2017 26th IEEE International Symposium on
Robot and Human Interactive Communication (RO-MAN)
Lisbon, Portugal, Aug 28 - Sept 1, 2017.

Two Deep Approaches for ADL Recognition: a Multi-scale LSTM and a CNN-LSTM with a 3D Matrix Skeleton Representation

Giovanni Ercolano[1] and Daniel Riccio[1] and Silvia Rossi[1]

***Abstract*— In this work, we propose a deep learning approach for the detection of the activities of daily living (ADL) in a home environment starting from the skeleton data of an RGB-D camera. In this context, the combination of ad hoc features extraction/selection algorithms with supervised classification approaches has reached an excellent classification performance in the literature. Since the recurrent neural networks (RNNs) can learn temporal dependencies from instances with a periodic pattern, we propose two deep learning architectures based on Long Short-Term Memory (LSTM) networks. The first (MT-LSTM) combines three LSTMs deployed to learn different time-scale dependencies from pre-processed skeleton data. The second (CNN-LSTM) exploits the use of a Convolutional Neural Network (CNN) to automatically extract features by the correlation of the limbs in a skeleton 3D-grid representation. These models are tested on the CAD-60 dataset. Results show that the CNN-LSTM model outperforms the state-of-the-art performance with 95.4% of precision and 94.4% of recall.**

I. INTRODUCTION

People with mild cognitive impairment, like many older people, prefer, when possible, to live in their own home. These people are at high risk of not sufficiently taking care of themselves, so the possibility of having a mobile robot for tracking the user [1] and for monitoring the proper handling of their personal needs, the so-called Activities of Daily Living (ADL), and the detection of deviations from common patterns is a primary challenge in supporting this type of people and their caregivers [2].

In this context, the main purpose of this work is the development of a classification algorithm for the recognition of the ADL in an indoor environment with the extraction of the skeleton data by an RGB-D camera. In literature, excellent classification performance is achieved by the combination of ad-hoc feature extraction/selection algorithms with supervised classification approaches [3], [4], [5]. A discussion on related works is provided in Section V. On the contrary, in this paper, we want to explore the ability of Deep Neural Networks to act as features extractors and learn long-term temporal dependencies by using a type of Recurrent Neural Networks (RNNs) that is called Long Short-Term Memory (LSTM) [6].

In particular, we propose two different deep models based on LSTM for learning the temporal dependencies. The first model is a multi-scale LSTM (MT-LSTM) and it is mainly inspired by the work of [7] in the domain of text analysis, differing for the absence of the feedback connections between the LSTMs. In the second model, we explore the possibility to use a combination of CNN and LSTM (CNN-LSTM) networks to extract automatically the spatial and temporal information from the raw data. The CNN features extractor capabilities are typically used in case of image recognition. Since skeletal data has a different structure, we defined an input model that learns limb correlations as in [8]. These two models are tested on the Cornell CAD-60 activity dataset [9]. Results show that the CNN-LSTM model outperforms the state of the art results.

[1]are with the Department of Electrical Engineering and Information Technologies, Università degli Studi di Napoli Federico II, Napoli, Italy {daniel.riccio,silvia.rossi}@unina.it

II. MATERIALS AND METHODS

In this section, we introduce some background concepts regarding the deep learning algorithms that we combined in two architectures described in Section III.

A. RNN and LSTM

Recurrent Neural Networks (RNNs) are models of Artificial Neural Networks (ANN) characterized by a sharing of parameters across different parts of the model allowing the generalization to sequence's lengths that did not appear in the training set. The RNN accepts an input vector $(x^1, x^2, \ldots, x^{T-1}, x^T)$ mapping it to a fixed length sequence of hidden state vectors $(h^1, h^2, \ldots, h^{T-1}, h^T)$. Every hidden vector at time t receives the input vector at time t and the previous hidden vector at time $t-1$ using the equation $h^t = g(Ux^t + Wh^{t-1} + b)$, where g is the non-linear activation function, U is the matrix of weights between the input and hidden layer, W is the matrix of weights between the hidden layer at time t and the one at time $t-1$, and b is the bias vector. The weight matrices U, W and the bias vector b are trained with the backward propagation. The backward propagation algorithm for the RNN is called Back-Propagation Through Time (BPTT), that is an extension of the back-propagation.

