A NEW RESIDUAL NEURAL NETWORK ARCHITECTURE USING SKELETON DATA FOR HUMAN ACTIVITY RECOGNITION

Yongda Lin[*] *Xavier Cortés*[†] *Donatello Conte*[‡]
Université de Tours
LIFAT EA 6300, Tours, France

Abstract

Deep learning method has been efficiently applied to Human Activity Recognition in recent years. There are several types of neural network architectures such as convolutional, recurrent with long short term memory, etc. that have been further studied. The recent researches have shown their capability of learning features automatically from videos. However, many challenges still remain open. The key to let machine learn spatio-temporal information remains to be discovered. This paper proposes an original architecture of residual neural network using skeleton data. A new complex linear layer aimed at retrieving the internal relationships of neural cell will be discussed. Our method has been applied on some well-known public data-sets, mainly UTKinect-Action3D and Florence 3D actions. Combining with temporal pre-processing of human activity video, it achieves 99.25% and 99.3% accuracy respectively on UTKinect-Action3D and Florence 3D actions data-set, over-performing by 2% the best state-of-the-art results.

Key Words: human activity recognition, skeleton data, deep learning, residual neural network, cell internal relationship, spatio-temporal information

[*]E-mail address: yongda.lin@etu.univ-tours.fr
[†]E-mail address: xavier.cortes@univ-tours.fr
[‡]E-mail address: donatello.conte@univ-tours.fr

1 Introduction

The recognition of human actions has been a subject of research since the early 1980s, because of its promising results in many applications (see e.g. [**?**], [**?**]). We define gestures like the basic components that describe the meaning of movements; "Raising an arm" and "shaking the supports" are movements in this category. So actions are one-person activities that can be composed of several gestures organized over time; "Walking", "shaking" and "drinking" are examples of simple actions. Then activities are complex sequences of actions performed by many people or objects like "Playing basketball" that is an example of activity consisting of actions such as running, shooting or dribbling. Compared with object recognition in images, human action recognition is much more complex because an action contains spatio-temporal information.

Among the methods of literature (see Section 2), those that have proved to be the most effective to solve the problem dealt with in this paper, are the ones based on structural representation of the information through skeleton data. However, how to represent relationships between the points in the skeleton data along the images sequence, is still an open issue.

The main contribution of this chapter is to propose a new way to learn the representation of spatial relationship within the video, by a learning framework based on deep learning.

The remainder of the paper is organized as follows: after discussing the state-of-the-art and our contributions with respect to it (Section 2), the proposed model is illustrated in Section 3; the effectiveness of our approach is evaluated with several experiments in Section 4 and, finally, conclusions and perspectives will end the article (Section 5).

2 Related Works

Human activity recognition has been the focus of attention for many researchers in recent years. Within this field there are different kinds of approaches, depending on the input data used to perform the recognition and the human behaviour intended to be recognized. For instance, we can categorize the human activities into: gestures [**?**], atomic actions [**?**], human-to-object or human-to-human interactions [**?**], human group behaviours [**?**] among others. In [**?**] there is an extensive survey that summarizes different human activity recognition approaches.

Methods to recognize activities performed by a single human in the scene on RGB videos has been addressed following different strategies. From classical bags of visual words [**?**] or graph-based representations [**?**] to deep learning algorithms [**?**, **?**]. There are works that propose to use data collected by another kind of sensor, such as wearable devices or smartphones [**?**, **?**, **?**].

On the other hand, there are authors that propose to perform the human activity recognition on data captured by RGB-D sensors. These kinds of sensors provide a RGB image, as well as a cloud of 3D points generated using depth sensors. For instance, in [**?**], the authors propose the use of hyper-surface normals, containing geometry and local motion information, extracted from depth sequences, to perform the recognition. Going one step further, there are authors that propose to extract the skeleton joints from depth data, before performing the classification. They assume that an activity could be described by its actions and these actions are defined by the behavior of the skeleton joints over time.

