HAND GESTURE DETECTION AND RECOGNITION FOR GESTURE-BASED PATIENT REHABILITATION AND ASSISTANCE SYSTEMS

A

Thesis submitted

in Partial Fulfilment of the Requirements

for the Degree of

Doctor of Philosophy

by

H Pallab Jyoti Dutta

Department of Electronics and Electrical Engineering
Indian Institute of Technology Guwahati
Guwahati - 781039, Assam, India
December, 2024

To

My Parents

Paresh Chandra Dutta & Jun Moni Dutta

Certificate

This is to certify that the thesis entitled "**Hand Gesture Detection and Recognition for Gesture-based Patient Rehabilitation and Assistance Systems**", submitted by **H Pallab Jyoti Dutta** (186102008), a research scholar in the *Department of Electronics and Electrical Engineering, Indian Institute of Technology, Guwahati*, for the award of the degree of **Doctor of Philosophy**, is a record of an original research work carried out by him under my supervision and guidance. The thesis has fulfilled all requirements as per the regulations of the institute and in my opinion has reached the standard needed for submission. The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Date:

Place: Guwahati

Prof. M. K. Bhuyan

Dept. of Electronics and Electrical Engg.,

Indian Institute of Technology, Guwahati,

Guwahati - 781039, Assam, India.

Acknowledgements

I am grateful to the Almighty God for his divine blessings and support. This thesis would not have been possible without several people's immense help and support. I would like to convey my acknowledgment to all of them.

First and foremost, I feel it is a great privilege to express my deepest and most sincere gratitude to my supervisor Prof. M. K. Bhuyan for his excellent guidance throughout my study. It is very difficult to describe my feelings in words to acknowledge my supervisor for his continuous guidance, constant motivation, and support throughout my doctoral studies. I am very much thankful to him for helping me with my research and other issues. It would be impossible for me to bring the research and thesis to this form without the ample facilities he provided in the IPCV laboratory and the freedom to work independently.

I am also very thankful to my doctoral committee members Dr. Kannan Karthik, Dr. Rishikesh Dilip Kulkarni, and Dr. Samit Bhattacharya for sparing their precious time out of their busy schedules to evaluate my progress and enrich this work with their invaluable suggestions and feedbacks.

I sincerely thank Dr. Debanga Raj Neog for his valuable suggestions on my research work and for collaborating with me. I would also like to thank the Head of the Department and other faculty members for their kind help in carrying out this work. I am also grateful to my friends, Mr. Saquib Mazhar, Dr. Govindaraj P., Dr. Aniruddha Mazumdar, Mr. Pharvesh Salman Choudhary, Dr. Pradipta Sasmal, Dr. Debajit Sarma, Mr. Nayan Moni Baishya, Mr. Bibek Goswami, Mr. Nadeem Atif, Mr. Rijuban Rangslang, Mr. Kamal, Mr. Nitin Yadav, my lab-mates and seniors. They have always been around to provide useful suggestions and companionship and created a fun-filled research environment.

I have no words to express my thanks and gratitude to my go-to friends, especially, Brij Nandan Tripathy, Vineet Kumar, Harshal Chaudhari, Allen Patnaik, Pranjali Singh, Mousumi Das, and Anik Ghosh. They helped me a lot in releasing the work pressure and channelling it into a positive direction. My work in this place, far away from home, definitely would have

been painful without their love and care that helped me to enjoy my life at IIT Guwahati. My stay at IIT Guwahati is incomplete without them.

My deepest gratitude goes to my parents for being my rock and supporting me throughout my life. The opportunities that they have given me and their immense sacrifices are the reasons for where I am and what I have accomplished so far.

(H Pallab Jyoti Dutta)

Abstract

Hand gestures serve as a natural and widely used means of human interaction, playing a crucial role in establishing seamless human-computer interactions. However, to facilitate effortless interactions, precise decoding of the hand gestures is essential, which is hindered by background clutter, variations in illumination, the presence of skin areas, such as hands or faces in the vicinity, occlusion, and variable hand shapes and sizes. Much work has been done in the literature to address these issues and recognize the gestures, but the generalization is yet to be achieved. This dissertation aims to address these concerns by developing a robust method for hand gesture recognition and applying it to create interfaces that enable human-computer interaction tailored to specific human needs.

A method is proposed to segment the hand region in an image that removes the irrelevant information from the background. For this, two segmentation models were proposed; one model utilizes spatial and channel attention and the other benefits from combining a convolution neural network and a linearized transformer unit. Moreover, a novel loss function optimizes the models to resolve class imbalance, ensure boundary smoothness, and retain the hand's shape. These segmented results were further utilized to obtain hand gesture recognition results in a two-stage arrangement. A novel adaptive kernel channel attention layer assists the recognition network in achieving accurate results. The recognition accuracy for two benchmark datasets was 93.8% and 98.0%, which highlights the preciseness of the proposed approach.

This two-stage approach is not very suitable for online applications. Therefore, three hand detection methods that localized the hand region and gesture class con-

currently were proposed. The first method is an anchor boxes-based RetinaNet-CBAM hand gesture detection model. The second and third methods are anchor-less and detect hand gestures using a detection transformer and CenterNet-based model, respectively. The best-performing model, i.e., the second method, achieved a recognition rate of 89.6% and 100% for two benchmark datasets. Once a robust detection model is available, it can be used to model gesture-operated interfaces for specific tasks. Hand keypoint detection also plays an important role in these interfaces, and hence, a robust keypoint detection model with a multiscale attention block is proposed. In this work, two interfaces were designed that cater to patients undergoing hand rehabilitation and patients in hospitals communicating with medical staff.

The proposed methods underwent thorough qualitative and quantitative analysis, revealing state-of-the-art performance even under challenging conditions. The seamless integration of the hand detection algorithm into the interfaces was also successfully accomplished. Patients using the rehabilitation interface reported noticeable improvements in hand functioning, while those utilizing the communication interface experienced smooth and efficient communication with medical staff. These outcomes underscore the effectiveness of the proposed methods, demonstrating their practical applicability in real-world scenarios.

Contents

List of Figures

List of Tables

List of Acronyms

AKCAL	Adaptive Kernel Channel Attention Layer
AR/VR	Augmented Reality/ Virtual Reality
ATM	Automated Teller Machine
BCE	Binary Cross Entropy
BFL	Binary Focal Loss
CAM	Channel Attention Module
CBAM	Convolutional Block Attention Module
CFAB	Channel Feature Attention Block
CTB	Convolution Transformer Block
DA-Net	Dual Attention Network
DETR	Detection Transformer
DTW	Dynamic Time Warping
EPE	End Point Error
FD	Fourier Descriptor
FPN	Feature Pyramid Network
HCI	Human Computer Interaction
HGD	Hand Gesture Detection
HKD	Hand Keypoint Detection
HKM	Hand Keypoint Module
HMM	Hidden Markov Model

HOG	Histogram of Oriented Gradients
IoU	Intersection over Union
MHA	Multihead Attention
MLP	Multilayer Perceptron
MSAB	Multiscale Attention Block
NMS	Non-maximal Suppression
NUS	National University of Singapore
PAM	Position Attention Module
PAS	Patient Assistance System
PCK	Percentage of Correct Keypoints
RCB	Residual Convolution Block
ROI	Region of Interest
SAM	Self-attention Mechanism
SFAB	Spatial Feature Attention Block
SIFT	Scale Invariant Feature Transform
SSD	Single Shot Multibox Detection
STB	Stereo Hand Pose Tracking Benchmark
SVM	Support Vector Machine
VGR	Vision-based Gesture Recognition
VHGR	Vision-based Hand Gesture Recognition

List of Symbols

$\mathbf{M}_q$	Query matrix	$Up \uparrow$	Upsampling layer
$\mathbf{M}_k$	Key matrix	$sigmoid$	Sigmoid activation function
$\mathbf{M}_v$	Value matrix	$D_{0.1}$	Dropout layer with dropout rate = 0.1
$softmax$	Softmax activation function	D	Dense Layer
$\hat{y}$	Predicted label	$\text{attention}_{\text{new}}$	Attention map from the attention module in the transformer unit
α	Balancing factor in Binary Focal Loss	$\mathcal{M}$	Max pooling layer
β	Modulating factor in Binary Focal Loss	$\circ$	$(f \circ g)(x) = f(g(x))$ which means applying the function g on x and then the function f on $g(x)$
∇	Gradient operator	$\oplus^1, \oplus^2, \oplus^3$	Skip-connections from the encoder to the decoder
S_P	Skeleton of predicted mask	$\oplus$	Element-wise addition
S_T	Skeleton of true mask	$\otimes$	Element-wise multiplication
ℓ_{smooth}	Smoothing loss	$\mathbb{A}$	Average pooling layer
ℓ_{skdice}	Dice loss with skeletal information	ℓ_{H}	Hungarian loss
$\mathcal{A}$	Confusion matrix	ℓ_{GIOU}	Generalized intersection over union loss
BN	Batch Normalization	ℓ_{skdice}	Least absolute deviation loss
LN	Layer Normalization	L_k	Penalty-reduced focal loss

$conv$	Convolution layer	L_{size}	Size loss
$\mathcal{R}$	ReLU activation function	L_{size}	Offset loss
DP_{conv}	Depth + pointwise convolution layer	ℓ_{IOU}	Intersection over union loss
$\mathcal{P}$	Converts the feature map into patches	H_{pred}	Predicted keypoint heatmap
$\mathcal{U}$	Unpatching into a feature map	H_{true}	True keypoint heatmap
$Transformer^{(n)}$	Stack n transformer layers sequentially	$\mathcal{K}$	Actual keypoints
μ_x, μ_y	mean along x and y directions	$\hat{\mathcal{K}}$	Predicted keypoints
σ_x, σ_y	standard deviation along x and y directions	$\hat{\mathcal{K}}$	Predicted keypoints

List of Publications

Journal Publications

1. **H Pallab Jyoti Dutta**, M. K. Bhuyan, D. R. Neog, K. F. MacDorman and R. H. Laskar, "Efficient hand segmentation for rehabilitation tasks using a convolution neural network with attention," in Expert Systems with Applications, 234, 121046, doi: https://doi.org/10.1016/j.eswa.2023.121046.
2. **H Pallab Jyoti Dutta**, M. K. Bhuyan, D. R. Neog, K. F. MacDorman and R. H. Laskar, "Patient Assistance System Based on Hand Gesture Recognition," in IEEE Transactions on Instrumentation and Measurement, vol. 72, pp. 1-13, 2023, Art no. 5018013, doi: https://doi.org/10.1109/TIM.2023.3282655.
3. **H Pallab Jyoti Dutta**, M. K. Bhuyan, D. R. Neog, K. F. MacDorman and R. H. Laskar, "A Hand Gesture-Operated System for Rehabilitation Using an End-to-End Detection Framework," in IEEE Transactions on Artificial Intelligence, vol. 5, no. 2, pp. 698-708, Feb. 2024, doi: https://doi.org/10.1109/TAI.2023.3251309.
4. **H Pallab Jyoti Dutta**, M. K. Bhuyan, R. K. Karsh, S. Alfarhood, and M. Safran, "Multiscale Attention-Based Hand Keypoint Detection," in IEEE Transactions on Instrumentation and Measurement, vol. 73, pp. 1-11, 2024, Art no. 5022811, doi: https://doi.org/10.1109/TIM.2024.3413196.
5. **H Pallab Jyoti Dutta**, M. K. Bhuyan, "Attention-Based 2-D Hand Keypoints Localization," in IEEE Sensors Letters, vol. 8, no. 9, pp. 1-4, Sept. 2024, Art no. 6011104, doi: https://doi.org/10.1109/LSENS.2024.3443072.
6. Debajit Sarma, **H Pallab Jyoti Dutta**, Kuldeep Singh Yadav, M.K. Bhuyan, and Rabul Hussain Laskar,"Attention-based hand semantic segmentation and gesture recognition

using deep networks," in Evolving Systems, 15, pp. 185–201, 2023, doi: https://doi.org/10.1007/s12530-023-09512-1.

Conferences

1. **H Pallab Jyoti Dutta**, Debajit Sarma, M. K. Bhuyan and R. H. Laskar, "Semantic Segmentation based Hand Gesture Recognition using Deep Neural Networks," 2020 National Conference on Communications (NCC), 2020, pp. 1-6, doi: https://doi.org/10.1109/NCC48643.2020.9055990.
2. **H Pallab Jyoti Dutta**, D. R. Neog, M.K. Bhuyan, M. Das and R. H. Laskar, "Two-Stage Hand Gesture Recognition based on Hand Keypoints Localization," 2022 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET), 2022, pp. 110-114, doi: https://doi.org/10.1109/WiSPNET54241.2022.9767161.
3. **H Pallab Jyoti Dutta**, K. Manivas, M. Bhuyan and M. K. Bhuyan, "An End-to-end Anchorless Approach to Recognize Hand Gestures using CenterNet," 2023 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT), BALI, Indonesia, 2023, pp. 1-6, doi: https://doi.org/10.1109/IAICT59002.2023.10205726.
4. S. Sharma, **H Pallab Jyoti Dutta**, M. K. Bhuyan and R. H. Laskar, "Hand Gesture Localization and Classification by Deep Neural Network for Online Text Entry," 2020 IEEE Applied Signal Processing Conference (ASPCON), 2020, pp. 298-302, doi: https://doi.org/10.1109/ASPCON49795.2020.9276713.
5. K.Y. Suguna, **H Pallab Jyoti Dutta**, M.K. Bhuyan, and R. H. Laskar, "Hand Gesture Recognition Using CBAM-RetinaNet," 2021 International Conference on Computer Vision and Image Processing (CVIP), Communications in Computer and Information Science (2022), vol 1568. Springer, Cham. doi: https://doi.org/10.1007/978-3031113499_38.

6. Kamal, D. R. Neog, **H Pallab Jyoti Dutta**, and M. K. Bhuyan, "Hand Function Assessment using Computer Vision for Hand Rehabilitation," 2024 IEEE Conference on Artificial Intelligence (CAI), Singapore, Singapore, 2024, pp. 659-664, doi: https://doi.org/10.1109/CAI59869.2024.00129.

1

Introduction

Contents

1.1 Introduction

In present times, computers have become an integral aspect of our daily lives and are extensively utilized. Consequently, there is a growing demand for interfaces that facilitate efficient Human-Computer Interaction (HCI) [1, 2]. HCI involves the design and implementation of technology that establishes an interface for communication between humans and computers. It is dedicated not only to improving the usability, dependability, and functionality of existing interfaces but also to creating new and inventive interfaces capable of being employed in natural, lifelike manners. These interfaces are sought after for interaction in virtual environments, such as computer games and virtual reality, as well as for teleoperation in robotic surgery.

The widespread adoption of personal computers, mobile devices, PDAs, and similar technologies underscores the universal need for interaction capabilities, catering not only to the general population but also to individuals with disabilities and the elderly. Conventional HCI tools, such as the mouse and keyboard, while fundamental, often pose challenges for persons with certain hand injury, disabilities, and the elderly. A promising solution to address these challenges is a vision-based approach to interaction. This approach envisions a non-contact virtual environment where traditional hardware like the mouse and keyboard is unnecessary, eliminating the need for users to be confined in front of a monitor or laptop to issue commands. This approach offers users the flexibility to work comfortably in any position, reducing the likelihood of fatigue or stress.

A vision-based hand gesture-controlled interface can come handy, where the user does not need to reach out to a particular hardware, and provide instructions to the interface via hand gesture. This set-up would definitely be effective and resourceful to that section of people who finds it difficult to resort to traditional methods of interacting with the computer.

1.2 Human-Computer Interaction

HCI studies the optimal design of interfaces enabling hassle-free and smooth interactions between humans and computers. It aims to make the interactions as easy as possible for an experienced as well as a new user. The goals of HCI concern “safety, utility, effectiveness,

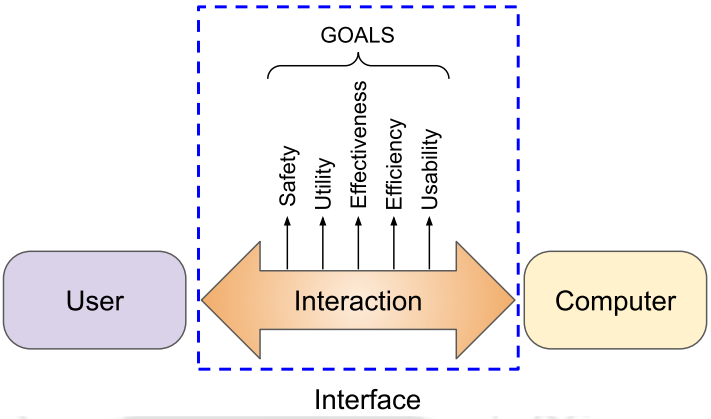

Fig. 1.1: Block diagram of a typical human-computer interaction.

efficiency, and usability" [3]. Fig. 1.1 gives an illustration of a typical HCI, which shows the three components, namely, user, computer, and interface. The user interacts with the computer through the interface where the computer may be a personal computer, embedded system, or any device with a processing unit. The user with a specific goal gives the necessary instruction through the interface to the computer, which is processed and communicated back to the user in a lucid form. The instructions are given mostly by gestures, speech, or text, and the interface serves as the connecting bridge between the user and the computer.

These interactions are intended to make the day-to-day life of people effortless. For instance, cash withdrawal from an automated teller machine (ATM). The interface allows easy withdrawal without the need for the user to fill out any form or wait in a queue at banks. Another example is operating lifts in multi-story buildings. These interactions will leave an amiable experience depending on how easily accessible and operable the interfaces are. Thus, hands being one of the natural and intuitive means of communication for humans, hand gestures play a key role in HCI. Moreover, the above-mentioned examples involve touch-based interactions, and the recent Coronavirus pandemic has prompted us to move from touch-based to touchless interactions. Though voice-based interactions are also possible, they are often not reliable in public places due to nearby noises affecting the voice commands.

Thus, hand gesture-based interactions can be user-friendly and appealing to a larger section of society. The hand gesture-based interactions would be beneficial for the deaf and mute to express themselves to others, for remote physiotherapy sessions for patients undergoing re-

habilitation, for car infotainment controls without taking eyes off the road, and many more. Therefore, the requirement for a robust HCI has increased with time along with the involvement of hand gestures for HCIs.

1.3 Hand Gesture Recognition

Vision-based gesture recognition (VGR) is a significant area of study crucial for advancing HCI systems. A gesture, in this context, refers to a meaningful and informative pose or physical movement involving the hands, arms, face, or body [2]. Within the realm of non-verbal communication between humans and computers, visual interfaces establish communication channels to deduce intentions from human behavior, including facial expressions and hand gestures. Specifically, the automated interpretation of hand gestures in interactions contributes to achieving the desired simplicity and naturalness in human-computer interaction. Given that the hand is the most commonly utilized body part [4], it becomes a primary focus in these endeavors.

1.3.1 Hand Gestures

Hand gestures are a natural and intuitive way of expressing oneself [1, 2] and studies show that even babies try communicating through hand gestures [5]. People from around the world always include hand gestures in their communication. Therefore, it is becoming a popular medium of interaction, especially with computers. Though hardware means of interaction using

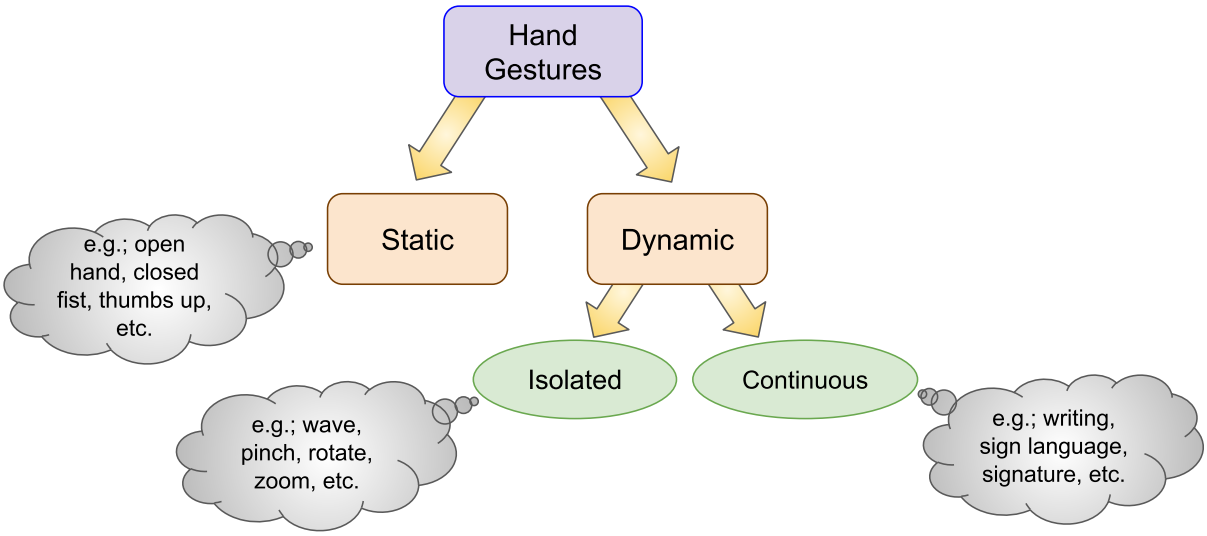

Fig. 1.2: Different type of hand gestures.

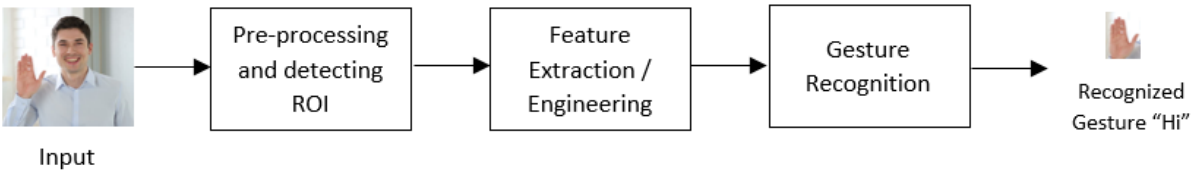

Fig. 1.3: Typical block diagram of a hand gesture recognition method.

a keyboard and mouse exist, they cannot provide the flexibility and expressiveness of hand gestures. On the other hand, hand gestures can perform symbolic and pointing commands to reciprocate keyboard and mouse to a great extent. Further, the recent advancements in vision-based hardware and hand gesture recognition methods have rendered significant involvements of hand gestures in HCI.

Hand gestures are mainly characterized by three attributes: hand posture or configuration, hand movements, and hand orientation [6]. Based on these attributes hand gestures can be classified into two categories, namely, static and dynamic. Static gestures do not change their form with time, i.e., the posture and orientation remain fixed and there is no movement. Dynamic gestures, on the other hand, contain movement and experience changes in posture and orientation with time. Fig. 1.2 shows the different types of hand gestures. Whenever a dynamic hand gesture is not followed by another gesture, it is an isolated gesture. Otherwise, it is a continuous one as there are other gestures following it with a break in the gesturing sequence.

1.3.2 Components of Hand Gesture Recognition

The primary objective of interfaces based on vision is to identify and comprehend visual information for communication purposes. The vision-based recognition approach is deemed more intuitive and user-friendly compared to sensor-based methods (such as glove-based recognition) or audio-based methods (such as speech/speaker recognition). It is easily deployable and applicable within the camera's field of view. The process for vision-based hand gesture recognition follows a sequential approach, as illustrated in Fig. 1.3, encompassing pre-processing and the detection of the Region of Interest (ROI), feature extraction/engineering, and recognition.

(i) Pre-processing and detecting ROI: Identifying the ROI, i.e., the gesturing body parts, holds significant importance as the precision of the VGR system relies on it. The prepro-

cessing step involves accurately segmenting gesturing body parts from images or videos, considering constraints, such as illumination variations, complex backgrounds, and instances of occlusion.

(ii) Feature extraction: The subsequent step involves selecting a mathematical representation of the gesture to succinctly describe it. Various authors have determined distinct features suitable for representing specific types of gestures. These features are broadly categorized as: a) shape, encompassing geometric or non-geometric features, b) texture or pixel value, c) 3D model-based features, and d) spatial features, including position and motion. In the context of Deep Learning-based approaches, there is no need for explicit feature extraction as the deep neural model inherently addresses this by uncovering feature maps, focusing on learning successive layers of increasingly meaningful representation.

(iii) Recognition: The final stage involves a classifier categorizes the incoming gesture features into predefined classes (supervised) or group them by their similarities (unsupervised). Different classifiers exixts for static (i.e., images) and dynamic (i.e., videos) gesture recognition, each with their own limitations.

1.4 Dataset

There are many hand gesture datasets available where the recognition techniques can be trained and tested accordingly [7]. The data samples for these datasets were taken under varied and challenging conditions like captured in the presence of backgrounds clutter, illumination variations, skin regions, occlusions, varying shapes and size of the hand, etc. Out of all the datasets available, a few of them are detailed below.

1.4.1 National University of Singapore (NUS)

The NUS Hand posture datasets [8] are divide into two parts- I and II.

NUS Hand posture data set I: It consists of 10 classes of hand gestures with uniform background. Each image is of size 160×120, and there are 24 images for each gesture. Both grayscale and color images are available.

NUS Hand posture data set II: It consists of 3 folders containing hand postures with clutter-

ing background (images of size 160×120), hand postures with human noises(images of size 320×240), and background images with no hand postures. There are a total of 2000 images for 10 classes, 750 images for 10 classes, and 2000 images in each folder in the respective order. The images are colored.

1.4.2 Ouhands

The Ouhands dataset [9] consists of about 3000 RGB images of hand gestures for 10 classes. The resolution of each image is 480 × 640 pixels. The images were taken from 23 individuals under complex conditions, such as background clutter, variations in illumination, different hand shapes and sizes, occlusions, etc. The dataset also contains segmented masks of the hand gestures, depth information, and bounding box annotations for each image.

1.4.3 HGR

The HGR dataset [10, 11] combines three datasets— HGR1, HGR2A, and HGR2B. The first dataset contains 899 images and 25 classes, the second dataset contains 85 images and 13 classes, and the third comprises 574 images and 32 classes. There are segmentation masks and hand joint locations for each image in the datasets.

1.4.4 HIU

The HIU dataset [12] contains 33,000 color images, corresponding hand segmentation masks, and hand joint locations.

1.4.5 Egohands

Egohands [13] is a segmentation dataset with 48 videos taken from a first-person view-point. There are 4800 annotated frames containing multiple hands, which complicates hand segmentation.

Ouhands and NUS contain labels for classification and HGR, HIU, and Egohands do not. Therefore, we evaluate segmentation using the Ouhands, HGR, HIU, and Egohands datasets and classification using the Ouhands and NUS datasets.

Other than hand recognition, hand keypoint detection is also explored, which is useful for working with certain hand operated interfaces. Hand keypoints detection localizes the hand joints in challenging conditions as experienced in hand gesture recognition. For hand keypoints detection, we used HIU dataset along with two other datasets, namely, FreiHAND and stereo hand pose tracking benchmark datasets.

1.4.6 FreiHAND

FreiHAND [14] dataset includes 130,240 RGB images for training and 3,960 for evaluation, each with a 224×224 pixel resolution. Each training sample includes a hand segmentation mask (224 × 224 pixels), 3D annotations for 21 hand keypoints, and 3D shape annotation. However, segmentation masks and hand keypoints are not included for evaluation samples, as the dataset web page handles their management for it being a part of a compettition.

1.4.7 Stereo Hand Pose Tracking Benchmark (STB)

STB dataset [15] comprises 18,000 images featuring six distinct backgrounds, each accompanied by 3D coordinates of the palm center and the finger joints, totaling 21 joints per image. These images include stereo and depth captures obtained simultaneously from a Point Grey Bumblebee2 stereo camera, an Intel Real Sense F200 active depth camera, and their corresponding camera parameters.

Samples of each dataset is shown in Fig. 1.4. The characteristics of the datasets are summarized in Table 1.1.

Table 1.1: Characteristics of the datasets.

Dataset	Purpose	Number of samples	Illumination Variations	Background clutter	Specularity	Varying size and shape	Other skin regions
Ouhands	Classification & Segmentation	3000	✓	✓	✓	✓	✓
NUS	Classification	240 + 2750	✓	✓	✓	✓	✓
HGR	Segmentation	899 + 85 + 574	-	✓	-	✓	-
HIU	Segmentation & Keypoint Detection	33000	✓	✓	-	✓	✓
Egohands	Segmentation	4800	✓	✓	-	✓	✓
FreiHAND	Keypoint Detection	130240 - training 3960 - evaluation	-	✓	-	✓	-
STB	Keypoint Detection	18000	✓	✓	-	-	✓

Fig. 1.4: A few data samples of each dataset used in this thesis. (a) OUHANDS, (b) HGR, (c) NUS, (d) HIU, (e) Egohands, (f) FreiHAND, and (g) STB datasets.

1.5 Literature Survey

Hand gestures offer a natural and spontaneous way to communicate [1, 16]. People convey much information through hand gestures, especially those with speech disorders, disabilities, or in hospital. Thus, hand gesture recognition plays a vital role in communication. Moreover, hand gestures find extensive application in HCI, robotics, virtual and augmented reality, and vehicle and home automation. Henceforth, automatic hand localization and gesture classification have drawn substantial attention in the AI research community to establish a bridge for human-computer and human-human interaction.

This thesis predominantly focuses on accurate comprehension of hand gestures, which is quintessential for the proper operation of any hand gesture-based interface. Therefore, previous works related to hand segmentation, recognition, detection, and keypoint localization have been studied to develop a system that ensures smooth operation using hand gestures.

1.5.1 Hand Gesture Segmentation

In computer vision and human-robot interaction [17], hand segmentation finds a wide range of applications, requiring hand pose estimation [18] and hand gesture recognition [16]. It aims to label hand region pixels as foreground and other pixels as background. Much research has been performed using designer-specific features. Kawulok et al. [19] used spatial and texture-based features to model the skin region. They generated skin probability maps and applied linear discriminant analysis to obtain features discriminating the skin areas, subsequently detecting the skin regions upon spatial analysis. Khan et al. [20] analyzed the effect of color space and illumination on accurately detecting skin regions. Moreover, they studied the performance of nine skin modelling algorithms and found that the color constancy algorithm improves skin detection. An unsupervised approach is adopted by [21] to obtain features that differentiate an image's skin and non-skin areas using adaptive discriminative analysis. Although color and skin-based detection worked for hand segmentation, these are hand-crafted features and can introduce bias, resulting in a not so ideal hand mask generation. Therefore, the spotlight shifted to deep learning because it promised better and human-level detection without needing designer-specified features.

In [16], residual blocks and atrous spatial pyramid pooling blocks produced segmented hand masks, enabling a classification network to recognize hand gestures. Khan *et al.* [22] performed hand segmentation in egocentric videos using RefineNet, a deep learning model, and improved the results by employing conditional random fields. Cai *et al.* [23] used a Bayesian convolution neural network (CNN) to estimate model uncertainty and share hand shape information across different domains to enable generalization in hand segmentation. Yang *et al.* [24] proposed SPSNet, a novel architecture for hand segmentation that fuses temporal and tracking proposals in depth videos and reduces the complexity of hand pose estimation. Wang *et al.* [18] proposed a two-stage CNN. The first stage generated a hand mask while the second estimated hand joints. Tsai *et al.* [25] proposed a two-stage segmentation network, where the first stage produces a rough segmentation mask, which is improved by the second stage employing a distance method. The first stage is a simplified U-Net architecture followed by a refined block in the second stage,

and the authors named the complete architecture Refined Simple U-Net. In [26], the authors created a source dataset that has been style-adapted to look like the target dataset. Then, a segmentation model is trained on the style-adapted dataset and a reference model on the original dataset. The intersection of the masks generated by the two networks produces the pseudo labels. These pseudo labels are used to update the segmentation model again, which eventually generates the required segmentation hand masks.

Methods from related research areas can be adapted to hand segmentation. Wu *et al.* [27] proposed a joint pyramid upsampling technique for dilated convolution to reduce its complexity and memory requirements for semantic image segmentation. Fu *et al.* [28] proposed DANet, which uses two self-attention mechanisms to capture a scene's contextual information. They encode position and channel information, yielding a feature representation for scene segmentation. Ronneberger *et al.* [29] developed U-Net, an architecture to segment medical images with vastly fewer training samples. It consists of a contracting path that encodes context and an expanding path that localizes the object. An extension, Attention U-Net, was proposed by [30]. Its attention gating mechanism focuses on the object of interest, ignoring irrelevant areas. This mechanism was modified by [31] to include residual learning. Although these three U-Net architectures were designed for medical images, they have been adapted to semantic segmentation because of their exceptional performance. Sun *et al.* [32] used Gaussian dynamic convolution with a dynamic receptive field for fast single image segmentation. Baheti *et al.* [33] proposed an encoder-decoder architecture, named Eff-UNet, for semantic segmentation where the encoder is an EfficientNet and the decoder is taken from UNet. In [34], Mask-RCNN was used to segment the hands of a humanoid robot in ego-centric images. TransUnet [35] is a transformer-based segmentation model for medical image segmentation that combines the benefits of U-Net and transformers. Here a CNN-transformer-based encoder passes the encoded feature map to a CNN decoder, which performs gradual upsampling. As with U-Net, skip-connections from the encoder to the decoder enable precise segmentation results. However, this approach requires more transformer units, resulting in more training parameters and greater computational complexity.

1.5.2 Hand Gesture Recognition

Many earlier studies in hand gesture recognition required prior design knowledge. Various feature descriptors like Fourier descriptor (FD) [36], scale-invariant feature transform (SIFT) [37], and histogram of oriented gradients (HOG) [38] were utilized for gesture recognition. Harding and Ellis [36] presented a novel use of Fourier Descriptor techniques for hand gesture trajectory recognition. Normalized hand centroid coordinates and time steps were transformed into scale and translation-invariant frequency domain data, which were then classified using a Probabilistic Neural Network. A real-time hand gesture-based system for interacting with applications or video games was proposed in [37], which detected and tracked bare hands using skin detection and hand posture contour comparison. The paper extracted keypoints from images using SIFT, clustered with K-means, and converted into histogram vectors for SVM classification. Feng and Yuan [38] carried out hand gesture recognition by classifying HOG features using SVM. HOG features were considered due to their insensitivity to illumination variations and hand orientation. Also, template-matching methods such as dynamic time warping (DTW) [39] and state-space models like hidden Markov models (HMM) [40] and support vector machine (SVM) [37, 38] were commonly employed as gesture classifiers. In [39], a fast parallel algorithm for hand shape classification was proposed that demonstrated how parallelization reduced classification time and enabled efficient search of large hand gesture sets. It further showed that combining shape contexts with appearance-based methods improved classification accuracy. An HMM-based method for gesture recognition was proposed in [40], which incorporated a threshold model to handle non-gesture patterns by calculating likelihood thresholds and confirming matched gestures. The initially large threshold model was optimized by merging similar states using relative entropy.

However, a challenge with many of these approaches arose from the difficulty in matching a wide range of gestures performed by different individuals, and with the advent of deep neural networks, this issue has been addressed to a large extent. Deep learning's feature learning mechanism operates automatically across various levels of representation, enabling the computational model to capture intricate structures inherent in the data implicitly and not get stuck

with a specific feature of preference.

Dadashzadeh *et al.* [16] proposed a network that segments the hand using residual convolution blocks and atrous convolutions. The original input and the segmented mask form a two-way networks whose final layer predicts the output of the hand gesture. Dutta *et al.* [41] adopted a similar approach, that is, labeling 34 classes of hand gestures after segmenting the hand region using U-Net. However, the images contained uniform backgrounds. Bao *et al.* [42] recognized hand gestures without localizing the hand and achieved good results on images with a simple background. Their performance on images with complex backgrounds was average, and most importantly, they considered only seven classes. Chevtchenko *et al.* [43] proposed a three-way feature extraction scheme to obtain hand gesture recognition. They concatenated features obtained from a CNN with the original input image, a CNN with Gabor filter output as input, and hand-crafted Zernike moments, Hu moments, and contour features of the hand. This makes the entire network a bit messy, which can be avoided as the goal of deep neural networks is to avoid hand-crafted features. In [44], the authors proposed a multi-scale region-based fully convolution network that performs hand detection and classification. They used different filter sizes to capture global and local features and achieve multi-scale feature extraction. This also highlights the usefulness of precise localization and classification for accurate hand gesture recognition in complex surroundings.

1.5.3 Hand Gesture Detection

Hand gesture detection is a form of object detection. Its development has been promoted by the Pascal VOC [45] and COCO [46] challenges. Pioneering object detection algorithms include region-based convolution neural networks (e.g., RCNN [47], Fast RCNN [48], Faster RCNN [49]) and you-only-look-once CNNs (e.g., YOLO [50], YOLOv2 [51], YOLOv3 [52]), SSD [53], RetinaNet [54]. RCNNs comprise two-stage deep architectures that generate region proposals. These region proposals undergo further post-processing to arrive at the intended results, that is, classification scores and bounding box coordinates. YOLO, single shot multi-box detector (SSD), and RetinaNet have only a single stage, which makes them much faster than RCNNs. Using a single deep neural network, they generate anchor boxes for feature maps

and predict classification scores and bounding box coordinates without the need to generate region proposals. However, they require post-processing steps, such as non-maximal suppression (NMS), hard negative mining, or both.

To eliminate the need for techniques requiring prior knowledge of regions, anchor generation, NMS, mining, etc., we explored the use of transformers. Transformers have already shown success in natural language processing [55] and are being tested in object detection. A detection transformer (DETR) [56] employs an end-to-end approach to predict objects using a bipartite matching procedure. Its simplified pipeline streamlines detection by eliminating anchor generation and non-maximal suppression. It leverages the transformer to capture long-range dependencies that help predict the object's class label and bounding box accurately. This approach is also useful for the localization and classification of hand gestures.

Bose *et al.* [57] experimented with two deep architectures, Faster RCNN and SSD, and integrated them with the Inception V2 module for precise and efficient real-time hand gestures recognition. The Faster RCNN model performed better than the SSD model. However, as a two-stage detector, Faster RCNN requires more computation than a single-stage network. In [58], a two-step approach to hand detection and recognition was proposed. The input image is fed to an object detector network to identify the region of interest, i.e., the hand, and subsequently to a CNN to classify the gesture. Bao *et al.* [42] designed a CNN to classify hand gestures without using a segmented mask or detection annotations to remove irrelevant regions. Dadashzadeh *et al.* [16] incorporated hand segmentation to detect the region of interest and integrated the mask generated from the RGB image to recognize hand gestures. A two-step system was proposed in [59], where a YOLOv3 network localized the hand region, and the gesture was recognized by a VGG-16 [60] network. In [44], a region-based multiscale fully connected deep neural architecture was developed, which simultaneously performed hand classification and bounding box regression to detect the hand in vehicles and outdoors. Yadav et al. [61] combined detection and tracking for bare-hand localization to reduce computational time and achieve near-real-time performance. The detection module constituted HandSNet and YOLO-H, customized from SqueezeNet network, and the tracking module used Kalman filter and point tracker. Gao *et*

al. [62] employed 3D hand pose estimation to recognize dynamic hand gestures. They combined hand keypoints information with depth and RGB data and classified the data using a deep neural network. Yuan *et al.* [63] proposed a deep convolution network that fused features derived from multiple sensors. They also developed a novel data glove to obtain motion information for the arm and knuckles. However, the glove impedes gesturing and induces fatigue. Moreover, a data glove adds to the system's cost and may not be readily available.

1.5.4 Hand Keypoint Detection

Hand keypoint detection has myriad applications in human-computer interaction, robotics, biomechanics, biomedicine, and other fields. For their smooth operation, these applications require the accurate detection and precise localization of hand keypoints [64]. Despite extensive research, challenges in this domain persist, and an effective generalized method to address these challenges remains to be developed. Hand keypoint detection can be performed using various input modalities, including depth cameras and leap motion controllers. However, they are costly and affected by ambient conditions [65]. Therefore, color cameras were considered as the medium for capturing hand images.

