Computer Systems
Science and Engineering

Tech Science Press

DOI: 10.32604/csse.2023.037258
Article

An Efficient and Robust Hand Gesture Recognition System of Sign Language Employing Finetuned Inception-V3 and Efficientnet-B0 Network

Adnan Hussain[1], Sareer Ul Amin[2], Muhammad Fayaz[3] and Sanghyun Seo[4,*]

[1]Department of Computer Science, Islamia College University, Peshawar, 25120, Pakistan
[2]Department of Computer Science and Engineering, Chung-Ang University, Seoul, 06974, Korea
[3]Department of Computer Engineering, Cyprus International University, Lefkosa, 99010, Rep. of North Cyprus
[4]College of Art and Technology, Chung-Ang.University, Anseong, 17546, Korea
*Corresponding Author: Sanghyun Seo. Email: sanghyun@cau.ac.kr
Received: 28 October 2022; Accepted: 13 January 2023

Abstract: Hand Gesture Recognition (HGR) is a promising research area with an extensive range of applications, such as surgery, video game techniques, and sign language translation, where sign language is a complicated structured form of hand gestures. The fundamental building blocks of structured expressions in sign language are the arrangement of the fingers, the orientation of the hand, and the hand's position concerning the body. The importance of HGR has increased due to the increasing number of touchless applications and the rapid growth of the hearing-impaired population. Therefore, real-time HGR is one of the most effective interaction methods between computers and humans. Developing a user-free interface with good recognition performance should be the goal of real-time HGR systems. Nowadays, Convolutional Neural Network (CNN) shows great recognition rates for different image-level classification tasks. It is challenging to train deep CNN networks like VGG-16, VGG-19, Inception-v3, and Efficientnet-B0 from scratch because only some significant labeled image datasets are available for static hand gesture images. However, an efficient and robust hand gesture recognition system of sign language employing finetuned Inception-v3 and Efficientnet-Bo network is proposed to identify hand gestures using a comparative small HGR dataset. Experiments show that Inception-v3 achieved 90% accuracy and 0.93% precision, 0.91% recall, and 0.90% f1-score, respectively, while EfficientNet-B0 achieved 99% accuracy and 0.98%, 0.97%, 0.98%, precision, recall, and f1-score respectively.

Keywords: Pretrained CNN; hand gesture recognition; transfer learning

1 Introduction

HGR is the initial stage of a computer's interpretation of human body language, having valuable human-computer interaction (HCI) applications, including virtual reality, video games, telesurgery, and TV control [1]. One of the most critical uses of HGR is sign language translation because they represent basic human emotions and communication information; the hand movements used in sign

language are arranged in a highly complicated fashion. The fundamental elements of these manual expressions are the local finger configuration and the global finger configuration, which refers to the orientation and placement of the hand concerning the body. All of these complementary primitives should be considered by an effective recognition system in a series of frames. It is challenging to compare the Euclidean space primitives because of the time dependence of these frames. The majority of recognition systems use the hand's regional configuration for HGR. These systems perform a hand segmentation preprocessing phase employing color-based models, or they accept a segmented region of the human hand as input [2,3]. Other HGR systems consider the overall body configuration and ignore the local finger configurations. With the limited number of well-defined and clear gestures, these methods have been performed efficiently for some HCI-related applications. They still need to improve when it comes to accurate sign language gesture identification [4].

Computer vision experts have utilized various algorithms and deep learning techniques to assist humans in solving problems [5]. Computer and human interaction improvement has been facilitated by using hand gestures in many software programs [6]. The HGR system is vital in developing human-computer interaction and is increasingly used in various domains. The application of HGR can now be seen in virtual reality [7,8], cognitive development assessment [9], games [10], assisted living [11,12], augmented reality [13], etc. The industry has recently become interested in HGR in various fields for human-robot interaction in manufacturing [14,15] and control of autonomous cars [16]. This aim of the real-time HGR system primarily focuses on classifying and identifying gestures in real-time enlivenment. To comprehend how a hand moves, we can employ a variety of algorithms and concepts from diverse disciplines, including image processing and neural networks [17]. The HGR system has a wide range of uses. For instance, we can use sign language to communicate with deaf persons who cannot hear.

In the literature, most researchers used conventional methods for HGR; they used classical feature extraction methods such as spatial-temporal features, shape descriptors, etc. [18]. In a specific environment, these hand-crafted features performed well, but the performance has degraded in the diverse condition of the dataset. Deep-Learning (DL) based approaches such as CNN [19] as well as stacked denoising autoencoder [20] architectures are used to overcome these limitations. Still, training CNN models from scratch is challenging for the following reasons. (i) A massive amount of image data is required to train the CNN model properly. (ii) Convergence problems can arise during CNN training, requiring repeated CNN layer adjustments and hyperparameters learning. As a result, creating a CNN model takes a lot of work. The dataset with fewer images used a transfer-learning technique to solve the above-mentioned problems. For this purpose, different CNN models are trained on large label datasets, such as VGG [21], ResNet [22], AlexNet [23], Inception-v3 [24], GoogleNet [25], and Efficientnet-B0 [26] are-finetuned on the HGR dataset. However, to recognize hand gestures in real-time, an efficient and robust hand gesture recognition system of sign language employing finetuned Inception-v3 and Efficientnet-Bo network is proposed. The main contributions of this research are summarized as under:

- A refined Inception-v3 and Efficientnet-Bo network-based hand gesture recognition system for sign language translation is proposed to recognize human hand gestures in a real-time environment efficiently.
- The effectiveness and reliability of the proposed system are assessed by employing publicly available standard American Sign Language (ASL) datasets, and the results show a significant performance than existing methods.
- By incorporating the proposed methodology, a real-time hand gestures recognition system has been introduced, and subject-neutral mode testing results have been conducted.

The remaining sections of the work are structured as follows. Section 2 presents the related work of HGR techniques. Section 3 describes the detailed methodology of the HGR system. The standard dataset, validation methods, and detailed experimental results of HGR are discussed in Section 4. Finally, the conclusion is shown in Section 5.

2 Related Works

Researchers present different methods for the HGR system in the current literature. There have been discussed multiple techniques for the implementation of the HGR system, including both the Conventional as well as the DL techniques. In this section, existing work has been conducted to understand the mechanism of HGR methods.

