

A Two-Stage Lightweight Framework for Real Time Static Hand Gesture Recognition for Smart-Home Control

Mingyang HUANG

huang@ustb.edu.cn

University of Science and Technology Beijing

Research Article

Keywords: static gesture recognition, smart-home control, gesture segmentation, U-Net, EdgeHandCNN

Posted Date: October 19th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10917994/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

A Two-Stage Lightweight Framework for Real Time Static Hand Gesture Recognition for Smart-Home Control

Authors: Mingyang HUANG^{a, b, c} *

*Corresponding author (Mingyang HUANG, Email: huang@ustb.edu.cn)

Affiliations:

a Key Laboratory of Intelligent Bionic Unmanned Systems, Ministry of Education,
University of Science and Technology Beijing, Beijing 100083, P. R. China;

b School of Artificial Intelligence, University of Science and Technology Beijing,
Beijing 100083, P. R. China;

c Institute of Artificial Intelligence, University of Science and Technology Beijing,
Beijing 100083, P. R. China

†These authors contributed equally to this work

Abstract

Robust static hand gesture recognition for smart homes is challenged by complex backgrounds and strict latency constraints. We propose a real-time, lightweight two-stage framework explicitly decoupling segmentation from classification. Incorporating a custom 16-class dataset with a dedicated non-gesture category effectively mitigates false triggers. For segmentation, an improved lightweight U-Net achieves 0.9793 mIoU internally and 0.8950 on the external EgoHands dataset within 7.34 ms. For classification, we introduce EdgeHandCNN, a lightweight feature aggregation paradigm optimized for edge deployment. Tailored for segmented foregrounds, it deeply extracts high-frequency morphological cues. Trained via a rigorous five-fold Out-of-Fold (OOF) strategy to prevent data leakage, EdgeHandCNN delivers 0.9955 accuracy at 1.88 ms inference speed. End-to-end prototype validation confirms its capability for accurate, low-latency contactless interaction.

Keywords: static gesture recognition; smart-home control; gesture segmentation; U-Net; EdgeHandCNN

1 Introduction

Static gesture recognition provides a contactless human-computer interaction modality for smart-home control. Compared to touch interfaces, remote controllers, and wearable devices, camera-based interaction minimizes physical contact and facilitates more intuitive operations in daily environments. However, its practical deployment is often hindered by background clutter, illumination variations, skin-color distractors, partial occlusions, and the strict latency constraints imposed by interactive control systems.

Existing gesture-recognition approaches can be broadly divided into device-based, conventional vision-based, and deep-learning-based methods. Computer vision-based hand gesture recognition has been reviewed as an important human-robot interaction modality, in which acquisition quality, foreground extraction, feature representation, and classification jointly affect recognition robustness^[1]. IMU-based data gloves can capture three-dimensional hand gestures with relatively stable motion measurements^[2]. Surface-electromyography-based (sEMG) gesture recognition provides another sensing route by recording muscle activity instead of visual appearance^[3]. Deep-learning methods have also been applied to sEMG-based prosthetic-hand gesture recognition, but such systems still depend on dedicated sensors and user-specific signal conditions^[4]. Conventional vision-based approaches usually rely on skin-color segmentation, contour analysis, handcrafted descriptors, and shallow classifiers^[5]. Structured reviews of vision-based hand gesture recognition indicate that illumination changes, background clutter, hand scale, and viewpoint variation remain central deployment challenges^[6].

Recent advances in deep learning have enabled gesture representations to be learned directly from image data. Lightweight hand-segmentation research shows that real-time hand-region extraction is feasible when compact models and landmark cues are combined^[7]. Real-time gesture monitoring systems further indicate that recognition pipelines need integrated detection, tracking, and classification under deployment constraints^[8]. Transformer-based static gesture recognition studies show that stronger global representation learning can improve class discrimination^[9]. MediaPipe Hands demonstrates that on-device hand tracking can operate in real time by combining palm detection with hand landmark estimation^[10]. Nevertheless, many existing systems are evaluated under relatively clean visual conditions, omit an explicit foreground-

suppression stage, or rely on models whose computational footprint is unfavorable for real-time desktop or edge deployment. For smart-home control, an effective pipeline must suppress background interference before classification while maintaining sufficiently low end-to-end latency for interactive use.

To address these issues, this study develops a static-gesture recognition pipeline oriented to smart-home control. A self-collected RGB dataset comprising 16 classes, including a no-gesture category, is established under a fixed-ROI acquisition protocol to maintain annotation consistency and reduce false triggering. For foreground extraction, a lightweight U-Net variant is designed through channel reduction, Batch Normalization, bias-free convolution, a BCE-Dice hybrid loss, and AdamW-based optimization. The segmented hand foreground is then classified by EdgeHandCNN.

The main contributions of this paper are summarized as follows.

(1) A paired RGB-mask dataset comprising 16 static gesture classes is constructed via a SAM-assisted annotation pipeline. A rigorous five-fold OOF protocol is formulated to connect the segmentation and classification stages while effectively mitigating data leakage.

(2) An improved lightweight U-Net is proposed for hand foreground extraction, alongside an interpretable skin-color-prior baseline. By reducing channel widths and introducing a hybrid BCE-Dice loss, the proposed segmentation model achieves robust foreground estimation with substantially lower computational overhead.

(3) A lightweight feature aggregation paradigm optimized for smart-home edge deployment is proposed. To process the segmented foregrounds efficiently, a dedicated architecture (EdgeHandCNN) is constructed. By coupling depthwise separable convolutions with Squeeze-and-Excitation (SE) and spatial attention mechanisms, this paradigm deeply optimizes the extraction of high-frequency morphological representations—such as finger extensions and palm contours—achieving an optimal trade-off between discriminative capacity and computational overhead.

(4) An end-to-end system prototype is developed and evaluated, validating the real-time command mapping, non-gesture rejection capabilities, and deployment feasibility of the proposed framework under smart-home interaction constraints.

The remainder of this paper is organized as follows. Section 2 details the dataset construction and hand foreground segmentation methodology. Section 3 presents classification experiments and system-level validation. Finally, Section 4 concludes the paper and outlines future research directions.

2 Dataset Construction and Hand-Foreground Segmentation

This section describes the data acquisition, annotation, augmentation, and segmentation pipelines utilized to generate foreground-normalized inputs for downstream classification.

2.1 Data Acquisition and Gesture Taxonomy

Image data were acquired using an RGB camera employing a fixed-ROI acquisition program. The program captured frames within a predefined ROI and supported mirrored display, class switching, and timed continuous capture functionalities. During acquisition, the hand was constrained within the ROI to ensure consistent posture sampling. This protocol mitigates annotation ambiguity, limits irrelevant background variations, and aligns the dataset with the intended recognition workflow.

According to the discrete command requirements of smart-home control and the available data distribution, 16 static gesture classes were collected, including digit gestures 1-9, palm up, palm down, fist, grasp, thumbs up, OK, and no-gesture. The class set was designed to cover representative home-control commands while retaining visually distinguishable hand configurations for classification. The no-gesture class was defined as an invalid input, whereas the remaining 15 classes were regarded as executable commands. The resulting original dataset, denoted as Dataset A, contains 500 images per class and 8,000 images in total. Representative examples of the 15 valid gestures are shown in Fig. 1.

The no-gesture class is retained as an explicit rejection state to improve segmentation and classification robustness. If the model is trained only on valid gestures, background frames and transient hand-entry states may be incorrectly interpreted as executable commands. By explicitly introducing a no-gesture category, the model can learn non-command samples without relying solely on hand-crafted rejection rules. To reflect application requirements, the gestures are further mapped to different home-control modes, as summarized in Table 1

In addition to the self-collected dataset, the public EgoHands dataset is used as Dataset D. EgoHands was introduced for detecting hands and recognizing activities in complex egocentric interactions^[11]. It contains 48 first-person interaction videos, 4,800 pixel-level annotated frames, and more than 15,000 hand instances. Compared with Dataset A, it exhibits larger viewpoint variation, frequent occlusion, and more complex backgrounds, making it suitable for cross-dataset segmentation evaluation. Partial sample images of Dataset D are illustrated in Fig. 2.

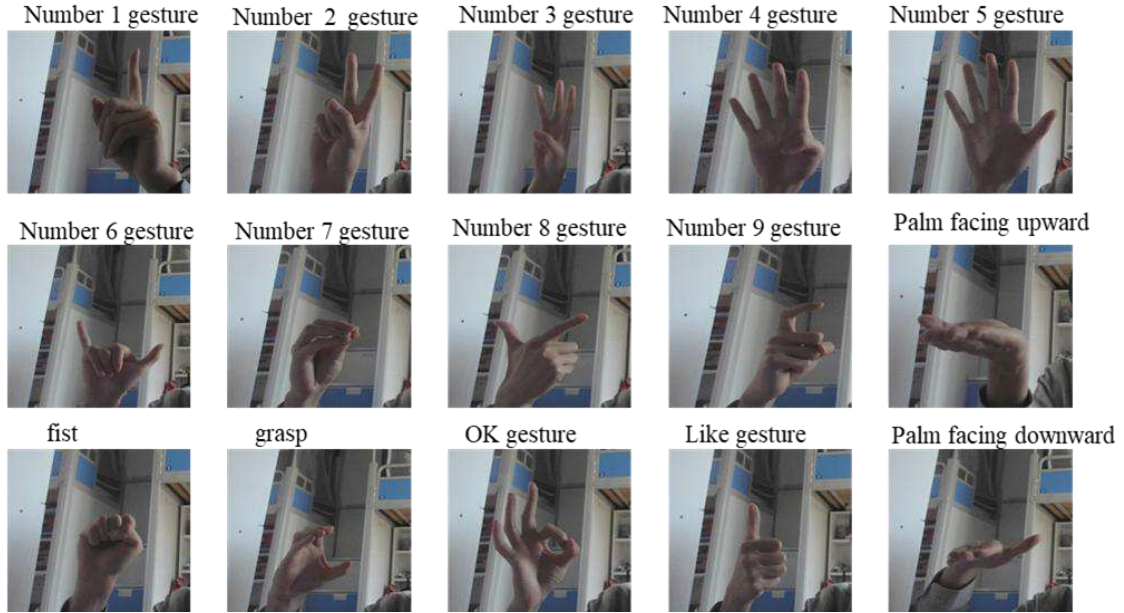


Fig. 1. Examples of the 15 valid gestures in Dataset A

Table 1. Mapping between 16 gestures and smart-home control modes

No.	Gesture name	Control mode
1	Thumbs up	Increase light brightness
2	Fist	Turn off whole-house devices
3	OK gesture	Confirm command / enable comfort mode
4	Palm up	Turn on lights
5	Palm down	Turn off lights
6	Grasp	Start robot vacuum cleaner
7	Digit 1 gesture	Control air conditioner switch
8	Digit 2 gesture	Control curtain opening and closing
9	Digit 3 gesture	Control smart-fan switch
10	Digit 4 gesture	Turn on television
11	Digit 5 gesture	Enable home-welcome mode
12	Digit 6 gesture	Control water heater
13	Digit 7 gesture	Enable movie-watching mode
14	Digit 8 gesture	Enable away-from-home mode
15	Digit 9 gesture	Broadcast whole-house device status
16	No-gesture	No action



Fig. 2. Partial sample images of Dataset D

2.2 Data Augmentation and Annotation

To reduce overfitting caused by consecutive-frame similarity and limited sample diversity, horizontal flipping, affine transformation, and color jitter were applied only to the training split of Dataset B. The same geometric transformation was applied to the paired mask, and nearest-neighbor interpolation was used for mask resizing. The original paired dataset contains 4,835 RGB-mask samples, with 3,862 samples in the training split. After augmentation, the training split increased to 7,784 samples, while the validation and test splits remained unchanged. Cutout is reserved for classification-stage augmentation to simulate localized foreground occlusion and missing hand details [12].