Since, several works have shown that this is an appealing approach to face the problem of human activity recognition. For instance, in [**?**] the authors propose a Multiple Kernel Learning (MKL) algorithm to perform the classification combining depth and skeleton data in a kernel-based machine while in [**?**] authors propose to perform the classification fusing the data at the feature level using several spatio-temporal interest point detectors. Faria et al. in [**?**] face the same problem using a model designed to combine multiple classifier likelihoods, assigning weights to evaluate the likelihood as a posterior probability, while in [**?**] the authors perform the classification by extracting action templates from 3D human joint data. Another interesting approach based on deep learning was presented in [**?**], in which the authors propose a model to perform the recognition on skeleton data using a Convolutional Neural Network (CNN) [**?**].

There are now many datasets that propose videos for action recognition providing directly skeleton data for each video. Two of the most common datasets (that will be described in detail in Section 4) are the UTKinect-Action 3D dataset [**?**] and the Florence 3D actions dataset [**?**]. Since that, numerous works dealt with skeleton data to recognize actions. Vemulapalli et al. [**?**] propose a new representation of the skeleton data: human skeleton is considered as a point in the Lie group (a special Euclidean group SE(3) [**?**]), by explicitly modeling the 3D geometric relationships between various body parts using rotations and translations; using the proposed skeletal representation, human actions are modeled as curves in this Lie group. Liu et al. propose several algorithm for action recognition by skeleton data. In [**?**] a Spatio-Temporal LSTM is introduced; a skeleton tree traversal algorithm is proposed which takes the adjacency graph of body joints into account and improves the performance of the network by arranging the most related joints together in the input sequence. In another paper [**?**] the same authors propose to extend the LSTM network with a Global Context-Aware Attention mechanism in which only most informative joints, in each frame of the skeleton sequence, are selected in order to improve the performances. In [**?**], the authors aggregate features using VLAD algorithm and then a metric learning method, inspired by the Large Margin Nearest Neighbor (LMNN) algorithm [**?**], is used with a k-NN classifier to recognize actions. Taha et al. [**?**] propose to describe an activity through labeling human sub-activities, then the activity recognition problem is viewed as sequence classification problem and Hidden Markov Models (HMM) are employed to recognition task.

Inspired by the last works, with respect to the state of the art, we propose to learn, by a deep learning architecture, the representation of spatial relationship of joints along the image sequences. Instead of give hypothesis about relationship between features, here we learn these relationships from data.

3 Proposed model

Our architecture is divided in two parts: spatial information learning and temporal information learning. It is depicted in Figure 1. We represent each video by a feature vector, which models the spatio-temporal information from skeleton data. We propose a method designed to extract these features in the following two steps: *Unsupervised learning* for temporal information extraction as well as *Residual neural network* for spatial relationship extraction.

1. *Temporal information extraction*: this step is related to selecting a representative set of actions and time-ordering them using information (skeletal data) extracted during data collection. This part is described in subsection 3.2.

2. *Spatial relationship extraction*: the goal of the next step is to learn the relationship between this set of representative actions. A residual neural network contains a connection skip layer, which focuses on the correlation of these actions and their normalization. We build this network with our complex linear layer as we explain in subsection 3.3.