2.1 Conventional Machine Learning-Based HGR Systems

Recently, several authors presented conventional machine learning base HGR systems such as Pedersoli et al. [27] proposed a method based on ML for dynamic hand gestures and static hand pose recognition. The authors used the ASL dataset for hand pose recognition; first, they segmented the hand region from the image, and then the segmented hand region was provided to Support Vector Machine (SVM) classifier for hand gesture classification. Huang et al. [28] presented three steps based HGR system. The first step was feature extraction, followed by training and recognition. In feature extraction, a hybrid technique is combined, which extracts edges and temporal features from each image and then provides these features to an ML-based Principle Component Analysis algorithm for training. A pre-trained Principal Component Analysis (PCA) model is used in the recognition step to recognize 18 different gestures. Skaria et al. [29] proposed an HGR system using hand gesture signatures generated by an ultra-wideband (UWB) impulse radar. First, the authors take signals from 14 hand gestures and turn them into 3D tensors consisting of range-doppler signature frame sequences. Next, the authors extract features from these signatures using CNN and fed these features to different classifiers, i.e., SVM, K-Nearest Neighbors (KNN), Long Short-Term Memory (LSTM), and Fully Convolutional Neural Networks (FCNN) for prediction. Haria et al. [30] proposed an HGR-based human and computer interaction method for dynamic hand gestures. The system translates gestures into actions, such as launching applications like PowerPoint, vlc media player. The authors claimed a high-level accuracy over other methods for controlling applications. Parvathy et al. [31] proposed a machine learning-based HGR system. This system consists of three main stages, i.e., segmentation, feature extraction, and classification. The authors extracted rotation and scale invariant key features from hand gesture images using discrete wavelet transform and modified speed of robust features descriptors. Next, the BOW technique is used to develop the fixed-dimension input vector that is required for the SVM classifier. Experimental results show that the authors achieved an accuracy of around 96.5% using the SVM classifier.

2.2 Deep Learning-Based HGR Systems

Currently, researchers have used deep learning methods for the HGR system. De Smedt et al. [32] presented a deep learning-based HGR system on the DHG dataset. In this system, RGB-D images are used where R is red, G is green, B is blue, and D is the depth channel of the image. A pre-trained VGG model is trained on a sequence of five consecutive images. Subsequently, Never ova et al. present multi-modal data skeletal and audio stream data, RGB-D images for multi-model classification task in [33,34]. The first convolutional layers are used to process each modality independently of the merged. An introduction of the multi-modal dropout (ModDrop) prevents meaningless co-adaptation of modalities. Waldron et al. [2] proposed multi-DL architectures for sequence feature globalization,

recognition, and hand segmentation. The authors used forty different hand gestures in an uncontrolled environment to evaluate the system. The experimental results reveal the system's performance as well as surpassing the state-of-the-art techniques. Mujahid et al. [35] proposed a technique for the detection and classification of hand gestures using DL models. They used two different models, i.e., DarkNet-53 and YOLO-V3, for HGR, even though no additional preprocessing steps are used for the enhancement of images. The proposed system detects hand gestures more accurately, even in low-resolution images. A deep learning-based method was suggested by Ozcan et al. [36] to enhance the performance of the HGR system through transfer learning. They finetuned the Alexnet model and performed experiments on sign language digits and Thomas Moeslund's gesture recognition datasets and achieved 94% and 98% accuracy for both datasets, respectively. Sahoo et al. [37] presented a real-time hand gesture recognition method using transfer learning techniques. They used AlexNet and VGG-16 to train on American Sign Language datasets. Neethu et al. [38] proposed deep learning-based HGR detection and recognition system. They first segment hand region and figures from images using their mask image and then provide those images to the CNN model for training. They achieved high-level SOTA performance. Oyedotun et al. [39] Proposed a method for HER based on deep learning models, and they used twenty-four different hand gestures taken from Thomas Moeslund's gesture recognition database. They used denoising auto-encoder and CNN; they are capable of learning the complex HGR task with lower error rates. Both models are trained and tested from the American Sign Language public database. Güler et al. [40] proposed a method for HGR using a capsule network with CNN. They used three different datasets, i.e., Cifar-10, HG14, and FashionMnist datasets, and four different CNN models, i.e., Reset50, Vgg16, DenseNet, and CapsNet. All four models are trained on each dataset and then compared to their results; the proposed hybrid model achieved the highest accuracy with 93.88% in the FashionMnist dataset, 81.42% in the Cifar-10, and 90% in the HG14 dataset. Sagayam et al. [41] proposed a technique for HGR using a well-tuned Deep Convolutional Neural Network (DCNN) model. The Cambridge Hand Gesture dataset is used to assess the CNN model performance. The accuracy achieved by the CNN model is 96.6%, while the specificity and sensitivity are 98% and 85%, respectively.

3 Proposed Methodology

This section provides a detailed overview of the HGR system. The suggested system entails three primary sections, which are data acquisition, preprocessing, and HGR, as shown in Fig. 1.

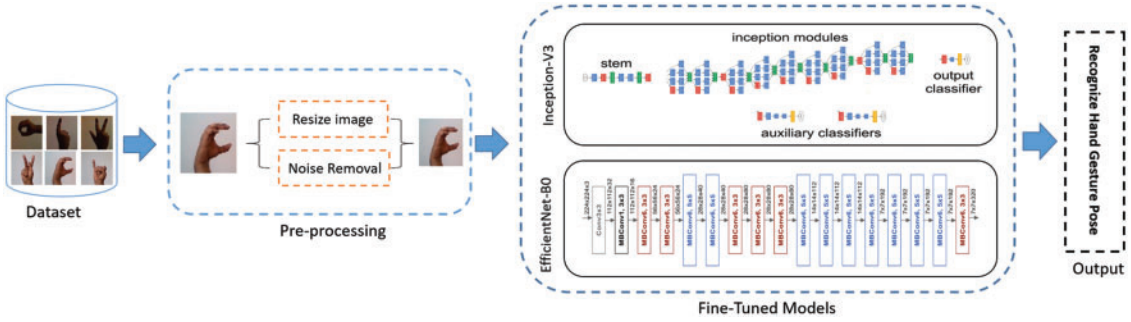

Figure 1: An efficient and robust hand gesture recognition system of sign language employing finetuned Inception-v3 and Efficientnet-Bo network for a static HGR system

3.1 Datasets

The suggested system is assessed by employing the easily available American Sign Language (ASL) dataset. The Kinect camera was used to create the ASL dataset [42], which includes 5440 color static gesture images and corresponding depth Maps. ASL consists of 37 poses, of which ten are based on numbers (0–9), and 26 are based on the alphabet.

This dataset is developed by using ten subjects, and each gesture sample is repeated 16 times. It is an incredibly challenging dataset due to human noises, complicated background, and certain samples of gesture that have been gathered by bending the wrists or elbows to a specific angle.

3.2 Preprocessing

Preprocessing is the most crucial step after data collection. It is used to prepare data before proceeding with the training and testing of the DL and ML models. Capturing images/videos through a mobile camera or other vision sensor containing factors that affect the results of the models. Different preprocessing techniques, such as resizing, noise removal, and data augmentation techniques, are applied to improve the quality of images or videos. In this study, we used image resizing, noise removal, and data augmentation techniques. The images were resized to 224 × 224 as they were not the same size, due to which the performance of the model was affected. Images have different types of noise, i.e., Gaussian noise, salt, pepper, etc.; for removing this kind of noise, Median and Gaussian filters are applied to images which make images smooth and clear. Also, we used the data augmentation technique to create fake data during training, due to which the performance of the trained model is improved, and chances of overfitting are reduced. We used the image rotation method for data augmentation. The rotation angle (θ) is 30°.