While the basic RNNs suffer the vanishing/exploding gradient problem [10], the Long Short Term Memory (LSTM) [6] can handle this problem and learn long-term dependencies. The LSTM can be seen as a block with an internal recurrence in addition to the outer recurrence of the RNN. Every LSTM block is a system of gating units: the state unit s^t (Equation 4), the forget gate unit f^t (Equation 1), the external input gate unit g^t (Equation 2) and the output gate unit o^t (Equation 3). An LSTM block receives in input the input vector x^t, the hidden vector h^{t-1} and the state vector s^{t-1} and returns in output the state vector s^t (Equation 4) and the hidden vector h^t (Equation 5) (see Figure 1). The gate and output units are computed using the following equations:

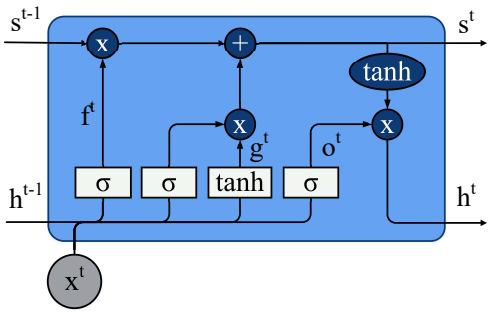

Fig. 1. A diagram of one LSTM

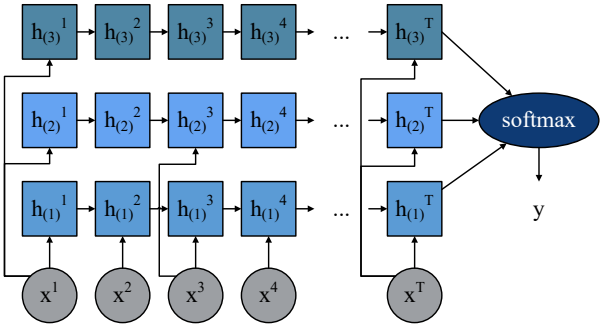

Fig. 2. Multi-scale LSTM architecture with three LSTM that capture different timescales dependencies considering speed variations of the motion

$$f^t = \sigma(b^f + U^f x^t + W^f h^{t-1}), \quad (1)$$
$$g^t = \tanh(b^g + U^g x^t + W^g h^{t-1}), \quad (2)$$
$$o^t = \sigma(b^o + U^o x^t + W^o h^{t-1}) \quad (3)$$
$$s^t = f^t s^{t-1} + g^t \sigma(b^i + U^i x^t + W^i h^{t-1}), \quad (4)$$
$$h^t = \tanh(s^t) o^t, \quad (5)$$

where $\sigma(x) = 1/(1 + e^{-x})$ is the sigmoid activation function, $\tanh(x) = (e^x - e^{-x})/(e^x + e^{-x})$ is the hyperbolic tangent activation function, U^k, W^k and b^k with $k \in \{f, i, g, o\}$ are respectively the weight matrices and the bias vectors. The weight matrices and bias vectors are the parameters that are trained with the BPTT.

B. CNN

Convolutional Neural Networks (CNNs) [11] are deep models for processing data with a grid-like topology like image data (2D grid). A CNN can be thought of as a hierarchy of one, two or more convolutional modules that progressively learn higher-level features followed by one or two full connected layers that classify the extracted features.

The main characteristic of a CNN are the *sparse connectivity*, the *parameter sharing*, and the *equivariant representations*. In detail, with respect to a traditional ANN that has each input node connected to each output node, the CNN, instead, typically has sparse weights making the kernel smaller than the input. Moreover, while the traditional neural network has a multiplication between each element of the input and each element of the weights matrix, in the CNN, each member of the kernel is used at every position of the 2D input grid; it means that we learn only one set of parameters instead of different sets for every location. Finally, the CNN has equivariance to translation, but not to scaling or rotation. Typically, a convolutional module is composed by:

- a **convolution layer** that is a bank of affine transformations of input or in other words a bank of convolutional filters (also called kernels) applied on the 2D grid input;
- a **detector layer** that applies a non-linear activation function (typically the rectified linear unit - RELU);
- a **pooling layer** that reduces the input size (therefore it reduces the number of parameters) and improves the statistical efficiency.

