Cognitive Computation (2023) 15:2014–2033
https://doi.org/10.1007/s12559-023-10174-z

Snapture—a Novel Neural Architecture for Combined Static and Dynamic Hand Gesture Recognition

Hassan Ali[1] · Doreen Jirak[2] · Stefan Wermter[1]

Received: 25 May 2022 / Accepted: 30 June 2023 / Published online: 17 July 2023

Abstract

As robots are expected to get more involved in people's everyday lives, frameworks that enable intuitive user interfaces are in demand. Hand gesture recognition systems provide a natural way of communication and, thus, are an integral part of seamless human-robot interaction (HRI). Recent years have witnessed an immense evolution of computational models powered by deep learning. However, state-of-the-art models fall short of expanding across different gesture domains, such as emblems and co-speech. In this paper, we propose a novel hybrid hand gesture recognition system. Our *Snapture* architecture enables learning both static and dynamic gestures: by capturing a so-called *snapshot* of the gesture performance at its peak, we integrate the hand pose and the dynamic movement. Moreover, we present a method for analyzing the motion profile of a gesture to uncover its dynamic characteristics, which allows regulating a static channel based on the amount of motion. Our evaluation demonstrates the superiority of our approach on two gesture benchmarks compared to a state-of-the-art CNNLSTM baseline. Our analysis on a gesture class basis unveils the potential of our *Snapture* architecture for performance improvements using RGB data. Thanks to its modular implementation, our framework allows the integration of other multimodal data, like facial expressions and head tracking, which are essential cues in HRI scenarios, into one architecture. Thus, our work contributes both to integrative gesture recognition research and machine learning applications for non-verbal communication with robots.

Keywords Co-speech gestures · Dynamic gesture recognition · Convolutional neural networks · Long short-term memory

Introduction

Gestures are a form of non-verbal communication prominently used in day-to-day communication. Therefore, they can play a fundamental part of human-robot interaction (HRI). Gestures are categorized in the literature as static and dynamic [1]. Static gestures portray meanings through hand postures. They can substitute words or be used together with words in the form of signs or emblems. Such gestures can be recognized by precisely interpreting the emphasized hand shape and spelled-out finger arrangements [2]. In contrast, a dynamic gesture has a temporal aspect articulated through hand movement. Therefore, recognizing it requires employing different techniques, e.g., segmenting and tracking the moving body limb (we refer to a good overview of gesture recognition techniques by Anwar et al. [3]).

However, such categorization of gesture types might be oversimplified. More specifically, static information is essential for recognizing dynamic gestures with similar movement paths. For example, the gesture commands "stop" and "go forward" have an identical motion with the arm extending forward. Understanding these two commands requires observing their unique hand shape and finger arrangements (open palm vs. extended finger). Furthermore, a precise interpretation of the distinctive characteristics of each hand gesture is desired for a smooth HRI experience. This is also vital in robot applications with safety concerns, e.g., medical or industrial applications. Confusion between gestures in such environments might have severe safety consequences.

This precise interpretation is challenging for approaches that rely solely on RGB data. However, RGB-based methods are beneficial [4] because of their convenience and potential

✉ Hassan Ali
hassan.ali@uni-hamburg.de

[1] Knowledge Technology, Department of Informatics, University of Hamburg, Vogt-Kölln-Str. 30, Hamburg 22527, Germany

[2] Robotics Brain and Cognitive Sciences, IIT Central Research Labs Genova, Via Enrico Melen 83, Genova 16152, Italy

compatibility with low-resource systems, such as robots [5]. Also, they facilitate the reproduction of results, especially as reproducibility issues related to deep learning are getting more attention from the scientific community [6]. Despite the recent development triggered by the deep learning trend using networks like 3DCNN, ResNet, and Inception V3, dynamic gesture recognition is still a challenging task.

Some of the factors that influence vision-based approaches are *indistinctive* and *subtle* movements [7]. *Subtle* movements refer to the slight movement of the hand and fingers at the peak with no clear arm movement. *Indistinctive* movements mean that multiple gestures follow a very similar path of motion. One limitation prominent in various approaches is the reliance on the motion path only (the interested reader can find a good overview in [8]). Consequently, some techniques lack the consideration of hand details, which leads to misclassifications between gestures with similar motion properties. Therefore, it is worth inspecting whether integrating hand details into the classification system would refine the performance of such models.

In this study, we propose a modular RGB-based approach called *Snapture*. Our architecture is an extension of the CNNLSTM [9] network, which is robust at learning motion patterns but limited at capturing hand details. By integrating hand information in a hybrid architecture, our model aims to improve the performance of the CNNLSTM. Since each dataset imposes different challenges due to the unique gesture vocabulary, we evaluate our approach on multiple domains: robot commands and co-speech gestures. This study is organized as follows: we present our literature review on recent gesture recognition systems. Then, we describe the datasets and our proposed *Snapture* framework, including its various components. Next, we discuss the experiments carried out in this study. After comparing the performance of our model to a CNNLSTM baseline, we discuss our results and conclude with potential directions for future research.

Related Work

Recent work in dynamic gesture recognition uses various preprocessing techniques for motion representation which tend to lose the hand details. One such technique called *star RGB* was proposed by dos Santos et al. [10]. Each gesture sequence was divided into three parts corresponding to the *pre-stroke*, *stroke*, and *post-stroke* stages as defined by Kendon [11]. The algorithm generated a motion representation for each part, further merged using the frame's color channels. The data was fed into a feature extraction model using pre-trained ResNet50 and ResNet101 networks. The features were weighted using a soft-attention mechanism, while a final classification was accomplished using a two-layered feedforward network. Similar to our work, the authors evaluated their approach in both the robot command and co-speech gesture domains. The system achieved an accuracy of ~0.98 and ~0.95 on the GRIT [12] and Montalbano [13] datasets. However, the architecture is overly complex, especially considering that the GRIT dataset is limited to 543 samples. Despite that, the system encountered confusion between multiple gestures with the same motion since it did not consider the hand shape. Therefore, the results confirm that the problems of *indistinctive* and *subtle* gestures are not trivial. The authors hypothesized that these issues could be addressed by integrating hand information into the system.

The stated hypothesis is supported by the work of Wu et al. [14]. The authors demonstrated increased performance concerning gestures with a similar motion by fusing RGB, depth, and skeleton data. The proposed approach, called Deep Dynamic Neural Network (DDNN), consisted of three networks corresponding to each modality. RGB and depth information was fed into a 3DCNN, while skeleton data was passed through a Deep Belief Network (DBN). A Hidden Markov Model (HMM) was responsible for the temporal modeling of gestures using a set of defined states. Each observation was classified by calculating the most probable path using the Viterbi algorithm. The authors reported a score of 0.816 on the Montalbano [13] dataset using the Jaccard index. When considering RGB and depth modalities, the authors showcased less confusion regarding *implicit* Montalbano movements. Thus, the results hint at the importance of integrating RGB data to preserve the hand pose information. However, the model was computationally intensive and required long training times of 5 days. Thus, its robotic applications might be limited.

Mazhar et al. [15] utilized RGB data when integrating static and dynamic recognition in their system. The authors proposed a framework called StaDNet consisting of two Inception V3 CNNs. Each CNN extracted the spatial features of one of the two hands. The authors stated that by cropping the CNN input to the hand and removing the background, the framework could learn *subtle* movements. The temporal learning was carried out using an LSTM network. The framework scored an accuracy of 0.8675 and 0.989 on the Chalearn 2016 [16] and OpenSign [17] datasets. However, the method still relied on other modalities for training besides RGB, such as a 2D body skeleton and Kinect depth estimators. Another requirement of this model is the two datasets for training: one static and one dynamic. This requirement implies that the architecture learns each gesture vocabulary separately. Thus, the model does not seamlessly classify a gesture based on its dynamic and static characteristics. Therefore, achieving such integration between static and dynamic recognition remains an open question.

A recent study proposed a system using a transformer-based architecture [18]. The study employed a self-attention mechanism for the sequence modeling of data streams from multiple sensors. A ResNet18 network was responsible for frame-level feature extraction. A temporal function was implemented using a transformer module of six encoders, while the sequences were classified using a fully connected neural network. The system achieved an accuracy of 0.876 and 0.962 on the NVGestures [19] and Briareo [20] datasets. However, the system did not perform very well with RGB data. In contrast, the scores dropped to 0.765 and 0.906 with a single RGB modality. The authors experimented with different combinations of modalities using a late fusion technique. The best results were reported using a fusion of depth, surface normals, infrared, and RGB. Thus, the system's applications are limited to constrained environments with sensors placed close to the operator's hand. Furthermore, the system encountered confusion between symmetric gestures, which illustrates the challenge of sequence modeling of dynamic gesture sequences.

Aditya et al. [21] used attention for modeling sequences in a multi-feature setup to classify continuous sign language. The architecture contained spatial and temporal components. The spatial module performed feature extraction on two channels: RGB and key points encoding the body and hand poses. The temporal part consisted mainly of convolution and pooling layers. Sequence modeling was supported by a self-attention mechanism, avoiding blurry frames in the input. However, a following bidirectional LSTM with CTC was needed to interpret the signs and align predicted words into sentences. Using a variety of configurations for attention, the authors found the best performance when attention followed temporal pooling. The system achieved a word error rate (WER) of 0.7 and 21.5 on the CSL [22–24] and RWTH-PHOENIX [25] datasets, respectively. However, the error rate increased to 31.2 with the RWTH-PHOENIX dataset when pooling was dropped. Thus, the approach only performed well in capturing short-term dependencies and fell short of modeling longer sequences.

Another approach for dealing with noisy frames was suggested by Cao et al. [26]. The architecture used a self-attention and transformer encoder module for input representation. A feedforward network was used for classification. However, they implemented an additional temporal sampling method to identify the meaningful frames. Using a sliding window technique, they performed a gesture detection step using a single-shot detector to identify whether the hand was present. Then, frames were sampled using step size and length values that were empirically found. Using the NVGestures [19] and EgoGesture [27] datasets, the approach achieved the following accuracy values: 0.807 and 0.926. The sampling technique was found to improve the performance to 0.832 and 0.938 on the NVGestures and EgoGesture datasets, respectively. However, the used datasets contain a mixture of emblems and control commands. Thus, such sampling using hand detection would be limited concerning co-speech gestures, for which the hand would need to be continuously present in all frames.

Similarly, Chen et al. [28] proposed a dynamic gesture recognition system based on an attention mechanism. Two networks, R2plus1D and ConvLSTM, extracted the long-term and short-term temporal features of gesture sequences. The relevance of the extracted features for a given sequence was learned by an RPCNet network. The approach accommodated the contextual information across channels by weighing the contribution of its temporal channels. Multiple experiments were conducted: using a channel fusion component, sequential channel and spatial components, and parallel channel and spatial components. The channel fusion as the first module produced the best accuracy (0.9353) on the EgoGesture [27] dataset. Using average and max pooling of the channel and spatial components further boosted the score to 0.9393. However, the accuracy boost was not significant compared to a base R2plus1D model (0.9276) that did not use attention. The authors also reported an accuracy of 0.997 and 0.693 on the SKIG [29] and IsoGD [16] datasets, respectively. However, they did not provide a comparison to the R2plus1D model using these two datasets. Therefore, the influence of attention on the overall architecture is not fully unveiled.

Tsironi et al. [9] proposed a recurrent network for the motion learning of robot commands. A step of hand segmentation was done using a pre-processing algorithm called the *differential image*. The processed data passed through a CNNLSTM architecture, which performed the hand's implicit feature extraction and motion tracking. Gestures with similar movements were challenging to the system due to the loss of hand details. In particular, the system confused gestures like "hello" and "no." Since the motion similarity was limited to a small subset of gesture classes, it did not overly influence the model's performance. The framework still achieved an accuracy of ~0.92 using the GRIT [12] dataset. However, this dataset is small-scale (543 samples), as previously mentioned. It also consists of only robot commands designed with distinct arm movements. Thus, the performance of such an approach remains unknown on natural gestures with less intense arm movement and more focus on the hand, such as co-speech gestures.

Proposed Model

As mentioned in our literature review, using multiple benchmarks can provide an insightful assessment of the system's performance. Therefore, we evaluate our architecture on multiple gesture domains: robot commands and co-speech gestures. In this section, we present the two datasets used to evaluate our framework. We also introduce our proposed *Snapture* architecture for hybrid gesture recognition and show our method for motion profile analysis of gesture sequences.

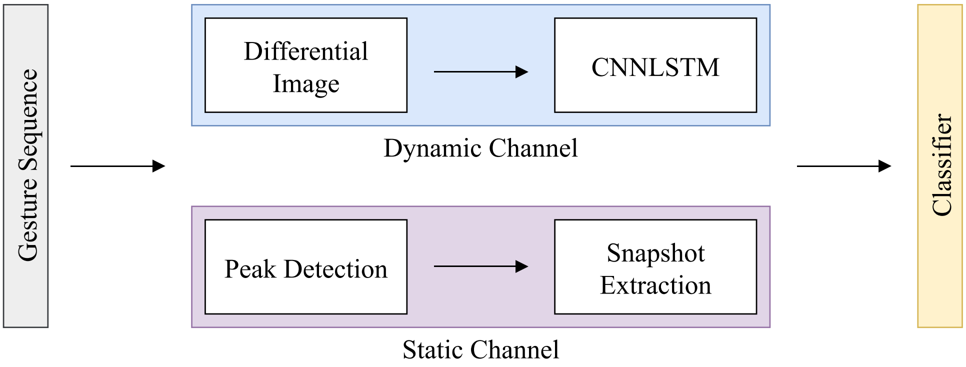


Fig. 1 An overview of the *Snapture* framework. The architecture consists of dynamic and static channels, fused into a final classifier. Thus, it performs a hybrid hand gesture recognition task

The GRIT Robot Commands Dataset

In the context of robot commands, the “Gesture commands for Robot InTeraction" (GRIT)[1] [12] is one of the few publicly available dynamic gesture datasets. The corpus contains 543 isolated gestures distributed over nine gesture classes and recorded with six subjects. Each gesture has a distinct arm movement. An exception is the case of classes “hello” and “no,” which are *indistinctive* movements. The dataset was collected under lab-controlled settings with a plain white background and no surrounding noise.