3.3 Architectures of Proposed Models

CNNs were introduced in the late 1980s by LeCun et al. [43], it contains different layers, such as the Convolutional layer, pooling layer, fully connected layer, and activation layer. Convolutional layers are used to extract unique features from input images. These Features are extracted using convolutional operation between kernel and input image. The kernel is a small rectangular size matrix of 5 × 5, 3 × 3, which slides over the image and produces the feature map. The most relevant data about the feature map is preserved by the pooling layer, which is then utilized to reduce the size of the feature map. There are several pooling layer types that are employed, including average pooling and maximum pooling. Fully connected layers classify images, while the output sigmoid layer predicts the final result [44]. Recently, CNN achieved great SOTA results relative to traditional methods in the field of computer vision, for example, medical image classification [45,46], face detection, text classification, etc. In this work, we used two different pre-trained models, Efficientnet-B0 and Inception-v3, to classify hand gesture images.

3.3.1 Inception-v3 Model

The Inception-v3 model is an updated version of the Inception-v1 model. The Inception-v3 model optimizes the network in a variety of ways for higher model adaptability. In comparison to the Inception-v1 and v2 models, it has a more extensive network. It is a deep CNN model that was trained on a low-configuration computer, as seen in Fig. 2. Training can be time-consuming and challenging, sometimes taking up to several days. Transfer learning is used to solve this problem, keeping the final layer of the model for use with new categories. The inception-V3 model has 48 layers where we freeze all the top layers of the model and include new layers according to our dataset classes.

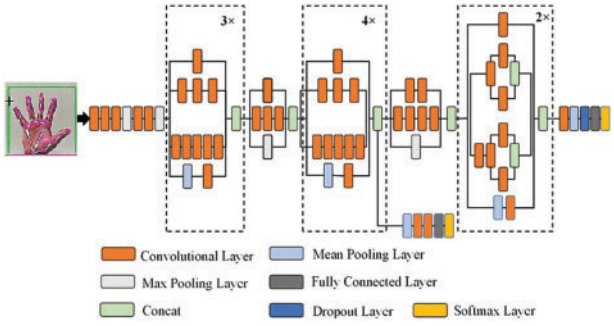

Figure 2: Architecture of the Inception-v3 model

Furthermore, a model like Inception-V3 is useful for complex DL problems. Other deep learning models like Vgg16, Vgg19, and AlexNet, etc., may not perform well for difficult tasks or extracting complex features because these models use a simple stack of convolutional layers, pooling layers followed by fully connected layers, but Inception-V3 models consist of 1×1 convolutions also known as pointwise convolutions followed by convolutional layers with different size of kernels applied simultaneously, and it contains more hidden layers. It allows inception to learn more complex features; that's why inception is used for more complex problems.

3.3.2 Efficientnet-B0 Model

The Efficientnet family of architectures was developed to find an appropriate method to scale CNNs and improve model performance. The authors suggest a compound scaling technique that uniformly scales width, depth, and resolution using a given set of coefficients. By using this technique, the authors were able to create the Efficientnet-B0 CNN architecture. The Efficientnet model group consists of eight models from B0 to B7, with each subsequent model number referring to variants with more parameters and higher accuracy. The base Efficientnet-B0 is shown in Fig. 3.

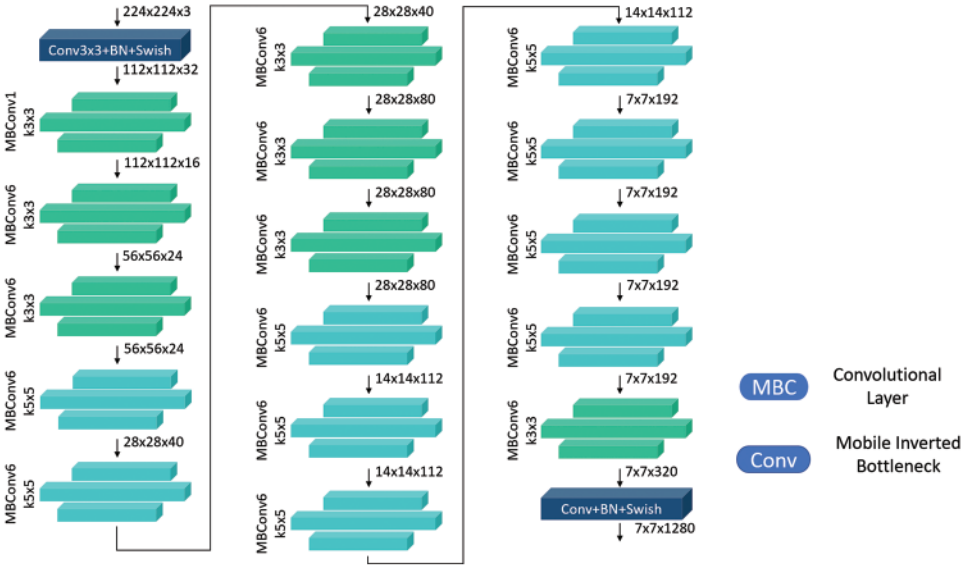

Figure 3: Architecture of the Efficientnet-B0

CNNs can capture richer and more complex characteristics by adjusting the network depth. However, the vanishing gradient [47] problem makes network training more difficult. The network can collect more fine-grained features by adjusting its width. Training is also easy. Fig. 3 shows the detailed baseline EfficientNet-B0 model that accepts 224 × 224 × 3 input images where 224x × 224 is the width and height of an image and 3 is the image dimension. This algorithm uses multiple convolutional layers with a 3 × 3 receptive field and the mobile inverted bottleneck convolutional to capture characteristics across layers. Below, Eqs. (1) to (5) shows scaling the width, depth, and resolution.

$$w = \beta^{\varphi}, \tag{1}$$

$$d = \alpha^{\varphi}, \tag{2}$$

$$r = \gamma^{\varphi}, \tag{3}$$

$$\text{s.t } \alpha.\beta^2.\gamma^2 \approx 2 \tag{4}$$

$$\alpha \geq 1, \beta \geq 1, \gamma \geq 1. \tag{5}$$

where w denotes width, d denotes height, and r denotes resolution, and the α, β, and γ are constant coefficients determined by a small grid search on the original small model. Network width, depth, and resolution are all uniformly scaled by EfficientNet using a compound coefficient ϕ.

On the other hand, wide and shallow networks are unable to capture high-level features. High-resolution images enable CNNs to detect more minute patterns, and more processing power and memory are needed to process larger images. That's why in this work, we used the Efficientnet-B0 model to evaluate their performance on the American Sign Language dataset.

Moreover, EfficientNet is particularly useful for employing deep learning on edge, as it reduces compute cost, battery usage, and also training and inference speeds. This kind of model efficiency ultimately enables the use of deep learning on mobile and other edge devices.

4 Experimental Results and Discussion

4.1 Data Acquisition

Data acquisition is the first step in any research. It is comprised of two components: data and acquisition. Data is unrefined, raw information that can be either organized or unorganized. Collecting data for the particular activity at hand is referred to as acquiring data. For the development of the HGR system, several sensors are available. Table 1 compares different sensors together with their benefits and limitations. The HGR method, which was designed by utilizing data gloves, is more accurate and reliable, as shown in Table 1, but it also restricted the user's hands by requiring them to wear uncomfortable gloves [48]. Hand tracking for HGR based on leap motion sensors is performed with high accuracy, but the coverage area of the hand is small. The vision-based sensor, in contrast to the other sensors, is amazingly effective because it doesn't require the user to place any objects in their hands. The vision sensor captures hand gestures with a free hand [49]. That's why researchers have used vision sensors mostly to develop efficient and cost-effective HGR systems. In this work, we developed the proposed HGR system using a camera sensor because it can correctly and easily segment the hand region from the image frame.

