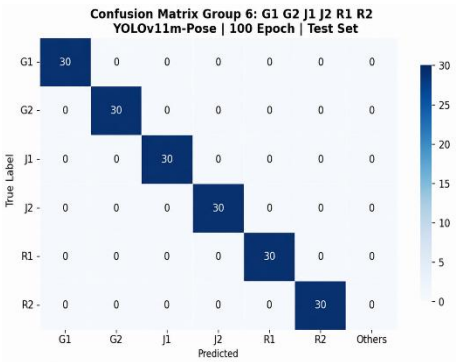

Figure 7. Confusion matrix for group 6

Figure 9 shows Group 6, which includes all moving letters, and all six classes also achieved perfect prediction without a single error. This achievement is particularly significant given that moving letters are inherently much more difficult to detect, as they require an understanding of the dynamics of movement between frames, rather than merely static positions. Overall, out of 870 predictions, there were only 5 errors. Among the static classes, the letters that experienced a decline in performance are C, H, K, P, S, T, and U. Class T experienced the most significant decline as it became the target of incorrect predictions from several other classes, because the keypoint configuration of letter T, formed by two hands with an upright index finger, shares visual similarity with letters K and P under certain shooting angles and lighting conditions. These misclassification patterns are consistent with the visual ambiguity analysis in the full study. To gain a deeper understanding of why each of these errors occurred, Figure 10 displays a direct visualisation of the five incorrectly predicted images.

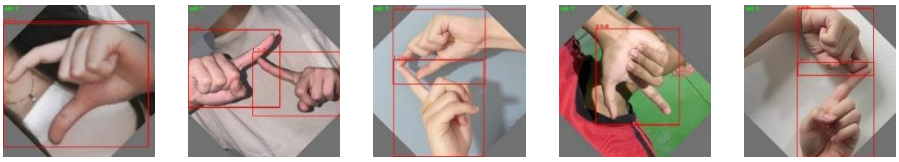

Figure 8. Visualisation of 5 incorrect predictions out of 870 total testing samples

Figure 10 shows the five incorrectly predicted images, along with the model's detection bounding boxes and confidence scores. The first error is C predicting U (confidence 0,37). The image of the letter C shows the thumb and index finger in a curved position, resembling the hand position component of the letter U from the camera's perspective. The very low confidence value (0,37) indicates that the model is highly uncertain, suggesting that the representation of the C keypoints in this image lies on the boundary of ambiguity between the two classes.

The second error is T predicting H (confidence 0,40 and 0,47), the model treats the two hands forming the letter T as two separate detections with low confidence, indicating that it interprets the two hands as independent entities rather than a single gesture. The third error is P predicting T (confidence 0,75 and 0,91), this is the only error with relatively high confidence, indicating that the configuration of keypoints in this image does indeed closely resemble the pattern of the letter T. The model is fairly certain but remains incorrect due to the high visual similarity between P and T under those specific conditions.

The fourth error is the model predicting C as S (confidence 0,27), the model predicts the letter C as S with very low confidence because the thumb and index finger curve in the same way as the single-hand S. The fifth error is K predicting T (confidence 0,45 and 0,89), the model detects the letter K as T with two bounding boxes, likely because the upright position of the index finger on the letter K visually resembles the keypoint components of the letter T. Overall, four out of five errors have low confidence, indicating the model is already showing uncertainty in difficult cases. This is a positive characteristic, indicating that the model is not overconfident in its incorrect predictions. All errors are edge cases resulting from variations in lighting, viewing angles, and visual ambiguity, rather than systematic failures. With an error rate of just 0,57% (5 out of 870 samples), this model demonstrates highly competitive performance and is ready to serve as the foundation for a reliable BISINDO detection system.

4. CONCLUSION

This study successfully developed a BISINDO gesture detection system based on the YOLOv11m-Pose algorithm. The system addresses the limitations of previous approaches that did not fully leverage keypoint-based pose estimation for sign language recognition. The results demonstrate that the proposed system achieved an overall accuracy of 99,43% on 870 test images, with a macro precision of 99,44%, macro recall of 99,43%, and mAP50(P) of 99,11%, surpassing previous studies that employed YOLOv8 99% and SVM-MediaPipe 98,82% across fewer and less complex classes. The multi-phase representation strategy for dynamic letters (G, J, and R), each divided into two movement phases, proved highly effective, with all six dynamic classes achieving perfect prediction scores. The parallel decrease of training and validation Pose Loss over 100 epochs confirms training stability with no sign of overfitting. The five misclassifications observed were attributed to edge cases involving visual ambiguity between similar hand configurations under specific lighting and viewing-angle conditions, rather than systematic model failure, as evidenced by the consistently low confidence scores in those incorrect predictions. This study has several limitations. The dataset is limited to 29 alphabetic BISINDO gestures and does not cover word or sentence-level recognition, and evaluation was conducted on image data rather than in real-time deployment conditions. Future research directions include extending the system to word and sentence-level BISINDO recognition, integrating speech synthesis for real-time communication support, and evaluating the system on edge computing devices or mobile platforms to assess its practical applicability for the deaf community.