The Montalbano V1 Co-Speech Dataset

The Montalbano dataset contains co-speech gestures and is publicly available. It was collected with about 50 participants as part of the *ChaLearn*[2] *Looking at People challenge* [13]. It contains around 14,000 Italian gestures spreading over 20 gesture classes. Each recorded video contains a subject in various indoor environments with noisy backgrounds. The Montalbano dataset provides multiple sensory data. However, we only use RGB data due to the advantages of vision-based systems, such as reproducibility and portability. Since the gestures are continuous with little to no pause, we convert them into isolated gestures by identifying the start and end of each movement. We make the annotations created for isolating the sequences and source code of the experiments presented in our work publicly available.[3]

Snapture—Hybrid Gesture Recognition

Our architecture consists of two main components: a dynamic channel for capturing the gesture’s movement and a static channel for the hand pose. A classifier is trained using the combined output of the channels. We analyze the motion profiles of the gesture sequences, which are utilized when extracting the hand pose. Our approach can be extended with an optional component for controlling the static channel to address the issue of blur in the frames. We present the details of the mentioned components in the following subsections. A simplified overview of our so-called SNAPshot capTURE (*Snapture*) architecture is shown in Fig. 1.

Motion Profile Analysis

Our approach for tackling the issues of *indistinctive* and *subtle* movements relies on fusing the hand motion and pose. We extract motion features by exploiting the temporal information across consecutive frames. The hand pose is interesting at the *stroke* phase of co-speech gestures, as described by Kendon [11]. However, the temporal relationship between frames and *stroke* in the studied datasets is unclear. Therefore, we analyze the gesture sequences in terms of motion and pause. Due to the lack of approaches for gesture analysis, we utilize the structural similarity index measure (SSIM) [30] as a metric for the similarity between consecutive frames. The SSIM calculation is shown in Eq. (1).

$$\begin{aligned} SSIM(x,y) &= \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}\,, \\ \sigma_{xy} &= \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y)\,, \\ C_1 &= (K_1L)^2\,, \\ C_2 &= (K_2L)^2\,, \end{aligned} \tag{1}$$

where x and y are spatially local windows of the input frames. μ_x and μ_y represent the mean intensity of x and y, respectively. Similarly, σ_x and σ_y denote the standard deviation. N is the number of pixels, while x_i and y_i represent the pixel values at index i of x and y, respectively. C_1 and C_2 are stability constants to avoid a division by zero. K_1 and K_2 are positive values much smaller than 1, while L is the pixel range (255 for 8-bit grayscale frames).

[1] https://www.inf.uni-hamburg.de/en/inst/ab/wtm/research/corpora.html

[2] http://chalearnlap.cvc.uab.es/

[3] https://github.com/hassanali-90/snapture/

Using the first frame as a reference, we can quantify the amount of motion and pause across the gesture time span. By inverting the equation, we can express change across frames. We express that in Eq. (2) and refer to it as the Inverted SSIM (ISSIM).

$$ISSIM = 1 - SSIM(I_i, I_0)\,, \tag{2}$$

where I_i and I_0 denote the grayscale frames at time steps i and 0, respectively.

We observe two variations of movements in the GRIT dataset based on our analysis. *Paused* gestures include a pronounced period of pause around the gesture peak. For example, "turn left" (cf. Fig. 2a) lacks motion around the peak since participants hold their hand briefly still. In contrast, in gestures, such as "turn" (cf. Fig. 2b), subjects continuously repeat a circular pattern. We refer to these movements as *repeating pattern* gestures. These unique characteristics of motion and pause of each gesture influence the design of our approach, as will be discussed later. In contrast to the GRIT dataset, the Montalbano gestures follow Kendon [11] model of gesticulation and concurrent speech. The intensity of the movement starts and ends gradually, with a clear peak in between, ex: gesture "ok" (cf. Fig. 3).

Dynamic Channel

The dynamic channel of *Snapture* is a CNNLSTM [9] implementation using PyTorch.[4] The network consists of a two-layer stacked convolutional neural network (CNN) followed by a long short-term memory (LSTM) network (cf. Fig. 4). The input to the network represents segmented gestures. The segmentation is done using the *differential image* algorithm. In Eq. (3), we show the algorithm's calculation as described by Tsironi et al. [9]. This algorithm operates on three subsequent frames subtracting every two consecutive frames of a three-frame sequence. The moving hand is extracted in the input by applying a bitwise AND operator to the output of the two subtraction operations.

$$\Delta_i = (I_i - I_{i-1}) \wedge (I_{i+1} - I_i)\,, \tag{3}$$

where Δ_i and Δ_{i-1} are the segmented gesture input frames at the current and previous time steps, respectively. I_{i-1}, I_i, and I_{i+1} denote the grayscale frames at time steps $i-1$, i, and $i+1$, respectively. $\wedge$ is the bitwise AND operator.

The stacked convolution layers have five and ten kernels of size 11×11 and 6×6, respectively. These are the same kernel size and number of filters of the CNN as the original CNN-LSTM model [9]. Each layer has a 1×1 stride, zero-padded input, and a hyperbolic tangent (Tanh) activation function. A max-pooling layer of size 2×2 follows each convolution layer. Following each convolution, batch normalization is used to reduce internal covariate shift [31] and speed up the training. We initialize the CNN's weights with values from a uniform distribution [32]. The output of the last convolution layer is flattened and propagated through the LSTM.

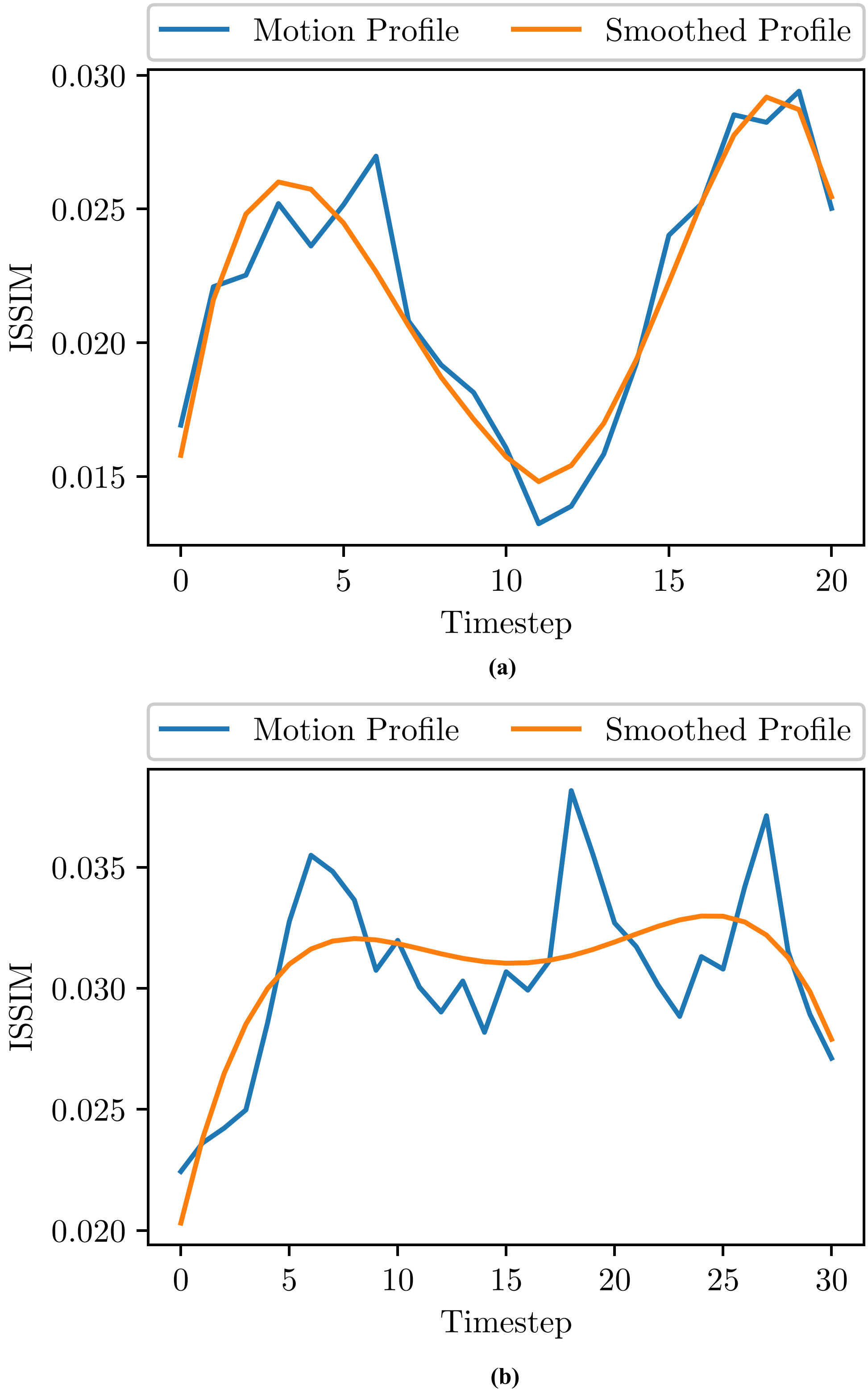
Since the input represents isolated gestures, each mini-batch has all the information needed for the network to produce a classification. Therefore, we opt to use a stateless LSTM. We configure the LSTM's cell state to produce a sequence-level classification for each gesture. Therefore, our model requires no additional post-processing steps and fits the concept of capturing a *snapshot* more intuitively. The LSTM's numbers of layers and neurons are selected using grid search. The optimal number of layers is 2 out of 1, 2, 4, and 8. The optimal number of neurons has resulted differently for the GRIT and Montalbano datasets. We choose 64 and 512 neurons for the GRIT and Montalbano datasets, respectively. We initialize the LSTM with weights from a uniform distribution with zero bias. After passing through dropout [33], the output of the LSTM is fused through concatenation with the static channel's output (explained in the next subsection). The combined outputs are propagated into a two-layered feedforward network followed by softmax, producing a probability distribution over the gesture classes.

Static Channel

This channel is responsible for capturing the specific hand shape and finger arrangements through a so-called *snapshot* at the gesture's peak. We detect and extract the gesture at the peak corresponding to the *stroke* phase. This provides hand pose information, which we fuse with the dynamic channel. As a result, our method integrates the characteristics of static and dynamic recognition systems.

Gesture Peak Detection According to Kendon [11] model of the relationship between gestures and concurrent speech, human gestures are described by five phases (cf. Fig. 5). Gestures start with a *rest phase*, representing a neutral position of the arms. In the *pre-stroke* or *preparation phase*, a gradual intensity in motion of one or both arms starts to unveil. Next is the *stroke* phase in which the static gesture properties, i.e., hand shape and finger configurations, completely unfold. These characteristics start to fade away in the *post-stroke* or *retraction* phase as the intensity of motion gradually decreases. The gesture ends again with a rest phase. The Montalbano gestures have a clear peak through the frames around the midpoint of the gesture sequence. Similar time steps are occupied by a pronounced pause in *paused* gestures (cf. "Motion Profile Analysis" section). Therefore, we define the peak as the frame in the middle of the gesture sequence.

