Vol. 7, No. 1, February 2026, Page. 490-501
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.1.5240

Alphabet Gesture Classification of Indonesian Sign Language Using Convolutional Neural Networks

Yanuar Gideon Simalango[1], Anindita Septiarini*[2], Masna Wati[3], Hamdani[4], Rajiansyah[5]

[1,2,3,4,5]Department of Informatics, Mulawarman University, Indonesia

Email: [2]anindita@unmul.ac.id

Received : Aug 5, 2025; Revised : Aug 5, 2025; Accepted : Oct 7, 2025; Published : Feb 15, 2026

Abstract

Indonesian Sign Language (BISINDO) serves as a communication medium for deaf individuals to engage with their environment. Alphabet gestures in BISINDO play a crucial role in the formation of words and sentences. Nonetheless, the automatic recognition of BISINDO alphabet movements remains a difficulty in the advancement of accessible technology. This research intends to categorize BISINDO alphabet gestures via the Convolutional Neural Network (CNN) model. The CNN approach was used due to its proficiency in recognizing visual patterns and images. The dataset comprises BISINDO alphabet gesture photos captured from diverse perspectives and lighting conditions. The data processing procedure encompasses pre-processing phases, including picture normalization, data augmentation, and the segmentation of the dataset into training, validation, and test subsets. The constructed CNN model has multiple convolutional and pooling layers to thoroughly extract visual characteristics. The study's results indicate that the CNN model can classify BISINDO alphabet gestures with a high accuracy of 90% on the test data. This model's deployment is anticipated to aid in the creation of automatic sign language translation programs, hence enhancing communication between the deaf community and the general populace. This study demonstrates the potential of CNN models to support the development of inclusive communication technologies for the hearing impaired in Indonesia, particularly for under-researched sign languages like BISINDO.

Keywords : *Augmentation, BISINDO, Preprocessing, Recognition, Translation*

1. INTRODUCTION

Sign language is the primary mode of communication for individuals who are deaf or speech-impaired, allowing them to interact effectively with their surroundings. In Indonesia, Indonesian Sign Language (BISINDO) functions as a key visual-spatial communication system. However, limited public awareness and understanding of BISINDO among hearing individuals still present barriers to inclusive interaction [1]. As the importance of inclusive communication becomes more widely recognized, there is a growing demand for assistive technologies that can bridge communication gaps between sign language users and the public. One of the main challenges in developing such technology lies in the accurate recognition and classification of hand gestures. Deep learning methods, particularly Convolutional Neural Networks (CNN), have shown strong potential in enhancing the performance of sign language recognition systems [2]–[4]. CNN is a type of artificial neural network that excels in visual pattern recognition and has been successfully applied in fields such as facial recognition, medical imaging, and object classification [5]–[9]. In sign language recognition, CNNs can be used to detect and classify hand movements or positions that represent letters or words. For BISINDO, this involves identifying specific hand shapes used to represent alphabetic signs. Research has demonstrated that CNNs are capable of extracting spatial features from hand images with high accuracy, even under varying conditions such as changes in lighting, camera angle, and hand shape [1]. The use of CNN in sign language recognition has been extensively explored for American Sign Language (ASL) and other systems, achieving impressive classification accuracy [10]–[13]. However, focused research on BISINDO remains limited. BISINDO has its own unique set of hand configurations, some of which involve more complex or dynamic movements compared to ASL, making recognition more challenging for standard models [14]. To develop a CNN model suited for BISINDO, it is essential to design a dataset that reflects its unique characteristics. Data collection, annotation, and augmentation are crucial steps to ensure accurate gesture recognition. For instance, [12] report that CNN models can classify static ASL gestures with over 95% accuracy when trained on large, well-augmented datasets. These methods adapted to BISINDO by curating image data specifically annotated for the Indonesian alphabetic signs. Similarly, [15] demonstrated that CNN-based classification of ASL achieved up to 96% accuracy using static image datasets, showcasing the effectiveness of CNN in handling consistent hand gestures. With proper dataset development and model training, similar outcomes may be achievable for BISINDO.

Recent research on CNNs focuses on enhancing processing efficiency and accuracy while minimizing training data requirements. This study employs the ResNet-50 architecture, commonly used in transfer learning, featuring layers such as flatten, dense (128 ReLU neurons), and softmax output (25 neurons) for classification. Using a dataset of 5,000 images (70:30 split for training and testing), the model achieved a remarkable accuracy of 99.15% with minimal loss of 0.25 at a learning rate of 0.01 and 50 epochs [16]. The proposed model demonstrates high precision in translating Indonesian Sign Language (BISINDO), offering significant potential to enhance communication accessibility for individuals with hearing disabilities [17]. Compared with previous works on Indian and Indonesian sign languages, this approach proves CNN's capability—particularly ResNet-50—in developing practical and inclusive tools for automated sign language recognition, contributing to bridging communication gaps and improving the quality of life for the deaf community through advanced AI technology [18].

