Pattern Recognition 76 (2018) 80–94

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Convolutional Neural Networks and Long Short-Term Memory for skeleton-based human activity and hand gesture recognition

Juan C. Núñez, Raúl Cabido, Juan J. Pantrigo, Antonio S. Montemayor, José F. Vélez*

Universidad Rey Juan Carlos, Madrid, Spain

ARTICLE INFO

Article history:
Received 2 December 2016
Revised 6 October 2017
Accepted 24 October 2017

Keywords:
Deep learning
Convolutional Neural Network
Recurrent neural network
Long Short-Term Memory
Human activity recognition
Hand gesture recognition
Real-time

ABSTRACT

In this work, we address human activity and hand gesture recognition problems using 3D data sequences obtained from full-body and hand skeletons, respectively. To this aim, we propose a deep learning-based approach for temporal 3D pose recognition problems based on a combination of a Convolutional Neural Network (CNN) and a Long Short-Term Memory (LSTM) recurrent network. We also present a two-stage training strategy which firstly focuses on CNN training and, secondly, adjusts the full method (CNN+LSTM). Experimental testing demonstrated that our training method obtains better results than a single-stage training strategy. Additionally, we propose a data augmentation method that has also been validated experimentally. Finally, we perform an extensive experimental study on publicly available data benchmarks. The results obtained show how the proposed approach reaches state-of-the-art performance when compared to the methods identified in the literature. The best results were obtained for small datasets, where the proposed data augmentation strategy has greater impact.

1. Introduction

Vision-based human action recognition concerns the task of automatically interpreting an image sequence to decide what action or activity is being performed by the subjects in the scene. It is a relevant topic in computer vision, with practical applications such as video surveillance, human-computer interaction, gaming, sports arbitration, sports training, smart homes, life-care systems, among many others [1,2]. Due to the huge possibilities for practical application, human activity recognition problems have received the attention of researchers in the fields of computer vision, artificial intelligence and machine learning. Researchers of the field organize different contests as, for example, the ChaLearn Looking at People challenge [3], and provide large datasets as NTU RGB+D [4]. As a consequence, it is possible to find a significant number of related works in the literature describing an extensive variety of methods and strategies to deal with this problem. In particular, in recent years, deep neural networks have been successfully applied in human action recognition problems as a suitable approach when relatively large datasets are available.

The toolkits of many affordable RGBD devices allow the acquisition of 3D data at interactive framerates. These devices can be used to capture human movements or hand poses, offering 3D coordinates of the joints as skeletons [5]. These skeletons can capture the evolution of the pose of a human body or hand and, therefore, they can be used to classify the activities or gestures performed by subjects in the area.

In this paper, we propose the combination of a Convolutional Neural Network (CNN) and a Long-Short Term Memory (LSTM) recurrent network for handling time series of 3D coordinates of skeleton keypoints. We have tested our proposal on six publicly available datasets.

Fig. 1summarizes the proposed system, in which the input data at each time step is presented to the CNN+LSTM network. The CNN is mainly responsible for capturing relevant features from the 3D data input on every time step, while the LSTM takes into account the time evolution of the 3D data series. Finally, the CNN+LSTM model generates a classification result for the presented model sequence.

An important contribution of this paper is that the proposed network architecture does not need to be adapted to the type of activity or gesture to be recognized as well as to the geometry of the 3D time-series data as input. Nonetheless, it obtains results that are competitive to previous works that need to make assumptions on those. Additionally, we present a data augmentation method that allows us to solve the problem of overfitting. The proposed augmentation techniques provide a significant performance improvement when applied to small datasets. Finally, it is also important to note that the proposed network architecture

* Corresponding author.
E-mail address: jose.velez@urjc.es (J.F. Vélez).

https://doi.org/10.1016/j.patcog.2017.10.033
0031-3203/

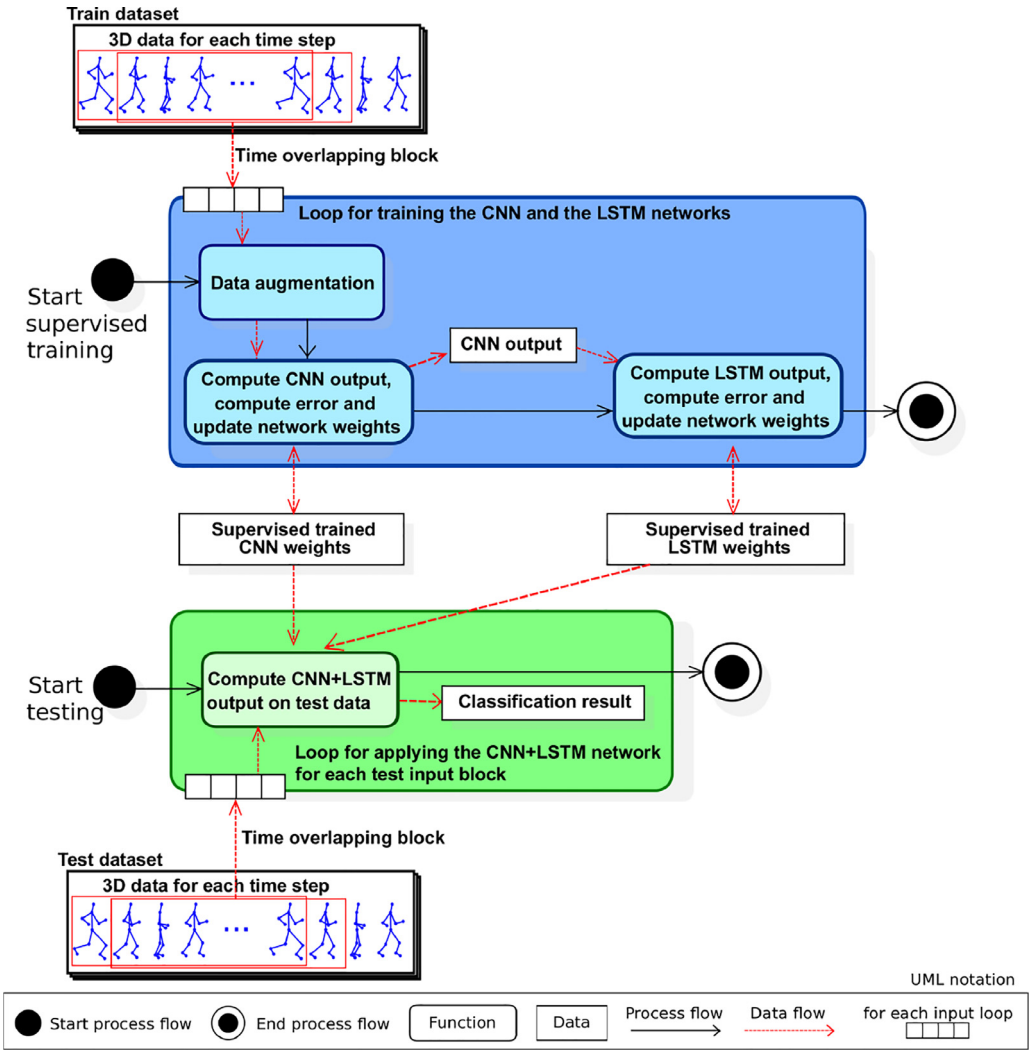

Fig. 1. UML Activity Diagram for the proposed system. The diagram shows two process flows, the upper one for the training process and the lower one for the testing process.

is lightweight enough to allow real time processing on embedded platforms.

The rest of the paper is organized as follows: Section 2 deals with the presentation of the related works in this research area. Section 3 details the proposed neural network architecture. Section 4 presents and discusses the results obtained. Finally, conclusions are outlined in Section 5.

2. Related work

This section reviews the state-of-the-art methods for the considered problems, specifically, in skeleton-based human activity recognition and hand gesture recognition problems.

2.1. Skeleton-based human activity recognition

Recently, a significant number of approaches to the problem of human activity recognition have been proposed, based on human skeleton kinematics [6]. However, although the success of deep learning techniques in the computer vision community started around 2012, most of the related works in this topic are based on classic machine learning methods. Xia et al. [7] presented an approach which uses histograms of 3D joint locations (HOJ3D) as a compact representation of postures and, afterwards, these HOJ3D are reprojected using linear decomposition analysis and clustered into k posture visual words, representing the prototypical poses of actions. Zanfir et al. [8] proposed a moving pose descriptor that considers position, speed and acceleration of the human body joints within a short time window around the current frame, and they established the classification using a modification of the k-nearest neighbour classifier. The work of Devanne et al. [9] is based on the use of trajectories, which consists of a motion channel corresponding to the evolution of the 3D position of all joint coordinates within frames of action sequence. The classification stage is addressed by a k-nearest neighbour classifier. Chrungoo et al. [10] represented the action as a histogram of direction vectors, obtaining a scale and speed invariant descriptor which is also computationally efficient. Vemulapalli et al. [11] uses geometric relationships between several body parts by means of rotations and translations in the 3D space. In this representation, human actions can be modelled as curves in a Lie group, and then, these action curves are mapped as a vector space. The classification stage includes a combination of dynamic time warping, Fourier temporal pyramid representation and linear SVM. Evangelidis et al. [12] propose a local skeleton descriptor to encode the relative position of joint quadruples and a Fisher vector representation to describe the skeletal quads contained in a (sub)action, based on a learned GMM and multi-level representation of those vectors. Zhang & Parker

[13] presented a bio-inspired predictive orientation decomposition approach to construct representations of people from 3D skeleton trajectories. Tao & Vidal [14] introduced a principled feature learning framework for learning motion patterns called Moving Poselet, and an algorithm for learning them. Coppola et al. [15] addressed the context of Ambient Assisted Living with a method based on qualitative trajectories to represent human actions that are learned and classified using HMMs. Ding et al. [16] introduced an action decomposition method into a sequence of atomic actions (known as actionlets) and the concept of the Spatio-Temporal Feature Chain that depicts the characteristic parameters of temporal sequential patterns. Ben Amor et al. [17] studied the evolution of the skeletons' shapes as trajectories on Kendall's shape manifold. Zhu et al. [18] modeled human actions by a sequence of key poses and atomic motions and uses an online classification method with the variable-length maximal entropy Markov model. A significant contribution of this paper is that it does not need to determine the start and end points of each human action previously. Cippitelli et al. [19] developed a system able to extract key poses to compose a feature vector which are classified with a multiclass Support Vector Machine. Wang et al. [20] addressed the action recognition task by mining a set of key-pose-motifs for each action class. A key-pose-motif is represented as a dictionary, containing a set of ordered poses, which are required to be close, but not adjacent, in the action sequences. Then, a sequence is classified by matching it to the motifs of each class and selecting the class that maximizes the matching score. Wang et al. [21] proposed a representation for action-snippets, called activated simplices. After that, each activity is represented by a manifold which is approximated by an arrangement of activated simplices. A sequence of action-snippets is classified by selecting the closest manifold and outputting the corresponding activity. Lillo et al. [22] introduced a hierarchical model for human action recognition using body joint locations. Their model categorizes complex actions by composing atomic actions. Specifically, for each atomic action the model generates temporal and spatial action annotations. The first ones estimate the action starting and ending times. The second ones infer the human body parts involved in executing the action.

