Moestopo International Review on Social, Humanities, and Sciences (MIRSHuS)
Vol. 6, No. 2 (2026) pp. 385-403
ISSN: 2775-9601

Enhancing Indonesian Sign Language Recognition Accuracy Using Transfer Learning for Deaf Learners

Yuvi Darmayunata*
University of Lancang Kuning, Pekanbaru, Indonesia
Lucky Lhaura Van FC
University of Lancang Kuning, Pekanbaru, Indonesia
Veby
University of Lancang Kuning, Pekanbaru, Indonesia
Didik Siswanto
University of Lancang Kuning, Pekanbaru, Indonesia
Devia Kartika
University of Lancang Kuning, Pekanbaru, Indonesia

*Correspondence: yuvidarmayunata@unilak.ac.id

ARTICLE INFO

Article History:
received: 23/07/2026
revised: 18/08/2026
accepted: 07/09/2026

Keywords:
Bisindo; Deep Learning; Education Accessibility; Sign Language Recognition; Transfer Learning

DOI:
10.32509/mirshus.v6i2.225

ABSTRACT

Recognition by comparing MobileNetV2 and EfficientNetB0 on 2,659 static BISINDO alphabet images distributed across 26 classes. The models were trained using the same two-stage transfer learning protocol and evaluated using accuracy, precision, recall, F1-score, and confusion matrix analysis. MobileNetV2 achieved a test accuracy of 95.8% and showed strong class-level performance, although errors remained concentrated among visually similar gestures. In contrast, EfficientNetB0 obtained 3.85% test accuracy, which is approximately chance level for a 26-class task. Accordingly, this result should be interpreted as evidence of an unsuccessful EfficientNetB0 training or fine-tuning configuration in the present experiment, rather than as evidence that the architecture is inherently unsuitable for BISINDO recognition. Because the study did not include signer-independent validation, external testing, or deployment benchmarks, the findings are limited to the present image-level experimental setting. Within these conditions, MobileNetV2 provided the stronger classification performance, while further verification and architecture-specific optimization are required before drawing broader conclusions about comparative robustness or practical deployment.

INTRODUCTION

Sign language serves as the primary communication medium for Deaf and hard-of-hearing individuals, enabling effective communication, education, and social participation. In

Indonesia, Bahasa Isyarat Indonesia (BISINDO) is widely used by the Deaf community as a natural language for daily interaction (Papatsimouli et al., 2023) (Das et al., 2023). However, communication barriers remain prevalent because most hearing individuals, including teachers, students, public service providers, and healthcare professionals, are not proficient in BISINDO (Dwijayanti et al., 2021) . These limitations often restrict equal access to education, public services, and social interaction, thereby reducing opportunities for Deaf individuals to participate fully in academic and community activities. As the implementation of inclusive education and digital accessibility continues to expand, there is an increasing demand for intelligent assistive technologies capable of bridging communication between Deaf and hearing communities (Xia et al., 2022). Automatic sign language recognition systems have therefore emerged as a promising solution because they enable hand gestures to be translated into textual or spoken language, facilitating more effective communication in educational and public environments (Daniels et al., 2021) (Kelana et al., 2025).

Recent advances in artificial intelligence, particularly in deep learning and computer vision, have substantially improved the performance of automatic sign language recognition systems (Sutjiadi, 2023) . Among various deep learning techniques, Convolutional Neural Networks (CNNs) have demonstrated remarkable capability in extracting discriminative visual features directly from hand gesture images without requiring handcrafted feature engineering (Iqbal et al., 2023). Nevertheless, training deep neural networks from scratch generally requires large-scale annotated datasets and considerable computational resources, both of which are often unavailable for local sign languages such as BISINDO (Dwijayanti et al., 2021). To overcome these limitations, transfer learning has become a widely adopted approach by adapting knowledge learned from large-scale image datasets, such as ImageNet, to specialized recognition tasks through fine-tuning. This strategy reduces computational cost, accelerates model convergence, and generally improves classification performance when only limited training data are available, making it particularly suitable for BISINDO recognition (Töngi, 2021) (Kindiroglu et al., 2024).

Despite these advances, developing an accurate recognition system for BISINDO remains challenging. Compared with internationally studied sign languages such as American Sign Language (ASL), BISINDO has relatively limited publicly available datasets and exhibits substantial variations in hand orientation, lighting conditions, signer characteristics, and gesture execution (Miah et al., 2024). These factors reduce model generalization and frequently lead to misclassification, particularly for visually similar hand gestures (Briones Cerquín et al., 2025). Consequently, recognition models developed for other sign languages cannot be directly generalized to BISINDO without appropriate adaptation and fine-tuning using representative local datasets. Addressing these challenges is essential to ensure that sign language recognition systems can operate reliably in practical educational settings while supporting accessible communication between Deaf and hearing communities (Rizka Fadhillah et al., 2024) (Agustiansyah & Kurniadi, 2025).

Several recent studies have explored deep learning approaches for BISINDO recognition using different model architectures.(Daniels et al., 2021) introduced a YOLO-based framework for Indonesian sign language detection and demonstrated the feasibility of real-

time recognition, although classification performance remained limited for visually similar gestures. (Joan et al., 2023) investigated transfer learning for BISINDO hand-sign detection and reported that pre-trained CNN models significantly improved recognition performance compared with conventional training approaches. Furthermore, (Rizka Fadhillah et al., 2024) implemented EfficientNet-based transfer learning and highlighted the importance of dataset quality and hyperparameter optimization for improving recognition accuracy. More recently, (Agustiansyah & Kurniadi, 2025) demonstrated that Vision Transformer (ViT) architectures achieved promising performance for BISINDO alphabet classification; however, these models generally require higher computational resources than lightweight CNN architectures, limiting their applicability on resource-constrained devices (Sadik et al., 2023).

Although previous studies have demonstrated the effectiveness of deep learning for BISINDO recognition, several research gaps remain (Lamaakal et al., 2026). First, relatively few studies have directly compared lightweight transfer learning architectures under identical experimental conditions using the same BISINDO dataset (Dwijayanti et al., 2021). Second, previous research has primarily emphasized overall classification accuracy while providing limited analysis of class-level performance and misclassification patterns among visually similar gestures (Abdullahi & Chamnongthai, 2022) (Hugar & Kagalkar, 2025). Third, the suitability of lightweight CNN architectures for educational assistive technologies requiring both high recognition accuracy and computational efficiency has not been comprehensively investigated. These limitations indicate the need for a systematic comparison of lightweight transfer learning models using a consistent experimental framework (Alaftekin et al., 2024) (Wahyudi et al., 2025).