This study proposes the development of an Alphabet Gesture Classification system for Indonesian Sign Language (BISINDO) using CNN. The proposed model aims to accurately recognize and classify static hand gestures representing the BISINDO alphabet, thereby supporting more inclusive communication between the deaf and hearing communities in Indonesia

2. METHOD

Figure 1 shows there are several stages of this research flow consisting of data collection, pre-processing, training and validation, CNN modelling using MobileNetV2, VGG16, and Efficient Net architectures, followed by evaluation and classification. At the first stage of this flow, data collection is carried out by creating a dataset consisting of images of BISINDO alphabet hand gestures. Data is gathered through a recording process using a camera with adequate lighting and a neutral background. In the second stage, the researchers will split the image data into two parts: training data and validation data. At the third stage, the researchers will use Python programming to resize the images using resizing, normalization, and augmentation techniques. The training data will then be used to build the model using CNN architectures, namely MobileNetV2, VGG16, and Efficient Net. In the fourth stage, the researchers will evaluate the modelling results using test data with classification and a confusion matrix.

2.1. Data Collection

The image data used in this study consists of two datasets: a public dataset and a private dataset. The public dataset was obtained from Kaggle (https://www.kaggle.com/datasets/agungmrf/indonesian-sign-language-bisindo) and contains 11,500 BISINDO images with a resolution of 640×640 pixels, distributed across 26 alphabet classes. Meanwhile, the private dataset was collected independently using an iPhone 8 Plus main camera (12 MP), resulting in 5,200 images representing the same 26 BISINDO alphabet classes. The private dataset includes various hand gestures and visual variations captured from non-expert participants, ensuring diversity and complementing the public dataset. The detailed information about both datasets is presented in Table 1.

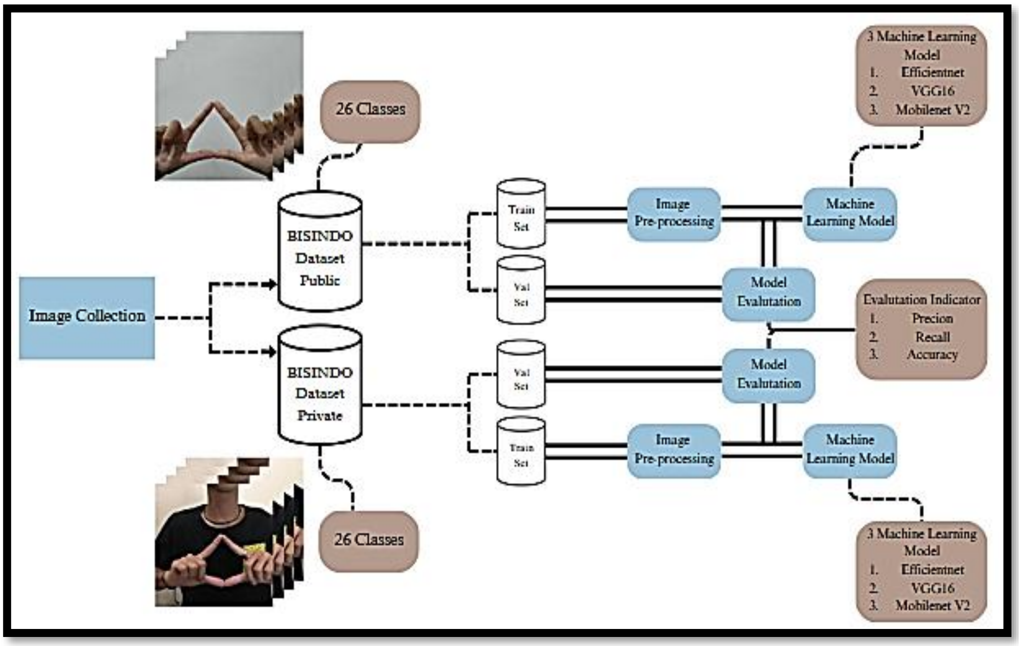


Figure 1. Architecture for BISINDO Sign Language Classification Using CNN

Table 1. Number of Dataset

Training Data						Validation Data					
Class	Private	Public	Class	Private	Public	Class	Private	Public	Class	Private	Public
A	160	350	N	160	350	A	40	88	N	40	88
B	160	351	O	160	357	B	40	88	O	40	90
C	160	343	P	160	357	C	40	86	P	40	90
D	160	348	Q	160	348	D	40	87	Q	40	87
E	160	352	R	160	350	E	40	89	R	40	88
F	160	357	S	160	343	F	40	90	S	40	86
G	160	360	T	160	360	G	40	90	T	40	90
H	160	348	U	160	355	H	40	87	U	40	89
I	160	360	V	160	357	I	40	90	V	40	90
J	160	360	W	160	360	J	40	90	W	40	90
K	160	348	X	160	355	K	40	87	X	40	89
L	160	357	Y	160	328	L	40	90	Y	40	83
M	160	355	Z	160	360	M	40	89	Z	40	90

2.2. Data Design

The dataset used for model development must be divided into a training set and a validation set. The training set is used to train the model to classify the data, while the validation set is used to test the trained model on previously unseen data. In this study, the training and validation data were split in a ratio of 80% to 20%.