Table 1: A review of the various sensors employed by the HGR system

Data Sensors	Wearable	Advantages	Limitations
Depth sensor (Kinect)	No	No color marker, easy hand segmentation	The first thing in the camera's frame should be a hand
Vision sensors (Camera)	No	Free to use	The effects of ambient and human noise
Leap motion	No	Hand tracking with absolute precision.	Less coverage area and consistently placing your hand over the sensor
Data glove	Yes	Robust as well as low cost	less user-friendly and less comfortability

4.2 Evaluation Parameters

Evaluation is the most important part of any research work. It is employed to check the effectiveness of the system. This study used Accuracy, Precision, Recall, and F1-Score as evaluation parameters.

Accuracy is defined as it is the ratio of correct predicted observations to all observations. Accuracy is evaluated through Eq. (6), where True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) are used.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

The proportion of accurately predicted positive observations to all of the predicted positive observations is known as precision shown in Eq. (7).

$$Precision = \frac{TP}{TP + TN} \tag{7}$$

The proportion of accurately predicted positive observations to all of the actual class observations is known as recall shown in Eq. (8).

$$Recall = \frac{TF}{TP + FN} \tag{8}$$

The weighted average of Precision and Recall is known as F1-Score. It is calculated through Eq. (9).

$$F1 - score = 2 \cdot \frac{Precision * Recall}{Precision + Recall} \tag{9}$$

4.2.1 Setting of Hyperparameters for Fine-Tuning

To finetune both CNN networks same hyperparameters are used, as shown in Table 2. The 32 batch size and 0.0001 learning rate with Adam Optimizer are used for the American Sign Language dataset. The proposed inception-V3 and EfficientNet-B0 are finetuned on the ASL dataset, where we freeze all the top layers of the model and include new layers according to our dataset classes. The same hyperparameters for finetuning are used for both models. This work is developed by using python and TensorFlow.

Table 2: The hyperparameters of proposed fine-tuned models

Models	Total no of classes	Batch size	Epochs	Loss	Learning rate	Optimizer	Time (ms) per inference step (CPU)	Time (ms) per inference step (GPU)
Inception-V3	36	32	30	Categorical cross-entropy	0.0001	Adam	42.2	6.9
EfficientNet-B0	36	32	30	Categorical cross-entropy	0.0001	Adam	46.0	4.9

4.2.2 Experimental Setup

In this work, both models are trained by using Python 3.8, Tensorflow 2.9, sklearn 1.0, matplotlib 3.5, and NumPy 1.21 libraries and Intel(R) Core(TM) i7 @ 2.80 GHz, dual processor, GeForce 442 GTX 1080 Graphics Cards, 24 GB RAM.

4.3 Experimental Results

Experiments are performed on two different pre-trained Inception-v3 and Efficientnet-B0 models. Accuracy, Precision, Recall, and f1-score are used to evaluate. The Inception-v3 and Efficientnet-B0 models are trained for 30 epochs. The training model is then saved in the .h5 extension for prediction. After the training of both Inception-v3 and Efficientnet-B0 models, the training accuracy of both models is 82% and 99%, while the validation accuracy of both models is 90% and 99%, respectively, as seen in Fig. 4. In addition, the static visual results of both two models are shown in Fig. 5 where the first row shows actual labels of the gestures, the second row show the predicted gestures of the inception-v3 model, and the third row shows the predicted gestures of the Efficientnet-B0 model. Red color labels show the wrong predicted gestures, while black color shows the correct predicted gestures. From Fig. 5, we examine that the Efficientnet-B0 performance is better than the Inception-v3 model. It also shows that the inception models are confused in some gesture poses like 'A,' 'G,' 'U,' 'R,' and 'Y.' A gesture pose 'A' is misclassified to pose 'C,' gesture 'G' to gesture 'P,' gesture 'U' to gesture 'D,' gesture 'R' to gesture 'K,' and gesture 'Y' to gesture '7'. Furthermore, Figs. 6 and 7 show the confusion matrix of both models, respectively. Finally, Table 3 shows gesture pose-wise precision, recall, and fi-score of both Inception-v3 and Efficientnet-B0 models. This shows that Inception-v3 does not perform well because the accuracy of some classes is not good as compared to Efficientnet-B0. On the other hand, Efficientnet-B0 performs outclass for all gesture poses.

4.4 Comparison with Contemporary Techniques

This section discusses the comparison of the suggested system with Contemporary methods. Table 4 displays a comparison of the suggested technique with recent existing techniques. The first three papers [38,42], and [4] used different deep learning models and achieved 91.6%, 96.6, and 95.5% accuracy, respectively. also, the F1-score of the three models is 86.9%, 91.1%, and 91% respective. While comparing these results with our proposed system, our proposed Inception-V3 model achieved 90% accuracy and 90% f1-score, which is less than contemporary models, but the proposed EfficientNet-B0 achieved the highest accuracy than all models, which is 99% and 98% f1-score. The results show that the proposed system performs well than the existing Contemporary methods. The model accuracy is used to compare the performance of the suggested method with Contemporary methods.

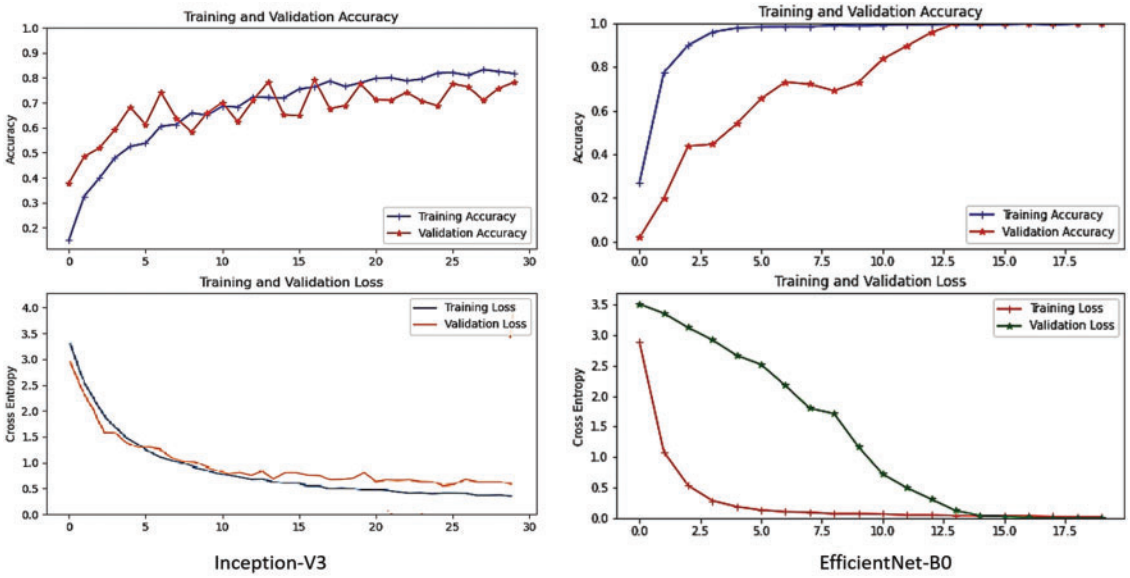

Figure 4: Training and validation graphs of both Inception-v3 and Efficientnet-B0 models