To address these gaps, this study investigates the application of transfer learning using two lightweight pre-trained CNN architectures, MobileNetV2 and EfficientNetB0, for Indonesian Sign Language recognition (Hugar & Kagalkar, 2025; Sutjiadi, 2023; Zufria et al., 2025). Both models are fine-tuned using the same BISINDO alphabet dataset and evaluated under identical experimental settings using accuracy, precision, recall, F1-score, confusion matrix analysis, and class-level performance metrics (Fauzi & Haqdu D, 2025) (Ridwan et al., 2024). Beyond comparing overall recognition accuracy, this study also analyzes common misclassification patterns among visually similar gestures to provide a more comprehensive evaluation of each model's performance (Baumgartl & Buettner, 2021) (Balat et al., 2024). The findings are expected to contribute to the development of efficient and accurate BISINDO recognition systems while providing practical insights into selecting lightweight deep learning architectures for inclusive educational technologies and assistive communication systems for the Indonesian Deaf community (Dwijayanti et al., 2021; Podder et al., 2023) .

METHOD

Figure 1 illustrates the overall research framework employed in this study. The process begins with the collection of the Indonesian Sign Language (BISINDO) dataset, followed by image preprocessing to improve data quality through normalization and augmentation. The preprocessed dataset is then used to train two transfer learning models, namely MobileNetV2 and EfficientNetB0, under identical experimental settings. Finally, both models are evaluated

using multiple classification performance metrics, and the results are compared to determine the most effective lightweight architecture for BISINDO recognition

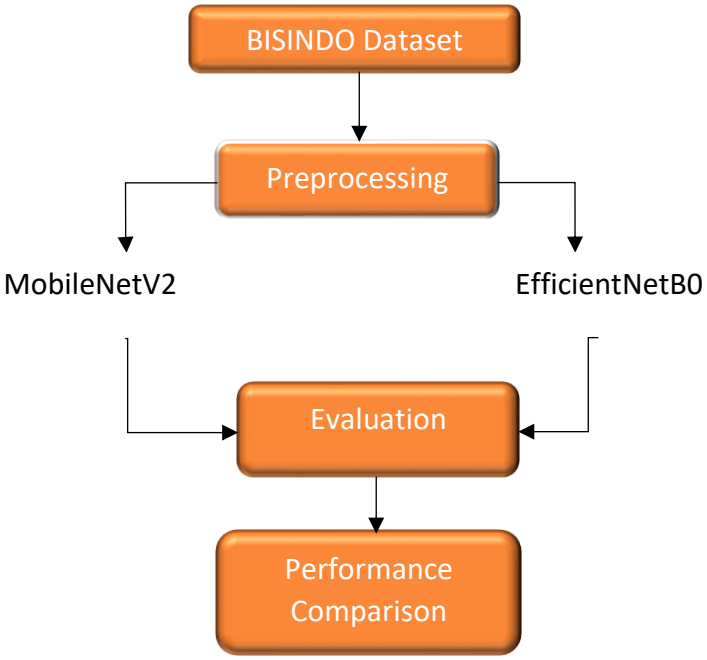

Figure 1. Overview of the Proposed Research Framework

This study employed an experimental research design to evaluate transfer learning for Indonesian Sign Language (BISINDO) recognition under a controlled image-classification setting. The primary data consisted of 2,659 static BISINDO alphabet images compiled from publicly available image sources, while secondary data were obtained from scientific literature on deep learning, transfer learning, and sign language recognition. The experiments were implemented in Python using TensorFlow and Keras, with OpenCV for image preprocessing. Model training and evaluation were conducted on a workstation equipped with an AMD Ryzen 7 6800HS processor, Radeon Graphics at 3.20 GHz, 16 GB RAM, and a 64-bit Windows operating system. The available dataset compilation record did not preserve complete source-level provenance fields, including the original dataset names, source-specific image counts, signer identifiers, session identifiers, and duplicate or near-duplicate flags. Therefore, the present study cannot independently verify source-wise composition or signer-level independence, and this limitation is considered when interpreting the reported generalization performance.

The methodology followed a two-stage transfer learning framework. Images were normalized from the original 0 to 255 intensity range to 0 to 1, and augmentation was applied only to the training subset using rotation, width and height shifts, shear transformation, zooming, and horizontal flipping. The available experimental record does not document a class-specific horizontal-flip mask or a linguistic validation procedure. The reported implementation must therefore be interpreted as using a global horizontal-flip setting, which may have generated semantically invalid samples for non-mirror-invariant signs. The compiled

dataset was divided into 2,113 training images, 260 validation images, and 286 testing images, with the subsets treated as mutually exclusive at the image-file level. However, because signer and recording-session metadata were not available in the dataset record used for this study, the split cannot be claimed to be signer-independent, and overlap in signer identity or recording conditions across subsets cannot be ruled out. MobileNetV2 and EfficientNetB0 were initialized with ImageNet weights. Their original classification layers were replaced with a Global Average Pooling layer, a Dense layer with 128 ReLU units, and a 26-class Softmax output layer. In the feature-extraction stage, the backbone remained frozen for 20 epochs. Fine-tuning then continued for 10 epochs with the backbone unfrozen and a lower learning rate. Performance was evaluated using accuracy, precision, recall, F1-score, and confusion matrix analysis. Because the same optimization settings were applied to both architectures without architecture-specific hyperparameter search, the comparison should be interpreted as a controlled baseline comparison rather than proof that each architecture was evaluated under its individually optimal configuration.

RESULT AND DISCUSSION

Bisindo Dataset Description

This section describes the BISINDO dataset used for model development and the preprocessing procedures applied before training. The dataset contains 2,659 static images representing 26 alphabet gestures from A to Z and was compiled from publicly available image sources. It was divided into 2,113 training images, 260 validation images, and 286 testing images. The subsets were separated at the image-file level to support model training, validation, and testing. However, complete dataset provenance was not retained in the experimental record available for this manuscript. Specifically, the source dataset names, repository URLs or DOIs, original dataset authors, image counts contributed by each source, signer identifiers, session identifiers, and duplicate or near-duplicate screening records were not available for verification. This missing provenance prevents confirmation that the testing set is independent at the signer or recording-session level. Consequently, the reported test performance should be interpreted as image-level performance on the compiled dataset rather than as a validated signer-independent estimate of real-world generalization. Future dataset curation should preserve source and signer metadata and should remove duplicate or near-duplicate images before partitioning.

