CNN-based real-time activity recognition system

Marta Carraro†, Davide Peron†

***Abstract*—Body sensors play an increasingly important role in our everyday life, they can be fundamental in terms of survival and health (think about pacemakers, bionic eye, bionic ear, etc.) or they offer us a new kinds of entertainment (such as fitness bands, sensors used in video games, etc.).**

In most cases, especially in sensors that make use of Inertial Measurement Units (IMUs), a motion recognition system is required. Depending on the application, these motions can be both little gestures or complex activity in which the entire body moves. A lot of effort has been put in find an efficient way to recognize in real time everyday activities and apply these solutions to a wide range of scenarios like first responders, assisted living rehabilitation, etc.

In this work CNN-based activity recognition systems are investigated and different architectures are tried. The goal learn a Convolutional Neural Network (CNN) capable of recognizing 11 different types of activities, showing that this type of Artificial Neural Networks (ANN) can give good predictions also with 1D data.

The resulting prediction accuracy in real-time application reveal that this architecture performs well in the learning phase but gives poor accuracy when tried on new data.

***Index Terms*—Activity recognition, Convolutional Neural Networks, Machine Learning, Real-time systems, Inertial sensors**

I. Introduction

Activity recognition systems are promising for the next-generation technologies, they will be used both for entertainment scenarios and to improve some aspects of the medical and survival sector. Existing solutions are usually implemented extracting hand-crafted features used as input for classifiers such as Support Vector Machines (SVM) [1]–[3]. This approach is consolidated and it leads to good results in terms of accuracy of the predictions, although hand-crafted features are data dependent and could not be generalized for different application domains. In the last few years a lot of effort has been put in implementing good Activity Recognition Systems (ARS) using non feature-dependent techniques to have a more general model and to reuse it in different scenarios.

This work improves the work made in [4] using a CNN-based technique to predict the proposed activities. In the original work a total of 19 features were extracted from the recorded signals, making the model strongly data-dependent. The aim of this work is to elaborate the dataset used in [4] and to learn a more general model to predict human activities in real time with a good accuracy presenting more complete results.

†Department of Information Engineering, University of Padova, email: {carrarom, perondav}@dei.unipd.it

Special thanks / acknowledgement go here.

Given their 2D nature, CNNs are usually applied to imaging field, such as the prediction of diseases classifying x-rays images or the recognition of object, people and animals. This work applies CNNs to 1D signals, proving that these kind of Neural Networks are not limited to 2D signals but they can have a wide range of applications. Moreover, since this approach does not require an ad-hoc dataset or a particular sensor to work, it can be applied (with some little changes) to any dataset with the same purpose.

Summing up what has been done in this work, the main contributions are reported in the following:

- a more general model CNN-based is used, to make this work reusable allowing other researchers to improve the architecture here implemented
- CNN is used in a non-typical 1D scenario, this proves the flexibility of this tool and the adaptability to a very wide range of applications
- more complete results regarding the accuracy reached in the test phase are presented, showing the behaviour of the model in situations where few data are available and these have an high variance.

The paper is structured as follows. In section II the state-of-art literature is presented, in section III are reported, at large, the main steps made by the implemented ARS to predict the activities. In section IV the signals collected in the dataset are described and the pre-processing algorithm to make them suitable for the learning framework is explained in detail. The learning framework is presented in V and the final results are commented in VI. Finally in VIII are reported the difficulty faced during the development of the system and conclusions are drawn.

II. Related Work

Activity recognition is a field evolving for more than twenty years, it started from very simple motion recognition and it gets more complex over the years. As just said, the classical approach to this problem is with a feature extraction techniques. In these solutions, ad-hoc features are extracted from the dataset, reducing the dimension of each signals from the recorded sample to a feature vector. In this way features can be classified using several classifiers to obtain an accurate prediction.

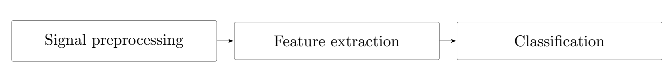

Fig. 1: Processing pipeline for feature extraction techniques

The main processing pipeline used from these solutions is reported in Fig. 1. The most used classifiers for feature extraction are Decision Trees, SVM, K-nearest neighbors and Naive Bayes although a lot of different classifiers can be used [5]. Features can be extracted either from a single sensor or from a large set of sensors, the first solution suffers an high computational complexity while the second is computational easier but certainly less portable and can create issues due to redundancy of the information. In [6] they collect data from 30 sensors positioned around the body and they classify features extracted from these data with a clustering algorithm. They chose to collect all the data in a central processing unit to perform a centralized classification. This approach suffers however an high cost and an high complexity to limit redundancy in the data. Varkey et al.'s research, uses only two accelerometer sensors placed at the wrist and ankle [7]. They collected data and they predict 6 different activity using an SVM as classifier, reaching an high accuracy.

