Applied Computer Science, vol. 20, no. 2, pp. 1–23
doi: 10.35784/acs-2024-13

Submitted: 2024-01-10 | Revised: 2024-05-02 | Accepted: 2024-05-06

Keywords: CNN segmentation, few-shot learning, hand gesture, disabled people

*Mohamed ELBAHRI* [0000-0001-5361-1567]*, *Nasreddine TALEB* [0000-0002-9688-3834]**,
*Sid Ahmed El Mehdi ARDJOUN* [0000-0002-1258-3907]***,
*Chakib Mustapha Anouar ZOUAOUI* [0000-0001-5176-4623]**

# FEW-SHOT LEARNING WITH PRE-TRAINED LAYERS INTEGRATION APPLIED TO HAND GESTURE RECOGNITION FOR PEOPLE WITH DISABILITIES

**Abstract**

*Employing vision-based hand gesture recognition for the interaction and communication of disabled individuals is highly beneficial. The hands and gestures of this category of people have a distinctive aspect, requiring the adaptation of a deep learning vision-based system with a dedicated dataset for each individual. To achieve this objective, the paper presents a novel approach for training gesture classification using few-shot samples. More specifically, the gesture classifiers are fine-tuned segments of a pre-trained deep network. The global framework consists of two modules. The first one is a base feature learner and a hand detector trained with normal people hand's images; this module results in a hand detector ad hoc model. The second module is a learner sub-classifier; it is the leverage of the convolution layers of the hand detector feature extractor. It builds a shallow CNN trained with few-shot samples for gesture classification. The proposed approach enables the reuse of segments of a pre-trained feature extractor to build a new sub-classification model. The results obtained by varying the size of the training dataset have demonstrated the efficiency of our method compared to the ones of the literature.*

## 1. INTRODUCTION

Artificial Intelligence (AI) allows computers to do tasks such as decision-making, challenge-solving, scene understanding, and human speech comprehension in any language. Most AI applications aim to automate a process or replace human tasks; this is seen in factories, hospitals and even on the roads; meeting an autonomous car is no longer surprising.

* Djillali Liabes University, Computer Science Department, Communication Networks, Architecture and Multimedia Laboratory, Algeria, mohamed.elbahri@univ-sba.dz
** Djillali Liabes University, Electronics Department, Communication Networks, Architecture and Multimedia Laboratory, Algeria, ne_taleb@univ-sba.dz, chakib@ipatdz.info
*** Djillali Liabes University, IRECOM Laboratory, Algeria, elmehdi.ardjoun@univ-sba.dz

However, a subset of the population has a genuine need for AI. Blind people or people with other disabilities have a difficulty executing essential human functions and daily living activities. Some of them cannot even express their needs, and keeping an eye on them all the time is a difficult task that requires hiring support and care staff. Suppose they can make a simple gesture with their hands or move one or many of their fingers. In that case, one can exploit these gestures to allow them to express themselves, call the nurse, move their beds, turn on/off the TV, control any equipment, and help them interact with intelligent systems and robots. It is well known that a massive amount of data is required for a good design of an artificial intelligence system. This constraint reduces the effectiveness of deep learning in domains where information is not widely available, such as diagnosing rare diseases where few medical images exist. The same case applies to the physiology of disabled individuals. The main objective of this study is to be able to adapt an intelligent system with limited data for interpreting the gestures of people with disabilities with distinctive aspects. The gesture recognition process requires an examination of the hand image once located within the observed scene. The literature survey described in (Bandini & Zariffa, 2020) thoroughly investigates the previous works in this field. The authors highlighted the latest taxonomy of hand signs computer vision methods and they have framed this problem into three macro-areas: localization, interpretation and application. The hand localization process consists of the identification, detection, tracking and estimation of the hand and the fingers in the observed scene; it also includes pose estimation and segmentation. The interpretation encloses the approaches of activity recognition, action/interaction recognition, gesture recognition and hand grasp analysis. Adapting deep learning based Object Detectors (OD) like Faster Region Based Convolutional Neural Network (Faster R-CNN) (Ren et al., 2015), You Only Look Once (YOLO) (Redmon et al., 2016), and Single Shot Detection (SSD) (Liu et al., 2016) for hand gestures recognition is very useful for both localization and gesture recognition by classification, where each class refers to a particular gesture or sign. However, training an OD requires a large adequate dataset. In our application, subjects have reduced hand functionality which inevitably induces that each case of hand has a specific form and specific characteristics. Object detection is a challenging task when applied in real-world circumstances involving new objects not seen in popular object detection datasets. To overcome this limitation, some works in Few-shot-learning have been undertaken, but in their best results, the mean average precision metric (Henderson & Ferrari, 2017; Tang et al., 2022) did not exceed 60% (Vu et al., 2022). The accuracy of an autonomous assistance system should be maximized, and a wrong decision might have devastating consequences for such vulnerable persons. Based on this background and in order to enhance such accuracy, the authors propose a combined system of hand localization and gesture recognition trained in two phases, which can be easily adapted for special cases. The first module involves localizing the hand with a faster R-CNN as an object detector. It is trained and fine-tuned on large hand datasets. This part has two objectives. The first one is accurate hand localization, while the second is extracting standard features based on the convolution layers of its backbone. The second module of the proposed architecture is a shallow convolutional neural network. Its convolutional layer is a segment of the first model's backbone fine-tuned during the second training phase. The main idea is that the first module is considered as a base model trained on a base dataset of common hand images for a global task, while the second one is considered as a new target domain for new tasks trained with a few-shot samples dataset of specific hand gestures.

While partial convolutional layers have recently emerged as an effective way to handle incomplete or corrupted images (Liu et al., 2023), their focused application has been limited to reconstructive photo tasks like inpainting. Partial convolutions selectively utilise available valid pixels while disregarding missing sections of images. This technique has yet to be extended to high-level computer vision problems within real domains. Our work looks at utilizing concepts of partial convolutions – namely selective computation and dynamic renormalization – for the distinct purpose of gesture recognition across people with diverse accessibility needs. By repurposing these methods in novel architectural combinations and directing them towards specialized skeletal representation inputs, we demonstrate the extensive untapped potential of partialized CNNs for efficiency gains in classification problems unrelated to image reconstruction.

The results obtained demonstrate that our approach performs well on both localization and gesture recognition for any type of case with few-shot of subject images, which leads to an easy adaptation for real-world applications.

The following is an overview of our contributions:

- Our work is one of the few ones to investigate the issue of disabled people's hand gesture recognition.
- We demonstrate that it is possible to use segments of a pre-trained model to construct another CNN.
- We offer the possibility of adaptation and training from few support samples to be very easily implemented for specific tasks, and this is through the technique of fusion and transfer learning between the two proposed modules forming the final frozen model ready for deployment without decreasing the hand detector accuracy.
- Through experiments, we demonstrate that the proposed approach consistently outperforms baseline techniques, especially when the number of samples is quite small. Our classifier adjusts to new gesture classifications much more quickly.

## 2. RELATED WORKS

This article proposes an approach that offers the possibility to adapt, with few shot samples, a hand gesture classifier for people with disabilities, which needs to address the process of object detection and image classification. Therefore, the authors rely on relevant works in automatic and intelligent assistance for people with disabilities, hand detection and hand gesture recognition.

In the work presented in Utaminingrum et al. (2017), the authors integrated a computer vision-based obstacle detection system into a wheelchair to ensure safe driving. In Yang et al. (2019), a personal care robot was employed to assist patients, where communication is based on a visual bridge through sign language of lips, eye, head and hand.

