A Deep Learning Framework for Recognizing Both Static and Dynamic Gestures

Mazhar, O.; Ramdani, Sofiane ; Cherubini, Andrea

DOI
10.3390/s21062227

Publication date
2021

Document Version
Final published version

Published in
Sensors

Citation (APA)
Mazhar, O., Ramdani, S., & Cherubini, A. (2021). A Deep Learning Framework for Recognizing Both Static and Dynamic Gestures. *Sensors*, *21*(6), Article 2227. https://doi.org/10.3390/s21062227

sensors

MDPI

Article

A Deep Learning Framework for Recognizing Both Static and Dynamic Gestures

Osama Mazhar [1,2,*], Sofiane Ramdani [1] and Andrea Cherubini [1]

[1] LIRMM, Université de Montpellier, CNRS, 34392 Montpellier, France; sofiane.ramdani@umontpellier.fr (S.R.); andrea.cherubini@lirmm.fr (A.C.)
[2] Cognitive Robotics Department, Delft University of Technology, 2628 CD Delft, The Netherlands
* Correspondence: o.mazhar@tudelft.nl

Abstract: Intuitive user interfaces are indispensable to interact with the human centric smart environments. In this paper, we propose a unified framework that recognizes both static and dynamic gestures, using simple RGB vision (without depth sensing). This feature makes it suitable for inexpensive human-robot interaction in social or industrial settings. We employ a pose-driven spatial attention strategy, which guides our proposed Static and Dynamic gestures Network—*StaDNet*. From the image of the human upper body, we estimate his/her depth, along with the region-of-interest around his/her hands. The Convolutional Neural Network (CNN) in *StaDNet* is fine-tuned on a background-substituted hand gestures dataset. It is utilized to detect 10 static gestures for each hand as well as to obtain the hand image-embeddings. These are subsequently fused with the augmented pose vector and then passed to the stacked Long Short-Term Memory blocks. Thus, human-centred frame-wise information from the augmented pose vector and from the left/right hands image-embeddings are aggregated in time to predict the dynamic gestures of the performing person. In a number of experiments, we show that the proposed approach surpasses the state-of-the-art results on the large-scale *Chalearn 2016* dataset. Moreover, we transfer the knowledge learned through the proposed methodology to the *Praxis gestures* dataset, and the obtained results also outscore the state-of-the-art on this dataset.

Keywords: gestures recognition; operator interfaces; human activity recognition; commercial robots and applications; cyber-physical systems

Citation: 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

Academic Editor: Frantisek Duchon

Received: 12 February 2021
Accepted: 17 March 2021
Published: 23 March 2021

1. Introduction

The modern manufacturing industry requires human-centered smart frameworks, which aim to focus on human abilities and not conversely demand humans to adjust to whatever technology. In this context, gesture-driven user-interfaces tend to exploit human's prior knowledge and are vital for intuitive interaction of humans with smart devices [1]. Gesture recognition is a problem that has been widely studied for developing human-computer/machine interfaces with an input device alternative to the traditional ones (e.g., mouse, keyboard, teach pendants and touch interfaces). Its applications include robot control [2–4], health monitoring systems [5], interactive games [6] and sign language recognition [7].

The aim of our work is to develop a robust, vision-based gestures recognition strategy suitable for human-robot/computer interaction tasks in social or industrial settings. Industrial applications where human safety is critical, often require specialized sensors compatible with safety standards such as ISO/TS 15066. Yet, scenarios which require human sensing in industry or in social settings are broad. Monocular cameras offer benefits which specialized or multi-modal sensors do not have, such as being lightweight, inexpensive, platform independent and easy to integrate. This is desirable for robotic assistants in commercial businesses such as restaurants, hotels, or clinics. We therefore, pro-

pose a unified framework for recognizing static and dynamic gestures from RGB images/video sequences.

A study of gestural communication [8] notes that most gestures used in assembly tasks are physically simple while no non-hand body language is involved in part manipulation. At first, we design a robust static hand gestures detector which is trained on a background substituted gestures dataset namely *OpenSign* [9], which contains 9 American Sign Language (ASL) gestures. Sign language is considered among the most structured set of gestures [10]. In this work, we employ sign language gestures only as a proof of concept—our static hand gesture detector can be adapted to other classes as well. The static hand gestures detector is detailed in [11]. For more generic and flexible gesture detection, we propose a multi-stream neural architecture for dynamic gestures recognition, which is integrated with our static hand gestures detector in a unified network.

Our unified network is named *StaDNet*—Static and Dynamic gestures Network. It learns to smartly focus on dominant input stream(s) to correctly recognize large-scale upper-body motions plus subtle hand movements, and therefore distinguish several inter-class ambiguities. The idea of *visual attention* presented in [12] is also embedded in *StaDNet*, which is eventually based the on human selective focus and perception. Thus, we develop a pose-driven hard spatial-attention mechanism, which focuses on the human upper body and on his/her hands (see Figure 1). It is also noteworthy that in the RGB images, scale information about the subjects (e.g., size of his/her body parts) is lost. To address this problem, we devise novel *learning-based depth estimators* to regress the distance of the hands and the upper-body from the sensor. Our depth estimators are trained on the ground truth depth obtained from the video sequences of Kinect V2. Once the parameters are learned, our algorithm is able to regress the relative depth of the body joints only from the 2D human skeleton. Therefore, in practice, we no longer require a depth sensor and *StaDNet* is able to detect static and dynamic gestures exclusively from the color images. This characteristic makes *StaDNet* suitable for inexpensive human-robot interaction in social or industrial settings.

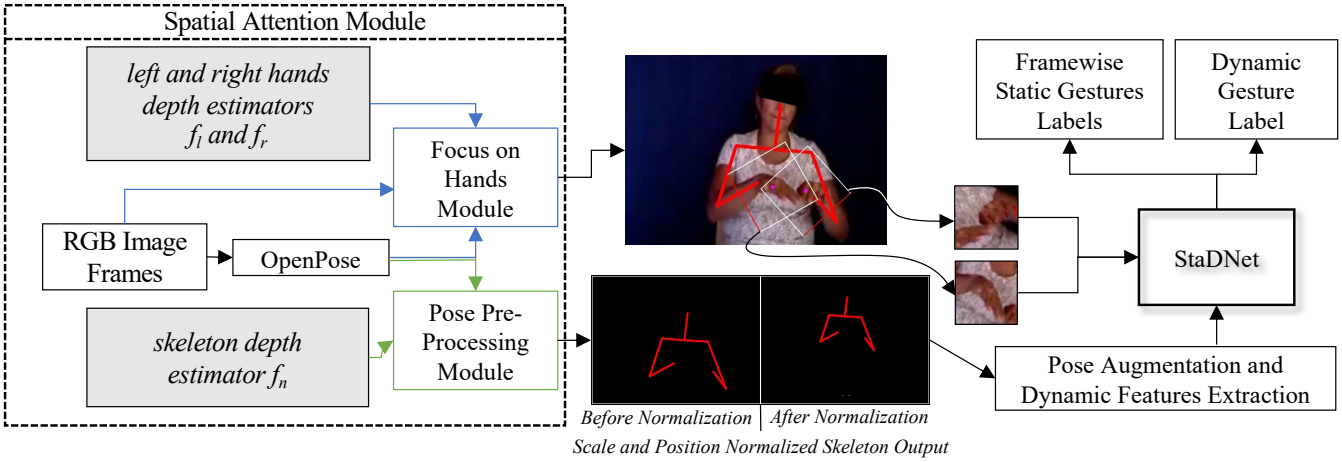

Figure 1. Illustration of our proposed framework. In Spatial Attention Module, we mainly have learning-based depth estimators (grey boxes), Focus on Hands (FOH) Module and Pose Pre-Processing (PP) Module. 2D skeleton is extracted by *OpenPose*. FOH exploits hand coordinates obtained from the skeleton and crops hand images with the help of hand depth estimators, while PP performs scale and position normalization of the skeleton with the help of skeleton depth estimator. The features from the normalized pose are extracted by Pose Augmentation and Dynamic Features Extraction Module and are fed to *StaDNet* together with the cropped hand images. *StaDNet* detects frame-wise static gestures as well as dynamic gestures in each video.

2. Related Work

The gesture detection techniques can be mainly divided into two categories: wearable strategies and non-wearable methods. The wearable strategies include electronic/glove-based systems [13,14], and markers-based vision [15] methods. However, these are often expensive, counter-intuitive and limit the operator's dexterity in his/her routine tasks. Conversely, non-wearable strategies such as pure-vision based methods, do not require structuring the environment and/or the operator, while they offer ease-of-use to interact with the robots/machines. Moreover, the consumer-based vision sensors have rich output, are portable and low cost, even when depth is also measured by the sensor such as Microsoft Kinect or Intel Realsense cameras. Therefore, in this research we opt for a pure vision-based method and review only the works with vision-based gestures detection.

Traditional activity recognition approaches aggregate local spatio-temporal information via hand-crafted features. These visual representations include the Harris3D detector [16], the Cuboid detector [17], dense sampling of video blocks [18], dense trajectories [19] and improved trajectories [20]. Visual representations obtained through optical flow, for example, Histograms of Oriented Gradients (HOG), Histograms of Optical Flow (HOF) and Motion Boundary Histograms (MBH) also achieved excellent results for video classification on a variety of datasets [18,21]. In these approaches, global descriptors of the videos are obtained by encoding the hand-crafted features using Bag of Words (BoW) and Fischer vector encodings [22]. Subsequently, the descriptors are assigned to one or several nearest elements in a vocabulary [23] while the classification is typically performed through Support Vector Machines (SVMs). In [24], the authors segmented the human silhouettes from the depth videos using Otsu's method of global image threshold [25]. They extracted a single Extended-Motion History Image (Extended-MHI) as a global representation for each gesture. Subsequently, maximum correlations coefficient was utilized to recognize gestures in a One-Shot learning setting. Other works that utilized One-Shot Learning for gesture recognition include [26–28].

