Real-Time Hand Gesture Recognition: Lightweight Keypoint-Based Approach with Medoid Similarity

Itsaso Rodríguez-Moreno [1,*], David Freire-Obregón [2], José María Martínez-Otzeta [1], and Modesto Castrillón-Santana [2]

[1] Department of Computer Science and Artificial Intelligence, University of the Basque Country (UPV/EHU), Donostia-San Sebastián, Spain
[2] University Institute of Intelligent Systems and Numeric Applications in Engineering (SIANI), University of Las Palmas de Gran Canaria (ULPGC), Las Palmas de Gran Canaria, Spain
Email: itsaso.rodriguez@ehu.eus (I.R.M.); david.freire@ulpgc.es (D.F.O.); josemaria.martinezo@ehu.eus (J.M.M.O.); modesto.castrillon@ulpgc.es (M.C.S.)
*Corresponding author

***Abstract*—Gesture recognition is crucial in computer vision, with applications in security, consumer electronics, and beyond. While deep learning techniques like Convolutional Neural Networks or pre-processing methods such as optical flow achieve high classification accuracy, they often rely on large pre-trained networks or computationally intensive pre-processing, making them unsuitable for real-time or resource-limited applications. This paper introduces a novel lightweight classifier that leverages skeleton features and hand-shape similarity to representative gestures for efficient gesture recognition. By focusing on keypoints that reflect the human body's structure and similarity to static gesture medoids, our approach reduces computational complexity while maintaining competitive performance, as demonstrated in tests on the public Jester database. This work offers an efficient solution suitable for gesture recognition applications requiring real-time processing with limited computational resources.**

***Keywords*—gesture recognition, lightweight, hand keypoints, Recurrent Neural Network (RNN)**

I. Introduction

Gesture recognition plays a key role in Computer Vision, driving advances across applications like sign-language recognition, gaming, and Human-Machine Interaction (HMI). This technology enables intuitive and efficient communication between humans and machines, leveraging natural human gestures to interact with electronic devices. As gesture recognition becomes more integrated into everyday technology, the demand for robust, efficient systems has grown significantly [1].

Nowadays, many HMI researches use deep learning techniques, such as Convolutional Neural Networks (CNNs), to implement gesture recognition. While effective, these methods typically depend on pre-trained models and substantial computational resources, making them hardly practical for real-time applications or devices with limited processing power. This computational complexity and resource demand present a notable challenge, particularly in scenarios requiring real-time interaction and low-latency responses [2].

Recent research has focused on developing more efficient gesture recognition approaches, utilizing lightweight models and feature extraction techniques that reduce computational overhead. One promising direction is skeleton-based recognition systems, which interpret gestures by analyzing human body keypoints. These systems can effectively capture essential gesture features while minimizing data processing and computational requirements, making them well suited for use in resource-constrained environments [3].

Our work introduces a novel lightweight classifier that leverages skeleton features and Procrustes distances relative to 42 prototypes of common hand gestures. These prototypes are defined as the medoids of hand shapes associated with each of the 42 predefined gestures. Medoids, computed based on Procrustes similarity, serve a role analogous to centroids but are used when averaging a set of objects is impractical or infeasible. Each medoid represents the hand shape within a gesture set that is most similar to all others in that set.

By focusing on keypoints that capture the structure of the human hand, our approach reduces computational complexity while achieving competitive performance. The additional computation of Procrustes distances is not only fast but also highly efficient, ensuring system performance remains uncompromised. A small neural network processes these features and generates the final decision, further enhancing speed and efficiency.

The proposed model is evaluated on the public Jester database [4], demonstrating its effectiveness and efficiency on a hand-gesture benchmark database. The results indicate that this method is a practical solution for

Manuscript received October 16, 2024; revised November 25, 2024; accepted January 7, 2025; published April 9, 2025.

real-time gesture recognition applications, making it suitable for a broader range of uses, including consumer electronics, gaming, and interactive systems with limited computational resources.

The structure of the paper is organized as follows. Section II explores research pertinent to our study. This is followed by a detailed presentation of our proposed approach in the methodology section, including a comprehensive pipeline setup description. The paper concludes with sections on the experiments and results in Section IV, and followed by a discussion and a summary of the conclusions.

II. Related Work

Various approaches have been developed to create accurate and efficient hand gesture recognition systems, generally categorized as either sensor-based or vision-based. Sensor-based methods typically use data gloves [5], Electromyography (EMG) signals [6] or Wi-Fi devices [7]. We will focus on vision-based methods, as our research falls into that category.

Depth information can be very useful to interpret correctly the detected gestures. Popular sensors like Microsoft Kinect and Intel RealSense provide depth data along with RGB information. In Ref. [8], an Indian sign language recognition system uses an ensemble of three Support Vector Machines (SVMs), each trained on different features extracted from Kinect data. Another approach is presented by Satybaldina and Kalymova [9], where the Simonyan and Zisserman use a modified VGG16 [10] for classifying gestures based on Intel RealSense data.

Most approaches make use of some form of segmentation to isolate the area of interest (hands) in the scene. Zhou *et al.* [11] segmentate the hands based on skin color, while in Ref. [12] an approach based on oriented gradient local binary patterns to represent both hand contour and texture features is presented. Convolutional neural networks can also be applied to this task, and extended to video sequences, as demonstrated in Ref. [13].

Libraries like MediaPipe Hands [14] now provide skeleton representations of hands detected in images. Several studies have used these features as a starting point to train classifiers. Osipov and Ostanin [15] train a SVM on skeleton features to classify static hand gestures in real-time. In Ref. [16], dynamic gestures are recognized by decoupling the gesture into hand posture variations and hand movements, which are then separately modeled using a 3D CNN and a 2D CNN, respectively.

Recent advancements in action recognition highlight the effectiveness of innovative spatio-temporal analysis techniques. The work described in Ref. [17] introduces Motion Fused Frames (MFFs), a method that integrates motion information into static images for improved classification accuracy, achieving notable results on the Jester test set. Meanwhile, Zhou *et al.* [18] developed the Temporal Relation Network (TRN), which equips convolutional networks with the ability to discern temporal relationships across video frames. TRN excels in tasks requiring an understanding of temporal dynamics, outperforming standard models on the Jester test set. According to the PapersWithCode [1] website, these two studies have achieved the highest reported hand gesture recognition accuracy on the mentioned Jester test set.