Table 1. Bisindo Dataset Statistics

Dataset	Number of Samples	Number of Classes
Training	2113	26
Validation	260	26
Testing	286	26

The three subsets contain all 26 gesture classes and provide a consistent basis for calculating class-level performance metrics. Nevertheless, class coverage alone does not guarantee independence between training and testing data. Because signer and session

metadata were unavailable, the possibility that visually related images from the same signer, background, or recording session occur across subsets cannot be excluded. This is a potential source of data leakage in image-level sign language recognition and may inflate apparent generalization performance. A stronger evaluation would use signer-independent or session-independent partitioning whenever the required metadata are available (Rizka Fadhillah et al., 2024).

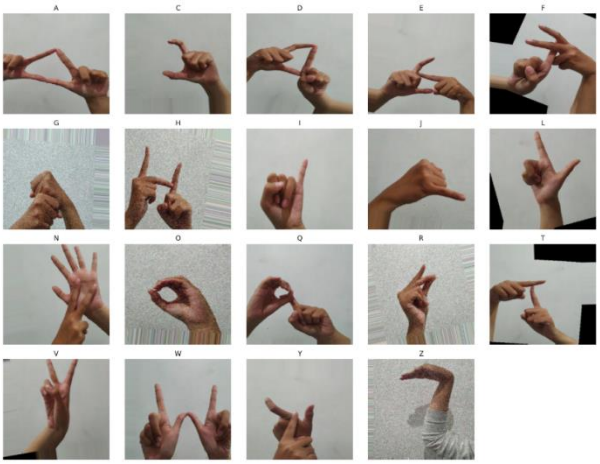

Figure 1. Example Bisindo Data Samples per Label

Figure 1 demonstrates the visual diversity of the BISINDO alphabet gestures. Although each class corresponds to a unique hand configuration, several gestures exhibit highly similar finger positions or hand orientations. Such similarities increase the complexity of the classification task because deep learning models must distinguish between subtle visual characteristics rather than coarse shape differences. This observation partially explains why certain gesture pairs become more susceptible to misclassification during model evaluation.

Data Pre-processing Stages

Data preprocessing is an essential stage in deep learning because it improves data consistency, reduces unwanted variations, and enhances the model's ability to learn discriminative features. Previous studies have demonstrated that appropriate preprocessing and augmentation strategies substantially improve recognition accuracy, especially when the available dataset is relatively small (Daniels et al., 2021; Kindiroglu et al., 2024) . In this study, preprocessing was performed using the Image Data Generator module available in the TensorFlow/Keras framework. The preprocessing pipeline consisted of two primary stages: pixel normalization and data augmentation. All input images were normalized by rescaling pixel values from the original range of 0–255 into the interval 0–1 using a rescaling factor of 1/255. Normalization reduces numerical instability during gradient optimization, accelerates model convergence, and enables more efficient weight updates throughout the training process. Furthermore, normalized inputs improve compatibility with transfer learning models pre-trained on the ImageNet dataset because the input distributions become more consistent with those encountered during pre-training (TensorFlow Developers, 2024).

To increase dataset diversity and reduce overfitting, augmentation was applied exclusively to the training subset. The configured transformations were random rotation (**20°**), horizontal translation (**20%**), vertical translation (**20%**), shear transformation (**20%**), zoom transformation (**20%**), horizontal flipping, and nearest-neighbour interpolation for filling empty pixels. Rotation, translation, shear, and zoom were intended to model modest variation in camera position and gesture execution; however, their semantic validity still depends on the transformed handshape remaining recognisable as the original class. Similar geometric augmentation is commonly used in small sign-language image datasets (Kindiroglu et al., 2024; Rizka Fadhillah et al., 2024). Horizontal flipping requires a stricter linguistic condition because a mirror image can alter handedness or left-right orientation. Contrary to the earlier wording, the archived experiment does not contain a class-specific whitelist, expert annotation sheet, or ablation test showing which BISINDO signs were mirror-invariant. It is therefore not possible to claim that horizontal flipping preserved the label for any particular class in the present run.

Class-level horizontal-flip status: classes A-Z were not individually documented as safe or unsafe. Accordingly, none of the 26 classes can be defended as mirror-invariant from the retained experimental record, and flip-generated training samples must be treated as a potential source of label noise. In a verified rerun, horizontal flipping should be disabled by default and enabled separately for a class only after at least two fluent BISINDO users independently confirm that the mirrored token retains the same alphabet label and acceptable handedness. Any disagreement, directional contrast, or change in hand configuration should exclude that class from horizontal flipping. A per-class augmentation mask and an ablation comparison with and without flipping should then be retained.

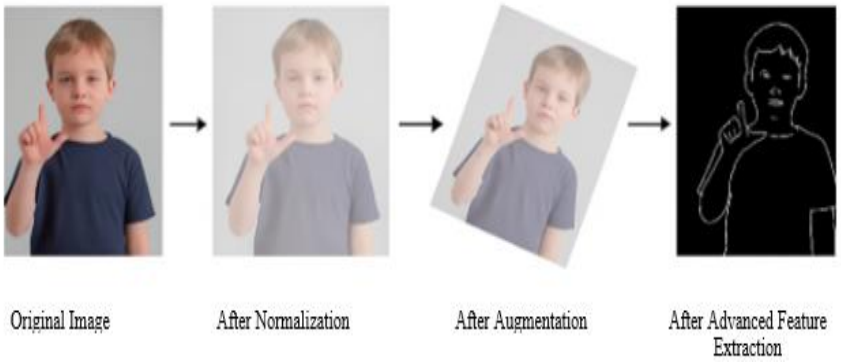

Figure 2. Illustration of Bisindo Data Pre-processing Stages (Conceptual)

As illustrated in Figure 2, each original image first undergoes normalization before being subjected to augmentation operations. The resulting augmented images provide greater variability during model training, enabling the transfer learning models to learn more generalized visual representations instead of memorizing specific training samples. This preprocessing strategy is particularly beneficial for relatively small datasets such as BISINDO because it effectively increases the diversity of training data without requiring additional manual data collection. Overall, the preprocessing pipeline established a consistent and

standardized input representation for both MobileNetV2 and EfficientNetB0. By reducing data variability unrelated to gesture semantics while simultaneously increasing training diversity, the preprocessing stage contributed to improved learning efficiency and provided a strong foundation for the transfer learning process evaluated in the subsequent sections.