Since 2015 several works based on deep learning have been proposed. Du et al.[23] presented an architecture based on several Tanh Bidirectional Recurrent Neural Networks (Tanh-BRNN) which converged in a LSTM and a final, fully-connected layer using a hierarchical decomposition of the skeletons as inputs. Vivek et al. [24] introduced a model for activity recognition based on a differential gating scheme for a LSTM, which uses as input a concatenation of the 3D coordinates obtained from the skeleton map after some normalizations. The first proposal which used a CNN+LSTM (LRCN) architecture for activity recognition in video sequences (not skeletons) was introduced by Donahue et al. [25]. Neverova et al. [26] used a CNN and a recurrent neural network to learn human identities from the motion patterns of smartphones provided by their accelerometers and gyroscopes. In a posterior work, Neverova et al. [27] presents a method for gesture detection and localisation based on multi-scale and multi-modal deep learning techniques. Lingfei et al. [28] presented an activity recognition system based on a CNN+MLP model which receives a two-dimensional structure (the movement of the skeleton joints with respect to time) as input. Wang et al. [29] proposed a model for activity recognition based on trajectory maps and CNNs. Yanghao et al. [30] described an architecture with three LSTM layers, a classification fully connected layer and a regression network to determine the start and end point of each action. Liu et al. [31] proposed a tree-structure to represent each skeleton and modified the LSTM architecture to visit joints in a sequence which follows the structure of the tree. Mahasseni and Todorovic [32] again used video sequences to train a CNN. Afterwards, the CNN output is complemented with the skeleton and it is used as the input for a LSTM model. Zhu et al. [33] proposed the use of a fully-connected deep LSTM which learns the co-occurrence features of skeleton joints. Hu et al.[34] focus on heterogeneous features learning for RGB-D activity recognition, by proposing a joint learning model to explore the shared and feature-specific components in RGB and depth simultaneously. Wang et al. [29] propose an encoding of 3D skeleton sequences into three 2D images, and use them to fine-tune existing ConvNets models for the classification of skeleton sequences without training the networks afresh. Song et al. [35] propose a model on top of a Recurrent Neural Network with Long Short-Term Memory, which learns to discriminate the relevant joints of skeleton paying different levels of attention to the outputs of different frames.

For a more extensive and exhaustive review of this research area, we suggest the survey paper by Han et al. [36] which focuses on the space-time representation of people based on 3D skeletal data.

2.2. Skeleton-based hand gesture recognition

To the best of our knowledge, there is significantly less work in the literature dealing with hand skeleton than those considering the human full-body skeleton.

In Ioenescu et al. work [37] a dynamic hand-arm gesture recognition technique is proposed, based on a 2D skeleton representation of the hand. For each gesture, the method generates a single image by superposition of the skeletons in each posture, which is considered as the dynamic signature of the gesture. The recognition phase consists of computing the Baddeley's distance of this signature with the ones from a gesture alphabet.

Sivarajesh et al. [38] proposed a hand gesture recognition system for HCI applications, based on a very similar idea to [37]. The proposed system computes the skeleton of the hand by using the distance transformation, and skeletons are superimposed on a single image called dynamic signature. Gesture is recognized by using the image Euclidean distance metric with respect to a gesture alphabet set.

Wang & Chan [39] used the depth map and the skeleton provided from a Kinect device. The hand shapes (depth) and corresponding textures (color) are represented as superpixels. Based on this representation, an Earth Mover's distance is proposed to measure the dissimilarity among different hand gestures.

De Smedt et al. [40] considered the hand skeleton provided by an Intel RealSense camera [41], that offers 22 hand joint's coordinates in 3D. To represent a hand gesture, the authors proposed to capture the hand shape time evolution based on skeleton joints, and the translation/rotation movements of the hand. The descriptors are encoded by a Fisher Vector representation obtained using a Gaussian Mixture Model. The temporal nature of gestures are encoded using a so-called "temporal pyramid". Finally, the classification process is performed by a linear Support Vector Machine classifier.

3. Our proposal

Convolutional Neural Networks (ConvNets or CNNs for brevity) are feed forward neural networks that are gaining success on many computer vision problems [42]. Usually, CNNs use three types of layers: convolutional, pooling and fully-connected (or dense) layers. Convolutional layers are used to discover local relations in their inputs. Pooling layers gradually reduce the dimension of the input. Usually, in a CNN one can find several levels of convolutional-pooling layers, and in each layer several convolutions are typically performed. Finally, the fully-connected layers are used to classify the input patterns into a finite number of classes.

A recurrent neural network (RNN) is a type of artificial neural network in which connections form cycles. These cycles allow the network to remember previous states, and that is the reason why they are commonly used to detect pattern sequences on time series. Long Short-Term Memory (LSTM) networks are a type of RNNs which are specifically designed to maintain long-term dependencies with a new structure called memory cell [43]. As in a regular RNN, at each time step, a pattern is presented to the network. This pattern and the last output of the LSTM are concatenated to create the next input to the network. The LSTM is structured in a way that the input is supplied to four blocks in the input layer. The first block produces values to the memory cell, and the other three blocks modulate the input (input gate), the internal memory (forget gate), and the output (output gate) of the cell. More information about the LSTM can be found in [25].

Usually, several layers are used in both of the aforementioned types of networks. The main idea is that higher-level features are obtained in final layers, by composing lower-level features that are obtained in previous layers [42]. The current success of CNNs and LSTMs would be impossible without the recent advances in the training of deep learning architectures [42,44–46].

The rest of this section is devoted to presenting our proposed network architecture, which is based on the concepts of CNN and LSTM explained above. We then present a detailed view of the system. Afterwards, the structure of the data is introduced and, finally, a data augmentation strategy to improve the performance of the method is suggested.

3.1. An overview of the proposed network architecture

The proposed architecture consists of a combination of a Convolutional Neural Network (CNN) followed by a Long Short-Term Memory (LSTM) recurrent network. CNNs are basically designed to explore high spatial local correlation patterns in images [47]. In our case, this network will focus on the detection of spatial patterns related to the position of the skeleton joints in 3D space. The LSTM recurrent network is then used to capture spatiotemporal patterns related to the time evolution of the 3D coordinates of the skeleton joints.

However, we first pre-train the CNN independently of the LSTM. This is accomplished by connecting the CNN to a fully-connected multilayer perceptron (MLP) of two hidden layers. Once we have a pre-trained CNN, we connect its output to the LSTM and we finish the training stage. Doing so, we get a more efficient and faster training stage than performing it in a single step, as we show in Section 4.2.

The input data is arranged in a three-dimensional block. Each dimension of this block coincides with the number of skeleton joints, J, in the first dimension, the number of consecutive frames, T, considered for determining the action/gesture in a second dimension, and the three spatial coordinates (x, y, z) of the joint in the third dimension. For example, for the full-body skeleton depicted in Fig. 2.a, the input data is composed of 20 joints, T time steps and three spacial coordinates. On the other hand, for the hand skeleton shown in Fig. 2.b, the input data is composed of 22 joints, T time steps and three spatial coordinates. At each time step, we shift the block one frame, releasing the oldest to include a new one in an overlapping mode. The input data structure is shown in Fig. 3, and it is more detailed in Section 3.3. So, in our model, the CNN takes into account the three-dimensional information of the data and some temporal dimension (T time steps) to generate the features detected in the input block. Later, to integrate the features detected in the consecutive overlapping blocks we use a LSTM. The internal cycle of the LSTM allows the system to maintain information beyond the last T time steps.

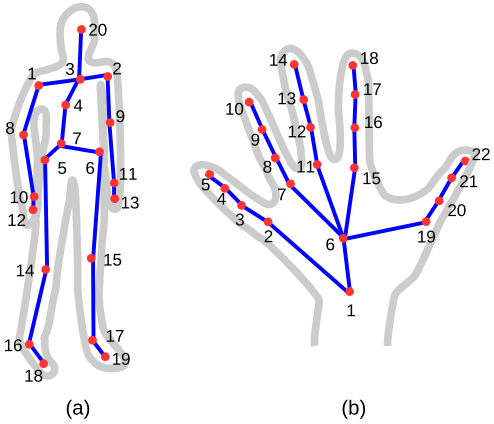

Fig. 2. (a) Examples of (a) a full-body skeleton (20 joints) and (b) a hand skeleton (22 points).

3.2. A detailed view of the network architecture

The structure of the CNN+MLP for training is shown in Fig. 4. The size of the input data depends on the particular problem. For that reason, some convolutional layers in the network have slightly different dimensions, depending on the problem at hand. Let T be the number of consecutive time steps considered in the action or gesture. The first convolutional layer filters the $T \times 20 \times 3$ skeleton input data (and $T \times 22 \times 3$ in the hand model), with 20 different kernels of size $3 \times 3 \times 3$. It outputs 20 blocks of $T-2 \times 18 \times 1$ data ($T-2 \times 20 \times 1$ in the hand model). Its corresponding pooling layer is a non-overlapping maximum operation (max-pooling reduction layer) in neighborhoods of $2 \times 2 \times 1$, halving the original dimensions of number of time steps and skeleton joints resulting in $(T-2)/2 \times 9 \times 20$ data (and $(T-2)/2 \times 10 \times 20$ in the hand skeleton). The second convolutional layer uses 50 kernels of size $2 \times 2 \times 20$ for the skeleton data ($2 \times 3 \times 20$ in the case of the hand) connected to a $2 \times 2 \times 1$ max-pooling layer, reducing the data stream to $(T-4)/4 \times 4 \times 50$ elements in both cases. The last convolutional layer filters the previous output with 100 kernels of size $3 \times 3 \times 50$, and then, a $2 \times 2 \times 1$ max-pooling reduction is performed again. This output of size $(T-12)/8 \times 1 \times 100$ is connected to the MLP of two fully-connected hidden layers of 300 and 100 neurons respectively and a softmax layer as the final output with a size equal to the number of activities to be recognized.

As in other regular deep neural network, we do not need any particular order of joints in the input data structure. The network will end up aggregating spatially distant information as a consequence of the natural behavior of the CNN. The final network is shown in Fig. 5. Once the CNN has been pre-trained, the MLP layers are disconnected and replaced by a LSTM of 100 hidden units. The number of weights learnt by the whole network (CNN+LSTM) was 132,130 (the CNN has 49,710 weights, while the LSTM has 82,420 ones).