Lightweight methods using neural networks have also been proposed. Zhou *et al.* [19] introduce FGDSNet, a two-stage gesture recognition network that uses a dual-branch structure for improved feature fusion. They introduce a Cosine Similarity-KL Divergence Attention Module (CoSKLAM) and a Gate Filtering, achieving real-time inference with a small parameter size. Zhou and Li [20] propose a lightweight static gesture recognition network, PEA-YOLO, to address background variety and lighting changes. Their approach combines adaptive spatial feature pyramids, attention mechanisms, and multi-path feature fusion, yielding high accuracy on experiments on OUHANDS [21] and NUSII [22] datasets.

Recent research also show some examples of hand skeleton extraction for lightweight approaches. For example, a model using an enhanced Gated Recurrent Unit (GRU) for skeleton dynamic graphs is proposed in Ref. [23] to balance model compression and accuracy. Experiments on DHG14/28 [24] and SHREC'17 [25] datasets show the model achieves state-of-the-art performance. A skeleton-based method to analyze dynamic gestures in multi-person scenarios, called ST-TGR is presented in [26]. ST-TGR uses RTMPose [27] for pose estimation and MoGRU, a multi-scale BiGRU with attention, to classify gestures via spatio-temporal representation, achieving superior accuracy and speed over baseline models on several datasets. Self-supervised feature extraction is explored in Ref. [28], where a skeleton-based hand gesture recognition model is proposed that leverages topology-aware self-supervised learning to derive representations from unlabeled data. Their experiments show that their model achieves state-of-the-art results on benchmark datasets and generalizes well with minimal labeled data.

When dealing with dynamic gestures, previous approaches must process both spatial and temporal dimensions of the data. These dimensions can be handled jointly or separately. Some methods employ deep neural networks to process both dimensions together, while others analyze each one independently. In our approach, we address the spatial dimension by measuring the similarity of a static pose in each frame against a set of representative poses that were precomputed. This allows for faster processing than using a deep neural network. Additionally, updating the model is straightforward: one can simply adjust the representative poses, similar to prototype-based nearest-neighbor methods. Our use of a low number of prototypes (42 in this work) also enhances the interpretability of the model's classifications. For the temporal dimension, our neural network has relatively low

[1]https://paperswithcode.com/sota/hand-gesture-recognition-on-jester-test

complexity compared to most of the methods discussed in this section.

Our work makes use of the hand shape representatives provided as open source by Rodriguez-Moreno *et al.* [29]. They developed a sign-language tutor that recognizes the sign performed by the user by computing the Procrustes distance to a set of previously computed representative shapes. We adopt those representatives or prototypes and compute new features based on the distance between the detected shape and each of them.

III. Method

A sketch of our workflow is as follows:

(1) Capture an image frame from a camera.
(2) Extract hand keypoints using MediaPipe Hands.
(3) Compare the shape of the hand, defined by its keypoints, with a set of representative hand medoids. This comparison returns the probability of the hand shape belonging to each class. These probabilities can be Weighted (WDC), Exponential (EFC), or Hard (HARD), as explained later.
(4) Use these probabilities as features, alongside the keypoints themselves, to form a temporal sequence. This sequence is fed into a ConvLSTM neural network to predict dynamic gestures over a time frame.

We now provide a more detailed explanation of this pipeline.

A. Overview

Let I represent a hand-gesture scene situated in a high-dimensional space $\mathbb{R}^n$. We define keypoints $K(I)$ for each frame as a transformation of I into a lower-dimensional space $\mathbb{R}^m$ (where $m < n$), effectively capturing the essential features I as a set of keypoints.

$$K: \mathbb{R}^n \rightarrow \mathbb{R}^m, K(I) = \bigcup_{i=1}^{m} k_i = \{q_1(I), q_2(I), \dots, q_m(I)\} \quad (1)$$

where:

- $K(I)$ represents the keypoints of the input image I, effectively encapsulating the significant features of the hand-gesture scene as a set of discernible coordinates.
- k_i denotes the i-th keypoint in $K(I)$, corresponding to the feature extraction function q_i that isolates specific, informative keypoints from the image I. Each q_i refers to a keypoint in the 3D coordinate system, capturing spatial information as coordinates (x, y, z).
- m is the total number of keypoints extracted, representing the complexity and detail captured in the lower-dimensional representation over time.

This transformation, K, reduces the dimensionality of the hand gesture representation while retaining its essential spatial characteristics. Then, features associated with distances to some representative hand shapes (also called medoids in the following text due to the method used for obtaining them) are computed from K to distill further meaningful information from the 3D coordinate system. By calculating these features, we refine the dimensional representation of the input image $K(I)$ from $\mathbb{R}^m$ into a more semantically meaningful embedding in $\mathbb{R}^p$, effectively encapsulating the critical features of the hand.

$$M: \mathbb{R}^m \rightarrow \mathbb{R}^p, M(I) = m\big(K(I)\big) \in \mathbb{R}^p \quad (2)$$

Here, M is a p-dimensional feature vector that may represent WDC, EFC, or HARD, depending on the specific transformation applied as detailed below, in Section B. To extend the keypoint transformation K and the medoid features transformation M across the temporal axis for all frames in the video sequence, we aggregate these transformations over time to construct a sequence of keypoints and medoid features. This temporal extension involves extracting $K(I)$ and $M(I)$ for each frame t in the sequence, creating comprehensive temporal sequences K and M that capture the evolution of hand gestures over time. These sequences are then fed into a Recurrent Neural Network (RNN) classifier, which can effectively model the temporal dependencies and dynamic patterns inherent in hand gestures.

Formally, for a video sequence with frames $I_1, I_2, \dots, I_t$, we define:

$$K(t) = \{K(I_1), K(I_2), \dots, K(I_t)\}$$

$$M(t) = \{M(I_1), M(I_2), \dots, M(I_t)\}$$

where $K(t)$ denotes the sequence of keypoints and $M(t)$ represents the sequence of medoid features embeddings across all frames. These sequences, $K(t)$ and $M(t)$, are then provided as input to the RNN classifier to capture and learn the temporal dynamics of the hand gestures (see Section D), allowing for accurate classification based on the evolution of features over time. This approach leverages the strengths of RNNs in handling sequential data, providing a robust framework for gesture recognition in video sequences.

B. Medoid-Based Feature Vector Computation

The MediaPipe Hand Tracking solution [14], integrated within the broader MediaPipe framework [30], can estimate the 3D positions of 21 key points on each hand. Each landmark is defined by three coordinates: (x, y, z), where z indicates depth relative to the wrist.