The problem of this technique is the strong data dependence even if it gets a good accuracy.

To overcome this lack of the classical ARS, deep learning is used in several ways. In [8] an Hidden Markov Chain (HMM) is used on data collected by 8 sensors. They firstly compute feature extraction on the data collected from each sensor in a distributed manner, then feature vectors are given in input to the HMM that classify the feature vectors in the activities labels. In [9] the authors perform a matrix factorization for dimensionality reduction and deep learning algorithm to automatically learn suitable features. In this work, they reduce the dimension of the dataset using matrix factorization and they elaborate the results in a Neural Network. The output of the Neural Network (NN) is a feature vector. Finally the activity is predicted classifying the feature vector with a Softmax classifier. The accuracy is quite good since they avoid hand-crafted features, but they still rely on features extracted from data.

The key to implement an adaptable and reliable algorithm to classify human activities is the use of deep learning without extracting features at all. A comparison between three different deep learning algorithms is made in [10]. They collected data from one single sensor and they predicted activities using Decision Trees (DT), ANN and Random Forest (RF). They stated that RF performs better than the other two, with an accuracy between 75% and 90% depending on the activity.

Milenkoski et al. use instead Long Short-Term Memory (LSTM) networks, a specific type of Recurrent Neural Networks (RNN), to perform activity prediction in real time on smartphones. They learn the model using a previously collected dataset to subsequently apply it to new raw data recorded from a smartphone. The accuracy reached is variable between 50% for the most difficult activities to recognize and 100% for the easiest. The overall accuracy is 88% for the data acquired and pre-processed in laboratory and 82% for the real time prediction [11].

An interesting solution is the application of CNN in the activity recognition field. In [12] a CNN is used to predict activities using data recorded by a tri-axis accelerometer sensor. They tried to predict very specific activities such as *Fetch cup from desk* or *Pour milk into cup* using a CNN made by 3 convolutional layers, one max-pooling layer and a fully connected layer, using a Softmax function to the output of the network. This application has a recognition rate of 99.8% although is not thought for real time prediction.

The goal of the next sections of this work is to combine a CNN architecture with a real time prediction model, to prove that this approach gives results comparable or better than the ones showed in [13].

III. Processing Pipeline

The work can be divided in 3 parts: the dataset creation, the neural network creation and the real time prediction. In order to build and ARS, a proper dataset was firstly created. All the signals have been divided in several overlapping windows of the same dimension, each window corresponds to a specific activity. The set of windows and the correspondent activity labels created are then divided in training set and test set to learn and assess the prediction model. The training set is used to fed a CNN made by 1D convolutional layers and fully connected layers while the test set is used to assess the accuracy of the learned model and to avoid overfitting problems.

When the NN is trained and tested, a new dataset is used in order to verify the effective robustness and generalization of the model in a real time application. The main steps of the processing pipeline are reported in Fig. 2.

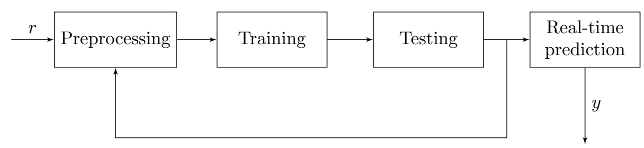

Fig. 2: Scheme of processing pipeline, r is the raw signal given in input to the preprocessing part, y is the label predicted by the prediction algorithm

IV. Signals and Dataset

A. Measurement setup

The signals the authors have worked on were provided by German Aerospace Center (DLR) official website [14]. The work is based on the data collected in the datasets *ARS_DLR_Data_Set_V2.mat* and *ARS_DLR_Benchmark_Data_Set.mat*. Both of them are made up of signals recovered by a Micro Electro-Mechanical System (MEMS) based IMU (an Xsens MTx-28A53G25) composed by a accelerometer, a gyroscope and a magnetometer. These measurement systems provide information about the inertial acceleration, the angular velocity and the magnetic field direction. Data are collected recording signals from 14 people while they perform some ordinary motion activities like *standing*, *sitting*, *running*, *jumping*, *lying*. The motion sensor is positioned over the pelvic region of each subject (Fig. 3).