3.3. Data arrangement

This section presents some notation needed for the data augmentation section. A model μ with J joints Γ_j is defined as:

$$\Delta = \{\Gamma_j, 1 \leq j \leq J\} \tag{1}$$

where each joint Γ_j results from a composition of 3D coordinates $(x, y, z)_j$:

$$\Gamma_j \equiv (x, y, z)_j \in \Re^3 \quad \forall j \in [1, J] \tag{2}$$

Block data are called Θ (see Fig. 3). The dimensions of Θ are defined by (i) the number of joints of the model J, (ii) the number

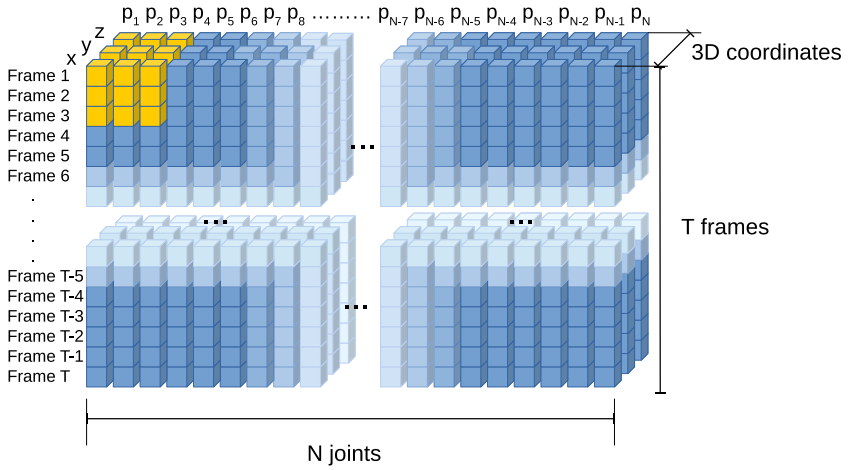

Fig. 3. The input data structure for the network. Note that the model in one instant is represented by a horizontal slice of $N \times 3$ input data. The first application of the $3 \times 3 \times 3$ convolution kernel will operate on the 3x3x3 first highlighted inputs (corresponding to 3 different 3D points (x,y,z) in a small spatiotemporal neighborhood of 3 time steps).

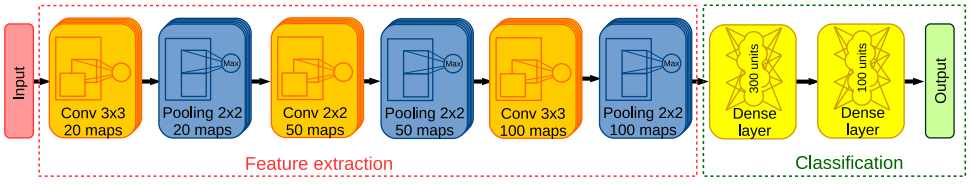

Fig. 4. The structure of the network during the pretraining stage consists of a CNN attached to a MLP.

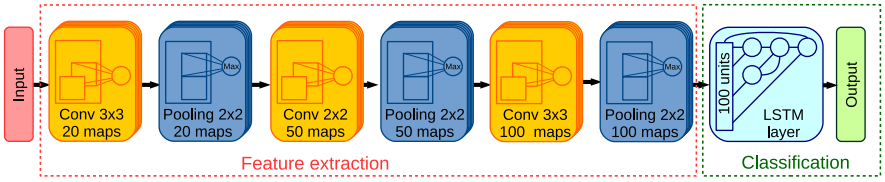

Fig. 5. The structure of the network during the final training stage consists of a CNN attached to a LSTM.

of considered time steps, T, and (iii) the 3D coordinates of each joint, respectively. Let Θ^{t+T} be a block of consecutive models from time step t to $t+T$. Then:

$$\Theta_t^{t+T} = \{\Delta_i, t \le i < t+T\} \tag{3}$$

Let Σ be a sequence composed of a set of blocks as follows:

$$\Sigma = \{\Theta_t^{t+T}, 1 \le t \le T_{total} - T\} \tag{4}$$

Therefore, a pre-training set of action labeled with label l, P_l, is composed of a number of K sequences Σ_k:

$$P_l = \{\Sigma_k \quad 1 \le k \le K, Label(\Sigma_k) = l\} \tag{5}$$

where $Label(\Sigma_k)$ is the action label assigned to the sequence Σ_k.

Finally, the whole set of pretraining data, Π, for L actions is described as:

$$\Pi = \{P_l, 1 \le l \le L\} \tag{6}$$

3.4. Data augmentation

Overfitting is a serious problem in deep neural networks [46]. It leads to an adequate performance on the training datasets, but a poor performance on the test ones, specially when the training dataset is small [48]. Data augmentation techniques are proposed in order to prevent overfitting when training datasets are small and/or they are not diverse enough (for example, similar sequences in scale, pose, light, etc.). Some skeleton datasets used in this work are quite small. For example, we only have 320 sequences available in MSRDailyActivity3D or 200 in the UTKinect-Action3D. Even the hand gesture dataset with 2800 sequences is insufficient for CNN and LSTM training, as millions of parameters have to be learnt. Therefore, we have performed different forms of data augmentation to enlarge these collections:

- *Scale*: A random scale factor f is applied to the 3D position of each joint Γ of the model Δ along the sequences. We have used a scale factor of ± 0.3 (that is, a uniformly distributed random number $f \in [\times 0.7, \times 1.3]$) for the human activity recognition problem and ± 0.2 for the hand gesture recognition problem, respectively. We chose these values in order to replicate the variability observed in the datasets. Then, applying the operator *Scale* with a scale factor f to a model Δ produces a new data set, called $Scale(\Delta, f)$, that can be described as:

$$Scale(\Delta, f) = \{f \cdot \Gamma_j, \ \forall j \in [1, J], f \in [f_{min}, f_{max}]\} \tag{7}$$

- *Shift*: A global displacement $d = (d_x, d_y, d_z)$ is applied to the whole model. For the human activity recognition problem, $d_x, d_y \in [-0.5, 0.5]$ meters and $d_z = 0$ (i.e., only displacements on the horizontal plane are allowed). For the hand gesture recognition problem, $d_x, d_y, d_z \in [-0.1, 0.1]$. The result of applying the *Shift* operator to a model Δ with a global displacement of d produces a new data set, called $Shift(\Delta, d)$, described as:

$$Shift(\Delta, d) = \{d + \Gamma_j, \ \forall j \in [1, J]\} \tag{8}$$

- *TimeInterpolation*: This operator generates new models by interpolating the positions of each joint between consecutive time

steps. In order to do so, a displacement vector $\overrightarrow{M}_{j,t}$ is computed for each joint j between two consecutive frames t and $t+1$. Then, a new joint position is obtained from the joint j at time t $\Gamma_{j,t}$ by applying a displacement of $r \cdot \overrightarrow{M}$, where r is a uniformly distributed random value in the range [0, 1]. When applied to the whole model, it can be described as follows:

$$TimeInterpolation(\Delta, r) = \{r \cdot \overrightarrow{M}_{j,t},\ \overrightarrow{M}_{j,t} = \Gamma_{j,t+1} - \Gamma_{j,t},\ \forall \Gamma_{j,t} \in \Delta_t\ \wedge\ \Gamma_{j,t+1} \in \Delta_{t+1}, r \in [0,1]\} \quad (9)$$

- *Noise*: We randomly select up to four joints from the model and then we apply a random noise to their positions. The amplitude of the perturbation is different for each coordinate of each selected joint, but it is the same at every time step along the whole sequence. This strategy is intended to provide robustness to the system. Then, for a set of models in a sequence $\{\Delta_t\}$ and a set of selected joints to be perturbed S, this operator can be described as follows:

$$Noise(\{\Delta_t\}, S) = \{\Gamma_{j,t} + n_j,\ \forall j \in S,\ \forall t \in [0,T]\} \quad (10)$$

where $\Gamma_{j,t} = (x,y,z)_{j,t}$ are the coordinates of the joint j at time t and $n_j = (n_x, n_y, n_z)_j$ is the constant noise applied to the selected joints. Depending on the problem the range of the perturbation is $[-0.3, 0.3]$ meters for the skeletons and $[-0.1, 0.1]$ for the hand joints. Again, we choose these value ranges to approximate the variability observed in the datasets.

- *Subsample*: The dataset is extended by obtaining different subsequences from a sequence. These subsequences should be long enough because the action has to be recognizable. In this way, given a sequence of models $\{\Delta_t\}$ with $t \in [0, T]$ we subsample with a period of m frames, starting from the frame d. Notice that, with this strategy in mind, we can obtain m different subsequences from each original one. The resulting set of models from each of the subsequences can be expressed as:

$$Subsample(\{\Delta_t\}, d, m) = \{\Delta_t,\ (t-d)\ mod\ m = 0\} \quad (11)$$

where $(t-d)\ mod\ m$ is the remainder of dividing $t-d$ by m.

4. Experimental results

This section describes the computational experiments that we performed to test the effectiveness of our proposal as well as to compare it to methods identified in the state of the art of human activity and hand gesture recognition. We have implemented our network design using the Theano framework.[1] The training stage was performed on GPU using an Intel Xeon E5-1620v3 server clocked at 3.5GHz with 16GB RAM and a 2015 NVIDIA GeForce GTX TITAN Black GPU with 6GB of GDDR5 memory and 2880 CUDA cores (15 streaming multiprocessors of the NVIDIA Kepler family). The test stage was conducted on different CPU and GPU platforms and results are described in Section 4.9.