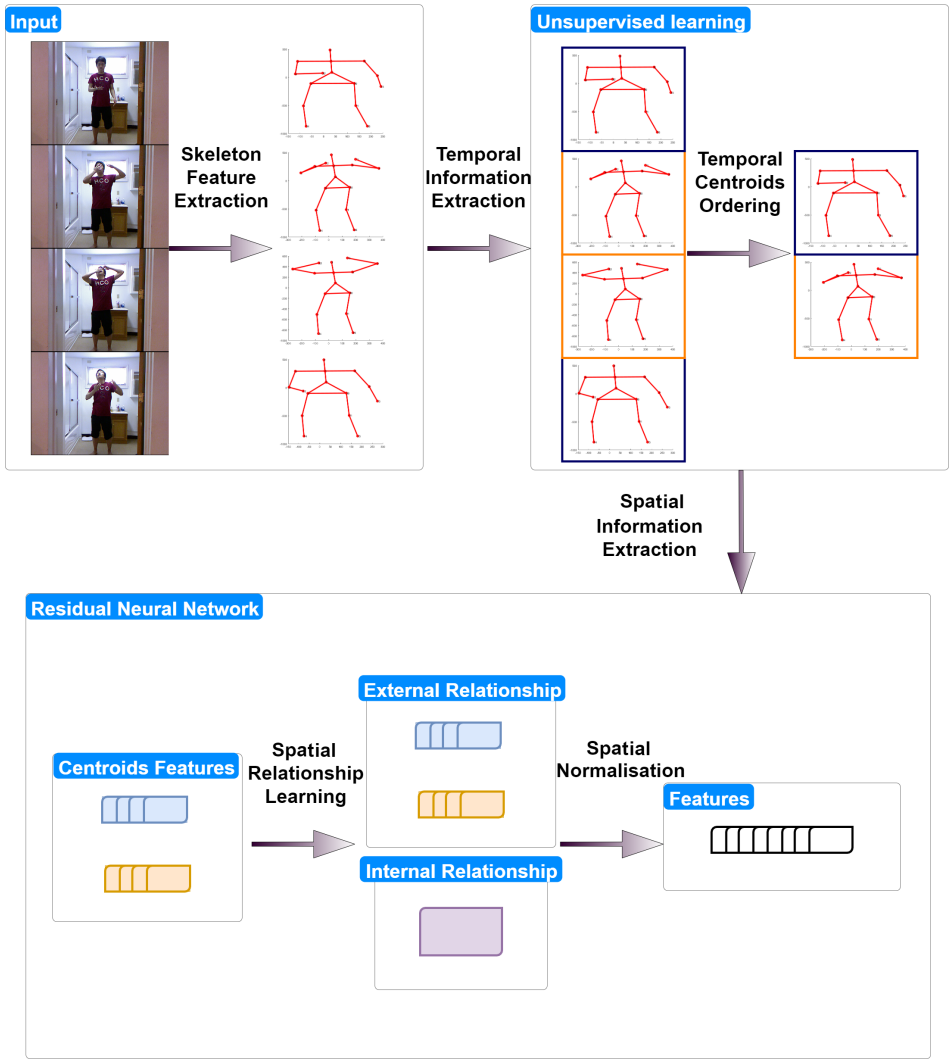


Figure 1: General architecture of our method

3.1 Skeleton data

The input of our method is the 3D spatial coordinates extracted from the joints of the human skeleton and the RGB images of each frame. The Kinect library provides 20 skeletal joints, but in order to discard irrelevant joints when performing classification, we only select 15 joints (see Figure 2): blue circles are the considered joints, while red circles are the discarded joints in our method. Numbers are used to identify each joint.

We use the point coordinates as data descriptors.

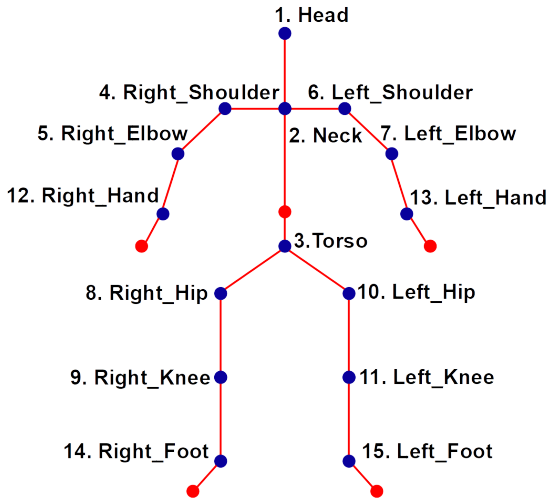


Figure 2: Skeleton joint positions provided by Kinect

3.2 Temporal information

First, it is important to know that videos in the data-sets have different lengths. Inspired by the work of [**?**], we propose an algorithm to normalize video frames size and extract temporal information at the same time. This is a pre-processing step concerning K-means clustering as well as a temporal centroids ordering.