Binary mask annotation was performed with an assisted SAM pipeline based on the Segment Anything model [13]. Foreground prompts were generated from YCrCb and HSV skin-color priors, background prompts were sampled inside the ROI, and the prompts together with a candidate box were fed into SAM to obtain multiple candidate masks.

When multiple masks were returned, the final mask was selected by maximizing a weighted score combining prior-mask overlap and SAM confidence. Specifically, the candidate with the largest value of 0.6 IoU plus 0.4 confidence was retained. The selected mask was further refined by morphological opening and closing, largest-component retention, and manual inspection. The final paired RGB-mask dataset is denoted as Dataset B.

2.3 Baseline Skin-Color Segmentation

To establish an interpretable reference for the learning-based models, a deterministic skin-color-prior baseline was implemented. This baseline encompasses color correction,

color space conversion, candidate region screening, adaptive chrominance modeling, thresholding, and morphological post-processing.

As a basic benchmark, a fixed global threshold was first applied in the YCbCr color space. A pixel was classified as skin only when its chrominance values satisfied the following empirical criterion:

$$\begin{cases} 133 \leq C_r \leq 173 \\ 77 \leq C_b \leq 127 \end{cases} \quad (1)$$

The second deterministic baseline applies Gray-World white balance to reduce global color cast ^[14]. The corrected image is then converted from RGB to YCbCr according to the following formulation:

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \frac{1}{256} \mathbf{A} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (2)$$

$$\mathbf{A} = \begin{bmatrix} 64.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112.000 \\ 112.000 & -93.786 & -18.214 \end{bmatrix} \quad (3)$$

Candidate pixels are generated by combining YCbCr and HSV constraints. HSV constraints are included because the HSV color representation separates hue and saturation from brightness-related variation ^[15]. The strict candidate region provides high-confidence samples for adaptive model estimation, whereas the loose region preserves uncertain boundary pixels that may be recovered through probabilistic modeling and morphological post-processing. The corresponding threshold settings are summarized in Table 2.

Table 2. Threshold settings for candidate regions in the skin-color-prior baseline

Region	Cr	Cb	S	V
Strict	$135 \leq C_r \leq 178$	$80 \leq C_b \leq 130$	$30 \leq S \leq 120$	$V \geq 25$
Loose	$125 \leq C_r \leq 185$	$70 \leq C_b \leq 150$	$S \geq 18$	$V \geq 25$

The two candidate masks were obtained by intersecting the corresponding YCbCr and HSV decisions:

$$\begin{cases} M_s = M_s^{YCbCr} \cap M_s^{HSV} \\ M_c = M_c^{YCbCr} \cap M_c^{HSV} \end{cases} \quad (4)$$

For each image, the strict candidate pixels were used to estimate an adaptive Cr-Cb distribution. The chrominance vector of the i -th candidate pixel is defined as follows:

$$z_i = \begin{bmatrix} C_{r,i} \\ C_{b,i} \end{bmatrix} \quad (5)$$

The mean vector and covariance matrix were estimated from N strict candidate pixels as follows:

$$\mu = \frac{1}{N} \sum_{i=1}^N z_i \quad (6)$$

$$\Sigma = \frac{1}{N-1} \sum_{i=1}^N (z_i - \mu)(z_i - \mu)^T \quad (7)$$

A Gaussian skin-likelihood map was then computed within the loose candidate region:

$$P(x, y) = \exp \left[-\frac{1}{2} (z - \mu)^T \Sigma^{-1} (z - \mu) \right] \quad (8)$$

Because HSV is less sensitive to brightness variation than raw RGB, the strict HSV decision was used as a soft reliability term rather than as a direct replacement for the Gaussian likelihood map:

$$M_s^{HSV}(x, y) = \begin{cases} 1, & (x, y) \in M_s^{HSV} \\ 0, & (x, y) \notin M_s^{HSV} \end{cases} \quad (9)$$

$$P'(x, y) = P(x, y) (0.78 + 0.22 M_s^{HSV}(x, y)) \quad (10)$$

After normalization to an 8-bit probability map, Otsu thresholding is applied with a lower bound to avoid an unrealistically small threshold under dark or low-contrast conditions ^[16]:

$$T = \max(20, 0.9 T_{otsu}) \quad (11)$$

The initial binary mask is first refined by median smoothing. Morphological opening and closing are used to suppress small isolated regions and fill local holes, following classical morphological image analysis ^[17]. Largest-connected-component filtering is then applied to retain the dominant hand region and remove disconnected noise components ^[18]. These post-processing operations suppress isolated skin-like noise and fill small holes in the hand region. However, they cannot fully eliminate ambiguity caused by facial regions, wood-textured surfaces, or other backgrounds with chrominance values similar to those of skin. The segmentation process for representative samples under the skin-color-prior model is illustrated in Fig. 3.

Compared with conventional global fixed-threshold segmentation algorithms, this method achieves stronger stability and effectively alleviates background interference. Nevertheless, it still belongs to traditional prior-driven segmentation schemes. False

positives and false negatives frequently occur under scenarios involving large-scale skin-like background areas, intense illumination, shadows or hand occlusion, as demonstrated in Fig. 4.



Fig. 3. Visualization workflow of gesture segmentation

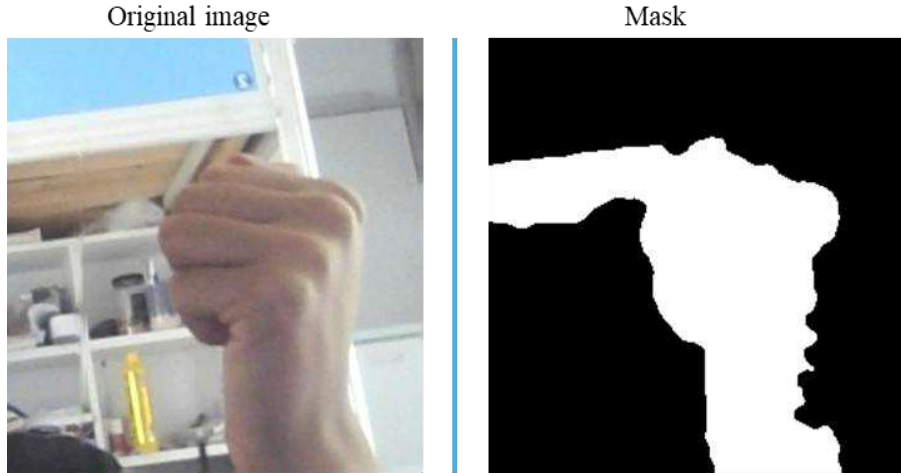


Fig. 4. False-detection and missed-detection cases

To establish a comparison with conventional global fixed-threshold methods, batch segmentation experiments were conducted on manually annotated RGB-mask pairs. These experiments were performed on a Windows 11 platform equipped with an Intel Core i5-11400H CPU and an NVIDIA RTX 3050 Ti GPU (4 GB VRAM), hereafter referred to as Hardware Environment A. Specifically, 900 sample pairs were uniformly extracted from Dataset B (60 images for each of the 15 valid gesture classes), along with 900 random samples from the EgoHands dataset, totaling 1,800 pairs for

evaluation. The quantitative results obtained in Hardware Environment A across both datasets are summarized in Table 3.

Table 3. Performance of traditional skin-color segmentation methods on Datasets B and D

Method	Images	IoU	Dice	Average speed
Global fixed threshold (D)	900	0.207	0.319	0.458 ms/image
Global fixed threshold (B)	900	0.661	0.794	0.421 ms/image
Skin-color-prior (D)	900	0.255	0.377	13.283 ms/image
Skin-color-prior (B)	900	0.738	0.846	13.130 ms/image

Table 3 shows that the skin-color-prior baseline improves IoU and Dice compared with the fixed-threshold method on both Dataset B and Dataset D. Nevertheless, its performance on Dataset D remains substantially lower, indicating that color priors are vulnerable to cross-domain changes such as first-person viewpoint variation, occlusion, and cluttered backgrounds in EgoHands. Although the fixed-threshold method is computationally faster, the adaptive prior provides more reliable foreground extraction at the cost of higher per-image latency. These limitations motivate the transition from deterministic color rules to learning-based U-Net segmentation.

2.4 U-Net-based Gesture Image Segmentation

2.4.1 U-Net Network Architecture and Loss Function

U-Net, originally introduced by Ronneberger et al. for biomedical image segmentation, is an encoder-decoder network characterized by a U-shaped topology^[19]. Its central mechanism is the skip connection, which fuses shallow spatial details with deep semantic representations. The basic architecture is shown in Fig. 5.