These features can be directly used as predictors for a machine learning algorithm, but it may be helpful to perform a preprocessing step. We transform the data into a 42-dimensional vector of unit length that represents the assigned scores by a distance-based classifier concerning a set of previously defined 42 gesture representatives. The representatives are those presented in [29]. They obtained those representatives as the hand keypoints medoids according to the Procrustes distance of a collection of the same static gesture performed repeatedly. Our hypothesis is that those 42 gesture representatives obtained from the constituents of the Spanish Sign Language cover enough variability to make it possible to represent any possible

hand gesture as an embedding in that space. Therefore, they can act as a basis (in a loose sense) for representing any possible gesture, even beyond those in Sign Languages. In this representation, any gesture can be expressed as a 42-dimensional vector in which each coefficient is related to a representative (medoid).

The feature vector $M(I)$ is obtained by comparing each frame's keypoints to each of the 42 medoid gestures. This involves computing the Procrustes distance D_i from each medoid to the input shape in frame I. Procrustes analysis, applied to two distinct shapes, involves performing a combination of transformations—such as scaling, rotation, and reflection—to minimize the differences between them. This process, which can be seen in Fig. 1, determines the Procrustes distance, a measure of the remaining dissimilarity between the shapes following alignment.

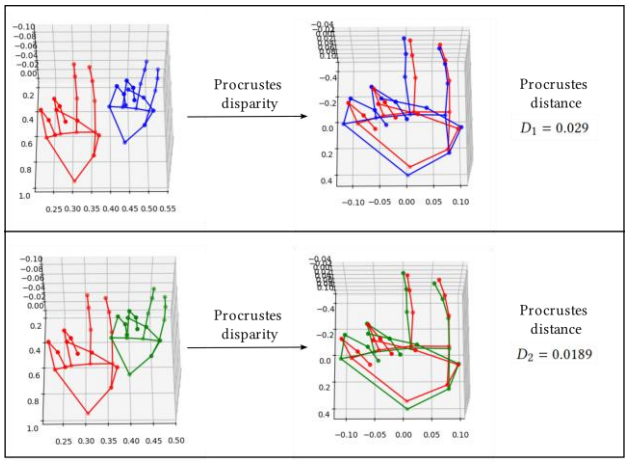

Fig. 1. Two examples of transformations applied in Procrustes analysis are presented along with the corresponding distances (D_i) obtained. In each example, the original hand shapes detected by MediaPipe are shown on the left, while the transformed hand shapes—adjusted to minimize their differences—are displayed on the right.

To convert these distances into features, we calculate class probabilities for each medoid by either of the following methods:

- **Weighted Distance Calculation (WDC)**: In WDC, the probability for each medoid is calculated as the inverse of the Procrustes distance, assigning higher probabilities to closer matches. This method is suitable for emphasizing gestures that closely resemble a particular medoid, enabling clear distinctions between classes:

$$P_i = \frac{1/D_i}{\sum_{j=1}^{n}(1/D_j)} \tag{3}$$

  where P_i denotes the probability for the i-th medoid, and D_i is the Procrustes distance to the i-th medoid.

- **Exponential Function Calculation (EFC)**: EFC applies an exponential decay to the Procrustes distances, effectively smoothing the probabilities, especially in cases where multiple medoids have similar distances to a given gesture. This approach is particularly useful for gestures that may not be strongly aligned with any single medoid but are reasonably close to several:

$$P_i = \frac{e^{-\frac{D_i}{D_{min}}}}{\sum_{j=1}^{n} e^{-\frac{D_j}{D_{min}}}} \tag{4}$$

  where D_{min} is the minimum Procrustes distance in the set, scaling the exponential for each medoid.

In both cases, the 42-dimensional vector $M(I)$ encapsulates the relationship of the input hand gesture to the medoid set, effectively summarizing the gesture as a weighted combination of these fundamental shapes. This feature vector $M(I)$ is generated for each frame in the video, forming a sequence $M(t)$ that tracks the evolution of gestures over time.

In addition to WDC and EFC, a third option, **Hard Classification (HARD)**, is also considered for assigning a single closest medoid per gesture based on the minimum Procrustes distance:

$$argmin_{i\in\{1,\ldots,m\}}\, D(X, M_i) \tag{5}$$

HARD classification is a simpler approach where each input gesture is assigned to a single representative medoid.

In this case, the $M(t)$ embedding has just one dimension, the index of the closest medoid.

As a toy example with three cases, assume an input frame I yields Procrustes distances $D_1 = 2.0$, $D_2 = 3.0$, and $D_3 = 5.0$ to three medoids M_1, M_2, and M_3. We compute probabilities using the three methods: Weighted Distance Calculation (WDC), Exponential Function Calculation (EFC), and Hard Classification (HARD):

In WDC, the probability for each medoid is computed as:

$$P_i = \frac{1/D_i}{\sum_{j=1}^{n}(1/D_j)}$$

Substituting the distances:

$$P_1 = \frac{1/2.0}{\left((1/2.0) + (1/3.0) + (1/5.0)\right)} \approx 0.484$$

$$P_2 = \frac{1/3.0}{\left((1/2.0) + (1/3.0) + (1/5.0)\right)} \approx 0.323$$

$$P_3 = \frac{1/5.0}{\left((1/2.0) + (1/3.0) + (1/5.0)\right)} \approx 0.193$$

Thus, the probabilities are:

$$P = [0.484, 0.323, 0.193]$$

For EFC, probabilities are computed as:

$$P_i = \frac{e^{-\frac{D_i}{D_{min}}}}{\sum_{j=1}^{n} e^{-\frac{D_j}{D_{min}}}}$$

where $D_{min} = 2.0$. Substituting the distances:

$$P_1 = \frac{e^{-1}}{e^{-1} + e^{-1.5} + e^{-2.5}} \approx 0.547$$

$$P_1 = \frac{e^{-1.5}}{e^{-1} + e^{-1.5} + e^{-2.5}} \approx 0.331$$

$$P_1 = \frac{e^{-2.5}}{e^{-1} + e^{-1.5} + e^{-2.5}} \approx 0.122$$

Thus, the probabilities are:

$$P = [0.547, 0.331, 0.122]$$

In HARD, the gesture is assigned to the medioid with the smallest Procrustes distance:

$$argmin_{i \in \{1,2,3\}} D_i$$

Here, $D_1 = 2.0$ is the smallest, so the gesture is assigned to medoid M_1.

The results for the three methods are summarized below:

- **WDC:** $P = [0.484, 0.323, 0.193]$
- **EFC:** $P = [0.547, 0.331, 0.122]$
- **HARD:** Assigned to M_1 (closest medoid)

C. Temporal Extension and Classification

To capture dynamic aspects of hand gestures, we aggregate the computed feature vectors $M(I)$ across each frame I_t in a video sequence, yielding a temporal sequence $M(t) = \{M(I_1), M(I_2), \ldots, M(I_t)\}$. Alongside the sequence of keypoints $K(t) = \{K(I_1), K(I_2), \ldots, K(I_t)\}$, these temporal sequences serve as inputs for a RNN classifier.