2.3. Pre-processing Data

The input data is prepared and processed prior to its integration into the CNN modelling procedure. The primary objective of preprocessing is to enhance model performance and augment forecast accuracy. This stage has three primary processes:

2.3.1. Resize

The preliminary phase of preprocessing involves resizing all hand gesture photos to achieve uniform proportions. This procedure is crucial since the input for CNN must maintain a stable size for both processing efficiency and structural consistency of the network. In this research, each image was downsized to pixels, a typical dimension suitable with numerous prevalent CNN designs, including VGGNet and ResNet. Resizing also decreases computer complexity while preserving essential information from the image. [19], [20] assert that the regular scaling of the picture dataset enhances the stability of the training process and expedites model convergence. They asserted that non-uniform image dimensions can induce propagation errors in the convolution layer, particularly when utilized on GPUs. Consequently, resizing pertains not merely to aesthetics or data structure, but is intrinsically linked to the technological integrity of the CNN model. Resizing also prevents distortion of image representation, which might result in misclassification.

2.3.2. Normalization

Subsequent to resizing the image, the following procedure is to normalize the pixel values. Normalization is achieved by transforming the pixel value range from 0–255 to 0–1 through division of each pixel by 255. This normalization aims to achieve a more consistent distribution of input data, facilitating faster and more stable model learning. CNN functions more efficiently when the input data is scaled to a limited and uniform range, as excessively high values might lead to vanishing or ballooning gradients during backpropagation. Research conducted by [21] demonstrated that basic normalizing techniques can enhance the accuracy of CNN models by 5–7% relative to the utilization of unprocessed images. They highlighted that neural networks exhibit heightened sensitivity to extreme pixel values, particularly during iterative training. Normalization results in a more balanced data distribution, hence enhancing the efficiency of the optimization process. Moreover, standardized pixel values enhance initial weighting and yield more uniform training parameters.

2.3.3. Augmentation

The final critical preprocessing phase in this study is data augmentation. Augmentation is performed to enhance visual diversity without increasing the quantity of original photos, by altering the image by rotation 15–30 degrees, horizontal flipping, zooming, and adjustments to brightness and contrast. [22] to previously unseen data. [23] assert that data augmentation can substantially enhance the model's ability to identify varied visual patterns and mitigate the likelihood of overfitting, particularly when the dataset is constrained in size. In their studies, the CNN model with supplemented data achieved a classification accuracy improvement of up to 12% relative to the model lacking augmentation. This indicates that augmentation functions not merely as a supplementary strategy, but as a crucial component in the development of a resilient system. Augmentation techniques also represent changes in actual settings, such disparities in camera angles, room illumination, or anatomical differences in the user's hand [24]. In the context of BISINDO, this is crucial as each human possesses a unique hand form and size and may gesture in diverse contexts.

2.4. Model Architecture

Classification model will be built using pre-processed images. The pre-trained model loaded into the Python environment using deep learning libraries such as TensorFlow or Keras. The pre-trained models to be used are MobileNetV2, VGG16, and Efficient Net.

a. EfficientNet

Figure 2 illustrates the EfficientNet architecture. EfficientNet is a CNN architecture that employs a compound scaling method to uniformly adjust depth, width, and resolution. This approach enables EfficientNet to achieve high accuracy while maintaining optimal computational efficiency.

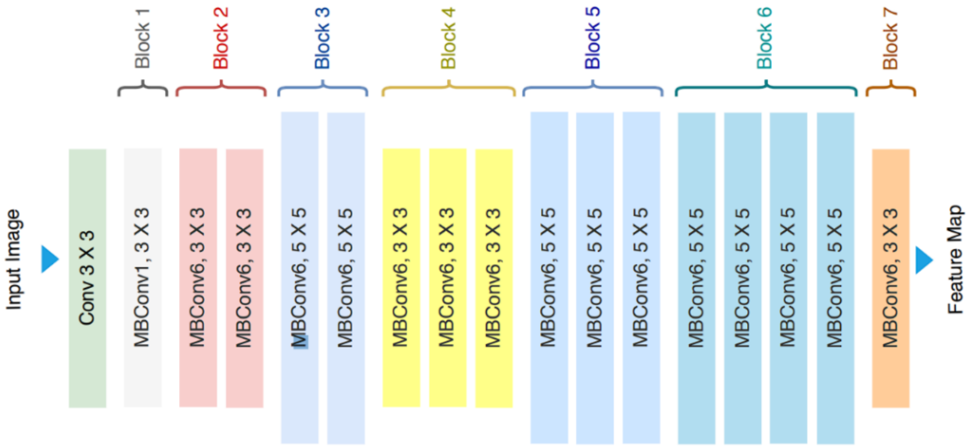


Figure 2. Architecture of EfficientNet-B0 with MBConv as Basic building blocks.

b. MobileNetV2

Figure 3 illustrates the MobileNetV2 architecture, a CNN designed to achieve high performance on mobile and embedded devices. MobileNetV2 is built upon an inverted residual structure, where residual connections link the bottleneck layers. The intermediate expansion layers employ lightweight depthwise convolutions to extract and filter features, introducing non-linearity efficiently. Overall, the MobileNetV2 architecture begins with an initial fully convolutional layer containing 32 filters, followed by 19 residual bottleneck layers that enhance feature representation while maintaining computational efficiency.