The weights of the kernels in the convolutional layer are the parameters to learn to perform the training with stochastic gradient descent. Thanks to sparse connectivity and parameter sharing, the number of CNN's weights is reduced compared to the feed forward network.

III. THE PROPOSED APPROACH

In this section, we introduce the two proposed models: one is a multi-scale LSTM architecture (MT-LSTM) to learn multi-scale temporal dependencies and the other is based on a combination of CNNs and LSTMs to extract the relevant features and learn possible periodic temporal patterns of the action. While the first model receives in input directly the pre-processed joints coordinates, in the second model, in order to be effectively processed by the CNN, joints data are opportunely transformed into a 3D grid.

A. Multi-scale LSTM

The first model is inspired by the Multi-scale LSTM in [7] and the WHDMM + 3ConvNets in [12] that considers the temporal scale of the activity, one with a composition of LSTMs in the domain of text analysis and the other with a composition of CNNs that analyze different temporal scale of the sequences. We want to test the potentialities of the LSTMs to detect the temporal dependencies by considering some different speeds of the motion, and so different lengths in the input vector. We give in input to the deep architecture only the pre-processed features of the skeleton data (see Section IV-B for details on pre-processing).

In detail, the proposed deep architecture is formed by a number of LSTM that work in parallel on a subset of the instance of the input. Each LSTM is concatenated in a merging layer. The last layer is a full-connected one with the softmax activation function for recognizing the activity. The first LSTM receives in input the full instance, the second one receives an instance that is constructed by taking one frame every two of the full instance, and the nth one receives the instance that is constructed taken one frame every 2^n frames of the full instance. The nth LSTM can capture different timescale dependencies considering speed variation of the motions (see Figure 2 in the case of 3 LSTM). In particular, by providing more or fewer frames of the same activity the network is trained on different possible velocities starting

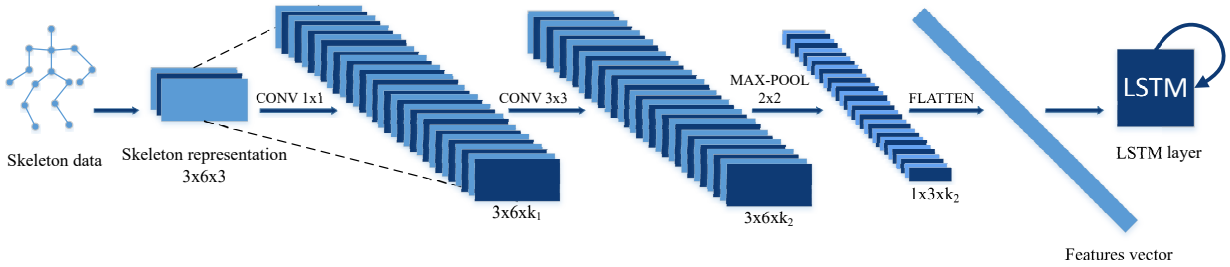

Fig. 3. Combination of a CNN for automatic features extraction from the skeleton representation and a LSTM

from the same input. This model differs from [7] for the absence of the feedback connections between the LSTMs and for the fixed number of LSTMs that is set according to the problem since we want to separately analyze the motions at different speed without change the BPTT.

B. CNN-LSTM

The second model aims to explore the combination of CNN for representation learning and of LSTM for temporal dependencies learning, that is proposed in many applications that concern spatio-temporal classification like in [13] for video description and in [14] for activity recognition from wearable devices data. The CNNs are deep neural networks for processing grid-like topology data (i.e., image data). The skeleton data can be represented as an image. In [8] the coordinates (x, y, z) became the RGB channels of an image. We propose a different representation of the joints values from 1D to 2D grid (a 3D grid when considering all the coordinates) that aims to be invariant to translation, rotation, and scale. To identify the possible dependencies of the limbs' data and to automatic extract the features, we propose a deep architecture with the concatenation of a CNN for the extraction of significant posture representation features, an LSTM for temporal dependencies identification, and a final full-connected layer with softmax activation function.