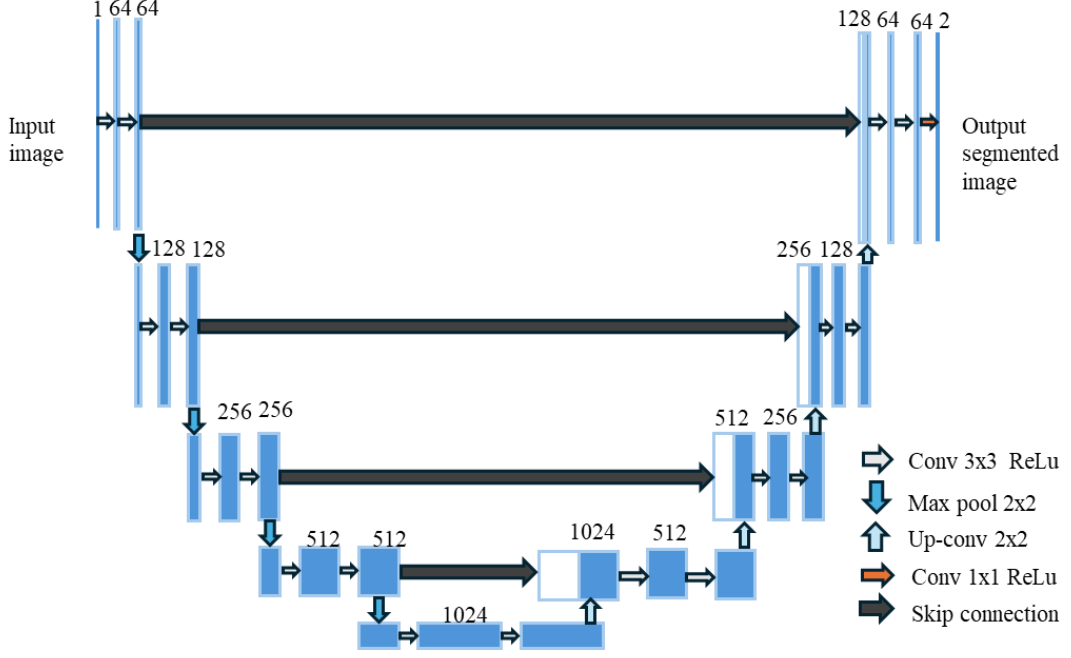


Fig. 5. Basic structure of U-Net

In the contraction path of U-Net, successive 3×3 convolutions and max-pooling operations reduce the spatial resolution while increasing the number of feature channels. In the expansion path, transposed convolution and standard convolution are used to recover the spatial resolution. At each decoder level, the upsampled feature map is concatenated with the encoder feature map at the corresponding scale, followed by two consecutive 3×3 convolutions. This architecture enlarges the receptive field while preserving boundary and structural information required for hand-mask reconstruction.

The conventional U-Net adopts a four-level encoder-decoder configuration with channel widths of 64, 128, 256, and 512 and a bottleneck width of 1024. Each convolutional block contains two 3×3 convolutions with ReLU activation, and the upsampling stage uses 2×2 transposed convolution. Under a three-channel RGB input and a binary-mask output, this configuration contains approximately 31.03 million parameters according to standard convolutional parameter counting.

Although the conventional U-Net provides strong representational capacity through its high-channel design, it also imposes considerable computational and storage overhead. In smart-home gesture control, hand segmentation serves only as the upstream stage of the recognition pipeline, so low latency and lightweight deployment are required in addition to stable foreground extraction. To alleviate the computational bottleneck on edge-oriented platforms, the proposed model preserves the encoder-decoder topology and skip connections of U-Net while scaling the channel widths down to 32, 64, 128, and 256. Batch Normalization and bias-free convolutions are further

integrated into each convolutional block to stabilize the feature distribution and improve optimization efficiency. Consequently, the parameter count decreases from approximately 31.03 million to approximately 7.76 million while maintaining multiscale semantic fusion.

For loss design, the baseline model uses binary cross-entropy (BCE), whereas the proposed model adopts a weighted combination of BCE and Dice loss. The BCE term constrains pixel-level foreground probabilities, while the Dice term explicitly optimizes global region overlap, which is especially beneficial for hand-mask learning under foreground-background imbalance. The BCE term is defined as:

$$L_{BCE} = -\frac{1}{N} \sum_{i=1}^N [y_i \log p_i + (1 - y_i) \log (1 - p_i)] \quad (12)$$

The Dice loss is defined as:

$$L_{Dice} = 1 - \frac{2 \sum_{i=1}^N p_i y_i + \varepsilon}{\sum_{i=1}^N p_i + \sum_{i=1}^N y_i + \varepsilon} \quad (13)$$

Accordingly, the total loss is formulated as:

$$L_{seg} = \lambda_1 L_{BCE} + \lambda_2 L_{Dice} \quad (14)$$

In the BCE term, N denotes the total number of pixels involved in loss computation, y_i denotes the ground-truth label of the i -th pixel, and p_i denotes the foreground probability after Sigmoid activation. In this study, hand-foreground pixels are labeled as 1, whereas background pixels are labeled as 0.

In the Dice term, ε is introduced to avoid division by zero. Dice loss complements BCE by emphasizing structural consistency of the predicted hand region, whereas BCE preserves stable pixel-level supervision from the perspective of binary classification.

For optimization, the original U-Net is commonly trained with Adam and a fixed learning rate. Although this setting is simple, it provides limited flexibility for balancing rapid early convergence against fine-grained late-stage refinement. Therefore, the improved U-Net is trained using AdamW, cosine annealing with warm restarts, and gradient clipping.

AdamW decouples weight decay from adaptive gradient updates, thereby improving regularization stability and generalization behavior^[20]. Cosine annealing with warm restarts periodically lowers and restores the learning rate, enabling the optimizer to escape shallow local optima while retaining fine-grained parameter refinement^[21]. Gradient clipping is further used to suppress excessively large parameter updates and improve training stability. In this study, AdamW uses $\beta_1 = 0.9$ and $\beta_2 = 0.999$, the initial

cosine period is 10, the period growth factor is 2, and the gradient-clipping threshold is 1.0.

2.4.2 Experimental Setup and Segmentation Result Analysis

To quantify segmentation performance, the proposed lightweight U-Net was benchmarked against the original U-Net under controlled experimental conditions. Training and validation were conducted on Ubuntu 22.04 with an Intel Xeon Platinum 8362 CPU and an NVIDIA RTX 3090 GPU with 24 GB memory, denoted as Environment B. Final testing was performed on Environment A, using the same software stack where applicable.

For Dataset B, only B_train was randomly augmented, and the augmented training subset contained 7,784 samples. B_val and B_test were kept fixed and received only deterministic resizing and normalization. During geometric augmentation, images and masks shared the same transformation parameters, and nearest-neighbor interpolation was used for masks. Dataset D was used in parallel experiments to compare the behavior of the two U-Net structures under an external first-person hand distribution.

Segmentation performance was measured using mean intersection over union (mIoU), Dice coefficient, and pixel accuracy (PA). Because mIoU and Dice directly quantify foreground overlap and boundary preservation, they were used as the primary metrics, whereas PA was used as an auxiliary indicator of pixel-level correctness. The mIoU and PA are defined as follows:

$$mIoU = \frac{1}{C} \sum_{c=1}^C IoU_c \quad (15)$$

$$PA = \frac{TP + TN}{TP + TN + FP + FN} \quad (16)$$

In the above equations, C denotes the total number of classes including the background, and IoU_c denotes the IoU of the c -th class. TP , TN , FP , and FN denote the numbers of correctly predicted foreground pixels, correctly predicted background pixels, incorrectly predicted foreground pixels, and missed foreground pixels, respectively. The validation curves of Dice, mIoU, and PA are used to compare the optimization behavior of the two U-Net variants.

The following subsections report the training curves and validation metrics on Datasets B and D.

(1) Dataset B

The training curves of the original U-Net model on Dataset B are presented in Fig. 6

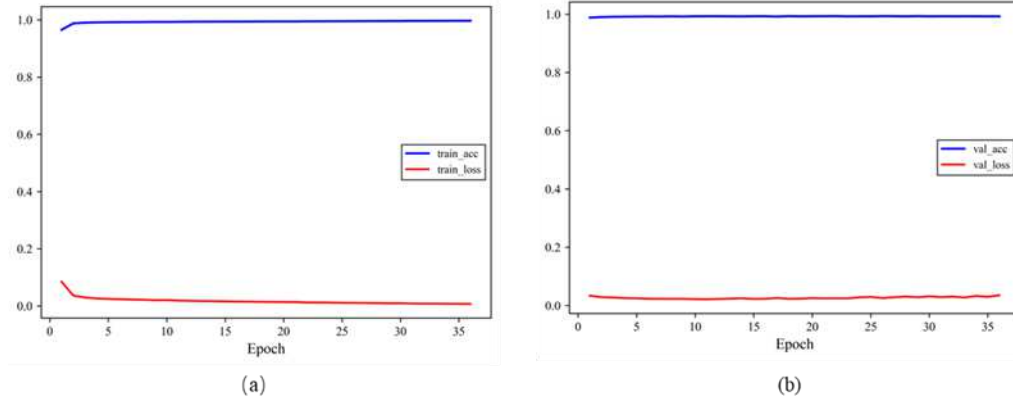


Fig. 6. Training curves of the original U-Net model on Dataset B

As illustrated in Fig. 6, the network converges steadily after roughly 10 training epochs, and the final validation accuracy achieves 99.29%. Moreover, the two loss curves indicate that the loss metrics continuously decline along with the iterative training process.

The validation mIoU and Dice curves of the original U-Net are displayed in Fig. 7

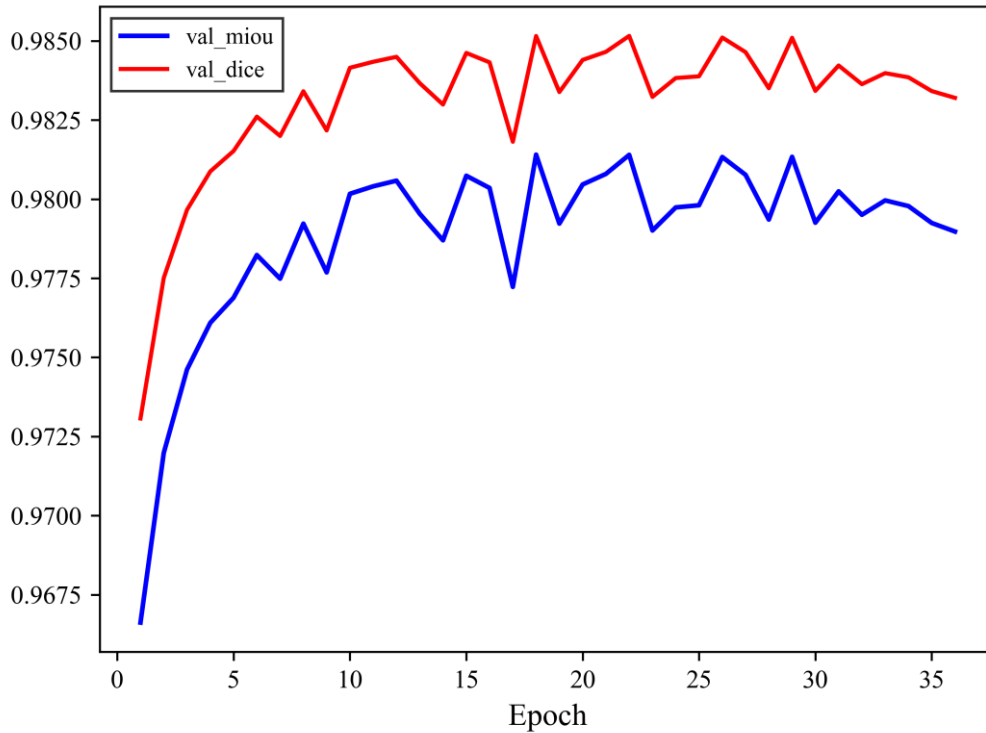


Fig. 7. Two evaluation metric curves of the original U-Net on the validation set of Dataset B

In Fig. 7, the mIoU and Dice metrics on the validation set rise rapidly in the early training stage and gradually converge after approximately 10 epochs, demonstrating

that the model accomplishes effective feature learning rapidly and enters a stable convergence state. Although slight fluctuations occur subsequently, the overall variation amplitude remains small without obvious degradation, which verifies the stable training process, favorable segmentation performance and strong generalization capability of the proposed model on the validation dataset.