[4] https://pytorch.org/



(a)

(b)

Fig. 2 The motion profile of GRIT gestures “turn left” (**a**) and “turn” (**b**). “turn left” is *paused* at the peak, while “turn” is with a *repeating pattern* due to the continuous intensity across its time span

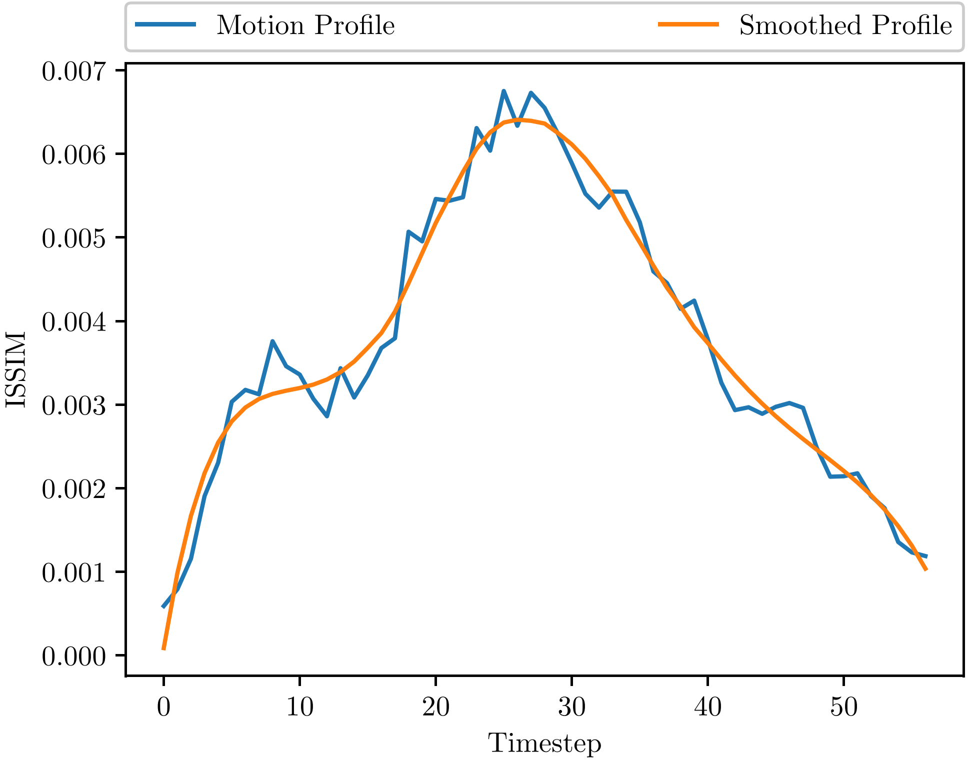


Fig. 3 The motion profile of the Montalbano gesture “ok.” It starts and ends with low intensity and has a clear peak around the midpoint of the timeline

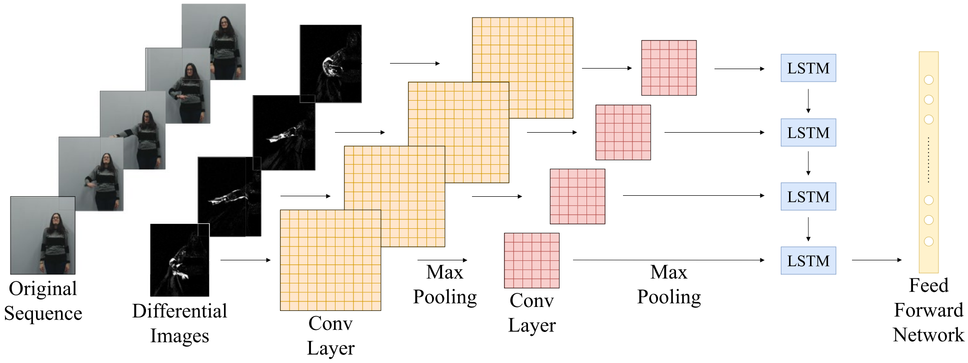


Fig. 4 The dynamic channel of *Snapture* is a CNNLSTM network consisting of two layers of CNN followed by an LSTM and a feedforward network. The input is pre-segmented using the *differential image* algorithm. For clarity, we show only five frames and increase the contrast of the differential images

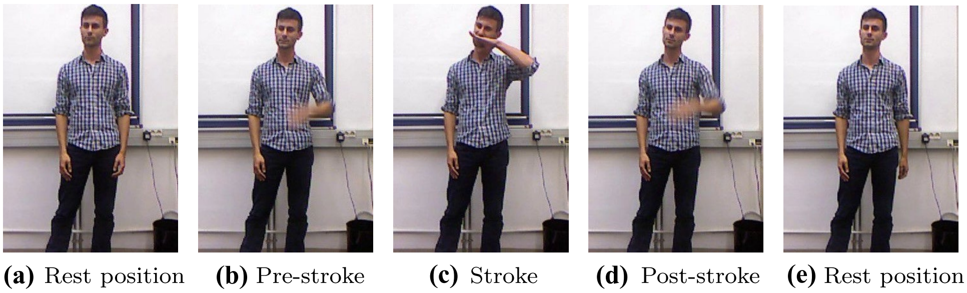


Fig. 5 The five gesture phases, according to Kendon [11]. Each gesture starts with a *rest phase*. In *pre-stroke*, the limb moves from the rest position into the *stroke* phase. The *stroke* phase contains the most expressive information. In post-stroke, the limb moves away from *stroke* back into *rest phase*

Gesture Peak Extraction We follow a skin detection technique to extract the hand from the frame. Our implementation uses Python and OpenCV.[5] First, the face is detected and removed because the input contains a full body image, and skin detection treats all visible skin equally. Next, the hand is segmented by converting into the orthogonal color space *YCbCr* [34]: *Y* representing the luminance, while *Cb*

[5] https://opencv.org/opencv-4-5-3/

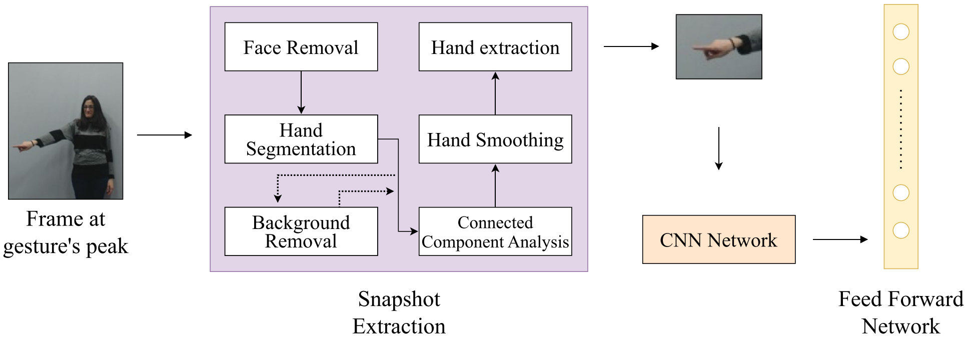


Fig. 6 The gesture peak extraction module of the *Snapture* approach. Using a skin detection technique, the hand shape and finger configurations are extracted from a target frame at the gesture's peak. Background removal is only applied to the Montalbano gestures (dotted line). The extracted hand is passed through a CNN and a feedforward network

and *Cr* indicate the chromaticity. This is done to avoid the high correlation between luminance, hue, and saturation in RGB [35]. Since various lighting conditions highly influence skin tones, we apply the threshold on chrominance only. We use the thresholds *Cb*=[80, 120] and *Cr*=[133, 173] proposed by Basilio et al. [36]. According to the authors, these threshold values are independent of skin tone. An additional step of background removal is applied to the Montalbano data using simple background subtraction. This is due to the complex surroundings, unlike GRIT. Next, we apply the connected component analysis, which describes the *YCbCr* mask in terms of BLOBs. These objects are then sorted by size and position. Due to the noisy background in the Montalbano dataset, we filter out objects that do not belong to the foreground, calculated in the step of background removal.

We pick the higher object in the frame to avoid assumptions about the subject's dominant hand. As we observe in the data, the hand performing the gesture is always in an upper position. The other hand is usually at rest or slightly raised. For gestures requiring two hands, both hands always make the same pose. Therefore, our algorithm has the freedom of picking up either hand in this case. A step of hand smoothing is applied using erosion and dilation morphological transformations. However, omitting this step does not influence the algorithm's output. Finally, an area around the detected hand is extracted from the original frame and resized to 64×48 pixels matching the CNN input. The configuration of the CNN network in the static channel is similar to the dynamic channel (cf. "Dynamic Channel" section). The hand features learned through this network are flattened and concatenated with the dynamic channel's output before feeding into the two-layered feedforward network. The gesture peak extraction module is depicted in Fig. 6.

Static Channel Control

One challenge when using RGB data is capturing hand details during rapid hand movements. This is caused by factors such as lower camera resolution and exposure time. Consequently, it leads to a blurry hand in the frame (cf. Fig. 7). This is pronounced for *repeating pattern* robot commands due to the intense movement. This phenomenon is a challenge to any vision-based approach due to the missing information in the input frames. However, we address it by regulating the static channel based on the amount of motion contained in a gesture. More precisely, we integrate the extracted static information only if the amount of motion lies below a threshold., i.e., the *stroke* phase contains a pause sufficient for the *snapshot* extraction.

We use the SSIM-based method (cf. "Motion Profile Analysis" section) as a quantitative metric for the amount of motion and pause. We split each gesture into three parts:

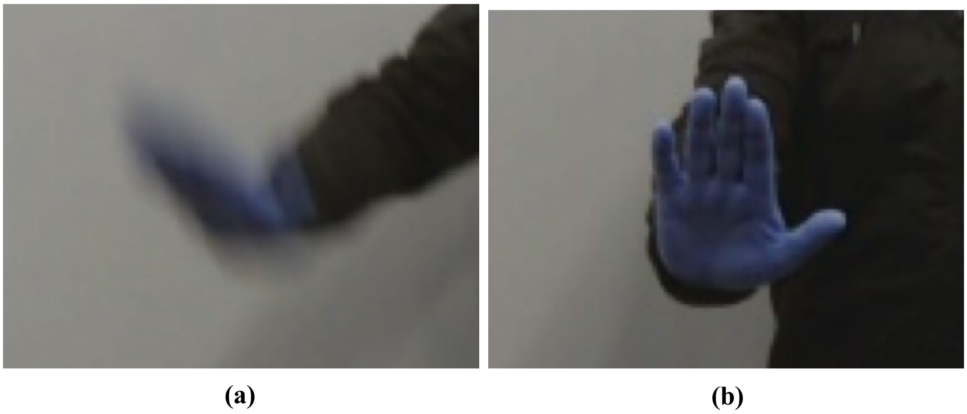


Fig. 7 *Repeating pattern* gestures, e.g., "circle" (**a**), contain a *blur* at the peak compared to *paused* movements, e.g., "stop" (**b**). The blurry hand at the gesture's peak for highly dynamic movements is challenging for RGB-based approaches. We bypass this issue by regulating the static channel of our approach

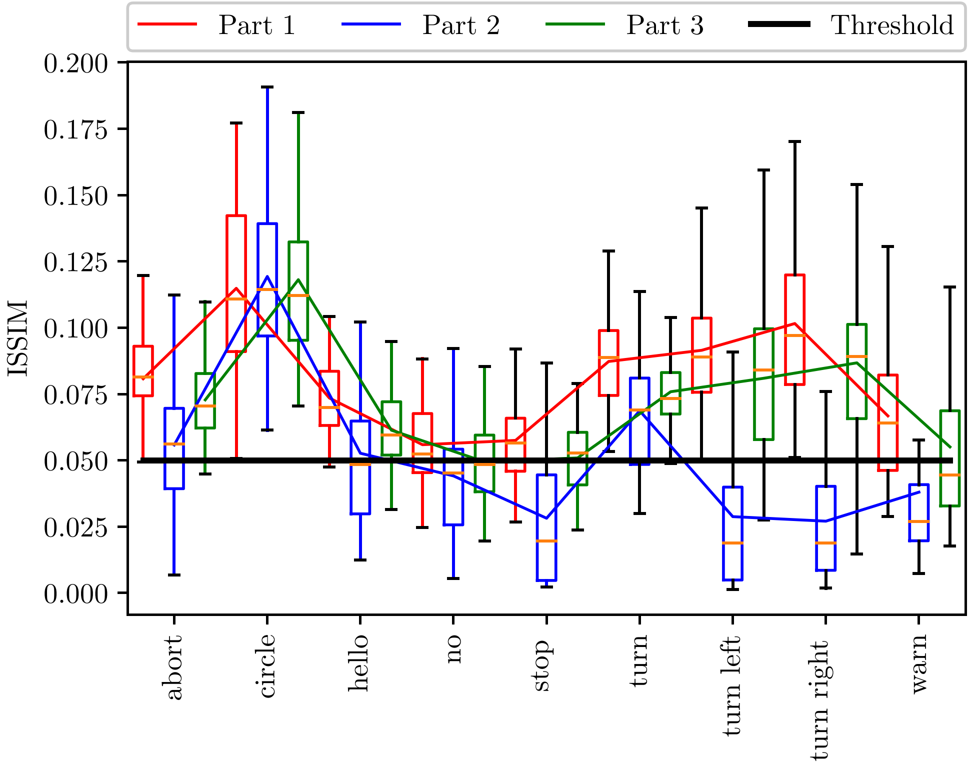


Fig. 8 Our motion analysis of the GRIT dataset after splitting gestures into three parts: *rest* to *pre-stroke* phases, *pre-stroke* to *post-stroke* phases, and *post-stroke* to *rest* phases. The second part contains more pause and facilitates capturing a *snapshot*. The black line denotes our defined threshold for regulating the static channel

(1) the first part represents all the frames in the *rest* and *pre-stroke* phases, (2) the second part contains the frames in *pre-stroke* and *post-stroke* phases, and (3) the third part consists of all the frames consecutively from *post-stroke* to *rest* phases. We assume the three parts to be of equal length for simplicity. The three parts and our defined threshold are visualized for the GRIT (c.f. Fig. 8) and Montalbano (c.f. Fig. 9) datasets. The average amount of motion in part 2 is less than in part 1 and part 3, which supports our choice of Kendon [11] *stroke* phase as the gesture's peak. It is also noticeable that most samples of *paused* gestures, such as "stop," "turn left," and "turn right," lie well below the threshold due to their pronounced period of pause. In contrast, the intensity of motion is high for *repeating pattern* gestures, e.g., "circle" (cf. Fig. 8), which corresponds with our definition of *repeating pattern* gestures (cf. "Motion Profile Analysis" section). On the other hand, most Montalbano gesture classes contain pause facilitating capturing a *snapshot*.

Experimental Procedure

In this section, we present the experiments carried out in this study. In each experiment, we evaluate and compare the following: (1) a CNNLSTM acting as a baseline for comparison, (2) our *Snapture* architecture, which predicts a class by integrating the hand shape and motion, and (3) *Snapture* with the threshold-controlled mechanism for regulating the static channel based on the sufficiency of pause to capture a *snapshot*. We will refer to this model as $Snapture_{thold}$. The purpose is to evaluate the influence of *subtle* and *indistinctive* gestures on the performance of each of the models in two gesture domains, as motivated earlier.