ACKNOWLEDGEMENTS

The authors are grateful to Mr. Ardian Yusuf Wicaksono, S.Kom., M.Kom., as the first supervisor, for his guidance, direction, and valuable feedback throughout the preparation of this research.

The authors also extend their sincere thanks to Mrs. Pima Hani Safitri, S.Kom., M.Kom., as the second supervisor, for her assistance, insightful suggestions, and encouragement from the initial stage of topic selection through to the completion of this research.

CREDIT AUTHORSHIP CONTRIBUTION STATEMENT

Achmad Dany Alfansyah: Idea, research methodology, research progress. **Ardian Yusuf Wicaksono, S.Kom., M.Kom.**: Supervision, conceptual development. **Pima Hani Safitri, S.Kom**: Supervision, writing - review and editing.

REFERENCES

[1] K. K. Candra and K. Kusrini, "Klasifikasi Gambar Bahasa Isyarat Indonesia (Bisindo) Pada Komunitas Tuli Menggunakan Machine Learning," *JUSITI*, vol. 14, no. 1, pp. 56–63, May 2025.
[2] A. Nugroho, R. Setiawan, A. Harris, and Beny, "Deteksi Bahasa Isyarat Bisindo Menggunakan Metode Machine Learning," *Processor*, vol. 18, no. 2, Nov. 2023.
[3] M. Malika and B. Berlianti, "Penggunaan Bahasa Isyarat SIBI dan BISINDO untuk Siswa Tuli di Sekolah Luar Biasa Negeri Pembina Medan," *Concept*, vol. 4, no. 3, pp. 78–87, Sep. 2025.
[4] I. D. M. B. A. Darmawan, Linawati, G. Sukadarmika, N. M. A. E. D. Wirastuti, and R. Pulungan, "Indonesian sign language system (SIBI) dataset: Sentences enhanced by diverse facial expressions for total communication," *Data Br.*, vol. 60, p. 111642, Jun. 2025.
[5] N. Palfreyman, "Managing Sign Language Data from Fieldwork," in *The Open Handbook of Linguistic Data Management*, A. L. Berez-Kroeker, B. McDonnell, E. Koller, and L. B. Collister, Eds. The MIT Press, 2022, pp. 267–276.
[6] R. Wiliam, C. Lufian, Meiliana, and A. Y. Zakiyyah, "Recognitions of Bahasa Isyarat Indonesia (BISINDO) Alphabets Using SVM and Mediapipe," in *2024 6th International Conference on Cybernetics and Intelligent System (ICORIS)*, 2024, pp. 1–5.
[7] N. R and H. Setiaji, "Developing Digital Teaching Media for Indonesian Sign Language (BISINDO)," *TEKNODIKA*, vol. 21, no. 1, pp. 65–75, Mar. 2023.
[8] T. Tao, Y. Zhao, T. Liu, and J. Zhu, "Sign Language Recognition: A Comprehensive Review of Traditional and Deep Learning Approaches, Datasets, and Challenges," *IEEE Access*, vol. 12, pp. 75034–75060, 2024.
[9] M. D. Kautsar, A. M. Hariono, and R. Akmal, "Attention vs LSTM: Improving Word-level BISINDO Recognition," *arXiv*, Feb. 2025.
[10] F. A. Setiawan, Z. A. Rohmah, and G. F. Laxmi, "Indonesian Sign Language (BISINDO) Alphabet Detection Using the You Only Look Once (YOLO) Algorithm Version 8," in *2024 International Conference on Computer, Control, Informatics and its Applications (IC3INA)*, 2024, pp. 388–393.
[11] S. Daniels, N. Suciati, and C. Fathichah, "Indonesian Sign Language Recognition using YOLO Method," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1077, no. 1, p. 12029, Feb. 2021.
[12] L. Liu, A. Kurban, and Y. Liu, "Improved YOLOv11pose for Posture Estimation of Xinjiang Bactrian Camels," *Int. J. Adv. Comput. Sci. Appl.*, vol. 15, no. 12, 2024.
[13] A. O. Hashi, S. Z. M. Hashim, and A. B. Asamah, "A Systematic Review of Hand Gesture Recognition: An Update From 2018 to 2024," *IEEE Access*, vol. 12, pp. 143599–143626, 2024.