Lately, the tremendous success of deep neural networks on image classification tasks [29,30] instigated its application in activity recognition domain. The literature on the approaches that exploit deep neural networks for gestures/activity recognition is already enormous. Here, we focus on related notables which have inspired our proposed strategy.

2.1. 3D Convolutional Neural Networks

Among the pioneer works in this category, [31] adapted Convolutional Neural Networks (CNNs) to 3D volumes (3D-CNNs), obtained by stacking video frames, to learn spatio-temporal features for action recognition. In [32], Baccouche et al. proposed an approach for learning the evolution of temporal information through a combination of 3D-CNNs and Long Short term Memory (LSTM) recurrent neural networks [33]. The short video clips of approximately 9 successive frames were first passed through a 3D-CNN features extractor while the extracted features were subsequently fed to the LSTM network. However, Karpathy et al. in [34] found that the stacked-frames architecture performed similar to the one with single-image input.

A Few-Shot temporal activity detection strategy is proposed in [35], which utilized 3D-CNNs for features extraction from the untrimmed input video as well as from the few-shot examples. A two-stage proposal network was applied on top of the extracted features while the refined proposals were compared using cosine similarity functions.

To handle resulting high-dimensional video representations, the authors of [36] proposed the use of random projection-based ensemble learning in deep networks for video classification. They also proposed rectified linear encoding (RLE) method to deal with redundancy in the initial results of the classifiers. The output from RLE is then fused by a fully-connected layer that produced the final classification results.

2.2. Multi-Modal Multi-Scale Strategies

The authors of [7] presented a multi-modal multi-scale detection strategy for *dynamic poses* of varying temporal scales as an extension to their previous work [37]. They utilized the RGB and depth modalities, as well as the articulated pose information obtained through the depth map. The authors proposed a complex learning method which included pre-training of individual classifiers on separate channels and iterative fusion of all modalities on shared hidden and output layers. This approach involved recognizing 20 categories from Italian conversational gestures, performed by different people and recorded with an RGB-D sensor. This strategy was similar in function to [34] except that it included depth images and pose as additional modalities. However, it lacked a dedicated equipment to learn evolution of temporal information and may fail when understanding long-term dependencies of the gestures is required.

In [38], authors proposed a multi-modal large-scale gesture recognition scheme on the *Chalearn 2016 Looking at People Isolated Gestures recognition* dataset [39]. In [40], ResC3D network was exploited for feature extraction, and late fusion combined features from multi-modal inputs in terms of canonical correlation analysis. The authors used linear Support Vector Machine (SVM) to classify final gestures. They proposed a *key frame attention mechanism*, which relied on movement intensity in the form of optical flow, as an indicator for frame selection.

2.3. Multi-Stream Optical Flow-Based Methods

The authors of [41] proposed an optical flow-based method exploiting convolutional network networks for activity recognition along the same lines of [34]. They presented the idea of decoupling spatial and temporal networks. The proposed architecture in [41] is related to the two-stream hypothesis of the human visual cortex [42]. The spatial stream in their work operated on individual video frames, while the input to the temporal stream was formed by stacking optical flow displacement fields between multiple consecutive frames.

The authors of [43] presented improved results in action recognition, by employing a trajectory-pooled two-stream CNN inspired by [41]. They exploited the concept of improved trajectories as low level trajectory extractor. This allowed characterization of the background motion in two consecutive frames through the estimation of the homography matrix taking camera motion into account. Optical flow-based methods (e.g., the *key frame attention mechanism* proposed in [38]) may help emphasizing frames with motion, but are unable to differentiate motion caused by irrelevant objects in the background.

2.4. CNN-LSTM and Convolutional-LSTM Networks

The work in [44] proposed aggregation of frame-level CNN activations through (1) Feature-pooling method and (2) LSTM network for longer sequences. The authors argued that the predictions on individual frames of video sequences or on shorter clips as performed in [34], might only contain local information of the video description, while it could also confuse classes if there are fine-grained distinctions.

The authors of [45] proposed a Long-term Recurrent Convolutional Network (LRCN) for multiple situations including sequential input and static output for cases like activity recognition. The visual features from RGB images were extracted through a deep CNN, which were then fed into stacked LSTM in distinctive configurations corresponding to the task at hand. The parameters were learned in an "end-to-end" fashion, such that the visual features relevant to the sequential classification problem were extracted.

The authors in [46] proposed a method to process sequential images through Convolutional-LSTM (ConvLSTM), which is a variant of LSTM containing a convolution operation inside the LSTM cell. In [47], the authors studied redundancy and attention in ConvLSTM by deriving its several variants for gesture recognition. They proposed Gated-ConvLSTM by removing spatial convolutional structures in the gates as they scarcely contributed to the spatio-temporal feature fusion in their study. The authors evaluated results on the *Chalearn 2016* dataset and found that the Gated-ConvLSTM achieved reduc-

tion in parameters size and in computational cost. However, it did not improve detection accuracy to a considerable amount.

2.5. Multi-Label Video Classification

The authors of [48] presented a multi-label action recognition scheme. It was based on Multi-LSTM network which tackled with multiple inputs and outputs. The authors fine-tuned VGG-16 pre-trained on ImageNet [49], on Multi-THUMOS dataset at the individual frame level. Multi-THUMOS is an extension of THUMOS dataset [50]. A fixed length window of 4096-dimensional "fc7" features of the fine-tuned VGG-16 was passed as input to the LSTM, through an attention mechanism, that weighted the contribution of individual frames in the window.

2.6. Attention-Based Strategies

The application of convolutional operations on entire input images tends to be computationally expensive. In [12], Rensink discussed the idea of *visual representation*, which implied that the humans do not form detailed depiction of all objects in a scene. Instead, their perception focuses selectively on the objects needed immediately. This was supported by the concept of *visual attention* applied for deep learning methods as in [51].

Baradel et al. [52] proposed a spatio-temporal attention mechanism conditioned on human pose. The proposed spatial-attention mechanism was inspired by the work of Mnih et al. [51] on glimpse sensors. A spatial attention distribution was learned conjointly through the hidden state of the LSTM network and through the learned pose feature representations. Later, Baradel et al. extend their work in [53] and proposed that the spatial attention distribution can be learned only through an *augmented pose vector*, which was defined by the concatenation of current pose, velocity and accelerations of each joint over time.

The authors of [54] proposed a three streams attention network for activity detection. These were statistic-based, learning-based and global-pooling attention streams. Shared ResNet was used to extract spatial features from image sequences. They also proposed a global attention regularization scheme to enable the employed recurrent networks to learn dynamics based on global information.

Lately, the authors of [55] presented the state-of-the-art results on the *Chalearn 2016* dataset. They proposed a novel multi-channel architecture, namely FOANet, built upon a spatial *focus of attention (FOA)* concept. They cropped the regions of interest occupied by the hands in the RGB and depth images, through the region proposal network and Faster R-CNN method. The architecture comprised of 12 channels in total with: 1 global (full-sized image) channel and 2 focused (left and right hand crops) channels for each of the 4 modalities (RGB, depth and optical flow fields extracted from the RGB and depth images). The softmax scores of each modality were fused through a sparse fusion network.

3. Datasets

For dynamic gestures classification, we use the *Chalearn 2016 Isolated Gestures dataset* [39], referred to simply as *Chalearn 2016* in the rest of the paper. It is a large-scale dataset which contains Kinect V1 color and depth recordings in 320×240 resolution of 249 dynamic gestures recorded with the help of 21 volunteers. The gestures vocabulary in the *Chalearn 2016* is mainly from nine groups corresponding to the different application domains: body language gestures, gesticulations, illustrators, emblems, sign language, semaphores, pantomimes, activities and dance postures. The dataset has 47,930 videos with each video (color + depth) representing one gesture. It has to be noted that the Chalearn 2016 does not take into account any specific industry requirements, and that Kinect V1 is obsolete. However, as we intend to target a broader human–robot interaction domain which includes the fast-growing field of socially assistive as well as household robotics, this requires robots to have the capacity to capture, process and understand human requests in a robust, natural and fluent manner. Considering the fact that the Chalearn 2016 dataset

offers a challenging set of gestures taken from a comprehensive gestures vocabulary with inter-class similarities and intra-class differences, we assumed the Chalearn 2016 suitable for training and benchmarking results of our strategy.

To demonstrate the utility of our approach on a different gesture dataset, we evaluate the performance of our model on the Praxis gesture dataset [56] as well. This dataset is designed to diagnose *apraxia* in humans, which is a motor disorder caused by brain damage. This dataset contains RGB (960×540 resolution) and depth (512×424 resolution) images recorded by 60 subjects plus 4 clinicians with Kinect V2. In total, 29 gestures were performed by the volunteers (15 static and 14 dynamic gestures). In our work, only dynamic gestures that is, 14 classes are considered while their pathological aspect is not taken into account that is, only gestures labeled "correct" are selected. Thus, the total number of considered videos in this dataset is 1247 with mean length of all samples equal to 54 frames. *StaDNet* is trained exclusively on color images of these datasets for dynamic gestures detection.