We have tested our proposal on six different datasets: five action/activity recognition benchmarks and one hand gesture recognition benchmark. The former are called MSR Action3D dataset [49], MSRDailyActivity3D dataset [50], UTKinect-Action3D dataset [7], NTU RGB+D [4] and Montalbano V2 [3]. The latter is the Dynamic Hand Gesture dataset [40]:

- The MSR Action3D dataset [49] includes 567 sequences in total, containing 20 actions, and each action is performed by ten subjects two or three times. However, 10 sequences are considered corrupted in the majority of the works and, therefore, we ignore them, performing our experimental study over the remaining 557 sequences. The actions of this dataset are typical in the context of interaction with video game consoles. For the experiments, authors describe three action subsets called AS1, AS2 and AS3. The subsets AS1 and AS2 are intended to group actions with similar movement, while AS3 is intended to group complex actions together. Training and testing split is well established in the literature [49]. Subjects 1,3,5,7,9 are used for training and 2,4,6,8,10 are used for testing (Protocol A). Some papers also report cross-validation results with all possible 5-5 splits, totalling 252 (Protocol B) [51].
- The MSRDailyActivity3D dataset [50] contains 16 daily activities executed by 10 different subjects. The considered activities are: *drink, eat, read book, call cellphone, write on a piece of paper, use laptop, use vacuum cleaner, cheer up, sit still, toss paper, play game, lay down on sofa, walk, play guitar, stand up, sit down*. Each activity is performed twice, one in the standing positions, and another in the sitting position. The dataset was captured by a Kinect sensor, and depth maps, RGB video and skeletons are provided.
- The UTKinect-Action3D Dataset [7] collects up to 10 types of human actions in indoor settings, including: *walk, sit down, stand up, pick up, carry, throw, push, pull, wave and clap hands*. Each action is carried out twice by 10 different people. Also, the sequences were captured using a Microsoft Kinect device, and RGB images, depth maps and skeleton joint locations are provided by the authors. It is composed of 199 sequences in total.
- The NTU RGB+D is a large scale dataset for human action recognition [4]. It consists of 56,880 action samples containing 4 different modalities of data for each sample: RGB videos, depth map sequences, 3D skeletal data and infrared videos. The dataset provides 60 action classes in total, which are divided into three major groups: 40 daily actions (such as drinking, eating, reading, etc.), 9 health-related actions (like sneezing, staggering, falling down, etc.), and 11 interactions (such as punching, kicking, hugging, etc.).
- The Montalbano V2 dataset was presented by Escalera et al. [3] in the context of the ChaLearn Looking at People Challenge 2014. This dataset provides more than 14,000 samples from a vocabulary of 20 Italian sign gesture categories in continuous data series, that we do not reproduce here for the sake of brevity.
- The Dynamic Hand Gesture dataset DHG-14/28 contains 14 gestures. Each one is executed 5 times by 20 participants in two ways, resulting in 2800 sequences. The gestures are subdivided into categories of fine and coarse [40]: *Grab* (G, fine), *Tap* (T, coarse), *Expand* (E, fine), *Pinch* (P, fine), *Rotation CW* (R-CW, fine), *Rotation CCW* (R-CCW, fine), *Swipe Right* (S-R, coarse), *Swipe Left* (S-L, coarse), *Swipe Up* (S-U, coarse), *Swipe Down* (S-D, coarse), *Swipe X* (S-X, coarse), *Swipe V* (S-V, coarse), *Swipe +* (S−+, coarse), *Shake* (Sh, coarse).

Notice that the first three datasets as well as the last one contain very few samples, while the fourth and fifth datasets are very large containing thousands of examples for training. This is important in order to analyze the performance of the methods.

The rest of this section is devoted to presenting and describing the training phase, the preliminary experiments for the network parameter setting and, finally, the experiments performed over each dataset.

4.1. Description of the training phase

Training is divided in two stages. In the first one, the CNN+MLP is trained to learn the most relevant features and relationships among the joints of the model for activity and gesture recognition. We use a step decay strategy to anneal the learning rate over time. The learning rate is initially set to 0.1, and then, it is grad-

[1] Theano official site at: https://github.com/Theano/

Table 1
Accuracy obtained for different values of T following protocol A.

T	AS1	AS2	AS3	Average
10	92.39	94.65	93.7	93.58
20	93.3	94.6	99.1	95.66
30	93.81	85.72	93.7	87.74

ually reduced by a 0.993 decay factor every 5 epochs without improvement. A Rectified Linear Unit (ReLU) is taken as the activation function. The batch size is set to 200, and the network is trained for up to 100 epochs. The CNN training time does not exceed three hours on GPU.

The second stage is devoted to training the whole system, i.e., the CNN+LSTM network. The weights computed in the previous stage are used to initialize the CNN. We use the Adadelta algorithm [52] that enables us to optimize the learning rate adaptively. The maximum number of epochs and batch size are set to 500 and 16, respectively. The training times for this second stage are around two hours on GPU.

The data augmentation operators, described in Section 3.4, are applied statically and dynamically during both training stages in such a way that the training dataset is augmented at each iteration.

We have also tried to train the whole model in only one stage, but the obtained results were not as good as those gathered when training in two stages. We also made several trials increasing the number of epochs, but we always got overfitting.

4.2. Preliminary experiments

We performed the preliminary experimentation on the MSR Action3D Dataset [49]. Subjects 1, 3, 5, 7, 9 are used for training and 2, 4, 6, 8, 10 are used for testing, which is called protocol A.

Our first experiment is devoted to set the parameter T, i.e., the number of time steps considered in the input data block (see Section 3.3). We consider $T = 10$, 20 and 30 time steps, that correspond to an approximate time period of 0.5, 1.0 and 1.5 seconds of data-acquisition, respectively. Table 1 depicts the results obtained in this experiment for the three values of T tested (one per row) on each subset (one per column).

As can be observed, the best accuracy results are obtained for $T = 20$ on average, ranking in the first or second place in the three subsets AS1, AS2 and AS3. For that reason, we set T to this value for the rest of the experimental design.

Our second preliminary experiment is devoted to showing the effect of the first training stage of the CNN described in the previous section. To this aim, we have performed the previous experiment with and without dividing the training phase into these two stages. Results are shown in Table 2, where each experiment is reported in a row.

Results reported in Table 2 show that the training in the two separate stages has a positive effect on the accuracy results as compared to the single-stage training. As a consequence, we adopt this two-stage training strategy for the rest of the experimental design.

To evaluate the impact of the data augmentation strategy quantitatively, we have performed a third preliminary experiment testing the accuracy of the network with and without the data augmentation stage on the MSR Action3D dataset. Table 3 shows the effect of each data augmentation transformation on the final accuracy metric using protocol A. In the table, it can be observed that noise, scale, and shift transformations improve the accuracy results on their own. Meanwhile, subsampling and time interpolation seem to penalize the results. We think that this penalty is related to the reduced size of the training set and the quality reduction introduced by these two data augmentation processes. However, when subsampling and time interpolation are applied in conjunction with noise, scale, and shift transformations, the training sample is much bigger and the accuracy improves again.

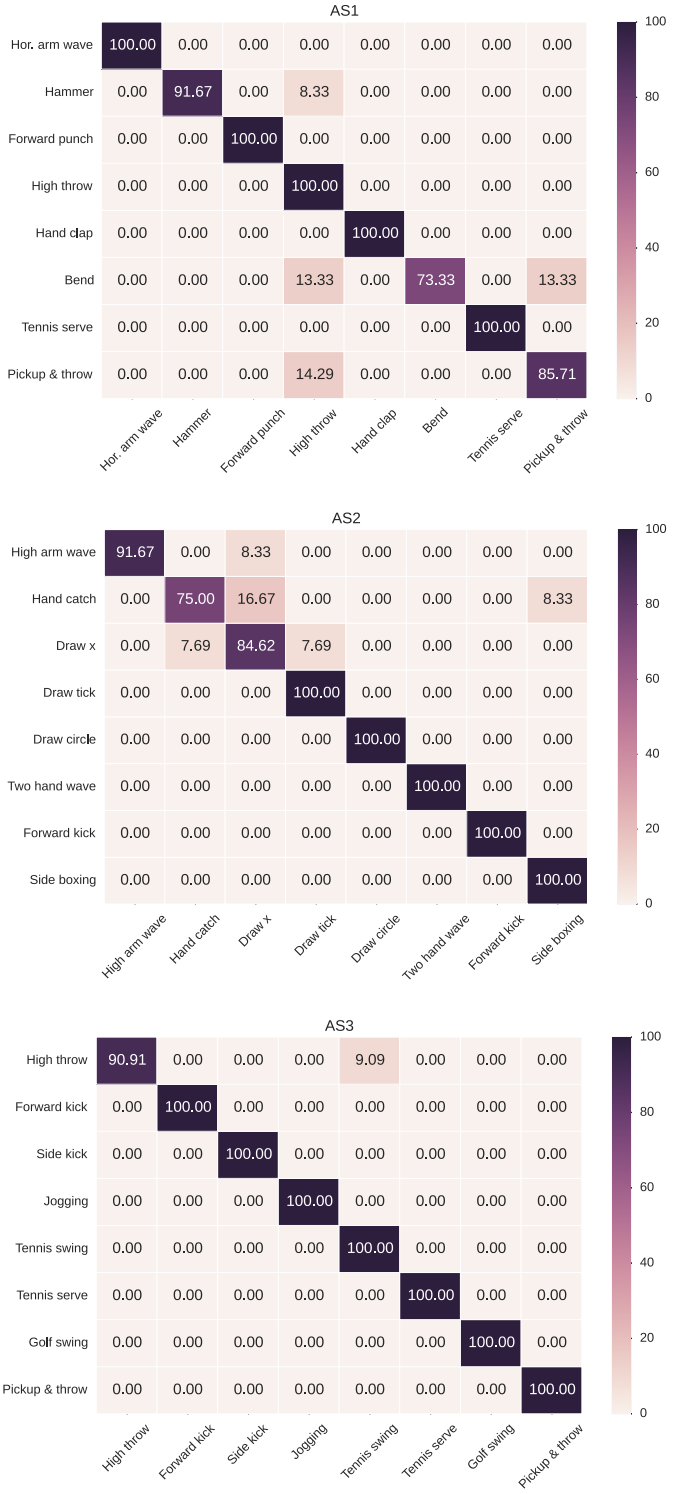

Fig. 6. Confusion matrix on MSR Action3D dataset training with subjects 1, 3, 5, 7, and 9 and testing with subjects 2, 4, 6, 8, and 10.

Table 2
Accuracy obtained when the network is trained in two steps as described in Section 4.1 Vs training the whole CNN+LSTM network in only one stage.