Experimental Settings

The training parameters of each experiment are selected using grid search and are listed in the following subsections. We run each of the models under similar conditions. The hardware specifications used for training and testing are as follows: (1) Ubuntu 18.04.5 LTS operating system; (2) Intel Core i7-4930K 3.40 GHz with six cores; (3) 8 GB of RAM; (4) NVIDIA GeForce GTX 1080 graphics card with 8 GB of memory. The performance of each model is evaluated using accuracy, F1-score, and training time metrics. We report the average performance of each model over five trials. In each trial, we repeat the steps of training and testing. We analyze the classification behavior using the confusion matrices.

GRIT Experiment

The search space and optimal hyperparameters for the experiment on the GRIT dataset are listed in Table 1.

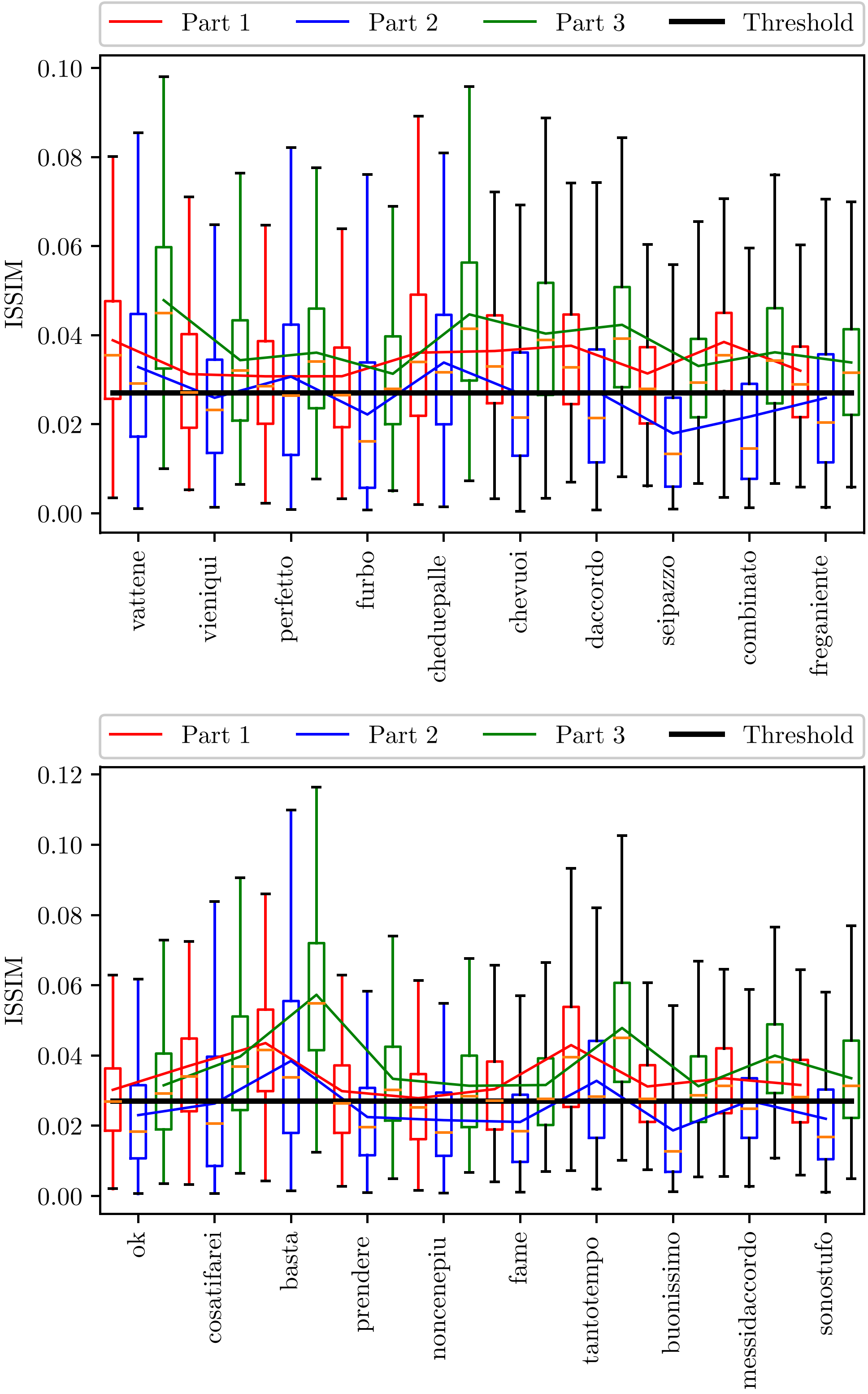


Fig. 9 Our motion analysis of the Montalbano dataset after splitting gestures into three parts: *rest* to *pre-stroke* phases, *pre-stroke* to *post-stroke* phases, and *post-stroke* to *rest* phases. Similar to GRIT, the second part contains more pause that facilitates capturing a *snapshot*. Our defined threshold for regulating the static channel is denoted by the black line

We choose the parameters using grid search and cross-validation while leaving out 30% of the data for testing. In this experiment, the optimal values are identical, which we explain by the similarity in architecture and training procedure across the models. We use the same data split ratio for each trained model to conduct a fair comparison. To avoid data imbalances, we use stratified sampling regarding class labels.

Table 1 The search space and optimal hyperparameter values (in bold) of each model in the GRIT experiment

Hyperparameter	CNNLSTM	Snapture*
Learning rate	[0.01, **0.001**, 0.0001]	[0.01, **0.001**, 0.0001]
Number of epochs	[10, 20, **40**]	[10, 20, **40**]
Mini-batch size	[16, 32, **64**, 128]	[16, 32, **64**, 128]
Optimizer	[**Adam**, SGD]	[**Adam**, SGD]

*Similar for Snapture$_{thold}$

Montalbano Experiment

Similar to the previous experiment, we report the hyperparameters in Table 2. Due to the considerable number of class labels, the search space is extended compared to the GRIT experiment. We choose the parameters using grid search and cross-validation while leaving out unseen data for testing. The dataset is part of the *ChaLearn Looking at People* challenge and is already split into training and test datasets. Each set contains unique subjects. To avoid any influence of subject variability, we implement our split with data from all participants. We follow this approach since we focus on comparing the classification behavior of the different models rather than taking part in the challenge. Our split consists of 70% and 30% of randomly selected data for training and testing, respectively. Stratified sampling is utilized for an approximately uniform distribution of class labels across the sets.

Results

In this section, we present the results of the experiments acquired using the GRIT and Montalbano datasets and under the experimental settings described earlier.

Results of the GRIT Experiment

The experiment results are summarized in Table 3. Our *Snapture* approach achieves slightly superior results compared to the CNNLSTM in terms of accuracy and F1-score.

Table 2 The search space and optimal hyperparameter values (in bold) of each model in the Montalbano experiment

Hyperparameter	CNNLSTM	Snapture*
Learning rate	[0.01, **0.001**, 0.0001]	[0.01, **0.001**, 0.0001]
Number of epochs	[20, 40, 60, 80, **100**]	[20, 40, 60, 80, **100**]
Mini-batch size	[16, 32, **64**, 128]	[16, 32, 64, **128**]
Optimizer	[**Adam**, SGD]	[**Adam**, SGD]

*Similar for Snapture$_{thold}$

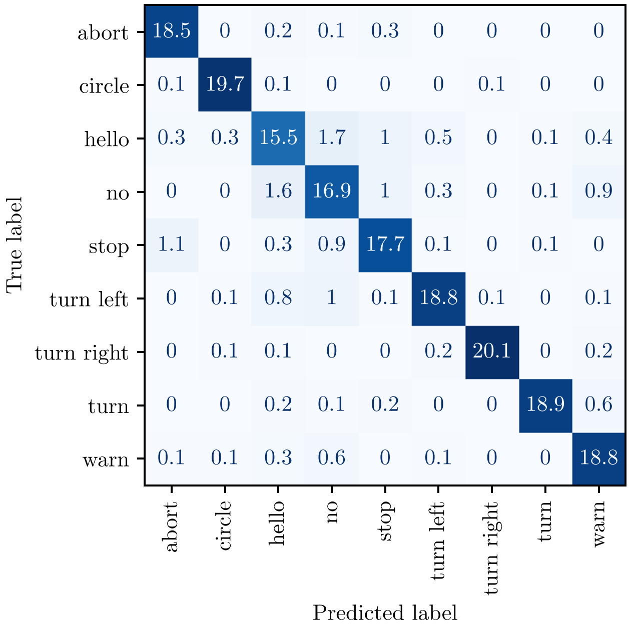
Table 3 The results of the GRIT experiment under the described settings. The reported metrics represent the mean of five trials, while the values in parentheses correspond to the standard deviation. The superior accuracy and F1-score values are in bold

Model	CNNLSTM	Snapture	Snapture$_{\textbf{thold}}$
Accuracy	0.91 (0.012)	0.924 (0.006)	**0.926** (0.008)
F1-score	0.913 (0.012)	**0.927** (0.005)	0.913 (0.012)
Time*	140.612 (0.255)	170.012 (1.027)	125.156 (1.117)

*In seconds

The scores across the *Snapture* and *Snapture$_{thold}$* variations are similar. The three models have a slight deviation across the five trials. We explain the marginal accuracy boost by three factors. First, GRIT robot commands have unique movement paths. Therefore, the CNNLSTM model is sufficient due to its motion-learning capabilities. Second, due to the *repeating pattern* gestures, most GRIT movements do not have sufficient pauses for capturing a *snapshot* (approximately 44%) based on our threshold definition. Combined with the small dataset size, our model may not have seen enough training data to learn the unique characteristics of hand shapes. Third, only approximately 44% of GRIT samples include a motion at the peak beneath the defined threshold. Therefore, the *Snapture$_{thold}$* acts similarly to a CNNLSTM model in 56% of the cases, and it is not able to contribute to a noticeable accuracy increase.

However, we analyze the results further through the confusion matrix of the average case (cf. Fig. 10), which



True label \ Predicted label	abort	circle	hello	no	stop	turn left	turn right	turn	warn
abort	18.5	0	0.2	0.1	0.3	0	0	0	0
circle	0.1	19.7	0.1	0	0	0	0.1	0	0
hello	0.3	0.3	15.5	1.7	1	0.5	0	0.1	0.4
no	0	0	1.6	16.9	1	0.3	0	0.1	0.9
stop	1.1	0	0.3	0.9	17.7	0.1	0	0.1	0
turn left	0	0.1	0.8	1	0.1	18.8	0.1	0	0.1
turn right	0	0.1	0.1	0	0	0.2	20.1	0	0.2
turn	0	0	0.2	0.1	0.2	0	0	18.9	0.6
warn	0.1	0.1	0.3	0.6	0	0.1	0	0	18.8

Fig. 10 The confusion matrix of the average case for the CNNLSTM on the GRIT dataset. The confusion is pronounced between the classes “hello” and “no,” “hello” and “stop,” “no” and “stop”

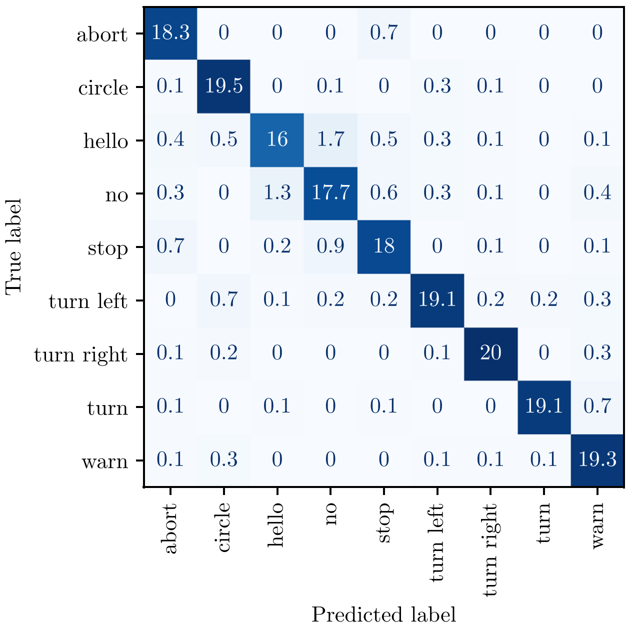


Fig. 11 The confusion matrix of the average case for *Snapture* on the GRIT dataset. The confusion is less pronounced between the classes "hello" and "no," "hello" and "stop," "no" and "stop." However, the performance is still negatively influenced by the false classification of some "hello" samples as "no"

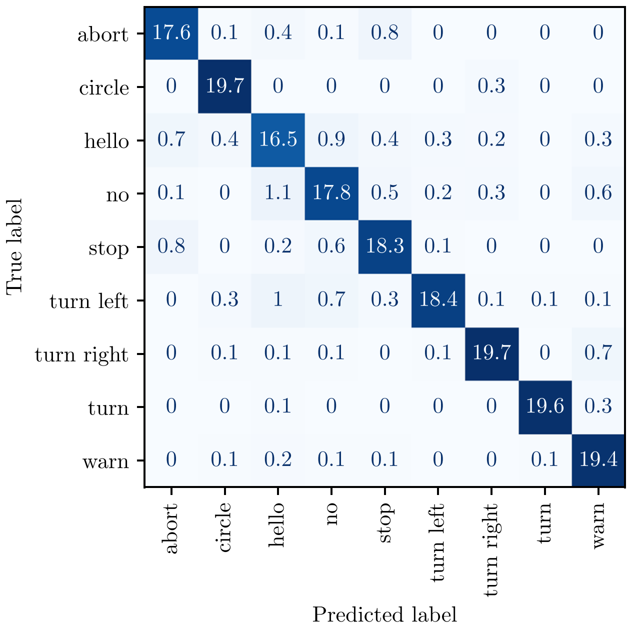


Fig. 12 The confusion matrix of the average case for *Snapture*$_{thold}$ on the GRIT dataset. Less confusion can be observed concerning class "circle," which confirms that the static channel should be disabled for such *repeating pattern* movements

is calculated using the average of predicted labels over all trials. The most confusion in the CNNLSTM model occurs between "hello" and "no," "hello" and "stop," "no" and "stop," and "stop" and "abort." These movements have a similar motion profile but differ in hand shape. Thus, this supports that *indistinctive* movements negatively influence the performance of CNNLSTM. On the other hand, the confusion between these classes is less pronounced in *Snapture* (cf. Fig. 11) due to the additional hand pose information. However, the misclassification of "hello" samples as "no" still negatively impacts the performance of *Snapture*. We observe that some participants perform "hello" and "no" rapidly, resulting in a *blur* effect and noisy input to the network. Therefore, the *Snapture*$_{thold}$ improves the situation (cf. Fig. 12) by excluding the *snapshot* in case the frame is not clear for interpreting the hand details. On the other hand, *repeating pattern* movements, e.g., "circle" yields comparable F1-score values across the three architectures (cf. Fig. 13) due to the distinctive movement. However, the number of false positives and true negatives associated with "circle" drops noticeably in *Snapture*$_{thold}$, further emphasizing that the static channel is indeed counterproductive for such movements.