For each frame, the human skeleton joints are divided into a trunk, and two arms and two legs, that are constituted by three joints each. We started with the intent to recognize the spatial dependencies between the human limbs that are bound by the human structure. Hence, given the vectors of three points a_l, a_r, t, l_l, l_r, that respectively represents the left arm, the right arm, the trunk, the left leg and the right leg, we model the following matrices representation:

$$\begin{bmatrix} a_l^x[0] & a_l^x[1] & a_l^x[2] & l_l^x[0] & l_l^x[1] & l_l^x[2] \\ a_l^y[0] & a_l^y[1] & a_l^y[2] & l_l^y[0] & l_l^y[1] & l_l^y[2] \\ a_l^z[0] & a_l^z[1] & a_l^z[2] & l_l^z[0] & l_l^z[1] & l_l^z[2] \end{bmatrix}$$

$$\begin{bmatrix} t^x[0] & t^x[1] & t^x[2] & t^x[0] & t^x[1] & t^x[2] \\ t^y[0] & t^y[1] & t^y[2] & t^y[0] & t^y[1] & t^y[2] \\ t^z[0] & t^z[1] & t^z[2] & t^z[0] & t^z[1] & t^z[2] \end{bmatrix}$$

$$\begin{bmatrix} a_r^x[0] & a_r^x[1] & a_r^x[2] & l_r^x[0] & l_r^x[1] & l_r^x[2] \\ a_r^y[0] & a_r^y[1] & a_r^y[2] & l_r^y[0] & l_r^y[1] & l_r^y[2] \\ a_r^z[0] & a_r^z[1] & a_r^z[2] & l_r^z[0] & l_r^z[1] & l_r^z[2] \end{bmatrix}$$

where a 2D grid is represented by either the combination of left arm and leg, the trunk, or the right arm and leg.

The first step is the features learning of each frame of the video, therefore, the matrices representation of the skeleton posture is given to the CNN as input. The proposed CNN is constituted by three layers (see Figure 3). The first layer is a convolutional layer that takes as input the three matrices representing the posture with size $3 \times 6 \times 3$ and it has a set of kernels of size 1×1 with stride 1 for spatial limb dependencies consideration. With k_1 kernels, its output sizes $3 \times 6 \times k_1$. A convolutional layer with kernels sizing 1×1 is equivalent to a parametric pooling layer that performs weighted linear recombination on the input feature maps. The second layer is a convolutional layer that takes as input the output of the first layer and it has a set of kernels of size 3×3 with stride 1. With k_2 kernels, its output sizes $3 \times 6 \times k_2$. The third layer, that halves the resolution of the grid, is a max pooling layer of size 2×2 with stride 2 and its output sizes $1 \times 3 \times k_2$. The max polling layer sizes 2×2 instead of 3×2 because we aim to consider more the informative content of the coordinates x and y and less the coordinate z. The last layer flatten its input concatenating the values in a vector that is the output of the CNN of length $1 \cdot 3 \cdot k_2$.

The output of the CNN is the input for the LSTM layer that accumulates the information of the temporal dependencies between each frame of the video. Since the output of the CNN is a multi-dimensional matrix, we added a flatten layer to flatten the matrix into one dimension and obtain a features vector by concatenation of the matrix weights. Differently from the previous model, this time the considered LSTM is composed of a single layer with the number of neurons that is equal to the number of the feature vector extracted by the CNN. Finally, the output of the LSTM is the input for a full-connected layer with a softmax activation function for inferring the correspondent activity performed in the video.

IV. EXPERIMENTAL EVALUATION

In this section, we introduce the used dataset and, accordingly, describe the specific configuration of the deep architectures. Finally, we describe our experimental results.