T	AS1	AS2	AS3	Average
Two stages training (CNN+MLP/CNN+LSTM)	93.3	94.6	99.1	95.7
One stage training (CNN+LSTM)	88.58	91.97	96.4	92.32

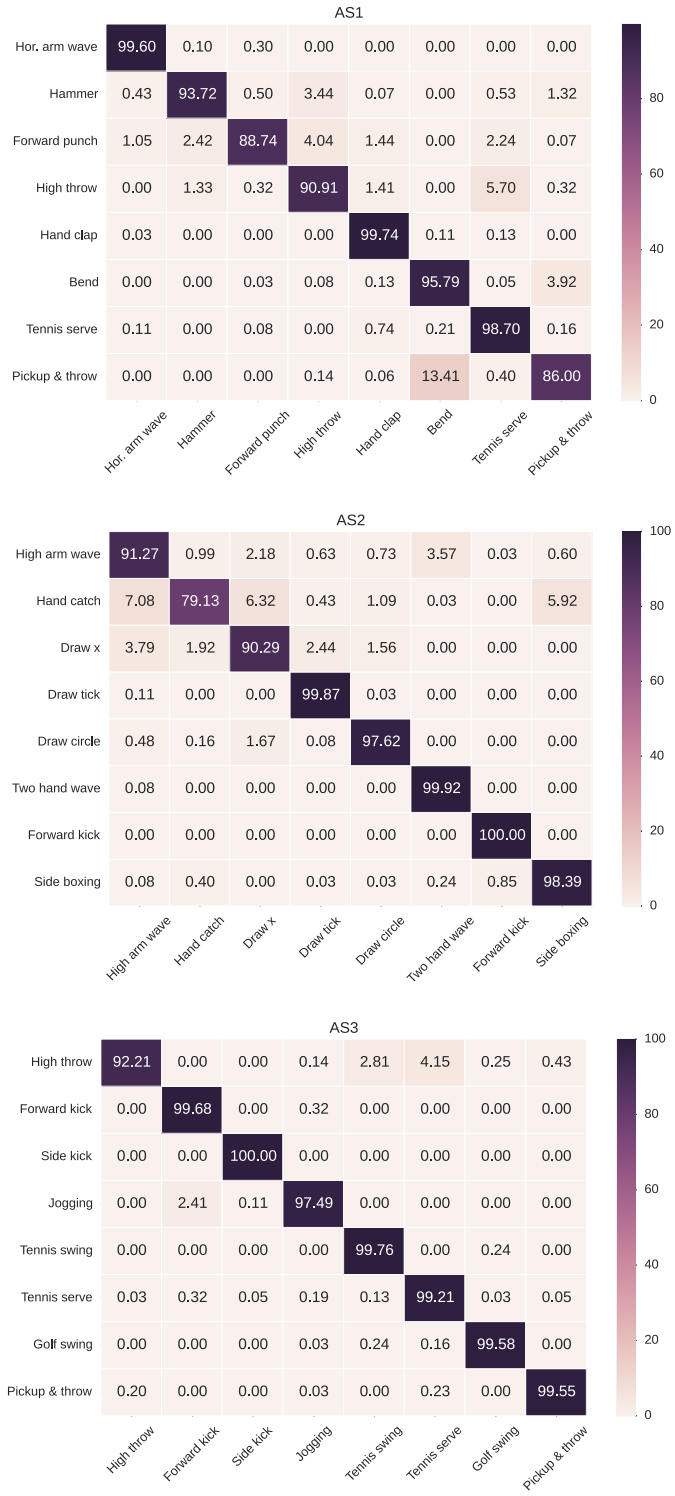

Fig. 7. Confusion matrix on MSR Action3D dataset training following the protocol of "average over all splits".

Table 3
Impact of each data augmentation transformation over the accuracy of the proposed method on the MSR Action3D Dataset.

Transformation	Accuracy			
	AS1	AS2	AS3	Average
No data augmentation	82.86	85.72	93.7	87.4
Noise	86.67	91.97	98.2	92.3
Scale	90.48	89.29	91	90.3
Shift	88.58	91.97	96.4	92.3
Subsample	79.05	85.72	90.1	85.0
Time interpolation	81.91	86.61	87.39	85.3
Noise + scale + shift	92.39	94.65	98.2	95.1
All	93.3	94.6	99.1	95.7

Table 4
Accuracy results for MSR Action3D Dataset following the Protocol A: training with subjects 1, 3, 5, 7, and 9 and testing with subjects 2, 4, 6, 8, and 10.

Method	Accuracy			
	AS1	AS2	AS3	Average
Wang et al., 2016 [20]	–	–	–	97.4
CNN+LSTM	**93.3**	**94.6**	**99.1**	**95.7**
Wang et al., 2016 [21]	–	–	–	95.0
Chen et al., 2015 [53]	98.1	92.0	94.6	94.9
Du et al., 2015 [23]	93.3	94.6	95.5	94.5
Tao & Vidal, 2015 [14]	89.8	93.6	97.0	93.5
Lillo et al., 2016 [22]	–	–	–	93.0
Vemulapalli et al., 2014 [11]	95.3	83.9	98.2	92.5
Gowayyed et al., 2013 [54]	92.4	90.2	91.4	91.3
Ben Amor et al., 2016 [17]	–	–	–	89.0
Cippitelli et al., 2016 [19]	79.5	71.9	92.3	81.2
Li et al., 2010 [49]	72.9	71.9	79.2	74.7

4.3. MSR Action3D dataset

Figs. 6 and 7 present the confusion matrices for each subset using the proposed architecture, following the protocols A (training with subjects 1, 3, 5, 7, and 9 and testing with subjects 2, 4, 6, 8, and 10) and B (average over all splits), respectively. Misclassification hardly ever occurs, and when it appears, paired actions are quite similar. For example, in the set AS1, the action *Pickup & throw* is often misclassified as *High throw* and *Bend*, while the action *Bend* is misclassified as *Pickup & throw* and *High throw*. Actually, *Pickup & throw* has just one more *throw* move than *Bend*, and the *throw* move often holds few frames in the sequence, making these two actions hard to distinguish. In the set AS2, the action *Hand catch* is misclassified as *Draw x*, *Side boxing* and *High arm wave*, while the action *Draw x* is misclassified as *Hand catch* and *Draw tick*.

Table 4 compares the accuracy results obtained by our proposal with respect to previous methods on the subsets AS1, AS2 and AS3, following the protocol A.

As can be seen from the results reported in Table 4, our proposed architecture ranks in second place (out of 12 methods) and not too far from the method ranked in the first place.

Table 5 shows the accuracy results obtained by our proposal in comparison to previous relevant methods in the literature, following the Protocol B.

Results depicted in Table 5 rank our proposal in second place (out of 5 methods) in terms of averaged accuracy, not too far from

Table 5
Accuracy results for MSR Action3D Dataset following the protocol B: average over all splits.

Method	Accuracy			
	AS1	AS2	AS3	Average
Faria et al., 2015 [51]	96.6	96.7	98.6	97.3
CNN+LSTM	**94.4**	**95.1**	**98.6**	**96.0**
Wang et al., 2016 [20]	–	–	–	94.4
Wang et al., 2016 [21]	–	–	–	91.4
Chen et al., 2015 [53]	–	–	–	87.9

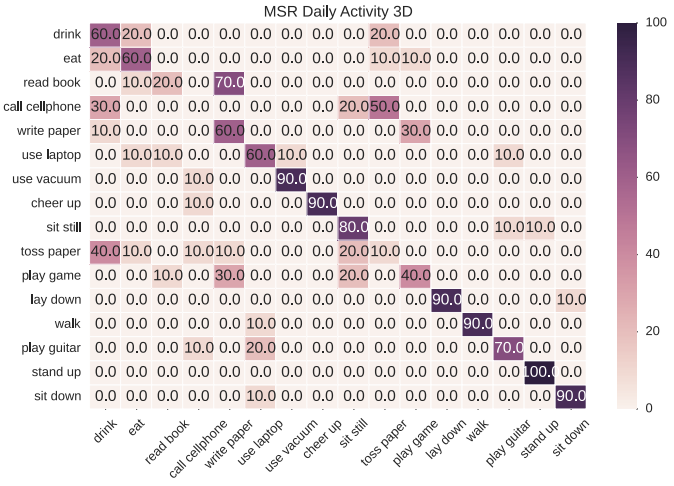

Fig. 8. Confusion matrix on MSRDailyActivity3D Dataset.

Table 6
Accuracy results for MSRDailyActivity3D Dataset.

Method	Accuracy
Faria et al., 2015 [51]	96.8
Behnam, 2015 [55]	87.5
Tao & Vidal, 2015 [14]	74.5
Zanfir et al., 2013 [8]	73.8
Wang et al., 2012 [50]	68.0
CNN+LSTM	**63.1**
Ben Amor et al., 2016 [17]	63.0

the best method. Finally, it is important to emphasize that using only the 3D skeleton information, the proposed architecture outperforms other state-of-the-art approaches that also use depth information.

Considering separate activities, the accuracy obtained for the *Forward punch* action is the best one when compared to the results obtained by the other works. Moreover, in 17 other actions, the accuracy results obtained by our method are as good as the best of the considered works.

4.4. MSRDailyActivity3D dataset

This dataset was designed to cover daily activities in the living room. When activities are performed in sitting positions, the 3D joints are very noisy. Furthermore, most of the activities require human-object interactions which make them more challenging. In this dataset, our proposal does not obtain good results as compared to the identified methods in the literature, as it can be appreciated from the results in Fig. 8 as well as Table 6. Specifically, it ranked in sixth place out of seven methods.

The works of Faria et al. [51] and Tao & Vidal [14] are the only ones that report results for each single action individually. When we compare our results for single actions, we concluded that we

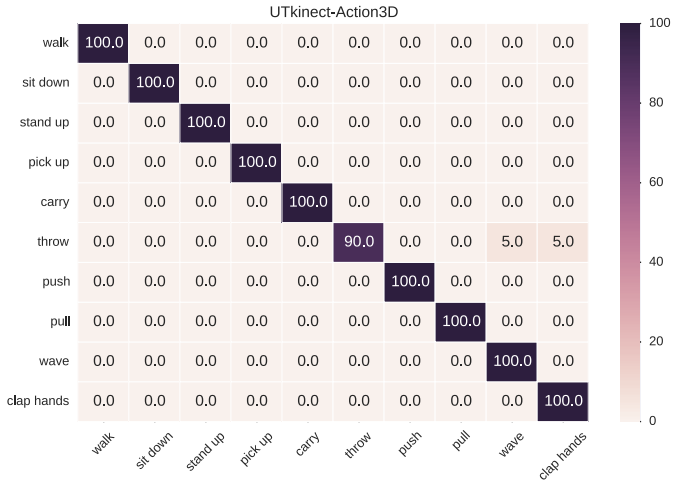

Fig. 9. Confusion matrix on UTKinect-Action3D dataset.

Table 7
Accuracy results for UTKinect-Action3D Dataset .

Method	Accuracy
CNN+LSTM	**99.0**
Vemulapalli et al., 2014 [11]	97.1
Liu et al., 2016 [31]	97.0
Wang et al., 2016 [21]	96.5
Cippitelli et al., 2016 [19]	95.1
Anirudh et al., 2105 [56]	94.9
Wang et al., 2016 [20]	93.5
Li & Fu, 2013 [57]	92.0
Liu et al., 2015 [58]	92.0
Zhu et al., 2013 [59]	91.9
Chrungoo et al. 2014 [10]	92.0
Jiang et al., 2015 [60]	91.9
Ding et al. [16]	91.5
Devanne et al., 2013 [9]	91.5
Xia et al., 2012 [7]	90.9
Theodorakopoulos et al., 2014 [61]	90.9

are not as far away as Table 6 suggest. In fact, we obtain the same accuracy results for the action *Stand up*, and for a significant number of actions, we are only 10% from them. However, the results for particular actions such as *Read book* (with an accuracy of 20%) and *Toss paper* (with an accuracy of 20%) are considerably worse than their results (beyond the 70%). That is the reason why our accuracy results are considerably lower on average.