4. Our Strategy

In this work, we develop a novel unified strategy to model human-centered spatio-temporal dependencies for the recognition of static as well as dynamic gestures. Our *Spatial Attention Module* localizes and crops hand images of the person, which are subsequently passed as inputs to *StaDNet* unlike previous methods that take entire images as input for example, [44,45]. Contrary to [48], where a pre-trained state-of-the-art network is fine-tuned on entire image frames of gestures datasets, we fine-tune *Inception V3* on a background-substituted hand gestures dataset, used as our CNN block. Thus, our CNN has learned to concentrate on image pixels occupied exclusively by hands. This enables it to accurately distinguish subtle hand movements. We have fine-tuned *Inception V3* with a *softmax* layer, to classify 10 ASL static hand gestures while the features from the last fully connected (FC) layer of the network are extracted as image-embeddings of size 1024 elements. These are used as input to the dynamic gestures detector in conjunction with the augmented pose vector which we explain in Sections 5.1.2 and 6.1. Moreover, in contrast to the previous strategies for dynamic gestures recognition/video analysis [7,52,53], which employed 3D human skeletons to learn large-scale body motion–and corresponding sensor modalities–we only utilize 2D upper-body skeleton as an additional modality to our algorithm. However, scale information about the subjects is lost in monocular images. To address this, we also propose *learning-based depth estimators*, which determine the approximate depth of the person from the camera and region-of-interest around his/her hands from upper-body 2D skeleton coordinates only. In a nutshell, *StaDNet* only exploits the RGB hand images and an augmented pose vector obtained from 8 upper-body 2D skeleton coordinates, unlike other existing approaches like [55], which include full-frame images in addition to hand images, depth frames and even optical flow frames altogether.

To reiterate, our method does not require depth sensing. We only utilized the (raw) depth map from Kinect V2 offline, to obtain ground truth depth values of a given 2D skeleton for our *learning-based depth estimators*. These values can be obtained from any state-of-the-art depth sensor. Once the depth estimators are trained, our method only requires RGB modality to process images and detect gestures on-line. We employ *OpenPose* [57] which is an efficient discriminative 2D pose extractor, to extract the human skeleton and human hands' keypoints in images. *OpenPose* also works exclusively on the RGB images. Thus, our method can be deployed on a system with any RGB camera, be it a webcam or an industrial color (or RGB-D) camera. Nevertheless, we only tested *OpenPose* in laboratory or indoor domestic environments and not in a real industry. Yet, since our framework is not restricted to the use of *OpenPose*, we could integrate another pose extractor system, better suited for the target application scenario.

5. Spatial Attention Module

Our spatial attention module is divided into two parts—*Pose Pre-processing Module* and *Focus on Hands Module* (see Figure 1). We detail these modules in the following.

5.1. Pose Pre-Processing Module

We first resize the dataset videos to $1080 \times C$ pixels, where C is the value of resized image columns obtained with respect to new row value, that is, 1080, while maintaining the aspect ratio of the original image (1440 in our work). The necessity to resize the input videos will be explained in Section 5.1.3. After having resized the videos, we feed them to *OpenPose*, one at a time, and the output skeleton joint and hand keypoint coordinates are saved for offline pre-processing. The pose pre-processing is composed of three parts, detailed hereby: *skeleton filter*, *skeleton position and scale normalization* and *skeleton depth estimation*.

5.1.1. Skeleton Filter

For each image, *OpenPose* extracts N skeleton joint coordinates depending on the selected body model while it does not employ pose tracking between images. The occasional jitter in the skeleton output and missing joint coordinates between successive frames may hinder gesture learning. Thus, we develop a *two-step* pose filter that rectifies occasional disappearance of the joint(s) coordinates and smooths the *OpenPose* output. The filter operates on a window of K consecutive images (K is an adjustable odd number, 7 in this work), while the filtered skeleton is obtained in the center frame. We note $\mathbf{p}_k^i = (x^i, y^i)$, the image coordinates of the ith joint in the skeleton output by *OpenPose* at the k-th image within the window. If *OpenPose* does not detect joint i on image k: $\mathbf{p}_k^i = \emptyset$.

In a *first step*, we replace coordinates of the missing joints. Only $\bar{r}$ (we use $\bar{r} = 7$) consecutive replacements are allowed for each joint i, and we monitor this via a coordinate replacement counter, noted r^i. The procedure is driven by the following two equations:

$$\mathbf{p}_K^i = \mathbf{p}_{K-1}^i \quad \begin{array}{l} \text{if } \mathbf{p}_K^i = \emptyset \\ \wedge\ \mathbf{p}_k^i \neq \emptyset\ \forall k = 1, \ldots, K-1 \\ \wedge\ r^i \leq \bar{r} \end{array} \tag{1}$$

$$\mathbf{p}_{k=1,\ldots,K-1}^i = \begin{cases} \emptyset & \text{if } \mathbf{p}_K^i = \emptyset \wedge r^i > \bar{r} \\ \mathbf{p}_K^i & \text{if } \mathbf{p}_{k=1,\ldots,K-1}^i = \emptyset \wedge \mathbf{p}_K^i. \neq \emptyset \end{cases} \tag{2}$$

Equation (1) states that the i-th joint at the latest (current) image K is replaced by the same joint at the previous image $K-1$ under three conditions: if it is not detected, if it has been detected in all previous images, and if in the past it has not been replaced up to $\bar{r}$ consecutive times already. If any of the conditions is false, we do not replace the coordinates and we reset the replacement counter for the considered joint: $r^i = 0$. Similarly, (2) states that the i-th joint coordinates over the window should not be taken into account that is, joint will be considered missing, if it is not detected in the current image K and if it has already been replaced more than $\bar{r}$ consecutive times (we allow only $\bar{r}$ consecutive replacements driven by (1)). This also resets the replacement counter value for the considered joint. Moreover, the i-th joint in all of the window's $K-1$ images is set to its position in the current image K, if it has never been detected in the window up to the current image.

In the *second step*, we apply Gaussian smoothing to each $\mathbf{p}^i$, over the window of K images. Applying this filter removes jitter from the skeleton pose and smooths out the joint movements in the image at the center of the filter window. Figure 2 shows the output of our skeleton filter for one window of images.

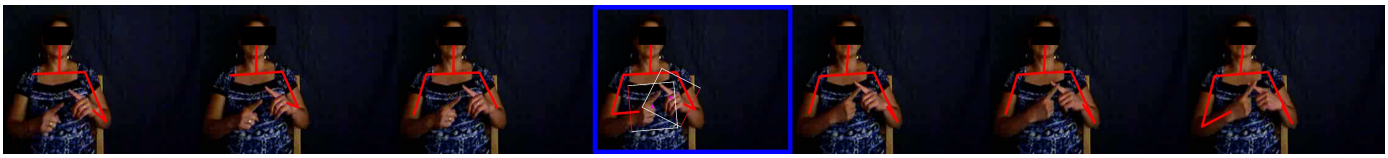

Figure 2. The *Skeleton Filter* described in Section 5.1.1. Images are arranged from left to right in chronological order. The central image shows the skeleton output by the filter. The six other images show the raw skeletons output by *OpenPose*. Observe that—thanks to Equation (1)—our filter has added the right wrist coordinates (shown only in the central image). These are obtained from the K-th frame, while they were missing in all raw skeletons from frame 1 to $K-1$.

5.1.2. Skeleton Position and Scale Normalization

Figure 1 includes a simple illustration of our goal for skeleton position and scale normalization. We focus on the 8 upper-body joints shown in Figure 3: $\mathbf{p}^{0,\ldots,7}$, with $\mathbf{p}^0$ corresponding to the *Neck* joint, which we consider as root node. *Position normalization* consists in eliminating the influence of the user's position in the image, by subtracting the *Neck* joint coordinates from those of the other joints. *Scale normalization* consists in eliminating the influence of the user's depth. We do this by dividing the position-shifted joint coordinates by the neck depth d_n, on each image, so the all joints are replaced according to:

$$\mathbf{p}^i \leftarrow \frac{\mathbf{p}^i - \mathbf{p}^0}{d_n}. \tag{3}$$

Since our framework must work without requiring a depth sensor, we have developed a skeleton depth estimator to derive the neck depth, $\widetilde{d}_n$ and use it instead of d_n in (3). This estimator is a neural network, which maps a 97-dimensional pose vector, derived from the 8 upper body joint positions, to the depth of the *Neck* joint. We will explain it hereby.

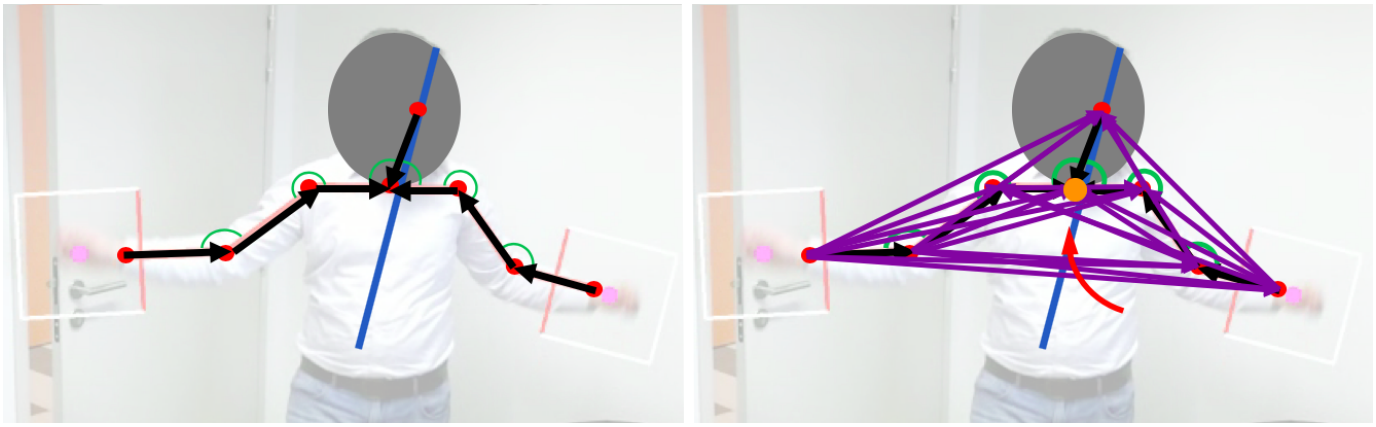

Figure 3. Features augmentation of the upper body. In the left image, we show 8 upper-body joint coordinates (red), vectors connecting these joints (black) and angles between these vectors (green). From all upper-body joints, we compute a *line of best fit* (blue). In the right image, we show all the vectors (purple) between unique pairs of upper-body joints. We also compute the angles (not shown) between the vectors and the line of best fit. From 8 upper-body joints, we obtain 97 components of the *augmented pose vector*.