The training curves of the improved U-Net model on Dataset B are shown in Fig. 8.

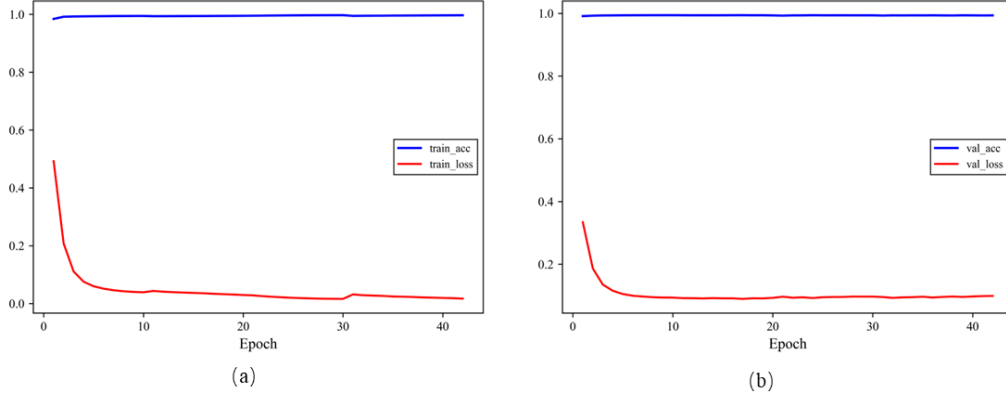


Fig. 8. Training curves of the improved U-Net model on Dataset B

As shown in Fig. 8, the network tends to stabilize gradually after about 14 training epochs, and the final validation accuracy reaches 99.35%. Nevertheless, its loss values in the early training stage are higher than those of the original U-Net. Since the two U-Net structures adopt different loss functions, their loss magnitudes are not comparable. It can be observed from the loss curves that the loss decreases steadily with increasing epochs, which demonstrates normal convergence of the model.

The mIoU and Dice curves of the improved U-Net on the validation set are presented in Fig. 9.

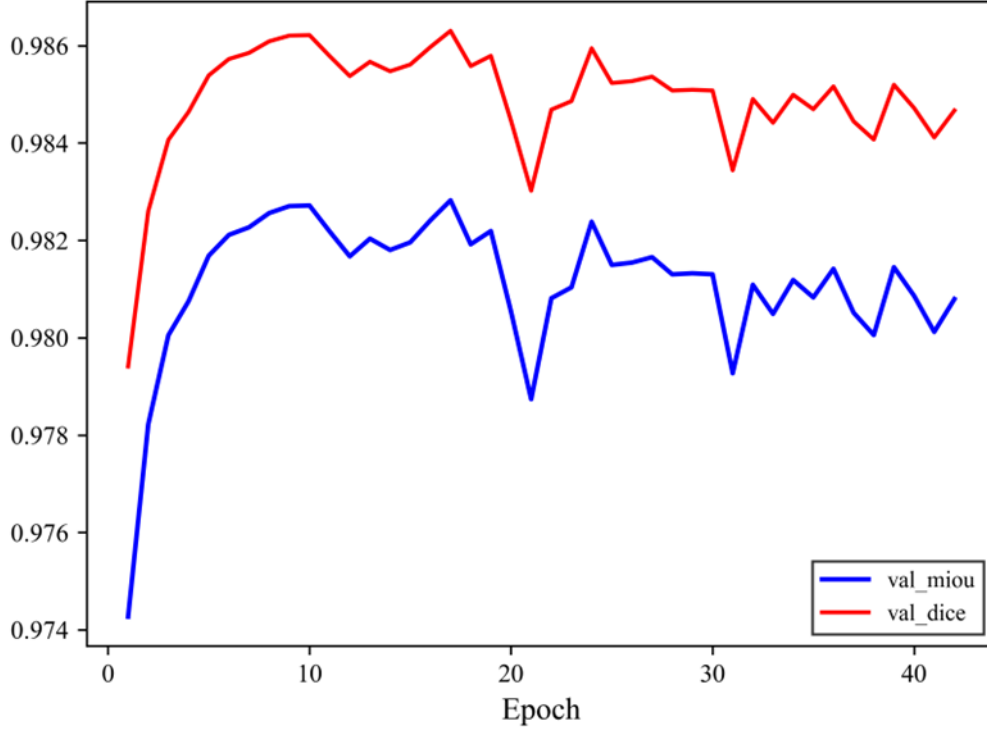


Fig. 9. Two evaluation metric curves of the improved U-Net on the validation set of Dataset B

Compared with Fig. 7, the mIoU and Dice metrics of the proposed model on the validation set also exhibit a rapid initial increase followed by gradual stabilization, indicating that the model rapidly learns hand region features and achieves convergence. In terms of the overall trend, both Dice and mIoU maintain higher values in the middle and late training stages. Specifically, the Dice score stabilizes at approximately 0.984–0.986, and the mIoU score ranges from 0.980 to 0.983, both slightly superior to the corresponding curves in Fig. 7. In summary, compared with the original network, the improved model achieves marginally higher segmentation accuracy and more stable convergent performance on the validation dataset.

(2) Dataset D

The training curves of the original U-Net model on Dataset D are illustrated in Fig. 10.

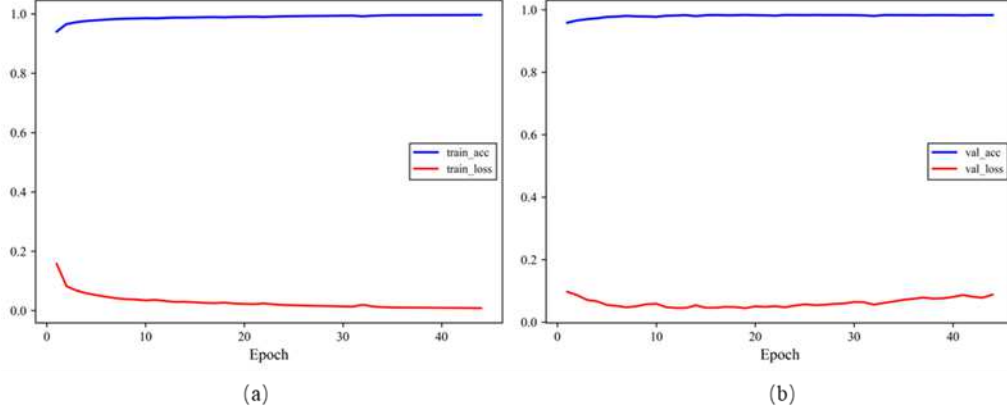


Fig. 10. Training curves of the original U-Net on Dataset D

As illustrated in Fig. 10, the network gradually converges after approximately 17 training epochs, and the final validation accuracy reaches 98.30%. Meanwhile, the loss curves indicate that the loss values continuously decrease as training proceeds.

The mIoU and Dice curves of the original U-Net on the validation set are presented in Fig. 11.

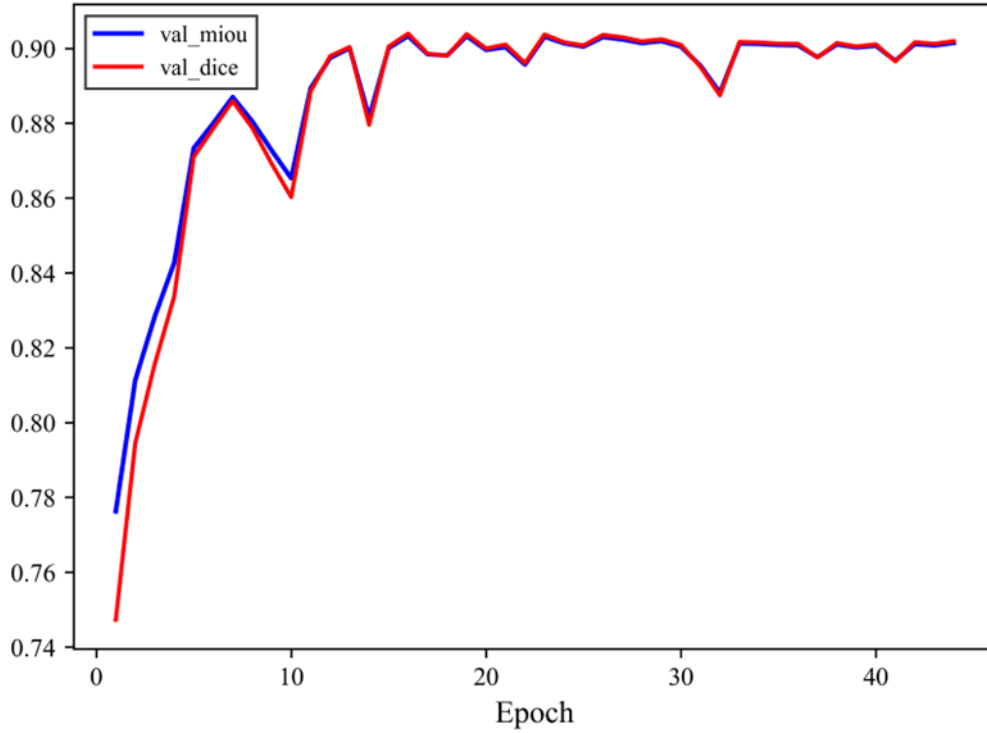


Fig. 11. Two evaluation metric curves of the original U-Net on the validation set of Dataset D

Compared with Fig. 7, the overall mIoU and Dice scores in Fig. 11 are lower, demonstrating that segmentation on Dataset D is more challenging. This difference mainly stems from distribution shifts, because Dataset D contains more complex backgrounds, stronger viewpoint changes, frequent occlusion, and more severe

illumination variation. Nevertheless, both metrics rise rapidly in the early stage and remain at a relatively high level, indicating that the original U-Net still converges effectively on the external dataset.