The RNN leverages the temporal structure of these sequences, learning patterns that characterize distinct gestures over time. This approach, utilizing both keypoints and medoid-based features, allows the classifier to robustly capture the temporal dynamics of hand gestures.

D. ConvLSTM Classifier with Multi-Stream Input Integration

The proposed classifier for hand gesture recognition combines Conv1D, LSTM, and multi-stream input integration to process and classify gestures efficiently. The architecture consists of four input streams: keypoints $K(t)$, i.e., (x, y, z), and a hand medoids feature vector $M(t)$. Recent applications of multi-stream architectures have successfully addressed various sequence-related problems [31]. As illustrated in Fig. 2, each stream undergoes Conv1D, MaxPooling1D, Dropout, Reshape, LSTM, and Flatten layers to extract meaningful features and capture temporal dependencies. The transformations applied to the input tensors χ_i are denoted as $f_i(\chi_i)$, with each channel i (e.g., $K(t)_x$, $K(t)_y$, $K(t)_z$, and $M(t)$) having its own distinct set of parameters (i.e., weights are not shared between channels). Subsequently, the outputs from all channels are concatenated and processed through a dense layer, followed by a SoftMax layer to generate the classification outputs $\hat{y}_{g,c}$.

The model employs the categorical cross-entropy loss function, which is well suited for multi-class classification tasks. The categorical cross-entropy loss can be defined as:

$$L = -\frac{1}{N}\sum_{g=1}^{N}\sum_{c=1}^{C} y_{g,c} \log(\hat{y}_{g,c}) \quad (6)$$

where:

- N is the number of samples in the batch.
- C is the number of classes.
- $y_{g,c}$ is a binary indicator (0 or 1) if class label c is the correct classification for sample g.
- $\hat{y}_{g,c}$ is the predicted probability that the sample g belongs to class c.

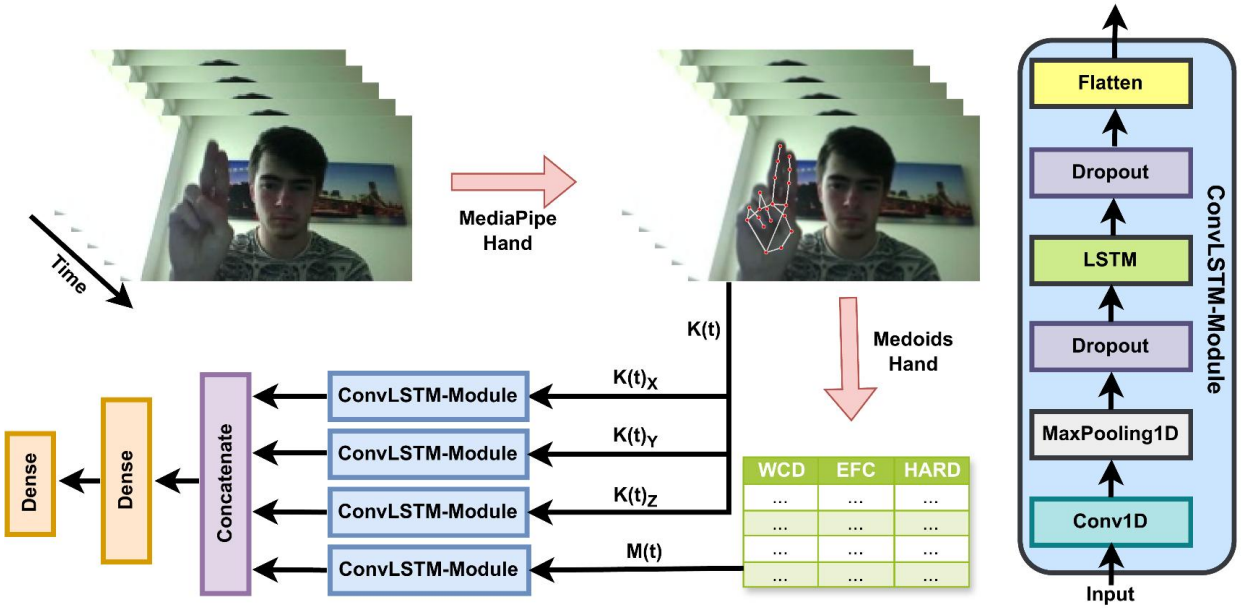

Fig. 2. The proposed hand gesture recognition pipeline consists of three stages: pre-processing with MediaPipe Hand to detect the hand skeleton K(t), medoid features computation to derive feature vectors M(t), and classification. The classification block processes four data streams (x, y, z, and the selected medoid feature) through ConvLSTM modules, concatenating the outputs for the final classification step.

IV. Experimental Famework

A. Experimental Setup

Dataset. The Jester dataset, described in [4], is a comprehensive video collection designed for hand gesture recognition. It comprises 148,092 labeled video clips of individuals performing 27 distinct, pre-defined hand gestures, such as sliding two fingers down, swiping left or right, and drumming fingers in front of a laptop camera or webcam at 30 fps. Over 1,300 unique crowd workers performed these clips, which are densely labeled to facilitate the training of machine learning models. The dataset is thoughtfully divided into training, validation, and test sets consisting of 118.5 K, 14.7 K, and 14.7 K videos, respectively, following an 8:1:1 split ratio. It also includes two "no gesture" classes to help distinguish between specific gestures and non-descriptive hand movements, enhancing the model's ability to recognize and accurately classify diverse hand actions. Our method relies on successfully detecting hand keypoints across sufficient frames to produce meaningful inputs for the neural network.

Metrics. In our analysis, we evaluated the performance of the classification model across 27 classes using accuracy and model loss as key metrics. This approach aligns with methodologies previously reported in the literature, specifically in studies such as [17, 18]. Accuracy serves as a fundamental metric, reflecting the proportion of correct predictions made by the model and offering a straightforward measure of its effectiveness in classifying the data correctly. Model loss, on the other hand, plays a significant role in understanding the behavior of the model. It offers insight into how well the model's predictions align with the actual values, essentially measuring the error in these predictions. By monitoring the loss, we gain valuable information about the model's convergence behavior and its generalization ability on unseen data.