5.1.3. Skeleton Depth Estimation

Inspired by [7], which demonstrated that augmenting pose coordinates may improve performance of gesture classifiers, we develop a 97 dimensional *augmented pose vector* $\mathbf{x_n}$ (subscript $\mathbf{n}$ means *Neck* here) from 8 upper-body joint coordinates. From the joints coordinates, we obtain—via least squares—a *line of best fit*. In addition to 7 vectors from anatomically connected joints, 21 vectors between unique pairs of all upper-body coordinates are also obtained. The lengths of individual augmented vectors are also included in $\mathbf{x_n}$. We also include the 6 angles formed by all triplets of anatomically connected joints, and the 28 angles, between the 28 (anatomically connected plus augmented) vectors and the *line*

of best fit. The resultant 97-dimensional augmented pose vector concatenates: 42 elements from abscissas and ordinates of the augmented vectors, their 21 estimated lengths and 34 relevant angles.

To obtain the ground-truth depth of *Neck* joint, denoted d_n, we utilize *OpenSign* dataset. *OpenSign* is recorded with Kinect V2 which outputs the RGB and the registered depth images with resolution 1080×1920. We apply our augmented pose extractor to all images in the dataset and—for each image—we associate $\mathbf{x_n}$ to the corresponding *Neck* depth. A 9 layers neural network f_n is then designed, to optimize parameters θ_n, given augmented pose vector $\mathbf{x_n}$ and ground-truth d_n to regress the approximate distance value $\widetilde{d_n}$ with a mean squared error of 8.34×10^{-4}. Formally:

$$\widetilde{d_n} = f_n(\mathbf{x_n}, d_n; \theta_n). \tag{4}$$

It is to be noted that the estimated depth $\widetilde{d_n}$ is a relative value and not in metric units, and that the resolution of ground truth images in *OpenSign* is 1080×1920. For scale normalization (as explained in Section 5.1.2), we utilize the estimated depth $\widetilde{d_n}$. Thus, the input images from the Chalearn 2016 dataset are resized such that the row count of the images is maintained to 1080. This is required as we need to re-scale the predicted depth to the original representation of the depth map in *OpenSign* (or to that of Kinect V2). Yet, the *StaDNet* input image size can be adapted to the user's needs if the depth estimators are not employed.

5.2. Focus on Hands Module

This module focuses on hands in two steps: first, by localizing them in the scene, and then by determining the size of their bounding boxes, in order to crop hand images.

5.2.1. Hand Localization

One way to localize hands in an image is to exploit Kinect SDK or middleware like OpenNI (or its derivatives). These libraries however do not provide accurate hand-sensing and are deprecated as well. Another way of localizing hands in an image is via detectors, possibly trained on hand images as in [58]. Yet, such strategies struggle to distinguish left and right hands, since they operate locally, thus lacking contextual information. To keep the framework generic, we decided not to employ specific hand sensing functionalities from Kinect–be it V1 or V2–or other more modern sensing devices. Instead, we localize the hand via the hand key-points obtained from *OpenPose*. This works well for any RGB camera and therefore does not require a specific platform (e.g., Kinect) for hand sensing.

OpenPose outputs 42 (21 per hand) hand key-points on each image. We observed that these key-points are more susceptible to jitter and misdetections than the skeleton key-points, particularly on the low resolution videos of the *Chalearn 2016* dataset. Therefore, we apply the same filter of Equations (1) and (2) to the raw hand key-points output by *OpenPose*. Then, we estimate the mean of all N_j detected hand key-point coordinates $\mathbf{p}^j$, to obtain:

$$\mathbf{p}^c = \frac{1}{N_j} \sum_{j=1}^{N_j} \mathbf{p}^j, \tag{5}$$

the hand center in the image.

5.2.2. Hand Bounding-Box Estimation

Once the hands are located in the image, the surrounding image patches must be cropped for gesture recognition. Since at run-time our gestures recognition system relies only on the RGB images (without depth), we develop two additional neural networks, f_l and f_r, to estimate each hand's bounding box size. These networks are analogous to the one described in Section 5.1.2. Following the scale-normalization approach, for each hand

we build a 54 dimensional augmented pose vector from 6 key-points. These augmented pose vectors ($\mathbf{x_l}$ and $\mathbf{x_r}$) are mapped to the ground-truth hands depth values (d_l and d_r) obtained from *OpenSign* dataset, through two independent neural networks:

$$\widetilde{d_l} = f_l(\mathbf{x_l}, d_l; \theta_l) \tag{6}$$

$$\widetilde{d_r} = f_r(\mathbf{x_r}, d_r; \theta_r). \tag{7}$$

In (6) and (7), f_l and f_r are 9-layer neural networks that optimize parameters θ_l and θ_r given augmented poses $\mathbf{x_l}$ and $\mathbf{x_r}$ and ground-truth depths d_l and d_r, to estimate depths $\widetilde{d_l}$ and $\widetilde{d_r}$. Mean squared error for f_l and f_r are 4.50×10^{-4} and 6.83×10^{-4}, respectively. The size of the each bounding box is inversely proportional to the corresponding depth ($\widetilde{d_l}$ or $\widetilde{d_r}$) obtained by applying (6) to the pure RGB images. The orientation of each bounding box is estimated from the inclination between corresponding forearm and horizon. The final outputs are the cropped images of the hands, $\mathbf{i_l}$ and $\mathbf{i_r}$. Now since our depth estimators f_n, f_l and f_r have been trained, we do not require explicit depth sensing either to normalize the skeleton or to estimate the hand bounding boxes.

6. Video Data Processing

Our proposed spatial attention module conceptually allows end-to-end training of the gestures. However, we train our network in multiple stages to speed-up the training process (the details of which are given in Section 8). Yet, this requires the videos to be processed step-by-step beforehand. This is done in four steps, that is, (1) 2D pose-estimation, (2) features extraction, (3) label-wise sorting and zero-padding and (4) train-ready data formulation. While prior 2D-pose estimation may be considered a compulsory step–even if the network is trained in an end-to-end fashion—the other steps can be integrated into the training algorithm.

6.1. Dynamic Features: Joints Velocities and Accelerations

As described in Section 5, our features of interest for gestures recognition are skeleton and hand images. The concept of augmented pose for scale-normalization has been detailed in Section 5.1.2. For dynamic gestures recognition, velocity and acceleration vectors from 8 upper-body joints, containing information about the dynamics of motion, are also appended to the pose vector $\mathbf{x_n}$ to form a new 129 components augmented pose $\mathbf{x_{dyn}}$. Inspired by [7], joint velocities and accelerations are computed as first and second derivatives of the scale-normalized joint coordinates. At each image k:

$$\dot{\mathbf{p}}_k^i = \mathbf{p}_{k+1}^i - \mathbf{p}_{k-1}^i \tag{8}$$

$$\ddot{\mathbf{p}}_k^i = \mathbf{p}_{k+2}^i + \mathbf{p}_{k-2}^i - 2\mathbf{p}_k^i. \tag{9}$$

The velocities and accelerations obtained from (8) and (9) are scaled by the video frame-rate to make values time-consistent, before appending them in the augmented pose vector $\mathbf{x_{dyn}}$. For every frame output by the skeleton filter of Section 5.1.1, scale-normalized augmented pose vectors $\mathbf{x_{dyn}}$ (as explained in Section 5.1.2) plus left $\mathbf{i_l}$ and right $\mathbf{i_r}$ hands cropped images (extracted as explained in Section 5.2) are appended in three individual arrays.

6.2. Train-Ready Data Formulation

The videos in the *Chalearn 2016* are randomly distributed. Once the features of interest ($\mathbf{i_l}$, $\mathbf{i_r}$ and $\mathbf{x_{dyn}}$) are extracted and saved in *.h5* files, we sort them with respect to their labels. It is natural to expect the dataset videos (previously sequences of images, now arrays of features) to be of different lengths. The average video length in this dataset is 32 frames, while we fix the length of each sequence to 40 images in our work. If the length of a sequence is less than 40, we pad zeros symmetrically at the start and end of the sequence. Alternatively, if the length is greater than 40, we perform symmetric trimming

of the sequence. Once the lengths of sequences are rectified (padded or trimmed), we append all corresponding sequences of a gesture label into a single array. At the end of this procedure, we are left with the 249 gestures in the *Chalearn 2016* dataset, along with an array of the ground-truth labels. Each feature of the combined augmented pose vectors is normalized to zero mean and unit variance, while for hand images we perform pixel-wise division by the maximum intensity value (e.g., 255). The label-wise sorting presented in this section is only necessary if one wants to train a network on selected gestures (as we will explain in Section 8). Otherwise, creating only a ground-truth label array should suffice.

7. Dynamic Gesture Recognition

To classify dynamic gestures, *StaDNet* learns to model the spatio-temporal dependencies of the input video sequences. As already explained in Sections 5.2 and 6.1, we obtain cropped hand images $\mathbf{i_l}$ and $\mathbf{i_r}$ as well as the augmented pose vector $\mathbf{x_{dyn}}$ for each frame in a video sequence. These features are aggregated in time through Long-Short Term Memory networks to detect dynamic gestures performed in the videos. However, we do not pass raw hand images, but extract image embeddings of size 1024 elements per hand. These image embeddings are extracted from the last fully connected layer of our static hand gesture detector and can be considered as rich latent space representations of hand gestures. This is done according to:

$$\begin{aligned} \mathbf{e_l}, \mathbf{p_l} &= g_{sta}(\mathbf{i_l}, \theta_{st}) \\ \mathbf{e_r}, \mathbf{p_r} &= g_{sta}(\mathbf{i_r}, \theta_{st}), \end{aligned} \tag{10}$$

with:

- g_{sta} the static hand gesture detector, which returns the frame-wise hand gesture class probabilities $\mathbf{p_{l,r}}$ and the embeddings vectors $\mathbf{e_{l,r}}$ from its last fully connected layers;
- θ_{st} the learned parameters of g_{sta}.