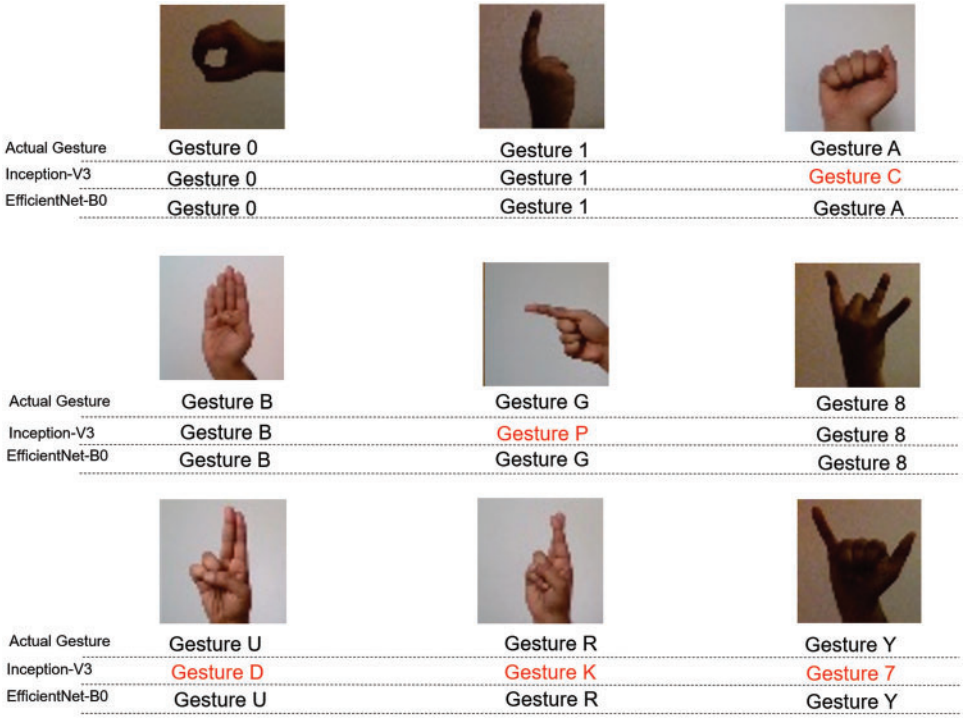

Figure 5: Static visual results of both Inception-v3 and Efficientnet-B0 models

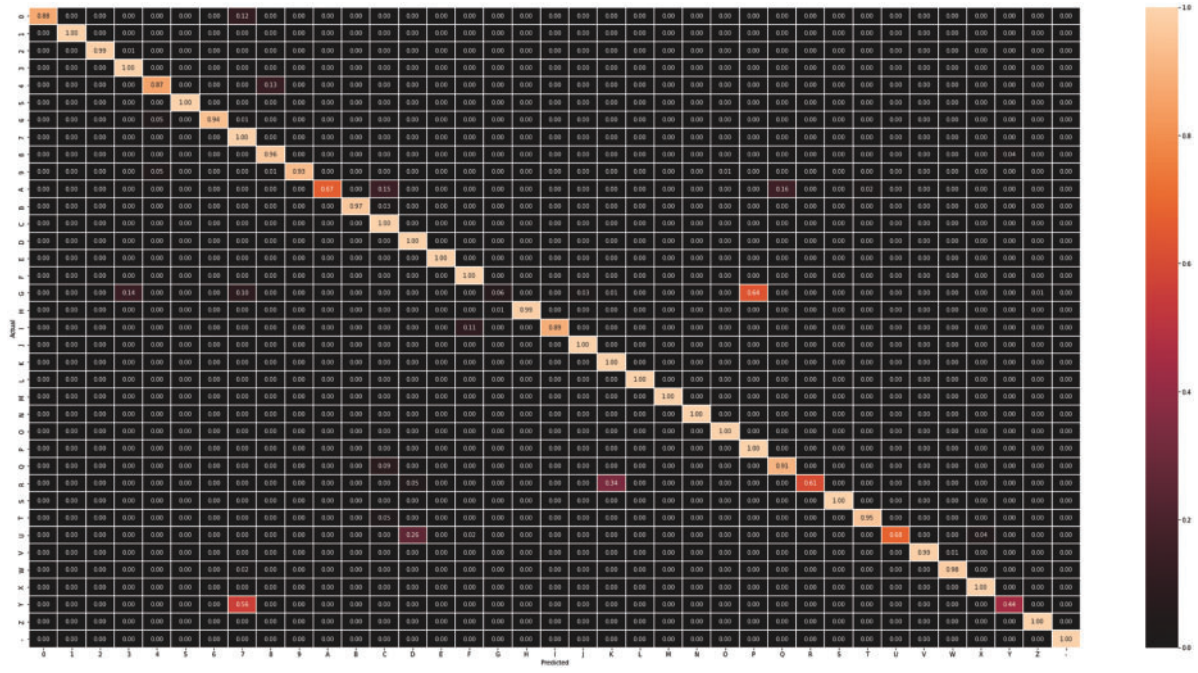

Figure 6: Confusion matrix of Inception-v3 model

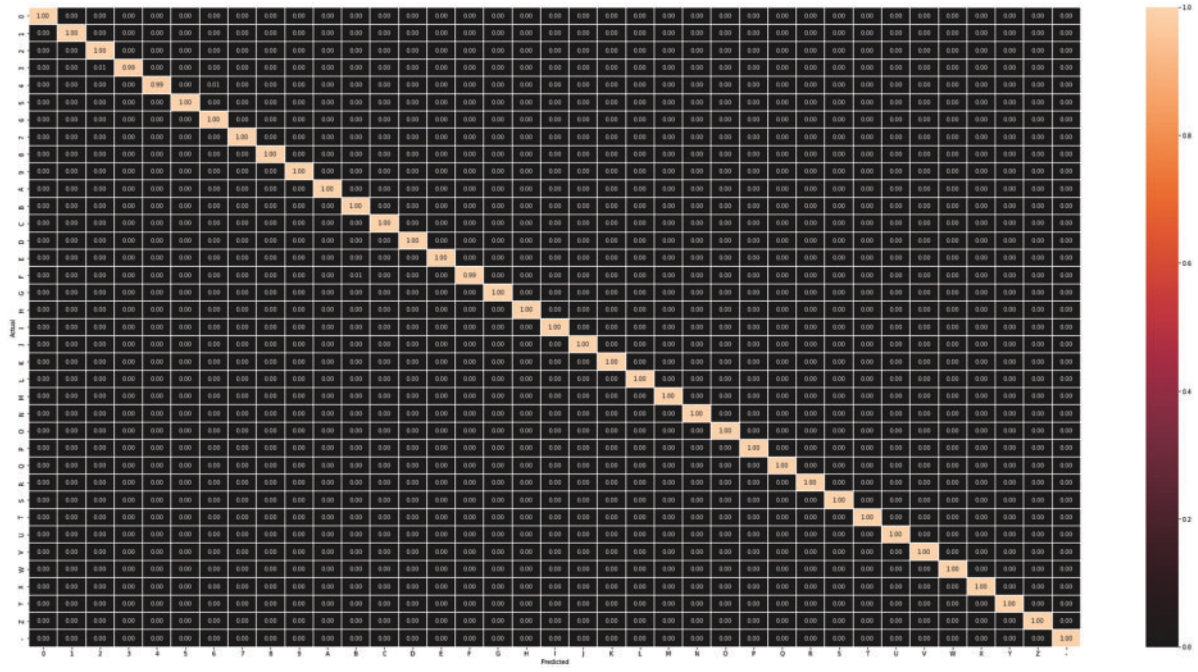

Figure 7: Confusion matrix of Efficientnet-B0 model

Table 3: Performance comparison between Inception-v3 and EfficientNet-B0 models

	EfficientNet-B0			Inception-v3		
Classes	Precision	Recall	F1-score	Precision	Recall	F1-score
0	1.00	1.00	1.00	1.00	0.88	0.93
1	1.00	1.00	1.00	1.00	1.00	1.00
2	0.99	1.00	0.99	1.00	0.99	0.99
3	1.00	0.93	0.96	0.87	1.00	0.93
4	1.00	0.90	0.94	0.90	0.87	0.88
5	1.00	1.00	1.00	1.00	1.00	1.00
6	0.98	1.00	0.98	1.00	0.94	0.97
7	1.00	1.00	1.00	0.55	1.00	0.71
8	1.00	1.00	1.00	0.87	0.96	0.91
9	1.00	1.00	1.00	1.00	0.93	0.96
A	1.00	0.92	0.95	1.00	0.67	0.80
B	0.94	1.00	0.96	1.00	0.97	0.98
C	1.00	1.00	1.00	0.75	1.00	0.86
D	1.00	1.00	1.00	0.76	1.00	0.86
E	1.00	1.00	1.00	1.00	1.00	1.00
F	0.98	0.91	0.94	0.89	1.00	0.94
G	1.00	1.00	1.00	0.88	0.06	0.12
H	1.00	1.00	1.00	1.00	0.99	0.99
I	1.00	1.00	1.00	1.00	0.89	0.94
J	0.96	1.00	0.97	0.97	1.00	0.98
K	1.00	1.00	1.00	0.74	1.00	0.85
L	1.00	1.00	1.00	1.00	1.00	1.00
M	1.00	0.87	0.93	1.00	1.00	1.00
N	1.00	1.00	1.00	1.00	1.00	1.00
O	1.00	1.00	1.00	0.99	1.00	0.99
P	1.00	1.00	1.00	0.61	1.00	0.76
Q	1.00	0.89	0.94	0.85	0.91	0.88
R	1.00	1.00	1.00	1.00	0.61	0.76
S	1.00	0.87	0.93	1.00	1.00	1.00
T	0.89	1.00	0.94	0.98	0.95	0.97
U	1.00	1.00	1.00	1.00	0.68	0.81
V	1.00	0.93	0.96	1.00	0.99	0.99
W	1.00	1.00	1.00	0.99	0.98	0.98
X	0.83	1.00	0.90	0.96	1.00	0.98
Y	1.00	1.00	1.00	0.92	0.44	0.59
Z	1.00	1.00	1.00	0.99	1.00	0.99
–	1.00	0.94	0.96	1.00	1.00	1.00
Average	**0.98%**	**0.97%**	**0.98%**	**0.93%**	**0.91%**	**0.90%**

Table 4: Comparison of proposed method with contemporary methods

Test models	Total no of classes	Model accuracy %	Precision%	Recall%	F1-score%
CNN [38]	33	91.6	91.5	82.7	86.9
DeepCNN [42]	09	96.6	85	98.12	91.1
SIFT_CNN [4]	24	95.5	84	98	91
Proposed Inception-V3	34	90	93	91	90
Proposed EfficientNet-B0	34	99	98	97	98