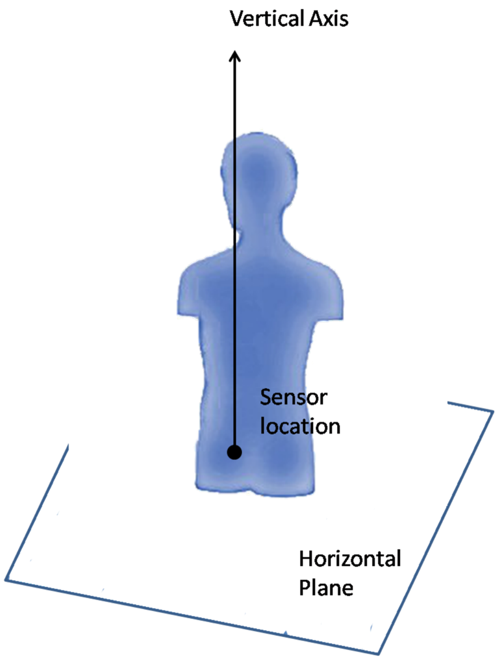

Fig. 3: Sensor position and the representation of the body frame

Although the datasets are used in different ways and to comply two different tasks of the ARS, their structure is exactly the same since their are collected under the same conditions. Both datasets are divided in activity sessions, 34 in *ARS_DLR_Data_Set_V2.mat* and 3 in *ARS_DLR_Benchmark_Data_Set.mat*, each session contains the following field:

- a matrix of 10 columns in which the first column represents the time domain and the other ones represents the IMU records over the three sensor axis;
- a rotation matrix that has the same dimension of the first field and allows to represent the measurement values in the global frame;
- a vector that contains the activity labels performed during the session (see Tab. 1);
- a vector that indicates when each activity starts and ends.

Label	Index	Description
'RUNNING'	0	running
'WALKING'	1	walking
'JUMPING'	2	jumping
'STNDING'	3	standing
'SITTING'	4	sitting
'XLYINGX'	5	lying
'FALLING'	6	falling
'TRANSUP'	7	getting up i.e.: from sitting to standing
'TRANSDW'	8	going down i.e.: from standing to sitting
'TRNSACC'	9	accelerating
'TRNSDCC'	10	deccelerating

TABLE 1: Activities took into consideration with the associated labels

B. Signal pre-processing

The first pre-processing applied to the dataset consists in representing the signals according to the global frame using the rotation matrix. The dataset considered already contains pre-processed data, sampled at T = 0.01s.

In Fig. 4, Fig. 5 and Fig. 6 is showed one of Susanna activity sessions. These figures represent the magnitude of accelerometer, gyroscope and magnetometer over the three global axis. Magnitude is plotted instead of the measurements for each axis given the visual meaningfulness of the magnitude, although magnitude is not used in the computation. In particular in Fig. 4 can be noticed the shift of the acceleration's mean around 9.8 m/s^2, value coherent with the gravitational constant value g. It also emerges in each of these figures how the transitory activities from standing to sitting and vice versa can be seen due to the drastic change of the signals.

Another sort of pre-processing has been made in order to fix the activity indexing of some recordings. It frequently happened to find that, considering two adjacent motion activities, the end of the first activity and the beginning of the second one were not temporarily neighboring. It happened also to find two activities temporarily overlapping: the end of the previous activity was indexed after the beginning of the second one. The authors resolved the problem removing the non indexed data and the data whose label was uncertain so as to not train the NN with wrongly labeled data.

Each session is finally represented as a long and straight matrix with nine columns (three for each measurement system) and a number of rows equals to the session duration.