The training curves of the improved U-Net model on Dataset D are displayed in Fig. 12.

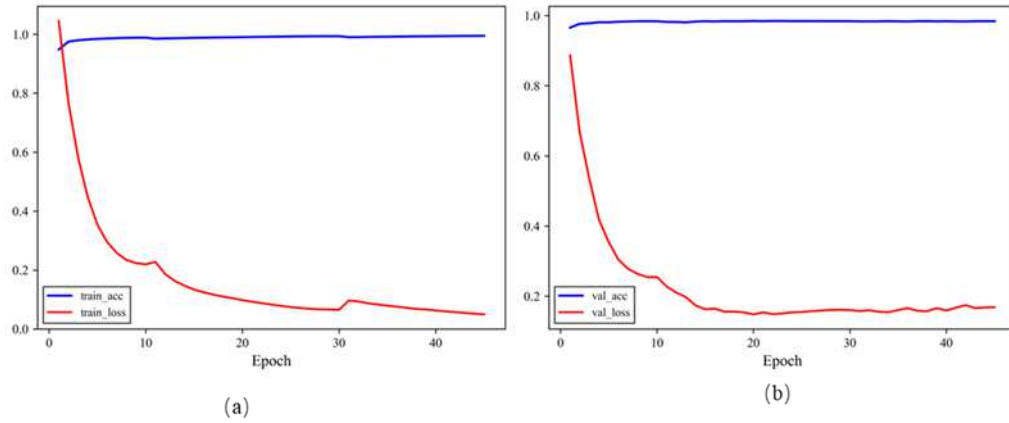


Fig. 12. Training curves of the improved U-Net model on Dataset D

As illustrated in Fig. 12, the network gradually converges after approximately 32 training epochs, and the final validation accuracy reaches 98.42%. The loss curves decrease continuously with training, verifying the normal convergence of the improved model.

The mIoU and Dice curves of the improved U-Net on the validation set are presented in Fig. 13

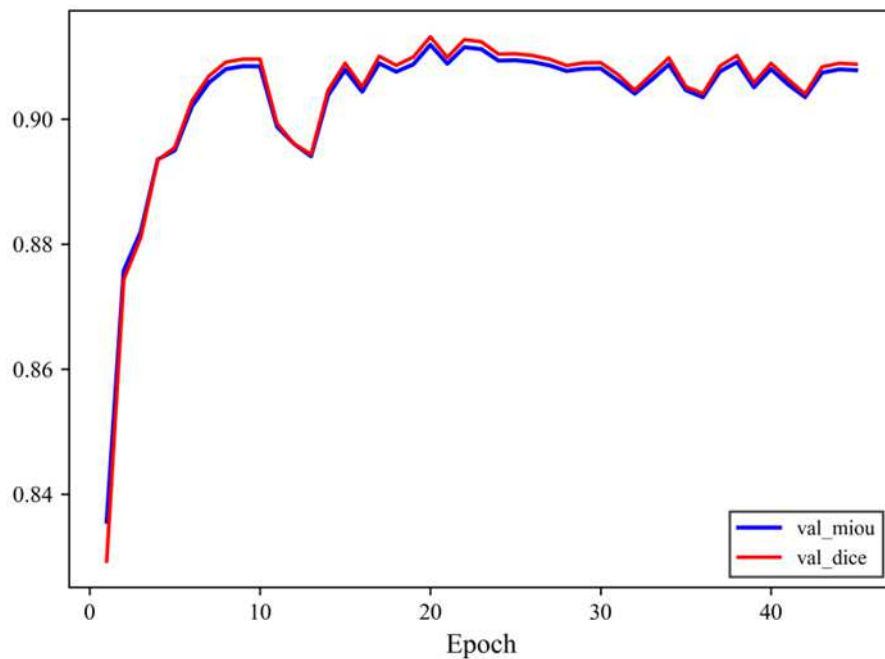


Fig. 13. Two evaluation metric curves of the improved U-Net on the validation set of Dataset D

Fig. 11 and Fig. 13 show that the mIoU and Dice scores of both the original U-Net and the improved U-Net rise rapidly at the initial training stage and then gradually converge. The improved U-Net maintains higher overall metric values and milder middle-to-late-stage fluctuations, indicating better segmentation precision and stability on the external dataset. Although the original U-Net also reaches a usable performance level, its larger fluctuations suggest weaker adaptability to distribution shifts.

(3) Performance on the two test datasets

Table 4. Test performance of original and improved U-Net models on Datasets B and D

Method	PA	mIoU	Dice	Average time(ms/image)	Parameters	Images	Dataset
Improved U-Net	0.9930	0.9793	0.9835	7.3445	7,763,041	501	B
Original U-Net	0.9923	0.9775	0.9850	21.1028	31,031,745	501	B
Improved U-Net	0.9820	0.8950	0.8947	7.4161	7,763,041	480	D
Original U-Net	0.9804	0.8867	0.8855	21.2710	31,031,745	480	D

As shown in Table 4, the improved U-Net achieves PA, mIoU, and Dice scores of 0.9930, 0.9793, and 0.9835 on Dataset B. On Dataset D, it achieves 0.9820 PA, 0.8950 mIoU, and 0.8947 Dice, which are higher than the corresponding scores of the original U-Net under the external distribution.

The improved model also reduces the parameter count from 31,031,745 to 7,763,041 and decreases the average inference time from 21.1028 to 7.3445 ms/image on Dataset B and from 21.2710 to 7.4161 ms/image on Dataset D. This indicates that channel-width reduction, Batch Normalization, bias-free convolution, and the BCE-Dice objective jointly improve parameter efficiency and deployment feasibility.

Considering segmentation accuracy, latency, and model size, the improved U-Net is selected as the segmentation model for the subsequent classification and prototype validation stages.

3 Gesture Classification and System-Level Verification

After foreground extraction, the classification module assigns each segmented hand image to a smart-home command category or to the no-gesture state. This section introduces the LeNet-5 baseline and the self-designed EdgeHandCNN classifier, describes the training protocol, and evaluates classification performance in terms of accuracy, macro-averaged metrics, confusion matrices, and inference latency.

3.1 LeNet-5 Baseline Classifier

LeNet-5 is a classical convolutional neural network proposed by LeCun et al. and initially developed for handwritten-digit classification [22]. It extracts local visual patterns through convolution, subsampling, and fully connected layers while controlling parameter growth through local connectivity and weight sharing. In this study, it is employed as a shallow convolutional baseline for performance comparison.

The canonical LeNet-5 architecture is shown in Fig. 14. It receives a 32×32 grayscale image, extracts edge and local-shape features through convolution and subsampling, and performs final discrimination using fully connected layers.

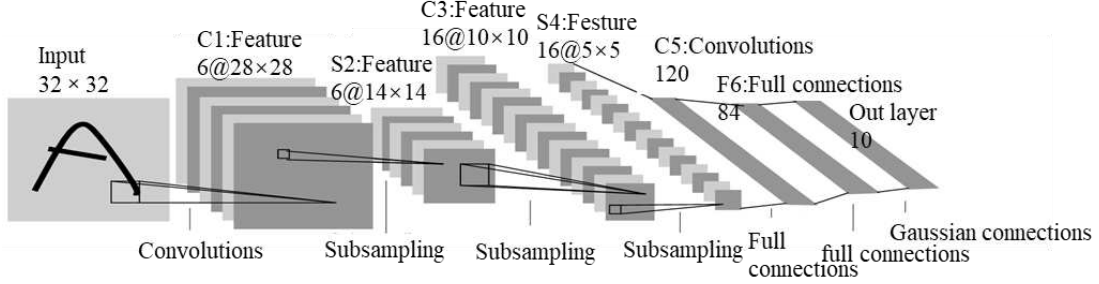


Fig. 14. Typical LeNet-5 network structure

To adapt LeNet-5 to the current 16-class RGB gesture task, three modifications were introduced. First, the input was expanded from a single-channel grayscale image to a three-channel RGB image, and all samples were resized to 224×224 . Second, to avoid a sharp increase in fully connected parameters caused by the larger input size, an adaptive average pooling layer was inserted after feature extraction to keep the subsequent fully connected structure manageable. Finally, the original output layer was replaced with a 16-class classifier corresponding to the gesture categories in this study. Dropout was added to the fully connected layers to reduce overfitting, and training was conducted using AdamW, cross-entropy loss, label smoothing, and cosine annealing learning-rate scheduling.

The cross-entropy loss is defined as:

$$L_{CE} = -\sum_{c=1}^C y_c \log p_c \quad (17)$$

In the cross-entropy formulation, y_c denotes the ground-truth label of class c , and p_c denotes the probability assigned to class c by the model. The Softmax probability is defined as:

$$p_c = \frac{e^{z_c}}{\sum_{k=1}^C e^{z_k}} \quad (18)$$

In the Softmax formulation, Z_c denotes the raw score assigned to class c . Conventional one-hot supervision assigns probability 1 to the true class and 0 to all other classes, which may produce overconfident decision boundaries on the training set. Label smoothing reduces overfitting to hard targets by assigning a small probability mass to non-ground-truth classes [23]. Because several static gestures in this dataset exhibit local similarity, this strategy is used to obtain a smoother output distribution and improve recognition stability for boundary-ambiguous samples. The label-smoothing coefficient is set to 0.15 in the classification experiments.

3.2 EdgeHandCNN Classifier

To overcome the representation bottlenecks of shallow networks while strictly adhering to the latency constraints of smart-home edge devices, this study introduces a lightweight morphological feature aggregation paradigm. The derived architecture, designated as EdgeHandCNN, deliberately abandons the redundant capacity of generic RGB-oriented backbones. Since the upstream segmentation stage has already filtered out background interference, the classification input is highly simplified. Consequently, this paradigm is explicitly optimized for segmented foregrounds, concentrating its computational budget on capturing high-frequency structural variations such as finger count, extension direction, and palm boundaries.

The overall architecture of EdgeHandCNN is shown in Fig. 15. The primary objective is to enhance the discrimination of fine-grained features, such as finger count, extension direction, and palm contours, while ensuring low parameter counts and short inference latencies.