American Sign Language (ASL) expression allowed people who cannot speak to express their thoughts. Inspired by this language, Chattoraj et al. (2017) have proposed a method based on the SIFT (Lowe, 2004) algorithm for features extraction to recognize hand gestures. Dradas & Georganas (2011) have fed an SVM classifier with the same features used in (Chattoraj et al, 2017) to recognize gestures. Panwar et al. (2012) have used a classical image segmentation to process images for hand gesture recognition; the proposed method is based on image shape. Zhao et al. (2023) proposed a hand tremor as a motor symptom to diagnose

and assess the Parkinson's disease. In (Fathi et al., 2011; Y. Li et al., 2015; Pirsiavash & Ramanan, 2012), the authors have based their works on the egocentrics feature extraction for hand gesture classification (Fang et al., 2020; Huiwei et al., 2017). Other works (Hung et al., 2015, 2016; Likitlersuang et al., 2019) have added sensors-based technologies to perceive gestures in space and are not limited to visual information; they detect, estimate and classify any hand movements with gloves, gyroscopes and hand belts. Wearing coloured gloves for hand segmentation is an easy solution to be implemented. This technique reached good results for hand detection in (Ishiyama & Kurabayashi, 2016; Liang et al., 2015). Hand detection had been dealt with previously with Skin colour-based approaches, Depth-based approaches and Hand-crafted features. Deep convolutional networks have demonstrated great effectiveness in solving challenging problems in the field of computer vision, while many researchers have turned their attention to this field for the particular problem of gesture recognition. Mohammed et al. (2019) have proposed a cascade of convolutional neural networks (CNNs): a one-stage CNN for object detection and a second CNN for gesture recognition. Xu et al. (2020) have used GAN ( Generative Adversarial Network) to reconstruct occluded hand parts after features extraction with CNN. The problem of partial hand occlusion was well solved in Sahoo et al. (2023). A work for both hand detection and rotation estimation has been proposed by Deng et al. (2018); they have used Faster R-CNN (Ren et al., 2015) to delimit region proposals in images containing hands and to extract local features; after that, the problem of rotation estimation has been solved using CNN. The faster R-CNN has also been employed by Srividya et al. (2019) for hand detection. As mentioned above, many CNN-based works have been proposed; however, CNN models for either image of gestures classification (Bao et al., 2017; Zhang et al., 2018) or object detectors (Liu et al., 2016; Redmon et al., 2016; Ren et al., 2015) need large image datasets exceeding thousands of images. As we employed a partial network for minimal data training, the approach introduced by the authors in Li et al. (2023) involves the Knowledge-Guided Semantic Transfer Network. This model integrates vision-based feature learning, knowledge transfer, and classifier learning within a unified framework to achieve optimal compatibility. In the hand recognition field, many researchers have spent a lot of time collecting and processing (annotating) thousands of images. Hsiao et al. (2014) have collected a dataset of around 240000 tuple images (colour and depth images), with a mask of the hand region. In Bao et al. (2017), the authors have proposed a large dataset for hand detection. The images were acquired in two environments: simple and complex backgrounds. The EgoGesture (first-person view) has focused on gesture recognition for human–computer interaction; here some works have employed a special technique consisting of a RealSense camera mounted on the head (Zhang et al., 2018).

## 3. PROPOSED METHOD

All object detection techniques are composed of many processing parts (features extractor, region proposal network (RPN), deep features extractor, bounding boxes regressor), and each object detection technique has its own set of processes. The common part is the features extractor, called the backbone. It consists of several convolution blocks, each of them with a certain set of filters. The backbone can be customized on the condition of proving its efficiency, while in general, it is a reference model such as VGG16 (Simonyan

& Zisserman, 2015), ResNet (Szegedy et al., 2015), Inception (Szegedy et al., 2017) and their later versions. An overview presented in (El Moataz, 2020) summarizes the accuracy results of combining each OD method with a particular backbone. The main contribution of this work is to select a specific convolution block of the OD backbone to construct another classifier network's backbone. OD outputs are tensors containing bounding boxes, classes, and scores. When the score of a detected object is greater than a certain threshold, a sparse set of cropped images is fed to the second module. Fig. 1 illustrates the global scheme of the proposed method, where we can observe that there are two modules (red and green bounding boxes), and each one represents a specific task. The first module task consists of training and producing a hand detector that is universal for all human hands. The second module task generates a CNN classifier for a specific instance.

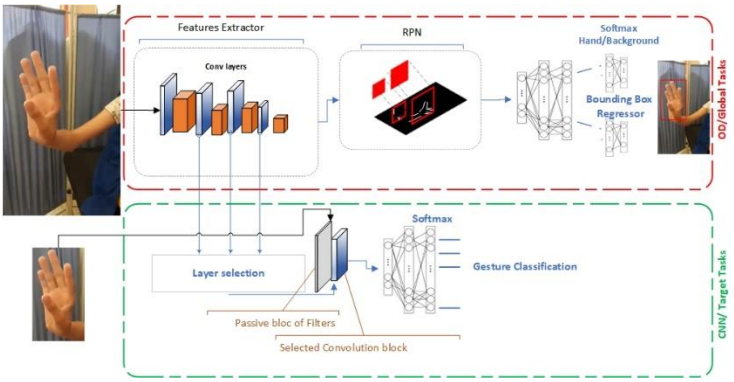


**Fig. 1. Global framework representation**

### 3.1. Problem formulation

Before explaining the overall system, we should first define some formal notions as follows:

Let $D$ denote the global images dataset used for training the final fused model. Since we propose two modules to train two models in two phases with two different datasets (the first one is large and the second one is small), we denote by $D_{base}$ a large dataset of images of hands, and by $D_{novel}$ a small dataset representing some gestures. Therefore, the global dataset can be expressed as $D = D_{base} \cup D_{novel}$ and $D_{base} \cap D_{novel} = \emptyset$. In addition, since we address the gestures classification problem in this work, we consider the "hand" object as a global class wrapping many shapes and gestures of hands that we denote $C_{global}$, while we consider gestures class $C_{novel} = \{C_{novel}^1, C_{novel}^2, \dots, C_{novel}^J\}$ as a set of $J$ sub-classes, where $J$ represents the number of gestures to classify. For the first module, we consider that we have only one global task denoted $T$. $T$ is a training task with dataset $D_{base}$ belonging to class $C_{global}$. $T$ results in a trained base model $M_{base}$ for the "hand" object detection in the query image.

The novel tasks denoted by $T_j'$ in the second module, where $j \in \{1,2,\dots,J\}$, is for training $M_{novel}$ with dataset $D_{novel}$.

### 3.2. OD training scheme

In this step, an object detector is trained with a large dataset. Existing techniques have not been improved here, but the authors have ensured that the model is fed with a good dataset containing thousands of images where the persons' hands are annotated with multiple scene contexts. This dataset is the fusion of two benchmark sets(Nuzzi et al., 2021; Zheng et al., 2018) and this is to adapt the model to detect hands with several angles of view, scales and shapes. The OD features extractor is fine-tuned, robust and ready to provide useful patterns for both detection and classification.

The OD model is trained following the next steps:

1. Defining the global architecture for the initial model.
2. Training model $M_{initial}$ with a universal dataset like the one presented in (Everingham et al., 2010).
3. Training $M_{initial}$ with $D_{base}$.

Process (c) represents $T$ and generates base model $M_{base}$ for OD.

### 3.3. Partial network-based gestures classification

The gesture classifier module is a shallow CNN consisting of two convolutional blocks and a fully connected layer. The first convolutional layer is a passive block, while the second layer partially reuses a block from the OD backbone network. This section explains how to enable the use of such a partial block. Notably, the proposed partial block concept differs from the typical technique of using an intermediate layer output, where the input would still come from the original network's input layer. In the presented approach, only a part of the original OD network is extracted and reused as a block in our gesture classifier. The authors formulate the passive block in a way that shapes its output to fit the expected input shape of the subsequent layers (The partial segmented block) to create our classifier CNN.

Mathematically, a convolution of an image with a set of blocks consists of multiple computations.

$$F = Conv(I^{w\times h\times depth}; b_0, b_1, \dots . b_N) \tag{1}$$

Where $F$ is the resulting features map, $I$ is the input image with three channels and $b_i$ is the $i^{th}$convolution block. Equation (1) can be rewritten with more details as follows:

$$F = Conv_N(Conv_{N-1}\big(Conv_{N-2} \dots . \big(Conv_0(I^{w\times h\times depth}; b_0\big) \dots . ; b_{N-1}\big); b_N \tag{2}$$

Where $0\ and\ N$ denote respectively the first and the last block indices.

The output of any CNN backbone is a result of the recursive call of the convolution function $Conv$. According to Equation (2), $Conv_N$ convolves the result of $Conv_{N-1}$ with block $b_N$'s filters and so on. The first executed item is $Conv_0$ and has as parameters the input image $I^{w\times h\times depht}$ and $b_0$. Fig. 2.A represents an example of a CNN feature extractor. Each block $n$ consists of filters $f_i^n = k^{width\times height\times depth}$ where $i \in \{1, \dots, Nf\}$ and $Nf$ denotes the number of filters of this block. The shape of the tensor coding any convolution block is $(f_{width}, f_{height}, f_{depth}, Nf)$, where $f_{width}, f_{height}$ and $f_{depth}$ refer to the filter's kernel width, height, and depth respectively. All filters $f^n$ of the same block $n$ share the

same kernel shape $k$. Block 0 in Fig. 2.A has $Nf = 4$ (i.e., four filters) and generates four intermediate feature maps. Its shape parameters $f_{width}, f_{height}, Nf$ are hyperparameters fixed without any restriction.