On a different note, the confusion between the classes "no" and "stop" is less pronounced in *Snapture* and *Snapture*$_{thold}$ compared to CNNLSTM. Despite the dissimilarity between the two classes, some subjects tend to perform "no" with a slight left and right hand movement around the wrist, making it very similar to "stop" in terms of arm movement (raised and directed towards the camera). The CNNLSTM struggles with this sort of *implicit* hand movements due to the loss of hand details. Therefore, our approach improves performance with the static channel.

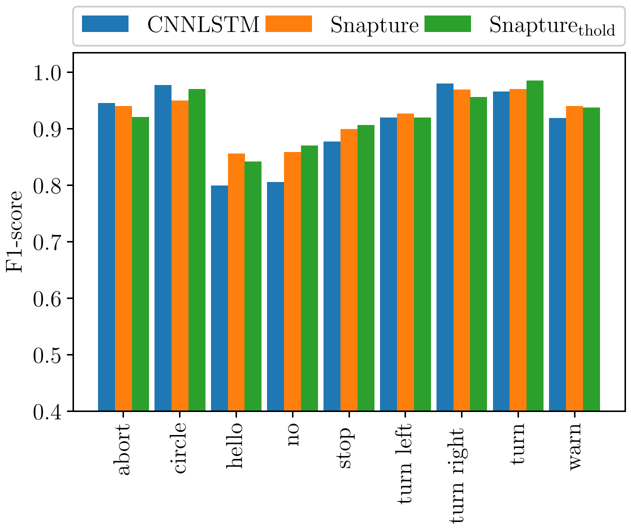


Fig. 13 A comparison of per-class F1-score values between the different approaches on the GRIT dataset. *Snapture* increases the score for classes "hello" and "no," while the performance across the remaining classes is comparable

Table 4 The results of the Montalbano experiment under the described settings. The reported metrics represent the mean of five trials, while the values in parentheses correspond to the standard deviation. The superior accuracy and F1-score values are in bold

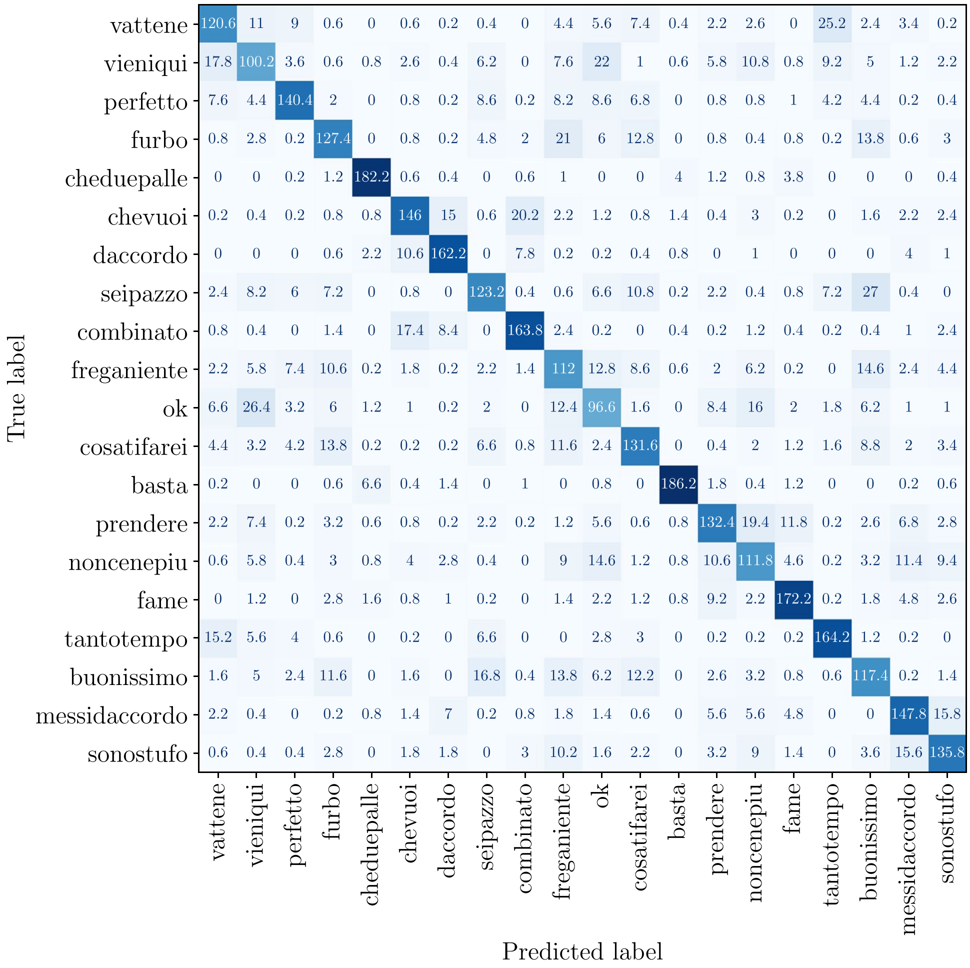
Model	CNNLSTM	Snapture	Snapture$_{\mathbf{thold}}$
Accuracy	0.699 (0.014)	0.755 (0.021)	**0.77** (0.008)
F1-score	0.701 (0.013)	0.752 (0.021)	**0.772** (0.007)
Time[*]	234.762 (0.115)	318.578 (0.428)	744.953 (0.724)

[*] In minutes

Results of the Montalbano Experiment

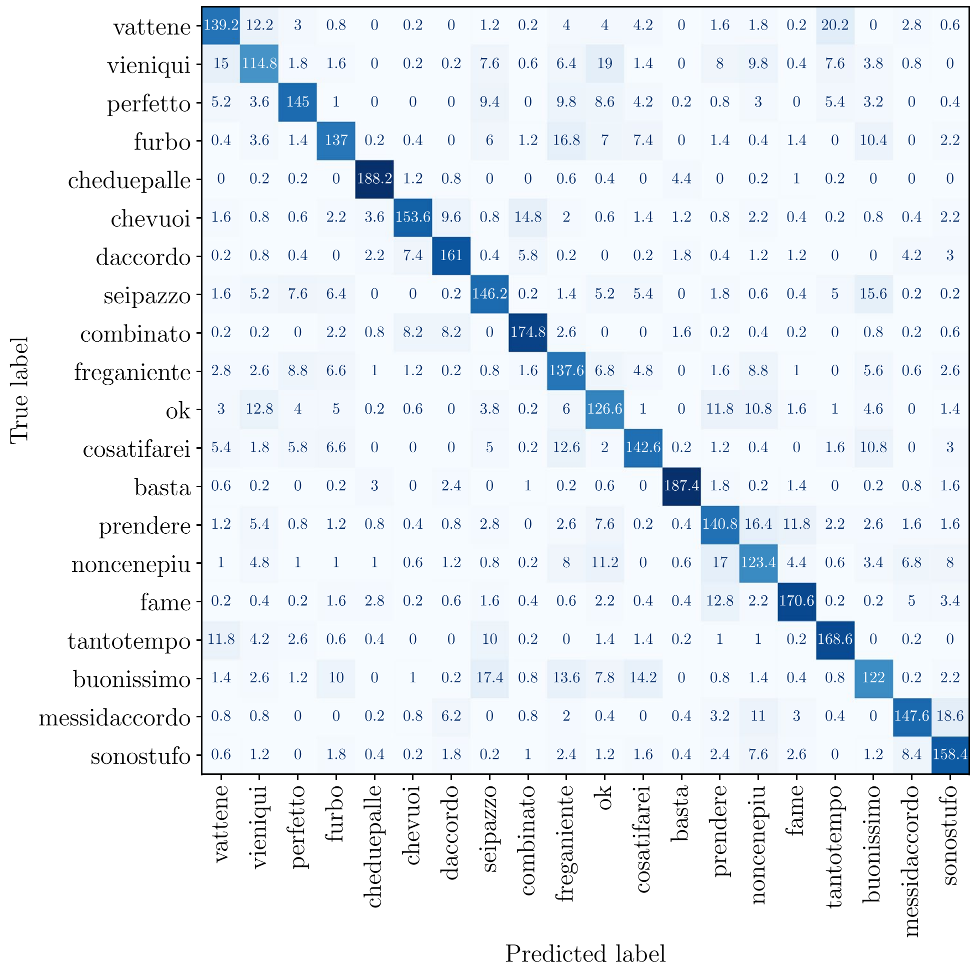
Our *Snapture* approach scores superior accuracy and F1-score compared to CNNLSTM. Also, the *Snapture$_{thold}$* improves the results even further (cf. Table 4). However, we observe a noticeable time increase in *Snapture$_{thold}$*. We explain that by the additional check for each sample to identify where it lies compared to the defined threshold. Approximately 70% of the Montalbano data contain a sufficient pause for a *snapshot*. Thus, it gives more insights into the performance of the *Snapture$_{thold}$* approach. By observing per-class performance, *Snapture* achieves superior per-class F1-scores compared to the CNNLSTM except for "basta" (both models achieve an identical score). Nonetheless, we report a boost in F1-score on all classes with the *Snapture$_{thold}$*.

Indistinctive Movements In CNNLSTM, multiple observations of classes "vattene" are miscalssified as "vieniqui," "perfetto," or "tantotempo" (cf. Fig. 14). We explain that by the similarity in hand motion. Compared to the CNNLSTM, an addition of ~19 and ~32 samples on average are correctly classified by the *Snapture* and *Snapture$_{thold}$*, respectively (cf. Figs. 15 and 16). Consequently, we observe F1-score improvements in the respective classes (cf. Fig. 17). For classes "vieniqui," "freganiente," "ok," "noncenepiu," and "buonissimo," the CNNLSTM achieves poor F1-score values (below 0.6). Most of the confusion of class "ok" is tied to false positives/negatives with one of the said classes. We explain that by the similarity in their motion. However, the total number of misclassified "ok" samples drops in *Snapture* and *Snapture$_{thold}$* by approximately 30. Therefore, we observe an increase in the F1-score. *Snapture* and *Snapture$_{thold}$* also enhance the F1-score of class "seipazzo." An additional average of ~23 and ~25 samples are correctly classified due to less confusion with "buonissimo."



	vattene	vieniqui	perfetto	furbo	cheduepalle	chevuoi	daccordo	seipazzo	combinato	freganiente	ok	cosatifarei	basta	prendere	noncenepiu	fame	tantotempo	buonissimo	messidaccordo	sonostufo
vattene	120.6	11	9	0.6	0	0.6	0.2	0.4	0	4.4	5.6	7.4	0.4	2.2	2.6	0	25.2	2.4	3.4	0.2
vieniqui	17.8	100.2	3.6	0.6	0.8	2.6	0.4	6.2	0	7.6	22	1	0.6	5.8	10.8	0.8	9.2	5	1.2	2.2
perfetto	7.6	4.4	140.4	2	0	0.8	0.2	8.6	0.2	8.2	8.6	6.8	0	0.8	0.8	1	4.2	4.4	0.2	0.4
furbo	0.8	2.8	0.2	127.4	0	0.8	0.2	4.8	2	21	6	12.8	0	0.8	0.4	0.8	0.2	13.8	0.6	3
cheduepalle	0	0	0.2	1.2	182.2	0.6	0.4	0	0.6	1	0	0	4	1.2	0.8	3.8	0	0	0	0.4
chevuoi	0.2	0.4	0.2	0.8	0.8	146	15	0.6	20.2	2.2	1.2	0.8	1.4	0.4	3	0.2	0	1.6	2.2	2.4
daccordo	0	0	0	0.6	2.2	10.6	162.2	0	7.8	0.2	0.2	0.4	0.8	0	1	0	0	0	4	1
seipazzo	2.4	8.2	6	7.2	0	0.8	0	123.2	0.4	0.6	6.6	10.8	0.2	2.2	0.4	0.8	7.2	27	0.4	0
combinato	0.8	0.4	0	1.4	0	17.4	8.4	0	163.8	2.4	0.2	0	0.4	0.2	1.2	0.4	0.2	0.4	1	2.4
freganiente	2.2	5.8	7.4	10.6	0.2	1.8	0.2	2.2	1.4	112	12.8	8.6	0.6	2	6.2	0.2	0	14.6	2.4	4.4
ok	6.6	26.4	3.2	6	1.2	1	0.2	2	0	12.4	96.6	1.6	0	8.4	16	2	1.8	6.2	1	1
cosatifarei	4.4	3.2	4.2	13.8	0.2	0.2	0.2	6.6	0.8	11.6	2.4	131.6	0	0.4	2	1.2	1.6	8.8	2	3.4
basta	0.2	0	0	0.6	6.6	0.4	1.4	0	1	0	0.8	0	186.2	1.8	0.4	1.2	0	0	0.2	0.6
prendere	2.2	7.4	0.2	3.2	0.6	0.8	0.2	2.2	0.2	1.2	5.6	0.6	0.8	132.4	19.4	11.8	0.2	2.6	6.8	2.8
noncenepiu	0.6	5.8	0.4	3	0.8	4	2.8	0.4	0	9	14.6	1.2	0.8	10.6	111.8	4.6	0.2	3.2	11.4	9.4
fame	0	1.2	0	2.8	1.6	0.8	1	0.2	0	1.4	2.2	1.2	0.8	9.2	2.2	172.2	0.2	1.8	4.8	2.6
tantotempo	15.2	5.6	4	0.6	0	0.2	0	6.6	0	0	2.8	3	0	0.2	0.2	0.2	164.2	1.2	0.2	0
buonissimo	1.6	5	2.4	11.6	0	1.6	0	16.8	0.4	13.8	6.2	12.2	0	2.6	3.2	0.8	0.6	117.4	0.2	1.4
messidaccordo	2.2	0.4	0	0.2	0.8	1.4	7	0.2	0.8	1.8	1.4	0.6	0	5.6	5.6	4.8	0	0	147.8	15.8
sonostufo	0.6	0.4	0.4	2.8	0	1.8	1.8	0	3	10.2	1.6	2.2	0	3.2	9	1.4	0	3.6	15.6	135.8