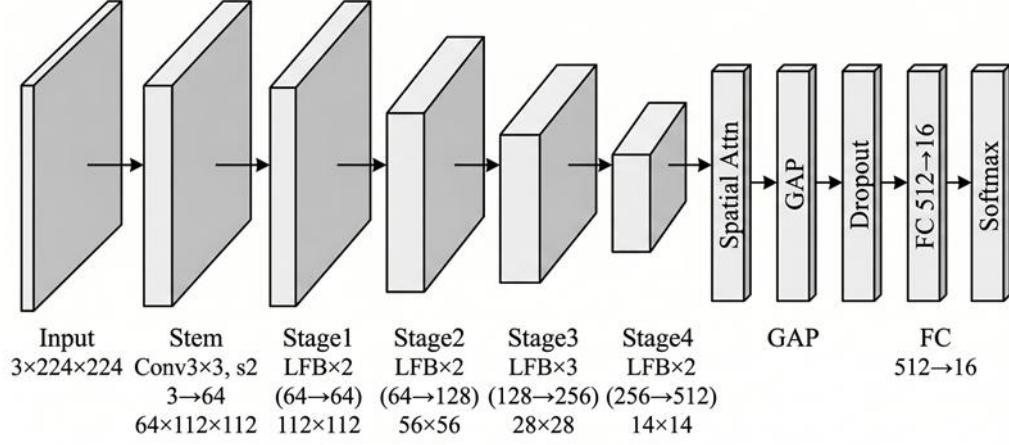


Fig. 15. Overall structure of EdgeHandCNN

For an RGB input of $224 \times 224 \times 3$, EdgeHandCNN first applies a stem module for initial feature extraction and downsampling, increasing the channel dimension to 64. The feature map is then processed by four hierarchical stages with output channels of 64, 128, 256, and 512, which progressively encode local edges, finger structures, and higher-level hand morphology.

The Local Feature Block (LFB) serves as the core computational unit of each feature extraction stage, and the internal structure of each stage is presented in Fig. 16.

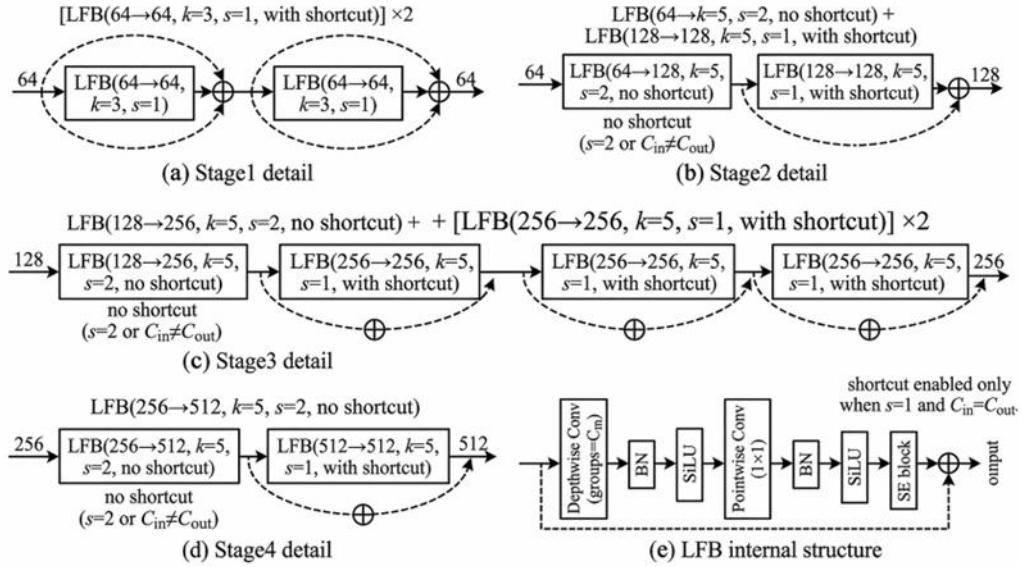


Fig. 16. The structure of LFB and each hierarchical stage

The Local Feature Block consists of depthwise convolution, pointwise convolution, batch normalization, SE-based channel attention, SiLU activation, and a residual connection when the input and output dimensions are compatible. SE-based channel attention recalibrates channel responses to emphasize informative hand morphology cues [24]. SiLU activation provides a smooth nonlinear transformation for feature

learning ^[25]. Depthwise and pointwise convolutions reduce computational redundancy, while residual connections improve feature propagation.

After the four stages, a spatial attention module highlights discriminative regions such as fingertips, palm edges, and joint transitions. The aggregated feature map is then passed through global average pooling, dropout, and a fully connected layer to produce the 16-class prediction.

In this study, LeNet-5 is retained only as a classical shallow-CNN baseline. The comparison is used to evaluate whether the self-designed lightweight architecture can better exploit segmented foregrounds, rather than to imply a stepwise derivation from LeNet-5.

3.3 Classification Performance Evaluation

The classification dataset, denoted as Dataset C, is assembled from two complementary sources. The first source is generated from Dataset B using a five-fold OOF protocol. Dataset B is partitioned into five subsets. In each iteration, the segmentation model is trained on four subsets and predicts foregrounds for the remaining hold-out subset. This cross-validated procedure ensures that the classifier is trained on realistically predicted foregrounds rather than idealized ground-truth masks, thereby reducing data leakage and improving deployment consistency. The second source consists of randomly sampled frames from collected gesture videos, which are segmented by the improved U-Net to obtain hand-foreground images. After merging the two sources, the final training, validation, and test sets contain 12,584, 1,072, and 1,101 images, respectively.

Accuracy, precision, recall, and F1-score were used to evaluate the two classification models. For the 16-class task, precision, recall, and F1-score were computed by macro averaging across all classes. Accuracy is defined as the proportion of correctly classified samples among all samples and is given by:

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} \quad (19)$$

Precision is defined as the proportion of true positive samples among all samples predicted as positive and is given by:

$$Pre = \frac{TP}{TP + FP} \quad (20)$$

Recall is defined as the proportion of true positive samples among all actual positive

samples and is given by:

$$Re = \frac{TP}{TP + FN} \quad (21)$$

F1-score is defined as the harmonic mean of precision and recall:

$$F1 = 2 \times \frac{Pre \times Re}{Pre + Re} \quad (22)$$

During classification training, only the training subset was augmented. Geometric and photometric transformations were combined with Cutout to increase sample diversity and reduce overfitting caused by inter-frame similarity^[12]. In the present two-stage framework, Cutout simulates localized foreground occlusion and imperfect segmentation artifacts, such as missing fingertips or broken palm edges. This encourages EdgeHandCNN to learn global hand morphology rather than relying on isolated local cues.

Training and validation were conducted on Environment B. Final testing was performed on Environment A, using the same software stack where applicable. Both networks were trained with AdamW optimization, cross-entropy loss, label smoothing, and cosine annealing learning-rate scheduling. LeNet-5 was trained for 60 epochs, and EdgeHandCNN was trained for 80 epochs. The batch size was 64 for both models, and the label-smoothing coefficient was 0.15. Fig. 17 shows the accuracy and loss curves for LeNet-5, and Fig. 18 shows the corresponding curves for EdgeHandCNN.

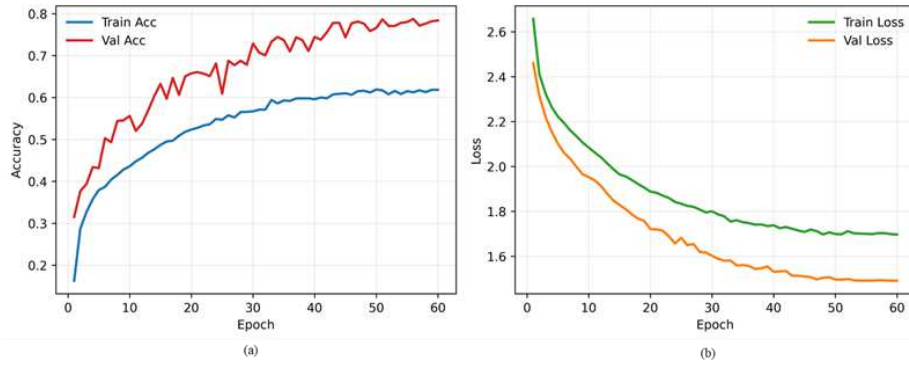


Fig. 17. Accuracy and loss curves of LeNet-5 on Dataset C

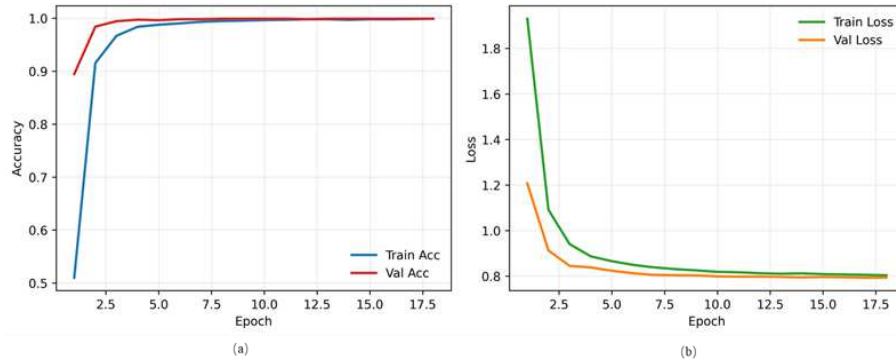


Fig. 18. Accuracy and loss curves of EdgeHandCNN on Dataset C

Fig. 17 demonstrates that the traditional LeNet-5 network exhibits relatively slow convergence speed throughout the entire training process, and finally plateaus at a low validation accuracy of around 0.78. Insufficient feature extraction capability and weak discriminative representation lead to its unsatisfactory steady-state recognition performance. In contrast, Fig. 18 clearly illustrates that the proposed EdgeHandCNN achieves rapid convergence merely within a small number of training epochs, and continuously climbs to an optimal validation accuracy up to 0.9990. Such remarkable performance fully verifies that EdgeHandCNN possesses far superior feature modeling and feature fitting ability for segmented hand foreground images compared with classical lightweight networks.

Fig. 19 and Fig. 20 display the detailed test-set confusion matrices of the two classification architectures. It can be observed that LeNet-5 suffers from serious misclassification phenomena, and frequently confuses adjacent finger gestures as well as gesture categories with highly analogous local contour structures. By comparison, EdgeHandCNN exhibits minimal classification errors among extremely visually similar gesture categories, and maintains highly consistent prediction results across most classes, which reflects more distinct and distinguishable feature distribution as well as sharper decision boundaries between different gesture types.

Comprehensive quantitative recognition metrics including overall accuracy, precision, recall and F1-score of the two gesture classification models are statistically summarized in Table 5.