Ovur *et al.* [66] developed a system to track palm poses using two leap motion controllers to overcome occlusion. Their self-calibrating algorithm allows flexible sensor placement. Although their system operates in real-time without pre-processing, it may only be effective in specific environments. In particular, the sensors' inability to consistently and accurately recognize the hands might limit their use for productivity-related tasks. Bhuyan *et al.* [67] proposed a robust method for fingertip detection and hand pose recognition. A skeletal hand model was created using fingertip and metacarpophalangeal joint positions where the hand features were modelled as 3D Gaussian distributions, resulting in high recognition rates. Kourbane and Genc [68] proposed a novel method of extracting local and global features by processing a full scene color image and its corresponding cropped hand image in parallel. These dual-perspective features were passed through separate graph convolution networks to estimate 2D keypoints, and then combined to estimate the 3D hand pose. This method, however, had reduced performance for uniform backgrounds. Zhang *et al.* [69] proposed a hierarchical approach to hand pose estima-

tion with hand segmentation as the pre-preprocessing step. Pose estimations for the fingers and palm were performed separately and combined for the final hand pose estimate. A 3D hand pose estimation approach for depth image was proposed that used a feedback loop of deep networks to correct errors made by a convolutional neural network in predicting the 3D pose [70]. This approach eliminated the need for fitting a 3D model to the input data, avoiding the complexities of designing a fitting function and algorithm. Simon *et al.* [71] presented multiview bootstrapping, a method for training fine-grained detectors for occlusion-prone keypoints like hand joints using a multi-camera system. Initially, a keypoint detector generated noisy labels from multiple views, which were triangulated in 3D or marked as outliers. These refined labels were then used as new training data to iteratively improve the detector. Yang *et al.* [72] modified the Nonparametric Structure Regularization Machine [73] by replacing its backbone with an HRNet integrated with a shuffle attention network [74] to estimate the 2D hand pose. In [65], a three-stage approach was proposed. The first stage extracted features using a UNet with a pretrained ResNet-34 encoder. The second stage generated 2D heatmaps, converted to 2D keypoint coordinates using a latent heatmap representation. The third stage estimated the 3D hand pose using a tree structure to predict the hand bones. A lightweight hand pose estimation network was developed for embedded systems [75]. It regressed a bounding box around the hand to crop the region of interest within which it performed 2D hand keypoint localization. A two-stage approach involving the generation of hand masks followed by pose estimation was proposed in [18]. Hand masks were obtained using a fully convolutional network and then hand poses were estimated using a cascaded approach of concatenating the intermediate heatmap with the final heatmaps. Mueller et al. [76] also proposed a two-stage network for hand pose estimation in RGB-D images where they extracted the hand region by determining the hand center location. Subsequently, the 3D hand joints are regressed from the hand region and refined using a kinematic pose fitting technique. Santavas *et al.* [77] proposed a single-stage 2D hand pose estimation network that uses a DenseNet backbone and an attention block employing a self-attention mechanism [55].

1.6 Motivation

From the literature survey presented in this thesis, it is evident that a significant amount of work is needed for efficiently decoding or recognizing different hand gestures under challenging imaging conditions. Additionally, combining multimodal hand gesture-based information to operate interfaces is another important issue. Accordingly, this thesis looks into several aspects concerned with hand segmentations, hand gesture recogniton, hand gesture detection, and hand keypoints localization and aims at developing suitable algorithms that take care of some of the limitations of the existing methods. The motivation behind this research work are given below:

- Background clutter and the presence of skin regions other than the hand include spurious information in the hand recognition task. Though existing literature proposes methods to tackle these challenges, arm regions or the faces of other persons in the background tend to interfere with the feature engineering process. Therefore, a pre-processing step that reduces the impact of the background and focuses only on the hand region is essential.
- Current hand gesture recognition methods struggle to achieve high precision, reliability, and occlusion resilience, partly resulting from sensor limitations. Thus, a robust hand gesture recognition model needs to be developed that generalizes the hand gesture recognition task across different challenging imaging conditions. This is also the key to an error-free and user-friendly operation of a hand-controlled interface.
- Most works generate recognition results at the expense of memory and computational complexities. Therefore, a robust hand gesture system that performs simultaneous gesture localization and classification is required. Also, for certain tasks, information related to all pixels belonging to the hand region is not required. Simply localizing the hand is sufficient. In such cases, hand gesture detection plays a crucial role in reducing the workload and generating effective results.
- Traditional interaction devices, such as keyboard and mouse, are not primarily preferred by persons with hand injuries, disabilities or the elderly. Moreover, installing these devices in outdoor or indoor environments, like hospitals, rehabilitation centers, food stalls,

etc., is not conducive. Hence, a contactless way of human-computer interaction operated by hand has a very bright prospect. For instance, hand rehabilitation done at home through interaction with computers without the need to visit medical centers will help the patients perform the exercises without the traffic and travel hassles.

- The hand's intricate morphology, especially the arrangement of the fingers, affects the accurate localization of hand keypoints. Hand keypoints detection methods exist in the literature, but they are not precise for every imaging condition. Moreover, hand keypoints are quintessential for hand pose estimation and performing specific operations in human-computer interfaces. Therefore, due to its importance in hand gesture-based application, a robust keypoint detection method is needed.

1.7 Objectives

Motivated by the points discussed in the previous section, the objectives of the thesis are defined as follows:

- ⇨ To develop a hand segmentation model that removes the complex background. This step will be a pre-processing step so the focus will be on developing a light-weight segmentation model.
- ⇨ To develop an effective hand gesture recognition model that produces consistent hand gesture recognition results despite the presence of illumination variations, varying hand shapes and sizes, unseen surroundings, etc.
- ⇨ To develop a precise and efficient hand detection model capable of performing localization and classification of hand gestures simultaneously, suitable for online (near to real-time, but with an acceptable amount of latency) applications.
- ⇨ To develop an effective hand keypoint detection model, despite the presence of hand morphological challenges, which helps perform specific tasks in hand gesture controlled interfaces.
- ⇨ To develop interaction interfaces controlled by hand gestures to help patients with tasks such as hand rehabilitation and interaction with medical staff.

1.8 Organization

To accomplish the problem definitions, the thesis is organized into six chapters. A pictorial representation of the workflow of the thesis is shown in Fir. 1.5. The content of the chapters are summarized as follows.

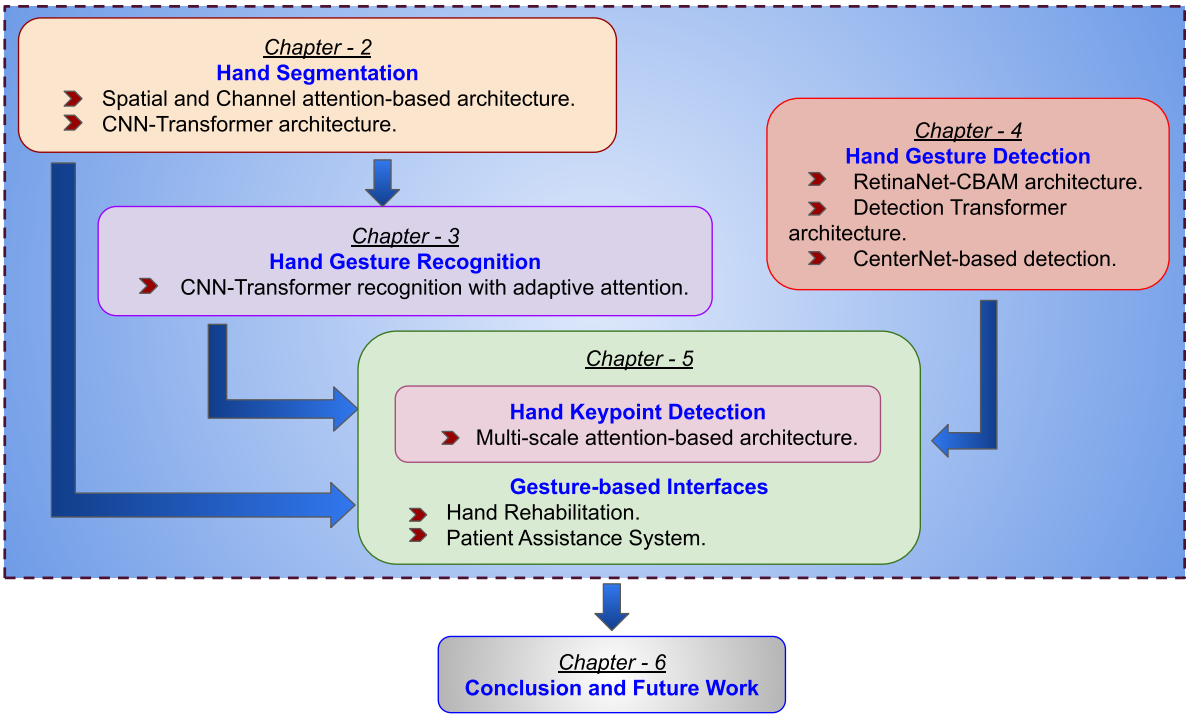

Fig. 1.5: An illustration of the thesis organization.

Chapter 1 introduces the thesis, briefly explaining HCI, hand gesture recognition, and datasets used in the work. This chapter also discusses the previous works, the motivation behind the thesis, and the problem definitions.

Chapter 2 describes two segmentation models to delineate the hand region from the background. It introduces two novel attention blocks in an encoder-decoder network and a novel loss function to achieve a precise hand mask. Moreover, another network is proposed that utilizes the benefits of a convolution neural network and transformer. It also addresses the quadratic dependency of transformers by linearizing it.

Chapter 3 introduces an adaptive kernel channel attention layer, which aids the proposed CNN-transformer architecture for hand gesture recognition in prioritizing features related to the

hand region. The input to this architecture consists of segmented hand masks obtained using a model discussed in the preceding chapter.

Chapter 4 explores hand gesture detection, which enables both the simultaneous localization of the hand and the recognition of the gesture. It proposes three approaches: one based on anchors and two anchor-less methods for detection. Additionally, it delves into the rationale behind selecting one method from the three and employing it to operate an interface.

Chapter 5 proposes a robust hand keypoint detection technique to localize hand joints despite the presence of challenges, such as finger proximity, self-occlusion, and complex backgrounds. This is accomplished through the utilization of a multi-scale attention block, which emphasizes pertinent features to ensure accurate localization of keypoints. It further describes the operation of two interfaces that are operated by hand gestures as an application of the proposed hand gesture and keypoint detection methods. One interface is tailored to aid patients recuperating from hand injuries, stiffness, paralysis, and similar conditions. It recommends rehabilitation exercises and evaluates performance, assigning a score based on the execution of the exercises. This feature enables patients to monitor their progress effectively. The other interface facilitates communication between patients and medical staff in a hospital setting. It enables patients to convey messages directly to the appropriate healthcare provider who can address their requests.

Chapter 6 concludes the thesis and highlights the major contributions of the thesis. It also discusses the possible future direction of the work.

2

Hand Segmentation

Contents

2.1 Introduction

Hand segmentation is a fundamental task in computer vision with applications ranging from human-computer interaction to gesture recognition. Hand segmentation primarily assigns the label '1' to the pixels belonging to the hand region in an image and the label '0' to the rest of the regions. This ensures the hand is the foreground and the non-hand regions are the background. The capability to delineate the hand shape is a crucial factor in enhancing the effectiveness of hand gesture recognition. However, there are challenges to accurate hand segmentation due to complex backgrounds, variations in ambient lighting, and occlusion [7, 41]. The complex background may contain objects or regions with colors similar to human skin. The model may learn characteristics of these regions as they resemble the hand color, which are mostly spurious features. Other objects in the background may also affect feature learning as the convolution neural network kernel may capture certain noisy characteristics while translating through the image. The variations in ambient lighting may also hamper accurate hand segmentation due to the change in color (mainly, luminance) features of the hand region. Similarly, occlusion resulting from own hand or other objects may distort the structure of the hand, confusing the segmentation model about the boundaries of the hand. This causes inaccuracies in delineating hand boundary.

Thus, a method is required to address these limitations, that can produce precise hand segmentation masks. This work proposes two novel methods to segment the hand region in these challenging conditions as described in the following sections. Both the method utilizes attention mechanism that helps the networks to learn features useful for the objective and suppress those that convey irrelevant information. The method that best generalizes the hand segmentation task inspite of challenging conditions and variations in hand shapes and size is considered for the next objective.

2.1.1 Intuition and Objective:

Hands gestures play an essential role in daily communication among people. Understanding these gestures using a vision-based method requires accurate localization of the hands. How-

ever, the performance of vision-based methods degrades under real-world conditions, such as background clutter, occlusion, variations in illumination, and overlapping skin regions like the hands and face [7]. Therefore, segmenting the hand region from the background would help overcome these hurdles as there would not be any adjoining regions in the image to affect gesture recognition. However, precise segmentation is also affected by these challenges.

Thus, this work aims to develop a robust hand segmentation to counter the challenges and determine a precise hand mask. This precise mask also finds its use in recognition tasks assisting recognition networks to capture structural and spatial features specific to the hand region.

2.2 Method 1: A CNN architecture with spatial and channel attentions

2.2.1 Overview

Segmentation can be modeled as a pixel-wise binary classification task with at least two classes—foreground and background. A deep neural network uses convolution layers to encode the features of the two classes. In a CNN, each receptive field of the convolution layers is sensitive to local features in the image. However, it becomes harder for a CNN to detect patterns at increasing spatial and temporal scales. With each layer, within-class pattern differences reduce the network's recognition accuracy. To overcome this, information from distal pixels of the same class must be integrated. Therefore, we propose an architecture that encodes distal pixel dependencies using a self-attention mechanism for spatial and channel dimensions, as shown in Fig. 2.1.

The spatial feature attention block (SFAB) implements spatial self-attention, and the channel feature attention block (CFAB) implements channel self-attention. SFAB integrates the spatial correlations of different pixels to obtain spatial attention maps that focus on regions likely to contain the object. CFAB complements SFAB by generating attention maps that focus on the object's class, capturing correlations among the channels of the input feature map. The proposed architecture is an encoder-decoder network with the encoder containing four convolution blocks, two SFAB and CFAB blocks, and a depthwise convolution layer through which the output of the encoder is propagated to the decoder. The decoder also contains four convolution

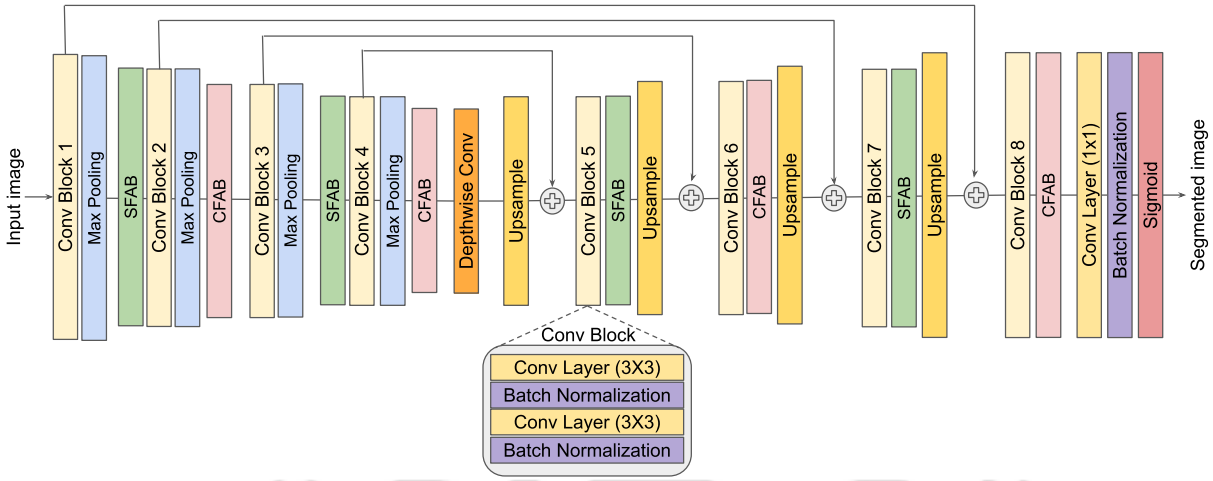

Fig. 2.1: The block diagram of the proposed segmentation method.

blocks and two SFAB and CFAB blocks. The output of the decoder passes through a sequential arrangement of a convolution layer with a 1×1 filter size, a batch normalization layer, and sigmoid activation to obtain the segmented mask.

The encoder's sequential ordering is SFAB–convolution block–max-pooling–CFAB. The intuition behind this ordering is that the SFAB attends to the object's regions, and this feature map is passed to the convolution block. The convolution block thus learns the features belonging to the foreground region, namely, those of the hand. A max-pooling layer is placed after the convolution block, followed by a CFAB. This layer captures distinctive features of each class to improve channel-specific attention [78]. A CFAB follows to capture channel-specific details and accentuates class-dependent features. The experimental analysis of this arrangement is detailed in subsection experiments and results. A pair of convolution and batch normalization layers comprise the convolution block, as shown in Fig. 2.1. Each convolution layer has a 3×3 filter size. The batch normalization layer standardizes the input feature map to reduce training time.

The input image propagates through a convolution block and a max-pooling layer to the first SFAB–convolution block–max-pooling–CFAB sequence. The resulting attention feature maps pass through another convolution block and max-pooling layer, followed by the second SFAB–convolution block–max-pooling–CFAB sequence. Before passing the resultant feature maps to the decoder, a depthwise convolution layer encodes it. The advantage of a depthwise

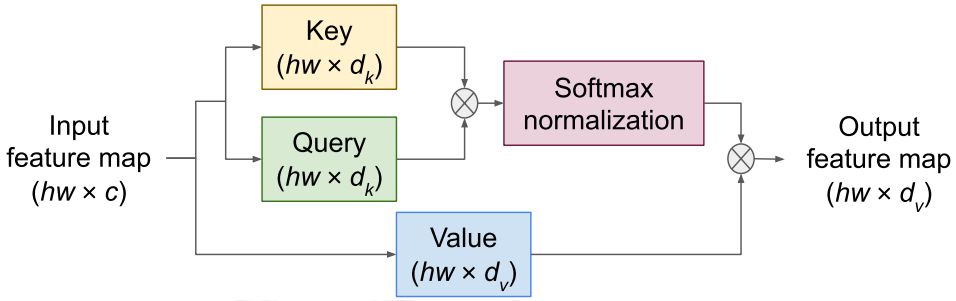

Fig. 2.2: Block representation of the self-attention mechanism.

convolution layer over a standard convolution layer is that it introduces much fewer training parameters to the deep neural network. The four max-pooling layers reduce the spatial dimension of the encoder's output feature maps to $\frac{1}{16}$th the input image. Therefore, the decoder upsamples the feature maps by a factor of 2 and concatenates the upsampled feature maps with the output of the previous convolution block, i.e., Conv Block 4. This skip connection incorporates details from the encoder layers into the decoder layers to arrive at a more refined segmentation mask at the decoder's output. The concatenated output is passed through a convolution block–SFAB–upsampling layer sequence. The resulting feature map is added with the output of Conv Block 3 and passed to a convolution block–CFAB–upsampling layer sequence. The output is then concatenated with Conv Block 2's output, passed through a convolution block–SFAB–upsampling layer sequence, and subsequently concatenated with Conv Block 1's output. Next, the propagated feature map is passed through the final convolution block, the CFAB, a sequence of convolution layer of filter size 1×1, a batch normalization layer, and a sigmoid activation layer to arrive at the final output, that is, the segmentation mask.

2.2.2 Self-attention mechanism

The self-attention mechanism (SAM) models long-range dependencies [55]. Initially designed for natural language processing, SAMs have been used extensively in computer vision to capture an image's global context [28, 79]. The SAM incorporates the global and local context by using convolution. A pictorial representation of a SAM is shown in Fig. 2.2.

A feature map $F \in \mathbb{R}^{h \times w \times c}$ is reshaped into $X \in \mathbb{R}^{hw \times c}$ and passed through the SAM to obtain the output attention map. Here, h, w, and c represent the feature map's height, width, and

channel. For the m^{th} position of X, $\exists$ a vector $\mathbf{x}_m \in \mathbb{R}^c$. Then, three vectors, namely, the query vector $\mathbf{x}_{qm} \in \mathbb{R}^{d_k}$, key vector $\mathbf{x}_{km} \in \mathbb{R}^{d_k}$, and value vector $\mathbf{x}_{vm} \in \mathbb{R}^{d_v}$, are constructed from $\mathbf{x}_m$. The SAM estimates the relevance of the n^{th} position to the m^{th} position, i.e., the current position, to improve the current feature vector's encoding. Thus, the correlation between the m^{th} query vector and the n^{th} key vector is obtained, i.e., ${\mathbf{x}_{qm}}^{\text{T}}\mathbf{x}_{kn}$ where $n \in 1, 2, ..., hw$. This dot product results in a score, which is divided by $\sqrt{d_k}$ to stabilize the gradients and then normalized by the softmax function to ensure positive scores. This score is represented as

$$score_{mn} = \frac{exp(\frac{{\mathbf{x}_{qm}}^{\text{T}}\mathbf{x}_{kn}}{\sqrt{d_k}})}{\sum_{j=1}^{hw} exp(\frac{{\mathbf{x}_{qm}}^{\text{T}}\mathbf{x}_{kj}}{\sqrt{d_k}})}. \tag{2.1}$$

It is the correlation between the position pairs m and n. The score is multiplied with the m^{th} value vector $\mathbf{x}_{vm}$ to obtain the weighted value vector at the m^{th} position. The weighted value vectors for other positions are obtained in the same way. Relevant positions are multiplied by a large score and are thus brought into focus, while irrelevant positions are multiplied by a small score and are thus ignored. Finally, the weighted value vectors are added to obtain a vector $\mathbf{x}_{\text{sum}_m}$, the SAM's output for the m^{th} position, which is expressed as

$$\mathbf{x}_{\text{sum}_m} = \sum_{j=1}^{hw} score_{mj} \times \mathbf{x}_{vj}. \tag{2.2}$$

For all the hw positions, we create the query matrix $\mathbf{M}_q \in \mathbb{R}^{hw \times d_k}$, key matrix $\mathbf{M}_k \in \mathbb{R}^{hw \times d_k}$, and value matrix $\mathbf{M}_v \in \mathbb{R}^{hw \times d_v}$. Thus, the SAM's attention map is given by

$$\text{Output feature map} = softmax\left(\frac{\mathbf{M}_q {\mathbf{M}_k}^{\text{T}}}{\sqrt{d_k}}\right)\mathbf{M}_v. \tag{2.3}$$

For self-attention $d_k = d_v$.

2.2.3 SFAB and CFAB

Although the SAM's ability to encode long-range dependencies, its memory usage and high computational complexity are drawbacks. Determining position-to-position correlation is $O((hw)^2)$ for memory and $O(d_k(hw)^2)$ for computation. Under memory constraints, the SAM

can cause an out-of-memory error. Therefore, drawing inspiration from [79], we propose two novel efficient attention blocks that generate attention maps along spatial and channel dimensions. These blocks, namely SFAB and CFAB, reduce memory use from quadratic to linear.

In this case, the queries $\mathbf{M}_q$, keys $\mathbf{M}_k$, and values $\mathbf{M}_v$ also exist. However, instead of calculating the correlations between positions, each channel of keys is considered a feature map and is multiplied by the values. This results in a global feature map weighting every position of the input feature map and highlighting its class-specific features. Now the queries at each position aggregate the global feature maps and generate the final attention map that attends to the object's class and position.

The SFAB uses this efficient way of calculating attention along the spatial dimension to focus on the location of the foreground hand region. At the outset, it arranges the queries, keys, and values into groups to introduce a parallel and independent way of attending to different representations of the feature map. Thus, there are $(\#channels/\#divisions)$ groups. The queries, keys, and values of the first group are represented as $\mathbf{M}_q^1$, $\mathbf{M}_k^1$, $\mathbf{M}_v^1 \in \mathbb{R}^{hw \times \frac{d_k}{\#divisions}}$, respectively. $\#channels$ and $\#divisions$ represent the number of channels and divisions, respectively. The associative property of matrix multiplication ensures $(\mathbf{M}_q\mathbf{M}_k{}^{\mathrm{T}})\mathbf{M}_v = \mathbf{M}_q(\mathbf{M}_k{}^{\mathrm{T}}\mathbf{M}_v)$. However, [79] state that, to realize this matrix associativity with the softmax function, two softmax functions are used—one for the queries and the other for the keys. These two softmax functions resemble the single softmax function of the SAM and approximate the required normalization. Thus, the attention expression for the first group is given by

$$attention^1 = softmax_q(\mathbf{M}_q^1)(softmax_k(\mathbf{M}_k^1)^{\mathrm{T}}\mathbf{M}_v^1). \tag{2.4}$$

Similarly, $attention^1, ..., attention^{\#divisions}$ are appended together and fed to a convolution layer of filter size 1×1 to obtain the final spatial attention map of dimension $hw \times c$, as shown in (2.5). The SFAB is illustrated in Fig. 2.3.

$$\text{SFAB attention map} = conv([attention^1, attention^2, ..., attention^{\#divisions}]). \tag{2.5}$$

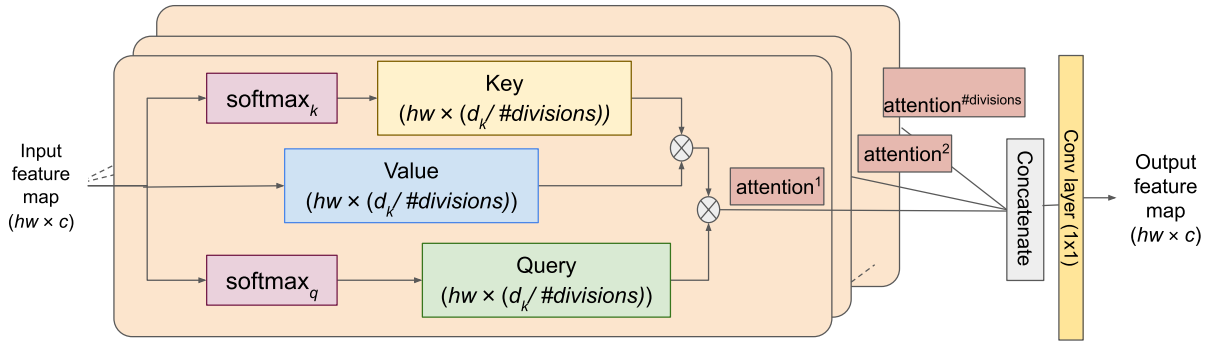

Fig. 2.3: Block diagram of the spatial feature attention block (SFAB).

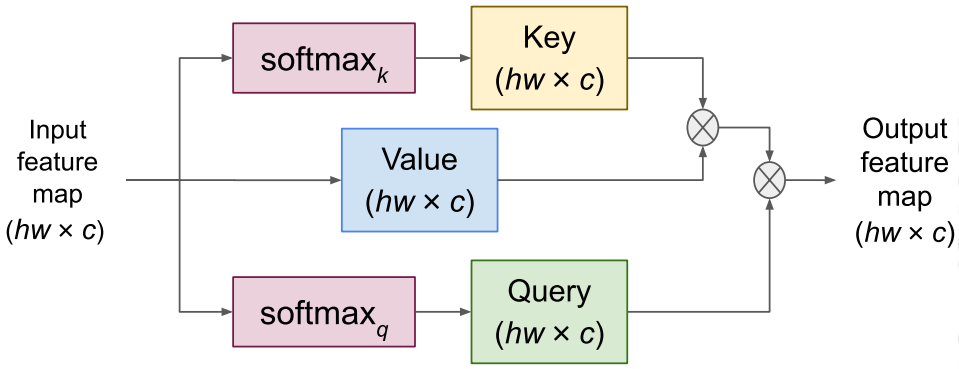

Fig. 2.4: Block diagram of the channel feature attention block (CFAB).

The CFAB has matrix multiplications like the SFAB but does not have divisions along channel dimensions to form groups and uses the entire key, query, and value matrices. Here, the channel relationship is encoded as attention maps, highlighting the semantics to segment the foreground hand from the background. Alternatively, it highlights the correlation between the m^{th} and n^{th} channels to attend to class-specific features. The CFAB is illustrated in Fig. 2.4, and the final channel attention map is shown in (2.6).

CFAB attention map =

$$softmax_q(\mathbf{M}_q)(softmax_k(\mathbf{M}_k)^{\mathrm{T}}\mathbf{M}_v), \tag{2.6}$$

where the attention map is of dimension $hw \times c$ and $\mathbf{M}_q, \mathbf{M}_k, \mathbf{M}_v \in \mathbb{R}^{hw \times c}$, respectively.

This calculation method reduces memory complexity from $O((hw)^2)$ to $O(d_k hw + d_k^2)$ and computational complexity from $O(d_k(hw)^2)$ to $O((d_k)^2 hw)$. This avoids a quadratic increase in complexity with the feature map's spatial dimensions, and lets the developer set the value of c.

2.2.4 Loss function

During the training of a deep learning model, we minimize a loss function to learn the model's optimal weights for a given task. Therefore, we propose a novel composite loss function that integrates hand regions, segments the hand's shape, ensures boundary smoothness and continuity, and corrects for class imbalance. The loss function has these three components:

2.2.4.1 Binary focal loss

Segmentation is a pixel-wise classification task assigning pixels to the foreground or background. For hand segmentation, the foreground, i.e., the hand, has fewer pixels than the background because it covers a small area of the image. Hence, their proportion is imbalanced. Although binary cross entropy (BCE) has been used for segmentation, it is biased towards the background class because it has more pixels. Therefore, we used binary focal loss (BFL) [54]. BFL reduces the contribution of background pixels that are easy to classify while balancing the contribution of foreground and background pixels that are hard to classify. BFL develops on BCE, which is given by

$$BCE(\hat{y}, y) = \begin{cases} -log(\hat{y}), & y = 1 \\ -log(1-\hat{y}), & otherwise \end{cases}, \tag{2.7}$$

where $\hat{y}$ denotes the predicted probability and y denotes the true label. BFL uses a modulating factor β to reduce the contribution of easy-to-classify pixels to the loss function and amplify the contribution of hard-to-classify pixels, which would otherwise be low. The balancing factor α corrects for the imbalance in the foreground and background pixels. Thus, the BFL is defined as

$$BFL(\hat{y}, y) = \begin{cases} -\alpha(1-\hat{y})^{\beta} log(\hat{y}), & y = 1 \\ -(1-\alpha)(\hat{y})^{\beta} log(1-\hat{y}), & otherwise \end{cases}. \tag{2.8}$$

As the probability of correct prediction tends to 1, the scaling factor tends to 0, making the function resistant to class imbalance.

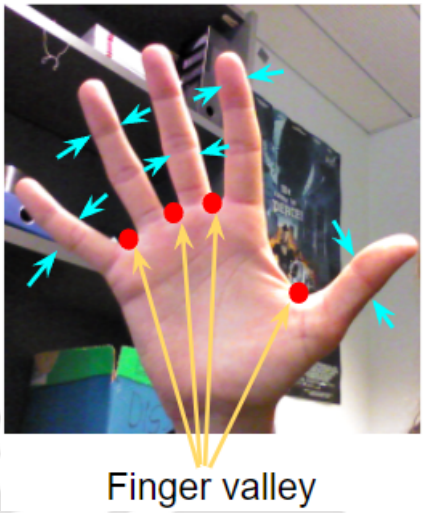

Fig. 2.5: An image showing the finger valley and the slender part of the hand (indicated by cyan arrows →←).

2.2.4.2 Smoothing loss

The hand is a deformable object, and the hand region's boundary in an image plays a pivotal role in the hand's accurate segmentation. The fingers need attention to obtain an accurate segmentation mask because of their slender shape and constricted regions relative to the palm. For instance, the segmentation mask may not be smooth around finger valleys, as shown in Fig. 2.5, owing to other skin regions, constricted regions, or background clutter. This may result in a coarse boundary in the finger region with some breaks. If discontinuities occur, the hand's boundary should be smoothed by reducing the energy along it to maintain continuity. Thus, we define a loss function to smooth the boundary of the segmented mask, given by

$$\ell_{\text{smooth}} = \sum_{i,j \in \hat{y}} \sqrt{|(\nabla \hat{y}_{u_{ij}})^2 + (\nabla \hat{y}_{v_{ij}})^2|}, \tag{2.9}$$

where $u_{i,j}$ and $v_{i,j}$ denote the horizontal and vertical directions of the image coordinates, respectively. ∇ is the gradient operator.

2.2.4.3 Dice loss with skeletal information

The Dice coefficient estimates the similarity between the predicted mask and ground truth. However, it ignores the hand's shape and continuity. This information is useful because the fingers are slender and often occlude each other, which can cause a segmented region to become disconnected from the main hand segmentation mask. Thus, information about the skeleton

Table 2.1: Training details

Environment	Python 3.6
GPU	Nvidia Tesla P100
Optimizer	Adam Optimizer
Learning rate	0.0001
α and β (for BFL)	0.25 and 2 (from [54])
Batch size	8
Epochs	20
Division of data	70% training, 15% validation, 15% testing
$d_k = d_v$	[16, 64] for encoder [128, 32] for decoder

of the hand would help the predicted hand regions stay connected despite occlusion, narrow regions, or background clutter. We used the centerline Dice [80] with the Dice coefficient to obtain the hand's connectivity and structure. The centerline Dice calculates the skeletons of the predicted mask and the ground truth, denoted by S_P and S_T, respectively. It attends to the intersection of the skeletons with the masks and determines the part of the skeletons that lie in the masks. The skeletal information holds the predicted mask together and iteratively improves it. Thus, the Dice loss with the skeletal information can be defined as

$$\ell_{\text{skdice}} = \kappa_1 \left(1 - \frac{2|\hat{y} \odot y|}{|\hat{y}| + |y| + \epsilon}\right) + \kappa_2 \left(1 - \frac{2(\frac{|S_P \odot y|}{|S_P| + \epsilon} \times \frac{|S_T \odot \hat{y}|}{|S_T| + \epsilon})}{\frac{|S_P \odot y|}{|S_P| + \epsilon} + \frac{|S_T \odot \hat{y}|}{|S_T| + \epsilon} + \epsilon}\right), \tag{2.10}$$

where κ_1 and κ_2 are constants set to 0.5. Additionally, $\odot$ refers to element-wise multiplication, and $|.|$ refers to the cardinality of the set.

Now, given the three loss functions defined above, we formulate the novel composite loss function given by

$$\ell_{\text{total}} = BFL + \ell_{\text{smooth}} + \ell_{\text{skdice}}. \tag{2.11}$$

2.2.5 Experiments and Results

This section describes experiments to evaluate the proposed model, the datasets, and the quantitative and qualitative results. The algorithm is implemented in Python, and the model is trained using an Nvidia Tesla P100 GPU. The training details are shown in Table 2.1. A uniform image size is chosen for the datasets. The learning rate, batch size, epochs, and d_k are determined using randomized search [81], and the α and β values are retained from [54]. We initialize a random number generator with a fixed seed to set the initial weights to ensure the

results can be reproduced. Also, we use a uniform weight initialization scheme, i.e., Xavier uniform initializer, to maintain near identical variances of its weight gradients across model layers. To quantify model performance, we adopt the F1-score given by

$$F1 - score = \frac{2}{|cls|} \frac{\sum_{i=1}^{|cls|} \frac{\mathcal{A}_{ii}}{\sum_{j=1}^{|cls|} \mathcal{A}_{ji}} \times \sum_{i=1}^{|cls|} \frac{\mathcal{A}_{ii}}{\sum_{j=1}^{|cls|} \mathcal{A}_{ij}}}{\sum_{i=1}^{|cls|} \frac{\mathcal{A}_{ii}}{\sum_{j=1}^{|cls|} \mathcal{A}_{ji}} + \sum_{i=1}^{|cls|} \frac{\mathcal{A}_{ii}}{\sum_{j=1}^{|cls|} \mathcal{A}_{ij}}}, \tag{2.12}$$

and the mean Intersection over Union (mIoU) given by

$$mIoU = \frac{1}{|cls|} \sum_{i=1}^{|cls|} \frac{\mathcal{A}_{ii}}{\sum_{j=1}^{|cls|} \mathcal{A}_{ij} + \sum_{j=1}^{|cls|} \mathcal{A}_{ji} - \mathcal{A}_{ii}}, \tag{2.13}$$

where $\mathcal{A}$ denotes the confusion matrix, and $|cls|$ denotes the number of classes. Average precision is given by $\frac{1}{|cls|} \sum_i \frac{\mathcal{A}_{ii}}{\sum_j \mathcal{A}_{ji}}$, and average recall by $\frac{1}{|cls|} \sum_i \frac{\mathcal{A}_{ii}}{\sum_j \mathcal{A}_{ij}}$.

2.2.5.1 Quantitative Assessment

The SFAB attends to the object's position in the input image, and the CFAB to its class (i.e., foreground or background). The blocks can be arranged sequentially or in parallel to obtain optimal encoder-decoder performance. Thus, we consider six arrangements: No SFAB or CFAB, Only SFAB, Only CFAB, SFAB–CFAB, CFAB–SFAB, SFAB and CFAB in parallel, and SFAB–convolution block–CFAB. Table 2.2 lists the architecture's performance for these arrangements. Using attention blocks gives better results than using no blocks or only one block. The blocks achieve a better F1-score when arranged in series than in parallel. Moreover, when the SFAB is placed ahead of CFAB, performance improves. Including a convolution block

Table 2.2: Performance comparison for different arrangements of the SFAB and CFAB in the architecture

Sequence	F1-score (%)	Inference time (ms)
No SFAB or CFAB	95.03	16.27
Only SFAB	95.45	16.15
Only CFAB	95.19	16.35
SFAB → CFAB	97.03	16.54
CFAB → SFAB	96.49	17.05
SFAB || CFAB	96.35	16
SFAB → Conv → CFAB	**97.33**	**15.03**

Table 2.3: Performance of the model for different numbers of groups created from the SFAB's input feature map

Number of groups	F1-score (%)	Inference (ms)
1 group	96.47	17.8
2 groups	**97.33**	15.03
3 groups	96.20	**14.4**
4 groups	96.01	14.9
5 groups	95.86	14.8

between SFAB and CFAB increases performance by 0.3%. A max-pooling layer was tested in SFAB–Conv–CFAB because it captures class-specific features. Similarly, an upsampling layer was included in the decoder. However, its inclusion had little impact on performance.

Table 2.3 presents the effect of dividing the SFAB into varying numbers of groups. The F1-score was highest for two groups and then decreased for more groups, while the inference time was lowest for three groups. To maximize accuracy, we divided the incoming feature map into two groups, resulting in an F1-score of 97.33% with an inference time of 15.03 ms. We opted for the best F1-score, giving more weight to segmentation mask accuracy than speed, forfeiting a 0.63 ms decrease in inference time.

Moreover, the different components of the proposed loss functions contributed to achieving a state-of-the-art result. Table 2.4 lists the model's performance for different loss functions. Because BFL performed better than BCE, the contributions of ℓ_{smooth} and ℓ_{skdice} were tested for BFL. Further experimentation revealed that, although ℓ_{smooth} and ℓ_{skdice} contributed to a good F1-score, the combination of BFL, ℓ_{smooth}, and ℓ_{skdice} resulted in even better performance. The proposed model's performance was validated by comparing it with a few baseline architectures. The performance of encoder-decoder models with attention, such as Attention U-Net [30] and RA-UNet [31] surpassed the performance of simple encoder-decoder models, such as U-Net

Table 2.4: Performance of the proposed model with different loss functions

Losses	F1-score (%)
BCE	96.49
BFL	96.93
BFL + ℓ_{smooth}	97.10
BFL + ℓ_{skdice}	97.25
BFL + ℓ_{smooth} + ℓ_{skdice}	**97.33**

Table 2.5: Comparison of different baseline models with the proposed model

Methods	F1-score (%)	mIoU (%)	Inference time (ms)
U-Net [29]	96.7	81.2	25
Attention U-Net [30]	96.8	85.5	29
RA-UNet [31]	96.9	84.9	32
FCN-8s [82]	95.5	80.3	63
CBAM [78]	96.3	82.9	38
DANet [28]	96.4	83.6	35
Proposed	**97.3**	**89**	**15**

Table 2.6: Comparison of the proposed method and state-of-the-art methods for the Ouhands dataset

Methods	F1-score (%)	mIoU (%)	Inference time (ms)	#Parameters
FCN-8s [82]	95.5	80.3	63	134 M
PSPNet [83]	97.0	80.2	50	79.44 M
DeepLabv3 [84]	**97.3**	87.3	43	75.30 M
HGR-Net [16]	96.3	82.6	21	**0.28 M**
CBAM [78]	96.3	82.9	38	7.97 M
DANet [28]	96.4	83.6	35	7.56 M
U-Net [29]	96.7	81.2	25	7.86 M
Segment Anything [85]	79.44	79.77	38	641.09 M
Proposed	**97.3**	**89.0**	**15**	1.02 M

[29] and FCN [82]. Therefore, we included more attention-based models for comparison, such as CBAM [78] and DANet [28]. We chose these six deep neural networks as baselines because their encoder-decoder architecture with attention is similar to ours. A novel network should perform better than the baselines to show its contribution. The results in Table 2.5 indicate that the proposed network performed better than the baselines, considerably improving inference time. Inference time was 10 ms faster than U-Net, the second-fastest architecture, because of our model's efficient spatial and channel attention. The F1-score was 0.6% higher than RA-UNet, which had the second-highest F1-score. Also, the mIoU value was much higher than for other models, 3.5% higher than the second highest.