Frame discretization. Not every frame is used in the classification process to streamline computation. Specifically, when the number of frames, denoted as $|G|$ (the cardinality of set G), falls below the specified threshold η, the function computes the padding needed, p, where $p = \eta - |G|$. To identify which frames to replicate for padding, the function computes p indices using linear interpolation between 0 and $|G| - 1$ inclusive. Conversely, when the number of elements $|G|$ exceeds η, the function selects which η frames to retain. The trimming frames indices are also calculated using a linear interpolation between 0 and $|G| - 1$ inclusive. Consequently, in both cases, padding and trimming, the frame selection follows this equation:

$$f_k = a + k \cdot \frac{b-a}{\eta-1} \quad \text{for } k = 0,1,2,\dots,\eta-1 \qquad (7)$$

where f_k represents the k-th frame in the sequence. The index a marks the starting index for frame selection, while index b represents the endpoint. The total number of frame indexes to be selected is denoted by η. The variable k is used to iterate through the sequence, starting from 0 and incrementing up to $\eta - 1$, to select the appropriate frames within the specified range from a to b. This method preserves the structural integrity and temporal distribution of the video data, optimizing it for further processing. We have established a minimum requirement of frames for effective processing, with Υ defined as the videos meeting this criterion. This ensures that the input fed into the neural network is consistent and reliable, forming a robust foundation for accurate gesture recognition.

Implementation Details. The initial learning rate was set to 0.001, with a mini-batch size of 64. However, due to memory constraints encountered while analyzing state-of-the-art models, the mini-batch size was reduced to 10 in the experiments presented in the next section.

B. Experimental Evaluation

Table I presents the test accuracy and loss for various inputs and η, offering a comprehensive view of model performance under different sequence-length conditions. The first four rows display results using homogeneous inputs, indicating the effects of individual data types (XYZ, WCD, EFC, HARD) on model outputs. These inputs are tested across multiple η settings: 5, 15, 25, and 35 frames. Generally, accuracy increases as the number of frames grows, suggesting that more frames provide a richer dataset for the model to learn from, thus enhancing its predictive capabilities. Notably, the test accuracy improves significantly as the frame count increases from 5 to 25 for all input types, but there is a slight decrease or stabilization when moving to 35 frames. For example, input XYZ reaches a peak accuracy of 91.7% for 25 frames, with a slight drop to 90.9% at 35 frames, hinting at a potential overfitting or inefficiency in utilizing additional frames beyond a certain point. Meanwhile, the loss consistently decreases, suggesting that the model is effectively minimizing errors with more data. However, the rate of improvement in loss reduction slows or plateaus at higher frame counts.

The last three rows illustrate results with heterogeneous inputs, where combinations of different data types (XYZ ∪ WCD, XYZ ∪ EFC, XYZ ∪ HARD) are used. These combined inputs consistently yield higher accuracies and lower losses than most single input types, especially evident in the 25-frame setting where accuracy peaks and loss reaches its minimum across the dataset. For example, the combination of XYZ ∪ WCD achieves the highest accuracy of 92.4% at 25 frames. This enhancement in performance with heterogeneous inputs suggests that the model benefits from a diverse range of data features, leading to a more robust generalization and a stronger predictive performance.

Table II presents a comparative analysis of different approaches for gesture recognition on the Jester test set, focusing on both performance and computational efficiency. The entries in the table provide insights into various metrics, including accuracy, number of parameters, preprocessing time, training time per epoch, and inference time. While the 20BN Jester System [18], an earlier approach, reports only accuracy (82.3%), it serves

as a baseline for evaluating the improvements introduced by later methods.

The Multiscale TRN and the 8-MFFs-3f1c approaches use ResNet101 as their base model, contributing to their impressive accuracy scores of 94.7% and 96.2%, respectively. Built on a robust deep learning architecture, these systems manage to capture complex temporal and spatial patterns critical for accurate gesture recognition. However, the use of such a sophisticated base model results in a high computational demand, as evidenced by their large model sizes (over 42.8 million parameters) and training times on a GPU NVIDIA GeForce RTX 2080 (exceeding 23.2 min per epoch). Inference times greater than 40 ms further suggest that, while they offer high accuracy, these systems may not be suitable for real-time applications requiring rapid decision-making.

TABLE I. TEST ACCURACY AND TEST LOSS FOR VARIOUS INPUTS AND FRAME COUNTS. THE FIRST FOUR ROWS PRESENT RESULTS WITH HOMOGENEOUS INPUTS, WHILE THE LAST THREE ROWS SHOW RESULTS WITH HETEROGENEOUS INPUTS

Inputs	$\eta = 5$ Frames		$\eta = 15$ Frames		$\eta = 25$ Frames		$\eta = 35$ Frames	
	Accuracy ↑	Loss ↓	Accuracy ↑	Loss ↓	Accuracy ↑	Loss ↓	Accuracy ↑	Loss ↓
XYZ	82.2%	0.54	89.8%	0.35	91.7%	0.33	90.9%	0.31
WCD	77.8%	0.66	83.6%	7.0	85.6%	0.51	84.6%	0.53
EFC	77.4%	0.64	83.1%	7.0	84.1%	0.56	83.7%	0.54
HARD	59.3%	1.37	64.6%	1.12	65.9%	1.14	65.5%	1.14
XYZ ∪ WCD	84.7%	0.51	91.7%	0.31	92.4%	0.31	91.5%	0.32
XYZ ∪ EFC	84.6%	0.50	91.2%	0.31	91.9%	0.31	91.6%	0.31
XYZ ∪ HARD	84.7%	0.50	91.4%	0.32	92.1%	0.31	91.8%	0.31

TABLE II. PERFORMANCE AND TIME METRICS (τ) ON THE JESTER TEST SET

Approach	Accuracy ↑	#Parameters ↓	Prepro. τ ↓	Train τ ↓	Inference τ ↓
20BN Jester System	82.3%	N/A	N/A	N/A	N/A
VideoLSTM [18]	85.8%	N/A	N/A	N/A	N/A
3D-ShuffleNetV2 0.25x [32]	86.9%	0.83 M	0 ms	1.2 min	0.23 ms
3D-SqueezeNet [32]	90.7%	2.1 M	0 ms	2.3 min	1.85 ms
Multiscale TRN [18]	94.7%	> 42.8 M	9 ms	> 23.2 min	> 40 ms
8-MFFs-3flc [17]	96.2%	> 42.8 M	0 ms	> 23.2 min	> 40 ms
Ours	92.4%	0.8 M	1.9 ms	1.1 min	0.12 ms

The lightweight models, such as 3D-ShuffleNetV2 0.25x and 3D-SqueezeNet, represent significant progress in balancing performance with computational efficiency. VideoLSTM achieves an accuracy of 85.8%, marking an improvement over the 20BN Jester System. However, its computational metrics are not detailed in this evaluation. In contrast, 3D-ShuffleNetV2 0.25x delivers an accuracy of 86.9% with an exceptionally compact model size of just 0.83 million parameters. Additionally, it features a short training time of only 1.2 min per epoch and a fast inference time of 0.23 ms, making it well-suited for real-time applications.