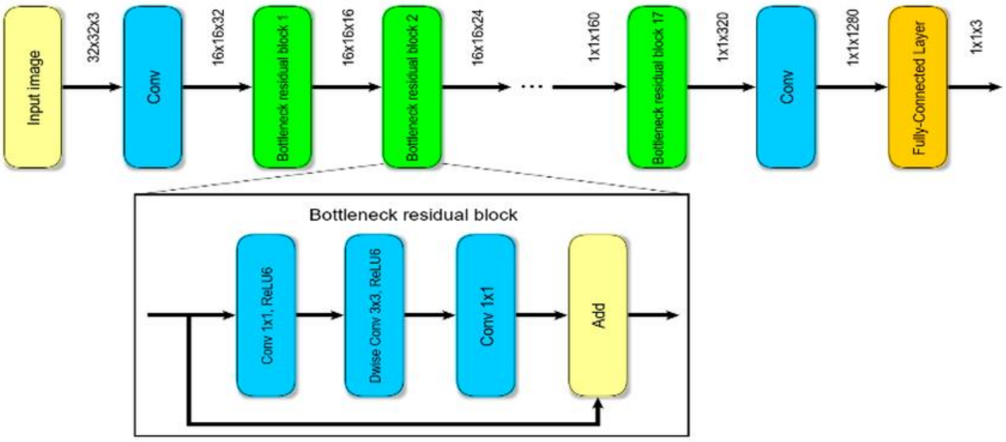


Figure 3. Architecture of MobileNetV2

c. VGG16

Figure 4 illustrates the VGG16 architecture. VGG16 is a Convolutional Neural Network (CNN) model consisting of 16 layers, including 13 convolutional layers and 3 fully connected layers. This architecture is widely recognized for its simplicity and strong performance in computer vision tasks such as image classification and object recognition. By combining convolutional and max-pooling layers, VGG16 is able to progressively learn hierarchical visual features, making it a popular and reliable choice in various deep learning applications.

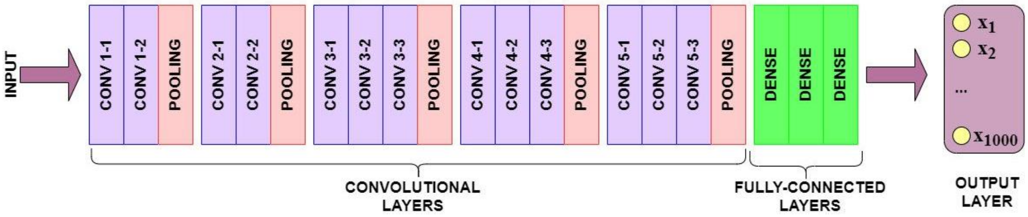


Figure 4. Architecture of VGG16

2.5. Evaluation

In this stage, a classification score report is generated using the predictions and ground truth labels. The evaluation metrics used in this study include Accuracy (Equation 1), Precision (Equation 2), and Recall (Equation 3) [25]. These metrics help assess the model's performance in recognizing BISINDO alphabet gestures from both private and public datasets.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

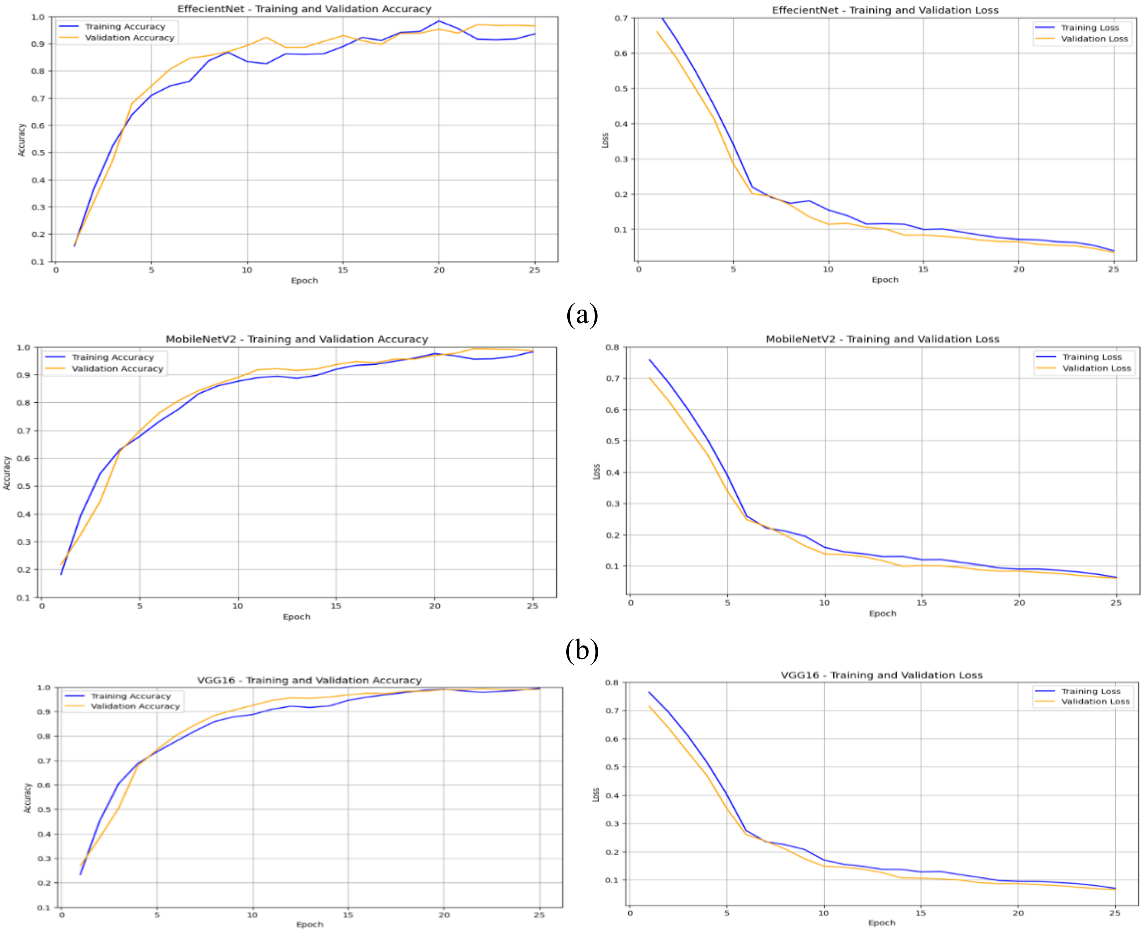
$$precision = \frac{TP}{TP + FP} \tag{2}$$