2.2.5.2 State-of-the-art Comparison

We also compared the proposed model with state-of-the-art hand segmentation approaches. Tables 2.6 and 2.7 list the performance of the state-of-the-art methods for the Ouhands and HGR datasets, respectively. For the Ouhands dataset, the proposed model and DeepLabv3 [84] were tied for the highest F1-score at 97.3%. However, the mIoU value of 89% is the best among other competing models. At 15 ms, the proposed model outperformed all others for inference

Table 2.7: Comparison of the proposed method and state-of-the-art methods for the HGR dataset

Methods	F1-score (%)	mIoU (%)	Inference time (ms)	#Parameters
Hettiarachchi and Peters [86]	96.9	-	-	-
Kawulok et al. [19]	95.6	-	-	-
Kawulok [87]	90.8	-	-	-
FCN-8s [82]	97.7	91.2	63	134 M
PSPNet [83]	98.7	94.0	50	79.44 M
DeepLabv3 [84]	98.8	97.2	43	75.30 M
HGR-Net [16]	98.2	95.8	21	**0.28 M**
OR-Skip_net [88]	96.9	94.3	38	9.72 M
Lumini et al. [89]	96.7	-	-	-
Segment Anything [85]	92.8	94.7	39	641.09 M
Proposed	**99.3**	**98.3**	**10**	1.02 M

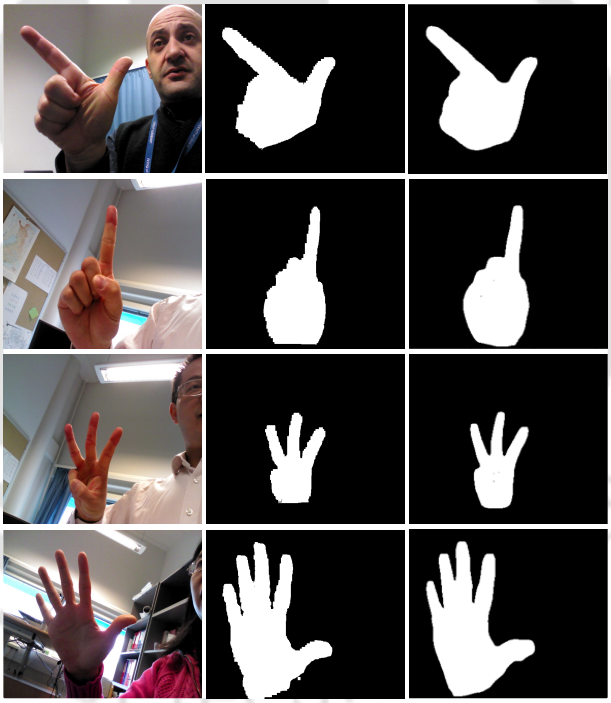

Fig. 2.6: Segmentation results for the Ouhands dataset. The first column contains the input color image, the second contains the ground truth masks, and the third contains the segmented masks.

time. The proposed model had the second-fewest parameters after HGR-Net [16].

For the HGR dataset, the proposed model performed phenomenally, attaining an F1-score of 99.3% and mIoU of 98.3% in 10 ms, surpassing state-of-the-art methods.

2.2.5.3 Qualitative Assessment

Figs. 2.6 - 2.8 show the qualitative results of the proposed method. Fig. 2.6 displays the segmentation results for the Ouhands dataset. The results show the hand's shape (column 3). It

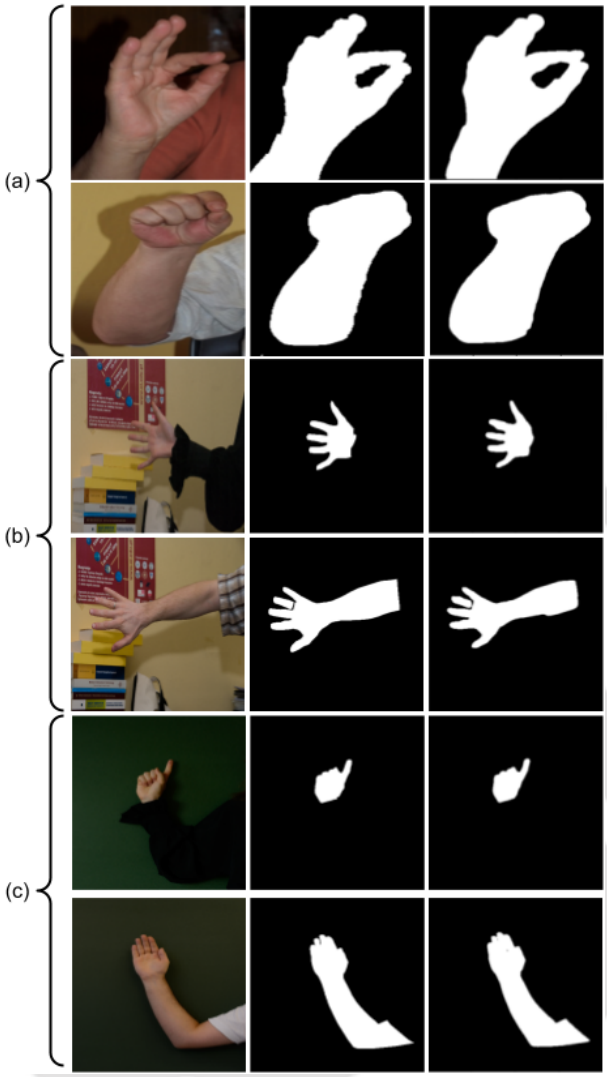

Fig. 2.7: Segmentation results for HGR datasets: (a) HGR1, (b) HGR2a, and (c) HGR2b. The first column contains the input color images, the second contains the ground truth masks, and the third contains the segmented masks.

has a smooth boundary without any gaps. Similarly, Fig. 2.7 shows the predicted segmentation masks for the HGR datasets (column 3).

However, the model failed under certain conditions, as shown in Fig. 2.8. For example, it failed when the hand and face were the same color and overlapping (column 1), when the hands and face lacked distinct hand regions (column 2), and when darkness made the hand almost unrecognizable (column 3). These issues are addressed in the next method.

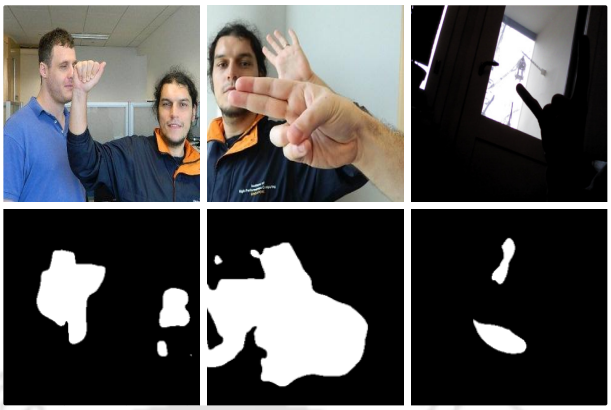

Fig. 2.8: Some failed segmentation cases.

2.3 Method 2: A CNN-transformer architecture

2.3.1 Overview

The proposed segmentation employs an encoder that accepts a color image as input and encodes it in a compressed latent space. The encoder is a combination of convolution and transformer blocks to embed the local and finer semantics of the object and the long-range (global) relations in an image. The convolution block is arranged at the beginning to encode the structural and shape features, which results in a compact representation of the input. This compact representation is fed to the transformer, which further encodes it into object-specific high-level features. These high-level features do not convey any meaning visually but possess semantically significant attributes for classification tasks. Moreover, transformers contain multi-head self-attention mechanisms. If these mechanisms are placed towards the end stage, it improves prediction performance [90]. Thus, they are added to the later part of the encoder. The latent feature maps generated by the encoder are fed to the decoder, gradually upsampling the feature maps to match the dimension of the input image. After each upsampling, the decoder's resultant feature maps are concatenated with the encoder's corresponding dimension feature maps to obtain precise segmentation results. The decoder has no transformers because it only decodes whatever has been encoded without the need to capture the high-level abstract representation. The block diagram of the encoder-decoder architecture is shown in Fig. 2.9.

An input image, $\mathrm{I} \in \mathbb{R}^{H \times W \times C}$ is given to a block of convolution, batch normalization, and ReLU layers. The resultant feature map, $F \in \mathbb{R}^{H' \times W' \times C'}$ is passed through three blocks of

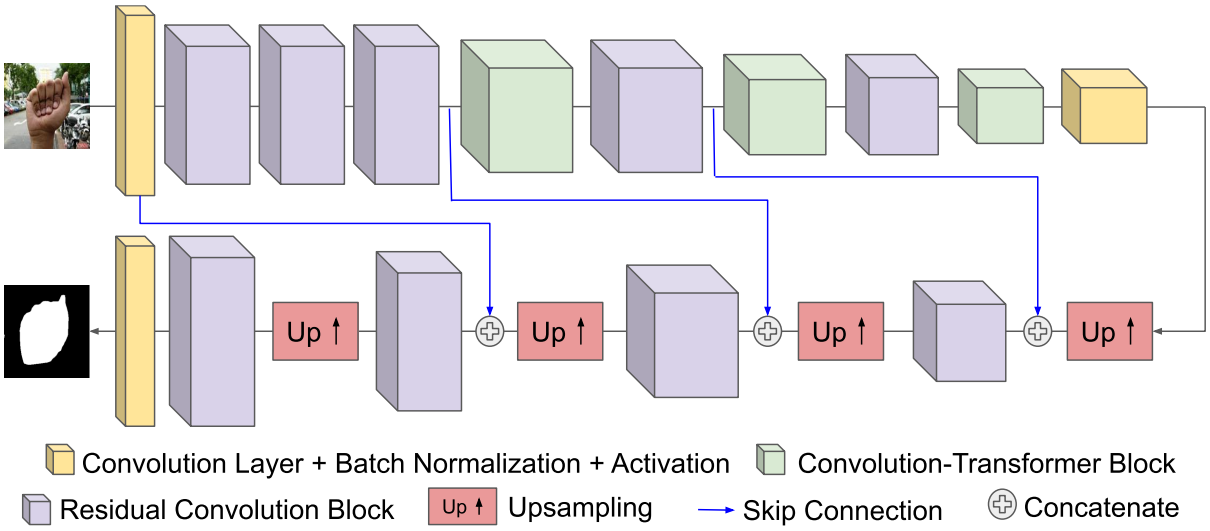

Fig. 2.9: The encoder-decoder architecture for segmentation.

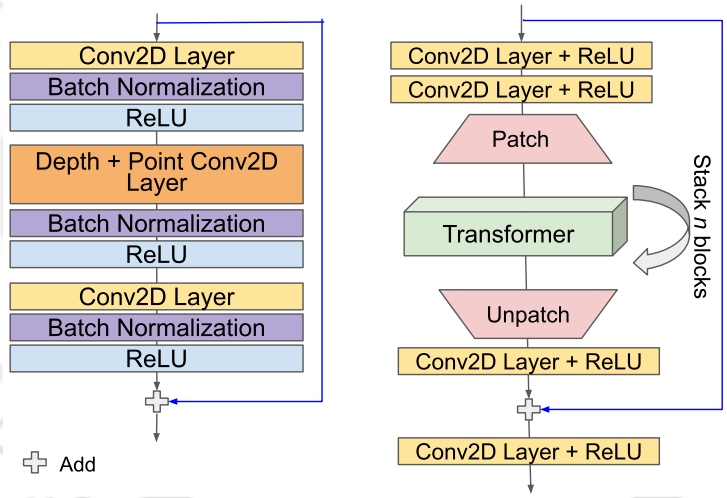

Fig. 2.10: Left: The Residual Convolution Block. Right: The Convolution-Transformer Block.

residual convolution blocks (RCB), shown in Fig. 2.10 (left), and the output is given as

$$F' = (\mathcal{R} \circ BN \circ conv \circ \mathcal{R} \circ BN \circ DP_{conv} \circ \mathcal{R} \circ BN \circ conv)(F) + F, \tag{2.14}$$

where BN is the batch normalization layer, $conv$ is the convolution layer, $\mathcal{R}$ is ReLU activation, and DP_{conv} is the depth + pointwise convolution layer. The representation $(f \circ g)(x)$ in the equation means $f(g(x))$. The depth + pointwise convolution layer adds fewer parameters to a model's training parameters than a convolution layer. Thus, it is an efficient means of learning a feature map's representation. The addition of F in (2.14) represents skip-connections to counteract vanishing gradients. The downsampled F' is fed to a convolution-transformer block

(CTB), shown in Fig. 2.10 (right), which embeds the spatial inductive bias and distal pixel dependencies. CTB's output is given as

$$f_1 = (\mathcal{R} \circ conv)\{(\mathcal{R} \circ conv \circ \mathcal{U} \circ Transformer^{(n)} \circ \mathcal{P} \circ \mathcal{R} \circ conv \circ \mathcal{R} \circ conv)(F') + F'\}. \tag{2.15}$$

Here $\mathcal{P}$ represents dividing the feature map F'' from the second convolution layer of CTB into patches, i.e., $\mathcal{P} : F'' \rightarrow F''', F''' \in \mathbb{R}^{(h_1 w_1) \times \eta \times C''}$, which imitate the sequential arrangement to be given to the transformer unit. The height and width of each patch is denoted by h_1 and w_1, respectively. C'' is the channel number of F'' and $\eta = \frac{H''W''}{h_1 w_1}$ with H'' and W'' being the height and width of F'', respectively. The transformer unit, represented by $Transformer^{(n)}$, is stacked n times before passing the feature map to the "unpatching" operation, $\mathcal{U}$ that maps the transformer output F_{trans} to a convolution-like feature map, i.e., $\mathcal{U} : F_{trans} \rightarrow F'''', F'''' \in \mathbb{R}^{H'' \times W'' \times C''}$.

The encoder contains a CTB after the third, fourth, and fifth RCB, followed by a block of convolution, batch normalization, and ReLU layers, as shown in Fig. 2.9. The encoder's last convolution block maps its output to a high dimensional latent space, from which the feature map f_1 is fed to the decoder. The decoder is a sequential arrangement of convolution blocks represented as

$$f_{seg} = (sigmoid \circ BN \circ conv \circ \text{RCB} \circ Up\uparrow \circ \text{RCB} \circ \oplus^3 Up\uparrow \circ \text{RCB} \circ \oplus^2 Up\uparrow \circ \text{RCB} \circ \oplus^1 Up\uparrow)(f_1). \tag{2.16}$$

$Up\uparrow$ is the upsampling of the incoming feature map by a factor of 2. $\oplus^1, \oplus^2$, and $\oplus^3$ are the skip-connections from the encoder's fourth RCB, third RCB, and first ReLU activation layer, respectively. The sigmoid activation layer outputs the salient feature map f_{seg} emphasizing the hand region and removing the background.

2.3.2 Transformer unit

The transformer unit proposed here does not use the self-attention mechanism generally used by transformers. The proposed transformer unit linearizes the quadratic self-attention by

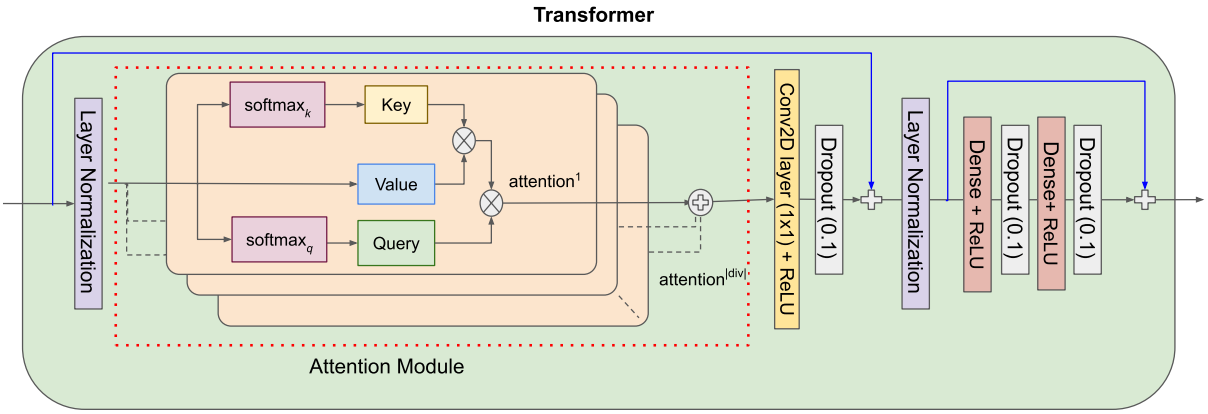

Fig. 2.11: The transformer unit.

ending the quadratic dependency on the spatial dimension of the patch. Thus, the computational complexity and memory requirements are reduced significantly. The block diagram of the transformer unit is shown in Fig. 2.11.

Transformers encode an image's global characteristics using a multi-head attention mechanism that combines several self-attention mechanisms (SAM) [55]. However, the SAM has $O(d_k(hw)^2)$ computational complexity. Moreover, for systems with limited memory, its $O((hw)^2)$ memory usage can result in an out of memory error. Thus, adopting the approach described in the subsection 2.2.3, we propose a transformer unit with linear, instead of quadratic complexity. The attention map obtained using (2.5) treats each channel of the key as a feature map. The channel is multiplied by value to compute global features that weight every location of the incoming feature map. The global features are accumulated by multiplying query with the product of key and value. The final product ensures attention to class-specific features and their location.

Due to this modification, the memory complexity of the transformer unit is reduced from $O((hw)^2)$ to $O(d_khw + d_k^2)$ and computational complexity from $O(d_k(hw)^2)$ to $O((d_k)^2hw)$. Thus, the quadratic increase in complexity with the feature map's spatial dimensions is avoided. The complexity now depends on d_k, which is set by the developer.

The sequential features from the patch block are fed to the layer normalization layer of the transformer unit. Since layer normalization deals with sequential data better than batch normalization and is independent of batch size, it is adopted for the transformer's normalization

Table 2.8: Training details

Language	Python 3.6
GPU	Nvidia Tesla P100
Optimizer	Adam Optimizer
Learning rate	0.0001
Batch size	8
Epochs	20
Division of data	70% training, 15% validation, 15% testing
$d_k = d_v$	[16, 64] for encoder [128, 32] for decoder

[91]. The transformer's output feature map, f_{trans} is given by

$$f_{int} = LN[(D_{0.1} \circ \mathcal{R} \circ \text{attention}_{\text{new}} \circ LN)(\mathcal{P}) + \mathcal{P}].$$
$$f_{trans} = (D_{0.1} \circ \mathcal{R} \circ \mathcal{D} \circ D_{0.1} \circ \mathcal{R} \circ \mathcal{D})(f_{int}) + f_{int}. \tag{2.17}$$

Here, LN is the layer normalization layer, $D_{0.1}$ is the dropout layer with dropout rate set to 0.1, and $\mathcal{D}$ is the dense layer. f_{int} is the intermediate feature map that is given to the feed-forward network of the transformer. Two skip connections facilitate transfer of finer details to the later layers.

2.3.3 Experiments and Results

This subsection reports the evaluation of the proposed architecture based on the quantitative and qualitative results. Training information is given in Table 2.8. The F1-score and mIoU (mean Intersection over Union) were adopted as the performance measure as used in the previous method. The loss function proposed in the previous method was considered, which is defined by (2.11).

2.3.3.1 Quantitative Assessment

We quantitatively analyzed the importance of the transformer unit, which employ the attention mechanism, in generating precise segmentation masks. This unit captures the long range dependencies and focuses on the characteristics critical to hand segmentation. Table 2.9 lists the F1-scores of the model with and without the transformer unit. The F1-score increases 2.48% due to the inclusion of the transformer unit.

Table 2.10 reports the effect of dividing the query, key, and value matrices, used in the

Table 2.9: Performance with and without transformer unit.

Transformer Unit	F1-score
Without	95.25
With	97.73

Table 2.10: Performance of the segmentation model for different numbers of groups of query, key, and value matrices.

Number of groups	F1-score (%)	Inference (ms)
1	96.97	16.9
2	**97.73**	13.9
3	96.80	**13.4**
4	96.51	13.9
5	96.26	13.6

transformer unit, into groups of different sizes on the segmentation model's performance. Based on observation, the division of the matrices into two groups reports the maximum F1-score as observed in the case of SFAB proposed for the previous segmentation method (Method 1). Since this resulted in better segmentation maps, two groups were selected even though three groups resulted in a slightly lower inference time.

Table 2.11: Comparison of state-of-the-art models with the proposed segmentation model.

Methods	F1-score (%)	mIoU (%)	Inference time (ms)	#Parameters
FCN-8s [82]	95.5	80.3	63	134 M
Attention U-Net [30]	96.8	74.8	29	6.44 M
RAU-Net [31]	96.9	83.8	32	11.62 M
DeepLabv3 [84]	97.1	87.3	43	75.30 M
DeepLabv3+ [92]	97.3	88.5	56	85.12 M
HGR-Net [16]	96.3	82.62	21	**0.28 M**
CBAM [78]	96.3	82.91	38	7.97 M
DANet [28]	96.4	83.69	35	7.56 M
U-Net [29]	96.7	81.26	25	7.86 M
U-Net + Triplet Attention [93]	85.4	70.18	25	7.88 M
U-Net + CondConv [94]	92.7	75.39	31	9.7M
PSPNet [83]	95.7	80.2	18	0.96 M
ICNet [95]	95.3	81.6	24	6.83 M
HRNet [96]	96.8	88.5	38	28.60 M
SegNet [97]	96.1	80.6	27	2.95 M
Segment Anything [85]	79.44	79.77	38	641.09 M
Proposed	**97.7**	**89.67**	**13**	0.88 M

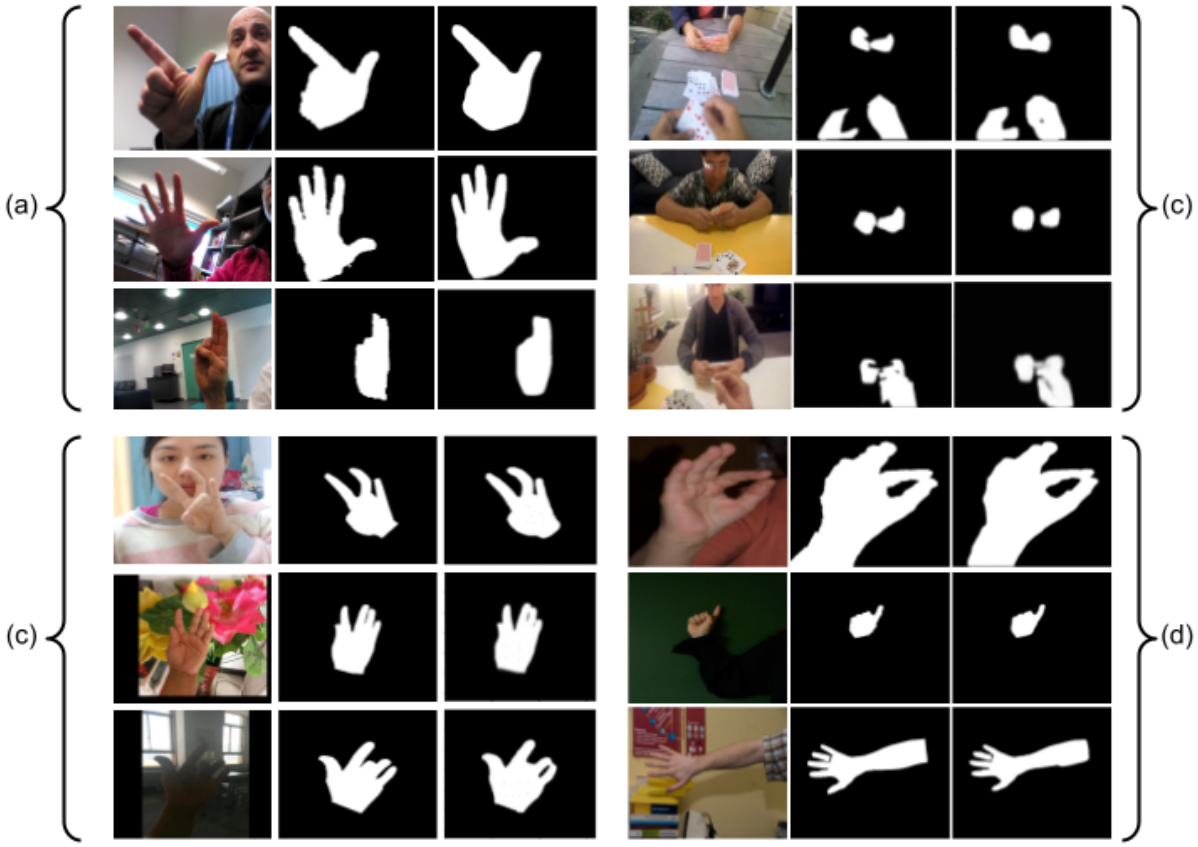

Fig. 2.12: The segmentation masks for (a)Ouhands, (b)Egohands, (c)HIU, and (d)HGR datasets. The first column of each sub-image shows the color images, the second column shows the ground truth, and the third column shows the detected salient regions.

2.3.3.2 State-of-the-art Comparison

Moreover, the proposed segmentation model was compared with a few state-of-the-art models, as shown in Table 2.11. The proposed segmentation model performs better than the other techniques. Its F1-score, mIoU, and inference time are the best among the compared models, and its parameter count is also low, second only to HGR-Net.

2.3.3.3 Qualitative Assessment

The qualitative result of the segmentation model for four datasets is shown in Fig. 2.12. The results indicate that the segmentation maps are robust despite variations in illumination, occlusion, and the presence of skin regions and background clutter. Moreover, they produce good segmentation results for multiple hands as seen for the egohands dataset.

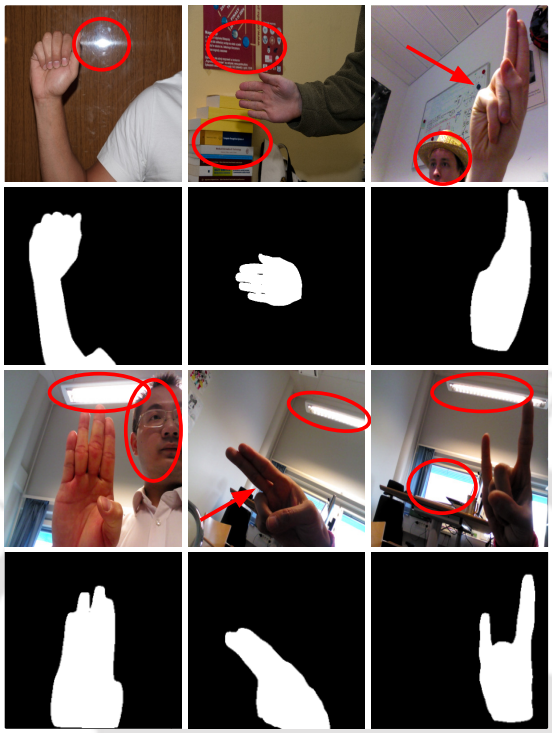

Fig. 2.13: In these qualitative results, red circles indicate challenging conditions, and arrows indicate specularity and backlighting.

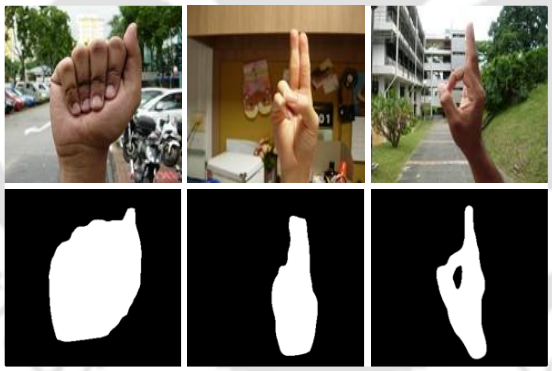

Fig. 2.14: Segmentation results for the NUS dataset with a model trained with Ouhands.

2.4 Which of the two methods to choose?

The proposed models performed well for images with different hand poses and orientations, the presence of a human face and background clutter, variations in illumination, and specularity. This is shown in Fig. 2.13, which highlights challenging conditions with an arrow or circle. Owing to the attention mechanism, the model does not deviate from the region of interest. Normalization and using geometrically altered training samples make the model robust against changes in light, orientation, and scale. Though the qualitative results displays similar

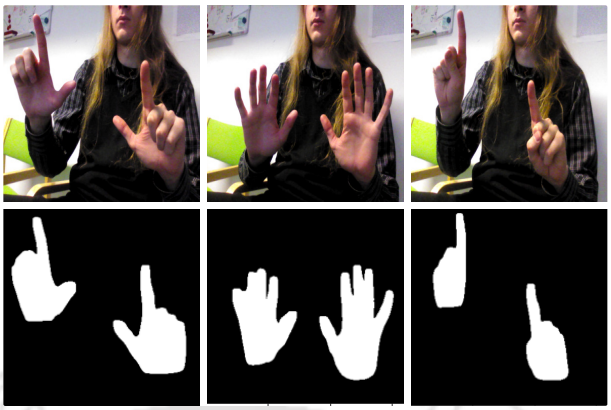

Fig. 2.15: Segmentation results for images with two hands.

performance, the quantitative results of the method 2 is slightly better than the method 1 in terms of F1-score, mIoU, and inference time (as seen in Table 2.6 and Table 2.11).

Moreover, the method 2 generalizes better. We trained the method 2 model with the Ouhands dataset and tested with the NUS dataset [8], which is primarily a classification dataset. Despite being completely unaware of the conditions where the images were captured, the segmentation model produced encouraging results as shown in Fig. 2.14. The model was also tested on supplementary data samples from Ouhands containing two hands in an image. Fig. 2.15 shows that the segmentation masks are precisely detected for two hands even though training samples contained only one hand.

Thus, based on the above results, it can be concluded that the proposed method performs hand segmentation accurately and efficiently and can be considered for integration into a recognition network for better hand gesture recognition.

2.5 Contribution

- A novel architecture for hand segmentation was proposed (Method 1) that uses an attention mechanism to capture long-range (global) characteristics and obtain accurate hand masks.
- The architecture has two novel and efficient attention blocks, one for spatial features to emphasize the hand's location and one for channel features to emphasize the pixel class.
- Another novel segmentation model was proposed (Method 2) that combined the benefits

of a CNN and a transformer to obtain a segmentation map, precisely separating the hand from the background.

- The transformer's self-attention is replaced with an efficient attention mechanism to reduce computational complexity and memory requirements.
- A compound segmentation loss function is proposed to maintain the hand's geometrical shape and smooth boundary, especially around the fingers. It also corrects for class imbalance.

2.6 Summary

Hand gesture recognition plays a crucial role in day-to-day communication among human beings. Interpreting these gestures by a machine for a robust human-machine interaction requires accurate recognition of the gestures in complex or challenging conditions. Primarily, the machine needs to localize the hand to understand a gesture, and hand segmentation becomes very essential in such a situation. Therefore, two deep neural architectures were designed that generated precise hand segmentation masks. One architecture utilized two attention blocks, the SFAB and CFAB, which efficiently calculated attention in two dimensions–spatial and channel. The other architecture combined the benefits of convolution and transformer to obtain segmentation results. The second architecture benefitted from the efficient modification of the attention mechanism proposed for the first architecture, which helped in linearizing the quadratic complexity of a transformer. Thus, both the architecture devised a way to reduce the memory and computational complexities. Furthermore, a composite loss function was proposed to address class imbalance, ensure boundary smoothness, and retain the geometric shape of the hand, especially around the fingers. The qualitative and quantitative results obtained for both methods were remarkable and they generated precise segmentation masks quite efficiently. However, the second method performed slightly better than the first method in generalizing the segmentation task and in inference time. Therefore, the second method is chosen as the preprocessing step and can be combined with the recognition step without substantially increasing memory complexity.

3

Hand Gesture Recognition

Contents

3.1 Introduction

One of the prominent objectives of HCI is to establish an engaging and streamlined interaction. Hardware-based interactions are the most accurate but they restrict the users to a fixed location and may be cumbersome to use. Touchless interaction presents itself as a strong contender. Its requirement is further bolstered by the recent coronavirus pandemic. Vision-based hand gesture recognition (VHGR) is a capable touchless strategy for HCI, which is robust and cost-effective [7]. Since hands are a natural and spontaneous medium of interaction, VHGR is an intuitive medium of HCI [98, 99]. For instance, VHGR can find use in AR/VR applications, remote control of equipment, robotics, rehabilitation, and many more [62, 65, 100, 101]. However, VHGR has a few limitations that are associated with any vision-based system and are not as accurate as standard keyboard and mouse-based interactions. Nevertheless, the hand gestures are capable of communicating pointing and representative commands as the mouse and keyboard with ample precision.

Quite a few methods have been proposed to achieve an effective means of hand gesture recognition addressing these limitations, such as varying hand shapes and sizes, lighting conditions, background, and occlusions. But there is scope for improvement. Therefore, this work proposes a two-stage hand gesture recognition: the first stage preprocess the image to contain only the hand and a uniform background and the second stage involves an attention-based deep neural classification network. The details of the complete recognition network are detailed in the subsequent sections.

3.1.1 Intuition and Objective

People convey much information through hand gestures, especially those with speech disorders, disabilities, or in hospital. As hand gesturing is a natural medium of interaction among humans [1,16,102], the necessity for a computer to decode the gesture efficiently and accurately for a robust HCI is of the essence. A robust VHGR network that addresses the limitations of a vision-based system, can act as a communication bridge between the machine and humans.

Thus, developing a VHGR network, which can generalize hand gestures in challenging

conditions is the prime objective of this work. If not completely, the network should produce optimal results with a considerable tradeoff between accuracy and inference time.

3.2 Methodology

3.2.1 Overview

This work proposes an effective integration of hand segmentation with hand gesture classification problem to attain state-of-the-art gesture recognition results. The CNN-transformer architecture proposed in the previous chapter is used to obtain a segmentation mask containing only the hand with a uniform background. Since the background clutter is removed, the segmentation mask does not contain any regions irrelevant to classification. Therefore, this mask is fed to the proposed classification network. The highlights of the classification network are the novel convolution-transformer feature engineering model and AKCAL. AKCAL uses adaptive kernels at different scales to ensure automatic feature extraction, ensuring the network focuses on only the hand region, not on the adjoining background region. This enhances gesture recognition accuracy. Furthermore, the architecture benefits from the combination of convolution neural network (CNN) and transformers. The advantage of CNN lies in its capturing of local information in feature maps, making useful assumptions about the targeted task (i.e., spatial inductive bias), and sharing weights. Transformer, in contrast, excels in fusing global contextual information, generalizing, and including an attention mechanism. Therefore, we propose a recognition method that combines the strengths of both architectures.

The segmentation mask (f_{seg}) passes through a convolution—batch normalization—ReLU block and a max pooling layer. This feature map is fed to a sequential arrangement of three blocks: the depth + point convolution layer, convolution—batch normalization—ReLU block, and AKCAL. Next, the input is added as a skip connection to the output. A max pooling layer follows, which halves the spatial dimension. This sequence of passing the feature map through a sequential arrangement of feature extracting blocks, and skip connection followed by a max pooling layer is carried out for three more levels before passing to a multi-layer perceptron (MLP) for classification. Towards the end of the network, convolution-transformer block (CTB)

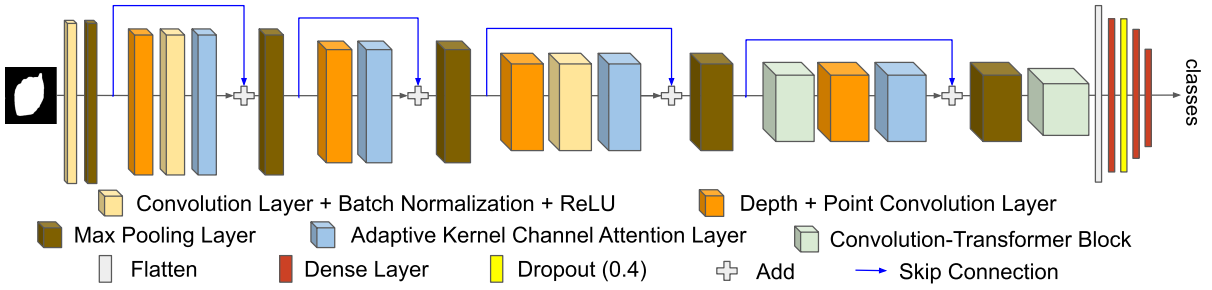

Fig. 3.1: The Classification Network.

replaces the convolution-batch normalization-ReLU block because of the transformer's ability to encode high-level class-specific features. The following set of equations describes the feature engineering approach.

$$\Phi_1 = (\mathcal{M} \circ \mathcal{R} \circ BN \circ conv)(f_{seg}) \tag{3.1}$$

$$\Phi_2 = \mathcal{M}[(\text{AKCAL} \circ \mathcal{R} \circ BN \circ conv \circ DP_{conv})(\Phi_1) + \Phi_1] \tag{3.2}$$

$$\Phi_3 = \mathcal{M}[(\text{AKCAL} \circ DP_{conv})(\Phi_2) + \Phi_2] \tag{3.3}$$

$$\Phi_4 = \mathcal{M}[(\text{AKCAL} \circ BN \circ conv \circ DP_{conv})(\Phi_3) + \Phi_3] \tag{3.4}$$

$$\Phi_5 = (\text{CTB} \circ \mathcal{M})[(\text{AKCAL} \circ DP_{conv} \circ \text{CTB})(\Phi_4) + \Phi_4] \tag{3.5}$$

where $\mathcal{M}, \mathcal{R}, conv, DP_{conv}$, and BN denote max pooling, ReLU, convolution, depth + pointwise convolution, and batch normalization layers, respectively. The feature map Φ_5 is flattened and passed to the classifier given by

$$\text{classes} = (\mathcal{D} \circ \mathcal{D} \circ D_{0.4} \circ \mathcal{D} \circ \text{flatten})(\Phi_5). \tag{3.6}$$

Here, $\mathcal{D}$ and $D_{0.4}$ represents dense and dropout (40%) layers, respectively. The number of neurons in the MLP's hidden layers are progressively reduced to lower the parameter count. The block diagram for the classification network is shown in Fig. 3.1 and the adaptive kernel channel attention layer is shown in Fig. 3.2.

3.2.2 Adaptive Kernel Channel Attention Layer

The incoming feature map, $\mathcal{X} \in \mathbb{R}^{x \times y \times z}$, is divided into two feature maps, $\mathcal{X}_1$ and $\mathcal{X}_2 \in \mathbb{R}^{x \times y \times z}$, encoded via 3×3 and 5×5 kernels, respectively. The goal of this division is to extract different

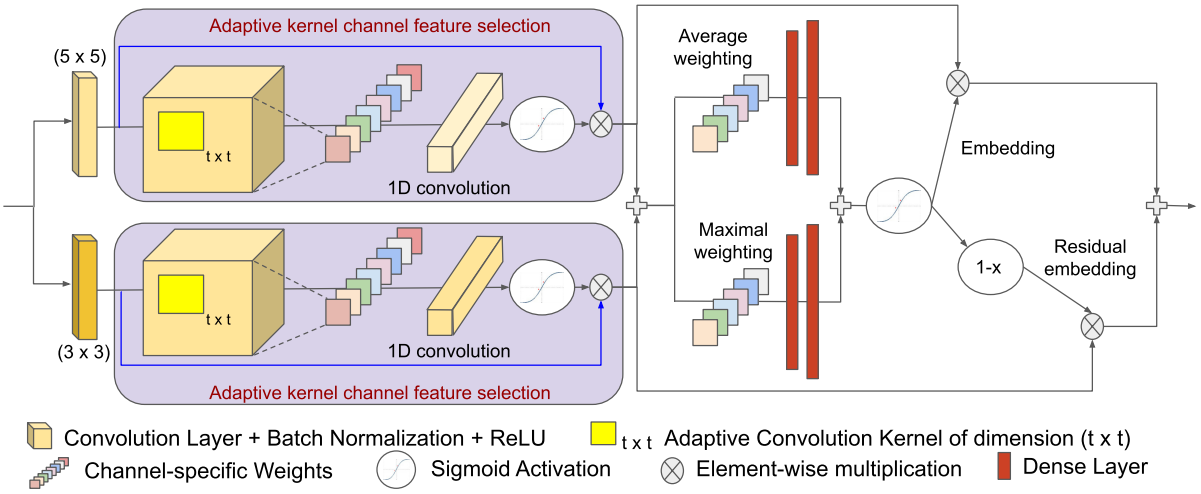

Fig. 3.2: The Adaptive Kernel Channel Attention Layer.

scale information using 3×3 and 5×5 kernels. Increasing the kernel size increases its receptive area to capture more information from a region but at the cost of more training parameters. Therefore, only two divisions are considered with different kernel sizes.

Next, feature maps from both paths are passed through an adaptive kernel with a channel feature selection module, which starts with a convolution—batch normalization—ReLU block. The kernel $t \times t$ for the convolution layer is adaptive so that the extent of feature interactions is not determined manually. Adjusting the parameter count allows for optimal encoding of features. Since the class-specific features are encoded across channels, we consider a function that defines a relation between channel Z and kernel size t. Moreover, the number of channels is a power of 2; therefore, the relationship is defined as

$$Z = 2^{(\rho t)} \Rightarrow t = \frac{\log_2 z}{\rho}, \ \rho \text{ is constant.} \tag{3.7}$$

Subsequently, the module encodes the global features of each channel via an average weighting scheme given by

$$Z_1^l = \frac{1}{xy} \sum_{u=1}^{x} \sum_{v=1}^{y} \mathcal{X}_{t\times t}^l(u, v). \tag{3.8}$$

$\mathcal{X}_{t\times t}^l(u, v)$ represents the l^{th} feature map after the adaptive kernel convolution layer. A 1D convolution layer further encodes this with an adaptive kernel t and sigmoid activation. The resulting

attention map is multiplied with the input to this module to emphasize the channel features and outputs the features $\mathcal{X}'_1$ and $\mathcal{X}'_2$ for the two paths, respectively. $\mathcal{X}'_1$ and $\mathcal{X}'_2$ are added and passed through two paths that weight them (using 3.8 and 3.9, respectively) to generate channel-specific weights.

$$Z_2^l = \max_{u\in\{1,\ldots,x\},v\in\{1,\ldots,y\}} \mathcal{X}^l_{t\times t}(u,v). \tag{3.9}$$

The inclusion of both average and maximal weighting increases classification accuracy because average weighing captures the feature map's soft global characteristics and maximal weighting captures their most significant characteristics. The weighted features are then passed through two dense layers and their outputs are added together to form an intermediate feature map $\mathcal{X}_{\text{int}}$. $\mathcal{X}_{\text{int}}$ is passed through a sigmoid layer to generate the embedding and residual embedding. These embeddings select classification-relevant channel features. The output of the AKCAL is

$$f_{cal} = (\text{embedding} \otimes \mathcal{X}'_1) + (\text{residual embedding} \otimes \mathcal{X}'_2). \tag{3.10}$$

where embedding is the weighted and encoded output of the adaptive kernel channel feature selection module. It is the output of AKCAL's last sigmoid activation. Residual embedding represents the residual weighted and encoded output obtained by subtracting embedding from 1. $\otimes$ denotes element-wise multiplication.