However, there is a mathematical constraint about the convolution of tensor $T^{width \times height \times depht}$ with block $b_n^{f_{width} \times f_{height} \times f_{depth} \times Nf}$. $Conv(T^{width \times height \times depth}; b_n^{f_{width} \times f_{height} \times f_{depth} \times Nf})$ can only be computed if $T^{depht} = b_n^{f_{depth}}$, i.e., the depth of filter $b_n^{f_{depth}}$ of the $n^{th}$ block must match the number of filters $Nf$ of block $b_{n-1}$. It is to note that, according to Fig. 2.A, each filter of block 2 has a depth equal to 5 which represents the number of filters $Nf$ of the previous block; the same rule is applicable for any CNN.

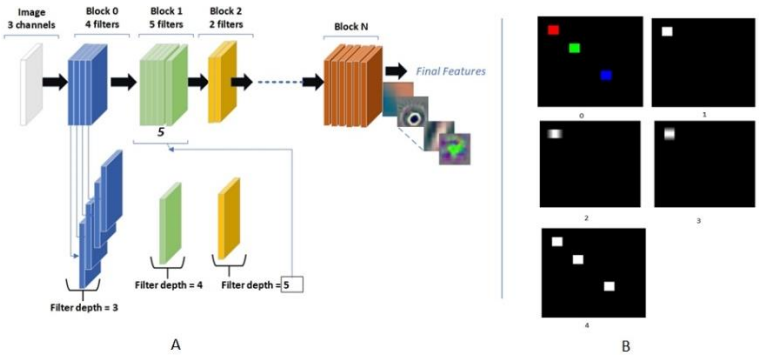


**Fig. 2. (A) represents an example of convolution blocks and their filters. (B) represents a simple RGB image (B.0) and its convolved results (B 1,2,3,4) with customized filters**

Unlike transfer learning techniques (Weiss et al., 2016), where the whole backbone is used, it is proposed to select the best block generating the best model. The proposed backbone architecture of a composed model $CM$ consists of a passive block and a selected block $b_i$ within backbone $M_{base}$ described in section 3.2. A passive block of filters, $P^{width \times height \times depth \times Nf}$ does not highlight any features (color or shape) before the training process. Its kernel is 0 except its center which is equal to 1. Figures Fig. 2.B. 1, 2, 3 and 4 result from the convolution of the RGB image (Fig. 2.B.0) with four kernels $(K_1, K_2, K_3, K_4)$, respectively, whose compositions are as follows:

$$K_1^{3\times3\times3} = \begin{bmatrix}0&0&0\\0&\mathbf{1}&0\\0&0&0\end{bmatrix}\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\end{bmatrix}\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\end{bmatrix}$$

$$K_2^{3\times3\times3} = \begin{bmatrix}0&0&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}\\0&0&0\end{bmatrix}\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\end{bmatrix}\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\end{bmatrix}$$

$$K_3^{3\times3\times3} = \begin{bmatrix}0&\mathbf{1}&0\\0&\mathbf{1}&0\\0&\mathbf{1}&0\end{bmatrix}\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\end{bmatrix}\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\end{bmatrix}$$

$$K_4^{3\times3\times3} = \begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$$

Each kernel extracts a particular characteristic. $K_1$ assigns importance to red color only, while $K_2$ and $K_3$ give importance to red color, horizontal and vertical edges. Unlike previous kernels, $K_4$ is passive and no special features are focused. This technique is similar to the one introduced by He et al. (2016) for residual network conception where they used $(1 \times 1)$ filters to increase or decrease the number of generated feature maps. In He et al. (2016), they randomly initialize this category of filters like the others, while in our proposed method we specify them as $K_4$ with zero biases.

A passive block $P$ will precede the chosen segment for two reasons: The first one is to ensure the constrained convolution, without it, only the first selected segment can be convolved with input image $I^{w\times h\times 3}$ since its depth is equal to $b_0^{f_{depth}}$. Therefore, to build all possible models with segments of $M_{base}$, each block $b_i$ is preceded by a customized passive block $P^{width\times height\times depth\times Nf}$ where $P^{depth} = I^{depth} = 3$ and $P^{Nf} = b_i^{f_{depth}}$. The second reason is to offer the model the possibility to fit the new task since training and tuning $P$ parameters will allow learning more characteristics. Hence, the feature maps $F$ generated by the $i^{th}$ $CM$ backbone is:

$$F^i = Conv_1(Conv_0(I^{w\times h\times depth}; P^{width\times height\times depth\times Nf}); b_i) \quad (3)$$

Finally, we obtain $N$ composed models $CM$ with different backbones and all are followed by fully connected *SoftMax* layers. The following algorithm summarizes the steps to create all possible composed models; it trains them and selects the best one for the focused task.

**Algorithm 1** Base model segmentation/ Composed model creation / Training all models/Best model selection

**Input:**

Trained base model $M_{base}$ with $D_{base}$
$C_{novel}$ and $D_{novel}$

***Base model segmentation*** **(Stage 1)**

1: Load and open $M_{base}$
2: Initialize an empty set: $Set_{of\ blocks} \leftarrow \emptyset$
3: **for** *layer* in $M_{base}.layers$ **do**:
4: **if** the *layer* is 'conv_layer' // Exclude all layers except convolution
5: $Set_{of\ blocks} \leftarrow Set_{of\ blocks} \cup layer$

***Composed model creation*** **(Stage 2)**

1: Initialize an empty set: $CM \leftarrow \emptyset$
2: Initialize $width \leftarrow height \leftarrow hyper\_param^1$
3: Initialize $depth \leftarrow I^{depht}$
4: **for** $b$ in $Set_{of\ blocks}$ **do**
5: $Nf \leftarrow b^{depth}$
6: $create\_tensor(P^{width\times height\times depth\times Nf})$
7: Initialize all elements of $P \leftarrow 0$
8: **for** $x$ in range ( $P^{depth}$)
9: **for** $y$ in range ( $P^{Nf}$)
10: $P[\left\lceil\frac{width}{2}\right\rceil, \left\lceil\frac{height}{2}\right\rceil x, y] \leftarrow 1$
11: $cm \leftarrow build\_model(P, b, Fully_{connected}, Softmax^{C_{novel}})$

12: $CM \leftarrow CM \cup cm$

***Training all models*** **(Stage 3)**

1: Initialize epoch ← $hyper\_param^2$
2: Initialize $histories \leftarrow \emptyset$
3: **for** $cm$ in $CM$ **do**
4: $history \leftarrow$ train $(cm, C_{novel}, D_{novel}, epoch)$
5: $histories \leftarrow histories \cup history$

***Best model selection*** **(Stage 4)**

1: Select a $metric \leftarrow Accuracy$
2: $indice\_best\_model \leftarrow argmax(hisories^{metric})$
3:$best\_model \leftarrow CM_{indice\_best\_model}$

Algorithm 1 consists of four stages. The first one browses all $M_{base}$ layers, seeks for convolution blocks and inserts them into set $Set_{of\ blocks}$. The second stage builds $N$ models where $N = Card(Set_{of\ blocks})$. All models share the same global architecture: $P \rightarrow$ Max_pooling $\rightarrow Conv\_block1 \rightarrow$ Max_pooling $\rightarrow Full\_Connected \rightarrow Softmax^{C_{novel}}$, but have different initial weights and shapes of their convolution blocks. The authors mention that $Conv\_block1$ layer retains the same activation function. The third stage trains all models with $D_{novel}$ and saves the histories containing their respective losses and accuracies to keep track of their performances. At the fourth stage, all models are trained, and the best one providing the best accuracy is selected. The accuracy of each model is computed on the test dataset after training according to formula 4.

$$Acc = \frac{TP+TN}{TP+FN+FP+TN} \quad (4)$$

Where TP, TN, FP and FN denote True positive, True negative, False positive and False negative respectively.

## 4. EXPERIMENTS

In this section, the implementation details are first introduced, the used datasets are explored. Subsequently, the authors briefly introduce the object detector fine-tuned for a global task. The results obtained were then presented with a limited dataset and compared with recent works. In the ablation studies presented in this section, the authors investigate the effectiveness, level, and sizes of the generated CNNs. The effect of different training sizes on the accuracy of the generated models was then analyzed, providing a comprehensive comparison with other work. Detailed comparisons in terms of accuracy and inference time with deep networks are also presented.