A. Dataset

Since our long-term goal is to deploy ADL recognition for elderly health-care monitoring, we decided to train the network with the Cornell Activity Datasets - CAD-60. The CAD-60 [9] is an activities dataset captured by using a Microsoft Kinect sensor. It is formed by 60 RGB-D videos

performed by 4 subjects (two male, two female, one left-handed), performing 12 activities in 5 different environments. Each video comes with RGB and depth images, and the skeletons. There are 38 data for each frame that describe the skeleton data and the correspondent frame number.

We apply a temporal sliding window scheme, obtaining instances of 140 frames, since the smallest activity video is of 147 frames (e.g. with an activity video of 147 frames we obtain 8 instances of 140 frames). Therefore, the full input sequence sizes 140 frames and it is given as input to the CNN-LSTM and MT-LSTM models.

The validation set is the 33% of the training set in all the configurations of the two models.

B. Data Pre-processing

The number of joints in CAD-60 is 15. We pre-process only the coordinates of the joints of 3D Skeleton Data with the following steps:

1) **Symmetrization.** We double the 3D skeleton data with mirrored data of the raw ones to account for right and left-handed people.
2) **Translation.** We define the same origin of coordinates system as the average point between the points of the torso, left and right shoulder, left and right hand.
3) **Normalization.** We normalize the data translated on the new origin with the standard score: $X_{new} = (X - \mu)/\sigma$, to reduce the posture dependence from the limb's length and the person's height; the mean and the standard deviation is computed on all the 140 frames of every instance.

C. Model Settings

In the MT-LSTM model, we considered three layers with one LSTM that has a full instance of 140 frames, the second LSTM has an instance of 70 frames and the third LSTM has an instance of 35 frames. Each LSTM block of the MT-LSTM and CNN-LSTM models is composed of 45 LSTM units due to the number of the training instances. The settings of the CNN and the LSTM are important for gradient convergence and for avoiding over-fitting on a tiny dataset. The LSTM weights are initialized without pre-training by using the random initialization procedure (Glorot normal initializer, also called Xavier normal initializer). In the experiments, we found that dropout set at 0.25 after the max-pooling layer of the CNN and dropout set at 0.5 on LSTM layer (both on input-to-hidden and hidden-to-hidden connections) performed well. The number of kernels in the two convolutional layers is set at 32 since we want to reduce the number of parameters without adding too many kernels.

D. Classification Results

The experimental setup follows the same experiment settings of [15] and runs the "New Person". The New Person setting is also called "Leave One Out (LOO)" cross-validation and it is intended for checking the generalization of the classifier. The model is trained on three of the four people and tested on the fourth. In this work, the New Person approach is selected because it is the most considered in all the works using CAD-60. Since CAD-60 consists of twelve activities performed by four people in five different environments, the dataset is usually evaluated by splitting the activities according to the considered environment and by computing the global performance of the algorithms with the average precision and recall among all the environments.

TABLE I
PRECISION AND RECALL OF THE MT-LSTM MODEL

		MT-LSTM	
Location	Activity	Precision	Recall
Bathroom	brushing teeth	95.67%	100.00%
	random + still	96.34%	85.90%
	rinsing mouth	93.93%	92.79%
	wearing lens	79.36%	98.18%
	Average	93.58%	92.09%
Bedroom	drinking water	95.92%	89.64%
	opening pill container	84.56%	99.51%
	random + still	99.74%	89.38%
	talking on phone	88.66%	97.64%
	Average	95.33%	92.04%
Kitchen	cooking (chopping)	97.11%	99.47%
	cooking (stirring)	97.69%	96.89%
	drinking water	91.26%	377.13%
	opening container	79.76%	99.88%
	random + still	100.00%	88.45%
	Average	96.15%	95.02%
Living room	drinking water	88.01%	91.23%
	random + still	99.74%	90.19%
	relaxing on couch	100.00%	100.00%
	talking on couch	100.00%	100.00%
	talking on phone	88.64%	98.17%
	Average	96.55%	95.11%
Office	drinking water	87.64%	87.35%
	random + still	92.95%	84.87%
	talking on phone	76.18%	98.43%
	working on computer	99.73%	100.00%
	writing on whiteboard	67.54%	75.52%
	Average	85.86%	87.78%
Overall Average		**93.32%**	**92.41%**

Table I and II shows the classification precision and recall of the two proposed models for each environment.