The 3D-SqueezeNet pushes the boundaries of lightweight models further, achieving an accuracy of 90.7% with a modest 2.1 million parameters. Although its inference time is slightly longer at 1.85 ms, it maintains preprocessing and training times comparable to those of 3D-ShuffleNetV2. Both models demonstrate the potential of smaller, faster networks to achieve competitive results while avoiding the computational demands of larger architectures.

In contrast, our proposed method achieves an optimal balance between accuracy and efficiency. With an accuracy of 92.4%, it approaches the high benchmarks set by the more complex models while dramatically reducing the number of parameters to just 0.8 million, a size even smaller than that of the 3D-SqueezeNet model. This substantial reduction in model size simplifies the training process and may lower the risk of overfitting, thereby enhancing the model's generalizability to unseen data.

From a computational perspective, our approach exhibits exceptional efficiency. The preprocessing time is remarkably low at just 1.9 ms, and the training time per epoch is significantly reduced to only 1.1 min. Most notably, the inference time of 0.12 ms per input substantially improves the model's suitability for real-time applications, outperforming the state-of-the-art models in speed without a substantial compromise in accuracy. This efficiency makes our approach particularly attractive for deployment in environments where computational resources or time are limited.

C. Error Analysis

This section presents an error analysis of our lightweight classifier, designed to identify 27 distinct hand gestures. As demonstrated in the previous section, the proposed classifier's streamlined architecture has been optimized for rapid processing and minimal resource consumption, making it well-suited for real-time applications. Despite its advantages in terms of speed and efficiency, like all models, it is not entirely immune to errors. Here, we systematically examine the types of errors, their frequencies, and the conditions under which they are most likely to occur. This analysis not only highlights the current model's limitations but also sets the foundation for targeted improvements. By examining both quantitative metrics and qualitative examples, we provide a detailed understanding of the classifier's strengths and weaknesses. This comprehensive approach aims to guide future refinements, enhancing the model's accuracy and robustness for deployment in real-world scenarios.

Table III presents the mapping between numerical labels and their corresponding hand gestures. For example, the numerical label 0 is assigned to the gesture labeled as *doing other things*. This table serves as a helpful reference for understanding the association of each label with its specific gesture, aiding in the interpretation of model outputs and the analysis of classifier performance. In this context, the confusion matrix for the combined XYZ ∪ WCD input on the Jester database test set is provided in Fig. 3. This matrix provides a visual representation of the classification performance, detailing the accuracy with which each gesture is predicted under the aggregated input conditions.

TABLE III. INDEX AND GESTURE MAPPING

Idx	Gesture	Idx	Gesture
0	Doing other things	14	Stop Sign
1	Drumming Fingers	15	Swiping Down
2	No gesture	16	Swiping Left
3	Pulling Hand	17	Swiping Right
4	Pulling Two Fingers	18	Swiping Up
5	Pushing Hand Away	19	Thumb Down
6	Pushing Two Fingers Away	20	Thumb Up
7	Rolling Hand Backward	21	Turning Hand Clockwise
8	Rolling Hand Forward	22	Turning Hand Counterclockwise
9	Shaking Hand	23	Zooming In With Full Hand
10	Sliding Two Fingers Down	24	Zooming In With Two Fingers
11	Sliding Two Fingers Left	25	Zooming Out With Full Hand
12	Sliding Two Fingers Right	26	Zooming Out With Two Fingers
13	Sliding Two Fingers Up		

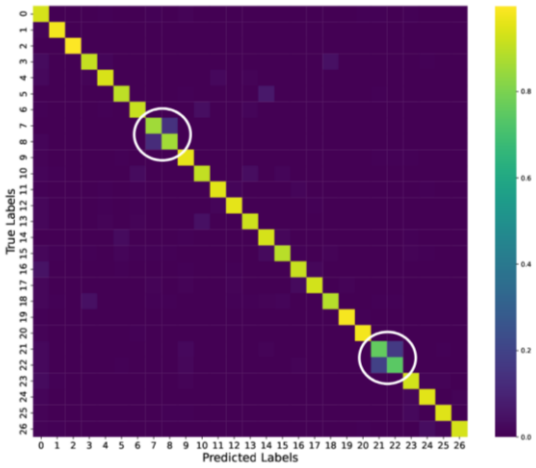

Fig. 3. XYZ ∪ WCD confusion matrix on the Jester database test set. Circles are drawn on the conflicting class pairs 7–8 and 21–22.

The confusion matrix for the XYZ ∪ WCD model on the Jester database test set generally demonstrates excellent performance, particularly with high accuracy rates along its diagonal, indicating that most gestures are classified with substantial precision. However, there are notable exceptions that highlight specific challenges in gesture classification. Specifically, hand rolling and turning gestures exhibit confusion between similar movements.

For instance, gestures 7 (rolling hand backward) and 8 (rolling hand forward) show significant cross-activation: gesture 7 is correctly identified 85.6% of the time, but 14.0% is misclassified as gesture 8. Similarly, gesture 8 is accurately identified 86.0% of the time but is misclassified 12.1% as gesture 7. This confusion is likely due to the physical similarity between the forward and backward rolling motions, which the model may struggle to differentiate.

A similar pattern occurs between gestures 21 (turning hand clockwise) and 22 (turning hand counterclockwise). Gesture 21 is correctly identified 75.9% of the time, but a significant 16.7% of cases are misclassified as gesture 22. Gesture 22, in turn, is accurately recognized 74.2% of the time but is confused with gesture 21 in 19.1% of cases. This error may arise from the rotational symmetry of the gestures, making it difficult to distinguish the direction of hand rotation.

These findings suggest that while the model performs well overall, it faces challenges when classifying gestures with closely related movements.

Figs. 4 and 5 provide a visual illustration of sequences our classifier has incorrectly identified, highlighting specific challenges in gesture recognition. In the first scenario, depicted in the top two sequences, the classifier appears to struggle with discerning movements where the hand is moved forward and backward. This suggests a potential sensitivity issue in detecting subtle depth variations, which are crucial for accurately interpreting these gestures. Despite attempts to refine model performance by increasing the number of frames per sequence (as indicated by the experimental results with $\eta = 35$), the classifier shows only a slight improvement in distinguishing between gestures 7 (rolling hand backward) and 8 (rolling hand forward), with accuracy rates adjusting marginally by 0.6 percentage points.