### 4.1. Experimental setup

Experiments were conducted using the following hardware setup: Intel Core i7 with 64 GB RAM and NVIDIA 3090 RTX GPU. The models are implemented using the TensorFlow framework, along with the object detector that was also fine-tuned using the same framework. This enables the possibility of future fusion between the models for further

enhancements or integration. For the experiments on the MU-ASL-Digit and MU-ASL-Alphabet benchmarks (Barczak et al., 2011), data augmentation was not employed. Data augmentation was only used for other experiments, with a factor of 4. The Adam optimizer was used, and its effectiveness was observed in these experiments. For each training, the batch size is set to 4 and the learning rate was initially set to 0.001 and decreased by 10% every 50 epochs. The models were trained for 200 epochs, and callbacks were integrated to save the best version based on accuracy on the validation data.

### 4.2. Datasets

Many datasets have been used in this work for both training and comparison. As mentioned in section 3.2, two merged datasets have been employed for the base model training steps. In the Oxford dataset (Mittal et al., 2011) (Fig. 3. A),13050 hand instances are annotated; before training, we have added a script to generate new bounding boxes with aligned axes. We have also added 15000 hand instances of 48 subjects obtained from the dataset realized by Bambach et al. (2015) (Fig. 3. B). To evaluate our gesture classifier, a recent benchmark has been used (Nuzzi et al., 2021), it consists of 2400 images per subject. Five subjects performed some gestures with their hands (Fig. 3. C). We have performed many experimental comparisons in terms of accuracy, precision and time of inference on two challenging benchmarks for static hand gesture sign language (SL) recognition: the first one is the Massey University (MU) ASL dataset (Barczak et al., 2011) composed of 1820 images of 26 gestures for letters (‘a’to ‘z’) referred as MU-ASL-Alphabet and 700 images of 10 other gestures for digits (‘0’ to ‘9’) denoted as MU-ASL-Digit. The second challenging SL benchmark with light variations is ASL finger spelling (Pugeault & Bowden, 2011), known in the literature as (ASL-FS-Color). This dataset contains more than 60000 images of 24 gestures. To experiment with our proposed approach in the complicated scenes, we have compared our results with the baseline architecture on the HaGrid dataset (Kapitanov et al., 2022) composed of 552992 samples representing 18 gestures.

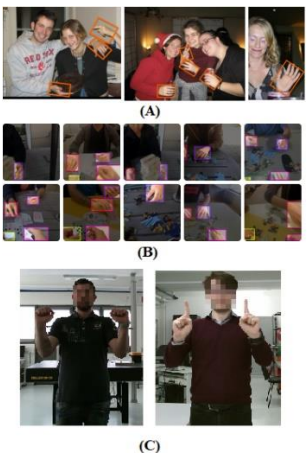


**Fig. 3. Examples of instances of some datasets examples** (Bambach et al., 2015; Kapitanov et al., 2022; Mittal et al., 2011)

In addition to these benchmarks of gestures, several sequences of images of people with disabilities were collected to verify the authors' approach to this case study. As shown in Fig. 4, disabled people have special hand shapes and gestures that do not exist in any collected datasets in previous works.

**Fig. 4. Example of disabled people's hand gestures**

### 4.3. Object detector (base model)

In their experiments, the authors chose Faster R-CNN (Ren et al., 2015) as the OD module and Inception V3 as its backbone network. To improve its mean average precision (MAP), the OD was trained with both Oxford (Mittal et al., 2011) and egocentric (Bambach et al., 2015) datasets. Its MAP goes beyond 63% which is a very good score for Faster R-CNN. In the next experiments, we have constructed our classifiers according to Algorithm 1 using segments of this trained backbone.

### 4.4. Gestures classification results with limited data

The authors conducted many experiments to evaluate the effectiveness of their gesture classifier. In the comparison presented in this section, the superior results are highlighted in bold. Since our objective is to reach a good accuracy using few samples, we split the dataset in (Nuzzi et al., 2021) into a 15% partition for training and an 85% partition for the test, while the traditional training approaches use the inverse ratio. When using Algorithm 1, the authors obtained a model able to classify 12 gestures, trained with only 10 images for each class "gesture". They evaluated their model on the validation dataset that consists of 1800 images. According to the confusion matrix presented in Fig. 5, the model accuracy exceeds 85%. Some gestures like ('collab',' Five',' Nine',' Punch') are very well recognized while some like ('Eight') are confused with others. It is to note that the 85% obtained accuracy is an acceptable score since our training is conducted using only a few samples.

To deepen the study, the authors conducted experiments with the MU-ASL-Digit and MU-ASL-Alphabet and compared our results with recent state-of-the-art works. With these manipulations, they also reversed the split ratio between training and validation. The performance of their approach is compared in terms of accuracies with other works in Table 1.

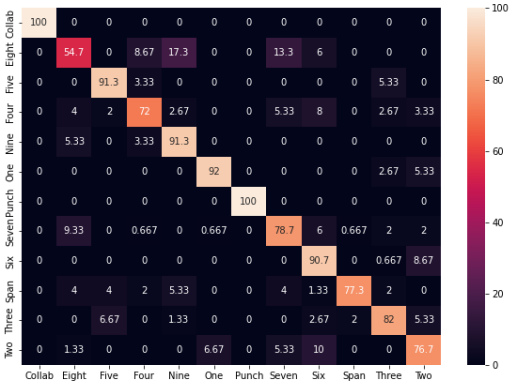


**Fig. 5. Confusion matrix of gesture classification on dataset with our approach (Nuzzi et al., 2021)**

In the MU-ASL-Digit dataset, the proposed model achieves an average accuracy of 97.57% with a (20% train ,80% test) splitting ratio and 99.29% of accuracy with a (80% train, 20% test) splitting ratio. In both splitting data scenarios, our proposed classifier outperforms the techniques Network (Fang et al., 2020; Huiwei et al., 2017; Rahim et al., 2021; Sahoo et al., 2019, 2023) and achieves performance levels close to those in (Damaneh et al., 2023).

**Tab. 1. Results comparison of the hand gesture classification on Mu-ASL-Digit dataset**

| **Methods** | **Accuracy (%)** |
|---|---|
| NNM Factorization and Compressive Sensing (Huiwei et al., 2017) | 87.80 |
| PCA based CNN and SVM (Sahoo et al., 2019) | 95.00 |
| Geometric features and Fisher Vector (FV) (Fang et al., 2020) | 95.30 |
| CNN based on the fusion of features (Rahim et al., 2021) | 96.56 |
| Dual-stream Dense Residual Fusion Network (Sahoo et al., 2023) | 96.14 |
| CNN, ORB descriptor and Gabor filter (Damaneh et al., 2023) | 99.80 |
| The proposed method (20%-80%) | 97.57 |
| The proposed method (80%-20%) | 99.29 |

I In the same context, the authors conducted experiments with the MU-ASL-Alphabet dataset, and the results are summarized in Table 2. Similarly to the previous experiment, the results demonstrate that their method outperforms the other techniques, providing superior outcomes.

**Tab. 2. Results comparison of the hand gesture classification on Mu-ASL-Alphabet dataset**

| **Methods** | **Accuracy (%)** |
|---|---|
| KNN and Gabor filter + DWT (Virender et al., 2018) | 50.83 |
| PCA based CNN and SVM (Sahoo et al., 2019) | 92.60 |
| Feature extractor techniques (Sharma et al., 2020) | 96.96 |
| GFA-Residual Stream (Sahoo et al., 2023) | 93.38 |
| SF-dense Stream (Sahoo et al., 2023) | 92.57 |
| Dual-stream Dense Residual Fusion Network (Sahoo et al., 2023) | 96.14 |
| The proposed method (20%-80%) | 96.07 |
| The proposed method (80%-20%) | 98.35 |

### 4.5. Ablation studies

To gain a deeper insight into the efficacy of the proposed framework for generating optimal CNNs, two distinct ablation studies were condacted. The first focuses on the architecture, sizes, and accuracy of the CNNs, providing insights into the impact of these elements on the overall performance. The second ablation study delves into the training process, specifically examining the influence of varying training data sizes.

#### 4.5.1. Analyzes of generated models

The authors carried out an experiment on a simple dataset for a specific task. The objective is to measure the reliability of each model generated from a segment of $M_{base}$. In this experiment, 22 models are trained after being created by algorithm 1, where the number of filters in the passive block and the segmented block ranges from 64 to 192 each with a (3x3) kernel. Fig. 6 represents all generated models' accuracies and losses. Every $M_{base}$ backbone generates many feature levels. The low-level trait is shown in yellow, the low- and mid-level trait in green, the high- and mid-level trait in blue, and the high-level trait in red.