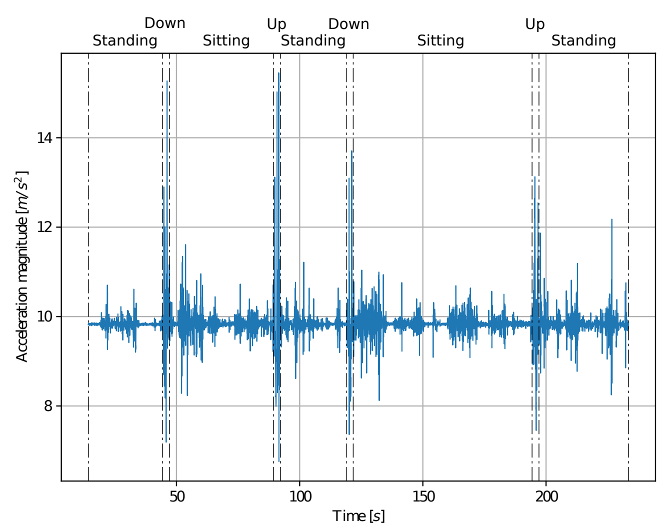

Fig. 4: Acceleration norm

C. Dataset creation

ARS_DLR_Data_Set_V2.mat Since the CNN needs a fixed input, each session matrix has been divided in patterns correspondent to a specific activity. Every activity, then, has been divided in overlapping windows with stride equals to 3 and length equal to 27 samples, that is the

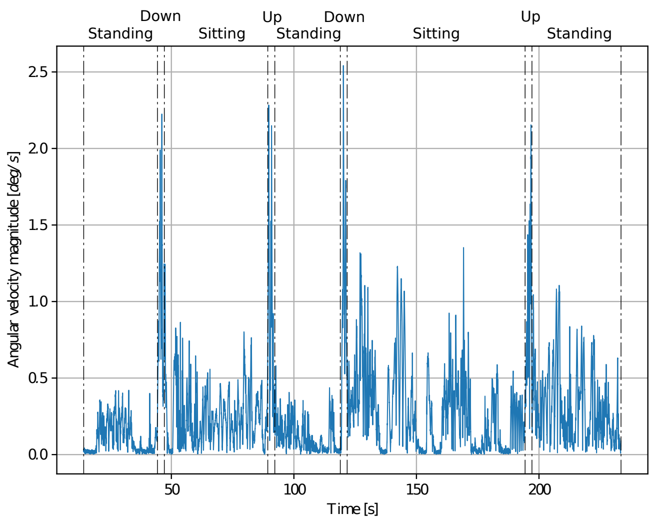

Fig. 5: Angular velocity norm

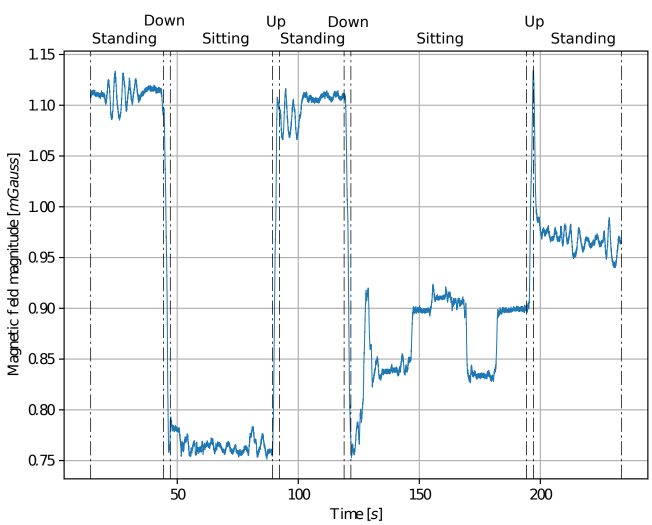

Fig. 6: Magnetic field norm

shortest activity length in the whole dataset. The obtained dataset is made by several windows associated to a specific activity label, no transitional windows from an activity to another has been taken. The associated labels are then one hot encoded and, to ensure the independence between an input and the subsequent one in the CNN, the dataset is finally shuffled.

ARS_DLR_Benchmark_Data_Set.mat It is composed by 3 activity sessions and it was used for real time prediction purposes. The signals are segmented in 27 length patterns with stride 5. The algorithm takes into account also transitional windows in order to simulate a realistic real time prediction trial even if they are not considered in the computation of the performance metrics.