For each frame of a video sequence of length N, the obtained hand image embeddings $\mathbf{e_l}$, $\mathbf{e_r}$ and augmented pose vector $\mathbf{x_{dyn}}$ are subsequently fused in vector ψ, and then passed to stacked LSTMs followed by g_{dyn} network. This network outputs dynamic gestures probability p_{dyn} for each video:

$$\begin{aligned} \psi &= [\mathbf{e_l}; \mathbf{x_{dyn}}; \mathbf{e_r}] \\ p_{dyn} &= g_{dyn}(LSTMs(\sum_{i=1}^{N} \psi_i, \theta_{LSTMs}), \theta_{dyn}). \end{aligned} \tag{11}$$

The g_{dyn} network consists of a fully connected layer and a *softmax* layer which takes the output of LSTMs as input; θ_{LSTMs} and θ_{dyn} are model parameters to be learned for the detection of dynamic gestures, while p_{dyn} is the detected class probability obtained as output from the *softmax* layer. The illustration of our network is presented in Figure 4. We employ dropout regularization method between successive layers to prevent over-fitting and improve generalization, and batch-normalization to accelerate training.

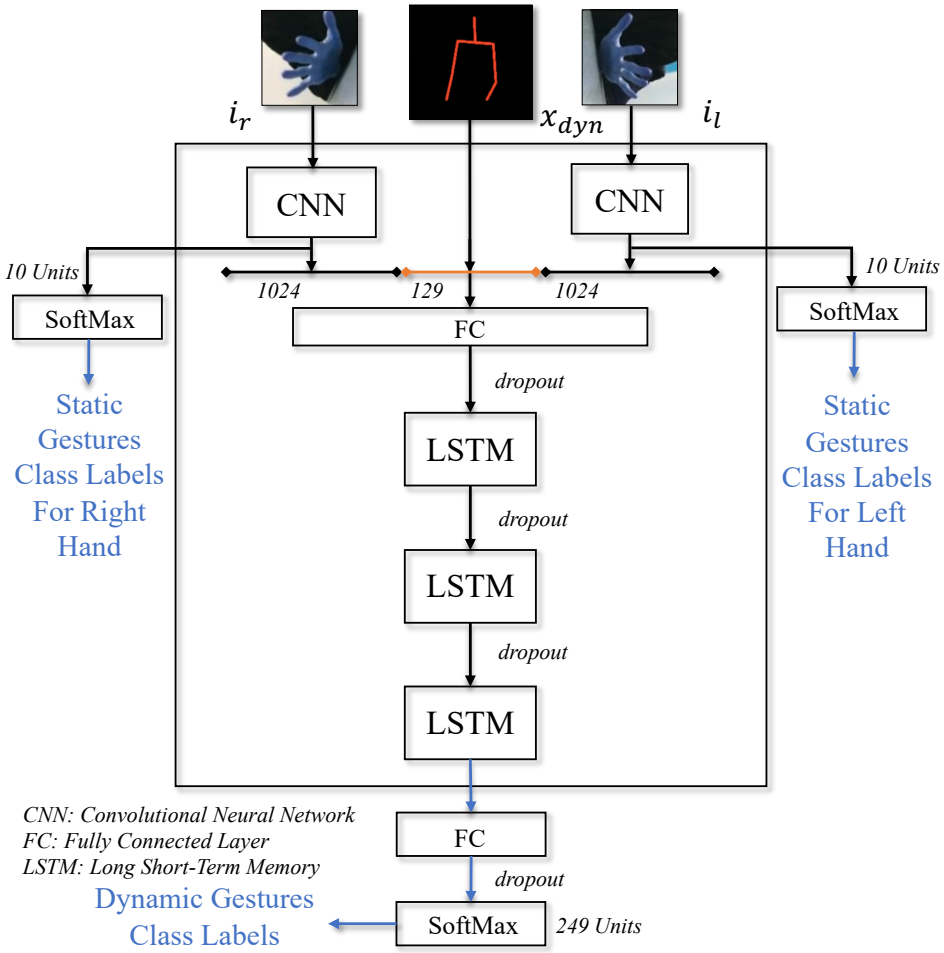

Figure 4. Illustration of *StaDNet* for static and dynamic gestures recognition. We perform intermediate fusion to combine hand image embeddings and augmented pose vector.

8. Training

The proposed network is trained on a computer with Intel© Core i7-6800K (3.4 GHz) CPU, dual Nvidia GeForce GTX 1080 GPUs, 64 GB system memory and Ubuntu 16.04 Operating system. The neural network is designed, trained and evaluated in Python-Keras with tensorflow back-end, while skeleton extraction with *OpenPose* is performed in C++.

The *Chalearn 2016* dataset has 35,875 videos in the provided training set, with only the top 47 gestures (arranged in descending order of the number of samples) representing 34% of all videos. The numbers of videos in the provided validation and test sets are 5784 and 6271, respectively. The distribution of train, validation and test data in our work is slightly different from the approach proposed in the challenge. We combine and shuffle the provided train, validation and test sets together, leading to 47,930 total videos. For weight initialization, 12,210 training videos of 47 gestures are utilized to perform pre-training with a validation split of 0.2. We subsequently proceed to train our network for all 249 gestures on 35,930 videos, initializing the parameters with the pre-trained model weights. In this work, we utilize the *Holdout* cross-validation method, which aligns with the original exercise of the *Chalearn 2016* challenge. Thus, we optimize the hyper-parameters on the validation data of 6000 videos, while the results are presented on the test data of the remaining 6000 videos.

As already explained in Section 3, we utilize only 1247 videos for 14 correctly performed dynamic gestures from the *Praxis Cognitive Assessment Dataset*. Given the small size of this dataset, we adapt the network hyper-parameters to avoid over-fitting.

9. Results

For the *Chalearn 2016* dataset, the proposed network is initially trained on 47 gestures with a low learning rate of 1×10^{-5}. After approximately 66,000 epochs, a top-1 validation accuracy of 95.45% is obtained. The parameters learned for 47 gestures are employed to initialize weights for complete data training for 249 gestures as previously described. The network is trained in four phases. In the first phase, we perform weights initialization, inspired by the *transfer learning* concept of deep networks, by replacing the classification layer (with *softmax* activation function) by the same with output number of neurons corresponding to the number of class labels in the dataset. In our case, we replace the *softmax* layer in the trained network for 47 gestures plus the FC layer immediately preceding it. The proposed model is trained for 249 gestures classes with a learning rate of 1×10^{-3} and a decay value of 1×10^{-3} with *Adam* optimizer. The early iterations are performed with all layers of the network locked except the newly added FC and *softmax* layers. As the number of epochs increases, we successively unlock the network layers from the bottom (deep layers).

In the second phase, network layers until the last LSTM block are unlocked. All LSTM blocks and then the complete model are unlocked, respectively in the third and fourth phase. By approximately 2700 epochs, our network achieves 86.69% top-1 validation accuracy for all 249 gestures and 86.75% top-1 test accuracy, surpassing the state-of-art methods on this dataset. The prediction time for each video sample is 57.17 ms, excluding pre-processing of the video frames. Thus, we are confident that the online dynamic gesture recognition can be achieved in interaction time. The training curve of the complete model is shown in Figure 5 while the confusion matrix/heat-map with evaluations on test set is shown in Figure 6. Our results on the *Chalearn 2016* dataset are compared with the reported state-of-the-art in Table 1.

Table 1. Comparison of the reported results with ours on the Chalearn 2016. The challenge results are published in [59].

Method	Valid %	Test %
StaDNet (ours)	86.69	86.75
FOANet [55]	80.96	82.07
Miao et al. [38] (ASU)	64.40	67.71
SYSU_IEEE	59.70	67.02
Lostoy	62.02	65.97
Wang et al. [60] (AMRL)	60.81	65.59

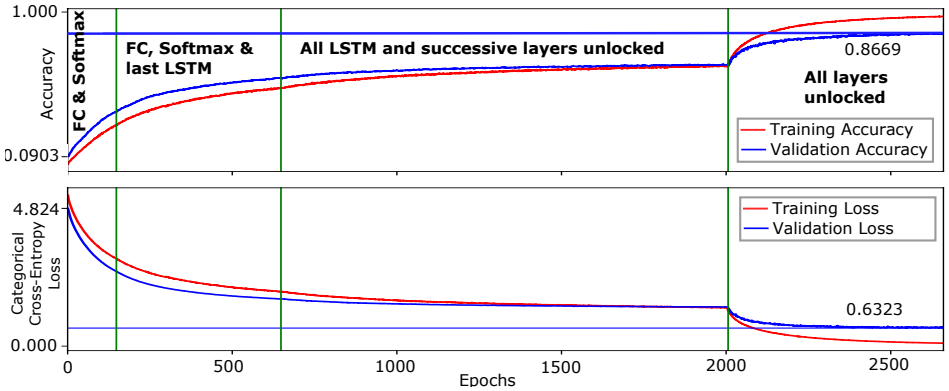

Figure 5. Training curves of the proposed Convolutional Neural Network (CNN)–Long Short term Memory (LSTM) network for all 249 gestures of the Chalearn 2016. The network is trained in four phases, distinguished by the vertical lines.