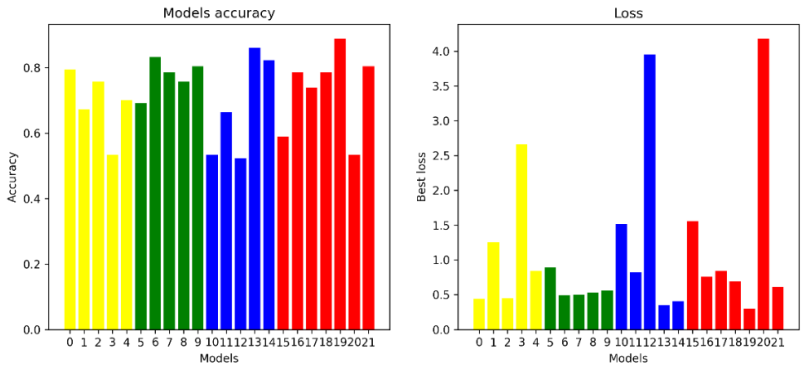


**Fig. 6. Composed models' performances**

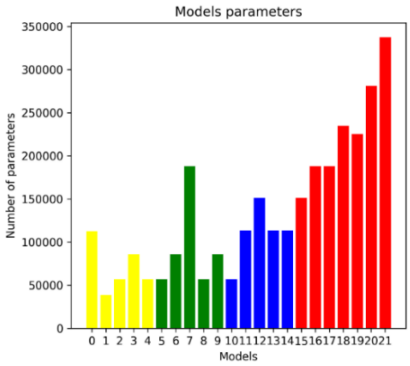


**Fig. 7. Composed models' parameters (Number of parameters of the convolutional layers)**

Hence, models 0 to 4 are built with blocks belonging to the low-level feature extractor of $M_{base}$ and so on. We observe that the accuracy and the loss vary from one model to another.

The $CM_{19}$ accuracy reaches 96% and its loss is less than 0.25 which is very a good result, whereas $CM_{20}$ does not fit the new task even if it belongs to the same level as that of $CM_{19}$. The models' performance variation is due to the feature nature generated by a specific block, some of them are useful for a specific task and others are not.

In the usual training processes, the decision regarding the number of convolutional blocks and the quantity of filters allocated to each block is a crucial hyperparameter determined during the model's design phase. Typically, the augmentation of filter numbers is pursued with the aspiration of enhancing feature extraction, which subsequently contributes to the improved convergence of the CNN. The proposed method revolves around the meticulous selection of apt feature maps within a pre-trained model and their adaptation to a fresh task. This selection process can yield blocks with fewer parameters, resulting in heightened efficiency. Fig. 7 serves as a graphical representation of the parameter counts within the convolutional layers of each model generated. It's important to note that these counts exclude the fully connected layer component. Upon closer examination of Fig. 7, we observe that $CM_{19}$, known for its commendable performance, boasts a more modest parameter count when compared to $CM_{18}$, $CM_{20}$, and $CM_{21}$. This distinction arises from the utilization of feature maps that were specifically tailored to excel in a particular task. In Tang et al. (2022), the authors claim that most existing methods focus more on high-level features. At the same time, they observe that low-level attention maps can capture more subtle parts containing detailed information that has yet to be fully explored. In our proposed work, the selected blocks can also be an intermediate block of the feature map rather than necessarily the one with the highest order or number of filters.

In Fig. 8, the training steps are explored. For greater clarity, the training graphic is divided into four segments, each representing a level (low, low-intermediate, upper-intermediate, and high). Every model is represented by a particular colour. In each segment, we observe that some models have a good fit on the specific task and reach good accuracy scores on validation data, whereas others overfit. It is to note that $CM_{19}$ is not always the best model. In other scenarios where the target task and $M_{base}$ are different, other results shall be obtained, and graphics in figures Fig. 6, Fig. 7 and Fig. 8 will be different, which means that another model has to be selected. The best model generated for a task depends on its ability to extract the right features, and each model is built from a segment of $M_{base}$ whose convolution filters vary. Thus, only an empirical search allows us to select the best model for a specific task.

To demonstrate the proposed approach's efficiency, a standard *CNN* model called *SM* was trained from scratch. The $SM$ architecture is the same as that of $CM_{19}$ , i.e., they both have the same number and dimensions of convolution blocks and the same fully connected layers, however, unlike $CM_{19}$, the $SM$ filter weights are randomly initialized, as done in ordinary training.

To objectively compare $CM_{19}$ and the $SM$ performances, both model fittings are conducted on $D_{novel}$. The models' parameters tunability during training is also evaluated with the same hyperparameters (optimizer, learning rate, batch size, data augmentation, regularization, and number of steps…). Fig. 9 shows the $CM_{19}$ and $SM$ training graphs. After 200 epochs, $CM_{19}$ outperforms $SM$ with a very large difference. Our obtained model accuracy exceeds 80% while the standard model is just over 50%. We also notice in the same figure (Fig.6.A) that the curve of the training graph of $CM_{19}$ is smoother than that of $SM$.

For both $CM_{19}$ and $SM$ graphs, we observe that the validation curve is on the training one, this is due to the dropout regularization which ignores randomly some neurons outputs during training and not in the validation process. Since our novel dataset $D_{novel}$ is small, the dropout is used to reduce overfitting during the training process of $CM_{19}$ and $SM$. Additional regularization methods, as suggested in (Zheng et al., 2018), find application in deep learning to mitigate overfitting. However, they can exert a notable influence on the model's layer weights, potentially diminishing the efficiency of our approach, given its reliance on pre-trained layers.

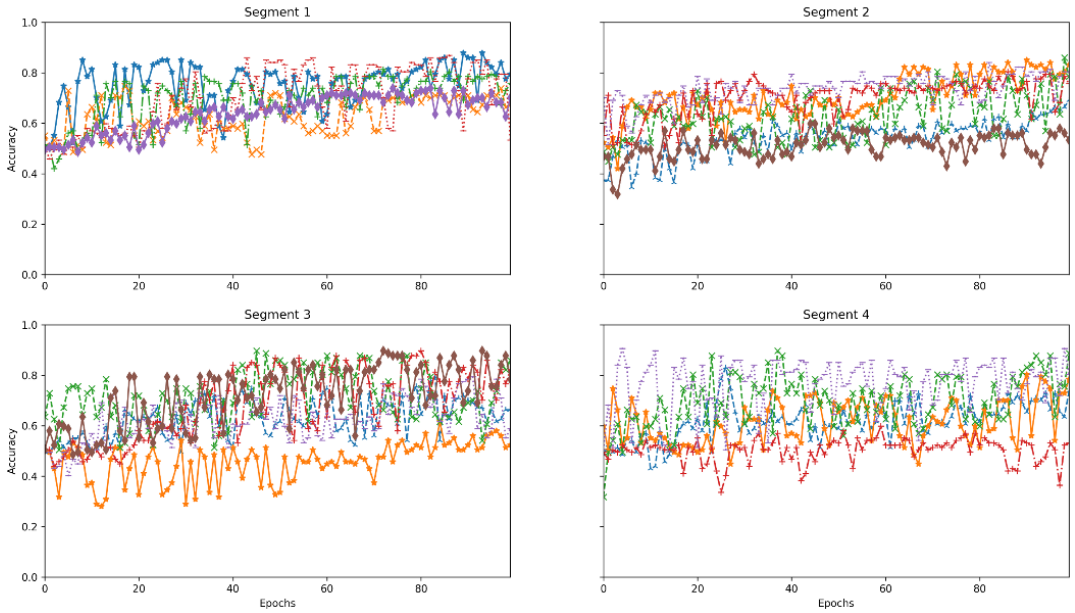


**Fig. 8. Composed Models' training graphs**

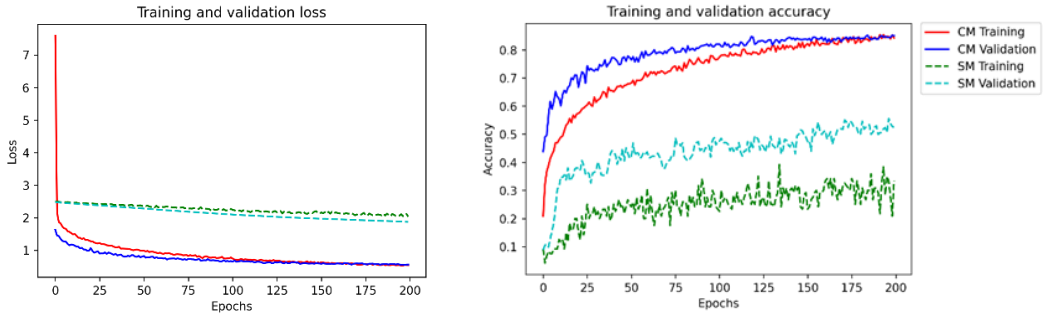


**Fig. 9. Training graphs of $CM_{19}$ and $SM$**

### 4.5.2. The effect of varying the training data size

The size of the training set and the quality of the data significantly impact the learning phase and can lead to overfitting or underfitting of the resulting network. In this section, an

experiment that involves varying the number of images used during training is presented. Fig. 10 shows the performance evolution based on the quantity of provided images. The authors used six variants consisting of 10, 20, 30, 40, 50 images and 80% of a subset from the MU-ASL-COLOR (Pugeault & Bowden, 2011) dataset.