4.5 Recognition of American Sign Language Gestures in Real-Time

The real-time recognition of American sign language gestures is implemented in python language with an RGB color camera. Fig. 8 shows the step-by-step procedure for HGR. As shown in Fig. 8, a color video is captured using a camera and then converted into frames; these frames are used as the input image for further processing. The recognition of the gesture in a real-time description is below.

- The camera capture video and then convert it into frames.
- Applying preprocessing steps on input frames where frames are resized according to finetuned models' sizes.
- Detect hand region from the input frames; for this purpose, a pre-trained model from media pipe is used, which detects hand key points from the image.
- Crop the detected hand key points and provide them to train models for prediction.

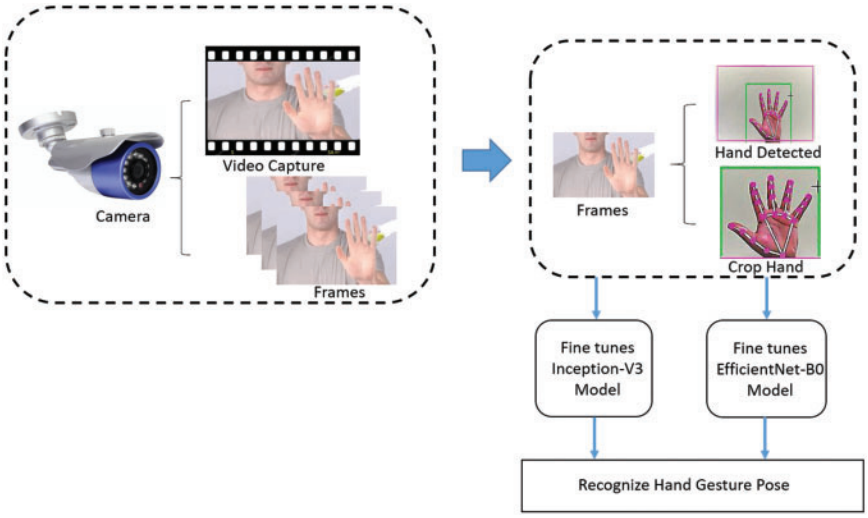

Figure 8: Real-time hand gestures recognition framework

In real-time experiments, we used a laptop camera whose resolution is 1920 × 1080 and frame rate is 30 fps. The capture frames were resized into 224 × 224, which is the desired size of the proposed models. Then performed, preprocessing on the capture frames; after preprocessing hand detector was

used to detect the hand region from the frame, and the hand region was cropped from the frame and provided to the trained model for prediction, and then shows the results in real-time with the predicted label. Real-time results are shown in Fig. 9. Where black color labels show the actual label of the gestures and red color shows the predicted label of the model. For prediction, we used Laptop Pc core i7-4 Gen with a 2.10 GHz processor and 8 GB RAM, which take 3.9 s for the loading trained model; the camera takes 6.7s for capturing the first frame, but it is then reduced to 0.13 s per frame when video starts properly, and prediction takes 0.12 s to predict hand gesture in real-time. These results show that our proposed system performs out class in real-time as well as in static conditions also.