For the MT-LSTM model (see Table I), due to the good discrimination ability for the four people, very good results are achieved in the Kitchen (96.15% and 95.02%) and Living Room (96.55% and 95.12%) environments, while in the Office we obtained the worst results (85.86% and 87.78%). These results confirm the MT-LSTM ability to deal with the activity temporal patterns. Like [5], "talking on the phone" and "drinking water" are similar activities in the Living Room and Office environments, therefore it is difficult for the model to disambiguate both, but unlike [5], "relaxing on couch" and "talking on couch", that are similar activities, are discriminated at 100% maybe for the recognition of the stationary (i.e., the extreme case of a periodic pattern) of "relaxing on couch" unlike the motion of "talking on couch". Note that also "cooking (chopping)" and "cooking (stirring)" are recognized better than [5]. The worst results in the Office on "writing on whiteboard", that has many differences with the other activities of this environment, are due because the model predicts "talking on the phone" or "random" with the third person of the test set, perhaps either for the presence

TABLE II

PRECISION AND RECALL OF THE CNN-LSTM MODEL

		CNN-LSTM	
Location	Activity	Precision	Recall
Bathroom	brushing teeth	100.00%	98.73%
	random + still	93.99%	93.41%
	rinsing mouth	94.60%	87.69%
	wearing lens	89.88%	98.00%
	Average	94.88%	93.92%
Bedroom	drinking water	94.92%	90.71%
	opening pill container	94.04%	96.90%
	random + still	99.46%	96.91%
	talking on phone	89.31%	94.43%
	Average	95.92%	94.73%
Kitchen	cooking (chopping)	88.77%	94.47%
	cooking (stirring)	91.11%	74.77%
	drinking water	99.06%	99.69%
	opening container	85.53%	95.52%
	random + still	95.27%	94.65%
	Average	93.05%	91.56%
Living room	drinking water	99.75%	93.10%
	random + still	98.93%	98.52%
	relaxing on couch	100.00%	100.00%
	talking on couch	100.00%	100.00%
	talking on phone	92.47%	99.22%
	Average	98.66%	98.46%
Office	drinking water	94.95%	90.45%
	random + still	97.78%	93.94%
	talking on phone	80.45%	95.53%
	working on computer	100.00%	100.00%
	writing on whiteboard	94.26%	85.25%
	Average	94.33%	92.97%
Overall Average		**95.40%**	**94.38%**

TABLE III

STATE-OF-THE-ART RESULTS ON CAD-60 DATASET

	"New Person"	
Algorithm	Precision	Recall
Zhu, et al., IVC 2014 [16]	93.2%	84.6%
Faria, et al., RO-MAN 2014 [3]	91.1%	91.9%
Shan, Akella, ARSO 2014 [4]	93.8%	94.5%
Parisi et al., Fron.Neurobot. 2015 [17]	91.9%	90.2%
Cippitelli et al., CIN 2016 [5]	93.9%	93.5%
MT-LSTM	**93.3%**	**92.4%**
CNN-LSTM	**95.4%**	**94.4%**

of noise in the data or for over-fitting on the training set or for similar temporal patterns.

Generally, CNN-LSTM improves throughout, especially in the Office and the Living room it achieves excellent performance. It handles better than the MT-LSTM model "talking on the phone" and "drinking water" that are so similar and difficult to discriminate in Living Room and Office environments, while "writing on whiteboard" is recognized better than the MT-LSTM model on the third person of the test set (100.00% and 50.00% instead of 0.00% and 0.00%). The worst case this time is in the Kitchen where the temporal pattern of the cooking activities was better recognized with the MT-LSTM, that, sometimes, fails to distinguish "cooking (chopping)", "cooking (stirring)" and "opening container" that are similar activities.

The CNN-LSTM model outperforms the state-of-the-art results of activity detection on the CAD-60 dataset with the 95.4% and 94.4% on precision and recall (see Table III).

V. RELATED WORKS

In this section, we introduce works that deal with ADL recognition on the same dataset we used in this work, but with different approaches. Finally, we discuss some approaches on activity recognition that deploy deep learning on different datasets.