[14] Y. Wu and others, "Monocular Unmanned Boat Ranging System Based on YOLOv11-Pose Critical Point Detection and Camera Geometry," *J. Mar. Sci. Eng.*, vol. 13, no. 6, p. 1172, Jun. 2025.
[15] R. Khanam and M. Hussain, "YOLOv11: An Overview of the Key Architectural Enhancements," *arXiv Prepr.*, 2024.
[16] A. S. M. Miah, M. A. M. Hasan, S. Nishimura, and J. Shin, "Sign Language Recognition Using Graph and General Deep Neural Network Based on Large Scale Dataset," *IEEE Access*, vol. 12, pp. 34553–34569, 2024.
[17] A. Imran, M. S. Hulikal, and H. A. A. Gardi, "Real Time American Sign Language Detection Using Yolo-v9," *arXiv*, 2024.
[18] D. Joan, V. Vincent, K. J. Daniel, S. Achmad, and R. Sutoyo, "BISINDO Hand-Sign Detection Using Transfer Learning," in *2023 IEEE 8th International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*, 2023, pp. 1–7.
[19] S. Airlangga, "BISINDO_FINAL." 2025.
[20] TGA, "BISINDO Computer Vision Model." 2025.
[21] Skripsi, "Bisindo - Huruf Abjad Computer Vision Model." 2025.
[22] D, "BISINDO Detection Computer Vision Model." 2026.
[23] J. D. H. Koresh, "Implementation and Efficient Analysis of Preprocessing Techniques in Deep Learning for Image Classification," *Curr. Med. Imaging Rev.*, vol. 20, p. e290823220482, Aug. 2024.
[24] U. Sarkar, A. Chakraborti, T. Samanta, S. Pal, and A. Das, "Enhancing ASL Recognition with GCNs and Successive Residual Connections," *arXiv*, Aug. 2024.
[25] K. M. Kavana and N. R. Suma, "Recognization of Hand Gestures Using Mediapipe Hands," *IRJMETS*, vol. 4, no. 6, pp. 4149–4156, Jun. 2022.
[26] R. Yu, Y. Zhang, and S. Liu, "CIMB-YOLOv8: A Lightweight Remote Sensing Object Detection Network Based on Contextual Information and Multiple Branches," *Electronics*, vol. 14, no. 13, p. 2657, Jun. 2025.
[27] X. Wang, K. Wang, and S. Lian, "A Survey on Face Data Augmentation," *arXiv*, 2019.
[28] K. et al. Alomar, "Data Augmentation in Classification and Segmentation: A Survey and New Strategies," *J. Imaging*, vol. 9, no. 2, 2023.
[29] J. Wei, Q. Wang, X. Song, and Z. Zhao, "The Status and Challenges of Image Data Augmentation Algorithms," *J. Phys. Conf. Ser.*, vol. 2456, no. 1, p. 12041, Mar. 2023.
[30] X. Wang, L. Kong, Z. Zhang, H. Wang, and X. Lu, "Keypoint regression strategy and angle loss based YOLO for object detection," *Sci. Rep.*, vol. 13, no. 1, p. 20117, Nov. 2023.
[31] S. Bosson-Amedenu, E. Boakye, E. Ayitey, and J. A. Addor, "Comparative performance of machine learning models in predicting arsenic exposure risk in Ghana's riverine ecosystems," *BMC Environ. Sci.*, vol. 2, no. 1, p. 26, Dec. 2025.
[32] J. Terven, D.-M. Cordova-Esparza, and J.-A. Romero-Gonzalez, "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, Nov. 2023.
[33] S. Yang and G. Berdine, "Confusion Matrix," *The Chronicles*, vol. 12, no. 53, pp. 75–79, Oct. 2024.
[34] M. F. Amin, "Confusion Matrix in Binary Classification Problems: A Step-by-Step Tutorial," *J. Eng. Res.*, vol. 6, no. 5, p. 0,

Dec. 2022.
[35] "Pose Estimation," *Ultralytics*. [Online]. Available: https://docs.ultralytics.com/tasks/pose/. [Accessed: 01-Apr-2026].
[36] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information," *arXiv*, 2024.

.