V. CNN ARCHITECTURE

The first dataset is composed by 322502 patterns with shape (27,9) and it includes both the training set and the test

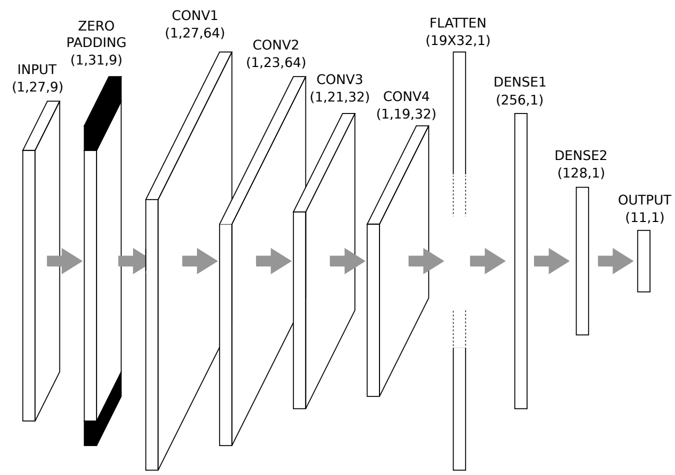

Fig. 7: CNN architecture for a single input. Only layers that change the size of the input are showed.

set. These data are divided in training set and test set using respectively the 80% and the 20% of the whole patterns.

The CNN architecture is schematically presented in Fig. 7 and in this section is explained in details using the showed labeling of the layers.

Even if the CNN works with matrices, has been decided to treat the single input pattern as a vertical 27 size vector with 9 channels. Zero Padding was firstly applied at the first and last two rows of the input in order to take in account also the borders in the convolutional layers.

Then four 1D convolutional layers are applied: the changing dimension of a single pattern is shown in Fig. 7, the row size is calculated as in Eq. (1) where r_l stands for the row size of the output of the l^{th} layer, r_{l-1} is the row size of the output of the $l-1^{th}$ layer, k is the kernel size and s is the stride used in the l^{th} layer. In Tab. 2 the row size for each layer is reported.

$$r_l = \frac{r_{l-1} - k}{s} + 1 \quad (1)$$

LAYER	r_{l-1}	k	s	OUTPUT
CONV1	31	5	1	27
CONV2	27	5	1	23
CONV3	27	3	1	21
CONV4	21	3	1	19

TABLE 2: Filters dimension along the pattern columns.

After each convolutional layer a Batch Normalization and a Rectified Linear Unit (ReLU) activation Function is applied.

After this feature learning block, follows a classification part made of three fully connected layers: to allow this passage a flattening of Conv4 output filters is necessary (see Flatten layer in Fig. 7). Then follows three fully connected layers of size respecively 256, 128 and 11 with ReLU activation function except the last one where Softmax function has been used. The last one is the output, each of its 11 elements contains the prediction probability of a label. Softmax functions needs to find the label that has the higher probability. Since the activities to classify using the NN are more than two, the loss function used was *categorical cross-entropy*.

In order to get a better generalization of the model, Dropout has been added. In particular the authors decided to adopt a Dropout of p = 0.15 after Conv1, Conv2 and Conv3, p = 0.25 before Flatten and p = 0.5 after Dense1 and Dense2. This choice was made according to the most commonly used CNN configuration proposed by Hinton et al.(p = 0.5 on each fully connected layer) and a more singular configuration proposed by Sungheon et al.(p = 0.1-0.2 between the convolutional layers) [**?**] [**?**]. The *training set* was trained for 10 epochs with a batch size of 128 and the optimizer used was *adam*.

VI. Results

The ARS previously described has been tried on a PC with an Intel Core i7-7700HQ microprocessor at 3.8 GHz, 8 GB of RAM, and an NVIDIA Ge-Force GTX 1050 as Graphic Card. The programming language used was Python with Keras and TensorFlow as learning framework. The most computational demanding operations was performed on the GPU using CUDA drivers. The learning part has been tried also performing all the computations on the CPU, the difference in time is dramatically: using the GPU each epoch takes about 26 seconds while using only CPU takes more than 2 minutes.

In the learning phase, the training accuracy reaches 93.75% while in the test dataset, activities are recognized the 94.6% of the times.

Once the model has been learned, the CNN is used to predict activities in real time. To assess the performance of this task, a set of predictions has been made on a raw signal recorded by the same sensor. The overall recognition rate of the real time prediction is 75.88%, precision and recall are showed in Fig. 8 and Fig. 9.

The accuracy in the prediction phase is quite low, since the dataset used was too small and had a great variance within the data. Moreover in that dataset some labels were missing.

As can be seen the accuracy is very good for longer or easy-to-recognize activities like *Running*, *Walking*, *Standing* and *Lying*, while is lower for shorter activities or activities that suffer of a big variance such as *Falling* and *Setting*.

In [13] precision and recall are plotted only for the main activities, they don't take into account the transition activities (*Up*, *Down*, *Ascending*, *Descending*). These gives a low accuracy since there are less sample in the dataset and since they can have an high variance.