$$recall = \frac{TP}{TP + FN} \tag{3}$$

3. RESULT

Figure 5 illustrates the training performance of the EfficientNet, MobileNetV2, and VGG16 models using the private BISINDO dataset. The graphs display both accuracy and loss values across 25 epochs, showing the models' learning progress and convergence behavior. EfficientNet demonstrates a faster convergence in the early epochs, while VGG16 achieves the most stable accuracy with minimal fluctuation in loss values. In contrast, MobileNetV2 exhibits moderate stability and a slightly slower convergence rate, indicating that its lightweight architecture may require more epochs to optimize learning on the private dataset. Overall, the comparison highlights VGG16's strong consistency and EfficientNet's efficiency in achieving high accuracy, as shown in Figure 5.

The performance of the three models trained on the public BISINDO dataset is presented in Figure 6, which depicts the trend of accuracy and loss over 25 epochs. The figure shows that all models experience steady accuracy improvement, with EfficientNet and VGG16 consistently outperforming MobileNetV2. The EfficientNet model achieves high accuracy with minimal overfitting, while VGG16 maintains stable loss reduction throughout the training process. This consistency suggests that both architectures generalize well on larger and more diverse datasets, while MobileNetV2 remains slightly behind in performance due to its lightweight structure optimized for computational efficiency. The progression curves of model performance can be clearly observed in Figure 6.

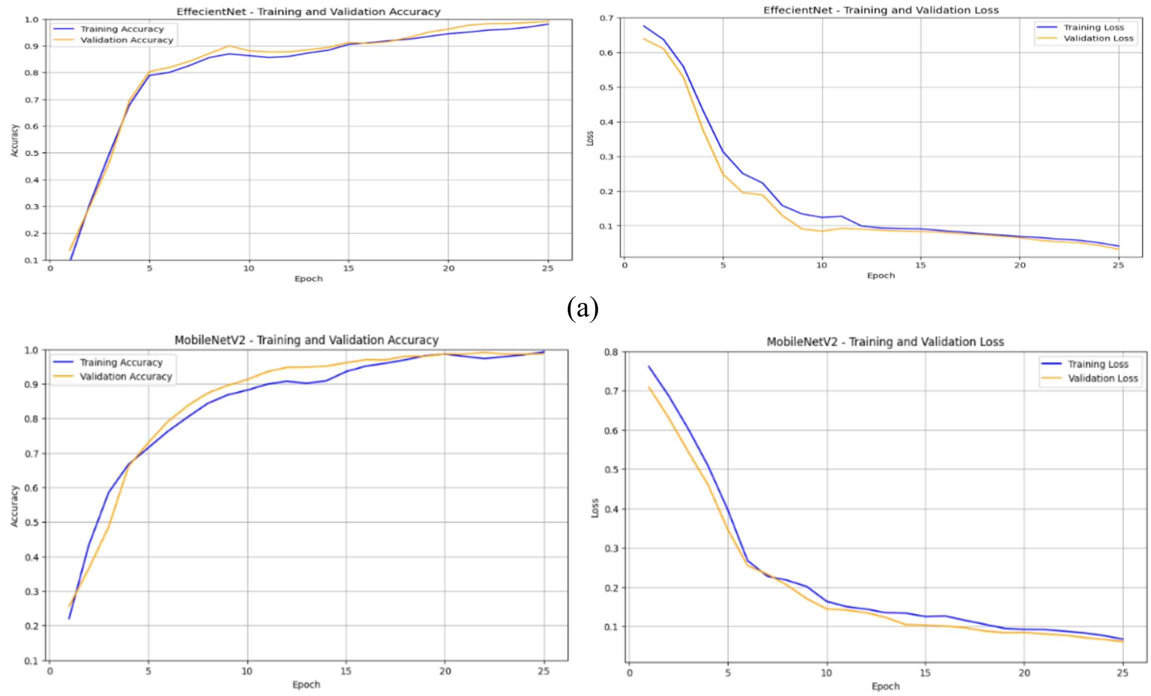


(a)

(b)