We recommend using the K-means algorithm to cluster the elements of each sequence, because in our case it has a good balance between accuracy and runtime. K-means clustering is designed to retrieve the representation actions in the activity video. It aims to divide N observations into K clusters, where each observation belongs to the cluster with the smallest distance between the observation and the prototype of the cluster. This leads to the division of the data space into Voronoi cells [**?**].

Then, a step for sorting these representative actions is performed. Therefore, the result of the preprocessing phase, not only contains the centroids features, but also the sequence of centroid labels along the image sequence. We assign each frame to the cluster it belongs to based on its characteristics (as we said above, to the cluster having the prototype with the small distance with the observation). Next, we sort the clusters in time according to their order of appearance. Taking a video with **7** frames, as an example, follows

[$C_1, C_1, C_1, C_2, C_2, C_1, C_3$**]** (after a K-means clustering, where **K** is equal to **3**). When ordering the sequence, we group together the frames with the same label and we keep the order of appearance of centroid labels. For the same example above, it is reasonable to put centroid *2* before centroid *3* when we try to reconstruct the sequence with size of **K**. However, there is an uncertainty between centroid *1* and centroid *2*. Our approach uses the strategy that takes only the maximum continuous size of centroids (*the longest chain where the same cluster appears consecutively*) into account. Following the same example, the result becomes **[**C_1, C_2, C_3**]** because the maximum continuous size of C_1 is *3* which appears before centroid *2*. We have now considered the order of appearance to infer the chronological order of the clusters to model the order of a human activity.

3.3 Spatial information

Here are the steps to learn the representation of spatial relationship of the centroids within a video. Considering each centroid of video frames as an unit, we can then separate two concepts of its spatial information:

- *External relationship*: Features which represents the relationship between different units.
- *Internal relationship*: Features which represents the relationship of members within a unit.

3.3.1 External relationship

External relationship is a common type information that often is defined by means of the relationships between different centroids. As external relationship we define a weight that measure the mapping from a centroid to all the other centroids. This kind of information can be easily learned and generated through a fully-connected linear layer (Equation 1):

$$\underset{N_{out}\times K}{y_{ex}} = \underset{N_{out}\times N_{in}}{A_{ex}} \times \underset{N_{in}\times K}{x} \tag{1}$$

where:

- N_{in}: Size of an input sample
- N_{out}: Size of an output sample
- K: Number of centroids in K-means clustering
- A_{ex}: Weight of external relationship betweencluster units
- x: Video data descriptors (Skeleton coordinates)

The A_{ex} in the Equation 1 is trying to project the coordinates of joints in a larger dimension in case that only one orthogonal linear transformation is not capable to explore the potential scalar projection between two identical elements in a same or smaller dimension. We then can naturally suppose that N_{out} is greater than N_{in}. This layer can be thought of as a convolutional hidden layer on which convolutions are performed on weighted inputs then passed to an activation function to produce the actual output.

3.3.2 Internal relationship

The internal relationship is inspired by [**?**]. This relationship aims to find the relevant and normalised spatial information within a centroid. This means that it no longer discovers the difference or the relationship between units (as for external relationship), but rather the relationship between the components (joints) inside a unit.

Internal relationship layer learns this relationship for all the centroids (Equation 2):

$$\underset{1\times K}{y_{in}} = \overline{\underset{N\times N}{A_{in}} \times \underset{N\times K}{x}} \tag{2}$$

where:

- N: Size of each input sample
- K: Number of centroids in K-means clustering
- A_{in}: Weight of internal relationship within unit
- x: Skeleton data of the video

3.3.3 Observations

It is important to emphasize that our model contains one and only one weight vector ($\underset{N\times N}{A_{in}}$ in Equation 2) for the whole video (all the centroids). Whereas for the relationship between one centroid and all other centroids (external relationship), the weight $\underset{N_{out}\times N_{in}}{A_{ex}}$ (Equation 1) is not unique. This configuration of parameter follows the logic (Figure 3) that the internal relationship is used for each activity where different people have different performance. On the other hand, the external relationship has objective to search the difference between the actions (representative centroids) in one activity.