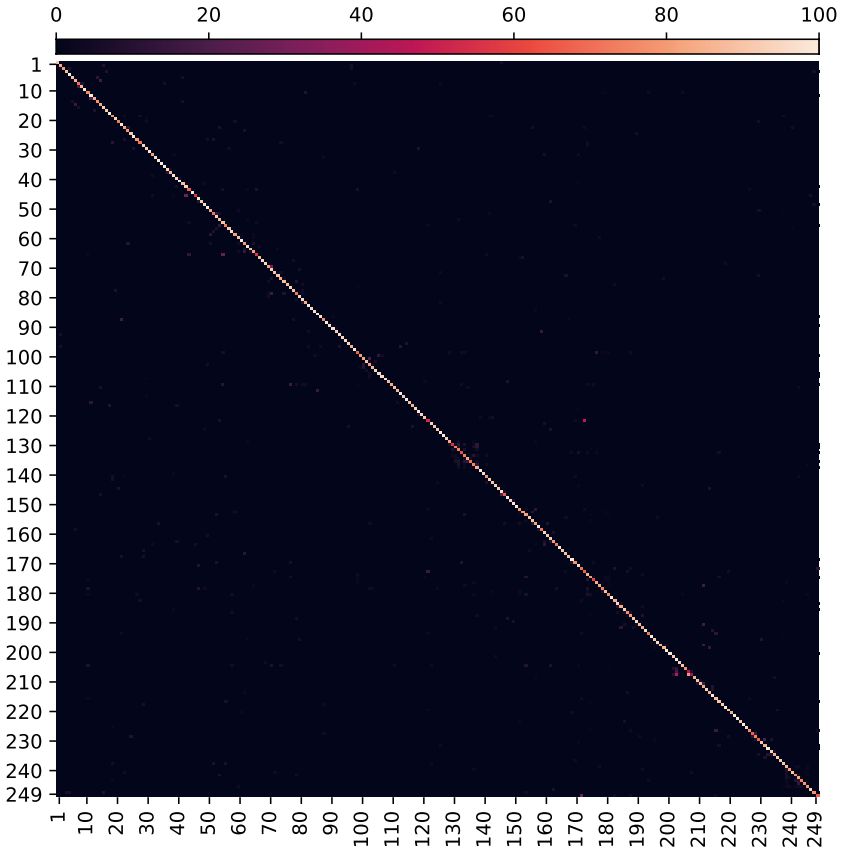

Figure 6. Illustration of the confusion matrix/heat-map of *StaDNet* evaluated on test set of the Chalearn 2016 isolated gestures recognition dataset. It is evident that most samples in the test set are recognized with high accuracy for all 249 gestures (diagonal entries, 86.75% overall).

Inspecting the training curves, we observe that the network is progressing towards slight over-fitting in the fourth phase when all network layers are unlocked. Specifically, the first *time-distributed FC* layer is considered the culprit for this phenomenon. Although we already have a dropout layer immediately after this layer, with dropout rate equaling 0.85, we skip to further dive deeper to rectify this. However, it is assumed that substitution of this layer with the strategy of *pose-driven temporal attention* [53] or with the *adaptive hidden layer* [61], may help reduce this undesirable phenomenon and ultimately further improve results. Moreover, recent studies argue that data augmentation that is, the technique of perturbing data without altering class labels, are able to greatly improve model robustness and generalization performance [62]. As we do not use any data augmentation on the videos in model training for dynamic gestures, doing the contrary might help to reduce over-fitting here.

For the *Praxis* dataset, the optimizer and values of learning rate and decay, are the same as for the *Chalearn 2016* dataset. The hyper-parameters including number of neurons in FC layers plus hidden and cell states of LSTM blocks are (reduced) adapted to avoid over-fitting. Our model obtains 99.6% top-1 test accuracy on 501 samples. The training curve of the *StaDNet* on the *Praxis* dataset is shown in Figure 7, the normalized confusion matrix on this dataset is shown in Figure 8, while the comparison of the results with the state-of-the-art is shown in Table 2. We also quantify the performance of our static hand gesture detector on a test set of 4190 hand images. The overall top-1 test accuracy is found to be 98.9%. The normalized confusion matrix for 10 static hand gestures is shown in Figure 9.

Table 2. Comparison of dynamic gestures recognition results on the Praxis gestures dataset; [56] also used a similar CNN-LSTM network.

System	Accuracy % (Dynamic Gestures)
StaDNet (ours)	99.60
Negin et al. [56]	76.61

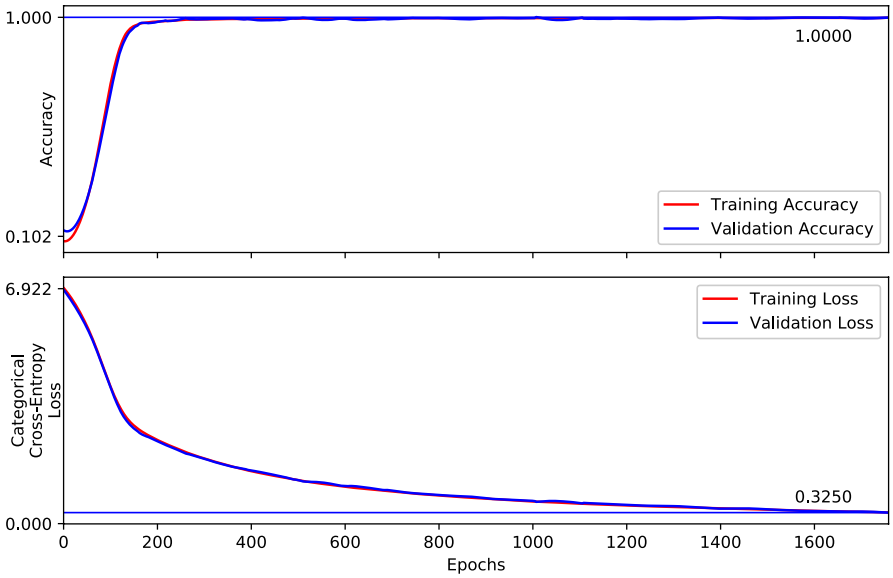

Figure 7. Training curves of *StaDNet* on the Praxis gesture dataset.

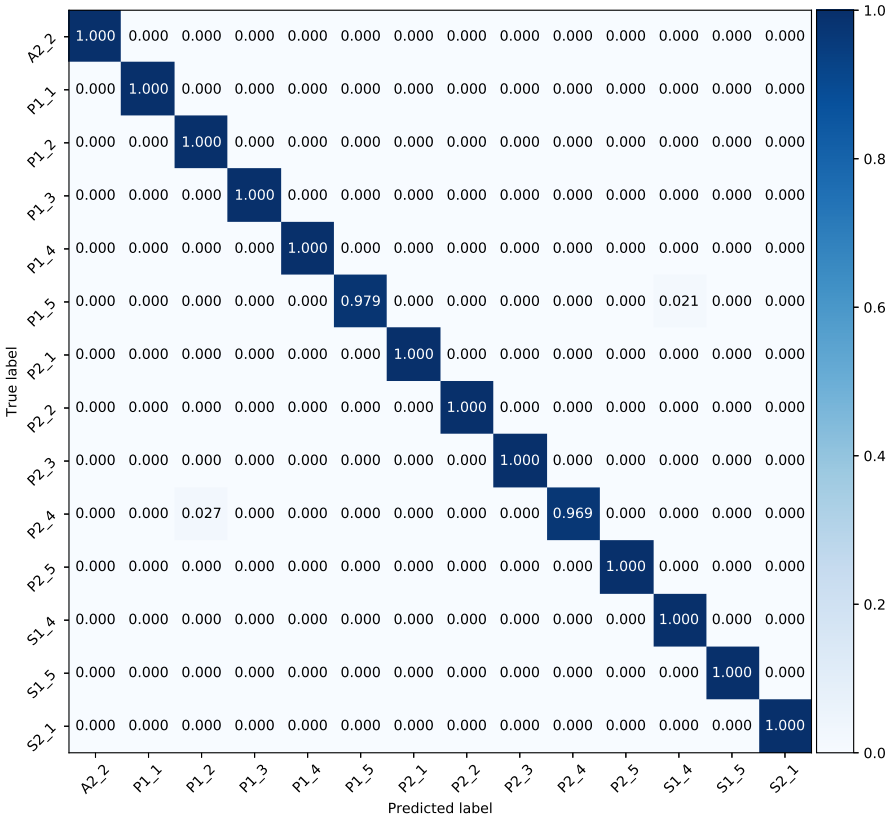

Figure 8. Normalized confusion matrix of the proposed model evaluated on test set of the Praxis dataset.