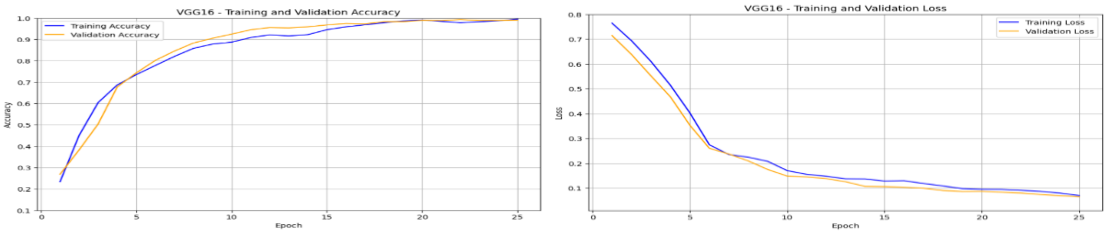
(c)

Figure 4 Accuracy and Loss Graph of (a) EfficientNet (b) Mobilenet V2 (c) VGG16 Private Dataset



(a)

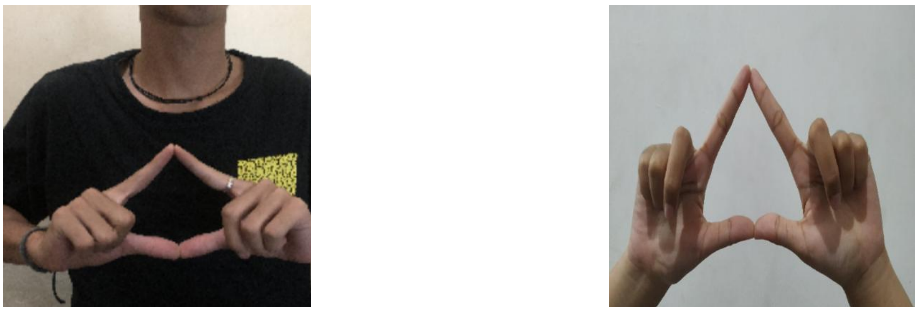
(b)



(c)

Figure 5 Accuracy and Loss Graph of (a) EfficientNet (b) Mobilenet V2 (c) VGG16 Public Dataset

An example of the system's classification performance on test data is shown in Figure 7, which presents sample images from both the private and public BISINDO datasets. The figure demonstrates how the trained CNN models correctly identify alphabet gesture classes such as "A" with high precision. The visual comparison between private and public samples indicates that the models can generalize well across different lighting conditions, backgrounds, and hand shapes. This confirms the robustness of the proposed approach in recognizing BISINDO alphabet gestures under varied real-world scenarios, as illustrated in Figure 7.



(a) (b)

Figure 6 Sample test data (a) Private (b) Public

The classification results of the three pre-trained models—EfficientNet, MobileNetV2, and VGG16—were tested on both private and public BISINDO datasets. The models successfully recognized the tested gesture class "A," with EfficientNet and MobileNetV2 achieving 100% accuracy on the public dataset, while VGG16 maintained consistent accuracy of 0.99 across both datasets. However, differences in processing time were observed, with EfficientNet performing the fastest (0.14 seconds on private data) and VGG16 requiring the longest computation time (up to 2.51 seconds). These findings indicate that although all models reached high accuracy, EfficientNet provides the most efficient computational performance, as shown in Table 2.

Table 2 Classification Result

Classification	EfficientNet		Mobilenet V2		VGG16	
	Private	Public	Private	Public	Private	Public
Predicted Class	A	A	A	A	A	A
Accuracy Score	1.0	1.0	0.95	1.0	0.99	0.99
Time (second)	0.14	0.7	1.48	1.6	2.51	1.89

The classification performance for each alphabet class in the BISINDO dataset across the three CNN architectures is analyzed in detail. The number of correct and incorrect predictions for both private and public datasets highlights that VGG16 generally achieves the highest accuracy, with perfect classifications for letters such as “A,” “G,” “O,” “W,” and “Z.” In contrast, letters with similar hand configurations—such as “M,” “N,” “L,” and “S”—show higher misclassification rates across all models. This result emphasizes that gesture similarity remains a key factor affecting recognition accuracy, as illustrated in Table 3.

Table 3. The Clasification Result of BISINDO

	EfficientNet				MobileNetV2				VGG16			
	Correct		Incorrect		Correct		Incorrect		Correct		Incorrect	
Class	Private	Public	Private	Public	Private	Public	Private	Public	Private	Public	Private	Public
A	38	84	2	4	38	84	2	4	40	88	0	0
B	38	84	2	6	38	85	2	5	38	82	2	8
C	37	85	3	1	37	80	3	6	40	85	0	1
D	36	82	4	8	38	81	2	9	39	82	1	8
E	38	87	2	2	38	87	2	2	40	87	0	2
F	38	89	2	1	39	85	1	5	38	89	2	1
G	38	90	2	0	39	88	1	2	40	90	0	0
H	37	85	3	2	38	81	2	6	39	82	1	5
I	38	83	2	7	38	86	2	4	39	86	1	4
J	38	88	2	2	37	84	3	6	39	88	1	2
K	36	87	4	0	35	86	5	1	40	83	0	4
L	34	79	6	11	37	84	3	6	39	89	1	1
M	34	81	6	9	37	79	3	11	32	81	8	9
N	38	75	2	15	37	79	3	11	40	69	0	21
O	38	90	2	0	38	90	2	0	39	90	1	0
P	30	90	10	0	38	87	2	3	39	86	1	4
Q	38	84	2	6	38	79	2	8	38	79	2	8
R	38	81	2	9	36	80	4	8	38	86	2	2
S	19	81	21	9	30	84	10	2	40	83	0	3
T	24	88	16	2	38	88	2	2	40	83	0	7
U	38	85	2	5	35	82	5	7	40	87	0	2
V	37	88	3	2	38	84	2	3	39	86	1	4
W	38	89	2	1	38	86	2	3	40	90	0	0
X	40	89	0	1	40	83	0	5	40	89	0	0
Y	37	81	3	9	39	74	1	13	38	80	2	3
Z	38	90	2	0	40	90	0	0	40	90	0	0