3.2.2.1 Loss Function

Categorical cross-entropy is used for classification loss, given by

$$\mathcal{L}_{\text{class}} = \sum_{j=1}^{|classes|} \sum_{i=1}^{|samples|} y_{ij}\, log\, \hat{y}_{ij} \tag{3.11}$$

3.3 Experiments and Results

This section reports the evaluation of the proposed hand gesture recogntion architecture based on the quantitative and qualitative results. The training information is given in Table 3.1.

Table 3.1: Training details

Environment	Python 3.6
GPU	Nvidia Tesla P100
Optimizer	Adam Optimizer
Learning rate	0.0001
Batch size	16
Epochs	30
Division of data	70% training, 15% validation, 15% testing

Table 3.2: Performance comparison of baselines with different combinations of supporting modules.

Models	Accuracy (%)
pre-trained VGG16 [60] (RGB images)	72.8
pre-trained VGG16 [60] (segmentation masks)	81.6
Encoder (RGB images)	85.6
Encoder (segmented masks)	87.1
Encoder + Triplet Attention [93]	72.4
CondConv Encoder [94]	84.2
Adaptive Kernel Encoder	89.8
Channel Attention Encoder	91.0
Adaptive Kernel Channel Attention Layer Encoder	**93.8**

Accuracy was adopted as the performance measure for classification, and is defined as

$$Accuracy = \frac{\sum_{i} \mathcal{A}_{ii}}{\sum_{i,j} \mathcal{A}_{ij}}. \tag{3.12}$$

where $\mathcal{A}$ denotes the confusion matrix.

The recognition rate of the proposed method is high with an accuracy of 93.8 % and 98.0 % for Ouhands and NUS II, respectively. The use of the segmentation masks helped achieve phenomenal results because it mitigated the major challenges encountered in hand gesture recognition. As shown in Table 3.2, using a pre-trained VGG16 [60] network on segmented images yielded better results than on RGB images. Similarly, an architecture similar to the encoder part of the segmentation method (termed as encoder) performed better with segmentation masks

Table 3.3: Performance for different values of rho to determine optimal adaptive kernel size t.

ρ values	Accuracy (%)
$\rho = 0.5$	84.2
$\rho = 1$	87.2
$\rho = 1.5$	89.6
$\rho = 2$	**93.8**
$\rho = 2.5$	90.2

Table 3.4: Performance comparison for different weighting schemes in AKCAL.

Weighting schemes	Accuracy (%)
no weighting	86.0
only average weighting	88.7
only max weighting	89.8
both average and max weighting	**93.8**

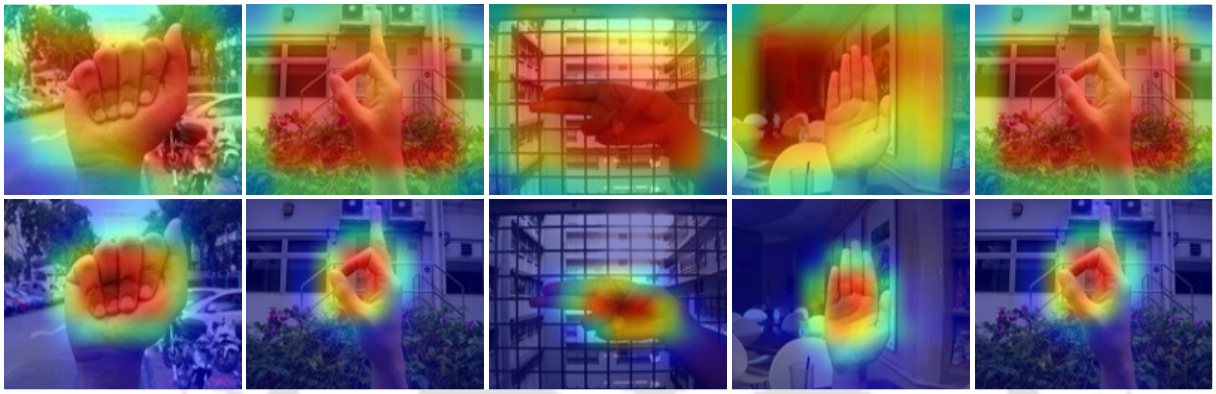

Fig. 3.3: Effect of AKCAL. Top: without AKCAL. Bottom: with AKCAL.

than color images. This indicates the benefits of using segmentation masks. Moreover, we tried different combinations, such as attaching different feature-enhancing modules to the encoder. The performance is recorded in Table 3.2. The AKCAL encoder had the best performance with 93.8 % accuracy.

Also, there is a hyperparameter ρ associated with the adaptive kernel size of AKCAL. We experimented with different values of ρ and found that $\rho = 2$ had the best result (shown in Table 3.3). The advantage of using weighting schemes is shown in Table 3.4. The best performance is derived when both average and max weighting schemes are employed. The effects of using AKCAL is depicted in Fig. 3.3. AKCAL resulted in a packed heatmap instead of a dispersed one, as shown in Fig. 3.3 (bottom). The intuition behind using AKCAL is that channel attention focuses on what to attend in a feature map. We showed the heatmap of input images to illustrate what locations the proposed network focus on to learn classification-related features in the presence and absence of the AKCAL layer. The AKCAL helps the network focus on only the hand region and not capture features from the adjoining background region. This results in the compact heatmap. But, without the AKCAL layer, the resulting heatmap does not converge on the hand region, suggesting that noisy information is also encoded. This reduces the recognition

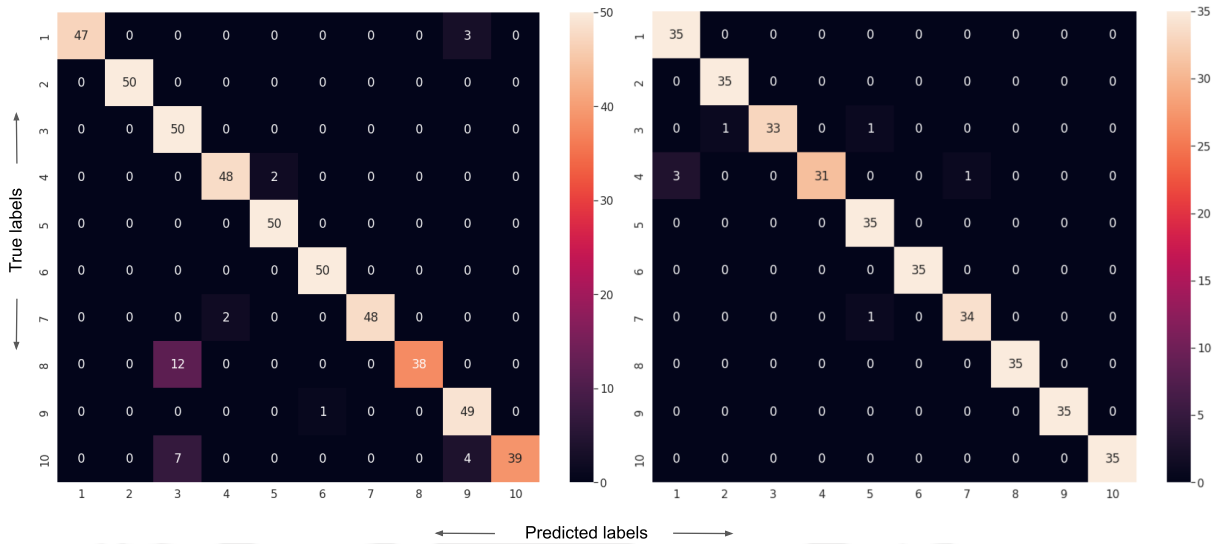

Fig. 3.4: Confusion matrix. Left: Ouhands. Right: NUS.

Table 3.5: Comparison of the proposed classification method with the state-of-the-art methods for Ouhands and NUS datasets.

Methods	Dataset	Accuracy (%)
Dadashzadeh *et al.* [16]	Ouhands	87.8
Matilainen *et al.* [9]	Ouhands	83.25
Bose *et al.* [57]	NUS	97.98
Aditya *et al.* [103]	NUS	94.7
Pisarady *et al.* [8]	NUS	94.36
Sharma *et al.* [59]	NUS	96.62
Bhaumik *et al.* [104]	Ouhands	65.1
Bhaumik *et al.* [105]	NUS	97.78
Sahoo *et al.* [106]	NUS	94.80
Proposed	Ouhands	**93.8**
	NUS	**98.0**

accuracy.

The confusion matrix for the two datasets is shown in Fig. 3.4. Table 3.5 compares some of the state-of-the-art methods with the proposed method, and it is observed that the proposed method performs better than the others. It further accentuates the robustness of the proposed method.

3.4 Contributions

- A novel classification network that uses an adaptive kernel channel attention layer (AKCAL) to increase hand gesture recognition accuracy is proposed.
- The transformer's self-attention is replaced with an efficient attention mechanism to re-

duce computational complexity and memory requirements.

3.5 Summary

Hand gesturing is a common means of interaction among people and also certain interfaces involving robotics, AR/VR applications, etc. HCI based on hand gestures provides flexibility and ease of operation to users. Moreover, VHGR is cost-effective, making it a popular medium to control interfaces of different gesture-based applications. Despite limitations of vision-based hand gesturing like background clutter, variable hand shapes and sizes, variable illumination, etc., streamlined operations of these interfaces are possible through accurate hand gesture recognition. Therefore, this work also proposes a robust hand gesture recognition method integrated with a segmentation model that provides a tool to control human-computer interfaces based on gestures. This recognition model capitalizes on the benefits of convolution and transformer networks and a proposed attention module named the adaptive kernel channel attention layer. The combination of the convolution layer and the transformer helps encode local and global dependencies and the attention module focuses the network learning. The proposed recognition method accepts the segmentation model's output as input and performs feature engineering on the segmented masks as it progresses through the stack of layers. The proposed method performs well on benchmark hand gesture datasets with better evaluation results than state-of-the-art methods. An accurate hand gesture recognition network ushers the development of an interface that helps human-computer communication. This communication may be decoding a hand gesture to another person or helping with rehabilitation exercises as a virtual physiotherapist or something else. This type of interface allows people to express their thoughts and ideas fluently and its smooth execution can be attributed to efficient and effective hand gesture recognition.

4

Hand Gesture Detection

Contents

4.1 Introduction

Object detection refers to a computer vision technique designed to identify objects and pinpoint their locations within an image. An object detection algorithm will provide the coordinates of recognized objects in an image, along with a confidence score indicating the algorithm's level of certainty regarding the accuracy of its predictions. In the context of hand gesture detection (HGD), the object of interest is the hand, and the confidence score determines the classification of the hand gesture.

HGD offers advantages over the approach of hand gesture segmentation followed by recognition, as it executes both processes concurrently. This simultaneous operation enables HGD to facilitate real-time or online applications without the delay associated with transitioning between segmentation and recognition phases. However, similar to other hand gesture-related challenges outlined in preceding chapters, HGD is also impeded by background clutter and variations in illumination and in the hand's size and angle. To overcome these challenges, the development of a robust hand detection module is imperative. Therefore, three hand gesture detection methods are proposed in this chapter. They are:

- A RetinaNet-CBAM architecture that utilizes prior anchor box information (fixed size bounding boxes that are adjusted iteratively to obtain final bounding box) to make its predictions.
- An anchorless detection transformer for hand gesture detection.
- An anchorless CenterNet-based architecture that treats the hand as a point.

The architecture details and performances of these methods are described in the subsequent sections.

4.1.1 Intuition and Objective

Till now the recognition of the hand gestures was done in a two-stage manner, i.e., localization or segmentation of the hand region followed by classification. However, this two-stage approach is not very intuitive for real-time or online applications. Hand gesture detection that simultaneously localizes and classifies the gesture would be more practical in such scenarios.

Also, obtaining pixel-by-pixel information is not always essential for operating a hand gesture-controlled HCI. Localization of the region where the hand is present, such as using a bounding box, would also convey the required information.

Therefore, the primary objective of this chapter is to propose a method that simultaneously regresses a bounding box encompassing the hand and predicts the class of the hand gesture for a hand-operated interface.

4.2 Method 1: RetinaNet-CBAM hand gesture detection architecture

4.2.1 Overview

HGD is widely explored for hand gesture recognition due to its inherent capability of recognizing the hand gesture's class with the location of the hand in the image. In other words, it recognizes "what" the object is and "where" it is located. This proposed HGD method is developed on the RetinaNet [54] architecture due to its lower inference time compared to two-stage detection architectures, like RCNN [47], Faster-RCNN [49], etc., and higher accuracy than single-stage detection architectures, like SSD [53], YOLO [50], etc. Moreover, multiscale detection gives a strong reason to consider RetinaNet for our HGD. Further, an attention mechanism is employed, which teaches the HGD model to focus on relevant information. Thus, a RetinaNet-based end-to-end hand gesture detection model integrated with Convolutional Block Attention Module(CBAM) [78] is proposed as shown in Fig. 4.1.

4.2.2 ResNet-18 backbone

ResNet-18 [107] is used as the backbone of the proposed RetinaNet-CBAM architecture. Residual Network uses shortcut connections (or skip connections), which add a previous feature map to the current feature map to embed finer details and overcome the vanishing gradient issue. When both the feature maps are of the same dimension, they are simply added (i.e., identity shortcuts). However, when the dimensions vary, either zero padding is done before adding, or 1×1 convolutions can be utilized to match the dimensions (i.e., projection shortcuts). When the sizes are different, shortcuts are done with a stride 2.

There are four stages in the ResNet-18 architecture and the output of the fourth stage is fed

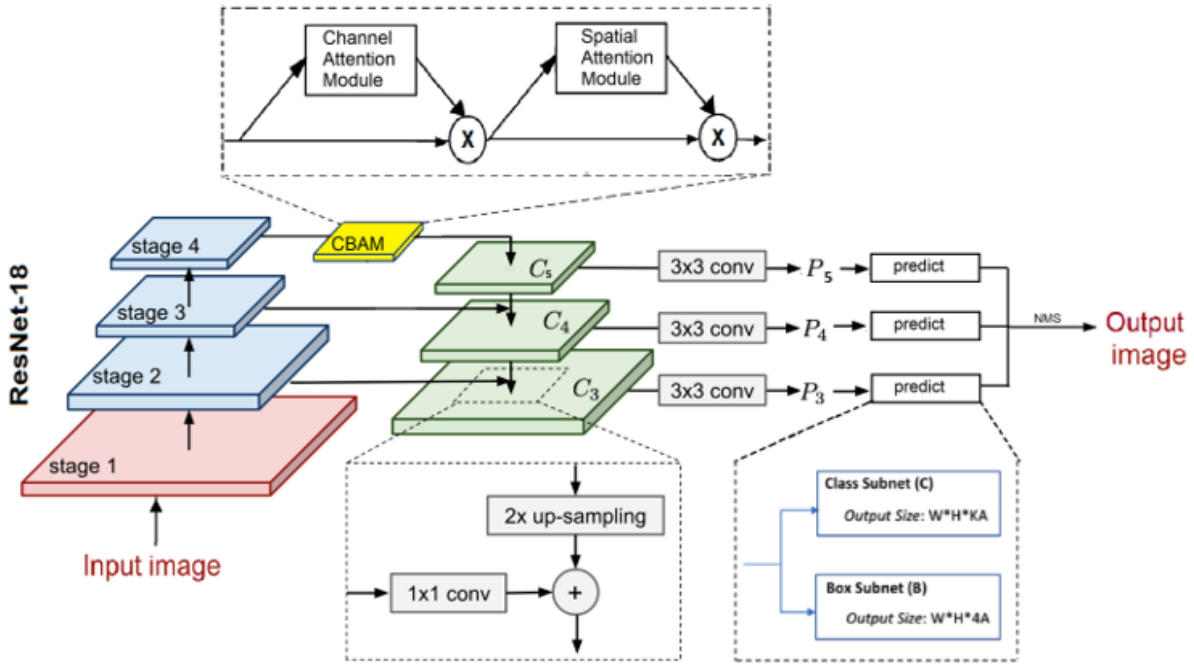

Fig. 4.1: Architecture of the proposed model.

to the CBAM layer. The fourth stage of the backbone contains the most compact features and focussing on this compact feature map helps filter out the irrelevant information. Subsequently, the focussed feature map is passed to the feature pyramid network (FPN) [108].

4.2.3 Convolutional Block Attention Module

CBAM has two sub-modules: channel attention module and spatial attention module. The feature map F obtained from the fourth stage of ResNet-18 is first passed through the channel attention module, where 'what' to focus on is obtained in the form of a 1D attention map M_c. This is then multiplied, element-wise, with F to get F'. Then, F' is passed through the spatial attention module, which produces a 2D attention map M_s, which shows 'where' to focus. This is again multiplied, element-wise, with F' to obtain F''.

$$F' = M_c(F) \otimes F \tag{4.1}$$

$$F'' = M_s(F') \otimes F' \tag{4.2}$$

$\otimes$ denotes element-wise multiplication. M_c and M_s are obtained as:

$$M_c(F) = sigmoid(MLP(\mathbb{A}(F)) + MLP(\mathcal{M}(F))) \tag{4.3}$$

Here, max pooling and average pooling are done along the spatial dimension.

$$M_s(F) = sigmoid(f^{7\times7}([\mathbb{A}(F); \mathcal{M}(F)])) \tag{4.4}$$

Here, max pooling ($\mathcal{M}$) and average pooling ($\mathbb{A}$) are done along the channel dimension.

sigmoid represents sigmoid activation function. F'' is of the same dimension as the original F, the feature map obtained from the last stage of ResNet-18.

4.2.4 Feature Pyramid Network

FPN is built on top of ResNet-18 architecture. FPN comprises a bottom-up and a top-down pathway with lateral connections. The bottom-up pathway is the standard convolution network for feature extraction. The spatial resolution decreases, as we go up. As we go down the path, the top-down pathway restores resolution with rich semantic information by upsampling the previous layer by 2 using nearest neighbor upsampling. Lateral connection performs element-wise addition between feature maps of the same spatial size from the bottom-up pathway (after going through 1×1 convolutions to match channel dimensions) and the top-down pathway. To reduce aliasing effects due to upsampling, a 3×3 convolution is performed and the final feature map is obtained.

The feature pyramid constructed has 5 layers called P_3 to P_7. Layer P_1 has the resolution 2^l lower than the input resolution (where l is the level in the pyramid). P_3 to P_4 are computed from the corresponding layer outputs from ResNet-18. P_5 is computed from the output of CBAM. P_6 is computed using 3×3 convolution with stride 2 and P_7 is computed by ReLU activation function, followed by 3×3 convolutions on P_6 with stride 2. The number of output channels is kept 256 for all the layers of the pyramid. A rich, multi-scale feature pyramid, obtained from a single input image, improves the performance significantly.

Anchor boxes

Anchor boxes [49] of different heights and widths are defined to capture the scales and aspect ratios of different classes in the training data. The use of anchor boxes enables a network to detect objects of different scales, multiple objects, and overlapping objects all at once, thus

increasing the overall speed.

The anchors used at levels P_2 to P_6 have areas 32^2 to 512^2, respectively. For each level, anchors of 3 aspect ratios are used: $\{1:2, 1:1, 2:1\}$. For better scale coverage, additional anchors of size $\{2^{1/3}, 2^{2/3}\}$ times the original sizes are added, making a total of 9 anchors for each level. For each anchor box, two things are to be predicted:

(i) The 4 location offsets against the anchor box (a vector of length 4 containing box coordinate offsets obtained from the box regression subnetwork).

(ii) The K number of class probabilities to tell us which class the bounding box belongs to (a one-hot vector of length K obtained from the classification subnetwork).

In total, we are predicting $(4 + K)$ values for one anchor box. A threshold of 0.5 is taken for the intersection-over-union (IoU) for the foreground objects and an IoU of [0, 0.4) for the background. If the overlap is in [0.4, 0.5), the anchor box is unassigned and it is ignored during training.

4.2.5 Classification Subnetwork

The Classification Subnetwork is a small fully connected network attached to each level of the feature pyramid. It is used to predict the probability of the hand being present at each spatial position for each of the $A = 9$ anchors and K classes. The parameters are shared across all pyramid levels. Four 3×3 convolution layers, each having $C = 256$ channels, are applied to each pyramid level, followed by ReLU activation, and a 3×3 convolution layer with $K * A$ filters. Then, sigmoid activation functions are applied to get an output of KA binary predictions per spatial location.

4.2.6 Box Regression Subnetwork

The Box Regression Subnetwork is used to obtain the relative offsets between the anchor and the ground truth box. The design of this subnetwork is the same as the classification subnetwork except that it outputs a $(4 * A)$ linear vector per spatial location for each of the anchor boxes. The parameters for the classification subnetwork and box regression subnetwork are not shared and are different from each other.

4.2.6.1 Focal Loss

We use focal loss [54] to cater to the extreme imbalance of a large number of easy background samples vs the sparse foreground samples and the hard background samples. It reduces the loss for the well-trained class such that, whenever the model is good at detecting background, it will reduce its loss and re-emphasize the training on the class and hard background samples. Focal loss can effectively discount the effect of easy negatives, focusing all attention on the hard negative examples.

Focal loss is taken as the loss on the classification subnetwork, applied to all the anchors in each sampled image. The total focal loss is the sum of the focal losses over all the anchors, normalized by the number of anchors assigned to a ground-truth box, and not the total number of anchors.

If $p \in [0, 1]$ is the estimated probability for the label $y = 1$. Let's define p_t as:

$$p_t = \begin{cases} p, & \text{if } y = 1 \\ 1 - p, & \text{otherwise} \end{cases} \tag{4.5}$$

Focal loss is defined as:

$$FL(p_t) = -\alpha_t * (1 - p_t)^{\beta} * log(p_t) \tag{4.6}$$

where $(1 - p_t)^{\beta}$ is the modulating factor to the cross entropy loss and $\beta \geq 0$ is tunable focusing parameter (default: 2). And

$$\alpha_t = \begin{cases} \alpha, & \text{if } y = 1 \\ 1 - \alpha, & \text{otherwise} \end{cases} \tag{4.7}$$

where $\alpha \in [0, 1]$ is weighting factor for addressing class imbalance.

Decoding is done on at most 1k top predictions for each level in the feature pyramid after thresholding the detector confidence at 0.05. NMS (threshold = 0.5) is used as post-processing for all the levels to get the final results.

Table 4.1: Precision, Recall, F1 score, and Accuracy comparison for different positions of CBAM in the network.

CBAM Position	F1 score(%)	Accuracy(%)
without CBAM	86.15	81.07
after FPN	83.58	78.32
before FPN: all(x2-x3-x4)	76.32	74.27
before FPN: x2-x3	83.54	80.09
before FPN: x2-x4	80.2	73.88
before FPN: x3-x4	83.19	76.75
before FPN: x2	85.93	81.16
before FPN: x3	81.56	74.71
before FPN: x4	**86.7**	**82.74**

4.2.7 Experiments and Results

The evaluation of the proposed model was carried out with the Ouhands dataset and the pre-trained ResNet-18 (on ImageNet-1K dataset) was used as the backbone of the model. The hyperparameters of the focal loss were kept the same as in RetinaNet, i.e. $\alpha = 0.25$ and $\beta = 2$. Adam optimizer was used for parameter optimization with a batch size of 2. The learning rate was kept at 10^{-5}. Since transfer learning was used, the network generalized better and showed significantly good results. Furthermore, batch normalization was used that ensure no over-fitting occurs. The confidence score and IoU threshold for the detection is taken as 0.5.

We experimented with the position of the CBAM module in the architecture for optimal performance. The proposed network was trained for 30 epochs. The positions explored are:

(i) after the FPN layer

(ii) after all the stages of ResNet-18 (x2-x3-x4)

(iii) after combinations of the outputs from stages of ResNet-18 (x2-x3, x2-x4, x3-x4)

(iv) after individual outputs from stages of ResNet-18 (x2, x3 and x4)

Table 4.1 shows F1-score and recognition accuracy comparison of networks with CBAM in different positions in the network. Table 4.2 shows the class-wise comparison of the mAP scores.

After experimentation, we chose the architecture with CBAM module after the output of the last stage of ResNet-18 (x4) before passing it to the FPN.

Fig. 4.2 shows a few examples of the detected hand along with the predicted class. The

Table 4.2: mAP score comparison for different positions of CBAM in the network.

CBAM Position	A	B	C	D	E	F	H	I	J	K
without CBAM	0.99	0.98	0.90	0.87	**0.96**	0.76	0.89	0.91	0.88	0.93
after FPN	0.99	0.98	0.89	0.86	0.76	**0.83**	0.86	0.84	**0.90**	0.92
before FPN: all	0.91	0.92	0.84	0.65	0.74	0.69	0.59	0.63	0.70	0.83
before FPN: x2-x3	0.97	0.98	0.90	0.71	0.82	0.78	0.76	0.86	0.85	0.89
before FPN: x2-x4	0.99	0.98	0.80	0.83	0.77	0.76	0.79	0.86	0.88	0.89
before FPN: x3-x4	**1.00**	**0.99**	0.79	0.85	0.92	0.82	0.86	0.89	0.87	0.91
before FPN: x2	0.99	0.98	0.90	0.87	0.93	0.78	0.86	0.87	0.88	0.93
before FPN: x3	0.97	0.96	**0.91**	0.80	0.89	0.76	0.81	0.93	0.86	0.89
before FPN: x4	0.98	0.98	0.88	**0.88**	0.92	0.73	**0.90**	**0.94**	0.89	**0.94**

Fig. 4.2: Hand gesture recognition outputs.

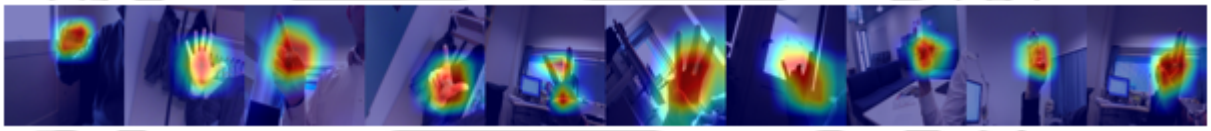

Fig. 4.3: Grad-CAM visualization outputs

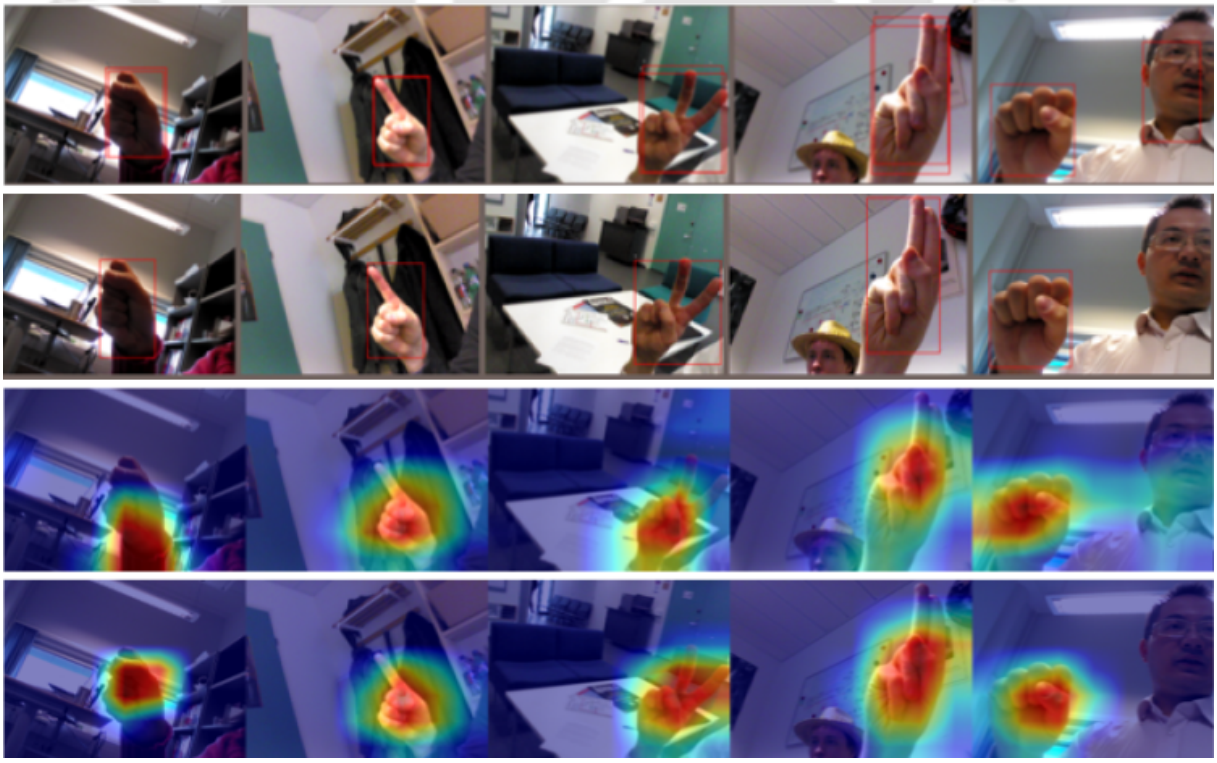

Fig. 4.4: Effects of CBAM on the Network. First row: without CBAM. Second row: with CBAM. Third row: Grad-CAM visualization for without CBAM case. Fourth row: Grad-CAM visualization for with CBAM case.

proposed RetinaNet-CBAM model predicts the hand gestures accurately in varying illumination and contrast conditions as well as against complex backgrounds. Grad-CAM [109] is used for visualizing the network. It uses gradients to calculate the importance of spatial locations in convolution layers. Grad-CAM results show attended regions because it calculates the gradients with respect to each class. The Grad-CAM visualization is computed for the last layer of the fourth stage of ResNet-18. Fig. 4.3 shows the Grad-CAM visualization outputs for the images in Fig. 4.2.

It is observed that without the attention module, the network outputs multiple detections for a single gesture and that too with incorrect classification. The first row of Fig. 4.4 shows a few examples of images where the model detects false positives and multiple gestures and the second row shows their corresponding output from the RetinaNet-CBAM model with the attention module. The third and fourth rows of Fig. 4.4 show the Grad-CAM visualization of the impact of the attention module for detection. The confusion matrix for the testing data was computed and it is shown in Fig. 4.5. The columns represent the predicted class and the rows represent the ground truth.

The proposed method though generates accurate bounding boxes with an F1-score of 86.7%, it uses anchor boxes for detection. The anchor boxes are predetermined bounding boxes of specified height and width placed at each input image location. For each bounding box, the network predicts a confidence score (based on IOU with ground truth) with respect to each class and box offset. Only a small fraction of these anchor boxes overlap sufficiently with the ground truth ones, which are considered positive. All the remaining anchor boxes are considered negative, which introduces an imbalance between the positive and negative anchor boxes, resulting in a negative impact on the detection accuracies. Also, these anchor boxes introduce many hyperparameters and design choices that include the number of boxes, sizes of boxes, and their aspect ratios. This adds even more complications to the model design. Therefore, we developed an anchorless transformer-based hand gesture detection method, which is described in the next section.

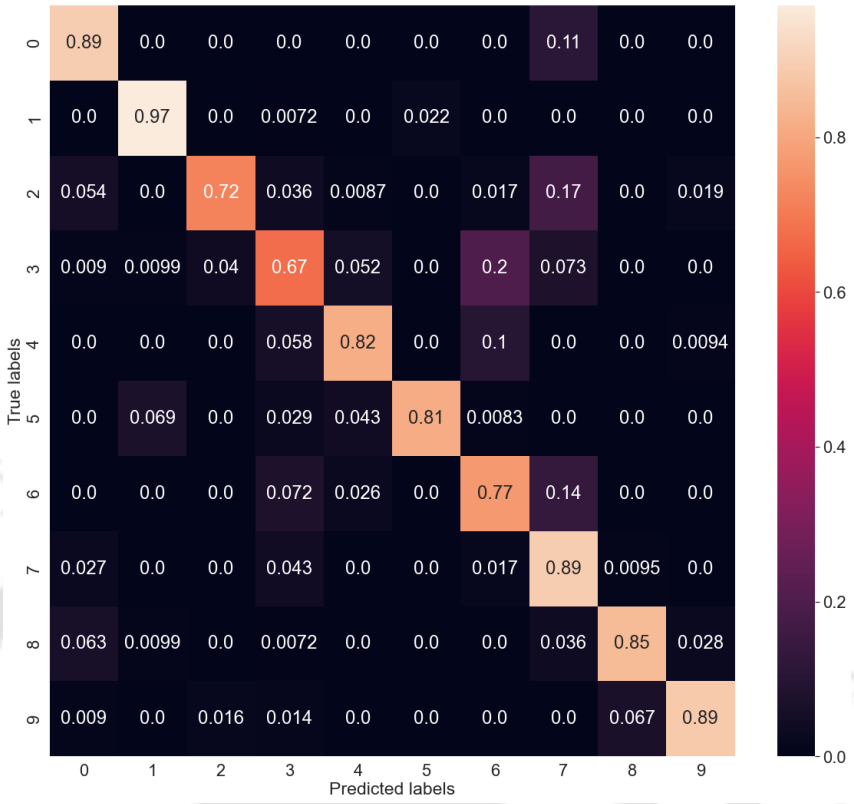

Fig. 4.5: Confusion matrix.

4.3 Method 2: Hand gesture detection based on detection transformer

The detection process comprises locating the hand and classifying the gesture simultaneously, which is achieved using a detection transformer [56]. The intuition for using a transformer is its ability to capture global information independently of anchor boxes and NMS. A transformer can use the attention mechanism to determine the relation between a bounding box's top left corner and the bottom right corner, irrespective of their distance. The architecture of the hand detection module is shown in Fig. 4.6. It constitutes a ResNet-50 backbone that accepts video frames as input images and passes the signal along with position embeddings to the transformer. The backbone captures the local details of the image, and the transformer network processes the image in its entirety. A transformer divides an image into patches to form a sequence. This position embedding indicates the position of the image patches. The position embeddings are calculated as described in [55]. The output from the transformer is propagated simultaneously to a classification and a regression head. The classification head consists of a fully connected layer with softmax, which measures the probability of the hand gesture belong-

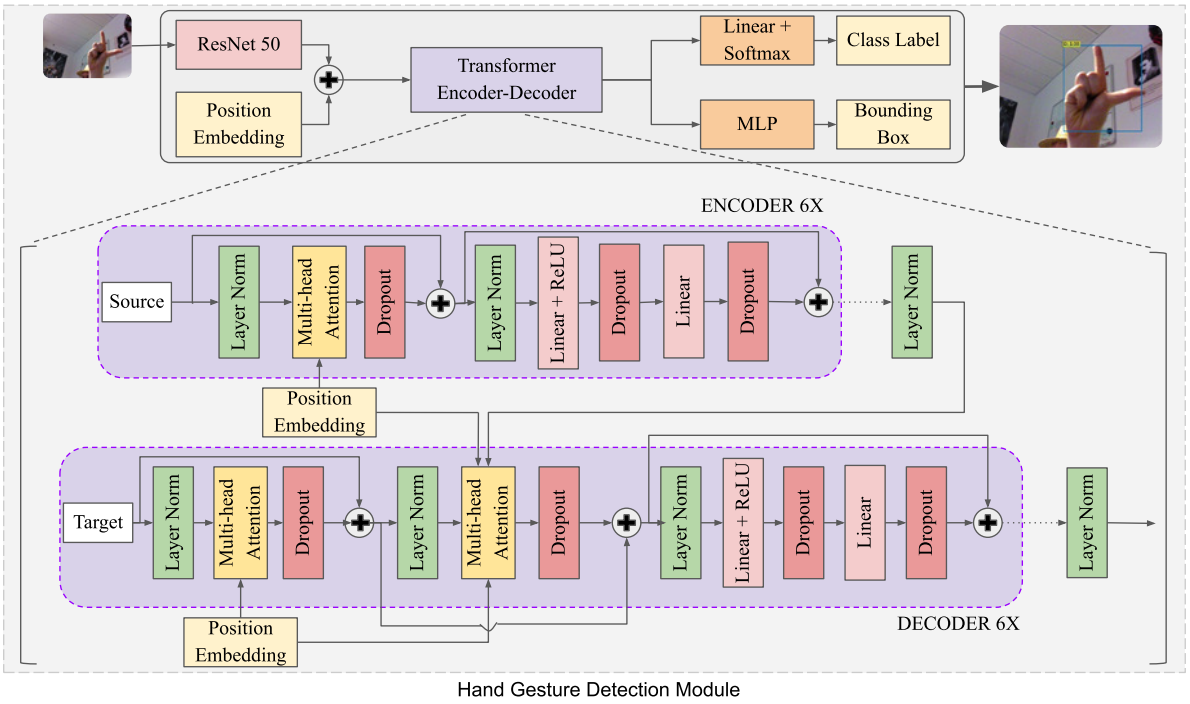

Fig. 4.6: The block diagram depicts the hand gesture detection module.

ing to each class, i.e., the classification score. The maximum classification score determines the class label. The regression head is a multilayer perceptron (MLP), which outputs the bounding box coordinates of the hand gesture.

4.3.1 Transformer Network

For the encoder part of the transformer network, the feature map from the backbone and the position embedding [55] produce a source sequence S, which propagates through a multi-head attention module and a feed-forward neural network to produce the encoder output. Six encoder units work sequentially. Their structure is shown in Fig. 4.6.

Multi-head attention (MHA) [55] is the backbone of the transformer, and can be defined as

$$MHA(S) = [A^1(S) \oplus ... \oplus A^8(S)]\mathbb{M}^p \tag{4.8}$$

where single attention head,

$$A^{(\cdot)}(S) = \frac{exp\left(\frac{\mathbf{M}_{qi}^{\mathrm{T}}\mathbf{M}_{kj}}{\sqrt{d/\eta}}\right)}{\sum_{j=1}^{N_{kv}} exp\left(\frac{\mathbf{M}_{qi}^{\mathrm{T}}\mathbf{M}_{kj}}{\sqrt{d/\eta}}\right)}$$

i represents the query index, and j represents the key-value index. The query, key, and values

of an attention mechanism [55] are arranged into matrices denoted by $\mathbf{M}_q, \mathbf{M}_k$, and $\mathbf{M}_v$, respectively. η represents the number of attention heads, and d represents the dimensionality of the embeddings. $\mathbb{M}^p$ is a projection matrix. N_{kv} represents the length of the key-value sequence. For multi-head self-attention, the query sequence equals the key-value sequence. The self-attention mechanism is intended to help the model focus on image regions relevant to making a prediction. It also helps preserve temporal dependencies in the sequence.

The decoder part has two MHA blocks. The first uses self-attention, and the second uses encoder-decoder attention. A target sequence, which is a learned position embedding, is fed to the first MHA. Then, this signal is passed to the second MHA with the output of the encoder. There are n embeddings, which encode the contextual information of the object and its relation to the background. The decoder processes this using a bipartite relation and outputs a signal deciphered by a feed-forward network to produce bounding box coordinates and classification scores. Just like the encoder, the decoder also has six units. For regularization, the encoder-decoder employs layer normalization and dropout layers.

The feed-forward network comprises an MLP that estimates the bounding box's normalized center (centroid) coordinates, height, and width. A linear layer with softmax activation predicts the class of the gesture localized by the bounding box.

4.3.2 Loss Function

This detection process generates n bounding box predictions, which is always greater than the actual number of bounding boxes (m) for the objects present in the image. As a result, the ground truth set is padded with $n - m$ "no object" labels to perform bipartite (one-to-one) matching between the predicted set (y) and the ground truth set (y^t). The optimal index ($\hat{I}$) of the element in the predicted set, corresponding to the minimum cost of matching the two sets, is obtained from

$$\hat{I} = \arg\min_{I \in S_n} \sum_{i=1}^{n} [-\mathbb{1}.\{\theta_i^t \neq \phi\}\, \mathbf{p}_{I(i)}(\theta_i^t) + \mathbb{1}.\{\theta_i^t \neq \phi\}\, \ell_{GIOU}(\psi_i^t, \psi_{I(i)})] \tag{4.9}$$

A ground truth set element i is represented as (θ_i^t, ψ_i^t), where θ_i^t is the actual class label and ψ_i^t is the actual bounding box coordinate. $\mathbf{p}_{I(i)}(\theta_i^t)$ is the predicted class probability of class θ_i^t for the $I(i)$th indexed element of the predicted set. In (4.9), ℓ_{GIOU} represents the generalized intersection over union loss between the actual bounding box ψ_i^t and predicted bounding box $\psi_{I(i)}$. $\mathcal{S}_n$ is the set containing the indices of the $n\,(=100)$ predictions.