Fig. 4. Misclassified sequences for gestures 7 (rolling hand backward), 8 (rolling hand forward), 21 (turning hand clockwise), and 22 (turning hand counterclockwise). The trajectory of the gestures is also shown to better understand the differences between them. Problematic gestures with the same label color underscore potential confusion.

Fig. 5. Misclassified sequences for gestures 3 (pulling hand in), 18 (swiping up), 5 (pushing hand away), and 14 (stop sign). The trajectory of the gestures is also shown to better understand the differences between them. Problematic gestures with the same label color underscore potential confusion.

The second scenario involves challenges related to finger rotation, showcased in the bottom two sequences. In this case, the classifier struggles due to insufficient capture of sequential information necessary to discern the subtle differences in finger orientation. This issue persists despite the increased frame count, as evidenced by the accuracy metrics for gestures 21 (turning hand clockwise) and 22 (turning hand counterclockwise), which show a decrease in performance for gesture 21 and only a minor improvement for gesture 22 when moving from $\eta = 25$ to $\eta = 35$.

In Fig. 5, additional misclassified sequences for gestures 3 (pulling hand in), 18 (swiping up), 5 (pushing hand away), and 14 (stop sign) reveal further limitations of the classifier. The confusion between gestures 3 and 18 can be attributed to overlapping motion trajectories, with both involving upward or inward hand movements that are difficult to differentiate without more precise temporal segmentation. Similarly, gestures 5 and 14 pose challenges due to their reliance on similar hand positions and stopping motions, which the classifier struggles to disambiguate, especially when contextual cues in the sequences are not adequately captured. These observations emphasize the need for improved temporal modeling and motion sensitivity in the classifier to enhance recognition accuracy for these gestures.

These findings indicate that adding more frames to the sequence does not effectively address the fundamental problem of capturing and processing the critical features required for distinguishing these gestures. The consistent issues across different frame settings highlight the need for further improvements in the classifier's ability to process and interpret the temporal and spatial details inherent in complex hand movements. This could involve refining the feature extraction methods or exploring more sophisticated temporal analysis techniques to handle the subtle dynamics of hand and finger gestures, such as investigating depth estimation techniques without compromising the execution time.

V. Conclusions

In this study, we present a novel lightweight classifier for gesture recognition that leverages hand-skeleton features, demonstrating its efficacy in addressing the computational constraints of existing deep learning models. By focusing on key points that reflect the human hand's structure, our approach significantly reduces computational complexity while maintaining competitive performance. Our findings on the Jester dataset highlight the practical advantages of our method, particularly for real-time and resource-constrained applications where traditional and best-performing models may struggle due to their heavy reliance on pre-trained networks and extensive preprocessing techniques.

Our experimental evaluation demonstrates the robustness and efficiency of our proposed classifier across diverse input conditions and frame counts. We observed that increasing the number of frames generally improved classification accuracy, indicating the model's ability to leverage richer temporal information for enhanced predictive capabilities. Notably, our method achieved a peak accuracy of 92.4% with minimal preprocessing and short training times, distinguishing it from more complex models that demand substantial computational resources.

A comparison with state-of-the-art models yielded promising results for our approach. While traditional deep learning architectures, such as the Multiscale TRN and 8-MFFs-3f1c excel in accuracy, their high computational demands limit their suitability in scenarios requiring rapid decision-making. In contrast, our lightweight classifier strikes a balance by achieving high accuracy (92.4%) with significantly fewer parameters and faster inference times (0.12 ms per input). This efficiency positions our model as a compelling alternative for deployment in environments where speed and computational efficiency are critical.

An in-depth error analysis offered insights into the classifier's strengths and areas for improvement. The confusion matrix revealed challenges in distinguishing subtle hand gesture variations, such as hand rolling and turning motions. These findings highlight the need for future improvements in feature extraction and temporal analysis techniques to further refine the classifier's ability to accurately recognize discern intricate hand movements. Addressing these challenges may involve exploring advanced depth estimation methods or incorporating more sophisticated temporal modeling approaches to improve classification accuracy across diverse gesture types.

Regarding ethical considerations, privacy and attention to diversity are of interest. Privacy is largely assured as only skeleton data is used, minimizing the risk of capturing sensitive personal information. Procrustes analysis, which normalizes for translations, rotations, and scaling, effectively eliminates differences in size within the original data, thus reducing the likelihood of inferring age or gender based on body proportions. Cultural differences in gesture performance may also pose challenges, as gestures can vary in meaning or execution across cultural contexts, potentially impacting recognition accuracy. Researchers must consider these factors, especially when deploying solutions trained in one cultural background within a different context, to ensure robust and inclusive performance.

To address cultural biases in gesture recognition, several mitigation strategies can be employed. One approach is to expand training datasets by including samples from diverse cultural backgrounds. This enhances the model's exposure to a wider range of gesture variations, improving its ability to generalize across cultures. Another strategy is to incorporate transfer learning, where models pre-trained on culturally diverse datasets can be fine-tuned for specific contexts. Additionally, adaptive learning techniques could enable models to adjust dynamically when deployed in new cultural environments. Researchers should also engage with local communities to co-design culturally sensitive gesture sets, ensuring that the system's interpretations align with local norms and practices. Combining these strategies can help create more inclusive and culturally aware gesture recognition systems.

Our research has practical implications for fields such as HMI, security systems, and consumer electronics. By providing a lightweight yet effective solution for gesture recognition, our classifier opens avenues for implementing responsive and intuitive interfaces in devices ranging from smartphones to smart home technologies. Moreover, its efficiency in real-time processing makes it well-suited for applications requiring instant gesture interpretation without sacrificing accuracy.

As further work, we plan to extend our evaluation by testing the proposed classifier on additional benchmark datasets to assess its generalization capabilities across diverse gesture recognition scenarios. Exploring the integration of depth information from RGB-D sensors could also enhance the classifier's performance by providing richer spatial context, potentially improving accuracy in distinguishing complex hand gestures.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

I. Rodríguez-Moreno, D. Freire-Obregón and J. M. Martínez-Otzeta conducted the research, implemented the experiments, analyzed the results, and wrote the original draft; M. Castrillón-Santana supervised the overall process, secured the funding, and contributed to the final draft; all authors had approved the final version.

FUNDING

This work has been partially funded by the Basque Government, Spain, under Research Teams Grant number IT1427-22, and the Spanish Ministry of Science (MCIU), the State Research Agency (AEI), the European Regional Development Fund (FEDER), under Grant number PID2021-122402OB-C21 (MCIU/AEI/FEDER, UE).

REFERENCES

[1] N. Mohamed, M. B. Mustafa, and N. Jomhari, "A review of the hand gesture recognition system: Current progress and future directions," *IEEE Access*, vol. 9, pp. 157422–157436, 2021.