4.5. UTKinect-Action3D dataset

This dataset features four challenges that make it particularly complex. First, action sequences were taken from different views. Second, there is significant variation among realizations of the same action (i.e: some subjects prefer to pick up the objects with one hand while others use both hands). Third, the duration of the action clips varies dramatically, and fourth, sequences contain object-person occlusions and body parts out of the field of view. All of these particular features make this dataset a very demanding one.

Because the UTKinect-Action3D dataset is not very large, it is common to use a “leave-one-sequence-out” cross validation strategy to estimate how accurately a predictive model will perform [7]. We also follow this strategy to compare the obtained results to the state-of-the-art approaches. Table 7 collects the accuracy obtained by the proposed CNN+LSTM architecture over the UTKinect-Action3D dataset as well as other works. As can be seen, the ac-

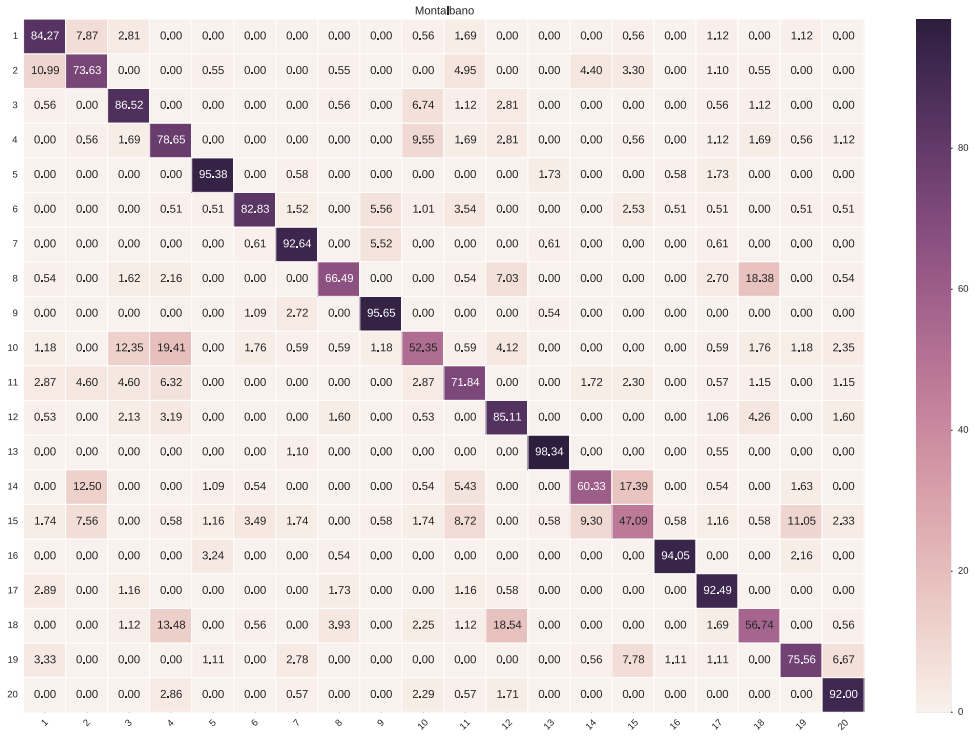

Fig. 10. Confusion matrix on Montalbano V2 dataset.

Table 8
Accuracy results for NTU RGB+D Dataset.

Method	Accuracy	
	Cross-Subject	Cross-View
Wang et al., 2016 [29]	76.3	81.1
Song et al., 2017 [35]	73.4	81.2
Liu et al., 2016 [31]	69.2	77.7
CNN+LSTM	**67.5**	**76.21**
Shahroudy et al., 2016 [4]	62.93	70.27
Hu et al., 2016 [34]	60.23	65.22
Du et al., 2015 [23]	59.07	63.97
Vemulapalli et al., 2014 [11]	50.08	52.76
Evangelidis et al. 2014 [12]	38.62	41.36

curacy results obtained by our proposal outperforms the previous works found in the literature.

Furthermore, Fig. 9 shows the confusion matrix for the 10 different actions. The proposed architecture rarely ever confuses actions, reporting impressive precision and accuracy measures. Finally, when we compared our accuracy results for each action individually, we found that our results are as good as the best one of the considered works.

4.6. NTU RGB+D dataset

Table 8 shows the results obtained on the NTU RGB+D dataset. This table reports two accuracy measures: cross-subject evaluation and cross-view evaluation. In the cross-subject evaluation, the 40 subjects are divided into training (40,320 samples) and testing (16,560 samples) groups. In the cross-view evaluation, all the samples of camera 1 (left and right 45 ° views of the actions) are selected for testing and samples of cameras 2 and 3 (front and two side views of the actions) for training. The training and testing sets have 37,920 and 18,960 samples, respectively.

Results in Table 8 show that the proposed method ranks in the fourth place for both considered quality metrics and, for the second one, not too far from the best methods. Notice that this dataset contains a huge number of examples that can be used for the training stage and our proposal was primarily designed for smaller training datasets. In the case of large datasets, the data augmentation methods could not be a relevant strategy as well as the two-stage training phase. For that reason, we think that this experiment is not the most favorable to demonstrate the benefits of our method.

Table 9
Accuracy results for MONTALBANO V2 Dataset.

Method	Jaccard index
Neverova et al. [27]	83.1
CNN+LSTM	**79.15***
Chang et al. 2014 [62]	78.16
Escalera et al. 2015 (Ismar) [3]	74.66
Escalera et al. 2015 (Quads) [3]	74.54
Escalera et al. 2015 (Terrier) [3]	53.90
Escalera et al. 2015 (YNL) [3]	27.06

4.7. Montalbano V2 dataset

Table 9 shows the results obtained by our proposal and the identified methods in the state of the art on the Montalbano V2 Dataset [3]. To evaluate the accuracy of action/interaction recognition, the Jaccard index is used [3].

As stated before, Montalbano V2 dataset contains continuous data series in which several activities can appear. One of the required stages of the challenge is the action segmentation, and its performance affects the computation of the Jaccard index. As our system is not prepared to deal with the problem of action segmentation, we use the available segmentation data of the sequences in order to compute the Jaccard index value (thus marked with an asterisk in Table 9). Taking this into account, our proposal ranks in second place.

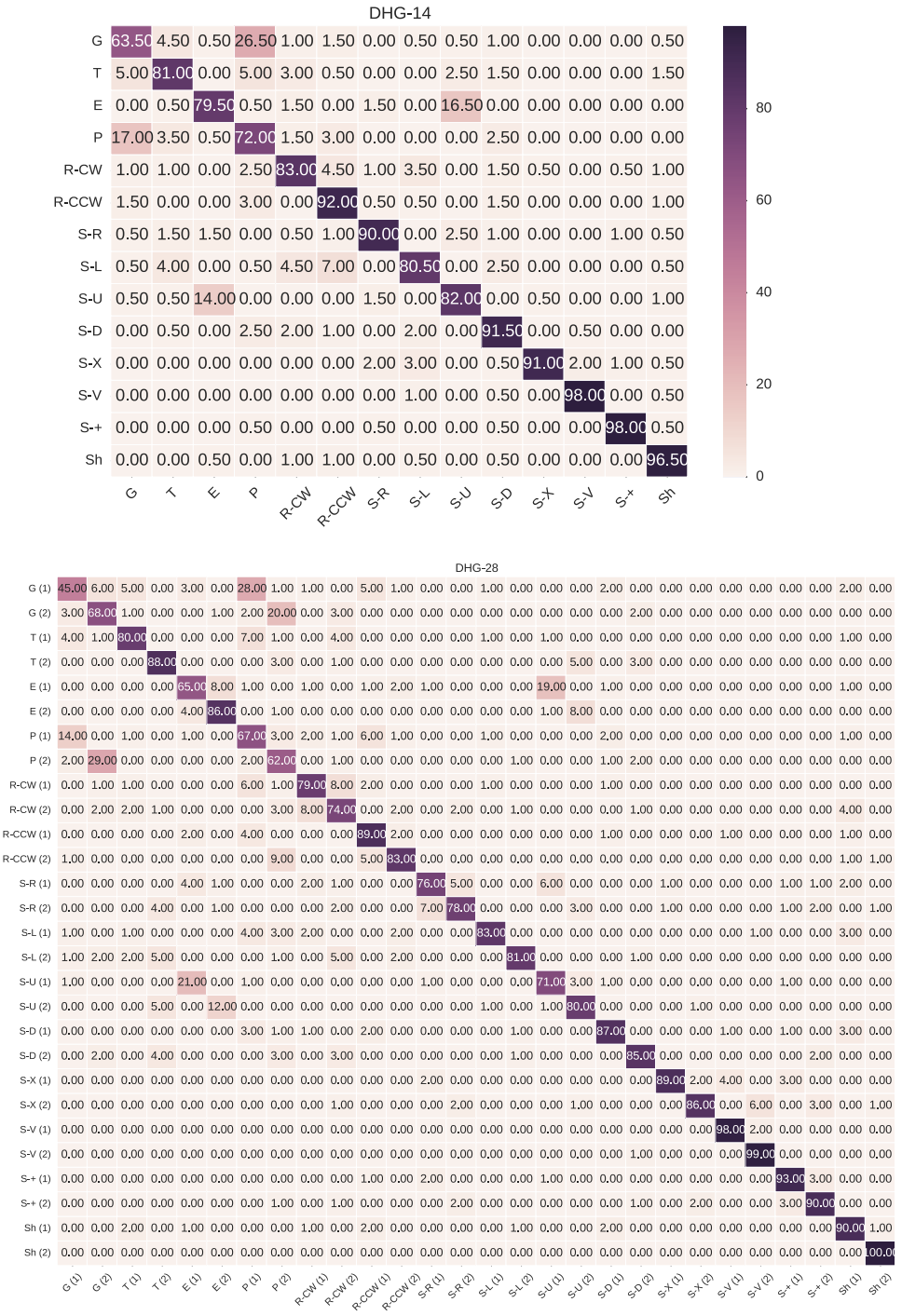

Fig. 11. Confusion matrices on DHG 14–28 dataset considering 14 (up) and 28 gestures (down), respectively.

Fig. 10 presents the confusion matrix for the 20 different gestures. On one hand, our method gets an accuracy value superior to 84% for half of the actions. On the other hand, misclassification for similar gestures is produced. For example, gesture 8 (*Sei pazzo*) is classified as 18 (*Buonissimo*), gesture 10 (*Nonme me friega niente*) as 3 (*Perfetto*) or gesture 14 (*Le vuoi prendere*) as 2 (*Viene qui*).

4.8. Dynamic hand gesture (DHG-14/28) dataset

In this experiment we followed a "leave-one-subject-out" design, i. e., we perform 20 experiments, each one using 19 subject for training and 1 subject for testing. The reported results are computed as the average over these 20 experiments.