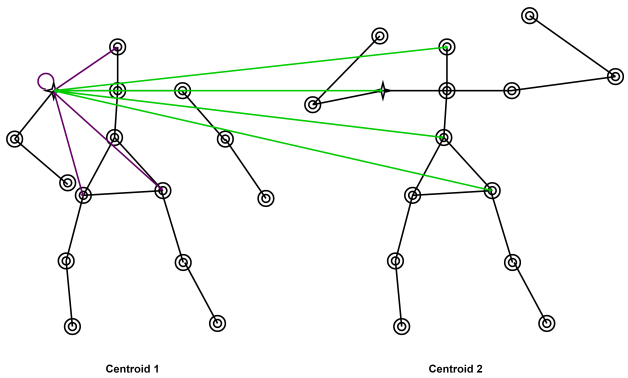


Figure 3: Relationship example between Centroid 1 and Centroid 2

In Figure 3 there is a visual example to expain the mean of external and internal relationships:

- Green lines represent the external relationship between Centroid 1 (*right_shoulder, head, torso, ..., right_hip members*) and Centroid 2 (*right_shoulder, head, torso, ..., right_hip members*).
- Purple lines represent the internal relationship between *right_shoulder* and *itself, head, torso, ..., right_hip members*.

3.3.4 Construction of our residual neural network

The next step in our approach concerns how to use these two relationships. Here we use a residual neural network with one connection skip layer. As we discussed in the former part, the internal relationship reveals the form of the actor in a video and the external explores the information of representative actions in an other dimension. We then can model these concepts in the following formula which attempts to extract the features of spatial information in a normalized way (see Equation 3). The connection skip layer here is used to sum up this normalization.

$$\underset{N_{out}\times K}{y^{'}} = \underset{N_{out}\times K}{y_{ex}} \;/\; \underset{1\times K}{y_{in}} + \underset{N_{out}\times K}{b} \tag{3}$$

Equation 3 formula can be explained in a more comprehensive way in Equation 4. Note that this equation only considers how to weight the different centroids, and does not involve the projection in a larger dimension yet.

$$\begin{aligned} y_i^{'} &= \frac{y_{ij}^{ex}}{y_i^{in}} + b_{ij} \\ &= \frac{\sum_i^K \sum_j^{N_{out}} w_{ij}^{ex} \times x_{ij}}{\frac{1}{N_{in}} \sum_i^{N_{in}} \left(w_i^{in} \times x_i\right)^2} + b_{ij} \end{aligned} \tag{4}$$

At the end, this residual neural network can be described as Figure 4. The input of network from our unsupervised learning goes into two layers at the same time. The fully connected linear layer works for the external relationship and the relatively connected linear layer is used for internal relationship. It is not logical for the norm to be negative when calculating, therefore we add a penalty factor by putting a Rectifier Linear Unit layer after this relatively connected linear layer [**?**].

Then, we can see that the learnt features are obtained by two different linear layers (*fully connected linear layer* and *relatively connected linear layer*). It is not a typical usage of connection skip in residual neural network, which is why we call it an original complex linear layer. This layer is shown in detail in Figure 4*(b)*. It is important to remark that this complex linear layer contains only one parameter (N_{out} in Equation 1). This avoids the problem of excessive human intervention when it comes to the parameters configuration in neural network.