For each variant, models were generated by tuning segments of the pre-trained network with an appropriate number of variant-specific samples. For example, for the first variant, the training process utilizes 10 images per gesture and the last one (80%) utilizes more than 11K images. For each variant, accuracy is evaluated on a validation dataset of 12K images.

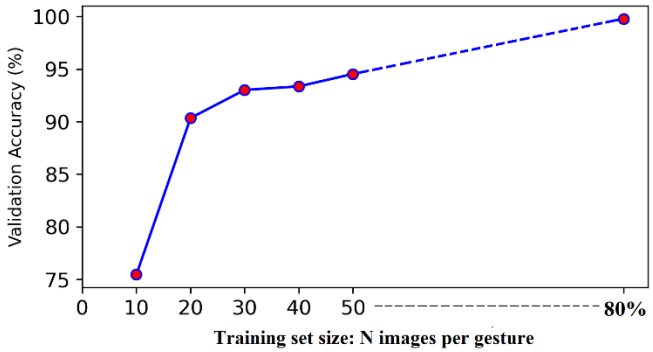


**Fig. 10. Evolution of the performance according to the training dataset size**

When using the ASL-FS-Color benchmark, the CNN generated by the proposed method also achieves good performance, surpassing other techniques mentioned in Table 3. Despite having fewer training samples, the proposed system outperforms the literature works. However, this is only the case when surpassing 30 images per gesture. With variants of 10 and 20 images per gesture, the results are not very conclusive.

**Tab. 3. Results comparison of the hand gesture classification on ASL-FS-Colordataset**

| **Methods** | **Accuracy (%)** |
|---|---|
| Intensive feature extrication (Bhaumik et al., 2022) | 81.72 |
| Residual block intensity feature (Sahoo et al., 2022) | 88.65 |
| GFA-Residual Stream (Sahoo et al., 2023) | 88.45 |
| SF-dense Stream (Sahoo et al., 2023) | 86.95 |
| Dual-stream Dense Residual Fusion Network (Sahoo et al., 2023) | 91.24 |
| The proposed method (10 images) | 75.45 |
| The proposed method (20 images) | 90.35 |
| The proposed method (30 images) | 93.02 |
| The proposed method (50 images) | 94.54 |
| The proposed method (80%-20%) | 99.77 |

We evaluated our approach in terms of precision, along with other metrics, and assessed its inter-class performance. The results were compared with a recent work (Sahoo et al., 2023) that provided good analyses using the MU-ASL-Color benchmark.

An average across 24 gestures of the precision, recall, and F1-score metrics is calculated for both the DeReFNet (Sahoo et al., 2023) classifier and three proposed variants. The best obtained result (See Table 4) surpasses the performance of DeReNet by approximately 2.5% in all three evaluated metrics, once the classifier is trained with 30 images or more. To assess the ability to distinguish hard and similar gestures, we present in Table 5 a comparison of inter-class similar gestures across the MU-ASL-Color dataset. Table 5 represents the accuracy of gestures classification for letters A, E, K, P, and X. The results are good overall, except for letter A, where the accuracy is relatively poor. Bold values represent the best scores, while the underlined ones represent the lowest ones. The obtained results demonstrate that our generated classifier achieves higher accuracy in distinguishing inter-class similar gestures compared to DeReFNet. This tabulated outcome validates the potential of the proposed approach.

**Tab. 4. Results comparison of the hand gesture classification on ASL-FS-Color dataset**

| **Methods** | **Precision** | **Recall** | **F1-Score** |
|---|---|---|---|
| DeReFnet (Sahoo et al., 2023) | 92.24 | 91.88 | 92.06 |
| The proposed method (20) | 90.61 | 90.35 | 90.30 |
| The proposed method (30) | 93.26 | 93.03 | 92.99 |
| The proposed method (50) | 94.80 | 94.53 | 94.53 |

**Tab. 5. Results comparison of hard gesture classification on ASL-FS-Color dataset**

| **Methods** | **A** | **E** | **K** | **P** | **X** |
|---|---|---|---|---|---|
| DeReFnet (Sahoo et al., 2023) | 94.1 | 75.3 | 66.4 | 74.6 | 80.04 |
| The proposed method (20) | 78.3 | 80.04 | 99.4 | 94.3 | 100 |
| The proposed method (30) | 77.9 | 90.2 | 97.6 | 94.3 | 100 |
| The proposed method (50) | 80.0 | 91.4 | 97.8 | 95.2 | 100 |

## 5. COMPARISON WITH DEEP NETWORKS

The conducted work involves constructing a shallow network using segments from any location of a trained deep network. The resulting network is less complex and maintains good performance. To substantiate these claims, experiments on the MU-ASL-Alphabet dataset were conducted. The authors reported some results from (Sahoo et al., 2023). They compared their CNN in terms of the number of parameters, accuracy and inference speed with DeRefNet, as well as CNN architectures ResNet50, VGG16 and DenseNet. According to Table 6, the proposed CNN is the shallowest among the CNN networks, with only 3.35 million parameters, while maintaining good performance on this benchmark dataset. Additionally, it achieves the fastest inference speed compared to the other models.

The proposed method was then compared with deep networks on the Hagrid dataset (Kapitanov et al., 2022). In Table 7, baseline models are trained with 622063 cropped images and tested with 54518. To prove the efficiency of our approach, we trained our models with a tiny dataset (0,7% of the total dataset) composed of 4320 images belonging to 18 gestures.

**Tab. 6. Comparison with CNNs in terms of speed, accuracy, and size. M indicate Million**

| CNN models | Number of parameters | Accuracy (%) | Inference time (ms) |
|---|---|---|---|
| VGG16 (Simonyan & Zisserman, 2015) | 138.35M | 93.71 | 12.40 |
| DenseNet (Huang et al., 2017) | 20.03M | 91.86 | 51.34 |
| Resnet 50 (He et al., 2015) | 25.58M | 92.62 | 14.28 |
| SENet (Hu et al., 2018) | 28.1M | 93.87 | 21.36 |
| DeReFnet (Sahoo et al., 2023) | 5.24M | 95.70 | 09.54 |
| Proposed CNN | 3.35M | 98.35 | 02.3 |

The results with 43888 images were then tested. The best-obtained accuracy on the test dataset is 68%. In their experiments, the authors trained the MobileNetV3 model with the same tiny dataset and under the same context of our model training process for a fair comparison. They found that MobileNetV3 with or without transfer learning does not exceed 6% of accuracy on the same test set.

The obtained low accuracy score of MobileNetV3 has led to deduce that this classification model is close to a random classification process where the chance of getting the right prediction among 18 classes is 1/18=5.5%; this proves that its training process is overfitted while our model fits better.

**Tab. 7. Results comparison of the hand gesture classification on Hagrid dataset**

| Model | Accuracy% |
|---|---|
| ResNet-18 | 98.72 |
| ResNet-152 | 99.11 |
| ResNeXt-50 | 98.99 |
| ResNeXt-101 | 99.28 |
| MobileNetV3 small | 96.78 |
| MobileNetV3 large | 97.88 |
| ViT-B/32 pre trained | 98.49 |
| Proposed method | 68.12 |
| MobileNetV3 small | 05.4 |
| MobileNetV3 small pre trained | 06.2 |

Based on these experiments, it can be concluded that a shallow CNN composed of pre-trained backbone segments can be very efficient and robust when we look for a good fitting with few samples. MobileNetV3 is very deep according to the obtained model which consists of two convolution blocks only, but training the baseline models without a large dataset, decreases considerably their accuracy. However, the authors' approach is not very efficient as a few shot-learning techniques. It does not handle new classes. This work considers hand gestures as subclasses belonging to the global class that is "hand." The presented approach can provide some areas for future research and provide a reference for transfer learning and classification of several shots.