Fig. 11 represents the confusion matrices when considering 14 (up) and 28 gestures (down), respectively. In both of them, we can observe that gestures *Expand* and *Pinch* are usually misclassified. The method obtains the lowest accuracy results for them (under 70%). However, the rest of gestures are classified with high accuracy.

We have also compared the obtained results to the work of De Smedt et al. [40]. Tables 10 and 11 report the results obtained by the two methods considered presented by category: fine, coarse and both gestures. We have found that our proposal obtains the

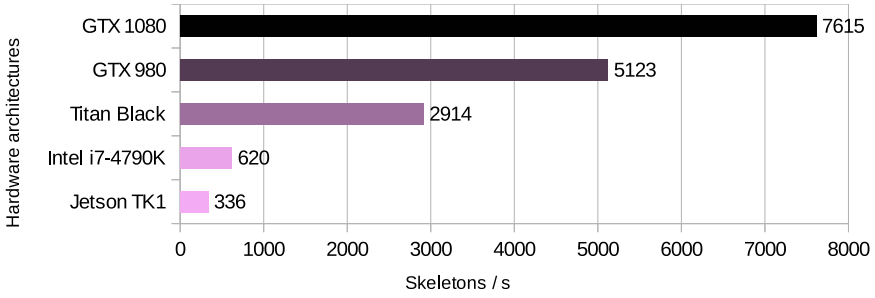

Fig. 12. Inference performance.

Table 10
Accuracy for DHG Dataset with 14 gestures.

Method	Accuracy		
	Fine	Coarse	Both
CNN+LSTM	**78.0**	**89.8**	**85.6**
De Smedt et al., 2016 [40]	73.6	88.3	83.1

Table 11
Accuracy for DHG Dataset with 28 gestures.

Method	Accuracy		
	Fine	Coarse	Both
CNN+LSTM	**71.8**	**86.3**	**81.1**
De Smedt et al., 2016 [40]	68.2	86.3	80.0

best results in all categories. Finally, the accuracy comparison for each individual gesture is favorable to our proposal in 9 of the 13 gestures. For the remaining gestures, our results are, at most, 5% lower than the best result.

4.9. Performance analysis

Different hardware architectures have been considered for the performance comparison at the inference stage. In particular, a NVIDIA Jetson TK1 board, an Intel Core i7-4790K CPU, and the use of three accelerators such as a 2014 NVIDIA GeForce TITAN Black GPU, a 2014 NVIDIA GeForce GTX 980 and a 2016 NVIDIA GeForce GTX 1080. The Jetson TK1 board is a 5 W embedded system that has a quad core ARM Cortex A15 CPU with a CUDA capable GPU with a single NVIDIA Kepler streaming multiprocessor (SMX) of 192 CUDA cores released in 2015. The i7-4790K processor is a high performance CPU from the Intel Core consumer line launched under the Haswell family in 2014 that offers 4 processing cores at a maximum clock frequency of 4 GHz. The Haswell microarchitecture supports Advanced Vector Extensions 2 (AVX2) as SIMD capability with 256-bit register width. This platform was also used as host to the hybrid CPU+GPU platforms for the evaluation of the 3 NVIDIA GPUs. The GeForce TITAN Black is a NVIDIA Kepler GPU from the 700 series with 15 Streaming Multiprocessors (SMX) of 192 CUDA cores, exhibiting a total of 2880 CUDA cores, with 6GB of GDDR5 memory. The GeForce GTX 980 is a more modern NVIDIA Maxwell GPU with 2048 CUDA cores in 16 streaming multiprocessors (SM) of 128 CUDA cores each, with 4GB of GDDR5 memory. The GeForce GTX 1080 is a NVIDIA Pascal-family GPU with 2560 CUDA cores in 20 Pascal SMs of 128 CUDA cores each with 8GB GDDR5X memory, which is considerably faster than previous GDDR5 memory. The different configurations between the number of streaming multiprocessors (SM) and the number of CUDA cores per SM is dependent on the architecture (GPU family) and it is not configurable.

Fig. 12 shows the performance of the different platforms for the inference stage, once the network has been trained. It is assumed that the usual target to perform training is the GPU platform, as the involved convolution operations and their high arithmetic intensive computations are much more efficient on GPU than on CPU. We can observe how the proposed network can perform activity recognition above what it is considered real time in video analysis (i.e., 30 skeletons/s) for every platform. The processing speed exceeds 300 skeletons/s even when the inference process takes place on an embedded architecture such as the NVIDIA Jetson TK1. The most recent GPU-considered platform performs up to 12 times faster than the CPU-only one. More importantly, we observe the good scalability of the raw performance as the GPU hardware evolves.

5. Conclusions

This paper presents a deep learning approach for activity and gesture recognition with 3D spatiotemporal data based on a combination of a Convolutional Neural Network (CNN) and a Long Short-Term Memory (LSTM) network. We have experimentally demonstrated that a CNN is able to extract relevant features from 3D skeleton data to subsequently tackle the activity recognition by means of a LSTM. We consider this fact as a major contribution of the paper. As an additional important contribution, we highlight the proposed data augmentation procedure, which improves the network performance and prevents overfitting as much as possible. We also present an extensive state-of-the-art study, which includes all the papers identified in the literature that address the problems considered from a perspective of 3D spatiotemporal data analysis. We have also identified the reference datasets in this research area. We apply our general proposal to human activity recognition based on 3D full body skeletons and hand gesture recognition from 3D hand skeletons. To test the suitability of our proposal, we have conducted an exhaustive experimental design on six benchmark datasets and a quantitative comparison of our method with respect to the related work. Experimental results show that our method obtains good performance results with datasets of very different sizes, specially remarkable considering the lightweight computation of our proposal. However, the best performance is obtained for small datasets, where the proposed data augmentation stage has greater impact on the learning process. Finally, it is also important to emphasize that our proposal can be easily generalized to other problems dealing with 3D spatiotemporal data, as it has no dependency on the particular shape of the skeleton or the gestures/actions to be recognized.

Acknowledgments

This research has been partially supported by the Spanish Government research funding refs. MINECO/FEDER TIN2015-69542-C2-1, MINECO/ES TIN2014-57458-R and the Banco de Santander and Universidad Rey Juan Carlos Funding Program for Excellence Research Groups ref. Computer Vision and Image Processing (CVIP).

References

[1] A. Farooq, C.S. Won, A survey of human action recognition approaches that use an RGB-D sensor, in: IEEE Transactions on Smart Processing and Computing, 4, 2015, pp. 281–290.

[2] P. Wang, W. Li, Z. Gao, J. Zhang, C. Tang, P. Ogunbona, Deep convolutional neural networks for action recognition using depth map sequences, Comput. Res. Repository (CoRR) abs/1501.04686 (2015).

[3] S. Escalera, X. Baro, J. Gonzalez, M. Bautista, M. Madadi, M. Reyes, V. Ponce, H. Escalante, J. Shotton, I. Guyon, ChaLearn Looking at People Challenge 2014: Dataset and Results, vol. 8925, Lecture Notes in Computer Science, pp. 459–473. 10.1007/978-3-319-16178-5_32

[4] A. Shahroudy, J. Liu, T.-T. Ng, G. Wang, Ntu RGB+D: a large scale dataset for 3d human activity analysis, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1010–1019, doi:10.1109/CVPR.2016.115.

[5] J. Zhang, W. Li, P.O. Ogunbona, P. Wang, C. Tang, RGB-D-based action recognition datasets: a survey, Pattern Recognit. 60 (Supplement C) (2016) 86–105. https://doi.org/10.1016/j.patcog.2016.05.019

[6] L.L. Presti, M.L. Cascia, 3D skeleton-based human action classification: a survey, Pattern Recognit. 53 (Supplement C) (2016) 130–147. https://doi.org/10.1016/j.patcog.2015.11.019

[7] L. Xia, C. Chen, J.K. Aggarwal, View invariant human action recognition using histograms of 3d joints, in: The IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2012, pp. 20–27, doi:10.1109/CVPRW.2012.6239233.

[8] M. Zanfir, M. Leordeanu, C. Sminchisescu, The moving pose: an efficient 3d kinematics descriptor for low-latency action recognition and detection, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2013, pp. 2752–2759, doi:10.1109/ICCV.2013.342.

[9] M. Devanne, H. Wannous, S. Berretti, P. Pala, M. Daoudi, A. Bimbo, Space-time pose representation for 3d human action recognition, in: Proceedings of the ICIAP 2013 International Workshops on New Trends in Image Analysis and Processing (ICIAP), 8158, Springer-Verlag New York, Inc., New York, NY, USA, 2013, pp. 456–464, doi:10.1007/978-3-642-41190-8_49.

[10] A. Chrungoo, S.S. Manimaran, B. Ravindran, Activity Recognition for Natural Human Robot Interaction, vol. 8755, Springer International Publishing, Cham, pp. 84–94. 10.1007/978-3-319-11973-1_9

[11] R. Vemulapalli, F. Arrate, R. Chellappa, Human action recognition by representing 3d skeletons as points in a lie group, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 588–595, doi:10.1109/CVPR.2014.82.

[12] G. Evangelidis, G. Singh, R. Horaud, Skeletal quads: human action recognition using joint quadruples, in: International Conference on Pattern Recognition (ICPR), 2014, pp. 3054–3063, doi:10.1109/ICPR.2014.772.

[13] H. Zhang, L.E. Parker, Bio-inspired predictive orientation decomposition of skeleton trajectories for real-time human activity prediction, in: IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 3053–3060, doi:10.1109/ICRA.2015.7139618.

[14] L. Tao, R. Vidal, Moving poselets: a discriminative and interpretable skeletal motion representation for action recognition, in: IEEE International Conference on Computer Vision Workshop (ICCVW), 2015, pp. 303–311, doi:10.1109/ICCVW.2015.48.

[15] C. Coppola, O. Martinez Mozos, N. Bellotto, et al., Applying a 3d qualitative trajectory calculus to human action recognition using depth cameras, IEEE/RSJ IROS Workshop on Assistance and Service Robotics in a Human Environment, IEEE, 2015.

[16] W. Ding, K. Liu, F. Cheng, J. Zhang, STFC: spatio-temporal feature chain for skeleton-based human action recognition, J. Vis. Commun. Image Represent. 26 (2015) 329–337, doi:10.1016/j.jvcir.2014.10.009.

[17] B.A. Boulbaba, J. Su, S. Anuj, Action recognition using rate-invariant analysis of skeletal shape trajectories, IEEE Trans. Pattern Anal. Mach. Intell. 38 (1) (2016) 1–13.

[18] G. Zhu, L. Zhang, P. Shen, J. Song, An online continuous human action recognition algorithm based on the kinect sensor, Sensors 16 (2) (2016) 161:1–161:18, doi:10.3390/s16020161.