Development Environment and Hardware

The implementation of the proposed BISINDO recognition system was carried out using the Python programming language with the TensorFlow and Keras deep learning libraries. These frameworks were selected because they provide comprehensive support for convolutional neural networks (CNNs), transfer learning, automatic differentiation, and GPU-accelerated computation, making them suitable for large-scale image classification tasks. The availability of pre-trained models within the Keras Applications module also facilitates efficient transfer learning by allowing previously learned visual representations to be adapted to new classification problems. Model training and evaluation were performed on a workstation equipped with an AMD Ryzen 7 6800HS processor, Radeon Graphics running at 3.20 GHz, 16 GB RAM, and a 64-bit Windows operating system. Although the hardware configuration represents a mid-range computing platform, it provides sufficient computational capability for fine-tuning lightweight convolutional neural network architectures such as MobileNetV2 and EfficientNetB0. The use of GPU acceleration substantially reduced the computational time required during both feature extraction and fine-tuning, enabling efficient experimentation with different model configurations. Unlike conventional CNN training from scratch, transfer learning significantly reduces computational requirements because the majority of low-level visual features have already been learned from the large-scale ImageNet dataset. Consequently, only task-specific parameters require adaptation during training, making this approach particularly suitable for applications involving relatively limited datasets such as BISINDO (Wahyudi et al., 2025).

Transfer Learning Model Configuration

This study employed two pre-trained convolutional neural network architectures, namely MobileNetV2 and EfficientNetB0, to evaluate their effectiveness for BISINDO alphabet recognition. Both models were initialized using ImageNet pre-trained weights while excluding the original classification layer (include_top=False). This strategy enables the models to preserve generic visual representations while adapting the final classification layer to the characteristics of BISINDO hand gestures (Tan & Le, 2021; Wahyudi et al., 2025) . A new classification head was added to each backbone network, consisting of a Global Average Pooling layer, a Dense layer with 128 neurons using ReLU activation, and a final Softmax layer containing 26 output neurons corresponding to the BISINDO alphabet classes. Global Average Pooling was selected because it substantially reduces the number of trainable parameters while maintaining discriminative spatial information extracted by convolutional layers, thereby reducing the likelihood of overfitting. The transfer learning process was performed in two sequential stages.

During the feature extraction stage, all convolutional layers were frozen (base_model.trainable=False) while only the newly added classifier layers were trained. This strategy enables the model to adapt efficiently to the target dataset without modifying the generic visual representations learned from ImageNet (TensorFlow Developers, 2024). The models were trained for 20 epochs using the Adam optimizer, categorical cross-entropy loss function, and an initial learning rate of 0.001. Subsequently, the models entered the fine-tuning stage, during which all convolutional layers were unfrozen (base_model.trainable=True). Fine-tuning allows the high-level feature representations to adapt specifically to BISINDO gesture characteristics that differ from natural images contained in ImageNet. To prevent excessive parameter updates that could damage previously learned representations, the learning rate was reduced to 0.0001, and training continued for an additional 10 epochs. Previous studies have demonstrated that progressive fine-tuning generally improves model convergence and classification accuracy while preserving the advantages of transfer learning (Agustiani et al., 2025; Zaelani & Miftahuddin, 2022). The optimization process employed the Adam optimizer, which combines adaptive learning rates with momentum-based optimization. Adam has consistently demonstrated stable convergence and competitive performance in transfer learning applications involving image classification tasks, particularly when only a limited amount of labeled training data is available (Kingma & Ba, 2014; TensorFlow Developers, 2024).

Table 2. Model Training Configuration Parameters

Parameter	MobileNetV2 (Initial)	MobileNetV2 (Fine-tune)	EfficientNetB0 (Initial)	EfficientNetB0 (Fine-tune)
Optimizer	Adam	Adam	Adam	Adam
Learning Rate	0.001	0.0001	0.001	0.0001
Epochs	20	10	20	10
Batch Size	32	32	32	32
Loss Function	Categorical Crossentropy	Categorical Crossentropy	Categorical Crossentropy	Categorical Crossentropy
Base Model Trainable	FALSE	TRUE	FALSE	TRUE

(Catatan: Sesuaikan nilai "Epochs" di Tabel ini agar konsisten dengan teks (Initial: 20, Fine-tune: 10).)

Applying identical training settings to MobileNetV2 and EfficientNetB0 provides experimental consistency, but it does not guarantee that the comparison is equally optimized for both architectures (Agustiani et al., 2025). In this study, both models used Adam, the same initial and fine-tuning learning rates, the same batch size, and the same two-stage schedule, and all backbone layers were unfrozen during the fine-tuning stage. These choices establish a controlled baseline. However, the optimal learning rate, depth of layer unfreezing, treatment of batch-normalization layers, and fine-tuning duration may differ between architectures. No architecture-specific hyperparameter optimization was conducted. Therefore, differences in performance cannot be attributed solely to architectural characteristics, particularly given the

chance-level EfficientNetB0 result. The EfficientNetB0 outcome should also be examined as a possible implementation or optimization failure before any architecture-level conclusion is made.

Training Performance Analysis

Training and validation behaviour was examined to assess whether the models learned useful representations and whether the optimization process was consistent with their final test performance. This distinction is essential in the present study because MobileNetV2 achieved high test accuracy, whereas EfficientNetB0 produced 3.85% test accuracy, approximately equal to random guessing in a 26-class classification problem. Therefore, smooth-looking curves or decreasing loss values cannot by themselves be used to claim satisfactory convergence for EfficientNetB0 if the final checkpoint fails to produce meaningful test discrimination.