The Hungarian loss function ($\ell_{\mathcal{H}}$) [56] for training the network is

$$\ell_{\mathcal{H}}(y^t, y) = \sum_{i=1}^{n}[-log\, \mathbf{p}_{\hat{I}(i)}(\theta_i^t) + \mathbb{1}.\{\theta_i^t \neq \phi\}[\kappa_1\, \ell_{GIOU}(\psi_i^t, \psi_{\hat{I}(i)}) + \kappa_2\, \ell_{LAD}(\psi_i^t, \psi_{\hat{I}(i)})]] \tag{4.10}$$

where,

$$\ell_{GIOU}(\psi_i^t, \psi_{\hat{I}(i)}) = 1 - \frac{|\psi_i^t \cap \psi_{\hat{I}(i)}|}{|\psi_i^t \cup \psi_{\hat{I}(i)}|} + \frac{|\psi'_{\hat{I}(i)} - \psi_i^t \cup \psi_{\hat{I}(i)}|}{|\psi'_{\hat{I}(i)}|} \tag{4.11}$$

$$\ell_{LAD}(\psi_i^t, \psi_{\hat{I}(i)}) = \|\psi_i^t \cap \psi_{\hat{I}(i)}\|_1 \tag{4.12}$$

ℓ_{LAD} is the least absolute deviation loss and $\kappa_1, \kappa_2 \in \mathbb{R}$. $\psi'_{\hat{I}(i)}$ is the largest box containing ψ_i^t and $\psi_{\hat{I}(i)}$.

4.3.3 Experiments and Results

This section covers the experimental evaluation of the system on two benchmark datasets: Ouhands [9] and NUS [8]. It discusses the experiments performed and the results obtained. The work was performed in Python on an Nvidia Tesla P100 GPU. To train the hand detection module, Adam optimization was used with a learning rate of 0.0001 and a batch size of 2.

For training the hand detection model, we used transfer learning. As each dataset has fewer samples than required to train a deep neural network, instead of training from scratch, the pre-trained weights of DETR [56] were finetuned by freezing the transformer weights and retraining the classification and regression heads. This helps the network learn spatial hierarchies, that is, universal and repurposable characteristics of the data, shared with the much larger COCO [46] dataset. The training process was performed for 50 epochs. The training was performed with 65% of the data, and the remaining 35% were equally divided between validation and testing. The testing set was isolated from the training and validation sets, and no testing data were

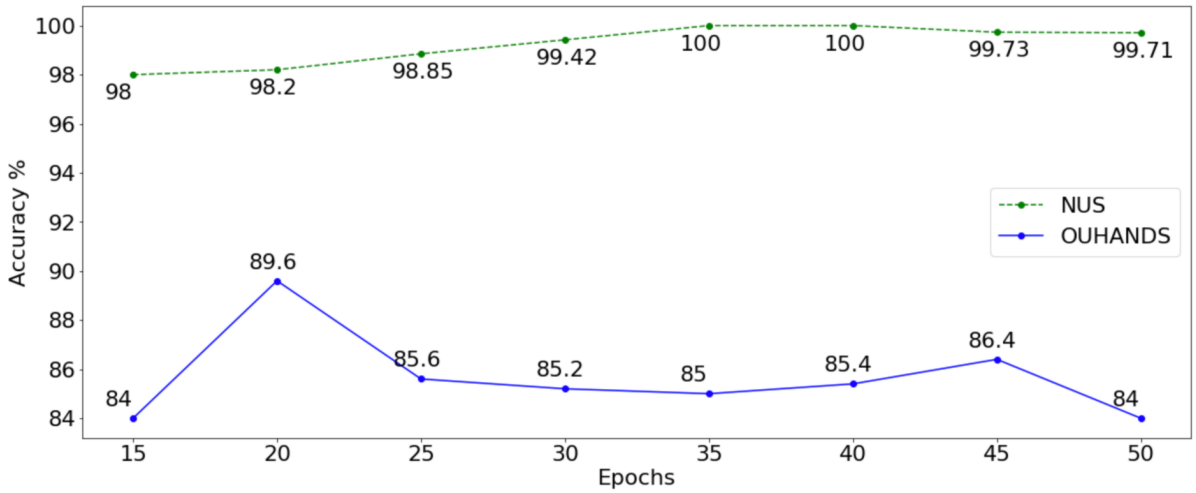

Fig. 4.7: A plot showing the validation accuracy score of the detection model for different epochs during training for the OUHANDS and the NUS datasets.

exposed during training.

For both datasets, the model's performance by epoch is shown in Fig. 4.7. The NUS model likely performed better than the OUHANDS model because the former contained fewer variations in illumination and hand size. The maximum change in the hand's bounding box area for the OUHANDS dataset was 0.52 square pixels, and the NUS dataset was 0.24 square pixels, indicating higher scale variability for OUHANDS. Regarding illumination, the mean difference between the intensity histogram's maximum and minimum value for the images of the datasets was 0.054 and 0.028 for the OUHANDS and NUS datasets, respectively, which indicates greater variation in OUHANDS. Based on validation accuracy, the model trained at the 20th and 35th epoch were used to test the OUHANDS and NUS datasets, respectively. The validation performance of the detection model is shown in Fig. 4.7.

The performance measures for the two datasets are the accuracy score and F1-score (defined by (4.1) and (2.12), respectively). The confusion matrices for the two datasets are shown in Fig. 4.8. Table 4.3 and Table 4.4 show the performance of the proposed hand gesture detection model for both datasets. The proposed method achieves an accuracy of 89.6% for OUHANDS and 100% for NUS. Similarly, the two datasets' F1-scores are 89.9% and 100%, respectively. These results demonstrate the system's effectiveness. The training and testing time details are as follows:

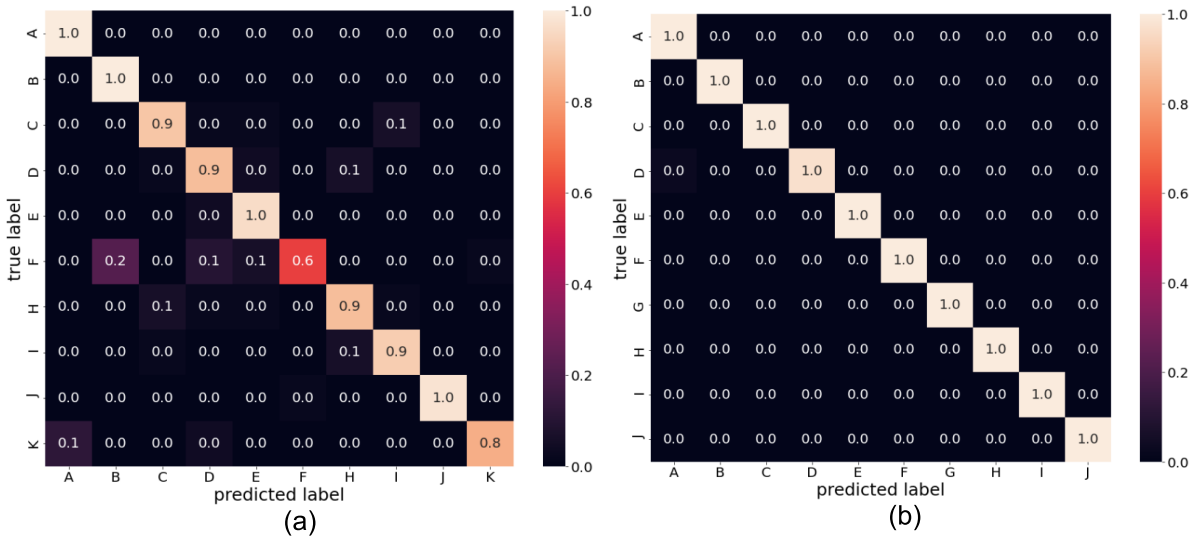

Fig. 4.8: Confusion matrix after testing on the (a) OUHANDS dataset and the (b) NUS dataset.

Table 4.3: Performance of the detection model on OUHANDS dataset

CLASS	PRECISION	RECALL	F1-SCORE
A	0.89	1.00	0.94
B	0.82	1.00	0.90
C	0.90	0.9	0.90
D	0.80	0.88	0.84
E	0.87	0.96	0.91
F	0.97	0.60	0.74
H	0.88	0.88	0.88
I	0.92	0.92	0.92
J	1.00	0.98	0.99
K	0.98	0.84	0.90
Avg	**0.903**	**0.896**	**0.899**

Table 4.4: Performance of the detection model on NUS dataset

CLASS	PRECISION	RECALL	F1-SCORE
All categories	1	1	1
Avg	**1**	**1**	**1**

- Training time: for OUHANDS = 7 hours, for NUS = 5 hours.
- Inference time for a single image = 56 ms.
- The number of trainable parameters = 41.28 M.

A comparison of the inference times of different methods and the proposed method is shown in Table 4.5. Fig. 4.9 and Fig. 4.10 depict the accurate localization of the hand gesture via the bounding box and the classification of the localized gesture.

Robustness: The detection results of the model for input images with human faces and other

Table 4.5: Inference time comparison for different methods.

Methods	Inference time per image
Faster RCNN	230 ms
Faster RCNN + CPF	130 ms
Retina Net	64 ms
Bhaumik et al. [105]	6.4 s
Sharma et al. [59]	97 ms
Bose et al. [57]	140 ms
Proposed	**56 ms**

Fig. 4.9: Detection of hand gestures from the OUHANDS dataset in the presence of a human face (top left), background clutter (top right), illumination variation (bottom left), and change in hand orientation (bottom right).

objects in the background, extreme variations in illumination, and hands of different shapes and sizes are shown in Fig. 4.9 and Fig. 4.10 to demonstrate the method's detection capability. Even when the hand is not upright (or affine transformed), it is detected successfully. Detection is effective because the attention mechanism is trained to attend to the hand region, making it impervious to background clutter. Moreover, using affine-transformed data and normalization make the model robust to geometric transformations and variations in illumination and scales.

State-of-the-art comparison: We evaluated the system's effectiveness by comparing it with state-of-the-art methods. Table 4.6 shows that the proposed work performs better than state-of-the-art methods. The proposed work uses only one input modality, an RGB image. It is a single-

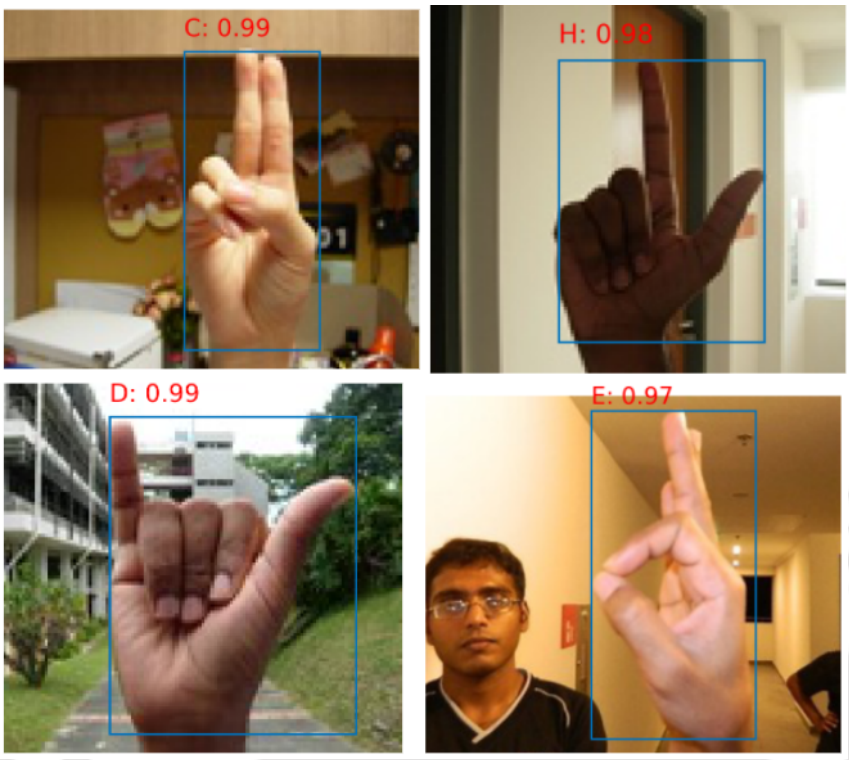

Fig. 4.10: Detection of hand gestures from the NUS dataset in the presence of background clutter (top left), illumination variation (top right), an outdoor environment (bottom left), and the human face with a different pose angle (bottom right).

stage end-to-end approach, for which the network's input is the color image, and its output is the class prediction and the bounding box coordinates. However, alternative approaches either use a two-stage network or multiple input modalities. This work also compares its results with those of a baseline [9] and to other object detection frameworks [57], as detailed in Table 4.6. The results demonstrate the system's reliable performance relative to others.

4.4 Method 3: An anchorless end-to-end hand gesture detection using CenterNet

In this method, another anchorless end-to-end, one-stage hand detection-based approach based on CenterNet [112] is proposed. It detects the object as a point, i.e., the center point of the bounding box encompassing the object, which regresses the object size. This eliminates the need for anchor boxes in CenterNet.

This proposed model uses an attention mechanism with CenterNet architecture. It consists of an encoder-decoder network with DA-Net module [28] (for two-way attention mechanism)

Table 4.6: Comparative study of the state-of-the-art works and the proposed work.

Papers	Objective	Dataset	Acc (%)	F-score (%)
Dadashzadeh *et al.* [16]	Segmentation & Classification	OUHANDS	87.8	88.1
Bose *et al.* [57] (two-stage network)	Localization & Classification	NUS	97.1	97.98
Bose *et al.* [57] (single-stage network)	Localization & Classification	NUS	97.17	97.17
Aditya *et al.* [110]	Classification	NUS	94.7	94.26
Pisarady *et al.* [8]	Classification	NUS	94.36	94.9
Sharma *et al.* [59]	Classification	NUS	96.62	97.43
Bhaumik *et al.* [105]	Classification	NUS	97.78	97.28
Sahoo *et al.* [106]	Classification	NUS	94.8	94.07
Tan *et al.* [111]	Classification	NUS	98.4	98.4
Bhaumik *et al.* [104]	Classification	OUHANDS	65.1	64.56
		NUS	98.75	97.0
Baseline [9]	Localization & Classification	OUHANDS	83.25	50
Retinanet [54]	Localization & Classification	OUHANDS	87.5	87
Faster R-CNN+CPF	Localization & Classification	NUS	95	95.06
Faster R-CNN	Localization & Classification	NUS	93.1	93
MS-FRCNN	Localization & Classification	NUS	87	87.12
MS-RFCN	Localization & Classification	NUS	78.23	76.14
Proposed	Localization & Classification	OUHANDS	**89.6**	**89.9**
		NUS	**100**	**100**

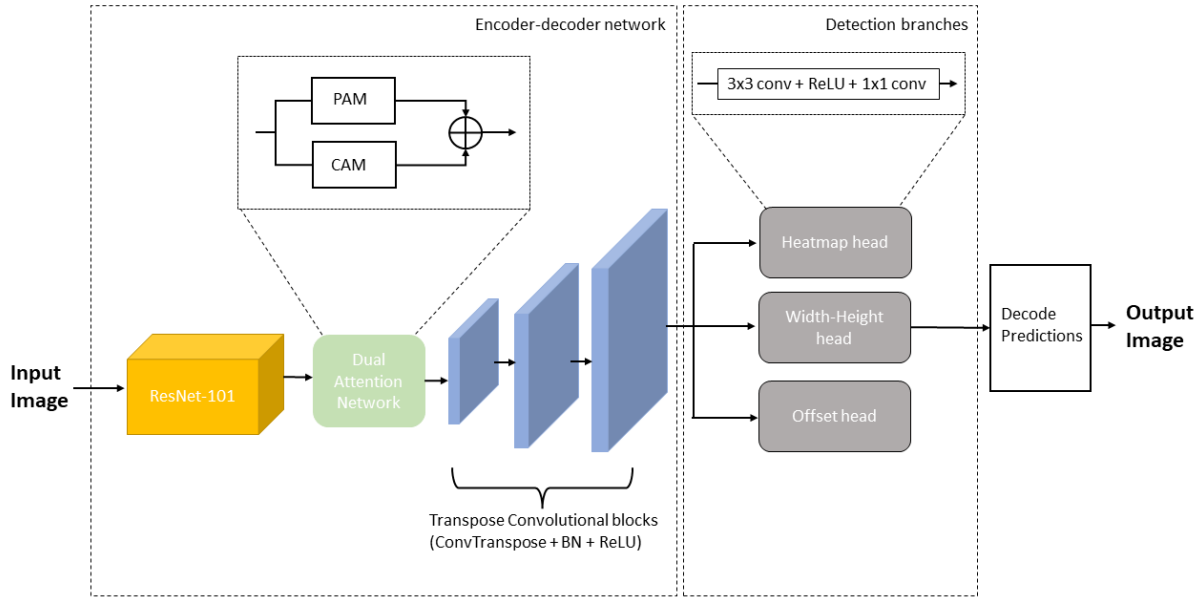

Fig. 4.11: Complete Architecture of the proposed model

followed by three detection branches, namely, Heatmap head, Width-height head and Offset head. The Encoder converts the input image to a low-resolution feature map and is fed to DA-Net to contextually relate local features with global ones. The attention network's output is upsampled and passed as input to the three detection branches to predict the center point, width, height, and offsets in the center point, respectively. Finally, these predictions are decoded to get the bounding box outputs. Fig. 4.11 shows the architecture of our proposed model.

The details of each module in the proposed network are explained in the following subsections.

4.4.1 Encoder

The purpose of the encoder is to extract the feature maps. It downsamples the input image using a series of convolutional layers and extracts high semantic features. ResNet-101 is used as the backbone of our network. In addition to feature encoding, it reduces the vanishing gradients issue encountered in deep neural architectures by using skip connections between the layers. These skip connections preserve the features that are extracted in the earlier layers by adding them to the current output. The last layer of the ResNet-101 block is fed to the DA-Net module to capture global inter-dependencies across spatial and channel dimensions.

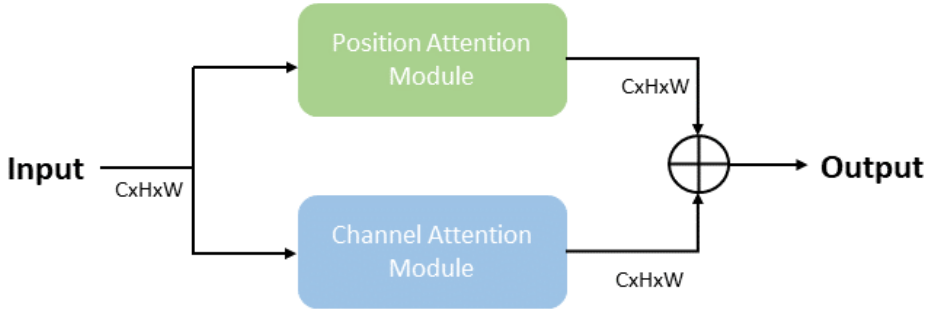

Fig. 4.12: Dual Attention Network (DA-Net)

4.4.2 Dual Attention Network (DA-Net)

The DA-Net module tells the network 'what' and 'where' to look at. The local features extracted by traditional convolution layers could lead to misclassification and poor localization as they fail to model contextual relationships over local features. In order to avoid this problem, DA-Net is added to the output of the encoder. It comprises two sub-modules connected in parallel: Position Attention Module (PAM) and Channel Attention Module (CAM). These modules capture the long-range contextual information along spatial and channel directions, respectively. The outputs of these modules are aggregated to obtain better feature representations for pixel-level prediction. The architecture of DA-Net is shown in Fig. 4.12 where H, W, and C denotes the input feature map's height, width, and channels, respectively.

4.4.2.1 Position Attention Module (PAM)

This module captures long range contextual information across spatial direction. As shown in Figure 4.13, the input feature map $A \in \mathbb{R}^{(C\times H\times W)}$ is fed to convolution layers, which produces two new feature maps, B and C, where $\{B, C\} \in \mathbb{R}^{(C\times H\times W)}$. Then, they are reshaped to $\mathbb{R}^{(C\times N)}$, where $N = H\times W$ denotes the pixels count. After that, matrix multiplication is done between the transpose of C and B, and the result is passed through a softmax layer to generate the spatial attention map $S \in \mathbb{R}^{(N\times N)}$.

$$s_{ji} = \frac{\exp\left(B_i.C_j\right)}{\sum_{i=1}^{N}\exp\left(B_i.C_j\right)} \tag{4.13}$$

where s_{ji} is the impact of i^{th} position on j^{th} position.

Meanwhile, the feature map A is fed to a convolution layer to produce a feature map $D \in$

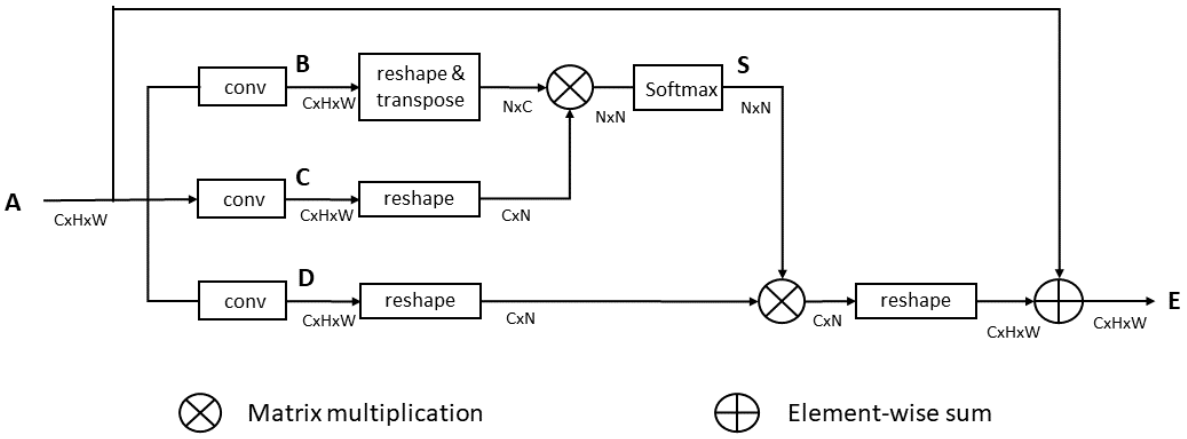

Fig. 4.13: Position Attention Module

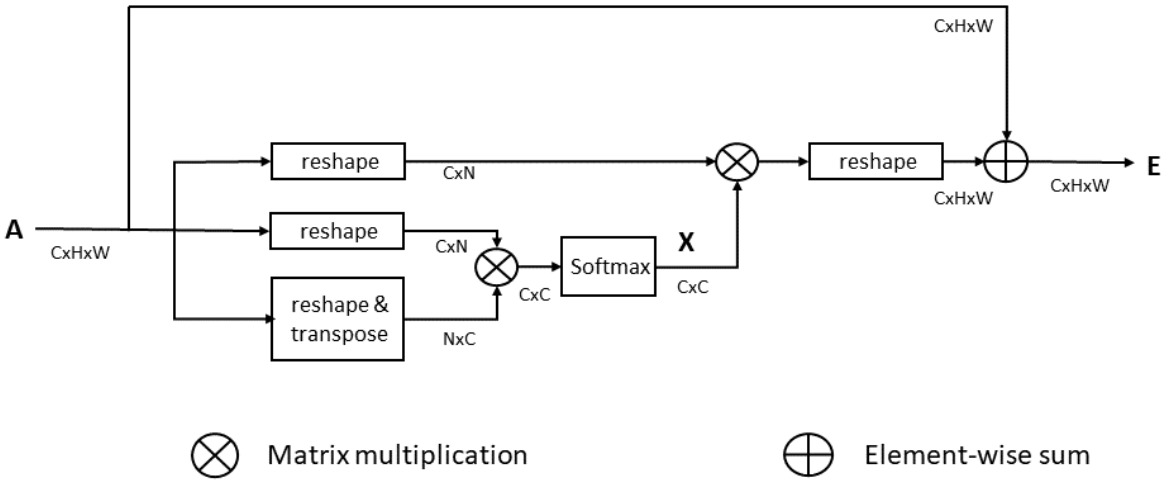

Fig. 4.14: Channel Attention Module

$\mathbb{R}^{(C\times H\times W)}$, which is reshaped to $\mathbb{R}^{(C\times N)}$. Matrix multiplication is done between D and transpose of S and the outcome is resized to $\mathbb{R}^{(C\times H\times W)}$. The result is multiplied with a learnable parameter α and then added (element-wise) to A to obtain the final output $E \in \mathbb{R}^{(C\times H\times W)}$.

$$E_j = \alpha \sum_{i=1}^{N} (s_{ji} D_i) + A_j \tag{4.14}$$

4.4.2.2 Channel Attention Module (CAM)

The purpose of CAM is to explicitly model the interdependencies across channels. As shown in Figure 4.14, the input feature map $A \in \mathbb{R}^{(C\times H\times W)}$ is reshaped to $\mathbb{R}^{(C\times N)}$. Matrix multi-

plication is performed between A and transpose of A and the result is passed through a softmax layer to generate the Channel Attention Map $X \in \mathbb{R}^{(C\times C)}$.

$$x_{ji} = \frac{\exp(A_i.A_j)}{\sum_{i=1}^{C}\exp(A_i.A_j)} \tag{4.15}$$

where x_{ji} is the impact of i^{th} channel on j^{th} channel.

Then, matrix multiplication is performed between the transpose of X and A, and the result is reshaped to $\mathbb{R}^{(C\times H\times W)}$. The result is multiplied by a learnable parameter ν and then added (element-wise) to A to obtain the final output $E \in \mathbb{R}^{(C\times H\times W)}$.

$$E_j = \nu\sum_{i=1}^{C}(x_{ji}A_i) + A_j \tag{4.16}$$

Finally, the output of PAM and CAM are aggregated by performing element-wise sums to take complete advantage of rich contextual information across spatial and channel directions.

4.4.3 Decoder

The decoder consists of a series of transpose convolution blocks connected in a cascade. When the input image is passed through the encoder, it gets downsampled, and a lot of spatial information is lost. In order to scale up the resolution, the output of the DA-Net is fed to the decoder. The decoder upsamples its input to one-fourth of the dimension of the input image.

4.4.4 Detection branches

The detection branches are responsible for predicting the center point, width, and height of the bounding box encompassing the hand portion. The decoder output is simultaneously fed to these three branches to obtain three sets of feature map predictions. Let us consider H and W as the height and width of input image, C as the number of classes and R as the downsampling factor.

- Heatmap head: It predicts the location and confidence scores of hand poses' center key-points (center scores) corresponding to different classes. It consists of 3×3 convolution, ReLU, 1×1 convolution, and sigmoid layers. The heatmap head outputs a $(\frac{W}{R}\times\frac{H}{R}\times C)$ dimensional feature map. This feature map consists of center scores of the detection with

respect to each class and each location. Peaks in the heatmap correspond to hand centers.

- Width-height head: It predicts the width and height of the bounding box for each center point. It consists of 3×3 convolution, ReLU, and 1×1 convolution layers. Width-height head outputs a $(\frac{W}{R} \times \frac{H}{R} \times 2)$ dimensional feature map from which width and height values are obtained.
- Offset head: When the input image is passed through the encoder-decoder network, it gets downsampled by a factor R. That is, a point (x, y) in the input image gets mapped to $(\lfloor \frac{x}{R} \rfloor, \lfloor \frac{y}{R} \rfloor)$ on the heatmap. When these locations on the heatmap are remapped to the input image, some precision may be lost for the predicted center keypoints. In order to recover this, the offset head is used to predict the offsets in the center keypoint along the horizontal and vertical directions. It consists of 3×3 convolution, ReLU, and 1×1 convolution layers. Offset head outputs a $(\frac{W}{R} \times \frac{H}{R} \times 2)$ dimensional feature map from which offsets are obtained. All the classes share the same offset values.

4.4.5 Loss function

The loss function consists of three components. The first component is penalty-reduced focal loss, the second is size loss, and the third is offset loss.

- Penality-reduced focal loss: It is the modified version of focal loss [54] calculated on the heatmap head. Each ground truth center point has one positive prediction, and the remaining predictions are negative. The penalty for negative predictions is lowered within a radius of the positive prediction rather than being applied equally to negative and positive locations.

  Let N be the number of keypoints in the image, W and H are the width and height of the input image, R be the downsampling factor (set to 4 in our case), C be the number of classes, $p \in \mathbb{R}^2$ is the ground truth keypoint of class c, where $c \in \{0, 1, 2, .., C-1\}$, a low-resolution equivalent, $\hat{p} = \lfloor p/R \rfloor$ is computed. All the ground truth keypoints are encoded with Gaussian bumps using an unnormalized 2D Gaussian, $\exp(-\frac{(x-\hat{p}_x)^2+(y-\hat{p}_y)^2}{2{\sigma_p}^2})$ to obtain a ground truth heatmap augmented with unnormalized Gaussians, $Y_{xyz} \in [0, 1]^{(\frac{W}{R} \times \frac{H}{R} \times C)}$ where σ_p is an object size-adaptive standard deviation [113].

The penalty-reduced focal loss is defined in (4.17).

$$L_k = \frac{-1}{N} \sum_{xyc} \begin{cases} (1 - \hat{Y}_{xyc})^{\upsilon} \log(\hat{Y}_{xyz}), if\ Y_{xyc} = 1 \\ (1 - Y_{xyc})^{\tau} (\hat{Y}_{xyc})^{\upsilon} \log(1 - \hat{Y}_{xyc}),\ otherwise \end{cases} \tag{4.17}$$

where υ and τ are hyperparameters.

- Size loss: It is the L1 loss calculated on the width-height head. If $({x_1}^{(k)}, {y_1}^{(k)}, {x_2}^{(k)}, {y_2}^{(k)})$ is the bounding box of object k with category c, $s_k = ({x_2}^{(k)} - {x_2}^{(k)}, {y_2}^{(k)} - {y_1}^{(k)})$ is regressed to the object size for each object k. The same prediction $\hat{S} \in \mathbb{R}^{(\frac{W}{R}, \frac{H}{R}, 2)}$ is used for all object categories. The size loss is defined in (4.18).

$$L_{size} = \frac{1}{N} \sum_{k=1}^{N} |\hat{S}_k - s_k| \tag{4.18}$$

- Offset loss: It is the L1 loss calculated on the offset head. In order to account for the discretization error caused by downsampling, a local offset $\hat{O} \in \mathbb{R}^{(\frac{W}{R}, \frac{H}{R}, 2)}$ is predicted for each center point. All classes share the same offset prediction. The offsets are trained using the loss given in (4.19).

$$L_{off} = \frac{1}{N} \sum_{p} |\hat{O}_{\hat{p}} - (\frac{p}{R} - \hat{p})| \tag{4.19}$$

The overall loss function is given as

$$L_{det} = L_k + \lambda_{size} L_{size} + \lambda_{off} L_{off} \tag{4.20}$$

where λ_{size} and λ_{off} are hyperparameters.

4.4.6 Decoding predictions

The predictions made by the three detection branches are decoded to obtain top k (which is 100 in our case) bounding boxes along with their class ids and confidence scores after thresholding the score at 0.2. Finally, the bounding box corresponding to the class with the highest confidence score is retained, and others are discarded. No post-processing like NMS is required for the predicted bounding boxes as they are estimated from the local peaks in the keypoint heatmap.

4.4.7 Experiments and Results

The proposed model is tested on Ouhands [9] and NUS [8] datasets. The loss functions' hyperparameters are kept same as in CenterNet, and the optimizer used for training is Adam with a batch size = 2 and learning rate = 10^{-4}. To avoid overfitting and facilitate model generalization, online data augmentation is used, where data is augmented during training by applying translation, scaling, horizontal flipping, rotation, and crop transformations.

The model is trained from scratch on Ouhands for 70 epochs and NUS for 40 epochs (it converged earlier). Figs. 4.15 and 4.16 show the output images with detected hand region and predicted hand gesture class on Ouhands and NUS hand gesture datasets, respectively. It can be seen that our model can detect the hand gestures accurately in complex backgrounds for different hand sizes and shapes. The confusion matrix for both the datasets are shown in Fig. 4.17.

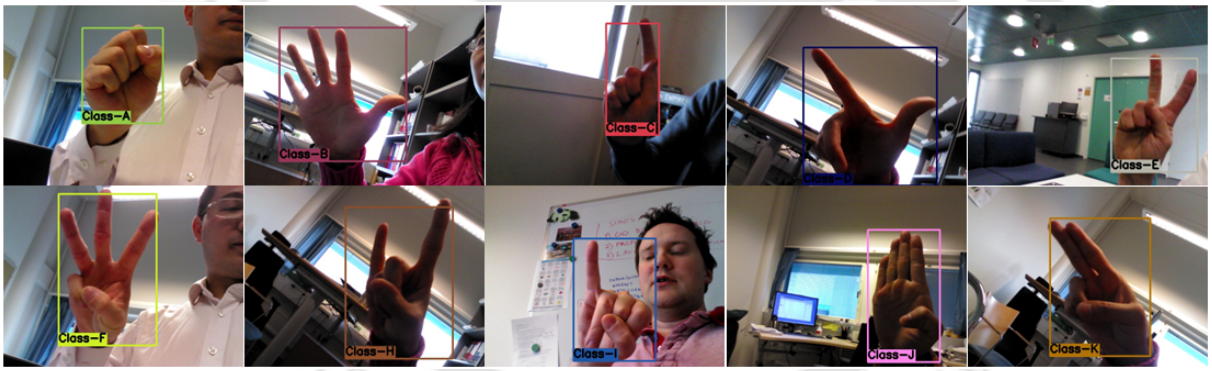

Fig. 4.15: Hand gesture detection outputs on Ouhands datsset.

Fig. 4.16: Hand gesture detection outputs on NUS datsset.

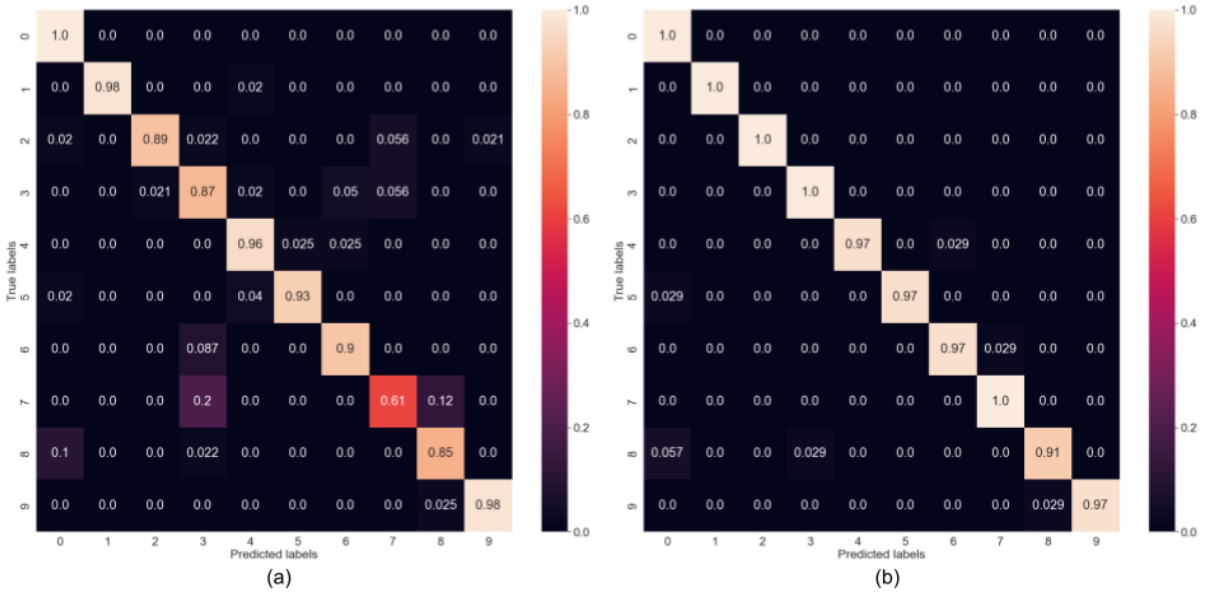

Fig. 4.17: Confusion matrix after testing on the (a) OUHANDS dataset and the (b) NUS dataset.

Table 4.7: F1-score and mAP comparison for Ouhands dataset

Backbone type	**F1-score**	**mAP**
ResNet-50 [107]	79.52%	73.62%
MobileNet [114]	75.47%	70.73%
Hourglass-52 [115]	73.65%	68.39%
DenseNet-201 [116]	82.21%	75.15%
ResNet-101	82.37%	75.73%
ResNet-101 + DA-Net	**84.40%**	**79.50%**

We evaluated the model with and without the DA-Net module to analyze its impact. As shown in Tables 4.7 and 4.8, with the addition of the DA-Net module, the mean F1-score values improved by 2% and 1.6% for the two datasets, respectively. Table 4.7 also compares the proposed model with other models. Since ResNet-101 was the second-best among the models, it was chosen for comparison with the proposed model to determine the best performing model on the NUS dataset (shown in Table 4.8).

These results show the effectiveness of this CenterNet-based model and it presents itself as a strong contender for hand gesture detection.

Table 4.8: F1-score and mAP comparison for NUS dataset

Backbone type	F1-score	mAP
ResNet-101	97.24%	96.28%
ResNet-101 + DA-Net	**98.83%**	**98.00%**

4.5 Which of the three methods to choose?

The three methods produce an acceptable hand localization results as observed in the qualitative results. However, the recognition accuracy of the three methods for the Ouhands dataset are 86.7 %, 89.6 %, and 84.4 %, respectively. Therefore, method 2 has the best accuracy followed by method 1 and method 3. Moreover, method 1 is an anchor-based method. The anchor boxes represents prior knowledge of bounding boxes of specified dimensions positioned at various locations within each input image. While a small portion of these boxes overlaps sufficiently with the ground truth boxes and is labelled as positive, the rest are labelled as negative, creating an imbalance that can impact detection accuracy. Furthermore, the use of anchor boxes introduces various hyperparameters and design choices, such as the number, size, and aspect ratios of the boxes, adding complexity to model design. Taking these considerations into account, an anchorless approach is selected, and based on its recognition accuracy, method 2 is deemed suitable for implementation in hand-control interfaces.

4.6 Contribution

- This detection transformer-based method is the first to integrate a detection transformer with hand gesture detection to the best of our knowledge, estimating the location of the hand in a frame and its classification score.
- Three hand detection methods were proposed– one works using prior anchor box information and the other two are anchorless detectors. Among the anchorless approaches, one utilized the benefits of CNN and transformer, while the other regresses a bounding box encompassing the hand, considering it as a point.

4.7 Summary

Hand gesture recognition on hand-segmented images typically follows a two-stage approach, yielding accurate results but introducing some latency and a reliance on the segmentation stage. Performing localization and recognition simultaneously eliminates this dependency and time lost during transitions between stages. Hand gesture detection adopts an end-to-end approach, simultaneously localizing the hand with a bounding box and predicting the hand gesture class. This study proposes three methods for hand gesture detection. The first method employs an end-to-end network based on the RetinaNet architecture with a convolutional block attention module to enhance performance. The second method utilizes a detection transformer to encode global information and reduce the frequency of spurious detections via bipartite matching. The third method regresses a bounding box around the hand by treating it as a point, employing the CenterNet architecture with parallel spatial and channel attention modules. While the first method relies on prior bounding box information, leading to an imbalance between positive and negative boxes, the second and third methods do not, rendering them more favorable. Moreover, method 2 exhibits superior recognition results, making it the chosen approach for application in hand gesture-based interfaces, as discussed in the subsequent chapter.

5

Hand Keypoints Detection

Contents

5.1 Introduction

Hands provide a natural medium for interaction with applications ranging from human-robot interaction to sign language, augmented and virtual reality, and home automation [99]. Precise hand pose estimation, crucial for smooth interaction, enables accurate hand movement and orientation estimation. Thus, hand pose estimation holds immense potential across a vast scope of applications.

Much research on hand pose estimation has been conducted with depth cameras [117, 118]. However, their performance degrades beyond controlled indoor environments; they are sensitive to changing lighting conditions, such as direct sunlight, which can interfere with their infrared sensors [65]. Moreover, the hardware comes at a premium. Leap motion sensors are also considered for pose estimation but they are less reliable due to their limited detection range and unreliable gesture registration [66]. By contrast, color cameras capture color and texture information, adding to the spatial relationship of keypoints, especially, under variations in lighting. They are also low-cost and widely available [68]. Data gloves [62] and other wearable devices for hand pose estimation restrict movement. For many applications, they can be cumbersome, fatigue-inducing, and uncomfortable. Considering these factors, a color camera is chosen as the input device. Nevertheless, using color cameras for pose estimation has drawbacks. Background clutter, variations in illumination, and self-occlusion impede system performance. We propose a method of tackling these limitations, and our results are encouraging.