First of all, a fundamental step for human activity recognition is the choice of the representation of the 3D skeleton data used for classification. The used features vector has to capture invariant properties of the motion shared across multiple users [18]. For example, for redundancy reduction, in [3], the authors define the centroid of the torso as the origin and computed the joints distances w.r.t. the torso centroid. In [5], action recognition is conducted starting from features vectors obtained using unsupervised learning to select the relevant "key poses" as informative postures to describe the activity. In detail, such poses are computed by a k-means clustering algorithm for each sequence. Therefore, for each activity, K centroids are computed to create the activity features vector. In the same direction, [4] identified key poses using kinetic energy to segment sequences of human action data, performing a robust algorithm respect to the temporal stretching of an action. The selected poses are the poses with the kinetic energy close to zero.

An approach that is more related to our consists in exploiting the automatic learning of the features by deep networks. In [19], the authors proposed an end-to-end hierarchical RNN to extract the skeleton features. With this approach, the skeleton is divided into five parts and fed into a structured neural network formed by bidirectional RNNs. The features extracted by the network are hierarchically fused to build a higher-level representation. To model the temporal dependencies of the features extracted, the final layer of the network is formed by an LSTM. The same authors, in [8], proposed to use a CNN where the representation of the skeleton sequence is a matrix of the concatenation of all frames in chronological order. The matrix is normalized and transformed into an image and it is fed into a CNN model for features extraction and recognition. These approaches were tested on different datasets with respect to the one considered here. Moreover, as discussed in Section III, the representation of the grid is different from the one proposed here.

In [20], a joint Classification-Regression RNN is proposed for activity detection and for localizing the start and end positions of the actions on the fly. They use the LSTM layers for automatic features extraction and for model the long-range temporal dynamics. The framework is constituted by the Deep LSTM Network that extracts the features automatically, by a classification module that classifies the actions and a regression module for better localize the start and end of actions. The detection is performed frame-by-frame, not requiring a sliding window design.

As in [19], in [21] Zhu et al. propose a fully connected deep LSTM network for skeleton based action recognition, introducing a regularization scheme to learn the co-occurrence features of skeleton joints. The framework is

formed by three LSTM layers and two feedforward layers, and it incorporates the co-occurrence regularization into the loss function. This network may automatically explore the conjunctions of discriminative joints and explore different co-occurrences for different types of actions due to the fully connected layer nature. Here, such role is played by the use of a CNN. Moreover, the approach proposed in [21] may not be suitable for the CAD-60 dataset due to the high number of hyper-parameters in the proposed model.

Finally, in [22], [23], [24], exploits the use of 3D CNN also to recognize spatiotemporal features in action recognition problems, applying convolutions on a time series of frames.

VI. CONCLUSIONS

In this work, we presented two deep learning architecture relying on the use of LSTM as memory cells that can learn periodic pattern from input sequences. The challenge of activity recognition from the skeleton data becomes more difficult for RGB-D camera errors, for tiny datasets, and for the different speed motion of the activity performed. The MT-LSTM model is proposed to handle different speed motion whereas the CNN-LSTM model is proposed to exceed the speed dependencies and automatically extracts patterns from the skeleton data with only two small convolutional layers, a pooling layer, and an LSTM.

In the field of gesture, action, and activity recognition there are a lot of public datasets showing different characteristics. For creating a robust and effective recognition system is important to record a large amount of training data. Since ADL large action datasets are not available, it is really challenging to generalize activity recognition systems to real-world applications. The CAD-60 dataset is small. Furthermore, the activity instances are not balanced numerically. Nevertheless, the two proposed approaches reached very good results and the CNN-LSTM model outperforms the state-of-the-art results of activity detection on the CAD-60 dataset with the 95.4% and 94.4%. The experiments show that the multi-scale temporal dependencies in small motion sequences of skeleton data are managed well by the MT-LSTM model, but with a different representation and the features extraction of the skeleton data, performed by the CNN-LSTM model, it is possible to better handle the limbs and temporal dependencies. It is noteworthy that the instances are recognized in a few seconds since 140 frames with $30fps$ correspond to $4.7s$.