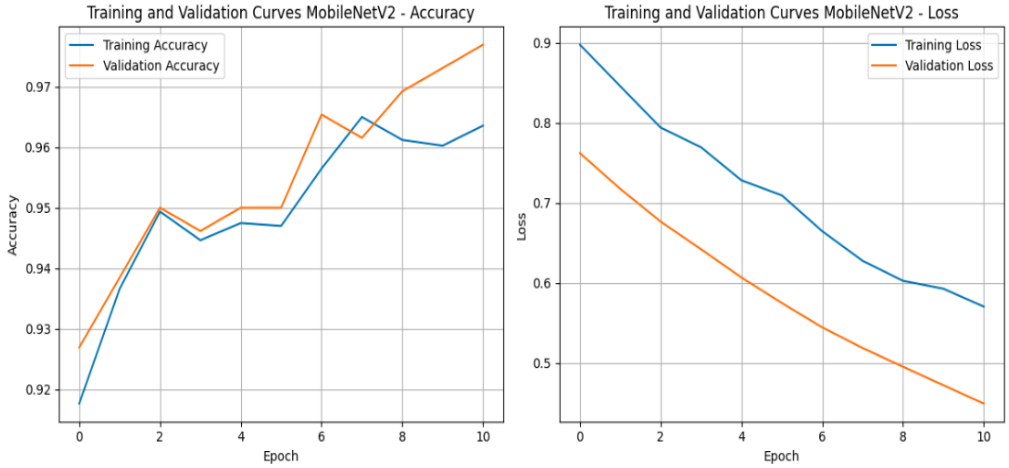

Figure 3. Training and Validation Curves for Best Model

Figure 3 should be interpreted primarily in relation to the best-performing model, MobileNetV2. Its learning curves show that training progressed toward a high-performing classifier, which is consistent with the final test accuracy of 95.8% and the class-level metrics reported below. By contrast, the reported 3.85% EfficientNetB0 test accuracy is inconsistent with any claim that EfficientNetB0 achieved satisfactory task-level convergence. A 3.85% result corresponds approximately to chance-level prediction for 26 classes. Therefore, the present manuscript does not treat stable-looking EfficientNetB0 training or validation curves as evidence of successful recognition performance. The EfficientNetB0 experiment requires re-verification using the exact saved checkpoint, training and validation metrics, test evaluation, classification report, and confusion matrix before its failure can be confidently attributed to the architecture itself (Balat et al., 2024; Fauzi & Haqdu D, 2025).

The MobileNetV2 result supports the usefulness of transfer learning for adapting ImageNet features to the BISINDO alphabet task under the present dataset conditions (Rachmawati et al., 2023; Ridwan et al., 2024). However, the EfficientNetB0 result shows that a transfer learning pipeline can fail despite using the same nominal training protocol. This

difference also highlights why identical hyperparameters should not automatically be interpreted as equally suitable hyperparameters. Architecture-specific fine-tuning choices may materially influence final performance, especially when the target dataset is small and contains limited variation (Balat et al., 2024; Töngi, 2021).

For MobileNetV2, the alignment between learning behaviour and test performance provides a coherent basis for subsequent precision, recall, F1-score, and confusion matrix analysis. The same conclusion cannot currently be extended to EfficientNetB0. Until the EfficientNetB0 run is re-verified, its 3.85% result should be reported transparently as a failed or non-convergent task-level outcome in this experimental configuration. Training accuracy, validation accuracy, training and validation loss, the exact test checkpoint, precision, recall, F1-score, classification report, and confusion matrix for EfficientNetB0 should be retained and reported in a future verified rerun. This interpretation avoids overstating convergence and keeps the discussion consistent with the observed test result. Similar findings have been reported in recent transfer learning studies, where progressive fine-tuning contributes to improved convergence stability and enhanced recognition performance across various image classification tasks (Baumgartl & Buettner, 2021b). Accordingly, only MobileNetV2 can be described as having achieved task-level convergence in the reported experiment. EfficientNetB0's 3.85% test accuracy represents an unsuccessful or insufficiently verified outcome, and no satisfactory generalization claim is made for that run. The preprocessing and shared hyperparameter configuration also cannot be declared appropriate for EfficientNetB0 until the experiment is reproduced and checked.

Error Analysis and Discussion

A detailed evaluation of the best-performing model, MobileNetV2, was conducted to examine its classification capability at the individual class level and to identify common error patterns. While the overall classification accuracy reached **96%**, analysing precision, recall, F1-score, and confusion matrices provides deeper insights into the strengths and limitations of the proposed transfer learning framework. Per-class evaluation is particularly important in sign language recognition because overall accuracy alone may conceal difficulties in distinguishing visually similar gestures (Balat et al., 2024).

Per-Class Performance Analysis

Table 3 summarizes the precision, recall, and F1-score obtained for each BISINDO alphabet class. Most gesture classes demonstrated outstanding recognition performance, with 20 out of 26 classes achieving perfect F1-scores (1.00). These include classes A, B, D, F, G, H, J, K, L, O, P, Q, R, T, X, and Z, indicating that the model consistently recognized these gestures across all testing samples. Such results suggest that the extracted visual features were sufficiently distinctive to enable accurate discrimination among these gesture classes.

Table 3. Per-Class Performance Metrics for the Best Model

Label	Precision	Recall	F1-Score	Support
A	1	1	1	11
B	1	1	1	11

C	0.79	1	0.88	11
D	1	1	1	11
E	1	0.91	0.95	11
F	1	1	1	11
G	1	1	1	11
H	1	1	1	11
I	0.92	1	0.96	11
J	1	1	1	11
K	1	1	1	11
L	1	1	1	11
M	0.88	0.64	0.74	11
N	0.71	0.91	0.8	11
O	1	1	1	11
P	1	1	1	11
Q	1	1	1	11
R	1	1	1	11
S	1	0.82	0.9	11
T	1	1	1	11
U	1	0.73	0.84	11
V	1	0.91	0.95	11
W	0.92	1	0.96	11
X	1	1	1	11
Y	0.85	1	0.92	11
Z	1	1	1	11
accuracy			0.96	286
macro avg	0.96	0.96	0.96	286
weighted avg	0.96	0.96	0.96	286

However, several gesture classes exhibited comparatively lower recognition performance, namely C, E, I, M, N, S, U, V, W, and Y. Among these classes, M produced the lowest F1-score (0.74) because its recall value decreased to 0.64, indicating that four of the eleven true samples were incorrectly classified. Likewise, class N achieved an F1-score of 0.80, primarily due to its relatively low precision (0.71), suggesting that samples from other classes were frequently predicted as N. Similar behaviour was observed for class C, whose lower precision (0.79) indicates the presence of false-positive predictions. These findings indicate that classification errors are not uniformly distributed across all BISINDO gestures but are concentrated among gestures sharing similar hand configurations or finger arrangements. Previous studies have reported that hand shape similarity, subtle finger positioning, and slight variations in wrist orientation remain major challenges in static sign language recognition systems because CNN-based feature extractors primarily rely on visual appearance rather than contextual motion information (Balat et al., 2024; Rachmawati et al., 2023). Compared with previous BISINDO recognition studies that evaluated only a very limited number of

testing samples per class, the present study employed 11 held-out image-file-level test samples for every gesture class. This balanced support enables a descriptive comparison of class-level errors within the compiled dataset, but it does not establish signer-independent generalization.

Confusion Matrix Analysis

To further investigate classification behaviour, Figure 6 presents the confusion matrix generated by the MobileNetV2 model.