Fig. 14 The confusion matrix of the average case for CNNLSTM on the Montalbano dataset. The confusion between gesture classes with *indistinctive movement* is pronounced, e.g., "vattene," "vieniqui," "perfetto," and "tantotempo"



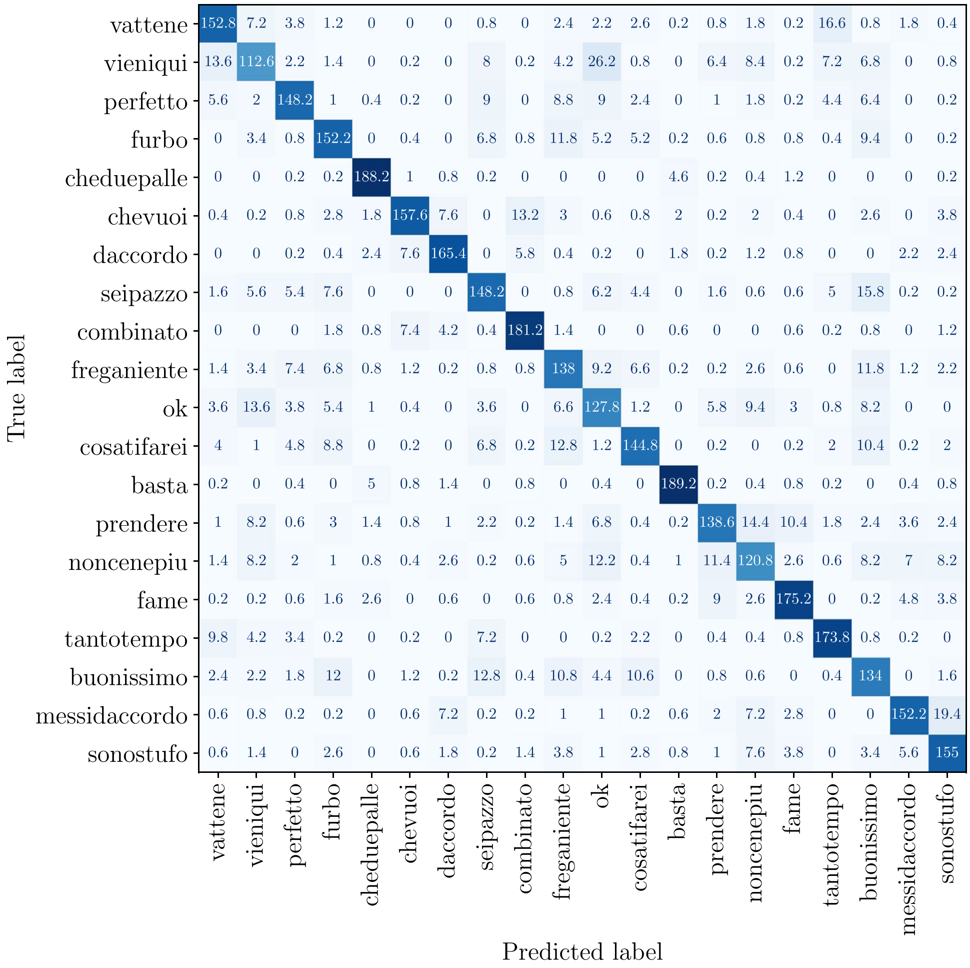
True label \ Predicted label	vattene	vieniqui	perfetto	furbo	cheduepalle	chevuoi	daccordo	seipazzo	combinato	freganiente	ok	cosatifarei	basta	prendere	noncenepiu	fame	tantotempo	buonissimo	messidaccordo	sonostufo
vattene	139.2	12.2	3	0.8	0	0.2	0	1.2	0.2	4	4	4.2	0	1.6	1.8	0.2	20.2	0	2.8	0.6
vieniqui	15	114.8	1.8	1.6	0	0.2	0.2	7.6	0.6	6.4	19	1.4	0	8	9.8	0.4	7.6	3.8	0.8	0
perfetto	5.2	3.6	145	1	0	0	0	9.4	0	9.8	8.6	4.2	0.2	0.8	3	0	5.4	3.2	0	0.4
furbo	0.4	3.6	1.4	137	0.2	0.4	0	6	1.2	16.8	7	7.4	0	1.4	0.4	1.4	0	10.4	0	2.2
cheduepalle	0	0.2	0.2	0	188.2	1.2	0.8	0	0	0.6	0.4	0	4.4	0	0.2	1	0.2	0	0	0
chevuoi	1.6	0.8	0.6	2.2	3.6	153.6	9.6	0.8	14.8	2	0.6	1.4	1.2	0.8	2.2	0.4	0.2	0.8	0.4	2.2
daccordo	0.2	0.8	0.4	0	2.2	7.4	161	0.4	5.8	0.2	0	0.2	1.8	0.4	1.2	1.2	0	0	4.2	3
seipazzo	1.6	5.2	7.6	6.4	0	0	0.2	146.2	0.2	1.4	5.2	5.4	0	1.8	0.6	0.4	5	15.6	0.2	0.2
combinato	0.2	0.2	0	2.2	0.8	8.2	8.2	0	174.8	2.6	0	0	1.6	0.2	0.4	0.2	0	0.8	0.2	0.6
freganiente	2.8	2.6	8.8	6.6	1	1.2	0.2	0.8	1.6	137.6	6.8	4.8	0	1.6	8.8	1	0	5.6	0.6	2.6
ok	3	12.8	4	5	0.2	0.6	0	3.8	0.2	6	126.6	1	0	11.8	10.8	1.6	1	4.6	0	1.4
cosatifarei	5.4	1.8	5.8	6.6	0	0	0	5	0.2	12.6	2	142.6	0.2	1.2	0.4	0	1.6	10.8	0	3
basta	0.6	0.2	0	0.2	3	0	2.4	0	1	0.2	0.6	0	187.4	1.8	0.2	1.4	0	0.2	0.8	1.6
prendere	1.2	5.4	0.8	1.2	0.8	0.4	0.8	2.8	0	2.6	7.6	0.2	0.4	140.8	16.4	11.8	2.2	2.6	1.6	1.6
noncenepiu	1	4.8	1	1	1	0.6	1.2	0.8	0.2	8	11.2	0	0.6	17	123.4	4.4	0.6	3.4	6.8	8
fame	0.2	0.4	0.2	1.6	2.8	0.2	0.6	1.6	0.4	0.6	2.2	0.4	0.4	12.8	2.2	170.6	0.2	0.2	5	3.4
tantotempo	11.8	4.2	2.6	0.6	0.4	0	0	10	0.2	0	1.4	1.4	0.2	1	1	0.2	168.6	0	0.2	0
buonissimo	1.4	2.6	1.2	10	0	1	0.2	17.4	0.8	13.6	7.8	14.2	0	0.8	1.4	0.4	0.8	122	0.2	2.2
messidaccordo	0.8	0.8	0	0	0.2	0.8	6.2	0	0.8	2	0.4	0	0.4	3.2	11	3	0.4	0	147.6	18.6
sonostufo	0.6	1.2	0	1.8	0.4	0.2	1.8	0.2	1	2.4	1.2	1.6	0.4	2.4	7.6	2.6	0	1.2	8.4	158.4

Fig. 15 The confusion matrix of the average case for *Snapture* on the Montalbano dataset. The confusion concerning gesture classes with *indistinctive* and *implicit* movements, e.g., "vattene," "noncenepiu," and "ok," is less pronounced than for the CNNLSTM

On a different note, classes that share motion and hand shape are challenging for our approach. For example, classes "vattene," "vieniqui," and "tantotempo" use a similar open palm at the peak (cf. Fig. 18). Therefore, the confusion between such classes is still noticeable in *Snapture* and *Snapture*$_{thold}$ despite the hand shape information.

Implicit Movements Besides the motion similarity, some gestures include a delicate hand movement at the peak. For example, "sonostufo" includes a subtle hand movement against the chest. Similarly, "noncenepiu" and "buonissimo" include a rotational motion of the extended index and thumb fingers around the wrist. Due to the pre-processing, these hand details are lost. Consequently, they are not picked up by the CNNLSTM. However, the confusion related to these classes is noticeably less in *Snapture* and *Snapture*$_{thold}$ (cf. Figs. 15 and 16). On the other hand, the confusion regarding class "buonissimo" is only slightly boosted in *Snapture* and *Snapture*$_{thold}$. We explain that by observing that "buonissimo" and "furbo" are similar in motion and hand shape, i.e., extended index finger. The difference lies in the position the finger touches the face (under the eyes vs. on the cheek). Efficiently recognizing these gestures requires additional modalities, which our study does not consider. However, we will discuss this point later. Moreover, since *snapshot* is captured using one frame at the gesture's peak, it is subject to influence by the corresponding hand orientation and light reflection. Thus, it becomes more challenging to distinguish between an open palm and an extended index finger, especially since the input is in grayscale (cf. Fig. 19).

Explicit Movements Five Montalbano gestures are two-handed. We observe two types of movements under this category based on how the arms are extended. "Chevuoi" and "combinato" are performed using symmetric hand movements in which both arms move from the rest to make a distinct shape at chest level. Due to the motion similarity, the CNNLSTM comes short in F1-scores, most noticeable for "chevuoi," while *Snapture* and *Snapture*$_{thold}$ present a noticeable F1-score boost for these classes (cf. Figure 17). On the other hand, gestures "cheduepalle" and "basta" are symmetric but made with a movement of both arms to the body side. Both gestures are used when a person is acting decisively and implying "enough." Therefore, the movement



	vattene	vieniqui	perfetto	furbo	cheduepalle	chevuoi	daccordo	seipazzo	combinato	freganiente	ok	cosatifarei	basta	prendere	noncenepiu	fame	tantotempo	buonissimo	messidaccordo	sonostufo
vattene	152.8	7.2	3.8	1.2	0	0	0	0.8	0	2.4	2.2	2.6	0.2	0.8	1.8	0.2	16.6	0.8	1.8	0.4
vieniqui	13.6	112.6	2.2	1.4	0	0.2	0	8	0.2	4.2	26.2	0.8	0	6.4	8.4	0.2	7.2	6.8	0	0.8
perfetto	5.6	2	148.2	1	0.4	0.2	0	9	0	8.8	9	2.4	0	1	1.8	0.2	4.4	6.4	0	0.2
furbo	0	3.4	0.8	152.2	0	0.4	0	6.8	0.8	11.8	5.2	5.2	0.2	0.6	0.8	0.8	0.4	9.4	0	0.2
cheduepalle	0	0	0.2	0.2	188.2	1	0.8	0.2	0	0	0	0	4.6	0.2	0.4	1.2	0	0	0	0.2
chevuoi	0.4	0.2	0.8	2.8	1.8	157.6	7.6	0	13.2	3	0.6	0.8	2	0.2	2	0.4	0	2.6	0	3.8
daccordo	0	0	0.2	0.4	2.4	7.6	165.4	0	5.8	0.4	0.2	0	1.8	0.2	1.2	0.8	0	0	2.2	2.4
seipazzo	1.6	5.6	5.4	7.6	0	0	0	148.2	0	0.8	6.2	4.4	0	1.6	0.6	0.6	5	15.8	0.2	0.2
combinato	0	0	0	1.8	0.8	7.4	4.2	0.4	181.2	1.4	0	0	0.6	0	0	0.6	0.2	0.8	0	1.2
freganiente	1.4	3.4	7.4	6.8	0.8	1.2	0.2	0.8	0.8	138	9.2	6.6	0.2	0.2	2.6	0.6	0	11.8	1.2	2.2
ok	3.6	13.6	3.8	5.4	1	0.4	0	3.6	0	6.6	127.8	1.2	0	5.8	9.4	3	0.8	8.2	0	0
cosatifarei	4	1	4.8	8.8	0	0.2	0	6.8	0.2	12.8	1.2	144.8	0	0.2	0	0.2	2	10.4	0.2	2
basta	0.2	0	0.4	0	5	0.8	1.4	0	0.8	0	0.4	0	189.2	0.2	0.4	0.8	0.2	0	0.4	0.8
prendere	1	8.2	0.6	3	1.4	0.8	1	2.2	0.2	1.4	6.8	0.4	0.2	138.6	14.4	10.4	1.8	2.4	3.6	2.4
noncenepiu	1.4	8.2	2	1	0.8	0.4	2.6	0.2	0.6	5	12.2	0.4	1	11.4	120.8	2.6	0.6	8.2	7	8.2
fame	0.2	0.2	0.6	1.6	2.6	0	0.6	0	0.6	0.8	2.4	0.4	0.2	9	2.6	175.2	0	0.2	4.8	3.8
tantotempo	9.8	4.2	3.4	0.2	0	0.2	0	7.2	0	0	0.2	2.2	0	0.4	0.4	0.8	173.8	0.8	0.2	0
buonissimo	2.4	2.2	1.8	12	0	1.2	0.2	12.8	0.4	10.8	4.4	10.6	0	0.8	0.6	0	0.4	134	0	1.6
messidaccordo	0.6	0.8	0.2	0.2	0	0.6	7.2	0.2	0.2	1	1	0.2	0.6	2	7.2	2.8	0	0	152.2	19.4
sonostufo	0.6	1.4	0	2.6	0	0.6	1.8	0.2	1.4	3.8	1	2.8	0.8	1	7.6	3.8	0	3.4	5.6	155

Fig. 16 The confusion matrix of the average case for *Snapture*-$_{thold}$ on the Montalbano dataset. The confusion concerning gesture classes "vattene," "furbo," and "buonissimo" is less pronounced than for the CNNLSTM

of the arm is quite firm, making it unique from the rest of the gesture vocabulary. Consequently, the CNNLSTM is efficient at picking up these movements. *Snapture* and *Snapture*$_{thold}$ only slightly improve over the performance of the CNNLSTM concerning these *explicit* movements since the hand shape and finger arrangement play a minimal role in their recognition. In Fig. 20, we display a comparison between an *implicit* and *explicit* movement and their corresponding pre-processing step.