The evaluation metrics of accuracy, precision, and recall for each pre-trained model on private and public datasets are summarized. The VGG16 model achieved the highest accuracy of 97% on private data and 96% on public data, demonstrating strong and stable performance. EfficientNet obtained slightly lower accuracy on the private dataset (89%) but matched VGG16 on public data (96%), while MobileNetV2 maintained a consistent accuracy of 94% across both datasets. Overall, VGG16 exhibited the best balance between accuracy, precision, and recall, confirming its reliability for BISINDO alphabet classification, as presented in Table 4.

Table 4. Evaluation Results of Pre-Trained Model

Model pre-trained	Efficientnet		Mobilenet V2		VGG16	
	Private	Public	Private	Public	Private	Public
Accuracy	90%	96%	94%	94%	97%	96%
Precision	89%	96%	94%	94%	97%	96%
Recall	89%	96%	94%	94%	97%	96%

4. DISCUSSION

The BISINDO alphabet classification process consists of four main stages: image preprocessing, implementation of pre-trained models, model training, and performance evaluation. During preprocessing, images were augmented through resizing (224×224 pixels), rotation, shearing, and horizontal flipping to enhance dataset diversity and reduce overfitting. The augmented data were then used to train three pre-trained models—EfficientNet, MobileNetV2, and VGG16—for 25 epochs. Based on the confusion matrix results, EfficientNet and VGG16 achieved the highest accuracy of 96% on the public dataset, while MobileNetV2 obtained 94%. On the private dataset, VGG16 demonstrated superior recall and precision values, whereas EfficientNet recorded the lowest performance. Most classification errors occurred with gesture pairs such as **M–N**, **E–F**, and **L–I**, which share similar hand configurations and shapes. The letter **Z** achieved the best recognition performance, with only two misclassifications using EfficientNet. Overall, VGG16 exhibited stable and consistent results across datasets, while EfficientNet showed a notable accuracy gap between the public (96%) and private (89%) datasets. These findings highlight the effectiveness of CNN-based architectures in recognizing BISINDO alphabet gestures and contribute to the advancement of deep learning–based assistive technologies for low-resource sign languages, extending the application of CNNs beyond widely researched systems such as ASL.

5. CONCLUSION

This study presents significant findings in the classification of BISINDO alphabet gestures through the implementation and evaluation of three pre-trained Convolutional Neural Network (CNN) models: EfficientNet, MobileNetV2, and VGG16. The dataset used comprises 16,700 images, including 11,500 public images and 5,200 private images, distributed across 26 alphabet classes. The dataset was divided into two parts: 80% for training and 20% for testing. Experimental results show that all three models achieve strong classification performance with comparable accuracy when trained on the same dataset. Among them, EfficientNet and VGG16 achieved the highest accuracy of 96% on the public dataset, while MobileNetV2 followed closely with 94%. The VGG16 model demonstrated the most stable and consistent performance, achieving 97% accuracy on private data, whereas EfficientNet showed a decrease in accuracy to 89% on private data, indicating less robustness across different data sources. Overall, the results highlight that selecting an appropriate architecture—particularly VGG16—plays a crucial role in improving accuracy and reliability for automatic sign language recognition. Future work could focus on real-time deployment on mobile or embedded devices and integration into educational or assistive learning platforms to promote inclusive communication for the hearing-impaired community in Indonesia.

ACKNOWLEDGEMENT

The authors would like to express their gratitude to Mulawarman University for the support provided throughout this research.

REFERENCES

[1] S. Singh, A. Yadav, And G. C. Nandi, “Real-Time Static Hand Gesture Recognition Using Convolutional Neural Networks,” Computers & Electrical Engineering, Vol. 92, pp. 107109, 2021.

[2] M. Tran and H. Le, “Vietnamese Sign Language Recognition Using Convolutional Neural Networks and Data Augmentation,” International Journal of Advanced Computer Science and Applications, Vol. 12, No. 6, pp. 476–482, 2021.

[3] T. Bui, D. Nguyen, And N. Vo, “Skin Segmentation Preprocessing for Vietnamese Sign Language Recognition Using CNN,” Multimedia Tools and Applications, Vol. 82, No. 1, pp. 1097–1113, 2023.

[4] A. Chatterjee, M. Singh, And R. Jain, “Transfer Learning with Mobilenetv2 for Lightweight Indian Sign Language Recognition,” Procedia Computer Science, Vol. 171, pp. 1231–1238, 2020.