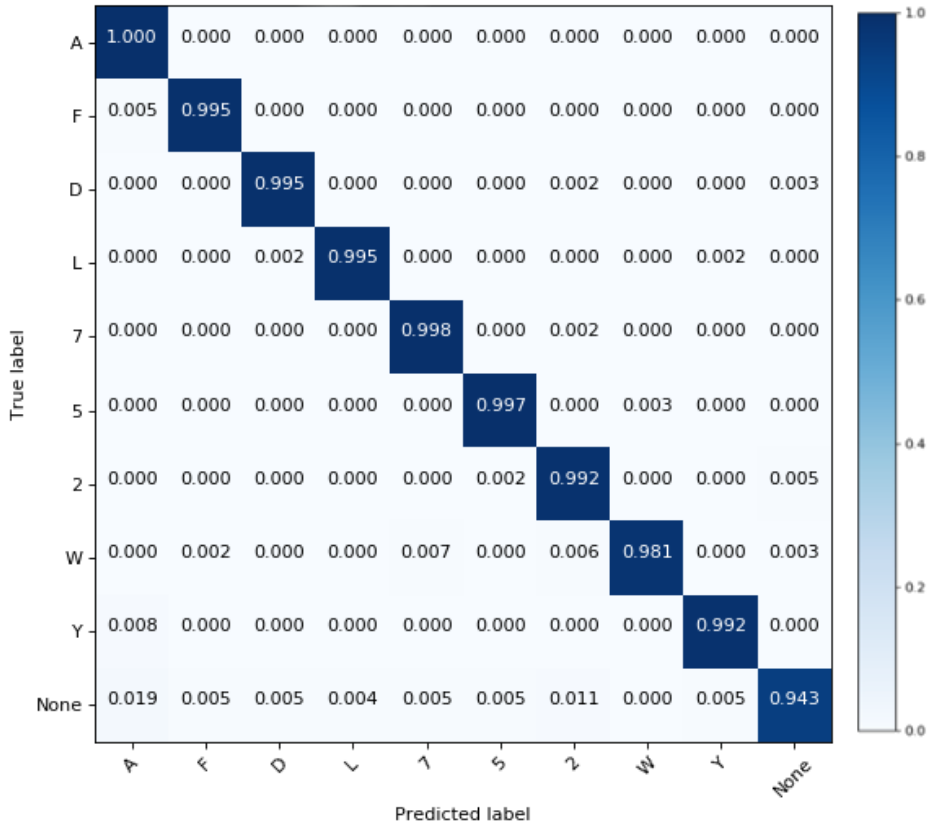

Figure 9. Normalized confusion matrix for our static hand gesture detector quantified on test-set of *OpenSign*. This figure is taken from [11] with the authors' permission.

We devised robotic experiments for gesture-controlled safe human-robot interaction tasks as already presented in [11]. These are preliminary experiments that allow the human operator to communicate with the robot through static hand gestures in real-time while dynamic gestures integration is yet to be done. The experiments were performed on BAZAR robot [63] which has two Kuka LWR 4+ arms with two Shadow Dexterous Hands attached at the end-effectors. We exploited *OpenPHRI* [64], which is an open-source library, to control the robot while corroborating safety of the human operator. A finite state machine is developed to control behavior of the robot which is determined by the sensory information for example, hand gestures, distance of the human operator from the robot, joint-torque sensing and so forth. The experiment is decomposed into two phases: (1) a teaching by demonstration phase, where the user manually guides the robot to a set of waypoints and (2) a replay phase, where the robot autonomously goes to every recorded waypoint to perform a given task, here force control. A video of the experiment is available online (http://youtu.be/lB5vXc8LMnk, accessed on 22 March 2021) and snapshots are given in Figure 10.

Figure 10. Snapshots of our gesture-controlled safe human-robot interaction experiment taken from [11] with the authors' permission. The human operator manually guides the robot to waypoints in the workspace then asks the robot to *record* them through a gesture. The human operator can transmit other commands to the robot like *replay*, *stop*, *resume*, *reteach*, and so forth with only hand gestures.

10. Conclusions

In this paper, a unified framework for simultaneous recognition of static hands and dynamic upper-body gestures, *StaDNet* is proposed. A novel idea of learning-based depth estimator is also presented, which predicts the distance of the person and his/her hands, exploiting only the upper-body 2D skeleton coordinates. By virtue of this feature, monocular images are sufficient and the proposed framework does not require depth sensing. Thus, the use of *StaDNet* for gestures detection is not limited to any specialized camera and can work with most conventional RGB cameras. Monocular images are indeed sensitive to the changing lighting conditions and might fail to work in extreme conditions for example, during sand blasting operation in the industry or during fog and rain in the outdoors. To develop immunity against such lighting corruptions, data augmentation strategies such as [65] can be exploited. One might argue that employing HSV or HSL color models instead of RGB might be more appropriate to deal with changing ambient light conditions. However, *StaDNet* actually relies on *OpenPose* for skeleton extraction and on the hand gesture detector from our previous work [11]. *OpenPose* is the state-of-art in skeleton extraction from monocular camera and takes RGB images as input. Furthermore, our static hand gesture also takes RGB images as input and performs well with 98.9% top-1 test accuracy on 10 static hand gestures as we show in Figure 9. In spite of that, we are aware that HSV or HSL had been commonly used for hand segmentation in the literature by thresholding the values of Hue, Saturation and Value/Lightness. This indeed intrigues our eagerness to train and compare the performance of deep models for hand gesture detector in this color model/space, which we plan to do in our future work.

Our pose-driven hard spatial attention mechanism directs the focus of *StaDNet* on upper-body pose to model large-scale body movements of the limbs and, on the hand images for subtle hand/fingers movements. This enables *StaDNet* to out-score the existing approaches on the Chalearn 2016 dataset. The presented weight initialization strategy addresses the imbalance in class distribution in the Chalearn 2016 dataset, thus facilitates parameters optimization for all 249 gestures. Our static gestures detector outputs the predicted label frame-wise at approximately 21 fps with the state-of-the-art recognition accuracy. However, class recognition for dynamic gestures is performed on isolated gestures videos, executed by an individual in the scene. We plan to extend this work for continuous dynamic gestures recognition to demonstrate its utility in human-machine interaction. This can be achieved in one way by developing a binary motion detector to detect start and end instances of the gestures. Although a multi-stage training strategy is presented, we envision an end-to-end training approach for online learning of new gestures.

Author Contributions: Conceptualization, O.M., S.R. and A.C.; methodology, O.M.; software, O.M.; validation, O.M.; formal analysis, O.M.; investigation, O.M.; resources, S.R., A.C.; data curation, O.M.; writing—original draft preparation, O.M.; writing—review and editing, O.M., S.R. and A.C.; visualization, O.M.; supervision, A.C. and S.R.; project administration, A.C. and S.R.; funding acquisition, A.C. All authors have read and agreed to the published version of the manuscript.

Funding: The research presented in this article was carried out as parts of the SOPHIA and the OpenDR projects, which have received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 871237 and 871449 respectively.

Data Availability Statement: The datasets used in this research are publicly available on their respective websites.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, G.; Wu, H.; Jiang, G.; Xu, S.; Liu, H. Dynamic gesture recognition in the internet of things. *IEEE Access* **2018**, *7*, 23713–23724. [CrossRef]
2. Kofman, J.; Wu, X.; Luu, T.J.; Verma, S. Teleoperation of a robot manipulator using a vision-based human-robot interface. *IEEE Trans. Ind. Electron.* **2005**, *52*, 1206–1219. [CrossRef]

3. Tölgyessy, M.; Hubinskỳ, P.; Chovanec, L.; Duchoň, F.; Babinec, A. Controlling a group of robots to perform a common task by gestures only. *Int. J. Imaging Robot.* **2017**, *17*, 1–13.
4. Tölgyessy, M.; Dekan, M.; Duchoň, F.; Rodina, J.; Hubinskỳ, P.; Chovanec, L. Foundations of visual linear human–robot interaction via pointing gesture navigation. *Int. J. Soc. Robot.* **2017**, *9*, 509–523. [CrossRef]
5. Jung, P.G.; Lim, G.; Kim, S.; Kong, K. A wearable gesture recognition device for detecting muscular activities based on air-pressure sensors. *IEEE Trans. Ind. Inform.* **2015**, *11*, 485–494. [CrossRef]
6. Park, H.S.; Jung, D.J.; Kim, H.J. Vision-based Game Interface using Human Gesture. In Proceedings of the Pacific-Rim Symposium on Image and Video Technology, Hsinchu, Taiwan, 10–13 December 2006; pp. 662–671.
7. Neverova, N.; Wolf, C.; Taylor, G.W.; Nebout, F. Multi-scale Deep Learning for Gesture Detection and Localization. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 474–490.
8. Gleeson, B.; MacLean, K.; Haddadi, A.; Croft, E.; Alcazar, J. Gestures for industry intuitive human-robot communication from human observation. In Proceedings of the 2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI), Tokyo, Japan, 3–6 February 2013; pp. 349–356.
9. Mazhar, O. OpenSign-Kinect V2 Hand Gesture Data-American Sign Language. *Mendeley Data* **2019**. [CrossRef]
10. Starner, T.; Weaver, J.; Pentland, A. Real-time american sign language recognition using desk and wearable computer based video. *IEEE Trans. Pattern Anal. Mach. Intell.* **1998**, *20*, 1371–1375. [CrossRef]
11. Mazhar, O.; Navarro, B.; Ramdani, S.; Passama, R.; Cherubini, A. A Real-time Human-Robot Interaction Framework with Robust Background Invariant Hand Gesture Detection. *Robot. Comput. Integr. Manuf.* **2019**, *60*, 34–48. [CrossRef]
12. Rensink, R.A. The Dynamic Representation of Scenes. *Vis. Cogn.* **2000**, *7*, 17–42. [CrossRef]
13. Neto, P.; Pereira, D.; Pires, J.N.; Moreira, A.P. Real-time and continuous hand gesture spotting: An approach based on artificial neural networks. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013; pp. 178–183.
14. Wong, W.K.; Juwono, F.H.; Khoo, B.T.T. Multi-Features Capacitive Hand Gesture Recognition Sensor: A Machine Learning Approach. *IEEE Sensors J.* **2021**, *21*, 8441–8450. [CrossRef]
15. Zhu, C.; Sheng, W. Motion-and location-based online human daily activity recognition. *Pervasive Mob. Comput.* **2011**, *7*, 256–269. [CrossRef]
16. Laptev, I. On Space-time Interest Points. *Int. J. Comput. Vis.* **2005**, *64*, 107–123. [CrossRef]
17. Dollár, P.; Rabaud, V.; Cottrell, G.; Belongie, S. Behavior Recognition via Sparse Spatio-Temporal Features. In Proceedings of the 2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, Beijing, China, 15–16 October 2005; pp. 65–72.
18. Wang, H.; Ullah, M.M.; Klaser, A.; Laptev, I.; Schmid, C. Evaluation of local spatio-temporal features for action recognition. In Proceedings of the Bmvc 2009-British Machine Vision Conference, London, UK, 7–10 September 2009.
19. Wang, H.; Kläser, A.; Schmid, C.; Liu, C. Action Recognition by Dense Trajectories. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Colorado Springs, CO, USA, 20–25 June 2011; pp. 3169–3176.
20. Wang, H.; Schmid, C. Action Recognition with Improved Trajectories. In Proceedings of the 2013 IEEE International Conference on Computer Vision, Sydney, Australia, 1–8 December 2013; pp. 3551–3558.
21. Wang, H.; Oneata, D.; Verbeek, J.; Schmid, C. A Robust and Efficient Video Representation for Action Recognition. *Int. J. Comput. Vis.* **2016**, *119*, 219–238. [CrossRef]
22. Sánchez, J.; Perronnin, F.; Mensink, T.; Verbeek, J. Image classification with the Fisher Vector: Theory and Practice. *Int. J. Comput. Vis.* **2013**, *105*, 222–245. [CrossRef]
23. Kantorov, V.; Laptev, I. Efficient Feature Extraction, Encoding and Classification for Action Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 2593–2600.
24. Wu, D.; Zhu, F.; Shao, L. One shot learning gesture recognition from rgbd images. In Proceedings of the 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Providence, RI, USA, 16–21 June 2012; pp. 7–12.
25. Otsu, N. A threshold selection method from gray-level histograms. *IEEE Trans. Syst. Man, Cybern.* **1979**, *9*, 62–66. [CrossRef]
26. Fanello, S.R.; Gori, I.; Metta, G.; Odone, F. Keep It Simple And Sparse: Real-Time Action Recognition. *J. Mach. Learn. Res.* **2013**, *14*, 2617–2640.
27. Konečnỳ, J.; Hagara, M. One-shot-learning gesture recognition using hog-hof features. *J. Mach. Learn. Res.* **2014**, *15*, 2513–2532.
28. Wan, J.; Ruan, Q.; Li, W.; Deng, S. One-shot learning gesture recognition from RGB-D data using bag of features. *J. Mach. Learn. Res.* **2013**, *14*, 2549–2582.
29. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
30. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556.
31. Ji, S.; Xu, W.; Yang, M.; Yu, K. 3D Convolutional Neural Networks for Human Action Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *35*, 221–231. [CrossRef] [PubMed]
32. Baccouche, M.; Mamalet, F.; Wolf, C.; Garcia, C.; Baskurt, A. Sequential Deep Learning for Human Action Recognition. In Proceedings of the International Workshop on Human Behavior Understanding, Amsterdam, The Netherlands, 15–19 October 2011; pp. 29–39.
33. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]