Discussion

We proposed a hybrid gesture recognition architecture called *Snapture*. It integrated the hand pose alongside movement through modular static and dynamic channels. Our work was motivated by the limitation of RGB techniques, such as the CNNLSTM network, across different gesture domains. Therefore, we evaluated our approach in the context of robot commands and co-speech gestures. In our experiments, we compared the performance of *Snapture* to a CNNLSTM baseline using the GRIT [12] and Montalbano [13] datasets. Our analysis demonstrated that *Snapture* improved the classification of *indistinctive* and *subtle* movements. We believe the unique characteristics of our approach make it potentially beneficial in the following domains: (1) emblematic hand gestures, which substitute words to convey a particular meaning, and (2) co-speech gestures, which accompany words as means of verbal communication. Furthermore, our system is compatible with mobile systems and robot applications due to the few required data streams. We use only RGB frames to extract the motion and hand pose. The participants stand freely in front of the camera without needing environments with constrained setup conditions [18]. We also avoid issues that result from using skeleton data to extract the hand, such as the occasional loss of joint information [15]. Thus, our approach is one of the few pure RGB-based models that operate on the Montalbano dataset.

Recent RGB-based approaches are influenced by similarities in hand movements [9] and the loss of delicate small-scale motions at the peak [10]. The effects of this phenomenon are limited in the GRIT dataset because it includes robot commands with intense arm movements. On the other hand, these effects are more pronounced in the Montalbano

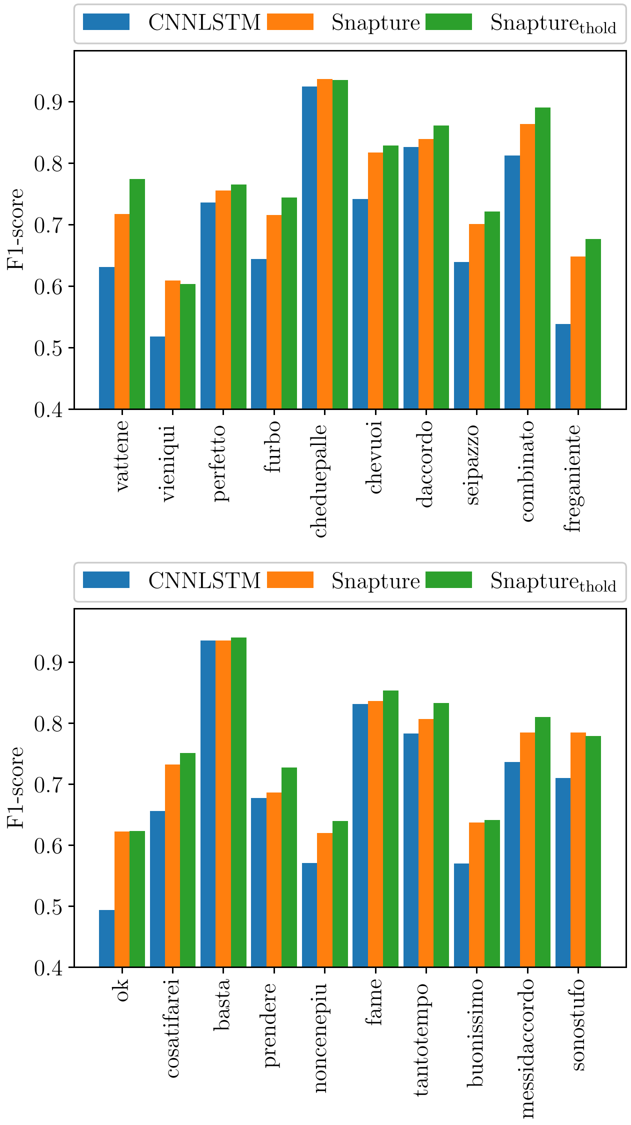


Fig. 17 A comparison of per-class F1-score values between the different approaches on the Montalbano dataset. *Snapture* improves the score for all classes except “basta.” The performance of *explicit* arm movements, e.g., “basta” and “cheduepalle,” is comparable across the three models

Italian gestures, which are part of human communication. These movements are more natural than robot control, have a simple motion path, and involve particular hand and finger configurations. Thus, capturing the hand pose in addition to the movement becomes more critical. However, this is challenging for recent approaches, which require intensive training of networks, such as 3DCNN [14], ResNet [10], and Inception V3 [15]. In contrast, our system captures the hand details by merely incorporating an additional static channel. We integrate the hand information at the gesture’s peak on top of the CNNLSTM model. Therefore, our architecture is easier to train yet effectively capable of addressing the issues of *indistinctive* and *subtle* movements. The simplicity of our approach facilitates a robot application due to its lightweight architecture.

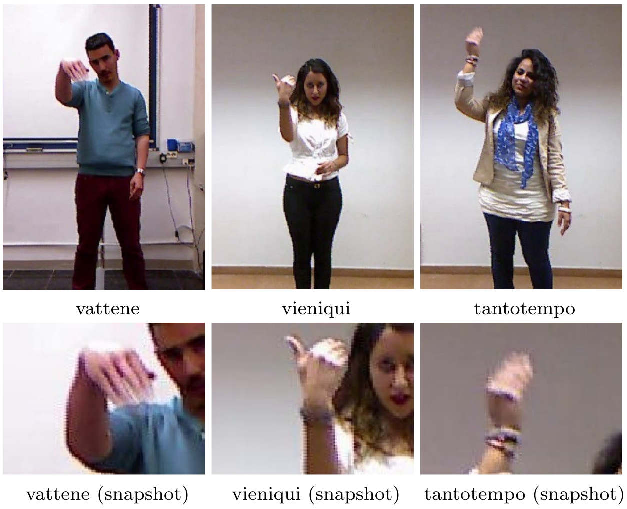

vattene
vieniqui
tantotempo
vattene (snapshot)
vieniqui (snapshot)
tantotempo (snapshot)

Fig. 18 Our *snapshot* extraction takes place using a single frame at the peak. Thus, a challenging scenario to our approach is when gestures that have a similar hand pose during the *stroke* phase

Furthermore, the scheme of fusing the static and dynamic features influences the system. Our approach operates on a single frame in the static channel, which has several advantages. First, it matches Kendon [11] model of gesticulation and concurrent speech. The literature shows that the *stroke* phase plays an essential role in recognition. Following this model also helps simplify our approach since it does not require dedicated networks for learning the short-term and long-term

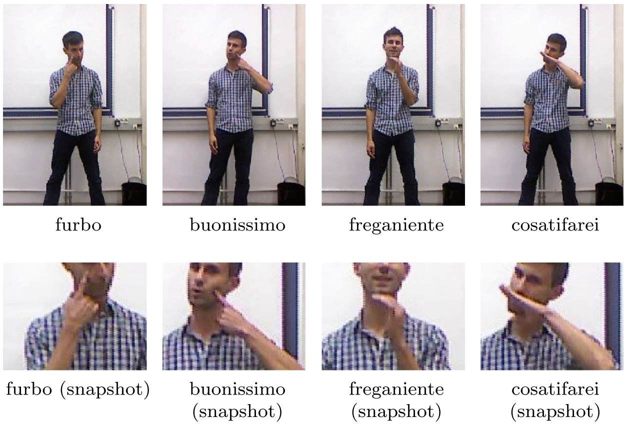

furbo
buonissimo
freganiente
cosatifarei
furbo (snapshot)
buonissimo (snapshot)
freganiente (snapshot)
cosatifarei (snapshot)

Fig. 19 Some challenges concerning class “buonissimo”: **a** similarity in hand motion and pose with “furbo.” Therefore, another modality is required, which is not considered by our approach, **b** similarity in hand orientation and light reflection causes misclassifications with “freganiente” and “cosatifarei.” It becomes challenging to interpret the open palm under these conditions

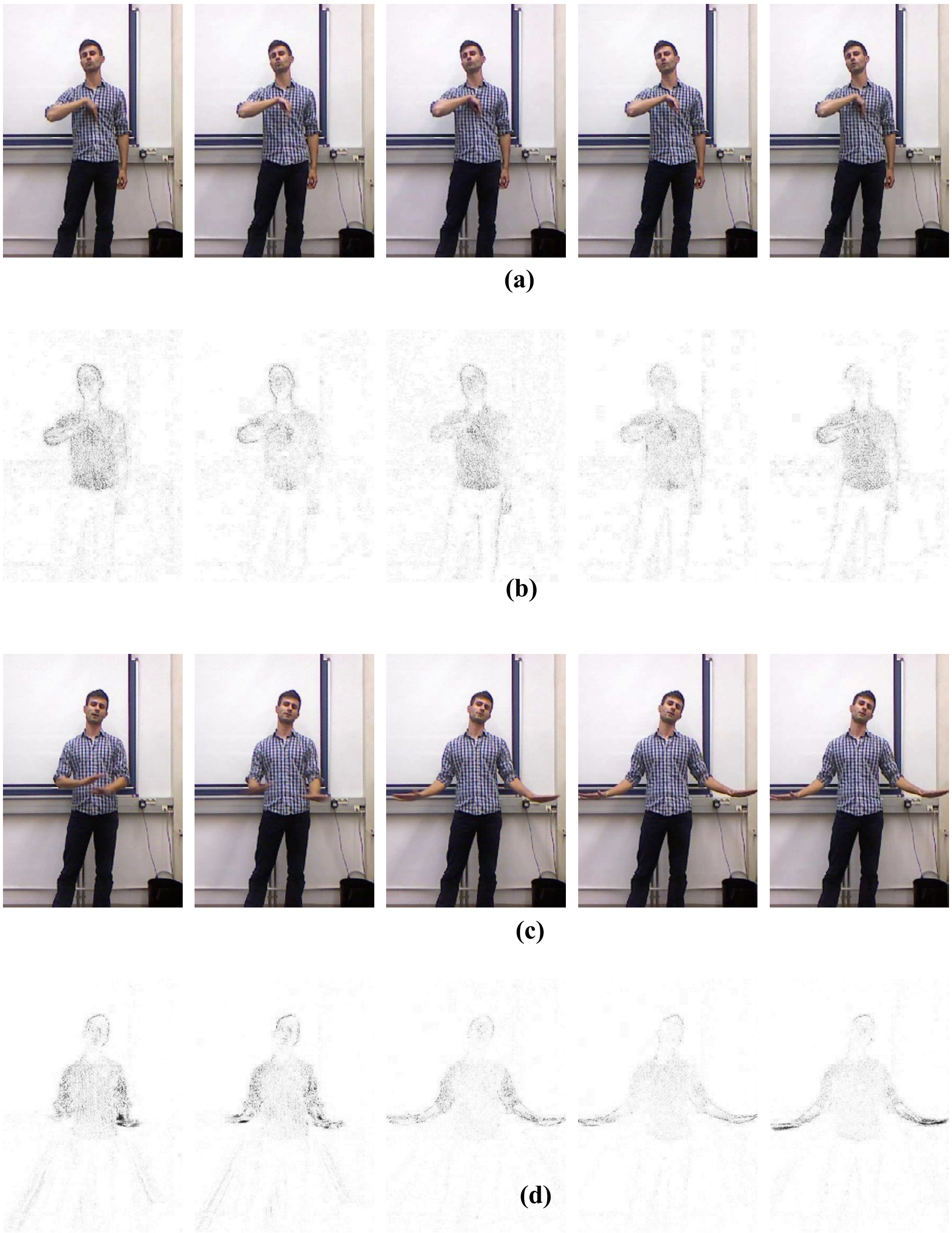


Fig. 20 A comparison between *implicit* and *explicit* hand movements. We observe missing hand details concerning "sonostufo" (**b**). In contrast, the *explicit* arm movement of "basta" is conserved (**d**). **a** and **c** depict the original sequence for clarity. We increase the contrast of **b** and **d** for clarity

dependencies [21, 28]. Second, the spatial and temporal traits are treated with equal importance. Thus, we avoid the issues of fusing features at each time step and the dominance of particular modalities in the learning process [14]. Therefore, we are better able to analyze the influence of each stream on the final outcome. Our experimental design provides evidence that classification performance concerning *indistinctive* and *subtle* movements can be boosted by learning hand details.