ACKNOWLEDGMENT

This work has been partially supported by MIUR within the PRIN2015 research project "User-centered Profiling and Adaptation for Socially Assistive Robotics - UPA4SAR".

REFERENCES

[1] M. Staffa, M. D. Gregorio, M. Giordano, and S. Rossi, "Can you follow that guy?" in *22th European Symposium on Artificial Neural Networks - ESANN*, 2014, pp. 511–516.

[2] S. Rossi, G. Ercolano, and M. Staffa, "Towards an adaptive user monitoring based on personality and activity recognition," in *Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 2017, pp. 269–270.

[3] D. R. Faria, C. Premebida, and U. Nunes, "A probabilistic approach for human everyday activities recognition using body motion from rgb-d images," in *The 23rd IEEE Intern. Symp. on Robot and Human Interactive Communication, RO-MAN*. IEEE, 2014, pp. 732–737.

[4] J. Shan and S. Akella, "3d human action segmentation and recognition using pose kinetic energy," in *IEEE International Workshop on Advanced Robotics and its Social Impacts*. IEEE, 2014, pp. 69–75.

[5] E. Cippitelli, S. Gasparrini, E. Gambi, and S. Spinsante, "A human activity recognition system using skeleton data from rgbd sensors," *Computational intelligence and neuroscience*, vol. 2016, 2016.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] P. Liu, X. Qiu, X. Chen, S. Wu, and X. Huang, "Multi-timescale long short-term memory neural network for modelling sentences and documents." in *EMNLP*. The Association for Computational Linguistics, 2015, pp. 2326–2335.

[8] Y. Du, Y. Fu, and L. Wang, "Skeleton based action recognition with convolutional neural network," in *3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, Nov 2015, pp. 579–583.

[9] "Cornell Activity Datasets: CAD-60 and CAD-120," http://pr.cs.cornell.edu/humanactivities/data.php.

[10] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[11] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Handwritten digit recognition with a back-propagation network," in *Neural Information Processing Systems (NIPS)*, 1989.

[12] P. Wang, W. Li, Z. Gao, J. Zhang, C. Tang, and P. O. Ogunbona, "Action recognition from depth maps using deep convolutional neural networks," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 4, pp. 498–509, 2016.

[13] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *IEEE conference on computer vision and pattern recognition*, 2015, pp. 2625–2634.

[14] F. J. Ordóñez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, 2016.

[15] J. Sung, C. Ponce, B. Selman, and A. Saxena, "Unstructured human activity detection from rgbd images," in *2012 IEEE International Conference on Robotics and Automation*, May 2012, pp. 842–849.

[16] Y. Zhu, W. Chen, and G. Guo, "Evaluating spatiotemporal interest point features for depth-based action recognition," *Image and Vision Computing*, vol. 32, no. 8, pp. 453–464, 2014.

[17] G. I. Parisi, C. Weber, and S. Wermter, "Self-organizing neural integration of pose-motion features for human action recognition," *Frontiers in neurorobotics*, vol. 9, p. 3, 2015.

[18] V. Magnanimo, M. Saveriano, S. Rossi, and D. Lee, "A bayesian approach for task recognition and future human activity prediction," in *The 23rd IEEE International Symposium on Robot and Human Interactive Communication, RO-MAN*, Aug 2014, pp. 726–731.

[19] Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural network for skeleton based action recognition," in *IEEE conference on computer vision and pattern recognition*, 2015, pp. 1110–1118.

[20] Y. Li, C. Lan, J. Xing, W. Zeng, C. Yuan, and J. Liu, "Online human action detection using joint classification-regression recurrent neural networks," in *14th European Conference on Computer Vision – ECCV, Part VII*. Springer, 2016, pp. 203–220.

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

[22] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proc. of the IEEE Intern. Conf. on Computer Vision*, 2015, pp. 4489–4497.

[23] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt, "Sequential deep learning for human action recognition," in *Intern. Workshop on Human Behavior Understanding*, 2011, pp. 29–39.

[24] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, Jan 2013.