VII. Summary of the project

This work has been useful to approach new machine learning tools both from a theoretical and a practical point of view, understanding the problems that one may encounter. In specific, we saw how ANN and CNN work and how can be implemented to solve a given task, we learned to use deep learning framework like Keras and TensorFlow together with advanced version control software. Moreover we improved our knowledge about Python and how to optimize it to work with big amount of data.

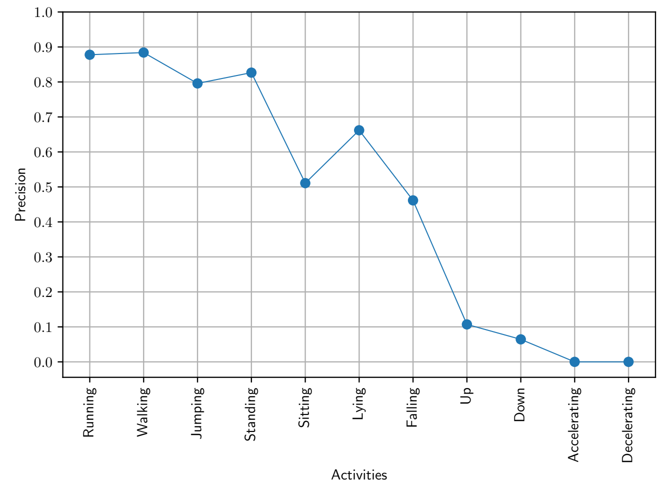

Fig. 8: Precision calculated for each activity to recognize

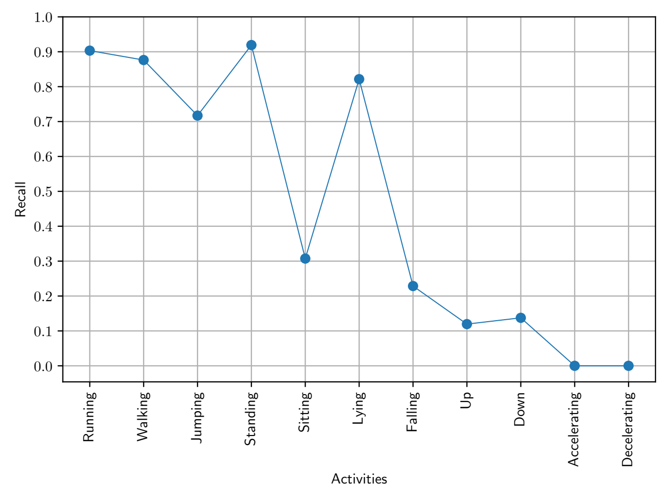

Fig. 9: Recall calculated for each activity to recognize

During the project we encountered several problems, we tried to solve them and we found a solution or a reasonable answer. Firstly, we noticed that the number of sample for each label is highly unbalanced, the solution we thought were to use data augmentation and to weight different each label in the CNN. We tried to implement the first solution creating new samples adding noise to the original signals. We tried with white noise with small variance but the result was a low accuracy. Maybe the solution is to estimate a good model for the noise and adding it instead of a standard white noise. In the second case we tried to weight different the classes but we can't adapt our architecture to the requirements of the apposite keras' function. Our solution has been to use a smaller stride for these activities to have more data.

In the learning phase the biggest problem was the time needed to train the network. We solved this problem installing CUDA drivers in Windows 10 and using the Graphic Card to learn the model.

In the last part, the main problem was how to elaborate the windows of the benchmark signals containing the transitions from an activity to another. Our first idea was to add to the training set some window correspondent to the transitions, and assign to them the label *TRANSIT* (i.e. *activity not recognized*), but the only result was a lower accuracy, so we decided to skip those parts when sliding other the benchmark signals.

Finally, we notice that in the benchmark dataset misses the label *TRANSDCC*, so both precision and recall has been put to zero even if these label were not present at all in the prediction phase.

VIII. Conclusions and future works

In this work, a novel approach to activity recognition has been proposed, using CNN as deep learning tool to predict automatically activities. The authors started from a dataset used originally to make activity recognition using a feature selection approach, they applied some preprocessing and they used the preprocessed data to learn the Neural Network. Then, the learned model has been applied to raw signals to show how this architecture is feasible for a real time system. Remarkable results has been obtained, an accuracy of over 94% in testing phase and an overall recognition rate of 75% in the real time prediction. Given its automatic nature, the described ARS can be used to learn other datasets, adjusting the dimension of the input and the labels to predict. In presence of a bigger amount of data, the CNN can be probably learned with more epochs without overfitting the data.