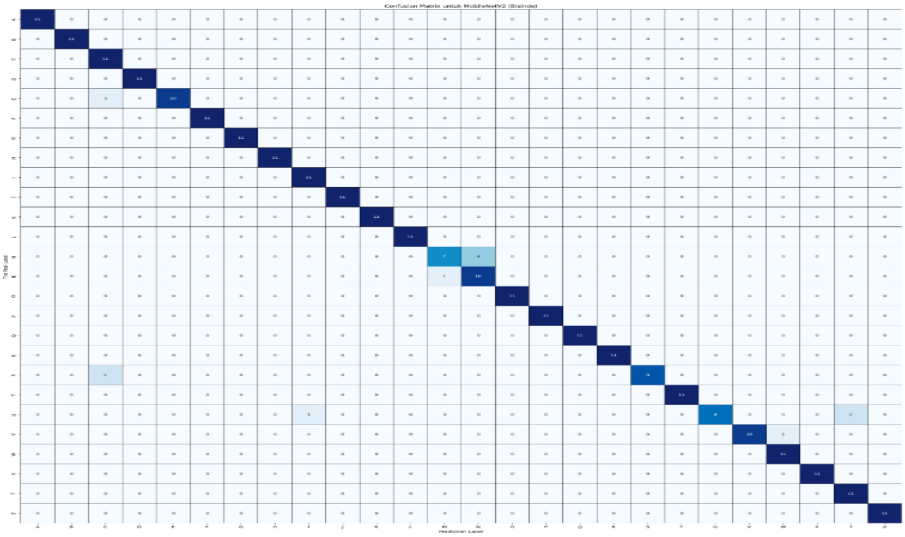

Figure 4. Confusion Matrix for the Best Model

The confusion matrix reveals that most gesture classes were correctly classified, as indicated by the dominant diagonal elements. Nevertheless, several systematic misclassification patterns were observed. The most frequent error occurred between the M and N gestures, where four samples of M were classified as N, while one sample of N was predicted as M. Additional confusion was observed between E and C, as well as S and C, indicating that these gesture pairs possess similar visual characteristics that are difficult to distinguish using static image representations. These error patterns suggest that MobileNetV2 successfully learned general visual features but encountered limitations when differentiating gestures with minimal inter-class variation. Similar observations have been reported by Abiyyu and Rahardi (2026), who found that lightweight transfer learning models often experience performance degradation when gesture classes differ only by subtle finger placement or palm orientation. Likewise, Fauzi et al. (2025) (Fauzi & Haqdu D, 2025b) demonstrated that MobileNet-based architectures generally achieve excellent computational efficiency but may confuse visually similar classes when trained on relatively limited datasets. Another contributing factor is the use of single-frame static images. Unlike video-based recognition systems, static-image approaches cannot utilize temporal motion cues that naturally distinguish sequential hand movements. Consequently, gestures that appear visually similar in a single frame become considerably more challenging to classify correctly. Recent studies have therefore suggested integrating temporal feature extraction methods, such as Long

Short-Term Memory (LSTM) networks or Vision Transformers, to improve discrimination among highly similar gestures (Balat et al., 2024; Ridwan et al., 2024b).

Qualitative Analysis of Classification Results

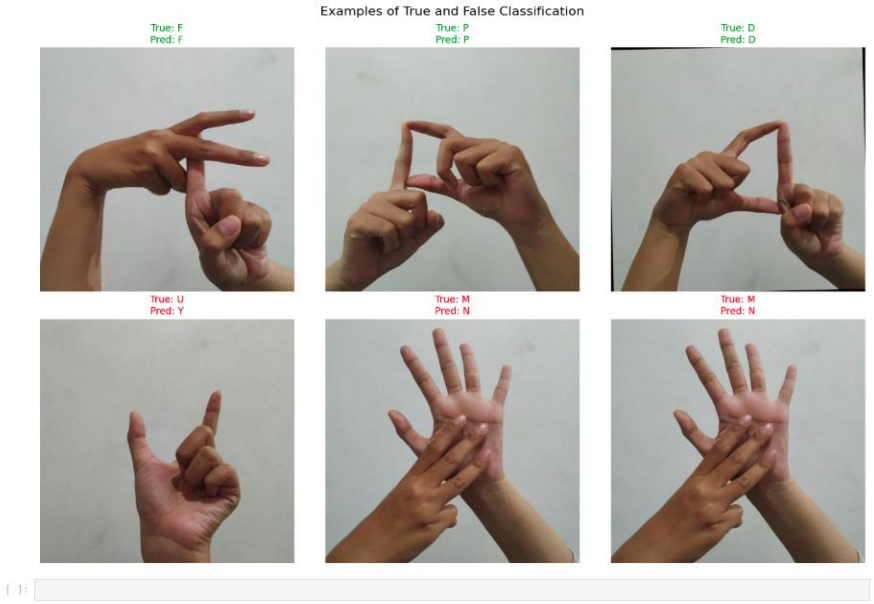

Figure 5. Examples of True and False Classification

The correctly classified examples illustrate the model's ability to learn stable and discriminative feature representations for gestures with distinctive visual structures. For example, gestures F, P, and D were consistently recognized with high confidence because their finger configurations differ substantially from those of other alphabet signs. Conversely, the misclassified examples demonstrate that recognition errors predominantly occur between visually similar gestures. In particular, the gesture U was incorrectly predicted as Y, while multiple M gestures were classified as N. These findings indicate that the extracted convolutional features were insufficient to capture subtle differences in finger overlap and hand orientation. Similar observations have been reported in recent sign language recognition studies, where confusion frequently occurs among gestures sharing comparable geometric structures despite high overall classification accuracy (Kindiroglu et al., 2024).

Technical Challenges and Research Implications

Several technical limitations affect the interpretation of the results. First, the BISINDO dataset is relatively small, and the available compilation record does not contain signer or session identifiers. Consequently, signer-independent evaluation could not be performed, and the possibility of signer-specific or session-specific visual information appearing across data subsets cannot be excluded. Second, the study uses static images, so temporal cues that may help distinguish visually similar gestures are unavailable. This limitation is relevant to confusions such as M and N, U and Y, and C and S. Third, variations in illumination, camera angle, image resolution, and background may influence feature extraction. Fourth, the retained record does not identify any BISINDO class as validated for horizontal flipping. If the

global flip setting produced a non-equivalent mirror image, the resulting training sample retained an incorrect label and may have affected either model. Although normalization and the other geometric transformations were used to reduce sensitivity to nuisance variation, no augmentation ablation, external dataset, cross-dataset evaluation, or real-world camera test was conducted. These limitations mean that the high MobileNetV2 accuracy should not be interpreted as evidence of robustness across unseen signers or deployment environments (Kindiroglu et al., 2024).