Table 5. Metrics and parameter counts of two classification models on the test set

Model	Accuracy	Precision	Recall	F1-score	Speed (ms/image)	Parameters
LeNet-5	0.7584	0.7970	0.7658	0.7681	1.0452	62,516
EdgeHandCNN	0.9955	0.9958	0.9956	0.9956	1.8757	1,029,874

As shown in Table 5, LeNet-5 achieves 0.7584 accuracy, 0.7970 macro-precision, 0.7658 macro-recall, and 0.7681 macro-F1. EdgeHandCNN achieves 0.9955 accuracy, 0.9958 macro-precision, 0.9956 macro-recall, and 0.9956 macro-F1. Although EdgeHandCNN has a larger parameter count of 1,029,874, its average classification time remains only 1.8757 ms/image, which satisfies the real-time requirement of the proposed pipeline.

Combining EdgeHandCNN with the improved U-Net yields a complete neural-network pipeline with 8,792,915 parameters, 28.2705 GFLOPs, and a model size of

about 92.9 MB. The core network inference time is roughly 9 ms per frame on the current desktop platform, whereas the full prototype latency is 30-40 ms per frame after frame acquisition, ROI handling, visualization, and interface overhead are included.

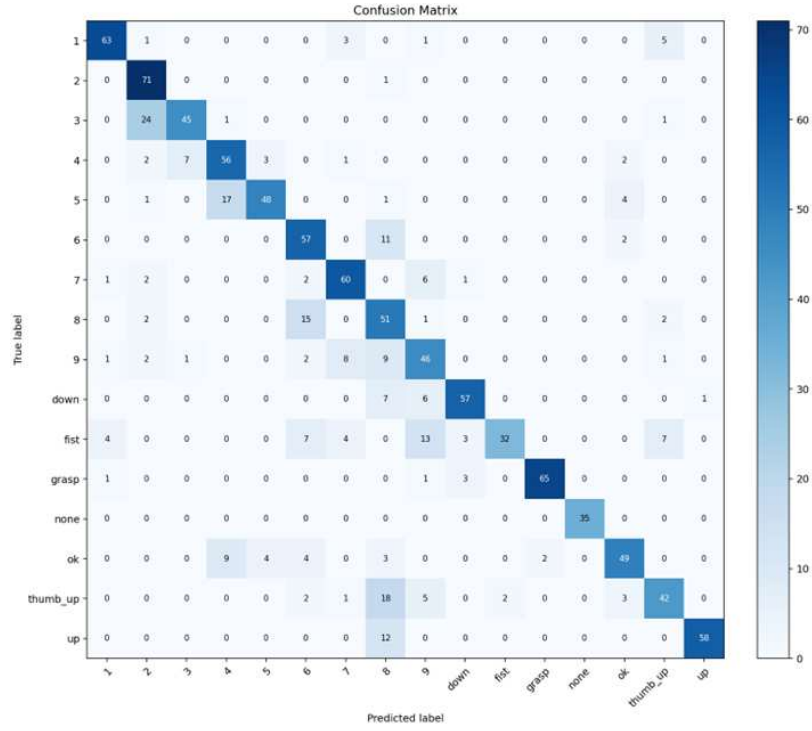


Fig. 19. Confusion matrix of LeNet-5 on the test set

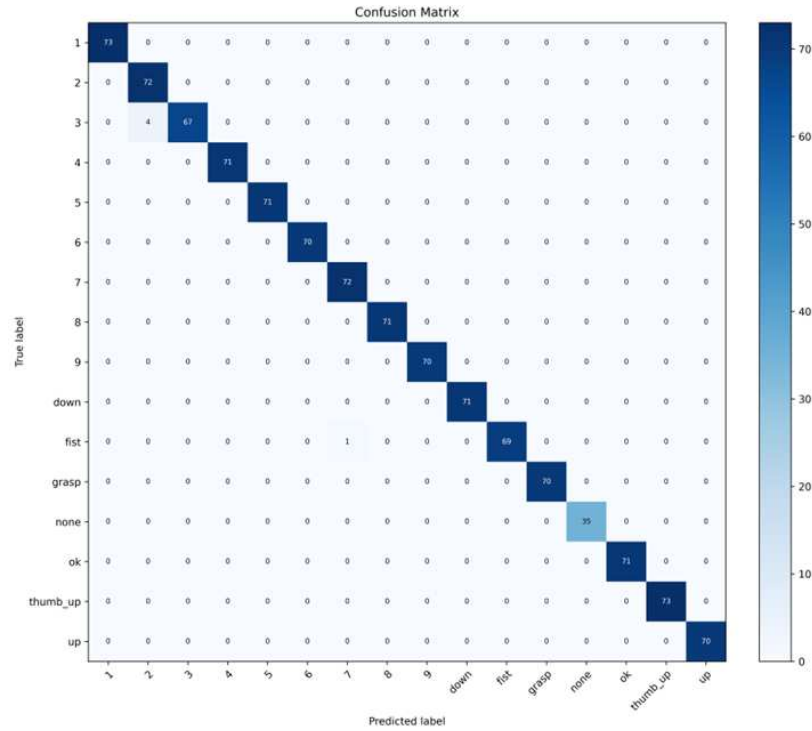


Fig. 20. Confusion matrix of EdgeHandCNN on the test set

The performance gap between LeNet-5 and EdgeHandCNN can be attributed to their structural differences. LeNet-5 provides only a shallow receptive hierarchy and limited abstraction capability, so it struggles with gestures that differ mainly in local finger configuration or subtle contour variation. EdgeHandCNN instead combines depthwise separable convolutions, SE-based channel recalibration, spatial attention, and residual learning within Local Feature Blocks. This design suppresses residual segmentation noise and focuses the representation on high-frequency morphological cues such as finger extensions, joint transitions, and palm boundaries.

These results confirm that foreground segmentation preserves the principal gesture topology, but reliable recognition of visually adjacent classes still requires stronger local context aggregation and adaptive feature recalibration than LeNet-5 can provide. Accordingly, EdgeHandCNN is selected as the final classification model for prototype validation.

3.4 End-to-End System Prototype Validation

To assess deployment-oriented feasibility, an end-to-end prototype was implemented around the cascaded improved U-Net and EdgeHandCNN pipeline. The prototype verifies whether foreground extraction, gesture classification, rejection of invalid frames, and command mapping can operate as a stable integrated workflow under interactive conditions.

The display interface of the prototype system is shown in Fig. 21. The prototype contains a parameter-setting panel, an image-display panel, and a recognition-result panel. The parameter-setting panel configures the segmentation model, classification model, inference device, foreground threshold, real-time interval, and ROI position. The image-display panel visualizes the live camera stream or an imported image together with the active ROI. The recognition-result panel reports the cropped ROI, segmentation mask, recovered foreground, predicted class, confidence, and Top-5 probabilities, enabling inspection of each processing stage.

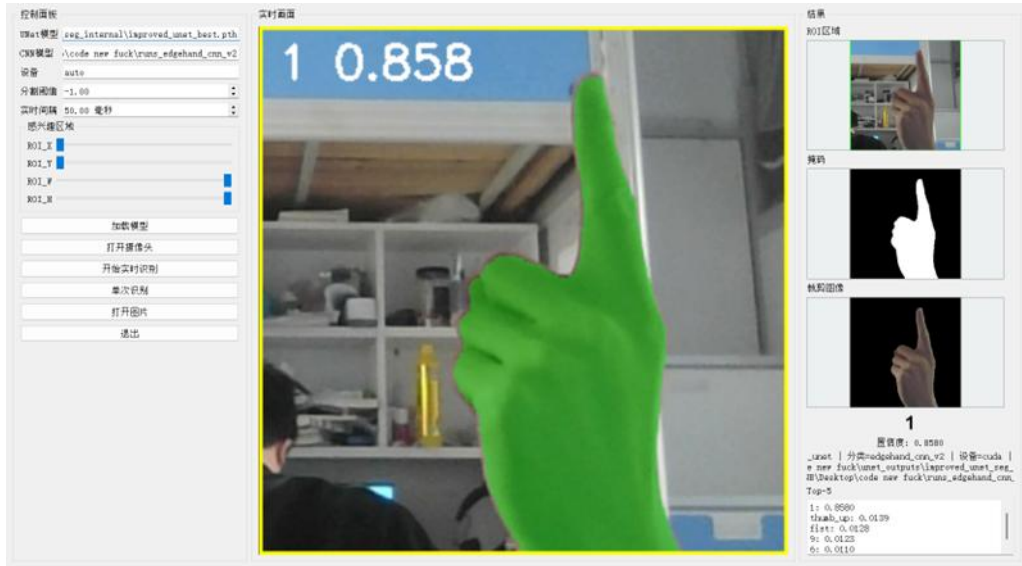


Fig. 21. Interface of the end-to-end system prototype

The recognition workflow is illustrated in Fig. 22. After ROI cropping, the frame is segmented by the improved U-Net, and the recovered foreground is forwarded to EdgeHandCNN for category prediction. To reject empty frames and transient motion artifacts before classification, a hard threshold based on foreground ratio is introduced. If the foreground occupies less than 3% of the ROI, the frame is directly assigned to the no-gesture state. This rejection mechanism reduces unstable command triggering caused by empty backgrounds or incomplete hand entry.