34. Karpathy, A.; Toderici, G.; Shetty, S.; Leung, T.; Sukthankar, R.; Li, F.F. Large-scale Video Classification with Convolutional Neural Networks. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1725–1732.
35. Xu, H.; Kang, B.; Sun, X.; Feng, J.; Saenko, K.; Darrell, T. Similarity r-c3d for few-shot temporal activity detection. *arXiv* **2018**, arXiv:1812.10000.
36. Zheng, J.; Cao, X.; Zhang, B.; Zhen, X.; Su, X. Deep ensemble machine for video classification. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *30*, 553–565. [CrossRef] [PubMed]
37. Neverova, N.; Wolf, C.; Paci, G.; Sommavilla, G.; Taylor, G.; Nebout, F. A Multi-scale Approach to Gesture Detection and Recognition. In Proceedings of the IEEE International Conference on Computer Vision Workshops, Sydney, Australia, 2–3 December 2013; pp. 484–491.
38. Miao, Q.; Li, Y.; Ouyang, W.; Ma, Z.; Xu, X.; Shi, W.; Cao, X. Multimodal Gesture Recognition based on the ResC3D Network. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 3047–3055.
39. Wan, J.; Zhao, Y.; Zhou, S.; Guyon, I.; Escalera, S.; Li, S.Z. Chalearn Looking at People RGB-D Isolated and Continuous Datasets for Gesture Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Las Vegas, NV, USA, 27–30 June 2016; pp. 56–64.
40. Tran, D.; Ray, J.; Shou, Z.; Chang, S.F.; Paluri, M. ConvNet Architecture Search for Spatiotemporal Feature Learning. *arXiv* **2017**, arXiv:1708.05038.
41. Simonyan, K.; Zisserman, A. Two-stream Convolutional Networks for Action Recognition in Videos. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 568–576.
42. Goodale, M.A.; Milner, A.D. Separate Visual Pathways for Perception and Action. *Trends Neurosci.* **1992**, *15*, 20–25. [CrossRef]
43. Wang, L.; Qiao, Y.; Tang, X. Action Recognition with Trajectory-Pooled Deep-Convolutional Descriptors. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 4305–4314.
44. Yue-Hei Ng, J.; Hausknecht, M.; Vijayanarasimhan, S.; Vinyals, O.; Monga, R.; Toderici, G. Beyond Short Snippets: Deep Networks for Video Classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 4694–4702.
45. Donahue, J.; Anne Hendricks, L.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Saenko, K.; Darrell, T. Long-Term Recurrent Convolutional Networks for Visual Recognition and Description. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 2625–2634.
46. Xingjian, S.; Chen, Z.; Wang, H.; Yeung, D.Y.; Wong, W.K.; Woo, W.c. Convolutional LSTM network: A Machine Learning Approach for Precipitation Nowcasting. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 802–810.
47. Zhu, G.; Zhang, L.; Yang, L.; Mei, L.; Shah, S.A.A.; Bennamoun, M.; Shen, P. Redundancy and Attention in Convolutional LSTM for Gesture Recognition. *IEEE Trans. Neural Networks Learn. Syst.* **2019**. [CrossRef]
48. Yeung, S.; Russakovsky, O.; Jin, N.; Andriluka, M.; Mori, G.; Fei-Fei, L. Every Moment Counts: Dense Detailed Labeling of Actions in Complex Videos. *Int. J. Comput. Vis.* **2018**, *126*, 375–389. [CrossRef]
49. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NA, USA, 3–6 December 2012; pp. 1097–1105.
50. Idrees, H.; Zamir, A.R.; Jiang, Y.G.; Gorban, A.; Laptev, I.; Sukthankar, R.; Shah, M. The THUMOS Challenge on Action Recognition for Videos "In the Wild". *Comput. Vis. Image Underst.* **2017**, *155*, 1–23. [CrossRef]
51. Mnih, V.; Heess, N.; Graves, A.; Kavukcuoglu, K. Recurrent Models of Visual Attention. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 2204–2212.
52. Baradel, F.; Wolf, C.; Mille, J. Pose-conditioned Spatio-temporal Attention for Human Action Recognition. *arXiv* **2017**, arXiv:1703.10106.
53. Baradel, F.; Wolf, C.; Mille, J. Human Action Recognition: Pose-based Attention Draws Focus to Hands. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 604–613.
54. Zheng, Z.; An, G.; Wu, D.; Ruan, Q. Global and Local Knowledge-Aware Attention Network for Action Recognition. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *31*, 334–347. [CrossRef]
55. Narayana, P.; Beveridge, R.; Draper, B.A. Gesture Recognition: Focus on the Hands. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 5235–5244.
56. Negin, F.; Rodriguez, P.; Koperski, M.; Kerboua, A.; Gonzàlez, J.; Bourgeois, J.; Chapoulie, E.; Robert, P.; Bremond, F. PRAXIS: Towards Automatic Cognitive Assessment Using Gesture Recognition. *Expert Syst. Appl.* **2018**, *106*, 21–35. [CrossRef]
57. Cao, Z.; Simon, T.; Wei, S.E.; Sheikh, Y. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
58. Panteleris, P.; Oikonomidis, I.; Argyros, A. Using a Single RGB Frame for Real time 3D Hand Pose Estimation in the Wild. In Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 12–15 March 2018; pp. 436–445.

59. Wan, J.; Escalera, S.; Anbarjafari, G.; Jair Escalante, H.; Baró, X.; Guyon, I.; Madadi, M.; Allik, J.; Gorbova, J.; Lin, C.; et al. Results and Analysis of Chalearn LAP Multi-modal Isolated and Continuous Gesture Recognition, and Real versus Fake Expressed Emotions Challenges. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 3189–3197.
60. Wang, H.; Wang, P.; Song, Z.; Li, W. Large-scale Multimodal Gesture Recognition using Heterogeneous Networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 3129–3137.
61. Hu, T.K.; Lin, Y.Y.; Hsiu, P.C. Learning Adaptive Hidden Layers for Mobile Gesture Recognition. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
62. Hendrycks, D.; Basart, S.; Mu, N.; Kadavath, S.; Wang, F.; Dorundo, E.; Desai, R.; Zhu, T.; Parajuli, S.; Guo, M.; et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. *arXiv* **2020**, arXiv:2006.16241.
63. Cherubini, A.; Passama, R.; Navarro, B.; Sorour, M.; Khelloufi, A.; Mazhar, O.; Tarbouriech, S.; Zhu, J.; Tempier, O.; Crosnier, A.; et al. A collaborative robot for the factory of the future: Bazar. *Int. J. Adv. Manuf. Technol.* **2019**, *105*, 3643–3659. [CrossRef]
64. Navarro, B.; Fonte, A.; Fraisse, P.; Poisson, G.; Cherubini, A. In pursuit of safety: An open-source library for physical human-robot interaction. *IEEE Robot. Autom. Mag.* **2018**, *25*, 39–50. [CrossRef]
65. Mazhar, O.; Kober, J. Random Shadows and Highlights: A new data augmentation method for extreme lighting conditions. *arXiv* **2021**, arXiv:2101.05361.