Computational constraints also limited the scope of model optimization. The same learning-rate schedule and fine-tuning procedure were used for both architectures, and extensive architecture-specific hyperparameter search was not performed. This is particularly important for interpreting the EfficientNetB0 result because its 3.85% test accuracy may reflect an unsuitable fine-tuning configuration, checkpoint issue, or other implementation-related factor. Future experiments should optimize each architecture separately and document the handling of batch-normalization layers, the exact layers unfrozen during fine-tuning, checkpoint-selection criteria, and repeated runs with controlled random seeds. Such verification is necessary before making a definitive comparative claim about architectural superiority.

Computational Efficiency and Deployment Limitations

This study evaluated transfer learning for Indonesian Sign Language (BISINDO) alphabet recognition by comparing MobileNetV2 and EfficientNetB0 under the same baseline training protocol. MobileNetV2 achieved 95.8% test accuracy and strong precision, recall, and F1-score values on the compiled image-level test set. EfficientNetB0, however, achieved only 3.85% test accuracy, approximately equivalent to random guessing for 26 classes. This result is not consistent with a claim of satisfactory EfficientNetB0 convergence and should be treated as a failed or insufficiently verified training and fine-tuning outcome in the present configuration. The comparison therefore shows that MobileNetV2 performed substantially better in this specific experiment, but it does not establish that MobileNetV2 is inherently superior to EfficientNetB0 across optimized configurations.

Class-level evaluation showed that most BISINDO gestures were recognized accurately by MobileNetV2, while errors were concentrated among visually similar signs with subtle differences in finger configuration and hand orientation. The interpretation of these findings is limited by the relatively small static-image dataset, incomplete source and signer metadata, the absence of signer-independent splitting, the lack of external or real-world testing, and the absence of direct computational deployment benchmarks. Accordingly, MobileNetV2 should be described as the better-performing model within the present dataset and experimental configuration, not as a fully validated robust or deployment-ready BISINDO recognition system. Future work should first re-run and verify EfficientNetB0 using architecture-specific optimization and documented checkpoints, then evaluate both models using signer-independent data partitions, external datasets, and measured latency, memory, model size, and mobile or CPU inference performance.

CONCLUSION

This study compared MobileNetV2 and EfficientNetB0 for BISINDO alphabet recognition under a shared baseline transfer learning protocol. MobileNetV2 achieved 95.8% accuracy with strong precision, recall, and F1-score values on the compiled image-level test set. EfficientNetB0 achieved only 3.85% accuracy, approximately chance level for 26 classes. The EfficientNetB0 run must therefore be considered unsuccessful or insufficiently verified, rather than evidence of satisfactory convergence or generalization. These findings support MobileNetV2's effectiveness only within the present dataset and configuration; they do not establish the general superiority of MobileNetV2 or the unsuitability of EfficientNetB0.

Class-level performance for MobileNetV2 showed that errors were concentrated among visually similar signs that differed subtly in finger configuration or hand orientation. Interpretation is further limited by the absence of signer-independent splitting, external testing, deployment benchmarks, and a documented class-level validation of horizontal flipping. Consequently, MobileNetV2 should be described only as the better-performing model in this experiment, not as a robust or deployment-ready system. Future work should re-run EfficientNetB0 with architecture-specific optimization and verified checkpoints, validate augmentation decisions with fluent BISINDO users on a class-by-class basis, evaluate both models on signer-independent and external data, and measure model size, latency, memory use, and mobile or CPU inference performance before practical implementation is claimed.

REFERENCE

Abdullahi, S. B., & Chamnongthai, K. (2022). American Sign Language Words Recognition of Skeletal Videos Using Processed Video Driven Multi-Stacked Deep LSTM. *Sensors*, *22*(4), 1406. https://doi.org/10.3390/s22041406

Agustiani, S., Haryani, H., Junaidi, A., Putri, R. R., & Adam Z, M. G. (2025). Comparative Optimization of EfficientNetB3, MobileNetV2, and ResNet50 for Waste Classification. *Jurnal Informatika*, *12*(2), 131–137. https://doi.org/10.31294/inf.v12i2.27533

Agustiansyah, Y., & Kurniadi, D. (2025). Indonesian Sign Language Alphabet Image Classification using Vision Transformer. *Journal of Intelligent Systems Technology and Informatics*, *1*(1), 1–9. https://doi.org/10.64878/jistics.v1i1.5

Alaftekin, M., Pacal, I., & Cicek, K. (2024). Real-time sign language recognition based on YOLO algorithm. *Neural Computing and Applications*, *36*(14), 7609–7624. https://doi.org/10.1007/s00521-024-09503-6

Balat, M., Awaad, R., Adel, H., Zaky, A. B., & Aly, S. A. (2024). *Advanced Arabic Alphabet Sign Language Recognition Using Transfer Learning and Transformer Models*. https://doi.org/10.48550/ARXIV.2410.00681

Baumgartl, H., & Buettner, R. (2021). *Developing efficient transfer learning strategies for robust scene recognition in mobile robotics using pre-trained convolutional neural networks* (Version 1). arXiv. https://doi.org/10.48550/ARXIV.2107.11187

Briones Cerquín, A. D., Tumay Guevara, J. A., & Ovalle, C. (2025). Mobile Application for Continuous Recognition and Classification of Sign Language Images through Deep Learning. *International Journal of Interactive Mobile Technologies (iJIM)*, *19*(07), 4–21. https://doi.org/10.3991/ijim.v19i07.52853

Daniels, S., Suciati, N., & Fathichah, C. (2021). Indonesian Sign Language Recognition using YOLO Method. *IOP Conference Series: Materials Science and Engineering*, *1077*(1), 012029. https://doi.org/10.1088/1757-899X/1077/1/012029

Das, S., Biswas, S. Kr., & Purkayastha, B. (2023). A deep sign language recognition system for Indian sign language. *Neural Computing and Applications*, *35*(2), 1469–1481. https://doi.org/10.1007/s00521-022-07840-y

Dwijayanti, S., -, H., Taqiyyah, S. I., Hikmarika, H., & Suprapto, B. Y. (2021). Indonesia Sign Language Recognition using Convolutional Neural Network. *International Journal of Advanced Computer Science and Applications*, *12*(10). https://doi.org/10.14569/IJACSA.2021.0121046

Fauzi, D. R., & Haqdu D, G. A. (2025b). Comparison of CNN Models Using EfficientNetB0, MobileNetV2, and ResNet50 for Traffic Density with Transfer Learning. *Journal of Intelligent Systems Technology and Informatics*, *1*(1), 22–30. https://doi.org/10.64878/jistics.v1i1.6