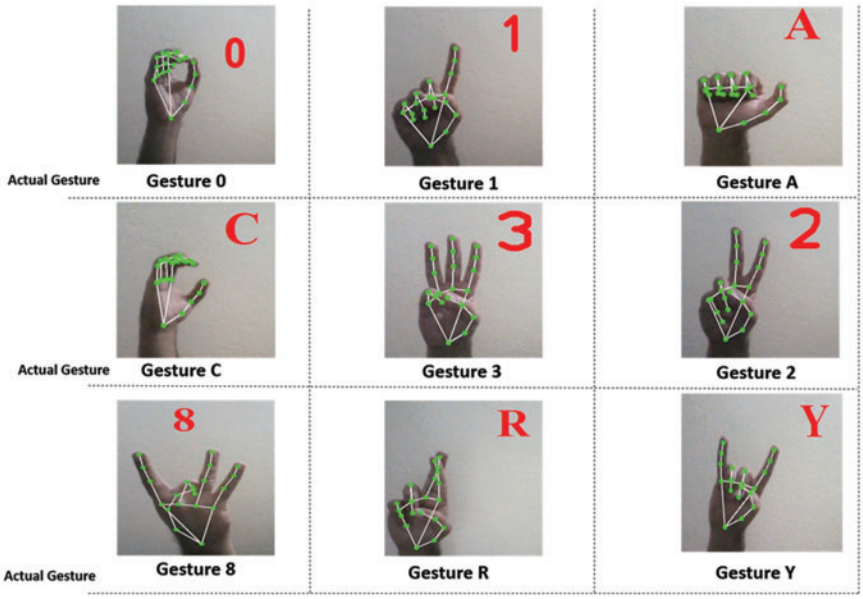

Figure 9: Results of proposed EfficientNet-B0 on real time data

4.6 Limitations of Proposed HGR System:

Like other conventional ML and DL models, our proposed real-time HGR system also has some limitations, which are listed below.

- In some cases, if the background and foreground colors are the same, then the hand detector cannot efficiently detect the hand region, which causes misclassification.
- The illumination conditions day and night time also affect the recognition.
- Sometimes hands are too close or too far from the camera, which is not in an ideal condition and also affects the recognition.
- Sometimes camera shutter cannot capture frames with the same speed of hand movement also affect the recognition.

5 Conclusion and Future Direction

In this work, we have developed a transfer learning-based HGR method, where we used two different deep learning models, i.e., Inception-v3 and Efficientnet-B0. The developed HGR system classifies both static and real-time hand gestures from images and videos with an accuracy of 90% and 99%, respectively. Despite the accuracy obtained by both models but there is still some space for

improvement. The mode can be improved to classify more than one gesture in real-time. The proposed system can help to improve the living system, which is used for computer and human interaction for both impaired and healthy people. Moreover, we compare the performance of the models with each other on the bases of accuracy, precision, recall, and fi-score, which shows that the Efficientnet-B0 performance is better than Inception-v3. Efficientnet-B0 classifies hand gesture poses in images and videos with 99%, 99%, 99%, and 99% accuracy, precision, recall, and f1-score, respectively, while Inception-v3 achieved 90%, 93%, 91%, and 90%, accuracy, precision, recall, and f1-score respectively. For future work, we will be focusing on smartphone applications or robotics for the HGR systems. Furthermore, to improve the accuracy of the HGR system, we will create a more advanced CNN with data fusion.

Funding Statement: This research work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (NRF-2022R1A2C1004657).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] A. S. Al-Shamayleh, R. Ahmad, M. A. Abushariah, K. A. Alam and N. Jomhari, "A systematic literature review on vision based gesture recognition techniques," *Multimedia Tools and Applications*, vol. 77, no. 21, pp. 28121–28184, 2018.

[2] M. B. Waldron and S. Kim, "Isolated ASL sign recognition system for deaf persons," *IEEE Transactions on Rehabilitation Engineering*, vol. 3, no. 3, pp. 261–271, 1995.

[3] M. Al-Hammadi, G. Muhammad, W. Abdul, M. Alsulaiman, M. A. Bencherif *et al.,* "Deep learning-based approach for sign language gesture recognition with efficient hand gesture representation," *IEEE Access*, vol. 8, pp. 192527–192542, 2020.

[4] B. Hu and J. Wang, "Deep learning based hand gesture recognition and UAV flight controls," *International Journal of Automation and Computing*, vol. 17, no. 1, pp. 17–29, 2020.

[5] M. H. Jarrahi, "Artificial intelligence and the future of work: Human-AI symbiosis in organizational decision making," *Business Horizons*, vol. 61, no. 4, pp. 577–586, 2018.

[6] S. Ahmed and S. H. Cho, "Hand gesture recognition using an IR-UWB radar with an inception module-based classifier," *Sensors*, vol. 20, no. 2, pp. 564, 2020.

[7] R. Alkemade, F. J. Verbeek and S. G. Lukosch, "On the efficiency of a VR hand gesture-based interface for 3D object manipulations in conceptual design," *International Journal of Human-Computer Interaction*, vol. 33, no. 11, pp. 882–901, 2017.

[8] Y. S. Lee and B.-S. Sohn, "Immersive gesture interfaces for navigation of 3D maps in HMD-based mobile virtual environments," in *Mobile Information Systems*, Vol. 2018. London, United Kingdom: Hindawi. 2018.

[9] F. Negin, P. Rodriguez, M. Koperski, A. Kerboua, J. Gonzàlezet *et al.,* "PRAXIS: Towards automatic cognitive assessment using gesture recognition," *Expert Systems with Applications*, vol. 106, no. 4, pp. 21–35, 2018.

[10] M. Roccetti, G. Marfia and A. Semeraro, "Playing into the wild: A gesture-based interface for gaming in public spaces," *Journal of Visual Communication and Image Representation*, vol. 23, no. 3, pp. 426–440, 2012.

[11] S. Gnanapriya and K. Rahimunnisa, "A hybrid deep learning model for real time hand gestures recognition," *Intelligent Automation & Soft Computing*, vol. 36, no. 1, pp. 1105–1119, 2023.

[12] S. Ul Amin, M. Ullah, M. Sajjad, F. A. Cheikh, M. Hijji *et al.,* "EADN: An efficient deep learning model for Anomaly detection in videos," *Mathematics*, vol. 10, no. 9, pp. 1555, 2022.

[13] M. S. Del Rio Guerra, J. Martin-Gutierrez and R. Acevedo, "Hand gestures in virtual and augmented 3D environments for down syndrome users," *Applied Sciences*, vol. 9, no. 13, pp. 2641, 2019.

[14] P. Tsarouchi, S. Makris and G. Chryssolouris, "Human-robot interaction review and challenges on task planning and programming," *International Journal of Computer Integrated Manufacturing*, vol. 29, no. 8, pp. 916–931, 2016.