[19] E. Cippitelli, S. Gasparrini, E. Gambi, S. Spinsante, A human activity recognition system using skeleton data from RGBD sensors, Comput. Intell. Neurosci. 2016 (2016) 4351435:1–4351435:14, doi:10.1155/2016/4351435.

[20] C. Wang, Y. Wang, A.L. Yuille, Mining 3d key-pose-motifs for action recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2639–2647, doi:10.1109/CVPR.2016.289.

[21] C. Wang, J. Flynn, Y. Wang, A.L. Yuille, Recognizing actions in 3d using action-snippets and activated simplices, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, in: AAAI'16, AAAI Press, 2016, pp. 3604–3610.

[22] I. Lillo, J.C. Niebles, A. Soto, A hierarchical pose-based approach to complex action understanding using dictionaries of actionlets and motion poselets, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1981–1990.

[23] Y. Du, W. Wang, L. Wang, Hierarchical recurrent neural network for skeleton based action recognition, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, doi:10.1109/CVPR.2015.7298714.

[24] V. Veeriah, V. Zhuang, G. Qi, Differential recurrent neural networks for action recognition, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015, pp. 4041–4049, doi:10.1109/ICCV.2015.460.

[25] J. Donahue, L.A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, T. D., K. Saenko, Long-term recurrent convolutional networks for visual recognition and description, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 2625–2634, doi:10.1109/TPAMI.2016.2599174.

[26] N. Neverova, C. Wolf, G. Lacey, L. Fridman, D. Chandra, B. Barbello, G. Taylor, Learning human identity from motion patterns, IEEE Access 4 (2015) 1810–1820, doi:10.1109/ACCESS.2016.2557846.

[27] N. Neverova, C. Wolf, G. Taylor, F. Nebout, Moddrop: adaptive multi-modal gesture recognition, IEEE Trans. Pattern Anal. Mach. Intell. 38 (8) (2016) 1692–1706. doi.ieeecomputersociety.org/10.1109/TPAMI.2015.2461544

[28] M. Lingfei, L. Fan, Z. Yanjia, H. Anjie, Human physical activity recognition based on computer vision with deep learning model, in: Proceedings of the IEEE International Instrumentation and Measurement Technology Conference (I2MTC), 2016, pp. 1211–1217, doi:10.1109/I2MTC.2016.7520541.

[29] W. Pichao, L. Zhaoyang, H. Yonghong, L. Wanqing, Action recognition based on joint trajectory maps using convolutional neural networks, in: Proceedings of the 2016 ACM on Multimedia Conference (MM '16), 2016, pp. 102–106, doi:10.1145/2964284.2967191.

[30] L. Yanghao, L. Cuiling, X. Junliang, Z. Wenjun, Y. Chunfeng, L. Jiaying, Online human action detection using joint classification-regression recurrent neural networks, in: Proceedings of the European Conference on Computer Vision (ECCV), in: Lecture Notes in Computer Science, 9911, 2016, pp. 203–220, doi:10.1007/978-3-319-46478-7_13.

[31] J. Liu, A. Shahroudy, D. Xu, G. Wang, Spatio-temporal LSTM with trust gates for 3d human action recognition, in: Proceedings of the European Conference on Computer Vision (ECCV), in: Lecture Notes in Computer Science, 9907, 2016, pp. 816–833, doi:10.1007/978-3-319-46487-9_50.

[32] B. Mahasseni, S. Todorovic, Regularizing long short term memory with 3d human-skeleton sequences for action recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 3054–3063, doi:10.1109/CVPR.2016.333.

[33] W. Zhu, C. Lan, J. Xing, W. Zeng, Y. Li, L. Shen, X. Xie, Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, in: AAAI'16, AAAI Press, 2016, pp. 3697–3703.

[34] J.F. Hu, W.S. Zheng, J.H. Lai, J. Zhang, Jointly learning heterogeneous features for rgb-d activity recognition, IEEE Trans. Pattern Anal. Mach. Intell. PP (99) (2017), doi:10.1109/TPAMI.2016.2640292. 1–1

[35] S. Song, C. Lan, J. Xing, W. Zeng, J. Liu, An end-to-end spatio-temporal attention model for human action recognition from skeleton data, in: Proceedings of the 2017 AAAI Conference on Artificial Intelligence, 2017, pp. 4263–4270.

[36] F. Han, B. Reily, W. Hoff, H. Zhang, Space-time representation of people based on 3d skeletal data: a review, Comput. Vis. Image Understanding 158 (Supplement C) (2017) 85–105. https://doi.org/10.1016/j.cviu.2017.01.011

[37] B. Ionescu, D. Coquin, P. Lambert, V. Buzuloi, Dynamic hand gesture recognition using the skeleton of the hand, EURASIP J. Appl. Signal Process. 13 (2005) 2101–2109, doi:10.1155/ASP.2005.2101.

[38] K. Reddy, P. Latha, M. Babu, Hand gesture recognition using skeleton of hand and distance based metric, Adv. Comput. Inf. Technol. 198 (2011) 346–354, doi:10.1007/978-3-642-22555-0_36.

[39] C. Wang, S.C. Chan, A new hand gesture recognition algorithm based on joint color-depth superpixel earth mover's distance, 4th International Workshop on Cognitive Information Processing (CIP), 2014, doi:10.1109/CIP.2014.6844497.

[40] Q. De Smedt, H. Wannous, J.P. Vandeborre, Skeleton-based dynamic hand gesture recognition, in: The IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2016, pp. 1206–1214, doi:10.1109/CVPRW.2016.153.

[41] Intel, RealSense SDK Developer Guide 6.0, Technical Report, 2015.

[42] Y. Lecun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444, doi:10.1038/nature14539.

[43] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780, doi:10.1162/neco.1997.9.8.1735.

[44] X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, in: G.J. Gordon, D.B. Dunson (Eds.), Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS-11), 15, 2011, pp. 315–323. Journal of Machine Learning Research - Workshop and Conference Proceedings

[45] R. Raina, A. Madhavan, A.Y. Ng, Large-scale deep unsupervised learning using graphics processors, in: Proceedings of the 26th Annual International Conference on Machine Learning, in: ICML '09, ACM, New York, NY, USA, 2009, pp. 873–880, doi:10.1145/1553374.1553486.

[46] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.

[47] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Comput. 1 (4) (1989) 541–551, doi:10.1162/neco.1989.1.4.541.

[48] A.K. Jain, J. Mao, K.M. Mohiuddin, Artificial neural networks: a tutorial, Computer 29 (3) (1996) 31–44, doi:10.1109/2.485891.

[49] W. Li, Z. Zhang, Z. Liu, Action recognition based on a bag of 3D points, in: The IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2010, pp. 9–14, doi:10.1109/CVPRW.2010.5543273.

[50] J. Wang, Z. Liu, Y. Wu, J. Yuan, Mining actionlet ensemble for action recognition with depth cameras, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 1290–1297, doi:10.1109/CVPR.2012.6247813.

[51] D.R. Faria, M. Vieira, C. Premebida, U. Nunes, Probabilistic human daily activity recognition towards robot-assisted living, in: 24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), 2015, pp. 582–587, doi:10.1109/ROMAN.2015.7333644.
[52] M.D. Zeiler, ADADELTA: an adaptive learning rate method, Comput. Res. Repository (CoRR) abs/1212.5701 (2012).
[53] C. Chen, R. Jafari, N. Kehtarnavaz, Action recognition from depth sequences using depth motion maps-based local binary patterns, in: Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV),Waikoloa Beach, HI, 2015, pp. 1092–1099, doi:10.1109/WACV.2015.150.
[54] M.A. Gowayyed, M. Torki, M.E. Hussein, M. El-Saban, Histogram of oriented displacements (hod): describing trajectories of human joints for action recognition, in: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence (IJCAI), in: IJCAI '13, AAAI Press, 2013, pp. 1351–1357.
[55] R.E.F. Behnam, Stats-calculus pose descriptor feeding a discrete HMM low-latency detection and recognition system for 3D skeletal actions, Comput. Res. Repository (CoRR) abs/1509.09014 (2015).
[56] R. Anirudh, P. Turaga, J. Su, A. Srivastava, Elastic functional coding of human actions: from vector-fields to latent variables, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3147–3155, doi:10.1109/CVPR.2015.7298934.
[57] G. Ling, C. Fu, Human action recognition using APJ3D and random forests, J. Softw. (JSW) 8 (9) (2013) 2238–2245, doi:10.4304/jsw.8.9.2238-2245.
[58] A. Liu, W. Nie, Y. Su, L. Ma, T. Hao, Z. Yang, Coupled hidden conditional random fields for RGB-D human action recognition, Signal Process. 112 (C) (2015) 74–82, doi:10.1016/j.sigpro.2014.08.038.
[59] Y. Zhu, W. Chen, G.-D. Guo, Fusing spatiotemporal features and joints for 3Daction recognition, in: The IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2013, pp. 486–491, doi:10.1109/CVPRW.2013.78.
[60] M. Jiang, J. Kong, G. Bebis, H. Huo, Informative joints based human action recognition using skeleton contexts, Signal Process. 33 (2015) 29–40, doi:10.1016/j.image.2015.02.004.
[61] I. Theodorakopoulos, D. Kastaniotis, G. Economou, S. Fotopoulos, Pose-based human action recognition via sparse representation in dissimilarity space, J. Vis. Commun. Image Represent. 25 (1) (2014) 12–23. https://doi.org/10.1016/j.jvcir.2013.03.008.
[62] J.Y. Chang, Nonparametric Gesture Labeling from Multi-modal Data, Springer International Publishing, Cham, pp. 503–517. 10.1007/978-3-319-16178-5_35

Juan Carlos Núñez received his M.Sc. in Computer Vision at Universidad Rey Juan Carlos in 2013. He is currently a Ph.D. student. From 2013 to present he is working inhms, especially in deep learning area. Verisk. His research interests include computer vision and machine learning algorithms.

Raúl Cabido received his Ph.D. in Computer Science in 2010 from Universidad Rey Juan Carlos. He is member of the CAPO research group in the Department of Computer Science. His research interests includes computer vision, GPU computing and acceleration of heuristics and metaheuristics procedures.

Juan J. Pantrigo is an Associate Professor at the Universidad Rey Juan Carlos (Spain) where he is a member of the CAPO research group. His main research interests focus on the interface among Computer Science, Artificial Intelligence, Computer Vision and Operations Research.

Antonio S. Montemayor is Associate Professor in Computer Science at Universidad Rey Juan Carlos, Madrid (Spain). He leads the CAPO research group and his research interests include computer vision, HPC, GPU computing, and real-time implementations.

José F. Velez is Associate Professor at the Universidad Rey Juan Carlos (Spain) and he is member of the GAVAB research group. His research interests are related to the developing of Computer Vision Industrial Applications derived from his experience working in research departments of several Spanish companies.