This work primarily focuses on precise 2D hand keypoint detection (HKD) because it is essential for hand pose estimation [18, 68].

The block diagram of the proposed HKD architecture is shown in Fig. 5.1. A color image

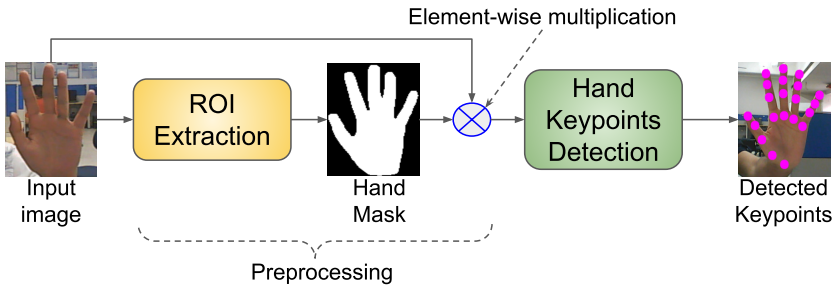

Fig. 5.1: The block diagram of the proposed hand keypoint detection approach.

undergoes pre-processing to extract the hand. Pre-processing is intended to remove background clutter, variations in illumination, and other skin regions, resulting in an image containing the hand with a uniform background. It begins with region of interest (ROI) extraction, which applies segmentation method discussed in chapter 2 to obtain the hand region. The hand mask obtained by ROI extraction is multiplied element-wise by the input image and fed to the proposed HKD model. The HKD model, equipped with a novel attention block, detects 21 hand keypoints, generated as heatmaps.

5.1.1 Intuition and Objective

HKD remains challenging despite extensive research due to the hand's intricate morphology, finger proximity, occlusion, complex backgrounds, and variable lighting conditions. Though advancements have been made in general human pose estimation, these innovations often fall short when applied to HKD [69]. Current methods struggle to achieve high precision, reliability, and occlusion resilience, partly resulting from sensor limitations. Moreover, HKD has a crucial role in a few hand gesture-operated interfaces. The proper operation of such interfaces depends on precise detection of the hand keypoints in challenging conditions.

Therefore, the goal is to develop an efficient and precise HKD method to generalize across different scenarios, overcoming the limitations of current approaches and assisting in the smooth operation of hand gesture-based interfaces.

5.2 Methodology

This section details the proposed approach for hand keypoint detection. First, the hand, our region of interest, is extracted from the input color image. The resulting image with background clutter removed, is fed to the hand keypoint detection model to estimate the keypoint locations.

5.2.1 ROI extraction

The ROI extraction stage employs the CNN-transformer architecture (discussed in chapter 2) to obtain the hand segmented masks. The requirement of this stage is discussed in the section experiments and results.

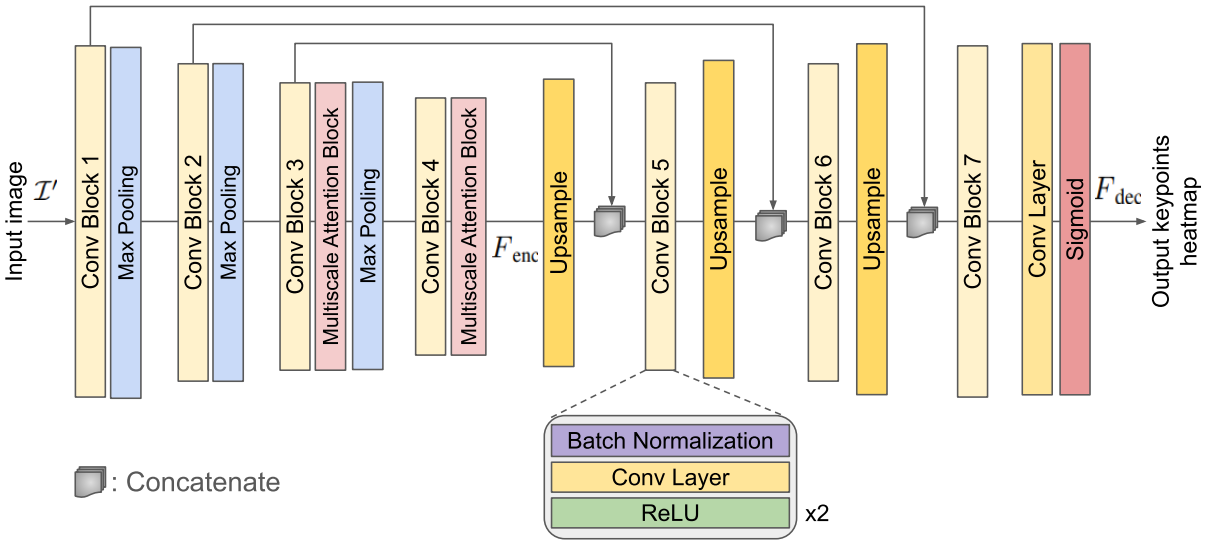

Fig. 5.2: The block diagram of the proposed hand keypoints detection model.

5.2.2 Hand Keypoints Detection

Hand keypoint detection is the localization of K hand joints in an image. This work considers $K = 21$ hand keypoints located at the wrist and three joints with a fingertip for each finger $((3 + 1) \times 5)$.

The input image $\mathcal{I} \in \mathbb{R}^{h \times w \times 1}$ is the hand-segmented image obtained from the ROI extraction stage. This segmented image may be of different spatial resolution, so it is resized to $\mathcal{I}' \in \mathbb{R}^{h \times h \times 1}$ and fed to the proposed hand keypoint detection (HKD) model, shown in Fig. 5.2. The HKD encoder contains four convolution blocks followed by a max pooling layer to capture the most significant details of the feature maps. After the third and the fourth convolution block, a multiscale attention block (MSAB) trains the network to focus on the hand's keypoint locations. Each convolution block performs the operation

$$f = (\mathcal{R} \circ conv \circ BN \circ \mathcal{R} \circ conv \circ BN)(X) \tag{5.1}$$

where X and f are the input and output feature maps, respectively. BN represents the batch normalization layer, $conv$ represents convolution layer, and $\mathcal{R}$ represents ReLU activation. The operation $(A \circ B)(X)$ represents $A(B(X))$. The resulting encoder output is reduced by a factor of

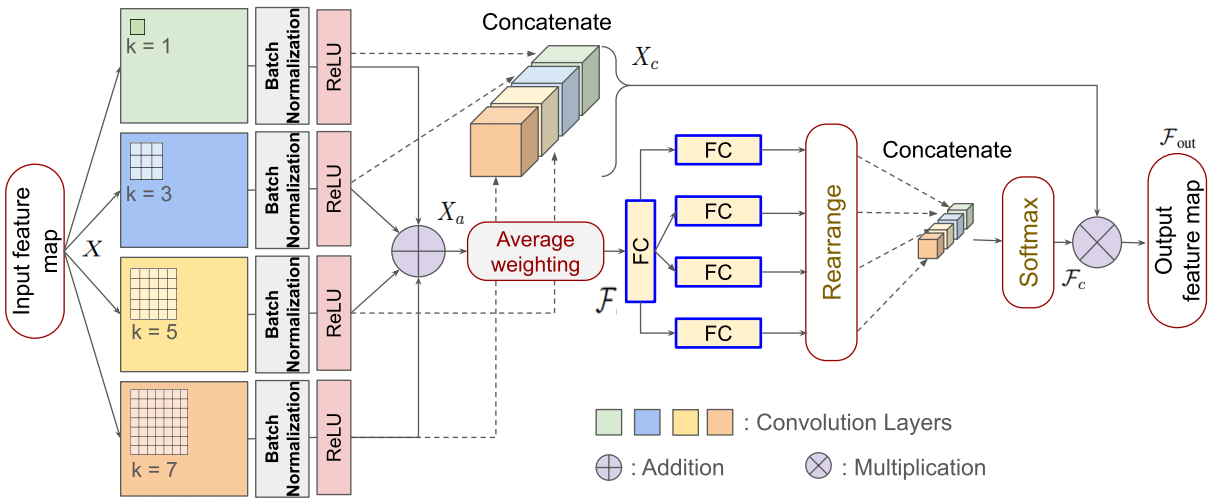

Fig. 5.3: The block diagram of the multi-scale attention block.

$\frac{1}{8}$ using three max pooling layers. Thus, the encoder's output is obtained by

$$F_{\text{enc}} = (MSAB \circ CB4 \circ M \circ MSAB \circ CB3 \circ M \circ CB2 \circ M \circ CB1)(\mathcal{I}') \tag{5.2}$$

where M represents the max pooling layer and CBi represents the i^{th} convolution block.

F_{enc} is given as input to the decoder, which is upsampled and concatenated with the encoder's corresponding dimension feature maps to obtain a feature map embedded with the finer details from the encoder. There are three upsampling layers and as many convolution blocks. The decoder's output comprises a convolution layer and a sigmoid activation layer to obtain 21 heatmaps of the localized keypoints. The decoder's output is given as

$$F_{\text{dec}} = [sigmoid \circ conv \circ CB7 \circ (U|CB1) \circ CB6 \circ (U|CB2) \circ CB5 \circ (U|CB3)](F_{\text{enc}}) \tag{5.3}$$

where $(U|CBi)$ represents upsampling followed by a concatenation of the feature map from the i^{th} convolution block.

5.2.3 Multiscale Attention Block

In the block diagram of the multiscale attention block, shown in Fig. 5.3, the input feature map $X \in \mathbb{R}^{h' \times w' \times c}$ is processed through four parallel convolution layers with kernel sizes of $1 \times 1, 3 \times 3, 5 \times 5$, and 7×7. This arrangement captures information at multiple scales because the larger receptive fields capture global dependencies better [126], and the smaller receptive

fields capture local information. These feature maps are then passed through batch normalization and ReLU activation layers. Following this, two paths are constructed that produce two feature maps, X_a and X_c, such that $X_a \in \mathbb{R}^{h' \times w' \times c'}$ and $X_c \in \mathbb{R}^{4 \times h' \times w' \times c'}$. In one of the paths, the four parallel outputs are added to accumulate four different features and regulate the incoming information. This results in X_a, which is passed through an average weighting scheme to encode soft and overall global information, and is given by

$$\mathcal{F} = \frac{1}{h'w'} \sum_{u=1}^{h'} \sum_{v=1}^{w'} X_a(u, v). \tag{5.4}$$

The output feature map $\mathcal{F}$ is passed through a fully connected (FC) layer to reduce the dimension and make it more compact. The compact feature map is now fed to four parallel FC layers, as shown in Fig. 5.3, to generate four attention weights. The four FC layers capture different long-range characteristics in four instances because the weights of each layer are updated independently, capturing different variations of the incoming data. Each of these four output features or weights is rearranged to obtain a feature of dimension $1 \times 1 \times c'$. Then, they are concatenated to obtain a feature map ($\in \mathbb{R}^{4 \times 1 \times 1 \times c'}$) that is passed through a softmax activation layer to normalize the attention weights. The rearrangement ensures that the resulting feature map has the same dimensions as that of X_c, which is obtained from the other path concatenating the four parallel outputs. The normalized attention weights attend to features relevant to keypoint detection. Therefore, multiplication with X_c further elevates the features it should focus on and suppresses the insignificant ones, enhancing keypoint detection accuracy. The final output of MSAB is given by

$$\mathcal{F}_{\text{out}} = X_c \otimes \mathcal{F}_c \tag{5.5}$$

where $\mathcal{F}_c$ is the output of the softmax layer and $\otimes$ represents the multiplication of $\mathcal{F}_c$ with every location of X_c. This final feature map emphasizes keypoint-relevant information. Further analysis of the module is shown in section experiments and results.

5.2.4 Loss Function

The loss function evaluates the accuracy of a model's predictions relative to actual values, guiding the model towards better performance by reducing error in each iteration. The HKD employs a modified IoU loss function adapted for heatmaps given by

$$\mathcal{L}_{IoU} = 1 - \frac{1}{K}\sum_{k=1}^{K}\Bigg[\Bigg(\sum_{i=1}^{h}\sum_{j=1}^{w} H^k_{pred}(i,j) \times H^k_{true}(i,j)\Bigg)\Big/\Bigg(\sum_{i=1}^{h}\sum_{j=1}^{w}(H^k_{pred}(i,j))^2 + \sum_{i=1}^{h}\sum_{j=1}^{w}(H^k_{true}(i,j))^2 - \sum_{i=1}^{h}\sum_{j=1}^{w} H^k_{pred}(i,j) \times H^k_{true}(i,j)\Bigg)\Bigg]; K = 21 \tag{5.6}$$

where $H^k_{true}(.)$ and $H^k_{pred}(.)$ represent the ground truth and the predicted heatmaps for each keypoint, respectively. Using heatmaps instead of keypoint coordinates for loss calculation offers clear advantages. It visually represents errors, showing how the model progresses and learns over each epoch [119]. The heatmaps's multimodality also helps rectify multiple peaks for a single keypoint, observed in the initial training phase, earlier than coordinate regression. Additionally, IoU metrics are unsuitable for discrete coordinates, further justifying heatmaps in this context.

The concept of IoU is borrowed from semantic segmentation. The only difference is that the values for HKD are continuous ($\in [0, 1]$), while those for segmentation are discrete ($\in \{0, 1\}$). To align predicted keypoints with the actual coordinates, for direct comparison, a 2D Gaussian function is employed, as detailed in Eq.(5.7). This function peaks at the keypoint locations, producing 21 distinct heatmaps that correspond to the 21 individual keypoints of the hand:

$$\mathcal{G}(u,v) = \frac{1}{2\pi\sigma_u\sigma_v} exp\Bigg(-\Bigg(\frac{(u-\mu_u)^2}{2\sigma_u^2} + \frac{(v-\mu_v)^2}{2\sigma_v^2}\Bigg)\Bigg) \tag{5.7}$$

where $\mu_u, \mu_v, \sigma_u,$ and σ_v represent mean and standard deviation along u and v directions, respectively. The more the overlap between the predicted and actual Gaussian peaks, the less the loss, and vice versa. Therefore, the goal is to reduce the distance between the predicted and actual Gaussian peaks, which the model learns iteratively.

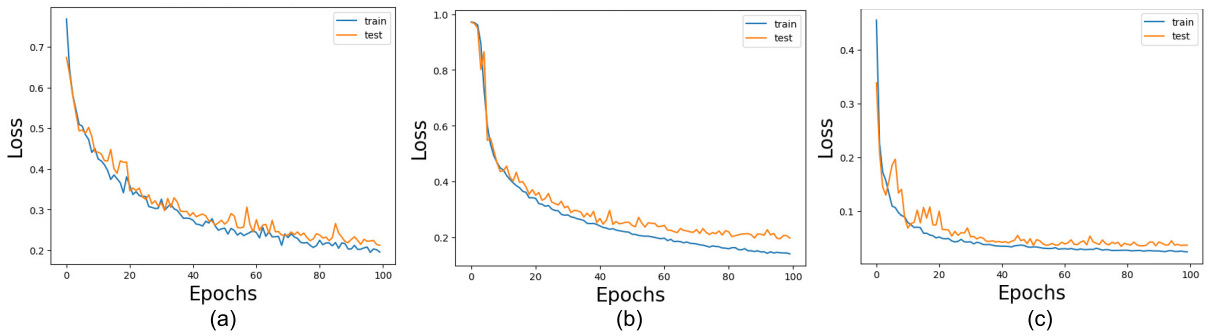

Fig. 5.4: The training and testing curves for three datasets, namely, (a) HIU, (b) FreiHAND, and (c) STB.

5.3 Experiments and Results

This section describes the datasets and performance measures used to evaluate the proposed HKD method, the experiments, and the qualitative and quantitative results. Our multiscale attention-based method is implemented in Python and trained on an Nvidia Tesla P100 GPU. The experimentation details are listed in Table 5.1. The training and testing phase is illustrated through loss curves. The progression of the loss curves, depicted in Fig. 5.4, confirms the effectiveness of the training process, with no indication of overfitting.

Table 5.1: Experimentation details.

Datasets	HIU, FreiHAND, STB.
Optimizer	Stochastic Gradient Descent.
Learning Rate	0.1, which is reduced by a factor of 0.5 when there is no improvement for 20 epochs.
Batch Size	10.
Epochs	100.
Image Size	256×256×3
Kernal initializer	Xavier uniform initializer

5.3.1 Performance Measures

Two performance measures were used to assess the detection results of the proposed HKD approach: mean end-point-error (EPE) and percentage of correct keypoints (PCK).

EPE is the Euclidean distance between the actual and predicted keypoints, given as

$$EPE = \frac{1}{K}\sqrt{\sum_{i=1}^{K}(\mathcal{K}_i - \hat{\mathcal{K}}_i)^2} \; ; K = 21, \tag{5.8}$$

Table 5.2: Performance of the proposed model for different kernels used in the MSAB.

Kernels	EPE (%)	PCK
1x1	4.99	0.9813
3x3	4.94	0.9820
5x5	4.73	0.9824
7x7	4.68	0.9841
1x1, 3x3	4.25	0.9851
1x1, 5x5	4.33	0.9846
1x1, 7x7	4.30	0.9799
3x3, 5x5	4.28	0.9849
3x3, 7x7	4.33	0.9843
5x5, 7x7	4.42	0.9839
1x1, 3x3, 5x5	3.81	0.9853
1x1, 3x3, 7x7	3.68	0.9861
1x1, 5x5, 7x7	3.85	0.9848
3x3, 5x5, 7x7	3.58	0.9868
1x1, 3x3, 5x5, 7x7	**3.3**	**0.9883**

where $\mathcal{K}_i$ and $\hat{\mathcal{K}}_i$ represent the actual and predicted keypoints, respectively.

PCK counts a keypoint prediction as correct if its normalized Euclidean distance from the actual keypoint falls within a predefined threshold. Mathematically,

$$PCK = \frac{1}{K}\sum_{i=1}^{K} \mathbb{1}\left(\frac{\sqrt{\sum_{i=1}^{K}(\mathcal{K}_i - \hat{\mathcal{K}}_i)^2}}{Z} \leq th\right) \tag{5.9}$$

where Z denotes max\{image width, image height\} and th denotes the threshold.

5.3.2 Ablation study

An ablation study was conducted to better understand the architectural formation of the proposed model. For ablation studies, HIU dataset was considered unless otherwise specified.

Different combinations of the kernels used in the MSAB and their performances are recorded in Table 5.2. The inclusion of larger kernels, such as 5×5 and 7×7 incorporate better encoding of long-range correlation between keypoints [126]. It is evident from our experiments as well that the inclusion of 5×5 and 7×7 kernels with 1×1 and 3×3 yielded better results. The four types of kernel encode different features with different receptive fields, which are crucial to keypoint detection. It collectively encodes the relationship of keypoints with respect to each

Table 5.3: Performance of the proposed model with the MSAB placed at different positions. BEC stands for best encoder configuration.

	Position	EPE(%)	PCK	# Parameters
Encoder	After CB1, CB2, CB3, CB4	3.61	0.9880	9.38 M
	After CB2, CB3, CB4	3.83	0.9873	9.29 M
	After CB3, CB4	**3.3**	0.9883	8.94 M
	After CB4	3.8	0.98	**7.54 M**
Decoder	BEC + After CB5	3.72	0.9876	10.34 M
	BEC + After CB5, CB6	3.54	**0.9886**	10.69 M
	BEC + After CB5, CB6, CB7	7.14	0.9273	10.78 M

Table 5.4: Performance of the proposed model for different batch sizes.

Batch size	**EPE(%)**	**PCK**
5	5.71	0.9645
10	**3.3**	**0.9883**
15	3.98	0.9821
20	3.85	0.9850

other as well as the morphology of the hand. The performance with four types of kernel is the best compared to three, two, and one type of kernel. Moreover, we experimented with the placement of the MSAB in the overall HKD architecture. Table 5.3 shows the performance of the proposed model for different placement positions of MSAB. When the MSAB is placed after CB3 and CB4 in the encoder, the best result is obtained. It obtained the best EPE with second second-best PCK value and parameter count. This best encoder configuration (BEC) is considered while experimenting with the MSAB positions in the decoder. The inclusion of the MSAB did not provide better results; however, it significantly contributed to the parameter count of the model. Therefore, we incorporated MSAB after CB3 and CB4 in the encoder-decoder architecture to construct the proposed model.

Further, we tested the performance of the proposed model for different numbers of batch

Table 5.5: Performance of the proposed model when trained for different epochs.

Epochs	EPE(%)	PCK
40	6.98	0.9544
60	6.03	0.9748
80	4.03	0.9856
100	**3.3**	**0.9883**
120	3.62	0.9873
140	3.69	0.9864

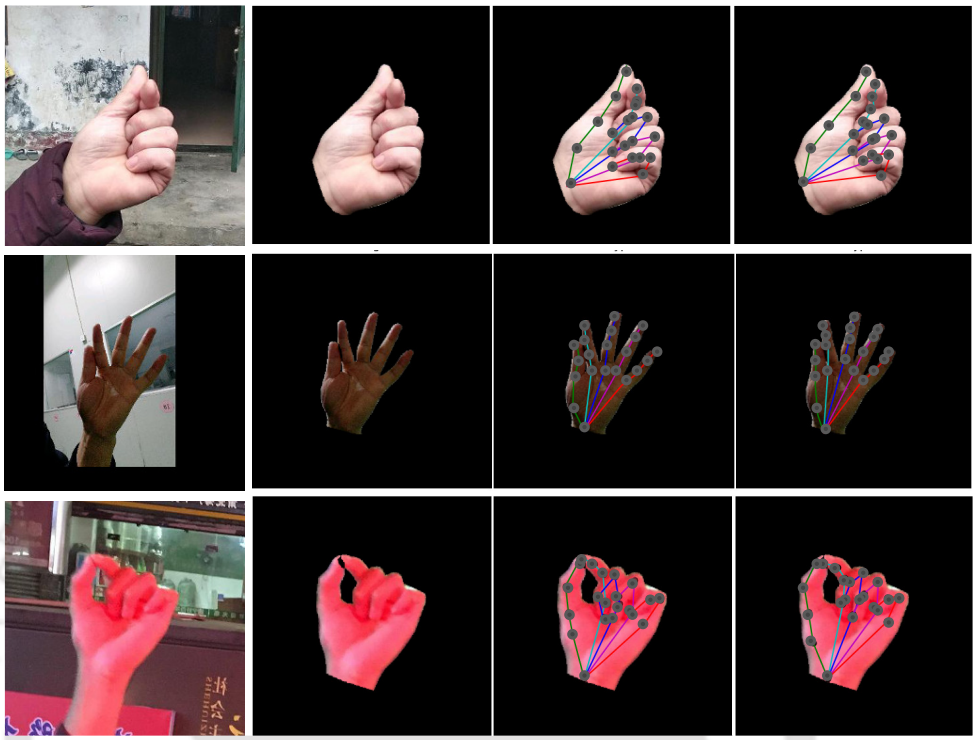

Fig. 5.5: The HKD results for HIU dataset. The first column shows the original color images, the second column shows the ROI extracted image, the third column shows the ground truth, and the fourth column shows the predicted keypoints.

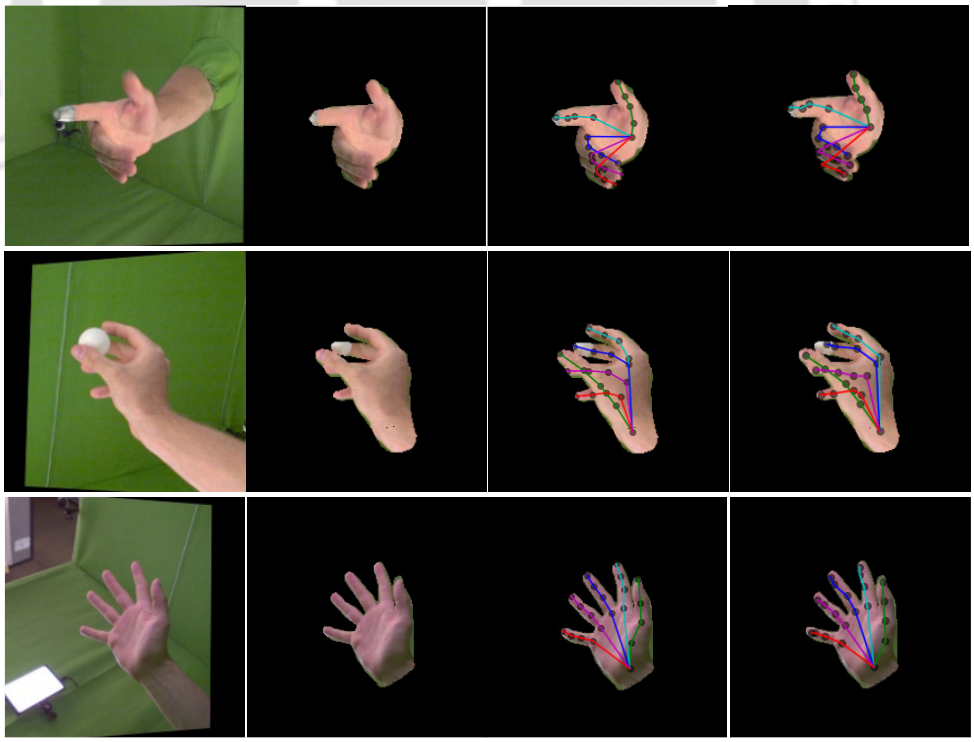

Fig. 5.6: The HKD results for FreiHAND dataset. The first column shows the original color images, the second column shows the ROI extracted image, the third column shows the ground truth, and the fourth column shows the predicted keypoints.

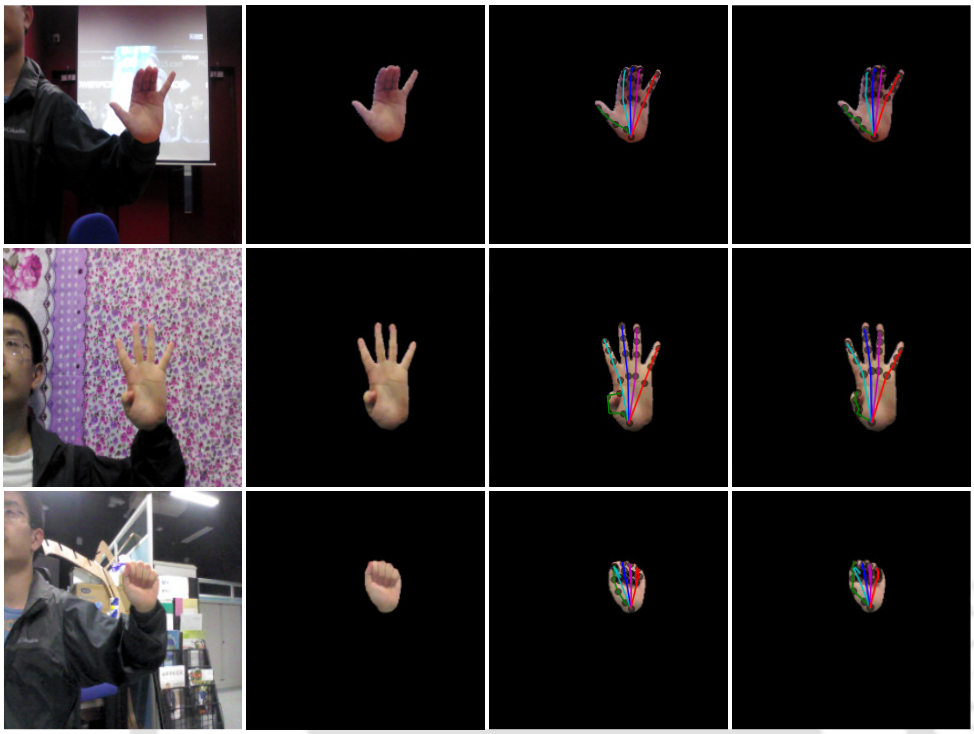

Fig. 5.7: The HKD results for STB dataset. The first column shows the original color images, the second column shows the ROI extracted image, the third column shows the ground truth, and the fourth column shows the predicted keypoints.

sizes and epochs, as shown in Tables 5.4 and 5.5, respectively. We observed a batch size of 10 with 100 epochs gave the best outcome.

5.3.3 Evaluation

The qualitative results are shown in Figs. 5.5, 5.6, and 5.7. The proposed method performs phenomenally despite variations in illumination, background clutter, and self-occlusion. Moreover, the method can detect the hand keypoints even when the person is holding an object. Nevertheless, we experimented only with segmented images containing the hand region with the background removed. We observed that keypoint detection is more accurate on the segmented images, as shown in Fig. 5.8, validating the importance of the ROI extraction stage. Arrows highlight the places where the detection was not accurate in the original images.

The proposed method was quantitatively evaluated by comparing its performance against several baseline models as shown in Table 5.6. The proposed method performs better than the baselines and has a mean EPE of 3.3% for HIU, 3.1% for FreiHAND, and 1.5% for STB datasets. The proposed encoder-decoder architecture embedded with the MSAB was evalu-

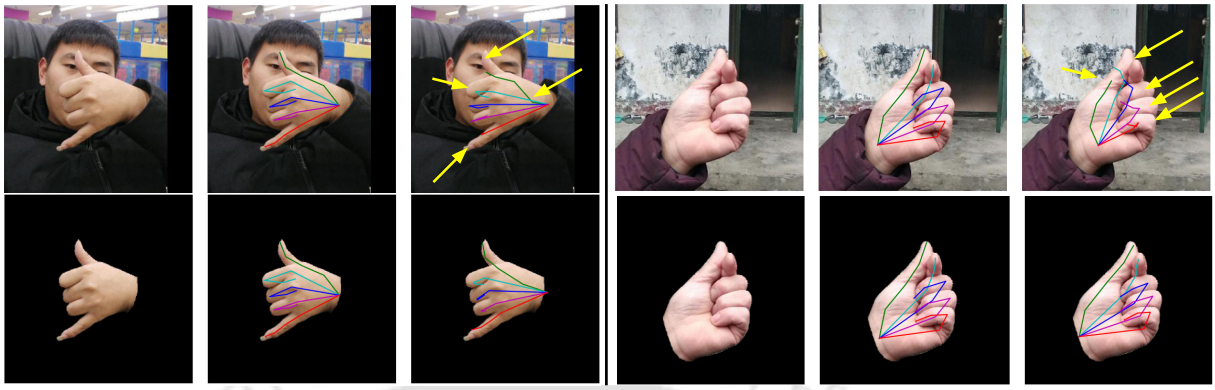

Fig. 5.8: Comparison of keypoint detection in original image and segmented image. The first row shows the original image and the second row shows the segmented image. The first and fourth columns show the input image, the second and fifth columns show the ground truth and the third and sixth columns show the predicted output. The arrows in the image highlight the locations where the keypoint detection has not been accurate.

Table 5.6: Quantitative comparison of the different baseline models with the proposed method in terms of EPE (%). The lower the EPE, the better the performance.

	Network	**# Parameters (Millions)**	**HIU EPE (%)**	**FrieHAND EPE (%)**	**STB EPE (%)**
Others	ShallowNet	1.95	5.3	3.6	2.57
	UNet	7.76	6.5	4.9	3.8
	HRNet	1.35	8.1	7.8	8.1
	SqueezeNet	4.48	11.6	13.5	5.94
	CPM	26.88	13.4	7.4	7.9
Proposed	Enc.-Dec.+ MS. atten.	8.94	**3.3**	**3.1**	**1.5**
	Enc.-Dec. + MS. atten. + supervision	8.97	4.5	5.9	2.6

Table 5.7: Performance of different attention modules with the proposed architecture.

Attention Modules	**EPE(%)**	**PCK**	**MAC(G)**	**# Parameters(M)**
SE Attention [120]	3.97	0.9867	11.18	1.99
CBAM Attention [78]	5.13	0.9812	11.18	2.03
Axial Attention [121]	7.38	0.9475	256.07	151.45
Shuffle Attention [122]	3.88	0.9860	11.18	1.95
Triplet Attention [93]	4.08	0.9845	11.18	1.95
Coordinate Attention [123]	4.13	0.9847	11.18	2.01
CoT Attention [124]	3.76	0.9871	12.36	2.67
SGE [125]	4.01	0.9838	11.18	1.95
Proposed	**3.3**	**0.9883**	22.46	8.94

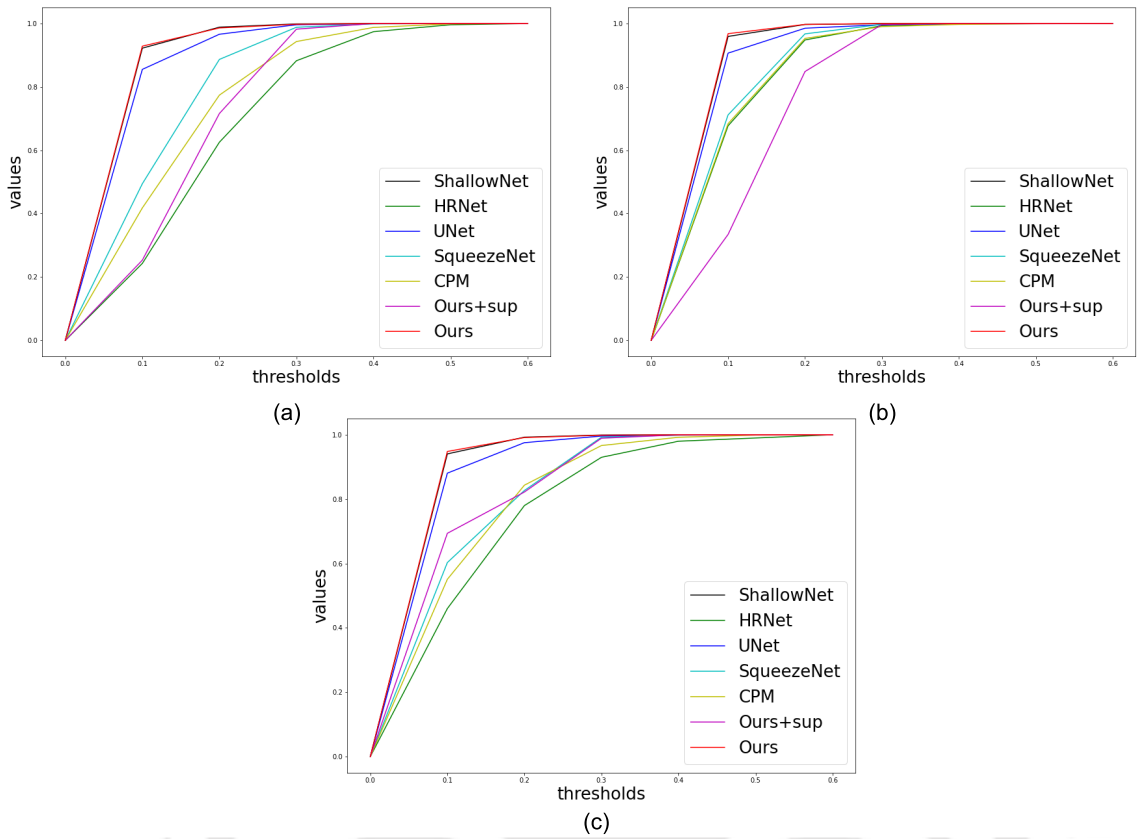

Fig. 5.9: PCK curves for different datasets.

ated with and without supervision. Supervision adds the output heatmap from one stage to an intermediate feature map to refine the next heatmap [126]. Though an established concept, supervision worsened performance. The inclusion of supervision will further increase the number of training parameters resulting in slower inference. In our case, supervision makes the model more complex, thereby increasing the model variance. With the increase in the number of stages, the performance degrades, suggesting the model's inability to generalize with increased complexity. Therefore, our model performed well without supervision. The ShallowNet used in the comparison is a smaller version of U-Net. Fig. 5.9 shows the PCK plot for different threshold values (*th*). Apart from the ShallowNet, the proposed method (without supervision) has a better PCK curve than the baseline models for the three datasets. The ShallowNet follows a similar trend for PCK but has more EPE values than the proposed method.

The effect of the MSAB on the keypoints detection was also tested, as follows:

➢ **Without MSAB**: EPE = 3.39 pixels, PCK@0.2 = 0.9858.

➢ **With MSAB**: EPE = 2.11 pixels, PCK@0.2 = 0.9883.

MSAB reduced the EPE value significantly, though the change of PCK at $th = 0.2$ was minimal. However, attention blocks were only added to the encoder because their addition to the decoder resulted in only negligible improvements. Moreover, we compared the performance of the MSAB with a few other attention modules, as shown in Table 5.7. The MSAB was replaced with these modules, and the performance was recorded. It is observed that MSAB is better at localizing the keypoints with the lowest EPE and highest PCK values.

Table 5.8: Comparison of the proposed method with the state-of-the-art methods for (a) FreiHAND, (b) STB, and (c) HIU datasets. The values are EPE (in pixels).

(a)

Methods	EPE
Zimmermann et al. [127]	9.14
An et al. [75]	9.81
Chen et al. [73]	19.32
Moon et al. [129]	25.9
Santavas et al. [77]	4
Kourbane [68]	2.83
UNet [29]	3.13
CPM [126]	4.73
Ohkawa et al. [128]	3.39
Artacho et al. [130]	4.28
Xu et al. [131]	3.25
Proposed	**2.11**

(b)

Methods	EPE
Zimmermann et al. [127]	5
An et al. [75]	13.17
Chen et al. [73]	13.26
Moon et al. [129]	13.02
Santavas et al. [77]	2.2
Kourbane [68]	1.49
UNet [29]	2.43
CPM [126]	5.05
Ohkawa et al. [128]	4.44
Artacho et al. [130]	5.83
Seo et al. [132]	7.49
Proposed	**1.14**

(c)

Methods	EPE
UNet [29]	4.16
CPM [126]	8.57
Ohkawa et al. [128]	4.29
Artacho et al. [130]	6.97
Proposed	**2.3**

We further compared the performance of the proposed method with the state-of-the-art methods, as shown in Table 5.8. The EPE values of the proposed method for the three datasets are the lowest among all the other methods, which shows the effectiveness of the proposed method under challenging conditions.

5.4 Application: Hand Gesture-based Interfaces

Hand gestures serve as a means for individuals to convey their thoughts and feelings, making interactions based on them inclusive and accessible to a broader segment of society. However, decoding these gestures may be necessary at times, especially when their intentions are not readily apparent. This decoding process is facilitated through an interface. Additionally, interfaces operated by hand offer a touchless approach and afford flexibility in their operation.

Hence, this study emphasizes two interfaces designed for specific groups of patients. The rationale for selecting these patient groups stems from the observation that while numerous hand gesture-based interfaces cater to the general population for human-computer interaction, fewer interfaces are tailored specifically to meet the needs of patients.

Between the two interfaces, one interface develops a hand gesture-operated system as an AI application to relieve discomfort and restore function in hand and arm movements caused by injuries and nerve and muscle complications. The second interface explores the possibility of developing an assistive system for smooth interaction between patients and medical staff via simple hand gestures.

The intuition for the first interface is that discomfort resulting from injuries or pathology can make communicating or performing daily activities hard. Physical therapy is an effective method to restore function to the hand. A system that trains patients to perform simple hand exercises should help to improve their condition. Thus, the objective is to develop a hand gesture-operated system that assists patients in performing rehabilitation exercises from the comfort of their home or any other environment where they feel at ease, eliminating the need to visit a therapy or medical facility.

Also, whenever a patient requires assistance, the medical staff should attend to their requirements swiftly and effectively. The communication between the patient and the medical staff must be optimal to deliver the best service. The patient may need help to use the restroom, eat, call someone, or switch on an electrical appliance. If the medical staff receives specific requests regarding a patient's needs, they may act promptly, but most healthcare facilities do not have such an efficient communication system. Primarily the facilities rely on bells and voice commands for communication. These modalities are ineffective when the patient needs to convey a specific message or is unable to speak. This is the intuition behind developing the second interface. The objective of this interface is to develop a patient assistance system that is a hand gesture-based interaction for communicating their needs to the medical staff. We intend to use hand gestures that can be gestured by simply changing finger arrangements and do not involve much hand movement. These simple hand gestures do not induce much strain on the hand, as

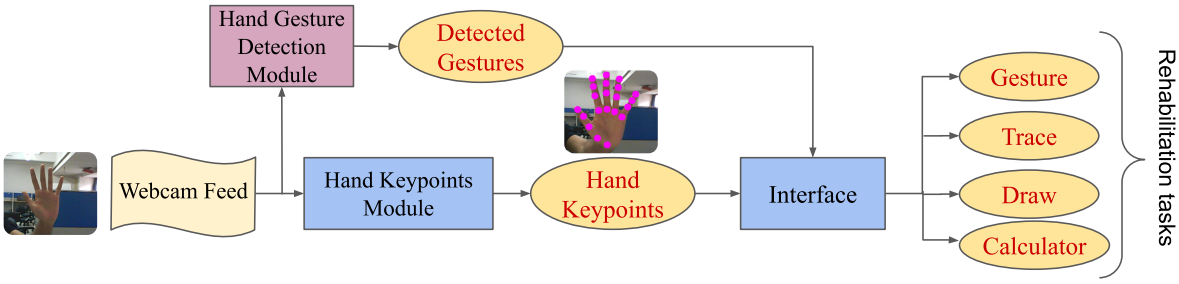

Fig. 5.10: The block diagram depicts the hand gesture interface for helping patients with hand and arm movement impairment.

the patients can gesture while resting their hands on the bed.