### 5.1. Hand detection and gesture classification

In the hand detection experiments studied, the Faster R-CNN network was trained, as described in section 4.3. The obtained detection as bounding boxes framing the object hand

is without gesture classification. For accurate detection and classification, the deployed model is a sequential fusion of the OD and the classifier. The output of the OD serves as an intermediate step used as input for the classifier. To demonstrate the value of this approach, the authors conducted experiments using two dataset variants differing in terms of size. In Fig. 11.A, it was observed that the MAP and IOU metrics curves are ascending when training is with a large hand dataset. In the present experiments, the authors proceeded to fine-tune a hand detector using a suitable dataset for a person's gesture. It was observed (See Fig 11.B) that its IOU score decreases until it reaches 0,18. This phenomenon is caused by a lack of training data. For this reason, a combination was proposed instead of tuning with several samples.

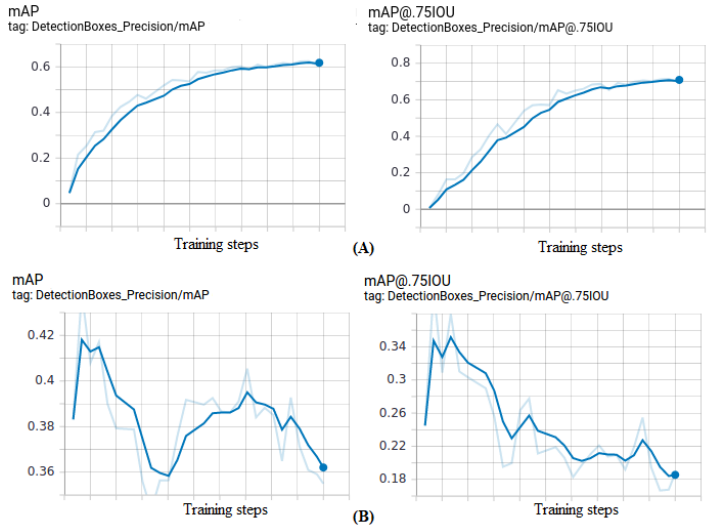


**Fig. 11. Evolution of the OD training with large and small datasets**

The authors tested the proposed combination for both detection and classification on reel subjects. Videos of disabled people are recorded showing some gestures. Fig. 12 represents some visual results. The proposed approach reached an average of 87% of precision and 70% of recall for detection.

**Fig. 12. Visual results of our proposed application to real cases**

## 6. CONCLUSION

This work aims to apply a gesture recognition intelligent system for people with disabilities. The proposed approach consists of a hand detector and a gesture classifier. Only the gesture classifier is specific and adapted to each case, while the hand detector is universal for all patients. Regarding accuracy, the proposed method demonstrates competitive performance even with a training process based on a reduced dataset. By incorporating relevant segments from a pre-trained network, the proposed model captures valuable features and demonstrates accurate predictions comparable to or even surpassing other deep architectures. Moreover, the proposed method shows cases of a smaller model size compared to traditional CNNs. By utilizing selected segments from a larger pre-trained network, the generated CNN effectively reduces the parameters and memory footprint while maintaining comparable or improved performances.

### Conflicts of Interest

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


### Acknowledgment

*The research being reported in this work was supported by the Algerian Directorate General for Scientific Research and Technological Development (DGRSDT).*


#### REFERENCES


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

Bandini, A., & Zariffa, J. (2020). Analysis of the hands in egocentric vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(6), 6846-6866. https://doi.org/10.1109/TPAMI.2020.2986648

Bao, P., Maqueda, A. I., del-Blanco, C. R., & García, N. (2017). Tiny hand gesture recognition without localization via a deep convolutional network. *IEEE Transactions on Consumer Electronics*, *63*(3), 251–257. https://doi.org/10.1109/TCE.2017.014971

Barczak, A. L. C., Reyes, N. H., Abastillas, M., Piccio, A., & Susnjak, T. (2011). A new 2D static hand gesture colour image dataset for ASL gestures. *Research Letters in the Information and Mathematical Sciences*, *15*.

El Moataz, A., Mammass, D., Mansouri, A., & Nouboud, F. (Eds.). (2020). Image and Signal Processing. *9th International Conference (ICISP 2020)*. Springer International Publishing. https://doi.org/10.1007/978-3-030-51935-3

Bhaumik, G., Verma, M., Govil, M. C., & Vipparthi, S. K. (2022). ExtriDeNet: An intensive feature extrication deep network for hand gesture recognition. *The Visual Computer*, *38*(11), 3853–3866. https://doi.org/10.1007/s00371-021-02225-z

Chattoraj, S., Karan, V., Tanmay, P., (2017). Assistive system for physically disabled people using gesture recognition. *2017 IEEE 2nd International Conference on Signal and Image Processing (ICSIP)* (pp. 60–65). IEEE. https://doi.org/10.1109/SIPROCESS.2017.8124506

Damaneh, M. M., Mohanna, F., & Jafari, P. (2023). Static hand gesture recognition in sign language based on convolutional neural network with feature extraction method using ORB descriptor and Gabor filter. *Expert Systems with Applications*, *211*, 118559. https://doi.org/10.1016/j.eswa.2022.118559

Dardas, N. H., & Georganas, N. D. (2011). Real-Time hand gesture detection and recognition using bag-of-features and support vector machine techniques. *IEEE Transactions on Instrumentation and Measurement*, *60*(11), 3592–3607. https://doi.org/10.1109/TIM.2011.2161140

Deng, X., Zhang, Y., Yang, S., Tan, P., Chang, L., Yuan, Y., & Wang, H. (2018). Joint hand detection and rotation estimation using CNN. *IEEE Transactions on Image Processing*, *27*(4), 1888–1900. https://doi.org/10.1109/TIP.2017.2779600

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, *88*, 303–338. https://doi.org/10.1007/s11263-009-0275-4

Fang, L., Liang, N., Kang, W., Wang, Z., & Feng, D. D. (2020). Real-time hand posture recognition using hand geometric features and Fisher Vector. *Signal Processing: Image Communication*, *82*, 115729. https://doi.org/10.1016/j.image.2019.115729

Fathi, A., Farhadi, A., & Rehg, J. M. (2011). Understanding egocentric activities. *2011 International Conference on Computer Vision* (pp. 407–414). IEEE. https://doi.org/10.1109/ICCV.2011.6126269

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778). IEEE. https://doi.org/10.1109/CVPR.2016.90

Henderson, P., & Ferrari, V. (2017). End-to-End Training of Object Class Detectors for Mean Average Precision. In S.-H. Lai, V. Lepetit, K. Nishino, & Y. Sato (Eds.), *Computer Vision – ACCV 2016* (pp. 198–213). Springer International Publishing. https://doi.org/10.1007/978-3-319-54193-8_13

Hsiao, Y.-S., Sanchez-Riera, J., Lim, T., Hua, K.-L., & Cheng, W.-H. (2014). LaRED: A large RGB-D extensible hand gesture dataset. *5th ACM Multimedia Systems Conference* (pp. 53–58). https://doi.org/10.1145/2557642.2563669

Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-Excitation Networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 7132–7141). IEEE. https://doi.org/10.1109/CVPR.2018.00745

Huang, G., Liu, Z., Maaten, L. V. D., & Weinberger, K. Q. (2017). Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2261–2269). IEEE. https://doi.org/10.1109/CVPR.2017.243

Huiwei, Z., Mingqiang, Y., Zhenxing, C., & Qinghe, Z. (2017). A method for static hand gesture recognition based on non-negative matrix factorization and compressive sensing. *IAENG International Journal of Computer Science*, *44*(1), 52–59.