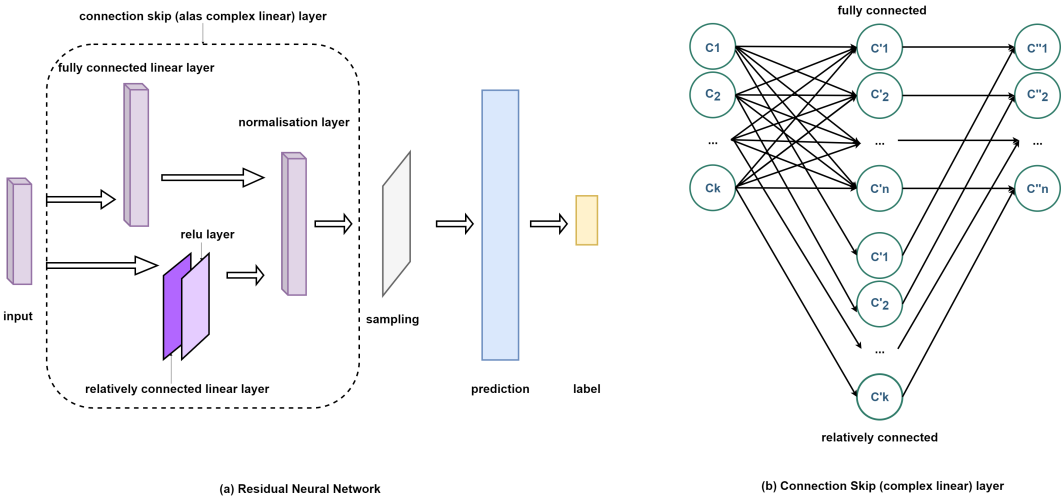


Figure 4: The connection skip (alas complex linear in our paper) layer in residual neural network

4 Experimental Results

In this section, we will show the experimental results obtained by our model, which uses different data sets commonly used for comparison purposes for human activity recognition problems. We also analyze how the configuration parameters affect the performance of the model and try to prove the advantages of the proposed residual neural network.

4.1 Setup

We performed experiments using two benchmark datasets (UTKinect-Action3D and Florence 3D actions), each dataset having a set of videos, including skeleton joints extracted from each frame. Each video is labeled with a specific activity performed by humans. As we discuss in subsection 3.1, input data for each frame contains 15 pose joints in 3D. We then can infer that $N_{in} = N = 45$ in Equation 1 and Equation 2.

The purpose of each experiment is to classify the action performed by a human regardless of the object performing the action and the environment in which the action is performed. We use *cross-validation* methodology to evaluate the experiments, ignoring one subject a time. In the following sections, we describe the results obtained for each dataset.

4.2 UTKinect-Action3D dataset

The UTKinect Action3 Dataset [**?**] has, in total, 200 videos of 10 subjects performing 10 different activities (*walk, sit down, stand up, pick up, carry, throw, push, pull, wave hands, clap hands*). Each subject performs each actions twice. Three channels were recorded: RGB, depth and skeletal joint positions. The three channels are synchronized. The frame rate is 30fps. For each frame, the dataset includes a RGB image as well as the data corresponding to the joints (the spatial coordinates).

In order to illustrate the performance of our method in different contexts, we show the average results varying the activity represented by a confusion matrix in Figure 5 with an accuracy of 99.25%. This result is achieved by our method after repeating the experiment 50 times. For each experiment, we set $K = 5$ for the K-means in our unsupervised learning and $N_{out} = 45$ for the fully connected linear layer in Equation 1. In terms of N_{out}, we can interpret that the 3D dimension for these skeleton joints is already representative and illustrative. The residual neural network did not need to speculate an extra information by projecting in a greater space.

	carry	clapHands	pickUp	pull	push	sitDown	standUp	throw	walk	waveHands
carry	1									
clapHands		1								
pickUp			1							
pull				1						
push					1					
sitDown						1				
standUp							1			
throw		0.025			0.05			0.925		
walk									1	
waveHands										1

Figure 5: Confusion matrix of UTKinect-Action3D Dataset

We can see that the most challenging activity in this dataset is the *throw*. There is 0.75% in a *throw* video might be misclassified. The similar activities among the dataset include *clap hands* and *push*. And the chance of being misjudged in *push* is greater than the *clap hands*. However, this confusion is not symmetrical. The precision of these two activities is very high.