[15] P. Neto, M. Simão, N. Mendes and M. Safeea, "Gesture-based human-robot interaction for human assistance in manufacturing," *The International Journal of Advanced Manufacturing Technology*, vol. 101, no. 1, pp. 119–135, 2019.

[16] G. Young, H. Milne, D. Griffiths and E. Padfield, "Designing mid-air haptic gesture controlled user interfaces for cars," *Proceedings of the ACM on Human-Computer Interaction*, vol. 4, no. EICS, pp. 1–23, 2020.

[17] H. Yu, X. Fan, L. Zhao and X. Guo, "A novel hand gesture recognition method based on 2-channel sEMG," *Technology and Health Care*, vol. 26, no. S1, pp. 205–214, 2018.

[18] A. R. Patil and S. Subbaraman, "A spatiotemporal approach for vision-based hand gesture recognition using Hough transform and neural network," *Signal, Image and Video Processing*, vol. 13, no. 2, pp. 413–421, 2019.

[19] W. Tao, M. C. Leu and Z. Yin, "American sign language alphabet recognition using convolutional neural networks with multiview augmentation and inference fusion," *Engineering Applications of Artificial Intelligence*, vol. 76, no. 3, pp. 202–213, 2018.

[20] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall *et al.,* "Convolutional neural networks for medical image analysis: Full training or fine tuning?" *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1299–1312, 2016.

[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," ArXiv Preprint ArXiv: 1409. 1556, 2014.

[22] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Las Vegas Nevada, pp. 770–778, 2016.

[23] A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 60, no. 6, pp. 84–90, 2012.

[24] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Las Vegas Nevada, pp. 2818–2826, 2016.

[25] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed *et al.,* "Going deeper with convolutions," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Boston, Massachusetts, pp. 1–9, 2015.

[26] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Int. Conf. on Machine Learning*, Long Beach, California, USA, PMLR, pp. 6105–6114, 2019.

[27] F. Pedersoli, S. Benini, N. Adami and R. Leonardi, "XKin: An open source framework for hand pose and gesture recognition using kinect," *The Visual Computer*, vol. 30, no. 10, pp. 1107–1122, 2014.

[28] C. -L. Huang and S. -H. Jeng, "A model-based hand gesture recognition system," *Machine Vision and Applications*, vol. 12, no. 5, pp. 243–258, 2001.

[29] S. Skaria, A. Al-Hourani and R. J. Evans, "Deep-learning methods for hand-gesture recognition using ultra-wideband radar," *IEEE Access*, vol. 8, pp. 203580–203590, 2020.

[30] A. Haria, A. Subramanian, N. Asokkumar and S. Poddar, "Hand gesture recognition for human computer interaction," *Procedia Computer Science*, vol. 115, pp. 367–374, 2017.

[31] P. Parvathy, K. Subramaniam, G. K. D. Venkatesan, P. Karthikaikumar, J. Varghese *et al.,* "Retraction note to: Development of hand gesture recognition system using machine learning," *Journal of Ambient Intelligence and Humanized Computing*, Springer, vol. 12, pp. 6793–6800, 2021.

[32] Q. De Smedt, H. Wannous, J. P. Vandeborre, J. Guerry, B. L. Saux *et al.,* "3D hand gesture recognition using a depth and skeletal dataset: Shrec'17 track," in *Proc. of the Workshop on 3D Object Retrieval*, Goslar, Germany, pp. 33–38, 2017.

[33] N. Neverova, C. Wolf, F. Nebout and G. W. Taylor, "Hand pose estimation through semi-supervised and weakly-supervised learning," *Computer Vision and Image Understanding*, vol. 164, pp. 56–67, 2017.

[34] N. Neverova, C. Wolf, G. Taylor and F. Nebout, "Moddrop: Adaptive multi-modal gesture recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 8, pp. 1692–1706, 2015.

[35] A. Mujahid, M. J. Awan, A. Yasin, M. A. Mohammed, R. Damasevicius *et al.,* "Real-time hand gesture recognition based on deep learning YOLOv3 model," *Applied Sciences*, vol. 11, no. 9, pp. 4164, 2021.

[36] T. Ozcan and A. Basturk, "Transfer learning-based convolutional neural networks with heuristic optimization for hand gesture recognition," *Neural Computing and Applications*, vol. 31, no. 12, pp. 8955–8970, 2019.

[37] J. P. Sahoo, A. J. Prakash, P. Pławiak and S. Samantray, "Real-time hand gesture recognition using fine-tuned convolutional neural network," *Sensors*, vol. 22, no. 3, pp. 706, 2022.

[38] P. Neethu, R. Suguna and D. Sathish, "An efficient method for human hand gesture detection and recognition using deep learning convolutional neural networks," *Soft Computing*, vol. 24, no. 20, pp. 15239–15248, 2020.

[39] O. K. Oyedotun and A. Khashman, "Deep learning in vision-based static hand gesture recognition," *Neural Computing and Applications*, vol. 28, no. 12, pp. 3941–3951, 2017.

[40] O. Güler and İ. Yücedağ, "Hand gesture recognition from 2D images by using convolutional capsule neural networks," *Arabian Journal for Science and Engineering*, vol. 47, no. 2, pp. 1211–1225, 2022.

[41] K. M. Sagayam, A. D. Andrushia, A. Ghosh, O. Deperlioglu, A. A. Elngar *et al.,* "Recognition of hand gesture image using deep convolutional neural network," *International Journal of Image and Graphics*, vol. 22, no. 3, pp. 2140008, 2022.

[42] B. Feng, F. He, X. Wang, Y. Wu, H. Wang *et al.,* "Depth-projection-map-based bag of contour fragments for robust hand gesture recognition," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 4, pp. 511–523, 2016.

[43] Y. LeCun, Y. Bengio and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[44] J. Ding, B. Chen, H. Liu and M. Huang, "Convolutional neural network with data augmentation for SAR target recognition," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 3, pp. 364–368, 2016.

[45] A. Hussain, M. Imad, A. Khan and B. Ullah, Multi-class classification for the Identification of COVID-19 in X-ray images using customized efficient neural network. in *AI and IoT for Sustainable Development in Emerging Countries*, 1st ed., vol. 105. Switzerland: Springer, pp. 473–486, 2022.

[46] M. B. Nejad and M. E. Shiri, "A new enhanced learning approach to automatic image classification based on Salp Swarm Algorithm," *Computer System Science and Engineering*, vol. 34, no. 2, pp. 91–100, 2019.

[47] S. X. Hu, S. Zagoruyko and N. Komodakis, Exploring weight symmetry in deep neural networks. in *Computer Vision and Image Understanding*, Vol. 187. Amsterdam, Netherland: Elsevier, pp. 102786, 2019.

[48] P. Sharma and R. S. Anand, "Depth data and fusion of feature descriptors for static gesture recognition," *IET Image Processing*, vol. 14, no. 5, pp. 909–920, 2020.

[49] L. Guo, Z. Lu and L. Yao, "Human-machine interaction sensing technology based on hand gesture recognition: A review," *IEEE Transactions on Human-Machine Systems*, vol. 51, no. 4, pp. 300– 309, 2021.