Hung, C.-H., Bai, Y.-W., & Wu, H.-Y. (2015). Home appliance control by a hand gesture recognition belt in LED array lamp case. *2015 IEEE 4th Global Conference on Consumer Electronics (GCCE)* (pp. 599–600). IEEE. https://doi.org/10.1109/GCCE.2015.7398611

Hung, C.-H., Bai, Y.-W., & Wu, H.-Y. (2016). Home outlet and LED array lamp controlled by a smartphone with a hand gesture recognition. *2016 IEEE International Conference on Consumer Electronics (ICCE)* (pp. 5–6). IEEE. https://doi.org/10.1109/ICCE.2016.7430502

Ishiyama, H., & Kurabayashi, S. (2016). Monochrome glove: A robust real-time hand gesture recognition method by using a fabric glove with design of structured markers. *2016 IEEE Virtual Reality (VR)*, 187–188. https://doi.org/10.1109/VR.2016.7504716

Kapitanov, A., Kvanchiani, K., Nagaev, A., Kraynov, R., & Makhlyarchuk, A. (2022). HaGRID - HAnd Gesture recognition image dataset. *ArXiv abs/2206.08219*. https://doi.org/10.48550/arXiv.2206.08219

Li, Y., Ye, Z., & Rehg, J. M. (2015). Delving into egocentric actions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 287–295). IEEE. https://doi.org/10.1109/CVPR.2015.7298625

Li, Z., Tang, H., Peng, Z., Qi, G.-J., & Tang, J. (2023). Knowledge-guided semantic transfer network for few-shot image recognition. *IEEE Transactions on Neural Networks and Learning Systems*, 1–15. https://doi.org/10.1109/TNNLS.2023.3240195

Liang, H., Yuan, J., & Thalman, D. (2015). Egocentric hand pose estimation and distance recovery in a single RGB image. *2015 IEEE International Conference on Multimedia and Expo (ICME)* (pp. 1–6). IEEE. https://doi.org/10.1109/ICME.2015.7177448

Likitlersuang, J., Sumitro, E. R., Cao, T., Visée, R. J., Kalsi-Ryan, S., & Zariffa, J. (2019). Egocentric video: A new tool for capturing hand use of individuals with spinal cord injury at home. *Journal of NeuroEngineering and Rehabilitation*, *16*, 83. https://doi.org/10.1186/s12984-019-0557-1

Liu, G., Dundar, A., Shih, K. J., Wang, T.-C., Reda, F. A., Sapra, K., Yu, Z., Yang, X., Tao, A., & Catanzaro, B. (2023). Partial convolution for padding, inpainting, and image synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(5), 6096–6110. https://doi.org/10.1109/TPAMI.2022.3209702

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single shot MultiBox detector. In B. Leibe, J. Matas, N. Sebe, & M. Welling (Eds.), *Computer Vision – ECCV 2016* (pp. 21–37). Springer International Publishing. https://doi.org/10.1007/978-3-319-46448-0_2

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*, 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94

Mittal, A., Zisserman, A., & Torr, P. (2011). Hand detection using multiple proposals. *Procedings of the British Machine Vision Conference 2011* (pp. 75.1-75.11). https://doi.org/10.5244/C.25.75

Mohammed, A. A. Q., Lv, J., & Islam, M. S. (2019). A Deep Learning-Based End-to-End composite system for hand detection and gesture recognition. *Sensors*, *19*(23), 5282. https://doi.org/10.3390/s19235282

Nuzzi, C., Pasinetti, S., Pagani, R., Coffetti, G., & Sansoni, G. (2021, March 8). HANDS: A dataset of static Hand-Gestures for Human-Robot Interaction. https://doi.org/10.17632/ndrczc35bt.1

Panwar, M. (2012). Hand gesture recognition based on shape parameters. *2012 International Conference on Computing, Communication and Applications* (pp. 1-6). IEEE. https://doi.org/10.1109/ICCCA.2012.6179213

Pirsiavash, H., & Ramanan, D. (2012). Detecting activities of daily living in first-person camera views. *2012 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2847–2854). IEEE. https://doi.org/10.1109/CVPR.2012.6248010

Pugeault, N., & Bowden, R. (2011). Spelling it out: Real-time ASL fingerspelling recognition. *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)* (pp. 1114–1119). https://doi.org/10.1109/ICCVW.2011.6130290

Rahim, M. A., Shin, J., & Yun, K. S. (2021). Hand gesture-based sign alphabet recognition and sentence interpretation using a convolutional neural network. *Annals of Emerging Technologies in Computing*, *4*(4), 20-27. https://doi.org/10.33166/AETiC.2020.04.003

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 779–788). IEEE. https://doi.org/10.1109/CVPR.2016.91

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *Advances in Neural Information Processing Systems*, *28*.

Sahoo, J. P., Ari, S., & Patra, S. K. (2019). Hand gesture recognition using PCA based deep CNN reduced features and SVM Classifier. *2019 IEEE International Symposium on Smart Electronic Systems (iSES) (Formerly iNiS)* (pp. 221–224). IEEE. https://doi.org/10.1109/iSES47678.2019.00056

Sahoo, J. P., Sahoo, S. P., Ari, S., & Patra, S. K. (2022). RBI-2RCNN: Residual block intensity feature using a two-stage residual convolutional neural network for static hand gesture recognition. *Signal, Image and Video Processing*, *16*(8), 2019–2027. https://doi.org/10.1007/s11760-022-02163-w

Sahoo, J. P., Sahoo, S. P., Ari, S., & Patra, S. K. (2023). DeReFNet: Dual-stream dense Residual fusion network for static hand gesture recognition. *Displays*, *77*, 102388. https://doi.org/10.1016/j.displa.2023.102388

Sharma, A., Mittal, A., Singh, S., & Awatramani, V. (2020). Hand gesture recognition using image processing and feature extraction techniques. *Procedia Computer Science*, *173*, 181–190. https://doi.org/10.1016/j.procs.2020.06.022

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *ArXiv abs/1409.1556*. https://doi.org/10.48550/arXiv.1409.1556

Srividya, M., Anala, M., Dushyanth, N., & Raju, D. V. S. K. (2019). Hand recognition and motion analysis using faster RCNN. *2019 4th International Conference on Computational Systems and Information Technology for Sustainable Solution (CSITSS)* (pp. 1–4). IEEE. https://doi.org/10.1109/CSITSS47250.2019.9031033

Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017). Inception-v4, inception-ResNet and the impact of residual connections on learning. *Thirty-First AAAI Conference on Artificial Intelligence* (pp. 4278–4284). https://doi.org/10.1609/aaai.v31i1.11231

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1–9). IEEE. https://doi.org/10.1109/CVPR.2015.7298594

Tang, H., Yuan, C., Li, Z., & Tang, J. (2022). Learning attention-guided pyramidal features for few-shot fine-grained recognition. *Pattern Recognition*, *130*, 108792. https://doi.org/10.1016/j.patcog.2022.108792

Utaminingrum, F., Fauzi, M. A., Wihandika, R. C., Adinugroho, S., Kurniawan, T. A., Syauqy, D., Sari, Y. A., & Adikara, P. P. (2017). Development of computer vision based obstacle detection and human tracking on smart wheelchair for disabled patient. *2017 5th International Symposium on Computational and Business Intelligence (ISCBI)* (pp 1–5). IEEE. https://doi.org/10.1109/ISCBI.2017.8053533

Virender, R., Nikita, Y., & Pulkit, G. (2018). American sign language fingerspelling using hybrid discrete wavelet transform-gabor filter and convolutional neural network. *Journal of Engineering Science and Technology*, *13*(9), 2655–2669.

Vu, A.-K. N., Nguyen, N.-D., Nguyen, K.-D., Nguyen, V.-T., Ngo, T. D., Do, T.-T., & Nguyen, T. V. (2022). Few-shot object detection via baby learning. *Image and Vision Computing*, *120*, 104398. https://doi.org/10.1016/j.imavis.2022.104398

Weiss, K., Khoshgoftaar, T. M., & Wang, D. (2016). A survey of transfer learning. *Journal of Big Data*, *3*, 9. https://doi.org/10.1186/s40537-016-0043-6

Xu, C., Cai, W., Li, Y., Zhou, J., & Wei, L. (2020). Accurate hand detection from single-color images by reconstructing hand appearances. *Sensors*, *20*(1), 192. https://doi.org/10.3390/s20010192

Yang, G., Wang, S., & Yang, J. (2019). Desire-Driven Reasoning for Personal Care Robots. *IEEE Access*, *7*, 75203–75212. https://doi.org/10.1109/ACCESS.2019.2921112

Zhang, Y., Cao, C., Cheng, J., & Lu, H. (2018). EgoGesture: A new dataset and benchmark for egocentric hand gesture recognition. *IEEE Transactions on Multimedia*, *20*(5), 1038–1050. https://doi.org/10.1109/TMM.2018.2808769

Zhao, A., Wu, H., Chen, M., & Wang, N. (2023). A spatio-temporal siamese neural network for multimodal handwriting abnormality screening of Parkinson's Disease. *International Journal of Intelligent Systems*, *2023*, 9921809. https://doi.org/10.1155/2023/9921809

Zheng, Q., Yang, M., Yang, J., Zhang, Q., & Zhang, X. (2018). Improvement of generalization ability of deep CNN via implicit regularization in two-stage training process. *IEEE Access*, *6*, 15844–15869. https://doi.org/10.1109/ACCESS.2018.2810849