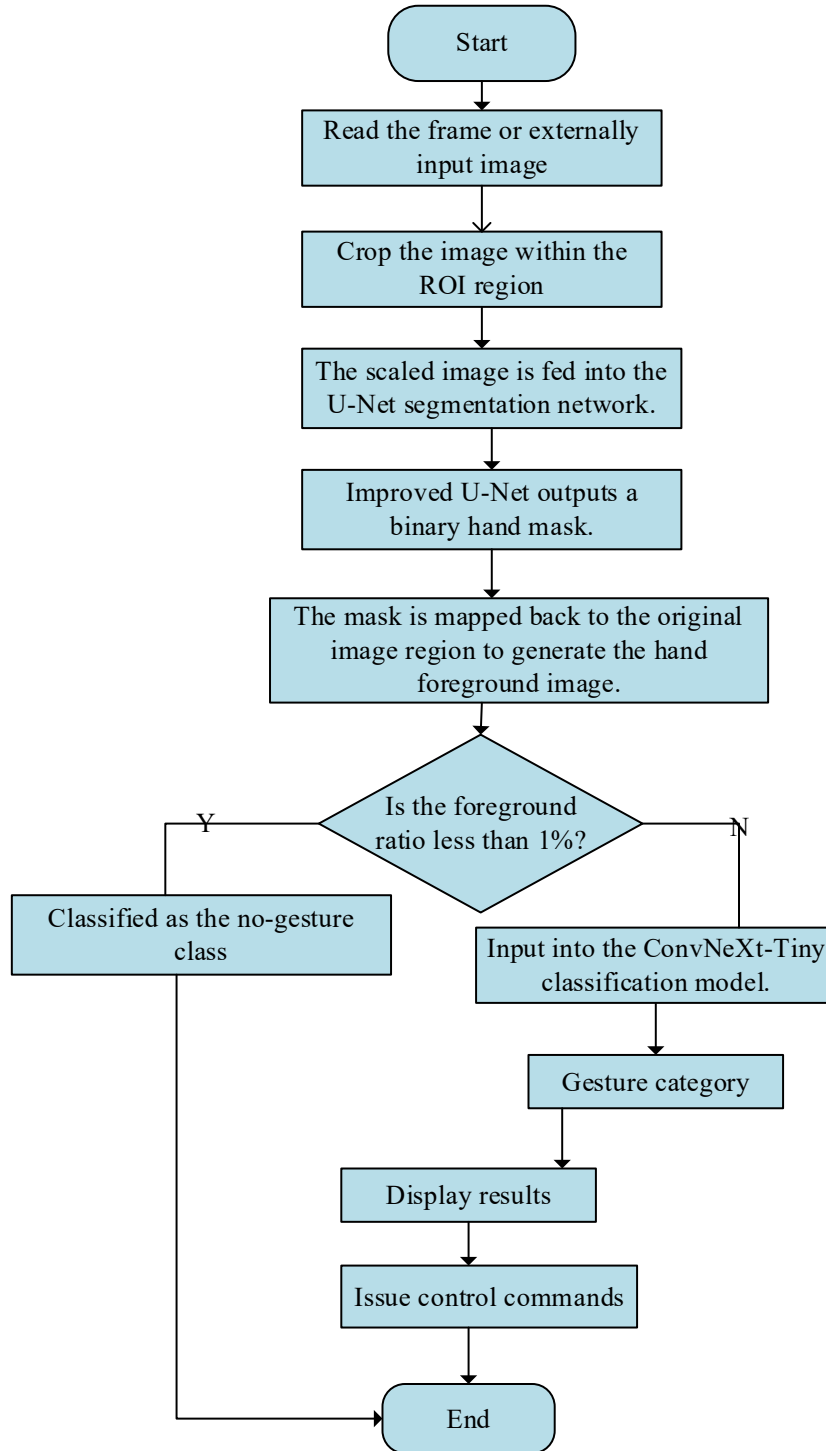


Fig. 22. Workflow of the end-to-end system prototype

In real-time camera mode, the system acquires frames at a fixed timer interval of 50 ms, and an exponential moving average is applied to the output probabilities to stabilize the predicted category. In single-image mode, the same segmentation-classification pipeline is executed on an imported local image, which confirms consistent behavior between streaming inference and offline inspection.

Prototype efficiency was evaluated by decomposing single-frame processing into segmentation inference, classification inference, and interface handling. The overall latency of approximately 30-40 ms per frame on a desktop platform fully satisfies the real-time constraints of smart-home interaction. Therefore, the prototype provides system-level feasibility evidence for the proposed framework, although embedded-device deployment still requires separate profiling.

4 Conclusion

This paper proposes a robust and real-time two-stage framework for static hand gesture recognition in smart-home control. By explicitly decoupling hand foreground extraction from gesture classification, the proposed pipeline effectively suppresses complex indoor visual interference while preserving a lightweight computational profile.

The proposed improved U-Net reduces the segmentation parameter count by 75% relative to the standard U-Net, while maintaining high foreground extraction accuracy and satisfactory generalization on both internal and external datasets. Based on the segmented hand foreground, the self-designed EdgeHandCNN achieves an accuracy of 0.9955 and a macro-F1 score of 0.9956, substantially outperforming the LeNet-5 baseline. The complete framework contains only 8.79 million parameters and 28.27 GFLOPs, with a storage requirement of approximately 93 MB. On a desktop platform, the core model inference time is approximately 9 ms per frame, and the end-to-end prototype latency remains within 30-40 ms per frame, corresponding to a practical throughput of approximately 25-33 FPS. These results demonstrate that the proposed framework achieves a favorable balance among accuracy, robustness, lightweight design, and real-time responsiveness, thereby satisfying the practical requirements of contactless smart-home interaction.

Future work will focus on extending the framework to dynamic gesture recognition, improving cross-user and cross-scene generalization, and investigating quantized deployment strategies for resource-constrained IoT edge devices.

Acknowledgements

This work was supported by no Fund.

Conflict-of-Interest Statement

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

Data Availability Statement

The data that support the findings of this study are available from the authors. Restrictions apply to the availability of these data, which were used under licence for this study. Data are available from the authors.

Conflict of interest

The authors do not have any conflict of interest with other entities or researchers.

Funding Declaration

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Ethics statement

This study involved secondary analysis of de-identified data obtained from existing datasets. No new participants were recruited for the present study. Ethical approval and informed consent were obtained by the original investigators at the participating sites. The present study used only de-identified data released for research purposes under the corresponding data-use terms.

Code availability

The code used for model training, evaluation, explanation analysis, and visualization is available from the corresponding author upon reasonable request.

Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of this work, the authors used ChatGPT and Codex (OpenAI) to assist with language editing, manuscript organization, and formatting. After using these tools, the authors reviewed and edited the content as needed, verified the scientific content, numerical results, and references, and take full responsibility for the coclinical trial number tent of the publication.

Clinical trial number

NCT followed by an 8-digit number (NCT00000419).

Approval (minimal risk)

Human participants are not involved in the study.

References

-
- [1] Qi J, Ma L, Cui Z, et al. Computer vision-based hand gesture recognition for human-robot interaction: a review[J]. *Complex & Intelligent Systems*, 2024, 10: 1581-1606. DOI: 10.1007/s40747-023-01173-6.
 - [2] Fang B, Sun F, Liu H, et al. 3D human gesture capturing and recognition by the IMMU-based data glove[J]. *Neurocomputing*, 2018, 277: 198-207.
 - [3] Ni S, Al-qaness M A A, Hawbani A, et al. A survey on hand gesture recognition based on surface electromyography: Fundamentals, methods, applications, challenges and future trends[J]. *Applied Soft Computing*, 2024, 166: 112235.
 - [4] Li W, Shi P, Yu H. Gesture Recognition Using Surface Electromyography and Deep Learning for Prostheses Hand: State-of-the-Art, Challenges, and Future[J]. *Frontiers in Neuroscience*, 2021, 15: 621885.
 - [5] Oudah M, Al-Naji A, Chahl J. Hand Gesture Recognition Based on Computer Vision: A Review of Techniques[J]. *Journal of Imaging*, 2020, 6(8): 73.
 - [6] Al Farid F, Hashim N, Abdullah J, et al. A Structured and Methodological Review on Vision-Based Hand Gesture Recognition System[J]. *Journal of Imaging*, 2022, 8(6): 153. DOI: 10.3390/jimaging8060153.
 - [7] Sanchez-Brizuela G, Cisnal A, de la Fuente-Lopez E, et al. Lightweight real-time hand segmentation leveraging MediaPipe landmark detection[J]. *Virtual Reality*, 2023, 27: 3125-3132.
 - [8] Meng Y, Jiang H, Duan N, et al. Real-Time Hand Gesture Monitoring Model Based on MediaPipe's Registerable System[J]. *Sensors*, 2024, 24(19): 6262.
 - [9] Zhang Y, Wang J, Wang X, et al. Static hand gesture recognition method based on the Vision Transformer[J]. *Multimedia Tools and Applications*, 2023, 82(20): 31309-31328.
 - [10] Zhang F, Bazarevsky V, Vakunov A, et al. MediaPipe Hands: On-device real-time hand tracking[EB/OL]. arXiv, 2020 [2026-04-17]. <https://arxiv.org/abs/2006.10214>. DOI: 10.48550/arXiv.2006.10214.
 - [11] Bambach S, Lee S, Crandall D J, et al. Lending A Hand: Detecting Hands a

-
- nd Recognizing Activities in Complex Egocentric Interactions[C]//Proceedings of the IEEE International Conference on Computer Vision. Santiago: IEEE, 2015: 1949-1957.
- [12] DeVries T, Taylor G W. Improved regularization of convolutional neural networks with Cutout[EB/OL]. arXiv, 2017 [2026-04-15]. <https://arxiv.org/abs/1708.04552>. DOI: 10.48550/arXiv.1708.04552.
- [13] Kirillov A, Mintun E, Ravi N, et al. Segment Anything[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. Paris: IEEE, 2023: 4015-4026. DOI: 10.1109/ICCV51070.2023.00371.
- [14] Buchsbaum G. A spatial processor model for object colour perception[J]. Journal of the Franklin Institute, 1980, 310(1): 1-26. DOI: 10.1016/0016-0032(80)90058-7.
- [15] Smith A R. Color gamut transform pairs[J]. ACM SIGGRAPH Computer Graphics, 1978, 12(3): 12-19. DOI: 10.1145/965139.807361.
- [16] Otsu N. A threshold selection method from gray-level histograms[J]. IEEE Transactions on Systems, Man, and Cybernetics, 1979, 9(1): 62-66. DOI: 10.1109/TSMC.1979.4310076.
- [17] Serra J. Image Analysis and Mathematical Morphology[M]. London: Academic Press, 1982.
- [18] Rosenfeld A, Pfaltz J L. Sequential operations in digital picture processing[J]. Journal of the ACM, 1966, 13(4): 471-494. DOI: 10.1145/321356.321357.
- [19] Ronneberger O, Fischer P, Brox T. U-Net: Convolutional networks for biomedical image segmentation[C]//Medical Image Computing and Computer-Assisted Intervention. Cham: Springer, 2015: 234-241. DOI: 10.1007/978-3-319-24574-4_28.
- [20] Loshchilov I, Hutter F. Decoupled weight decay regularization[C]//International Conference on Learning Representations. New Orleans: OpenReview, 2019.
- [21] Loshchilov I, Hutter F. SGDR: Stochastic gradient descent with warm restarts[C]//International Conference on Learning Representations. Toulon: OpenRev

iew, 2017.

- [22] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324. DOI: 10.1109/5.726791.
- [23] Szegedy C, Vanhoucke V, Ioffe S, et al. Rethinking the Inception Architecture for Computer Vision[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas: IEEE, 2016: 2818-2826. DOI: 10.1109/CVPR.2016.308.
- [24] Hu J, Shen L, Sun G. Squeeze-and-Excitation Networks[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City: IEEE, 2018: 7132-7141. DOI: 10.1109/CVPR.2018.00745.
- [25] Ramachandran P, Zoph B, Le Q V. Searching for activation functions[C]//International Conference on Learning Representations. 2018.