5.4.1 Interface 1: Hand Rehabilitation Interface

Target patients: Patients undergoing recovery from various hand-related conditions such as muscle, bone, or nerve injuries, strokes, stiffness, or experiencing tremors.

This section explains the integration of the hand gesture detection module and hand keypoints detection module to operate the hand gesture interface. The system's framework is shown in Fig. 5.10.

The interface was developed to help patients with hand and arm movement impairments resulting from tremor, stiffness, or muscle or nerve injury. The interface supports simple tasks like gesturing, tracing, drawing, and operating a calculator, which use the thenar, hypothenar, and midpalmer muscles. These muscles are associated with the hand's abduction, adduction, extension, flexion, and opposition. The tasks also activate and train the forearm muscle groups such as flexor and extensor muscles, and nerves such as brachial plexus, radial, ulnar, and median nerve, resulting in reduced discomfort and increased stability. The different muscles and nerves activated by the tasks performed on the interface are shown in Fig. 5.11. The interface supports four tasks, as shown in Fig. 5.12. The user selects a task by clicking the button. The tasks are listed below.

(i) Gesture: The objective is to ask the user to perform hand gestures where the muscles and nerves are activated. The hand gesture detection module recognizes the gesture and displays it on the interface with its gesturing score indicating how accurately the user

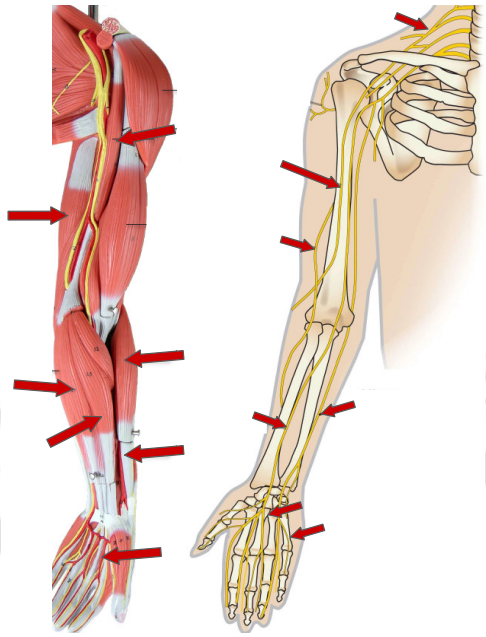

Fig. 5.11: The different muscles (left sub-image) and nerves (right sub-image) activated by the interface tasks.

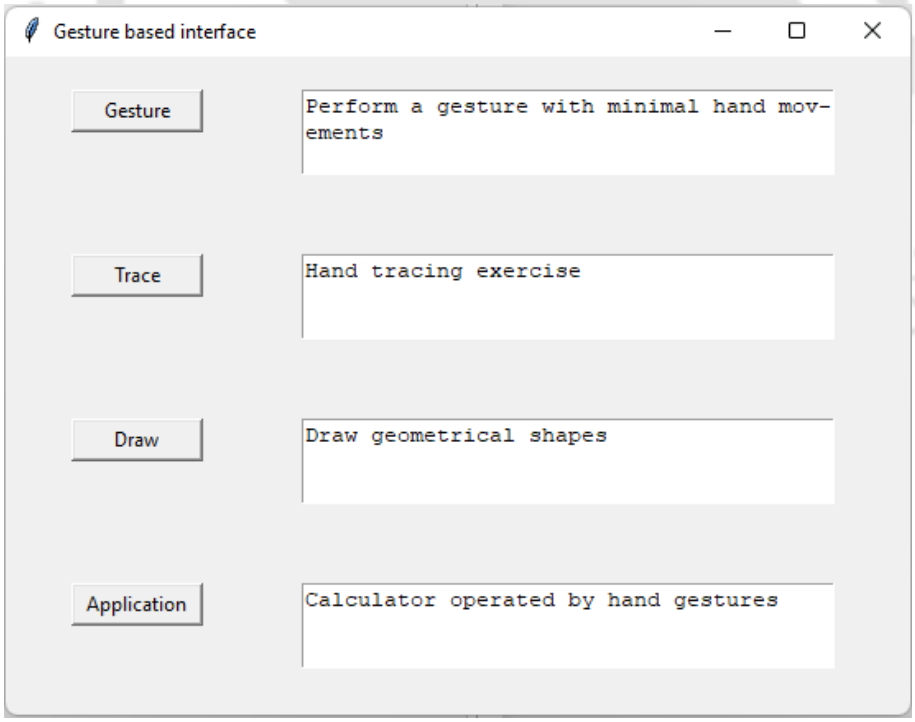

Fig. 5.12: The first window of the hand gesture-based interface comprising the components to perform different tasks for users with discomfort in hand and arm movement.

gestured. The bounding box information determines whether the hand is stable in one position. If the difference of the bounding box position in the successive frame exceeds a certain number of pixels (in this case, 25, which was empirically determined), the trem-

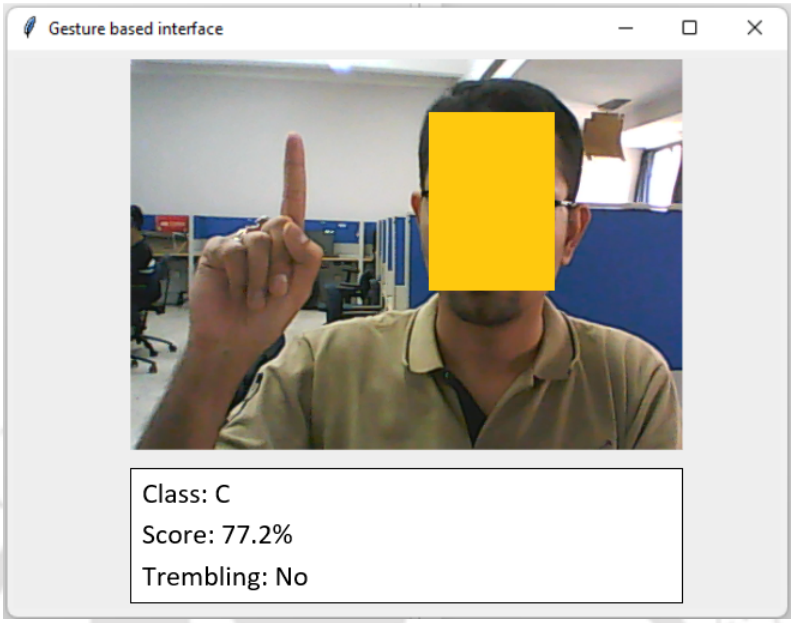

Fig. 5.13: The gesturing interface.

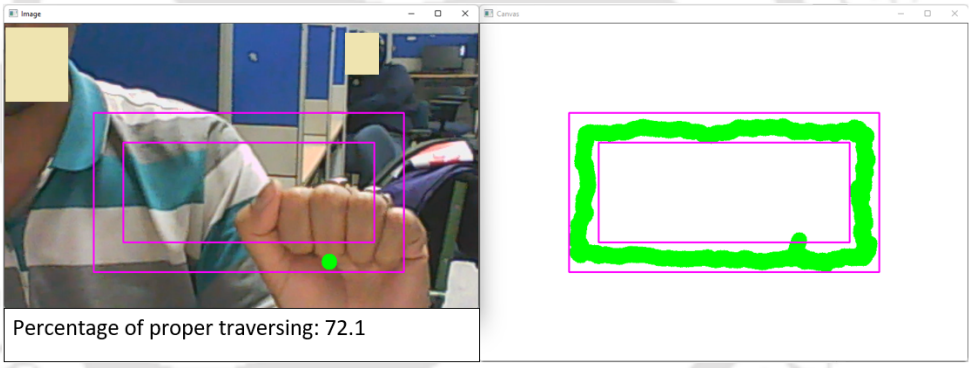

Fig. 5.14: The tracing interface.

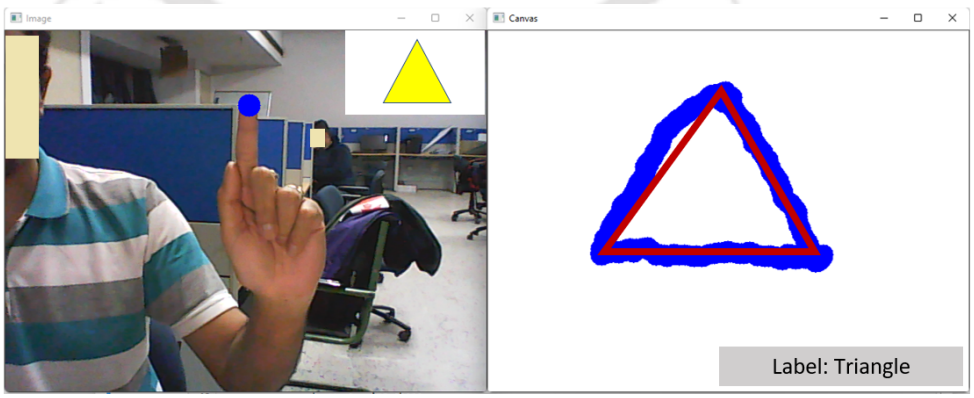

Fig. 5.15: The drawing interface.

bling status is set to "Yes"; otherwise it is set to "No." Fig. 5.13 shows the gesture interface.

(ii) Trace: This task aims to train the user to make an arm movement traversing within a

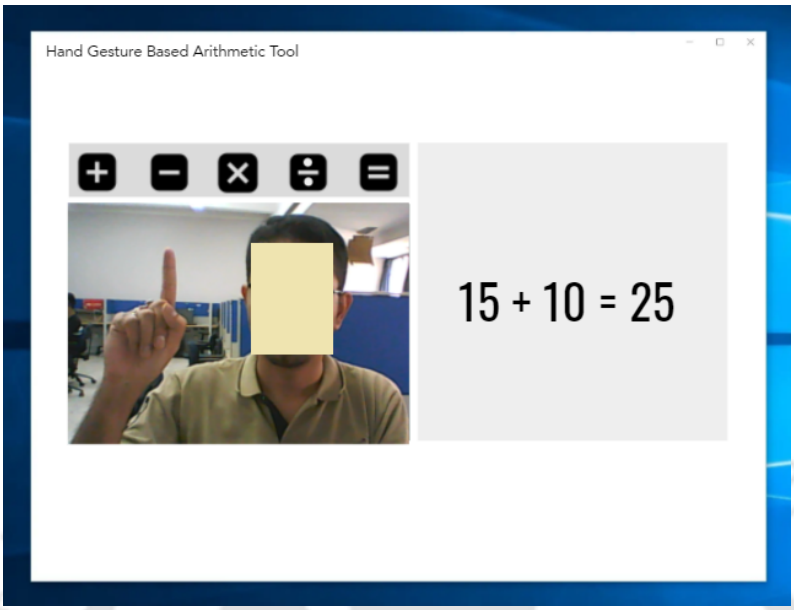

Fig. 5.16: The hand gesture operated calculator interface.

specified boundary. As shown in Fig. 5.14, the user is asked to traverse within two rectangles. The centroid of the hand's bounding box is used to monitor the movement. Common geometrical shapes are used to train the user to control the arm movement and, thus, exercise the arm muscles and the brachial plexus. A performance score is provided to track improvement in the traversing activity.

(iii) Draw: The previous step's progress is tested in this interface by removing the boundary feedback. Here the user is asked to draw a geometrical shape, and the shape is labelled and displayed. The actual shape of the drawn pattern is superimposed on the pattern. The index finger is used to draw the shape. Alternatively, the hand bounding box centroid may be used. Fig. 5.15 shows the drawing interface.

(iv) Application: Once the user has progressed in the above activities, the user can operate the calculator by hand. Here the user needs to keep the hand stable, or else the gesture is discarded. Also, the user's hand needs to traverse in a specific direction to select the required arithmetic operation. Here the input feed captured by a webcam is given to a hand keypoints module to generate the hand keypoints. The keypoints are used to detect the tip of the index finger, enabling the user to select the desired arithmetic operation. The interface control block accepts two types of input: the fingertip to se-

Table 5.9: Symbols used in Algorithm 1

I^k	Image frame
H	Height of the image
W	Width of the image
n_1, n_2	number 1, number 2
y	Output of the arithmetic operation
Δ	A threshold to consider the hand keypoint for selection of arithmetic operation
T	A set of intervals containing the spatial positions of the arithmetic operators
op	Arithmetic operators
x_1, y_1	Keypoint coordinates of the index finger

lect the arithmetic operation and the detected hand gestures (described in the previous section) to enter the numbers. Numbers and operators are displayed on the interface, as shown in Fig. 5.16. The figure shows that a user selected an operator ('=') with the tip of the index finger. Algorithm 1 shows the pseudocode for the interface. The values th_1 and th_2 are derived empirically, and the other elements of T are obtained as $[th_{i-1}, th_i] = [th_{i-3} + m - \frac{th_{i-2} - th_{i-3}}{2}, th_{i-2} + m + \frac{th_{i-2} - th_{i-3}}{2}]$. Here, m is the distance between the symbols, which is empirically determined. This system operates smoothly despite background clutter and variations in illumination. The interface's output is a string displayed on the monitor, such as "15 + 10 = 25." Table 5.9 lists symbols used in Algorithm 1.

5.4.1.1 Hand Keypoints Module

The hand keypoints module is equally essential as the hand gesture detection module as it trains the patients to achieve normal arm motion with hand control and movement-based exercises. The hand keypoints module used in this work is the proposed HKD method without the ROI extraction stage. The hand gestures in this case would be performed in a hospital or lab environment where the background is not cluttered. The HKD stage without any pre-processing is sufficient enough to give us good results.

5.4.2 Interface 2: Patient Assistance System

Target patients: All patients who are conscious and not in ventilation (have minimal wires or tubes attached) and can at least move their fingers. For example, patients admitted due to

Algorithm 1 Touchless gesture interface application

Input: $I^k, k \in [1, \infty) \& I^k \in \mathbb{R}^{H \times W}$ ▷ I^k: image frames with operators overlaid
Output: $y = n_1 \circ n_2, \circ \in \{+, -, \times, \div\}$ ▷ y: output of arithmetic operation

$\Delta \leftarrow$ height of the overlaid image + offset
$T \leftarrow \{[th_{(2i-1)}, th_{(2i)}] \,|\, i = 1 : 5\}$
$op \leftarrow [+, -, \times, \div, =]$
while $I^k exists$ **do**
    $keypoints \leftarrow HKM\_Model(I^k)$
    $x_1, y_1 = keypoints[index\_finger]$
    if $index\_finger\ is\ up \,\&\, y_1 < \Delta$ **then**
        for $j = 1 \rightarrow len(T)$ **do**
            if $x_1 \in T[j] \,\&\, j \neq len(T)$ **then** $\circ \leftarrow op[j]$ & **break**
            else if $x_1 \in T[j] \,\&\, j = len(T)$ **then** $\circ \leftarrow n_1 \circ n_2$ & **break**
            else continue
            end if
        end for
    end if
    if $y_1 > \Delta$ **then**
        $scores, box \leftarrow Hand\_Detection\_Module(I^k)$
        $num \leftarrow argmax(scores)$. ▷ num forms n_1 and n_2 progressively with each iteration
    end if
    if $n_1 \,\&\, n_2\ decided$ **then goto** step 10 and display
    end if
end while

some illness, underwent a laparoscopic surgery, wound treatment, orthopedic treatment, etc.

The patient assistance system (PAS) is a novel application of the proposed method, which lets patients convey specific messages to the medical staff when they need assistance. The PAS's setup is shown in Fig. 5.17, and the interface is shown in Fig. 5.18. The PAS's setup consists of a tablet with an in-built camera to capture image frames containing the hand region and display the interface to the patient, a stand to hold the tablet, a communication link (maybe wired or wireless), and a computer to receive the message on the medical staff end. Fig. 5.17 shows a typical setup. The tablet is placed at a suitable distance so that the interface is visible to the patient, and the patient does not hit it while getting out of bed. The communication link is established over the internet, which may be a wired connection or WiFi. This system has two aspects: one is accurate hand gesture recognition, and the other is optimal transmission management between the end users. This work mainly focuses on hand gesture detection to

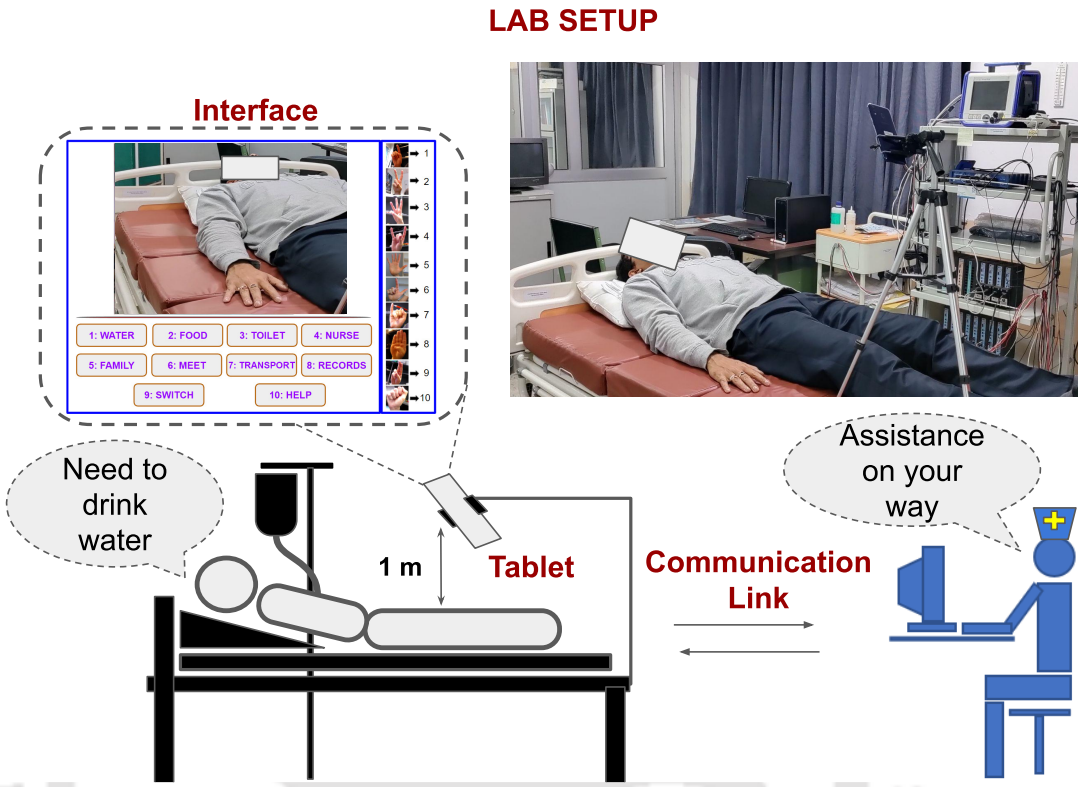

Fig. 5.17: A schematic representation of the patient assistance system setup.

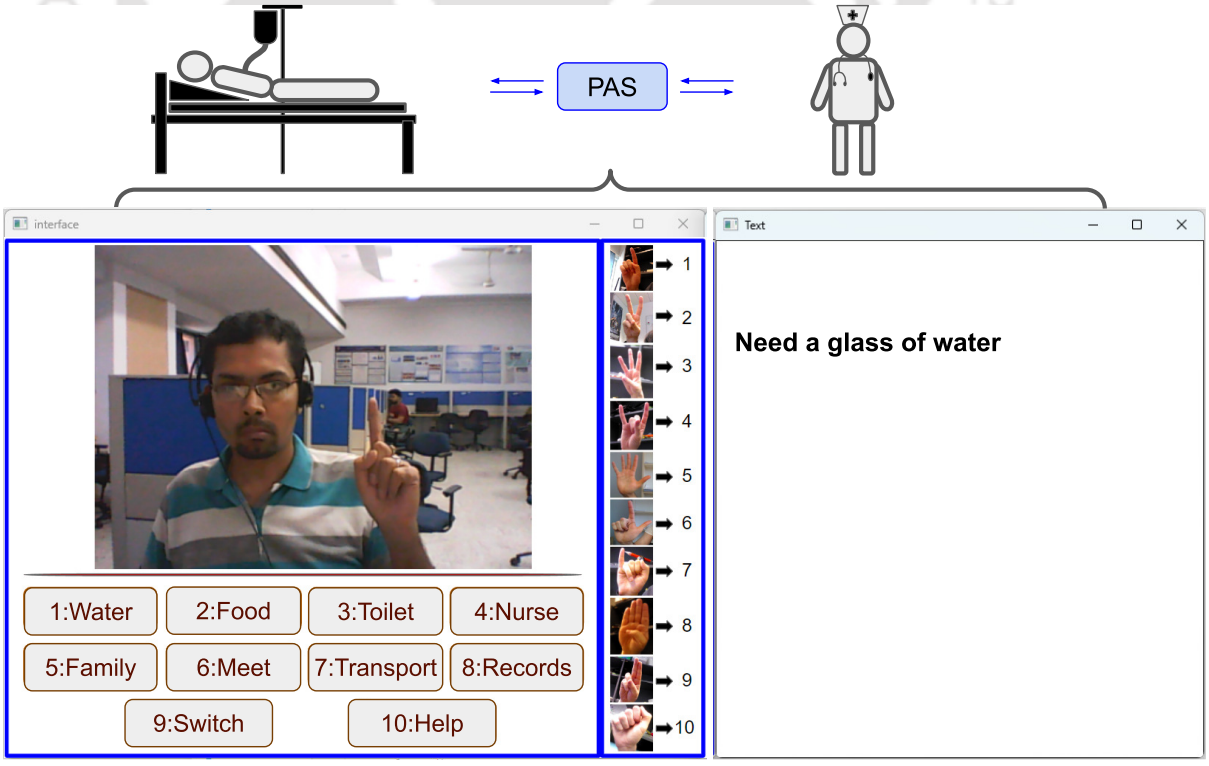

Fig. 5.18: The patient assistance system.

	Need a glass of water.
	Need something to eat.
	Help to go to toilet/bathroom.
	Call a nurse.
	Call a family member.
	Meet a doctor or therapist.
	Avail transportation service, i.e., ambulance or taxi.
	Request medical records.
	Operate a switch of an appliance, such as a fan, AC, light, or TV.
	Help with getting out of bed, walking, dressing, or bathing.

Fig. 5.19: The messages and their corresponding gestures.

develop a robust and efficient system that decodes the messages sent via hand gestures. Now to train the detection part of the system, we use publicly available hand gesture recognition image datasets captured in complex environments (discussed in Chapter 1). We used datasets that involve minimal hand movement and can be gestured by simply changing the finger positioning. Using publicly available datasets help us avoid restricting the system to a particular hospital environment but enables its training and testing to generalize to other environments.

This system is convenient for patients who are bedridden or otherwise lack mobility. It is operated by hand gestures that involve simple finger arrangements. The 10 available gestures are visible on the interface to help patients with gesturing. They correspond to the 10 classes in the dataset on which the classification method was trained and the 10 messages, shown in Fig. 5.19.

The patients gesture the specific hand gesture according to their needs, and the PAS recognizes it. The PAS captures the patient's hand gesture through the in-built camera of a tablet mounted on a stand above the bed the patient is lying on. The tablet displays the interface where the patient can get the live feed of gesturing. Once the patient gestures, the PAS's interface runs the proposed hand gesture detection algorithm to predict the hand location and gesture class label. All of this is done in the processing unit of the tablet/computer. Finally, the message as-

Fig. 5.20: (a) Different instances of hand gesture detection on the video frames. (b) The frames highlighting keypoints detected.

signed to the recognized gesture is transmitted via the wired or wireless communication link to the medical staff's computer. Upon receiving the message, it is forwarded to the staff member who can address it. Simultaneously, the medical staff acknowledges it by text, which the system converts to speech and plays in the patient's room. This assures patients that their requests are being attended to. Moreover, there is a bell attached to the system for the message "SOS" when the patient needs immediate attention. This is a more effective way to communicate than pressing a call button because the patient can convey different messages.

5.4.3 Evaluation

5.4.3.1 Interface 1

This subsection covers the performance of the proposed interface. The operation of the interface has already been explained. Here the algorithm's detection capability in online mode (i.e., using a webcam) is presented. Fig. 5.20 shows different instances where the detection was made for the webcam feeds under varying conditions, such as background clutter, hands of different shapes and sizes, different subjects, and the presence of human faces or specular reflection. It also shows the detected hand keypoints and a sub-image with the tip of the index finger.

System's effectiveness: Two groups of five South Asian male patients, aged 20 to 35,

Table 5.10: Study of the effectiveness of the gesturing interface on five patients. The number of successful gesturing out of 50 attempts is shown for each patient, and over the course of 10 observations, a steady improvement is seen. P represents patient.

Observations (out of 50 attempts of gesturing)	Ouhands					NUS					Mean	Standard deviation
	P 1	P 2	P 3	P 4	P 5	P 1	P 2	P 3	P 4	P 5		
1	1	0	0	2	5	2	2	0	5	2	1.9	1.75
2	1	0	1	8	6	2	4	3	10	10	4.5	3.58
3	5	5	1	10	10	10	10	10	18	22	10.1	5.82
4	6	9	5	18	14	10	15	11	15	25	12.8	5.65
5	8	10	15	22	26	18	22	26	21	32	20	7.05
6	10	19	28	25	25	23	23	30	24	37	24.4	6.66
7	20	31	27	29	26	35	33	35	30	38	30.4	4.98
8	28	38	31	33	31	34	40	39	35	40	34.9	4.01
9	32	41	38	35	40	39	43	45	40	39	39.2	3.51
10	39	45	42	40	40	41	43	48	45	42	42.5	2.65

Table 5.11: Study of the performance of five patients who did not use the gesturing interface. The number of successful gesturing out of 50 attempts is shown for each patient and for 10 observations. P represents patient.

Observations (out of 50 attempts of gesturing)	Ouhands					NUS					Mean	Standard deviation
	P 1	P 2	P 3	P 4	P 5	P 1	P 2	P 3	P 4	P 5		
1	1	1	0	0	0	1	0	0	2	1	0.6	0.66
2	2	1	1	2	2	3	1	0	2	2	1.6	0.8
3	3	3	2	2	3	3	2	1	1	3	2.3	0.78
4	3	5	3	4	5	5	5	2	3	4	3.9	1.04
5	6	9	5	7	8	8	9	7	5	8	7.2	1.4
6	10	10	8	11	11	12	11	10	11	13	10.7	1.26
7	12	13	10	13	13	15	13	14	14	16	13.3	1.55
8	15	16	13	16	15	15	16	17	16	18	15.7	1.26
9	18	19	16	17	19	19	19	21	20	20	18.8	1.4
10	20	22	19	20	21	21	23	21	24	22	21.3	1.41

with hand and arm movement complications, were considered to interact with the hand gesture interface. One group was trained in the rehabilitation tasks, and the other was not trained to study the effect of the gesturing interface on rehabilitation. Prior approval of IIT Guwahati's ethics review board was obtained before data acquisition. The patients were asked to gesture the 10 classes of OUHANDS and NUS datasets five times. Every three days, an observation was made, and 10 of those observations were recorded to keep track of the patients' improvement in gesturing. If they could gesture a particular class with a gesturing score above 60% and acceptable trembling (i.e., the difference between the hand bounding box is less than 25 pixels), the gesture would be considered a success. Thus, 50 attempts were made for each of the 10 observations (5 times gesturing $\times$ 10 classes = 50 attempts). The findings are shown in Table

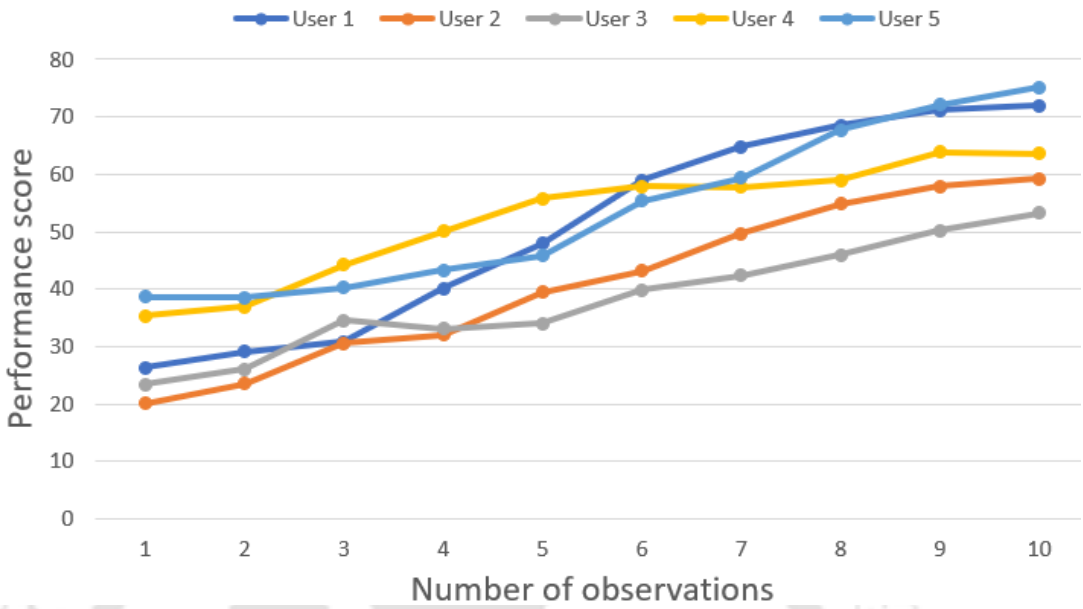

Fig. 5.21: Tracing performance for 10 observations.

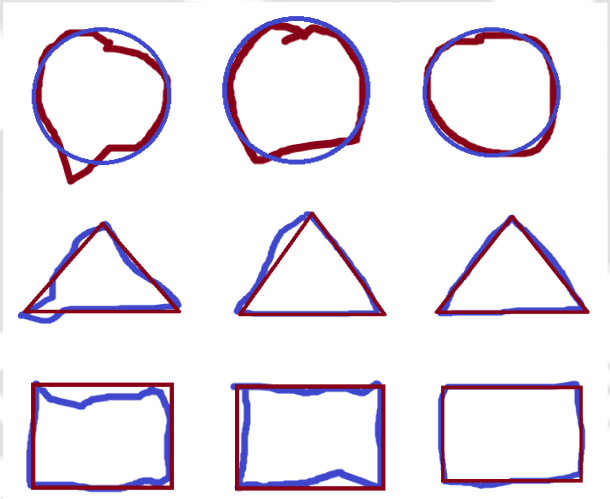

Fig. 5.22: Visual observation of the outcome of the draw interface. Improvement in hand movement is represented in column-wise order (from column 1 to column 3).

5.10 for the group using the gesturing interface and in Table 5.11 for the group not using the interface. The observations in Table 5.10 indicate that the continuous use of the interface helps patients recover normal gesturing. It also helps relieve hand stiffness and weakness by guiding hand or finger movements. However, Table 5.11 suggests that the group that did not use the interface showed slower recovery. The p-value for both groups is 0.026, which shows the rehabilitation tasks' statistical significance.

In addition, a performance score is used to monitor improvement in the user's path traversal

Table 5.12: Number of hand gestures accurately detected in the calculator interface in a 20-second interaction. The patients' performances were recorded as the rate of accurate detection of the gestures in an environment with variable illumination and background clutter. P represents patient.

		Illumination variation					Background clutter				
		P 1 (%)	P 2 (%)	P 3 (%)	P 4 (%)	P 5 (%)	P 1 (%)	P 2 (%)	P 3 (%)	P 4 (%)	P 5 (%)
classes	0	88.23	100	90	100	94.73	100	100	100	95	95
	1	100	100	94.73	100	90	100	100	100	100	100
	2	89.47	94.44	84.2	80	100	89.47	90	100	100	95
	3	94.73	89.47	94.73	100	88.23	94.73	95	100	89.47	94.73
	4	94.73	100	90	95	100	100	100	95	90	89.47
	5	100	89.47	95	88.23	88.23	94.73	89.47	88.23	90	100
	6	100	90	94.73	100	95	100	90	100	94.73	90
	7	94.44	84.2	88.23	80	100	100	95	100	94.73	100
	8	100	88.23	100	94.73	100	100	95	88.23	90	100
	9	100	89.47	100	90	95	100	100	90	100	100

for 10 observations. This is plotted in the graph shown in Fig. 5.21. Here, the patients' hand stability is gradually improved owing to their use of the interface. The patients tend to tremble less and control their hand movement better through use. This is shown in Fig. 5.22 too, where the patients draw a particular shape, and the deviation from the actual shape's boundary is observed visually.

The performance score is given by

$$\text{performance score} = \frac{\sum_{i=1}^{num} \Psi}{n} \tag{5.10}$$

$$\text{where } \Psi = \begin{cases} 1, (x_c, y_c) \in (R_1 - R_2) \\ 0, \text{otherwise} \end{cases} \tag{5.11}$$

(x_c, y_c) denotes the centroid of the hand bounding box, and *num* denotes the total number of centroids of the bounding box that constitute the path traversed. R_1 and R_2 represent the area of the large and small rectangle, as shown in Fig. 5.14.

Furthermore, the five patients were asked to interact with the calculator interface using the 10 hand gestures for 20 s each. They operated this interface once they were confident with the previous tasks. Their performance was recorded as the rate of accurately detecting the gestures in Table 5.12 in an environment with variable illumination (dim light, bright light, and light from the side) and background clutter. The classes used in the table represent the classes in the datasets in their respective order. However, to represent the decimal number system, they are

Table 5.13: Study of the performance of five users for five attempts on PAS interface.

Attempts	User 1	User 2	User 3	User 4	User 5
1	83.3	86.5	87.2	88.4	88.2
2	87.6	89.3	87.0	86.3	91.8
3	89.9	92.4	88.1	87.6	90.8
4	90.0	90.2	93,5	89.1	90.4
5	92.1	90.6	92.1	89.0	89.1

replaced by digits. The table indicates the interface recognizes the gestures accurately in most cases in the two challenging conditions. Thus, it works seamlessly in the online mode. It makes reliable inferences while avoiding the need for many sensors, sensors requiring physical contact with the user, or sensors restricting the user's movement. This interface's purpose is to verify the hand's normal functioning using an application that includes all of the hand rehabilitation tasks. It offers the usage of a hardware-free calculator. More importantly, using this interface gives a patient the gratification of regaining normal hand functioning.

5.4.3.2 Interface 2

Five users were asked to operate the interface in the lab environment and test its reliability. Due consent from the users and the institute's ethics committee was taken for the experiments. The users were asked to keep their arms still while performing the gestures displayed on the interface five times. The mean accuracies of the gestures for each attempt were recorded in Table 5.13. The accuracy ranged from 89.0 % to 92.1 %, which indicates the reliability and effectiveness of the PAS system. However, for failed cases, the users were asked to perform the gesture to call a nurse or a family member. Thus, the interface would present a way to establish communication even if a gesture was misrecognized. This is facilitated by a confirmation message after each gesture, which displays "yes" and "no" options. The user must select the desired option to proceed. In this case, "yes" is selected by an open hand gesture (class 5) and "no" by a fist (class 10). We also took a few real-scene images of the users to verify our proposed approach. As shown in Fig 5.23, the PAS associated the correct messages with the gestures posed by the users. It shows that the system is operable in a real scenario and performs reliably.

We further prepared the following questionnaire to obtain feedback about the usefulness of

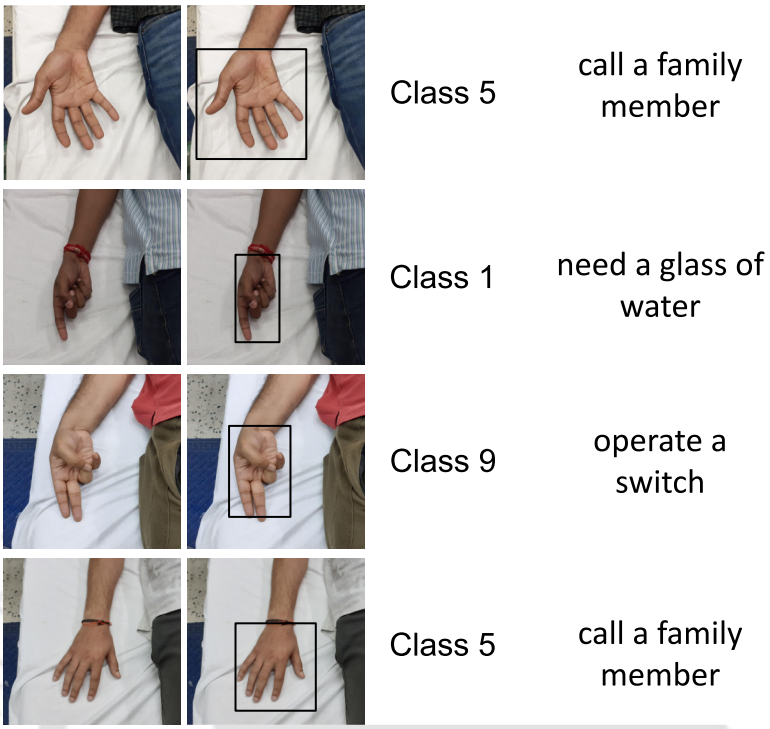

Fig. 5.23: The PAS output for users performing the gestures in the lab environment. The first column shows the image frames captured by the setup shown in Fig. 5.17, the second column shows the hand localization, the third column shows the classification results, and the fourth column shows the messages associated with the classes.

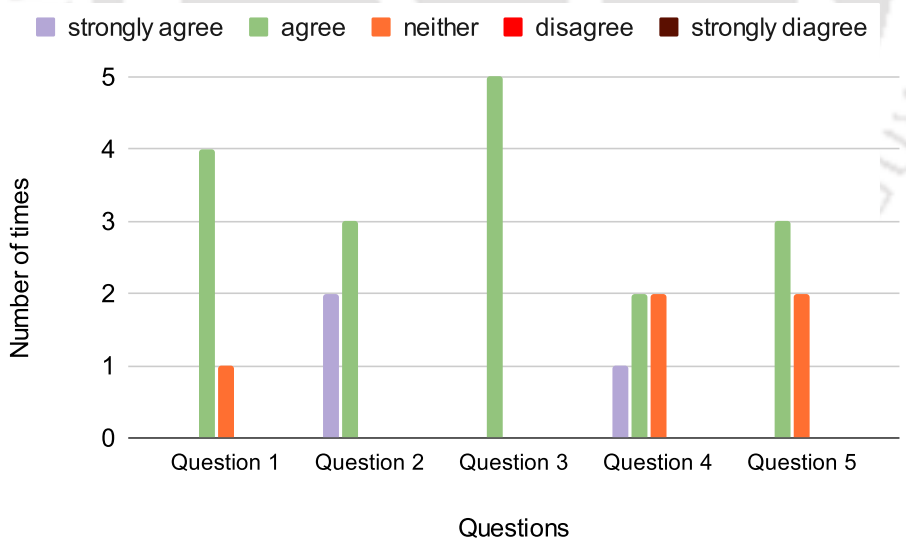

Fig. 5.24: Feedback response of the patients about the PAS.

the PAS:

1) The proposed system is operable without help after the initial demonstration of the system.

2) The proposed system is useful and gives a satisfactory feel.

3) The proposed system will be used by the patients again.

4) The proposed system is useful for medical staff and reduced miscommunication (due to a medical staff attending to a request he cannot address).

5) The proposed system needs more functions and improvement.

The results are summarized in Fig. 5.24. For the most part, the patients found the system operable and useful, and they would consider using it again. They also pointed out that they would appreciate further advancement related to its speed of operation and increased functionality.

5.5 Contribution

- This work proposed a simple encoder-decoder network with a novel multiscale attention block that teaches the network to focus on features relevant to hand keypoint detection.
- It used a modified IoU loss function that optimizes the model to precisely localize the keypoints in spite of background clutter, self-occlusion, and variations in illumination.
- We propose a hand gesture operated system to help patients with hand or arm injuries or complications improve their movement through therapeutic tasks that address different muscle groups and motor nerves.
- We propose a patient assistance system that translates the patient's hand gestures into messages conveyed to medical staff. This helps avoid confusion and results immediate action.

5.6 Summary

Hand keypoint detection plays a crucial role in many gesture-based applications. However, developing a generalized detection method has remained a long-standing problem. Several factors impede accurate detection: the fingers' distance from the camera and their nearness, self-occlusion, variations in illumination, and background clutter. To overcome these barriers, we propose a two-stage architecture. The first stage generates precise hand regions, eliminating adjoining skin regions and background clutter. The second stage incorporates a novel multi-scale attention block to detect keypoint coordinates precisely. Qualitative and quantitative evaluations found that the proposed architecture outperforms state-of-the-art models for the three

benchmark datasets. This advancement lays the groundwork for future 3D hand pose estimation developments and their applications. However, the proposed method has two-stages, rendering it computationally intensive for real-time applications. Though this method is capable of online applications (< 25 frames/s), efforts will be made to develop a single-stage keypoint detector for real-time use (= 25 frames/s). Moreover, the method does not give consistently accurate results in dim lighting conditions. Hence, the proposed method will be refined to predict keypoints in dim light or dark surroundings.