Hugar, G., & Kagalkar, R. M. (2025). Hybrid Dual-Stream Deep Learning Approach for Real-Time Kannada Sign Language Recognition in Assistive Healthcare. *Journal of Information Systems Engineering and Business Intelligence*, *11*(3), 393–406. https://doi.org/10.20473/jisebi.11.3.393-406

Iqbal, S., N. Qureshi, A., Li, J., & Mahmood, T. (2023). On the Analyses of Medical Images Using Traditional Machine Learning Techniques and Convolutional Neural Networks. *Archives of Computational Methods in Engineering*, *30*(5), 3173–3233. https://doi.org/10.1007/s11831-023-09899-9

Joan, D., Vincent, V., Daniel, K. J., Achmad, S., & Sutoyo, R. (2023). BISINDO Hand-Sign Detection Using Transfer Learning. *2023 IEEE 8th International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*, 1–7. https://doi.org/10.1109/ICRAIE59459.2023.10468194

Kelana, E. L., Anshori Prasetya, M. R., . M., & Zulfadhilah, M. (2025). Integrating the CNN Model with the Web for Indonesian Sign Language (BISINDO) Recognition. *Journal of Applied Informatics and Computing*, *9*(3), 883–896. https://doi.org/10.30871/jaic.v9i3.9345

Kindiroglu, A. A., Kara, O., Ozdemir, O., & Akarun, L. (2024). *Transfer Learning for Cross-dataset Isolated Sign Language Recognition in Under-Resourced Datasets* (Version 2). arXiv. https://doi.org/10.48550/ARXIV.2403.14534

Kingma, D. P., & Ba, J. (2014). *Adam: A Method for Stochastic Optimization* (Version 9). arXiv. https://doi.org/10.48550/ARXIV.1412.6980

Lamaakal, I., Yahyati, C., Maleh, Y., Makkaoui, K. E., & Ouahbi, I. (2026). An explainable hybrid CNN–transformer model for sign language recognition on edge devices using adaptive

fusion and knowledge distillation. *Scientific Reports*, *16*(1), 7143. https://doi.org/10.1038/s41598-026-38478-8

Miah, A. S. M., Hasan, Md. A. M., Tomioka, Y., & Shin, J. (2024). Hand Gesture Recognition for Multi-Culture Sign Language Using Graph and General Deep Learning Network. *IEEE Open Journal of the Computer Society*, *5*, 144–155. https://doi.org/10.1109/OJCS.2024.3370971

Papatsimouli, M., Sarigiannidis, P., & Fragulis, G. F. (2023). A Survey of Advancements in Real-Time Sign Language Translators: Integration with IoT Technology. *Technologies*, *11*(4), 83. https://doi.org/10.3390/technologies11040083

Podder, K. K., Ezeddin, M., Chowdhury, M. E. H., Sumon, Md. S. I., Tahir, A. M., Ayari, M. A., Dutta, P., Khandakar, A., Mahbub, Z. B., & Kadir, M. A. (2023). Signer-Independent Arabic Sign Language Recognition System Using Deep Learning Model. *Sensors*, *23*(16), 7156. https://doi.org/10.3390/s23167156

Rachmawati, I. D. A., Yunanda, R., Hidayat, M. F., & Wicaksono, P. (2023). Deep Transfer Learning for Sign Language Image Classification: A Bisindo Dataset Study. *Engineering, MAthematics and Computer Science Journal (EMACS)*, *5*(3), 175–180. https://doi.org/10.21512/emacsjournal.v5i3.10621

Ridwan, A. E. M., Chowdhury, M. I., Mary, M. M., & Abir, M. T. C. (2024b). *Deep Neural Network-Based Sign Language Recognition: A Comprehensive Approach Using Transfer Learning with Explainability* (arXiv:2409.07426). arXiv. https://doi.org/10.48550/arXiv.2409.07426

Rizka Fadhillah, I., Muharrom Al Haromainy, M., & Maulana, H. (2024a). Implementasi Model Transfer Learning Efficientnet Untuk Pendeteksian Bahasa Isyarat Indonesia (Bisindo) Pada Perangkat Android. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *8*(4), 7816–7822. https://doi.org/10.36040/jati.v8i4.10463

Sadik, R., Majumder, A., Biswas, A. A., Ahammad, B., & Rahman, Md. M. (2023). An in-depth analysis of Convolutional Neural Network architectures with transfer learning for skin disease diagnosis. *Healthcare Analytics*, *3*, 100143. https://doi.org/10.1016/j.health.2023.100143

Sutjiadi, R. (2023). Android-Based Application for Real-Time Indonesian Sign Language Recognition Using Convolutional Neural Network. *TEM Journal*, 1541–1549. https://doi.org/10.18421/TEM123-35

Tan, M., & Le, Q. V. (2021). EfficientNetV2: Smaller Models and Faster Training. *Proceedings of the 38th International Conference on Machine Learning, PMLR*.

TensorFlow Developers. (2024). *Transfer Learning and Fine-tuning Guide* [Computer software]. https://www.tensorflow.org/tutorials/images/transfer_learning

Töngi, R. (2021). *Application of Transfer Learning to Sign Language Recognition using an Inflated 3D Deep Convolutional Neural Network* (Version 1). arXiv. https://doi.org/10.48550/ARXIV.2103.05111

Wahyudi, I. P. A. T., Sudipa, I. G. I., Libraeni, L. G. B., Radhitya, M. L., & Asana, I. M. D. P. (2025a). Performance Comparison of MobileNet and EfficientNet Architectures in Automatic Classification of Bacterial Colonies. *Indonesian Journal of Data and Science*, *6*(2), 333–342. https://doi.org/10.56705/ijodas.v6i2.218

Xia, K., Lu, W., Fan, H., & Zhao, Q. (2022). A Sign Language Recognition System Applied to Deaf-Mute Medical Consultation. *Sensors*, *22*(23), 9107. https://doi.org/10.3390/s22239107

Zaelani, F., & Miftahuddin, Y. (2022). Perbandingan Metode EfficientNetB3 dan MobileNetV2 Untuk Identifikasi Jenis Buah-buahan Menggunakan Fitur Daun: Metode EfficientNetB3 dan MobileNetv2. *Jurnal Ilmiah Teknologi Infomasi Terapan*, *9*(1). https://doi.org/10.33197/jitter.vol9.iss1.2022.911

Zufria, I., Harumy, T. H. F., & Efendi, S. (2025). Identifying the best models for BISINDO alphabet gesture classification to support the communication needs of the deaf community. *Eastern-European Journal of Enterprise Technologies*, *6*(2 (138)), 26–41. https://doi.org/10.15587/1729-4061.2025.332096