[2] A. Mujahid, M. J. Awan, A. Yasin, M. A. Mohammed, R. Damaševičius, R. Maskeliūnas, and K. H. Abdulkareem, "Real-time hand gesture recognition based on deep learning YOLOv3 model," *Applied Sciences*, vol. 11, 4164, 2021.

[3] D. Freire-Obregón, M. Castrillón-Santana, P. Barra, C. Bisogni, and M. Nappi, "An attention recurrent model for human cooperation detection," *Computer Vision and Image Understanding*, 102991, 2020.

[4] J. Materzynska, G. Berger, I. Bax, and R. Memisevic, "The Jester dataset: A large-scale video dataset of human gestures," in *Proc. the IEEE/CVF International Conference on Computer Vision Workshops*, 2019.

[5] Y. Dong, J. Liu, and W. Yan, "Dynamic hand gesture recognition based on signals from specialized data glove and deep learning algorithms," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–14, 2021.

[6] A. Jaramillo-Yánez, M. E. Benalcázar, and E. Mena-Maldonado, "Real-time hand gesture recognition using surface electromyography and machine learning: A systematic literature review," *Sensors*, vol. 20, 2467, 2020.

[7] S. Tan and J. Yang, "Fine-grained finger gesture recognition using WiFi signals," arXiv preprint, arXiv:2106.00857, 2021.

[8] T. Raghuveera, R. Deepthi, R. Mangalashri, and R. Akshaya, "A depth-based Indian sign language recognition using Microsoft Kinect," *Sādhanā*, vol. 45, pp. 1–13, 2020.

[9] D. Satybaldina and G. Kalymova, "Deep learning based static hand gesture recognition," *Indonesian Journal of Electrical Engineering and Computer Science*, 2021.

[10] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. the International Conference on Learning Representations (ICLR)*, 2015.

[11] W. Zhou, C. Lyu, X. Jiang, P. Li, H. Chen, and Y.-H. Liu, "Real-time implementation of vision-based unmarked static hand gesture recognition with neural networks based on FPGAs," in *Proc. the 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2017.

[12] J. Sun, Z. Zhang, L. Yang, and J. Zheng, "Multi-view hand gesture recognition via Pareto optimal front," *IET Image Processing*, vol. 14, pp. 3579–3587, 2020.

[13] S. Paul, A. Bhattacharyya, A. F. Mollah, S. Basu, and M. Nasipuri, "Hand segmentation from complex background for gesture recognition," in *Proc. IEM Graph 2018, Emerging Technology in Modelling and Graphics*, 2020.

[14] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann, "MediaPipe hands: On-device real-time hand tracking," arXiv preprint, arXiv:2006.10214, 2020.

[15] A. Osipov and M. Ostanin, "Real-time static custom gestures recognition based on skeleton hand," in *Proc. the 2021 International Conference Nonlinearity, Information and Robotics (NIR)*, 2021.

[16] J. Liu, Y. Liu, Y. Wang, V. Prinet, S. Xiang, and C. Pan, "Decoupled representation learning for skeleton-based gesture recognition," in *Proc. the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[17] O. Köpüklü, N. Köse, and G. Rigoll, "Motion fused frames: Data level fusion strategy for hand gesture recognition," in *Proc. the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018, pp. 2184–2188.

[18] B. Zhou, A. Andonian, and A. Torralba, "Temporal relational reasoning in videos," in *Proc. the European Conference on Computer Vision*, 2017.

[19] G. Zhou, Z. Cui, and J. Qi, "FGDSNet: A lightweight hand gesture recognition network for human robot interaction," *IEEE Robotics and Automation Letters*, 2024.

[20] W. Zhou and X. Li, "PEA-YOLO: A lightweight network for static gesture recognition combining multiscale and attention mechanisms," *Signal, Image and Video Processing*, vol. 18, pp. 597–605, 2024.

[21] M. Matilainen, P. Sangi, J. Holappa, and O. Silvén, "OUHANDS database for hand detection and pose recognition," in *Proc. the 2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA)*, 2016.

[22] P. K. Pisharady, P. Vadakkepat, and A. P. Loh, "Attention based detection and recognition of hand postures against complex backgrounds," *International Journal of Computer Vision*, vol. 101, pp. 403–419, 2013.

[23] J. Ni, Y. Wang, G. Tang, W. Cao, and S. X. Yang, "A lightweight GRU-based gesture recognition model for skeleton dynamic graphs," *Multimedia Tools and Applications*, pp. 1–26, 2024.

[24] Q. De Smedt, H. Wannous, and J.-P. Vandeborre, "Skeleton-based dynamic hand gesture recognition," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016.

[25] Q. De Smedt, H. Wannous, J.-P. Vandeborre, J. Guerry, B. le Saux, and D. Filliat, "SHREC'17 track: 3D hand gesture recognition using a depth and skeletal dataset," in *Proc. the 3DOR-10th Eurographics Workshop on 3D Object Retrieval*, 2017.

[26] Z. Chen, W. Huang, H. Liu, Z. Wang, Y. Wen, and S. Wang, "ST-TGR: Spatio-temporal representation learning for skeleton-based teaching gesture recognition," *Sensors*, vol. 24, 2589, 2024.

[27] T. Jiang, P. Lu, L. Zhang, N. Ma, R. Han, C. Lyu, Y. Li, and K. Chen, "RTMPose: Real-time multi-person pose estimation based on MMPose," arXiv preprint, arXiv:2303.07399, 2023.

[28] O. Ikne, B. Allaert, and H. Wannous, "Skeleton-based self-supervised feature extraction for improved dynamic hand gesture recognition," in *Proc. the International Conference on Automatic Face and Gesture Recognition*, 2024.

[29] I. Rodriguez-Moreno, J. M. Martinez-Otzeta, and B. Sierra, "HAKA: HierArchical knowledge acquisition in a sign language tutor," *Expert Systems with Applications*, vol. 215, 119365, 2023.
[30] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee *et al.*, "MediaPipe: A framework for building perception pipelines," arXiv preprint, arXiv:1906.08172, 2019.
[31] D. Freire-Obregón, D. Hernández-Sosa, O. J. Santana, J. Lorenzo-Navarro, and M. Castrillón-Santana, "Towards bi-hemispheric emotion mapping through EEG: A dual-stream neural network approach," in *Proc. the International Conference on Automatic Face and Gesture Recognition Workshops*, 2024.
[32] O. Köpüklü, N. Kose, A. Gunduz, and G. Rigoll, "Resource Efficient 3D Convolutional Neural Networks," in *Proc. the 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 2019, pp. 1910–1919.