4.3 Florence 3D actions dataset

The Florence 3D actions Dataset (florence) [**?**] is composed of a collection of skeleton joints positions captured using a Kinect camera. This dataset includes 9 activities: *waving a hand, drinking from a bottle, answering a call, clapping, tight lace, sitting down, standing up, looking at a watch, and bowing*. During the acquisition process, 10 subjects were required to perform the above action 2 or 3 times. Therefore, there are 215 active samples in this dataset.

In Figure 6, we show the accuracy results achieved (99.3% in average) by our method after repeating the experiment 50 times. In this case, the parameter used for the K-means was $K = 7$ with $N_{out} = 80$ according to the metric presented in Equation 1. This is a challenging dataset in which some of related work did not outperform the state-of-art when published. The parameter of K-means ($K = 7$) is already very close to the smallest length of frames sequences (8), whereas the other videos contains $12 \sim 22$ frames. We can then assume that there is an information loss in this dataset.

Regarding to the $N_{out} = 80$ in Equation 1, we can infer that the input dimension to represent skeleton joints is not enough illustrative. The residual neural network add and speculate extra information when exploring the external relationship.

Finally, in Figure 6 we show the resulting confusion matrix of the experiments performed with this dataset. *Clap* and *drink from a bottle* are misclassified some times as *read watch* and *answer phone*. However most of the activities are recognized very well.

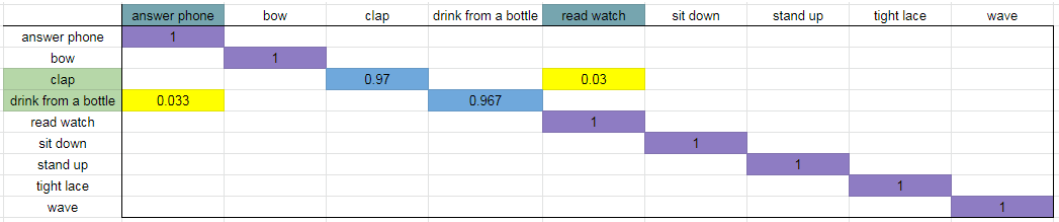


Figure 6: Confusion matrix of Florence 3D actions Dataset

4.4 Comparison with state of the art

We compared our proposed method with state-of-the-art method (described in Related work Section 2) on the two described benchmark datasets UTKinect-Action3D and Florence 3D Actions.

Table 1 and Table 2 show the results of comparison on, respectively, UTKinect-Action3D and Florence 3D benchmarks. It is easuly to see that the proposed approach gives the best results on all datasets. Specifically, it outperforms the state-of-the-art by 0.75% on UTKinect-Action dataset and by 3.1% on Florence3D-Action dataset.

Method	Accuracy
Cippitelli et al. [**?**]	95.1%
Liu et al. [**?**]	97.0%
Vemulapalli et al. [**?**]	97.1%
Luvizon et al. [**?**]	98.0%
Liu et al. [**?**]	98.5%
Proposed approach	**99.25%**

Table 1: Comparison with the state-of-the-art results on UTKinect-Action dataset

Method	Accuracy
Cippitelli et al. [**?**]	82.1%
Vemulapalli et al. [**?**]	90.9%
Luvizon et al. [**?**]	94.4%
Taha et al. [**?**]	96.2%
Proposed approach	**99.3%**

Table 2: Comparison with the state-of-the-art results on Florence3D-Action dataset

5 Conclusion

In this paper we have presented a new Residual Neural Network architecture using skeleton data. The method is composed by two majors steps: one for extracting spatio temporal information (in a unsupervised learning way); one for extract external and internal

relationships between representative actions generated by an unsupervised step. The main contribution of this work is to propose a new way to learn the representation of spatial relationship within the video, by a learning framework based on deep learning. The experimental results show that, despite its apparent simplicity, the performance of our method is very competitive overcoming the other state-of-the-art algorithms.

As future work, we propose different options to improve the model. First, it could be interesting to use new kinds of methods to learn features that represent the temporal information (however, the LSTM with our complex linear layer did not work well during our tests for the moment). Second, we propose to apply this architecture to solve other related problems of video activity recognition like, for example, Hand Gesture Recognition.