Using the second stage, i.e., the hand keypoint detection without the pre-processing stage and the hand gesture detection, two hand gesture-based interfaces were developed to cater to two distinct groups of patients. The first interface is tailored for patients undergoing hand rehabilitation due to injury or complications related to muscles or nerves, while the second interface is intended for hospital patients to establish direct communication with medical staff. The primary objective behind these interface developments is to facilitate comfortable rehabilitation sessions and establish clear, direct communication channels between patients and medical personnel. Accurate hand gesture detection is crucial for the successful operation of these interfaces, thus the detection transformer-based hand gesture detection method discussed in the previous chapter was employed for hand detection. Evaluation of the interfaces yielded promising results: Interface 1 aided patients in stabilizing and restoring normal hand and arm function, while Interface 2 accurately decoded hand gestures, enabling patients to effectively communicate their needs to medical staff. Unlike the conventional call button commonly found in hospitals, Interface 2 allows patients to send ten different messages.

6

Conclusion and Future Works

Contents

6.1 Introduction

This chapter concludes the thesis by highlighting the work done in each chapter. The thesis aims to develop a robust hand gesture recognition method that can be utilized in hand gesture-based interfaces establishing touchless interaction with computers. The areas delved into by this thesis include hand segmentation, hand gesture recognition, hand gesture detection, and hand keypoint detection. Different approaches were proposed related to these areas, which eventually assist in developing two interfaces targeted at patients undergoing hand rehabilitation and helping patients residing in hospitals to establish direct communication with the medical staff. The chapter-wise conclusions are as follows:

Chapter 1 serves as an introduction to the thesis, providing an overview of key concepts such as HCI, hand gesture recognition, and the datasets utilized throughout the study. In addition to outlining these foundational elements, the chapter delves into a discussion of prior research in the field, highlighting existing literature and approaches. Furthermore, it explores the underlying motivation driving the thesis and outlines the specific problem definitions that the study aims to address. Through this comprehensive overview, Chapter 1 sets the stage for the subsequent chapters, providing context and framing the research objectives within the broader landscape of hand gesture recognition.

Chapter 2 proposes two novel encoder–decoder hand segmentation networks where the first network segments the hand by encoding spatial and channel correlations using two attention blocks and the second network benefited from the amalgamation of CNN and transformer structures. The second approach further enhanced the efficiency of the transformer model by reducing its quadratic complexity to linear. These approach requires much less computation than benchmark self-attention mechanisms. Moreover, a novel loss function optimizes the segmentation models to resolve class imbalance, ensure boundary smoothness, and retain the hand's shape. The quantitative and qualitative results show the models' exceptional performance for images with different hand poses and orientations, the presence of a human face, background clutter, specularity, and variations in illumination. The first model attained an F1-score of 97.3%

for the Ouhands dataset with an inference time of 15.03 ms and the second model attained 97.73% for the same dataset with an inference time of 13.4 ms, better than baseline models. Furthermore, the second model generalizes hand segmentation to multiple hands and unseen environments and performs slightly better than the first. Therefore, it is chosen as the first stage prior to hand gesture recognition.

Chapter 3 proposes a robust hand gesture recognition method integrated with a segmentation model that provides a tool to control human-computer interfaces based on gestures. This recognition model capitalizes on the benefits of convolution and transformer networks and a proposed attention module named the adaptive kernel channel attention layer. The combination of the convolution layer and the transformer helps encode local and global dependencies, and the attention module focuses the network's learning on the hand region. The proposed recognition method accepts the segmentation model's output as input and performs feature engineering on the segmented masks as it progresses through the stack of layers. The proposed method performs well on benchmark hand gesture datasets with better evaluation results than state-of-the-art methods. It achieves 93.8% and 98.0% recognition accuracy on Ouhands and NUS datasets, respectively. An accurate hand gesture recognition network ushers the development of an interface that helps human-computer communication.

Chapter 4 introduces a novel hand gesture detection model aimed at localizing and recognizing hand gestures concurrently, thereby eliminating the reliance of the recognition stage on the segmentation stage and minimizing the time lost during transitions between stages. The proposed detection framework adopts an end-to-end approach and proposes three distinct methods for hand gesture detection. The first method employs an end-to-end network architecture based on RetinaNet, assisted by a convolutional block attention module to enhance its performance. The second method utilizes a detection transformer to encode global information and reduce spurious detections through bipartite matching. Meanwhile, the third method involves regressing a bounding box around the hand by treating it as a point, employing the CenterNet architecture with parallel spatial and channel attention modules. Unlike the first method, which relies on prior bounding box information and may result in an imbalance between positive and

negative boxes, the second and third methods do not exhibit this drawback, making them more favorable alternatives. Evaluation of the three models on the Ouhands dataset yielded F1-scores of 86.7%, 89.9%, and 84.4%, respectively.

Chapter 5 proposes a hand keypoint detection approach that extracts the hand region in the input image and, subsequently, detects the hand keypoints. A multiscale attention block is proposed to increase detection accuracy, which helps the network focus on features relevant to determining precise keypoint locations. The proposed approach performs remarkably with background clutter, other skin regions, self-occlusion, and illumination variations. It achieved a PCK value of 0.9883, 0.9916, and 0.9970 at th = 0.2 and a mean EPE value as low as 2.3 pixels, 1.14 pixels, and 2.11 pixels for HIU, STB, and FreiHAND datasets, respectively. These values indicate performance exceeding state-of-the-art methods, a conclusion supported by the qualitative results. Therefore, the proposed method is effective and resistant to most of the challenges of hand keypoint detection. The chapter further discusses two hand gesture-based interfaces developed to help hand rehabilitation patients perform exercises and hospital patients communicate directly with medical staff as an application of the proposed hand gesture and keypoints detection methods. The primary goals of these interfaces are to facilitate comfortable rehabilitation sessions and clear communication between patients and medical personnel, respectively. Accurate hand gesture detection is essential for their successful operation, and the detection transformer-based method from chapter 4 is employed for this purpose. Evaluation of the interfaces showed promising outcomes: Interface 1 helped patients stabilize and restore hand and arm function, while Interface 2 effectively decoded hand gestures for communication with medical staff. The system's effectiveness was evaluated by assessing its statistical significance and soliciting feedback from patients through a questionnaire to assess its usefulness.

6.2 Thesis Contribution

The major contributions of the thesis are as follows:

➢ A novel encoder-decoder architecture for hand segmentation is proposed that uses two novel and efficient attention blocks, one for spatial features to emphasize the hand's loca-

tion and one for channel features to emphasize the pixel class.

- ➢ A novel segmentation model is proposed that combines the benefits of a CNN and a transformer to obtain a segmentation map, precisely separating the hand from the background. This segmentation model preserves the inductive bias and long-range dependencies.
- ➢ The transformer's self-attention mechanism has been substituted with a more efficient attention mechanism to mitigate computational complexity and memory requirements.
- ➢ A compound segmentation loss function is proposed to maintain the hand's geometrical shape and smooth boundary, especially around the fingers. It also corrects for class imbalance.
- ➢ A new classification network is proposed, incorporating an adaptive kernel channel attention layer (AKCAL) to enhance the accuracy of hand gesture recognition.
- ➢ A detection transformer-based method is proposed, which is the first to integrate a detection transformer with hand gesture detection to the best of our knowledge, estimating the location of the hand in a frame and its classification score.
- ➢ Three hand detection methods are proposed– one works using prior anchor box information, and the other two are anchorless detectors. Among the anchorless approaches, one utilizes the benefits of CNN and transformer, while the other regresses a bounding box encompassing the hand, considering it as a point.
- ➢ A simple encoder-decoder network is introduced featuring a novel multiscale attention block, which instructs the network to prioritize relevant features for hand keypoint detection.
- ➢ A modified intersection-over-union (IoU) loss function is used that optimizes the model to precisely localize the keypoints in spite of background clutter, self-occlusion, and variations in illumination.
- ➢ A hand gesture-operated system is designed to aid patients with hand or arm injuries or complications in enhancing their movement capabilities through therapeutic tasks targeting various muscle groups and motor nerves.
- ➢ A patient assistance system is introduced that interprets the patient's hand gestures into

messages relayed to medical staff, thereby facilitating clear communication and prompt response, ultimately avoiding confusion.

6.3 Future Directions

- ✍ The thesis work predominantly with static hand gestures. The temporal aspect of gestures are not explored. The gestures varying with time can establish better one-to-one communication, such as sign language. Thus, a method needs to be devised to incorporate this temporal information and make the HCI further intuitive.
- ✍ In hand gesture recognition and keypoint detection, a two-stage approach is followed to achieve better accuracy. This makes the second stage dependent on the first for the best output. Therefore, it can achieve inference time suitable for online applications (<25 frames/s), not real-time (=25 frames/s). Therefore, a single-stage architecture needs to be developed to achieve real-time performance.
- ✍ In a few cases, the segmentation model needs improvement in segmenting hands that overlap with faces or in extremely low illumination.
- ✍ Hand keypoint estimation can be further improved by introducing morphological constraints, ensuring presence of keypoints within the hand boundary.
- ✍ Explore meta-learning for hand gesture classification to accommodate for limited data and adapt to any changes in the classification problem quickly. This ensures better generalizability and classification accuracy.
- ✍ Include a communication link in the rehabilitation interface to establish real-time remote communication with a specialist/therapist for consultation. Also, introduce more interactive exercises with the inclusion of mixed reality.

Bibliography

[1] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "Vision-based hand pose estimation: A review," *Computer Vision and Image Understanding*, vol. 108, no. 1, pp. 52–73, 2007, special Issue on Vision for Human-Computer Interaction. [Online]. Available: https://doi.org/10.1016/j.cviu.2006.10.012

[2] S. Mitra and T. Acharya, "Gesture recognition: A survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 3, pp. 311–324, 2007. [Online]. Available: https://doi.org/10.1109/TSMCC.2007.893280

[3] S. Thuseethan and S. Kuhanesan, "Effective use of human computer interaction in digital academic supportive devices," in *arXiv preprint arXiv:1501.00529*, 2015.

[4] M. Karam, "A framework for research and design of gesture-based human-computer interactions," Ph.D. dissertation, University of Southampton, October 2006. [Online]. Available: https://eprints.soton.ac.uk/263149/

[5] A. S. Dick, S. Goldin-Meadow, A. Solodkin, and S. L. Small, "Gesture in the developing brain," *Developmental science*, vol. 15, no. 2, p. 165–180, 2012. [Online]. Available: https://doi.org/10.1111/j.1467-7687.2011.01100.x

[6] Y. Nam and K. Wohn, "Recognition of space-time hand-gestures using hidden markov model," in *1996 ACM Symposium on Virtual Reality Software and Technology*, 1996, p. 51–58. [Online]. Available: https://doi.org/10.1145/3304181.3304193

[7] B. K. Chakraborty, D. Sarma, M. Bhuyan, and K. F. MacDorman, "Review of constraints on vision-based gesture recognition for human–computer interaction," *IET Computer Vision*, vol. 12, no. 1, pp. 3–15, 2018. [Online]. Available: https://doi.org/10.1049/iet-cvi.2017.0052

[8] P. K. Pisharady, P. Vadakkepat, and A. P. Loh, "Attention based detection and recognition of hand postures against complex backgrounds," *International Journal of Computer Vision*, vol. 101, no. 3, pp. 403–419, Feb. 2013. [Online]. Available: https://doi.org/10.1007/s11263-012-0560-5

[9] M. Matilainen, P. Sangi, J. Holappa, and O. Silvén, "Ouhands database for hand detection and pose recognition," in *2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA)*, 2016, pp. 1–5. [Online]. Available: https://doi.org/10.1109/IPTA.2016.7821025

[10] J. Nalepa and M. Kawulok, "Fast and accurate hand shape classification," in *Beyond Databases, Architectures, and Structures*, ser. Communications in Computer and Information Science, S. Kozielski, D. Mrozek, P. Kasprowski, B. Malysiak-Mrozek, and D. Kostrzewa, Eds. Springer, 2014, vol. 424, pp. 364–373. [Online]. Available: https://doi.org/10.1007/978-3-319-06932-6_35

[11] T. Grzejszczak, M. Kawulok, and A. Galuszka, "Hand landmarks detection and localization in color images," *Multimedia Tools and Applications*, vol. 75, no. 23, pp. 16 363–16 387, 2016. [Online]. Available: https://doi.org/10.1007/s11042-015-2934-5

[12] X. Zhang, H. Huang, J. Tan, H. Xu, C. Yang, G. Peng, L. Wang, and J. Liu, "Hand image understanding via deep multi-task learning," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 11 261–11 272. [Online]. Available: https://doi.org/10.1109/ICCV48922.2021.01109

[13] S. Bambach, S. Lee, D. J. Crandall, and C. Yu, "Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1949–1957. [Online]. Available: https://doi.org/10.1109/ICCV.2015.226

[14] C. Zimmermann, D. Ceylan, J. Yang, B. Russell, M. J. Argus, and T. Brox, "FreiHAND: A Dataset for Markerless Capture of Hand Pose and Shape From Single RGB Images," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 813–822. [Online]. Available: https://doi.org/10.1109/ICCV.2019.00090

[15] J. Zhang, J. Jiao, M. Chen, L. Qu, X. Xu, and Q. Yang, "A hand pose tracking benchmark from stereo matching," in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 982–986, doi: 10.1109/ICIP.2017.8 296 428. [Online]. Available: https://doi.org/10.1109/ICIP.2017.8296428

[16] A. Dadashzadeh, A. Targhi, M. Tahmasbi, and M. Mirmehdi, "HGR-Net: A fusion network for hand gesture segmentation and recognition," *IET Computer Vision*, vol. 13, pp. 700–707, 2019. [Online]. Available: https://doi.org/10.1049/iet-cvi.2018.5796

[17] Z. Ju, X. Ji, J. Li, and H. Liu, "An integrative framework of human hand gesture segmentation for human–robot interaction," *IEEE Systems Journal*, vol. 11, no. 3, pp. 1326–1336, 2017. [Online]. Available: https://doi.org/10.1109/JSYST.2015.2468231

[18] Y. Wang, C. Peng, and Y. Liu, "Mask-pose cascaded CNN for 2D hand pose estimation from single color image," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 11, pp. 3258–3268, 2019. [Online]. Available: https://doi.org/10.1109/TCSVT.2018.2879980

[19] M. Kawulok, J. Kawulok, and J. Nalepa, "Spatial-based skin detection using discriminative skin-presence features," *Pattern Recognition Letters*, vol. 41, pp. 3–13, 2014. [Online]. Available: https://doi.org/10.1016/j.patrec.2013.08.028

[20] R. Khan, A. Hanbury, J. Stöttinger, and A. Bais, "Color based skin classification," *Pattern Recognition Letters*, vol. 33, no. 2, pp. 157–163, 2012. [Online]. Available: https://doi.org/10.1016/j.patrec.2011.09.032

[21] B. Chakraborty and M. Bhuyan, "Image specific discriminative feature extraction for skin segmentation," *Multimedia Tools and Applications*, vol. 79, pp. 18 981–19 004, 2020. [Online]. Available: https://doi.org/10.1007/s11042-020-08762-4

[22] A. U. Khan and A. Borji, "Analysis of hand segmentation in the wild," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4710–4719. [Online]. Available: https://doi.org/10.1109/CVPR.2018.00495

[23] M. Cai, F. Lu, and Y. Sato, "Generalizing hand segmentation in egocentric videos with uncertainty-guided model adaptation," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 14 380–14 389. [Online]. Available: https://doi.org/10.1109/CVPR42600.2020.01440

[24] F. Yang and Y. Wu, "A soft proposal segmentation network (SPS-Net) for hand segmentation on depth videos," *IEEE Access*, vol. 7, pp. 29 655–29 661, 2019. [Online]. Available: https://doi.org/10.1109/ACCESS.2019.2900991

[25] T.-H. Tsai and S.-A. Huang, "Refined U-net: A new semantic technique on hand segmentation," *Neurocomputing*, vol. 495, pp. 1–10, 2022. [Online]. Available: https://doi.org/10.1016/j.neucom.2022.04.079

[26] T. Ohkawa, T. Yagi, A. Hashimoto, Y. Ushiku, and Y. Sato, "Foreground-aware stylization and consensus pseudo-labeling for domain adaptation of first-person hand segmentation," *IEEE Access*, vol. 9, pp. 94 644–94 655, 2021. [Online]. Available: https://doi.org/10.1109/ACCESS.2021.3094052

[27] H. Wu, J. Zhang, K. Huang, K. Liang, and Y. Yizhou, "FastFCN: Rethinking dilated convolution in the backbone for semantic segmentation," in *arXiv preprint arXiv:1903.11816*, 2019.

[28] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3141–3149. [Online]. Available: https://doi.org/10.1109/CVPR.2019.00326

[29] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *2015 Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, vol. 9351. Springer International Publishing, 2015, pp. 234–241. [Online]. Available: https://doi.org/10.1007/978-3-319-24574-4_28

[30] O. Oktay, J. Schlemper, L. L. Folgoc, M. Lee, M. Heinrich, K. Misawa, K. Mori, S. McDonagh, N. Y. Hammerla, B. Kainz, B. Glocker, and D. Rueckert, "Attention U-Net: Learning where to look for the pancreas," in *arXiv preprint arXiv:1804.03999*, 2018.

[31] Q. Jin, Z. Meng, C. Sun, H. Cui, and R. Su, "RA-UNet: A hybrid deep attention-aware network to extract liver and tumor in CT scans," *Frontiers in Bioengineering and Biotechnology*, vol. 8, 2020. [Online]. Available: https://doi.org/10.3389/fbioe.2020.605132

[32] X. Sun, C. Chen, X. Wang, J. Dong, H. Zhou, and S. Chen, "Gaussian Dynamic Convolution for Efficient Single-Image Segmentation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 5, pp. 2937–2948, 2022. [Online]. Available: https://doi.org/10.1109/TCSVT.2021.3096814

[33] B. Baheti, S. Innani, S. Gajre, and S. Talbar, "Eff-UNet: A novel architecture for semantic segmentation in unstructured environment," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020, pp. 1473–1481. [Online]. Available: https://doi.org/10.1109/CVPRW50498.2020.00187

[34] A. Almeida, P. Vicente, and A. Bernardino, "Where is my hand? Deep hand segmentation for visual self-recognition in humanoid robots," *Robotics and Autonomous Systems*, vol. 145, p. 103857, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889021001421

[35] J. Chen, Y. Lu, Q. Yu, X. Luo, E. Adeli, Y. Wang, L. Lu, A. L. Yuille, and Y. Zhou, "TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation," in *arXiv preprint arXiv:2102.04306*, 2021.

[36] P. Harding and T. Ellis, "Recognizing hand gesture using fourier descriptors," in *2004 17th International Conference on Pattern Recognition (ICPR)*, vol. 3, 2004, pp. 286–289. [Online]. Available: https://doi.org/10.1109/ICPR.2004.1334523

[37] N. H. Dardas and N. D. Georganas, "Real-time hand gesture detection and recognition using bag-of-features and support vector machine techniques," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 11, pp. 3592–3607, 2011. [Online]. Available: https://doi.org/10.1109/TIM.2011.2161140

[38] K.-p. Feng and F. Yuan, "Static hand gesture recognition based on hog characters and support vector machines," in *2013 2nd International Symposium on Instrumentation and Measurement, Sensor Network and Automation (IMSNA)*, 2013, pp. 936–938. [Online]. Available: https://doi.org/10.1109/IMSNA.2013.6743432

[39] J. Alon, V. Athitsos, Q. Yuan, and S. Sclaroff, "A unified framework for gesture recognition and spatiotemporal gesture segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 9, pp. 1685–1699, 2009. [Online]. Available: https://doi.org/10.1109/TPAMI.2008.203

[40] H.-K. Lee and J. Kim, "An HMM-based threshold model approach for gesture recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 10, pp. 961–973, 1999. [Online]. Available: https://doi.org/10.1109/34.799904

[41] H. P. Jyoti Dutta, D. Sarma, M. Bhuyan, and R. H. Laskar, "Semantic segmentation based hand gesture recognition using deep neural networks," in *2020 National Conference on Communications (NCC)*, 2020, pp. 1–6. [Online]. Available: https://doi.org/10.1109/NCC48643.2020.9055990

[42] P. Bao, A. I. Maqueda, C. R. del Blanco, and N. García, "Tiny hand gesture recognition without localization via a deep convolutional network," *IEEE Transactions on Consumer Electronics*, vol. 63, no. 3, pp. 251–257, 2017. [Online]. Available: https://doi.org/10.1109/TCE.2017.014971

[43] S. F. Chevtchenko, R. F. Vale, V. Macario, and F. R. Cordeiro, "A convolutional neural network with feature fusion for real-time hand posture recognition," *Applied Soft Computing*, vol. 73, pp. 748–766, 2018. [Online]. Available: https://doi.org/10.1016/j.asoc.2018.09.010

[44] T. H. N. Le, K. G. Quach, C. Zhu, C. N. Duong, K. Luu, and M. Savvides, "Robust hand detection and classification in vehicles and in the wild," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017, pp. 1203–1210. [Online]. Available: https://doi.org/10.1109/CVPRW.2017.159

[45] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, p. 303–338, Jun. 2010. [Online]. Available: https://doi.org/10.1007/s11263-009-0275-4

[46] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," in *2014 European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2014, pp. 740–755. [Online]. Available: https://doi.org/10.1007/978-3-319-10602-1_48

[47] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, p. 580–587. [Online]. Available: https://doi.org/10.1109/CVPR.2014.81

[48] R. Girshick, "Fast R-CNN," in *2015 IEEE International Conference on Computer Vision (ICCV)*. USA: IEEE Computer Society, 2015, p. 1440–1448. [Online]. Available: https://doi.org/10.1109/ICCV.2015.169

[49] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017. [Online]. Available: https://doi.org/10.1109/TPAMI.2016.2577031

[50] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788. [Online]. Available: https://doi.org/10.1109/CVPR.2016.91

[51] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6517–6525. [Online]. Available: https://doi.org/10.1109/CVPR.2017.690

[52] ——, "Yolov3: An Incremental Improvement," 2018, arXiv preprint arXiv:1804.02767.

[53] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," in *2016 European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2016, pp. 21–37. [Online]. Available: https://doi.org/10.1007/978-3-319-46448-0_2

[54] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2999–3007. [Online]. Available: https://doi.org/10.1109/ICCV.2017.324

[55] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *2017 31st International Conference on Neural Information Processing Systems*, 2017, p. 6000–6010. [Online]. Available: https://doi.org/10.5555/3295222.3295349

[56] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-End Object Detection with Transformers," in *2020 European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2020, pp. 213–229. [Online]. Available: https://doi.org/10.1145/3474085.3475285

[57] S. R. Bose and V. S. Kumar, "Efficient inception v2 based deep convolutional neural network for real-time hand action recognition," *IET Image Processing*, vol. 14, pp. 688–696(8), March 2020. [Online]. Available: https://doi.org/10.1049/iet-ipr.2019.0985

[58] A. A. Q. Mohammed, J. Lv, and M. S. Islam, "A deep learning-based end-to-end composite system for hand detection and gesture recognition," *Sensors*, vol. 19, no. 23, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/23/5282

[59] S. Sharma, H. Pallab Jyoti Dutta, M. Bhuyan, and R. Laskar, "Hand gesture localization and classification by deep neural network for online text entry," in *2020 IEEE Applied Signal Processing Conference (ASPCON)*, 2020, pp. 298–302. [Online]. Available: https://doi.org/10.1109/ASPCON49795.2020.9276713

[60] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *2015 3rd International Conference on Learning Representations, (ICLR)*, 2015. [Online]. Available: http://arxiv.org/abs/1409.1556

[61] K. S. Yadav, A. M. K., R. H. Laskar, and N. Ahmad, "Exploration of deep learning models for localizing bare-hand in the practical environment," *Engineering Applications of Artificial Intelligence*, vol. 123, p. 106253, 2023. [Online]. Available: https://doi.org/10.1016/j.engappai.2023.106253

[62] Q. Gao, Y. Chen, Z. Ju, and Y. Liang, "Dynamic Hand Gesture Recognition Based on 3D Hand Pose Estimation for Human–Robot Interaction," *IEEE Sensors Journal*, vol. 22, no. 18, pp. 17 421–17 430, 2022. [Online]. Available: https://doi.org/10.1109/JSEN.2021.3059685

[63] G. Yuan, X. Liu, Q. Yan, S. Qiao, Z. Wang, and L. Yuan, "Hand gesture recognition using deep feature fusion network based on wearable sensors," *IEEE Sensors Journal*, vol. 21, no. 1, pp. 539–547, 2021. [Online]. Available: https://doi.org/10.1109/JSEN.2020.3014276

[64] R. Li, Z. Liu, and J. Tan, "A survey on 3D hand pose estimation: Cameras, methods, and datasets," *Pattern Recognition*, vol. 93, pp. 251–272, 2019. [Online]. Available: https://doi.org/10.1016/j.patcog.2019.04.026

[65] B. Roumaissa and B. Mohamed Chaouki, "Hand pose estimation based on regression method from monocular RGB cameras for handling occlusion," in *Multimedia Tools and Applications*, 2023. [Online]. Available: https://doi.org/10.1007/s11042-023-16384-9

[66] S. E. Ovur, H. Su, W. Qi, E. De Momi, and G. Ferrigno, "Novel Adaptive Sensor Fusion Methodology for Hand Pose Estimation With Multileap Motion," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–8, doi: 10.1109/TIM.2021.3 063 752, 2021. [Online]. Available: https://doi.org/10.1109/TIM.2021.3063752

[67] M. Bhuyan, D. R. Neog, and M. K. Kar, "Hand pose recognition using geometric features," in *2011 National Conference on Communications (NCC)*, 2011, pp. 1–5. [Online]. Available: https://doi.org/10.1109/NCC.2011.5734786

[68] I. Kourbane and Y. Genc, "A Graph-Based Approach for Absolute 3D Hand Pose Estimation Using a Single RGB Image," *Applied Intelligence*, vol. 52, no. 14, p. 16667–16682, nov 2022. [Online]. Available: https://doi.org/10.1007/s10489-022-03390-x

[69] M. Zhang, Z. Zhou, and M. Deng, "Cascaded Hierarchical CNN for 2D Hand Pose Estimation from a Single Color Image," *Multimedia Tools and Applications*, vol. 81, no. 18, p. 25745–25763, jul 2022. [Online]. Available: https://doi.org/10.1007/s11042-022-12780-9

[70] M. Oberweger, P. Wohlhart, and V. Lepetit, "Training a Feedback Loop for Hand Pose Estimation," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 3316–3324. [Online]. Available: https://doi.org/10.1109/ICCV.2015.379

[71] T. Simon, H. Joo, I. Matthews, and Y. Sheikh, "Hand Keypoint Detection in Single Images Using Multiview Bootstrapping," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4645–4653. [Online]. Available: https://doi.org/10.1109/CVPR.2017.494

[72] S. Yang, D. He, Q. Li, J. Wang, and D. Li, "Hand pose estimation based on improved NSRM network," *EURASIP Journal on Applied Signal Processing*, vol. 2023, no. 1, p. 8, Dec. 2023. [Online]. Available: https://doi.org/10.1186/s13634-023-00970-y

[73] Y. Chen, H. Ma, D. Kong, X. Yan, J. Wu, W. Fan, and X. Xie, "Nonparametric Structure Regularization Machine for 2D Hand Pose Estimation," in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 370–379. [Online]. Available: https://doi.org/10.1109/WACV45572.2020.9093271

[74] Q.-L. Zhang and Y.-B. Yang, "SA-Net: Shuffle Attention for Deep Convolutional Neural Networks," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 2235–2239. [Online]. Available: https://doi.org/10.1109/ICASSP39728.2021.9414568

[75] S. An, X. Zhang, D. Wei, H. Zhu, J. Yang, and K. A. Tsintotas, "FastHand: Fast Monocular Hand Pose Estimation on Embedded Systems," *Journal of Systems Architecture*, vol. 122, no. C, jan 2022. [Online]. Available: https://doi.org/10.1016/j.sysarc.2021.102361

[76] F. Mueller, D. Mehta, O. Sotnychenko, S. Sridhar, D. Casas, and C. Theobalt, "Real-Time Hand Tracking under Occlusion from an Egocentric RGB-D Sensor," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 1163–1172. [Online]. Available: https://doi.org/10.1109/ICCV.2017.131

[77] N. Santavas, I. Kansizoglou, L. Bampis, E. Karakasis, and A. Gasteratos, "Attention! A Lightweight 2D Hand Pose Estimation Approach," *IEEE Sensors Journal*, vol. 21, no. 10, pp. 11 488–11 496, 2021. [Online]. Available: https://doi.org/10.1109/JSEN.2020.3018172

[78] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional Block Attention Module," in *2018 European Conference on Computer Vision (ECCV)*, vol. 11211. Springer International Publishing, 2018. [Online]. Available: https://doi.org/10.1007/978-3-030-01234-2_1

[79] S. Zhuoran, Z. Mingyuan, Z. Haiyu, Y. Shuai, and L. Hongsheng, "Efficient attention: Attention with linear complexities," in *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2021, pp. 3530–3538. [Online]. Available: https://doi.org/10.1109/WACV48630.2021.00357

[80] S. Shit, J. C. Paetzold, A. Sekuboyina, I. Ezhov, A. Unger, A. Zhylka, J. P. W. Pluim, U. Bauer, and B. H. Menze, "clDice - a Novel Topology-Preserving Loss Function for Tubular Structure Segmentation," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 16 555–16 564. [Online]. Available: https://doi.org/10.1109/CVPR46437.2021.01629

[81] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, p. 281–305, feb 2012. [Online]. Available: https://doi.org/10.5555/2188385.2188395

[82] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431–3440. [Online]. Available: https://doi.org/10.1109/CVPR.2015.7298965

[83] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6230–6239. [Online]. Available: https://doi.org/10.1109/CVPR.2017.660

[84] L. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking Atrous Convolution for Semantic Image Segmentation," in *arXiv preprint arXiv:1706.05587*, 2017.

[85] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," in *arXiv preprint arXiv:2304.02643*, 2023.

[86] R. Hettiarachchi and J. Peters, "Multi-manifold-based skin classifier on feature space voronoï regions for skin segmentation," *Journal of Visual Communication and Image Representation*, vol. 41, no. C, pp. 123–139, 2016. [Online]. Available: https://doi.org/10.1016/j.jvcir.2016.09.011

[87] M. Kawulok, "Fast propagation-based skin regions segmentation in color images," in *2013 10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, 2013, pp. 1–7. [Online]. Available: https://doi.org/10.1109/FG.2013.6553733

[88] M. Arsalan, D. S. Kim, M. Owais, and K. R. Park, "OR-Skip-Net: Outer residual skip network for skin segmentation in non-ideal situations," *Expert Systems with Applications*, vol. 141, p. 112922, 2020. [Online]. Available: https://doi.org/10.1016/j.eswa.2019.112922

[89] A. Lumini and L. Nanni, "Fair comparison of skin detection approaches on publicly available datasets," *Expert Systems with Applications*, vol. 160, p. 113677, 2020.

[90] N. Park and S. Kim, "How do vision transformers work?" in *International Conference on Learning Representations*, 2022.

[91] J. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," in *arXiv preprint arXiv:1607.06450*, 2016.

[92] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation," in *2018 European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2018, pp. 833–851. [Online]. Available: https://doi.org/10.1007/978-3-319-46448-0_2

[93] D. Misra, T. Nalamada, A. U. Arasanipalai, and Q. Hou, "Rotate to attend: Convolutional triplet attention module," in *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2021, pp. 3138–3147. [Online]. Available: https://doi.org/10.1109/WACV48630.2021.00318

[94] B. Yang, G. Bender, Q. V. Le, and J. Ngiam, "CondConv: Conditionally Parameterized Convolutions for Efficient Inference," in *2019 33rd International Conference on Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://doi.org/10.5555/3454287.3454404

[95] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "ICNet for Real-Time Semantic Segmentation on High-Resolution Images," in *2018 European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2018, pp. 418–434. [Online]. Available: https://doi.org/10.1007/978-3-030-01219-9_25

[96] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, W. Liu, and B. Xiao, "Deep high-resolution representation learning for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 10, pp. 3349–3364, 2021. [Online]. Available: https://doi.org/10.1109/TPAMI.2020.2983686

[97] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017. [Online]. Available: https://doi.org/10.1109/TPAMI.2016.2644615

[98] N. Adaloglou, T. Chatzis, I. Papastratis, A. Stergioulas, G. T. Papadopoulos, V. Zacharopoulou, G. J. Xydopoulos, K. Atzakas, D. Papazachariou, and P. Daras, "A comprehensive study on

deep learning-based methods for sign language recognition," *IEEE Transactions on Multimedia*, vol. 24, pp. 1750–1762, 2022. [Online]. Available: https://doi.org/10.1109/TMM.2021.3070438

[99] A. Bandini and J. Zariffa, "Analysis of the Hands in Egocentric Vision: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 6, pp. 6846–6866, 2023. [Online]. Available: https://doi.org/10.1109/TPAMI.2020.2986648

[100] H. P. J. Dutta, M. K. Bhuyan, D. R. Neog, K. F. MacDorman, and R. H. Laskar, "A Hand Gesture-Operated System for Rehabilitation Using an End-to-End Detection Framework," *IEEE Transactions on Artificial Intelligence*, vol. 5, no. 2, pp. 698–708, 2024. [Online]. Available: https://doi.org/10.1109/TAI.2023.3251309

[101] H. P. J. Dutta, M. Bhuyan, D. R. Neog, K. F. MacDorman, and R. H. Laskar, "Efficient hand segmentation for rehabilitation tasks using a convolution neural network with attention," *Expert Systems with Applications*, vol. 234, p. 121046, 2023. [Online]. Available: https://doi.org/10.1016/j.eswa.2023.121046

[102] G. Plouffe and A.-M. Cretu, "Static and dynamic hand gesture recognition in depth data using dynamic time warping," *IEEE Transactions on Instrumentation and Measurement*, vol. 65, no. 2, pp. 305–316, 2016. [Online]. Available: https://doi.org/10.1109/TIM.2015.2498560

[103] A. V. and R. R., "A deep convolutional neural network approach for static hand gesture recognition," *Procedia Computer Science*, vol. 171, pp. 2353–2361, 2020, third International Conference on Computing and Network Communications (CoCoNet'19). [Online]. Available: https://doi.org/10.1016/j.procs.2020.04.255

[104] G. Bhaumik, M. Verma, M. Govil, and S. Vipparthi, "ExtriDeNet: an intensive feature extrication deep network for hand gesture recognition," *The Visual Computer*, 2021. [Online]. Available: https://doi.org/10.1007/s00371-021-02225-z

[105] ——, "HyFiNet: Hybrid feature attention network for hand gesture recognition," *Multimedia Tools and Applications*, 2022. [Online]. Available: https://doi.org/10.1007/s11042-021-11623-3

[106] J. Sahoo, S. Sahoo, S. Ari, and S. Patra, "Rbi-2rcnn: Residual block intensity feature using a two-stage residual convolutional neural network for static hand gesture recognition," *SIViP*, vol. 16, p. 2019–2027, 2022. [Online]. Available: https://doi.org/10.1007/s11760-022-02163-w

[107] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778. [Online]. Available: https://doi.org/10.1109/CVPR.2016.90

[108] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature Pyramid Networks for Object Detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 936–944. [Online]. Available: https://doi.org/10.1109/CVPR.2017.106

[109] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 618–626. [Online]. Available: https://doi.org/10.1109/ICCV.2017.74

[110] A. V. and R. R., "A deep convolutional neural network approach for static hand gesture recognition," *Procedia Computer Science*, vol. 171, pp. 2353–2361, 2020, third International

Conference on Computing and Network Communications (CoCoNet'19). [Online]. Available: https://doi.org/10.1016/j.procs.2020.04.255

[111] Y. Tan, K. Lim, C. Tee, C. Lee, and C. Low, "Convolutional neural network with spatial pyramid pooling for hand gesture recognition," *Neural Computing and application*, vol. 33, p. 5339–5351, 2021. [Online]. Available: https://doi.org/10.1007/s00521-020-05337-0

[112] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," in *arXiv preprint arXiv:1904.07850*, 2019.

[113] H. Law and J. Deng, "CornerNet: Detecting Objects as Paired Keypoints," *International Journal of Computer Vision*, vol. 128, 2020. [Online]. Available: https://doi.org/10.1007/s11263-019-01204-1

[114] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," in *arXiv preprint arXiv:1704.04861*, 2017.

[115] A. Newell, K. Yang, and J. Deng, "Stacked Hourglass Networks for Human Pose Estimation," in *2016 European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2016, pp. 483–499. [Online]. Available: https://doi.org/10.1007/978-3-319-46484-8_29

[116] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2261–2269. [Online]. Available: https://doi.org/10.1109/CVPR.2017.243

[117] M. Rezaei, R. Rastgoo, and V. Athitsos, "TriHorn-Net: A Model for Accurate Depth-Based 3D Hand Pose Estimation," *Expert Systems with Applications*, vol. 223, no. C, aug 2023. [Online]. Available: https://doi.org/10.1016/j.eswa.2023.119922

[118] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun, "Cascaded hand pose regression," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 824–832, doi: 10.1109/CVPR.2015.7 298 683. [Online]. Available: https://doi.org/10.1109/CVPR.2015.7298683

[119] T. Pfister, J. Charles, and A. Zisserman, "Flowing ConvNets for Human Pose Estimation in Videos," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1913–1921. [Online]. Available: https://doi.org/10.1109/ICCV.2015.222

[120] J. Hu, L. Shen, and G. Sun, "Squeeze and Excitation Networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7132–7141. [Online]. Available: https://doi.org/10.1109/CVPR.2018.00745

[121] J. Ho, N. Kalchbrenner, D. Weissenborn, and T. Salimans, "Axial Attention in Multidimensional Transformers," in *arXiv preprint arXiv:1912.12180*, 2019.

[122] Q.-L. Zhang and Y.-B. Yang, "SA-Net: Shuffle Attention for Deep Convolutional Neural Networks," in *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 2235–2239. [Online]. Available: https://doi.org/10.1109/ICASSP39728.2021.9414568

[123] Q. Hou, D. Zhou, and J. Feng, "Coordinate Attention for Efficient Mobile Network Design," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 13 708–13 717. [Online]. Available: https://doi.org/10.1109/CVPR46437.2021.01350

[124] Y. Li, T. Yao, Y. Pan, and T. Mei, "Contextual transformer networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 2, pp. 1489–1500, 2023. [Online]. Available: https://doi.org/10.1109/TPAMI.2022.3164083

[125] X. Li, X. Hu, and J. Yang, "Spatial Group-wise Enhance: Improving Semantic Feature Learning in Convolutional Networks," in *arXiv preprint arXiv:1905.09646*, 2019.

[126] S. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4724–4732. [Online]. Available: https://doi.org/10.1109/CVPR.2016.511

[127] C. Zimmermann and T. Brox, "Learning to Estimate 3D Hand Pose from Single RGB Images," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 4913–4921. [Online]. Available: https://doi.org/10.1109/ICCV.2017.525

[128] T. Ohkawa, K. He, F. Sener, T. Hodan, L. Tran, and C. Keskin, "AssemblyHands: Towards Egocentric Activity Understanding via 3D Hand Pose Estimation," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 12 999–13 008. [Online]. Available: https://doi.org/10.1109/CVPR52729.2023.01249

[129] G. Moon, S.-I. Yu, H. Wen, T. Shiratori, and K. M. Lee, "InterHand2.6M: A Dataset and Baseline for 3D Interacting Hand Pose Estimation from a Single RGB Image," in *2020 European Conference on Computer Vision (ECCV)*, vol. 12365. Springer International Publishing, 2020. [Online]. Available: https://doi.org/10.1007/978-3-030-58565-5_33

[130] B. Artacho and A. Savakis, "UniPose: Unified Human Pose Estimation in Single Images and Videos," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 7033–7042. [Online]. Available: https://doi.org/10.1109/CVPR42600.2020.00706

[131] L. Xu, S. Jin, W. Liu, C. Qian, W. Ouyang, P. Luo, and X. Wang, "ZoomNAS: Searching for Whole-Body Human Pose Estimation in the Wild," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 4, pp. 5296–5313, doi: 10.1109/TPAMI.2022.3 197 352, 2023. [Online]. Available: https://doi.org/10.1109/TPAMI.2022.3197352

[132] K. Seo, H. Cho, D. Choi, and J.-D. Park, "Implicit Semantic Data Augmentation for Hand Pose Estimation," *IEEE Access*, vol. 10, pp. 84 680–84 688, doi: 10.1109/ACCESS.2022.3 197 749, 2022. [Online]. Available: https://doi.org/10.1109/ACCESS.2022.3197749