Another consideration for RGB-based methods is the issue of *blurry* frames caused by rapid hand movement and leading to noise in the input data. Due to a lack of literature concerning the analysis of gesture sequences, we employ an SSIM-based algorithm for analyzing motion profiles. This technique is used in a threshold-based fashion and further benefits our approach. The performance is improved by regulating the static channel and bypassing the *blurriness* issue. Thus, our method is comparable to others that use adaptive sampling of the input frames [26]. In both architectures, the performance is improved by reducing noise in the input frames. However, we control the noise by focusing on the hand's motion rather than just the presence of the hand, so that our approach works with co-speech gestures. However, one disadvantage of such empirical methods is that they do not guarantee generalization to new samples. Therefore, introducing robustness by learning the cut-off values of the threshold is desired. We hypothesize that approaches using attention-based methods, such as [21], might provide an alternative solution. However, this issue still negatively influences vision-based systems, and our results show that the *blur* phenomenon is challenging. We hope our work raises more attention to the quality of collected RGB gesture datasets and encourages more research in producing affordable, higher-quality cameras compatible with robots.

Our observations on the GRIT and Montalbano datasets show high variability in hand preference. Besides hand dominance, fatigue and injuries are among the most common factors that drive the interchangeable use of both hands. Therefore, a robust system that works with subjects regardless of the dominant hand is beneficial. Our system accomplishes that by extracting the pose of the hand actively used while making the gesture. Thus, it does not require mirroring videos of left-handed subjects [14] or a dedicated network for each hand [15]. Consequently, our approach facilitates higher flexibility, which should assist in less restrictive and guided HRI scenarios. However, this is a broader research domain, and more work could be done in this area. Another interesting finding is that our architecture is prone to confusion between classes with a similar hand pose at the *stroke* phase. Gestures such as "furbo" and "buonissimo" are almost identical at the peak with minor distinction. Thus, precisely recognizing these classes requires a more detailed interpretation of facial or speech information. Our modular architectural design facilitates that by incorporating additional channels. We hypothesize that such faulty system behavior can be avoided by including the body pose information. However, it remains an open question whether this is achievable in an architecture based on RGB data only.

Finally, our training scheme could be further improved. In our evaluation, we use a stratified sampler for the data splits to avoid imbalances of class labels. Having a representative test set is important due to the small size of the GRIT dataset and to make sure that challenging gestures, such as "hello" and "no," are represented fairly in the test set. Although our data split guarantees mutually exclusive sample sets, some of those samples might be recorded for the same participants. Since many participants were recorded in different surroundings and appearances, i.e., different outfits, each environmental and appearance combination presents a different challenge to our RGB approach. However, having a different training strategy, e.g., using a subject-wise split, would demonstrate the model's robustness to subject variability. Although this is not part of this study, we highlight it as a potential future improvement.

Conclusion

Our study presents a novel architecture called *Snapture*, which integrates static and dynamic information. Our use of RGB data only and our lightweight architecture allow compatibility with any system equipped with a camera, including robots. We also suggest an algorithm for analyzing gesture motion profiles, which is essential for revealing the unique characteristics of gestures. Our results show that incorporating the hand pose at the gesture's peak with motion information offers a solution to the issues of *indistinctive* and *subtle* movements. The results also demonstrate that these challenges are more prominent in the context of co-speech gestures than robot commands. Therefore, this hints at the substance of evaluating frameworks across multiple gesture domains. Additionally, our $Snapture_{thold}$ extension highlights the influence of RGB data quality for system performance and provides a means for optimization based on a *snapshot* of a gesture. Overall, our work contributes to bridging the gap between static and dynamic gestures allowing gesture applications that foster immersive and less controlled HRI experiences.

Acknowledgements This work was partially supported by the DFG under project CML (TRR 169) and BMWK under project KI-SIGS and EU under project TERAIS. The authors would like to thank Philipp Allgeuer for the insightful comments that helped improve the first draft of this manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL.

Data Availability The GRIT [12] and Montalbano [13] datasets analyzed in this work are available in https://www.inf.uni-hamburg.de/en/inst/ab/wtm/research/corpora.html and https://chalearnlap.cvc.uab.cat/dataset/12/description/.

Declarations

Conflict of Interest The authors declare no competing interests.

References

1. Escalera S, Guyon I, Athitsos V. Gesture recognition. 1st ed. Incorporated: Springer Publishing Company; 2018.
2. Siddharth S, Agrawal A. Vision based hand gesture recognition for human computer interaction: a survey. Artif Intell Rev. 2015;43:1–54. https://doi.org/10.1007/s10462-012-9356-9.
3. Anwar S, Sinha SK, Vivek S, Ashank V. Hand gesture recognition: a survey. In: Nath V, Mandal JK, editors. Nanoelectronics, circuits and communication systems. Singapore: Springer Singapore; 2019. p. 365–71.

4. Chakraborty B, Sarma D, Bhuyan M, MacDorman K. A review of constraints on vision-based gesture recognition for human-computer interaction. IET Comput Vis. 2017;12. https://doi.org/10.1049/iet-cvi.2017.0052.
5. Abdulazeez AM, Faizi S. Vision-based mobile robot controllers: a scientific review. Turkish J Comput Math Educ (TURCOMAT). 2021;12. https://doi.org/10.17762/turcomat.v12i6.2695.
6. Renard F, Guedria S, De Palma N, Vuillerme N. Variability and reproducibility in deep learning for medical image segmentation. Sci Rep. 2020;10. https://doi.org/10.1038/s41598-020-69920-0.
7. Vanamsterdam B, Clarkson M, Stoyanov D. Gesture recognition in robotic surgery: a review. IEEE Trans Biomed Eng. 2021;1–1. https://doi.org/10.1109/TBME.2021.3054828.
8. Asadi-Aghbolaghi M, Clapés A, Bellantonio M, Escalante HJ, Ponce-López V, Baró X, Guyon I, Kasaei S, Escalera S. A survey on deep learning based approaches for action and gesture recognition in image sequences. In: 2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017); 2017. p. 476–83.https://doi.org/10.1109/FG.2017.150.
9. Tsironi E, Barros P, Wermter S. Gesture recognition with a convolutional long short-term memory recurrent neural network. In: Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN). 2016. p. 213–8.
10. dos Santos CC, Samatelo JLA, Vassallo RF. Dynamic gesture recognition by using CNNs and star RGB: a temporal information condensation. Neurocomputing. 2020;400:238–54. https://doi.org/10.1016/j.neucom.2020.03.038. www.sciencedirect.com/science/article/pii/S092523122030391X.
11. Kendon A. Gesticulation and speech: two aspects of the process of utterance. In: The relationship of verbal and nonverbal communication. De Gruyter Mouton; 2011. p. 207–28. https://doi.org/10.1515/9783110813098.207.
12. Tsironi E, Barros P, Weber C, Wermter S. An analysis of convolutional long short-term memory recurrent neural networks for gesture recognition. Neurocomputing. 2017;268:76–86. https://doi.org/10.1016/j.neucom.2016.12.088. www.sciencedirect.com/science/article/pii/S0925231217307555.
13. Escalera S, Baró X, Gonzàlez J, Bautista MA, Madadi M, Reyes M, Ponce-López V, Escalante HJ, Shotton J, Guyon I. Chalearn looking at people challenge 2014: dataset and results. In: Agapito L, Bronstein MM, Rother C, editors. Computer Vision - ECCV 2014 Workshops. Cham: Springer International Publishing; 2015. p. 459–73.
14. Wu D, Pigou L, Kindermans PJ, Le N, Shao L, Dambre J, Odobez JM. Deep dynamic neural networks for multimodal gesture segmentation and recognition. IEEE Trans Pattern Anal Mach Intell. 2016;38:1–1. https://doi.org/10.1109/TPAMI.2016.2537340.
15. Mazhar O, Ramdani S, Cherubini A. A deep learning framework for recognizing both static and dynamic gestures. Sensors. 2021;21:2227. https://doi.org/10.3390/s21062227.
16. Wan J, Li SZ, Zhao Y, Zhou S, Guyon I, Escalera S. Chalearn looking at people RGB-D isolated and continuous datasets for gesture recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). 2016. p. 761–9. https://doi.org/10.1109/CVPRW.2016.100.
17. Mazhar O. OpenSign - Kinect v2 hand gesture data - American sign language. 2019. https://doi.org/10.17632/k793ybxx7t.1.
18. D'Eusanio A, Simoni A, Pini S, Borghi G, Vezzani R, Cucchiara R. A transformer-based network for dynamic hand gesture recognition. In: 2020 International Conference on 3D Vision (3DV). 2020. p. 623–32. https://doi.org/10.1109/3DV50981.2020.00072.
19. Molchanov P, Yang X, Gupta S, Kim K, Tyree S, Kautz J. Online detection and classification of dynamic hand gestures with recurrent 3D convolutional neural networks. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016. p. 4207–15. https://doi.org/10.1109/CVPR.2016.456.
20. Manganaro F, Pini S, Borghi G, Vezzani R, Cucchiara R. Hand gestures for the human-car interaction: The Briareo dataset. In: Image Analysis and Processing – ICIAP 2019. Springer International Publishing; 2019. p. 560–71. https://doi.org/10.1007/978-3-030-30645-8_51.
21. Aditya W, Shih T, Thaipisutikul T, Fitriajie A, Gochoo M, Utaminingrum F, Lin CY. Novel spatio-temporal continuous sign language recognition using an attentive multi-feature network. Sensors. 2022;22:6452. https://doi.org/10.3390/s22176452.
22. Huang J, Zhou W, Zhang Q, Li H, Li W. Video-based sign language recognition without temporal segmentation. In: AAAI Conference on Artificial Intelligence (AAAI). 2018.
23. Pu J, Zhou W, Li H. Iterative alignment network for continuous sign language recognition. In: Conference on Computer Vision and Pattern Recognition (CVPR). 2019. p. 4160–9. https://doi.org/10.1109/CVPR.2019.00429.
24. Zhou H, Zhou W, Li H. Dynamic pseudo label decoding for continuous sign language recognition. Int Conf Multimedia Expo (ICME). 2019. https://doi.org/10.1109/ICME.2019.00223.
25. Koller O, Forster J, Ney H. Continuous sign language recognition: towards large vocabulary statistical recognition systems handling multiple signers. Comput Vis Image Underst. 2015;141:108–25.
26. Cao Z, Li Y, Shin BS. Content-adaptive and attention-based network for hand gesture recognition. Appl Sci. 2022;12(4). https://doi.org/10.3390/app12042041,https://www.mdpi.com/2076-3417/12/4/2041.
27. Zhang Y, Cao C, Cheng J, Lu H. Egogesture: a new dataset and benchmark for egocentric hand gesture recognition. IEEE Trans Multimedia. 2018;20(5):1038–50. https://doi.org/10.1109/TMM.2018.2808769.
28. Chen G, Dong Z, Wang J, Xia L. Parallel temporal feature selection based on improved attention mechanism for dynamic gesture recognition. Complex Intell Syst. 2022. https://doi.org/10.1007/s40747-022-00858-8.
29. Klaser A, Marszalek M, Schmid C. A spatio-temporal descriptor based on 3D-gradients. In: Everingham M, Needham C, Fraile R editors. BMVC 2008 - 19th British Machine Vision Conference. British Machine Vision Association, Leeds, United Kingdom; 2008. p. 275:1–10. https://doi.org/10.5244/C.22.99.
30. Wang Z, Bovik A, Sheikh H, Simoncelli E. Image quality assessment: From error visibility to structural similarity. IEEE Trans Image Process. 2004;13(4):600–12. https://doi.org/10.1109/TIP.2003.819861.
31. Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Bach F, Blei D, editors. Proceedings of the 32nd International Conference on Machine Learning, Proceedings of Machine Learning Research (vol. 37). PMLR, Lille, France; 2015. p. 448–56. https://proceedings.mlr.press/v37/ioffe15.html.
32. Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. J Mach Learn Res - Proc Track. 2010;9:249–56.
33. Pham V, Bluche T, Kermorvant C, Louradour J. Dropout improves recurrent neural networks for handwriting recognition. In: 2014 14th International Conference on Frontiers in Handwriting Recognition. 2014. p. 285–90. https://doi.org/10.1109/ICFHR.2014.55.
34. Hsu RL, Abdel-Mottaleb M, Jain A. Face detection in color images. IEEE Trans Pattern Anal Mach Intell. 2002;1:696–706. https://doi.org/10.1109/34.1000242.
35. Qiu-yu Z, Lu J, Zhang M, Duan H, Lv L. Hand gesture segmentation method based on YCbCr color space and k-means clustering. Int J Signal Process Image Process Pattern Recog. 2015;8:105–16. https://doi.org/10.14257/ijsip.2015.8.5.11.
36. Basilio JAM, Torres GA, Pérez GS, Medina LKT, Meana HMP. Explicit image detection using YCbCr space color model as skin detection. In: Proceedings of the 2011 American Conference on

Applied Mathematics and the 5th WSEAS International Conference on Computer Engineering and Applications, AMERICAN-MATH'11/CEA'11. World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, USA; 2011. p. 123–8.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.