The solution proposed prove how CNN can be used in activity recognition instead of techniques based on hand-crafted features.

The ARS described can be extended to reach an higher accuracy. An improvement to the overall efficiency of the network can be to assign higher weights to the activities less frequent in the dataset, to have a more robust prediction also for those activities increasing the accuracy of the CNN. Another solution to this problem can be using a dataset with a balanced number of occurrences per label, in this case weighting differently the activities can be unnecessary. The natural continuation of this work will be to find a way to take into account also the transitions between different activities to make the prediction more robust.

References

[1] V. Elvira, A. Nazábal-Renteria, and A. Artés-Rodríguez, "A novel feature extraction technique for human activity recognition," in *2014 IEEE Workshop on Statistical Signal Processing (SSP)*, pp. 177–180, June 2014.

[2] W. Hamäläinen, M. Järvinen, P. Martiskainen, and J. Mononen, "Jerk-based feature extraction for robust activity recognition from acceleration data," in *2011 11th International Conference on Intelligent Systems Design and Applications*, pp. 831–836, Nov 2011.

[3] Z. A. Khan and W. Sohn, "Feature extraction and dimensions reduction using r transform and principal component analysis for abnormal human activity recognition," in *2010 6th International Conference on Advanced Information Management and Service (IMS)*, pp. 253–258, Nov 2010.

[4] K. Frank, M. J. V. Nadales, P. Robertson, and M. Angermann, "Reliable real-time recognition of motion related human activities using mems inertial sensors," in *ION GNSS 2010*, September 2010.

[5] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, "Activity recognition from accelerometer data," in *2005 17th Conference on Innovative Applications of Artificial Intelligence*, 2005.

[6] K. V. Laerhoven, A. Schmidt, and H. W. Gellersen, "Multi-sensor context aware clothing," in *Proceedings. Sixth International Symposium on Wearable Computers,*, pp. 49–56, 2002.

[7] J. P. Varkey, D. Pompili, and T. A. Walls, "Human motion recognition using a wireless sensor-based wearable system," *Personal and Ubiquitous Computing*, vol. 16, pp. 897–910, oct 2012.

[8] E. Guenterberg, H. Ghasemzadeh, V. Loseu, and R. Jafari, "Distributed Continuous Action Recognition Using a Hidden Markov Model in Body Sensor Networks," in *Distributed Computing in Sensor Systems* (B. Krishnamachari, S. Suri, W. Heinzelman, and U. Mitra, eds.), (Berlin, Heidelberg), pp. 145–158, Springer Berlin Heidelberg, 2009.

[9] B. Chikhaoui and F. Gouineau, "Towards automatic feature extraction for activity recognition from wearable sensors: A deep learning approach," in *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 693–702, Nov 2017.

[10] L. Xu, W. Yang, Y. Cao, and Q. Li, "Human activity recognition based on random forests," in *2017 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, pp. 548–553, July 2017.

[11] M. Milenkoski, K. Trivodaliev, S. Kalajdziski, M. Jovanov, and B. R. Stojkoska, "Real time human activity recognition on smartphones using lstm networks," in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pp. 1126–1131, May 2018.

[12] M. Panwar, S. R. Dyuthi, K. C. Prakash, D. Biswas, A. Acharyya, K. Maharatna, A. Gautam, and G. R. Naik, "Cnn based approach for activity recognition using a wrist-worn accelerometer," in *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 2438–2441, July 2017.

[13] K. Frank, M. J. V. Nadales, P. Robertson, and M. Angermann, "Reliable Real-Time Recognition of Motion Related Human Activities Using MEMS Inertial Sensors," *German Aerospace Center (DLR), Institute of Communications and Navigation*, 2010.

[14] "Deutsches zentrum für luft- und raumfahrt (DLR) - institut für kommunikation und navigation, nachrichtensysteme," 2010. https://www.dlr.de/kn/desktopdefault.aspx/tabid-8500/14564˙read-36508.

[15] M. Gadaleta and M. Rossi, "IDNet: Smartphone-based gait recognition with convolutional neural networks," *Pattern Recognition - Elsevier Ltd.*, vol. 74, pp. 25–37, Feb. 2018.