[5] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, And L. C. Chen, “Mobilenetv2: Inverted Residuals and Linear Bottlenecks,” Ieee Transactions on Pattern Analysis and Machine Intelligence, Vol. 42, No. 5, pp. 1257–1272, 2020.

[6] K. Nakamura and Y. Tanaka, "Application of EfficientNet for hand sign classification," Signal Processing: Image Communication, vol. 96, pp. 116344, 2021, Doi: 10.1016/j.image.2021.116344.

[7] A. Sharma and S. Kumar, "Advances in CNN architectures for image recognition," International Journal of Computer Vision, vol. 129, no. 6, pp. 2023-2041, 2023, Doi: 10.1007/s11263-023-01620-4.

[8] P. Kumar et al., "An overview of deep learning techniques in gesture recognition," International Journal of Pattern Recognition and Artificial Intelligence, vol. 36, no. 5, pp. 2256002, 2022, Doi: 10.1142/S0218001422560027.

[9] T. Wang, "Recent Trends in Deep Learning-Based Sign Language Translation," IEEE Access, vol. 10, pp. 122034-122047, 2022, Doi: 10.1109/ACCESS.2022.3224567.

[10] Y. Zhao, Y. Lin, And J. Wu, “Fast and Accurate CNN-Based Sign Language Recognition Using Real-Time Video Stream,” Neural Computing and Applications, Vol. 33, pp. 7927–7942, 2021.

[11] R. Elakkiya, N. Gopalakrishnan, And R. Prabu, “Sequence Modelling of Sign Language Recognition Using CNN-LSTM Hybrid Architecture,” Multimedia Tools and Applications, Vol. 82, pp. 17935–17958, 2023.

[12] K. Goyal, R. Singla, And T. Choudhury, “Deep Learning for Sign Language Recognition: Current Trends and Challenges,” Procedia Computer Science, Vol. 167, pp. 2481–2490, 2020.

[13] F. W. D. S. U. R. Jumaryadi, "Implementasi Convolutional Neural Network Dalam Klasifikasi Citra," Jurnal Teknik Informatika, vol. 6, no. 6, pp. 1530–1537, Dec. 2025.

[14] J. Zhang et al., "Sign Language Recognition with Deep Learning: A Review," IEEE Transactions on Neural Networks and Learning Systems, vol. 34, no. 2, pp. 456-472, 2023, Doi: 10.1109/TNNLS.2022.3201476.

[15] O. Alzubaidi Et Al., “Review of Deep Learning: Concepts, CNN Architectures, Challenges, Applications, Future Directions,” Journal of Big Data, Vol. 8, No. 1, Pp. 1–74, 2021.

[16] Z. H. Salsabila, R. R. Nurmalasari and L. Kamelia, "Indonesian Sign Language Translation System Using ResNet-50 Architecture-Based Convolutional Neural Network," 2024 10th

International Conference on Wireless and Telematics (ICWT), Batam, Indonesia, 2024, pp. 1-5, doi: 10.1109/ICWT62080.2024.10674686.

[17] M. M. Alnfiai, “Deep Learning-Based Sign Language Recognition for Hearing and Speaking Impaired People,” Intelligent Automation & Soft Computing, vol. 36, no. 2, pp. 1653–1669, 2023.

[18] A. N. Sihananto, E. M. Safitri, Y. Maulana, F. Fakhruddin, and M. E. Yudistira, "Indonesian Sign Language Image Detection Using Convolutional Neural Network (CNN) Method," Inspiration: Jurnal Teknologi Informasi dan Komunikasi, vol. 13, no. 1, pp. 13–21, 2023.

[19] L. Chen, Y. Li, And H. Wang, “Improving Hand Gesture Recognition Through Enhanced Image Preprocessing for CNN Models,” Journal of Visual Communication And Image Representation, Vol. 94, 103751, 2023.

[20] A. Lee and R. White, "Transfer learning in small-scale sign language datasets," Journal of Machine Learning Research, vol. 23, no. 42, pp. 1-20, 2022.

[21] S. Li and H. Fu, "Lightweight CNNs for Gesture Recognition on Mobile Devices," Pattern Recognition Letters, vol. 160, pp. 30-38, 2022, Doi: 10.1016/j.patrec.2022.06.012.

[22] A. Rahim, R. Khan, and S. Abbas, “Robust Sign Language Recognition Using CNN With Data Augmentation,” Journal of Ambient Intelligence and Humanized Computing, Vol. 13, No. 5, pp. 2395–2406, 2022.

[23] M. Lopez et al., "Data augmentation for hand gesture recognition using CNNs," Journal of Visual Communication and Image Representation, vol. 88, pp. 103539, 2022, Doi: 10.1016/j.jvcir.2022.103539.

[24] R. Hao, K. Namdar, L. Liu, M. A. Haider, And F. Khalvati, "A Comprehensive Study of Data Augmentation Strategies for Prostate Cancer Detection in Diffusion-Weighted MRI Using Convolutional Neural Networks," Journal of Digital Imaging, Vol. 34, Pp. 862–876, 2021, Doi: 10.1007/S10278-021-00478-7.

[25] N. Patel and D. Shah, "Optimizing CNN Models for Sign Language Recognition in Low-Resource Settings," Computer Vision and Image Understanding, vol. 220, pp. 103451, 2022, Doi: 10.1016/j.cviu.2022.103451.