A Novel Fitness Tracker Using Edge Machine Learning

Massimo Merenda
Department of Information Engineering, Infrastructures and Sustainable Energy (DIIES) and HWA s.r.l.-Spin Off of the University Mediterranea of Reggio Calabria
Reggio Calabria, Italy
massimo.merenda@unirc.it

Miriam Astrologo
Department of Information Engineering, Infrastructures and Sustainable Energy (DIIES) of the University Mediterranea of Reggio Calabria
Reggio Calabria, Italy
strmrm97e43f112a@studenti.unirc.it

Damiano Laurendi
Department of Information Engineering, Infrastructures and Sustainable Energy (DIIES) and HWA s.r.l.-Spin Off of the University Mediterranea of Reggio Calabria
Reggio Calabria, Italy
lrndmn96m24h224y@studenti.unirc.it

Vincenzo Romeo
Department of Information Engineering, Infrastructures and Sustainable Energy (DIIES) of the University Mediterranea of Reggio Calabria
Reggio Calabria, Italy
rmovcn96m14h224s@studenti.unirc.it

Francesco Giuseppe Della Corte
Department of Information Engineering, Infrastructures and Sustainable Energy (DIIES) and HWA s.r.l.-Spin Off of the University Mediterranea of Reggio Calabria
Reggio Calabria, Italy
francesco.dellacorte@unirc.it

***Abstract*—Several characteristics of the human body turn into postural behavior, recognizable also during sport activities. The presence of differences between body types could lead to different behavior of wearable and fitness-devote products. A new wearable based on machine learning techniques for the exercise detection and repetitions count is described in this work. A proper dataset has been obtained in order to offline train the network. Eventually, the machine learning algorithm has been implemented inside an edge device for real-time test e verification.**

Keyword--fitness tracker, machine learning, edge machine learning, embedded system

I. Introduction

Human Activity Recognition (HAR) is based on motion sensors analysis to deduce activities/actions performed by people. In the last period, this research field is constantly growing; there are many applications that could represent a significant turning point for wearable devices [1]–[3]. Nowadays many users use fitness-bands while training. Fitness-band features are able to keep under control sports performance like distance, training time or burned calories. There are some solutions to track the training routine, recognizing exercises and counting number of repetitions. Some of these are based on computer vision [4], [5] while others make use of smartphones equipped with inertial sensors [6]–[8]. In this work, a new solution implemented on a microcontroller (MCU)-based wearable device is discussed, exploiting the feasibility of edge machine learning in an IoT scenario [9].

This work proposes a Machine Learning (ML) HAR application for fitness environment exploiting Convolutional Neural Networks (CNN) [10]; starting from data acquisition, through edge ML system implementation on low power MCUs and with reduced form factor. The solution described below provides a method for automatic detection of body movement during exercise execution and reps counting, acting as a gym-fitness tracker. In section II the system overview will be depicted. In section III the hardware will be described in detail, while in the section IV the dataset acquisition will be illustrated. The neural network (NN) is described in section V. Finally, section VI presents discussion and results.

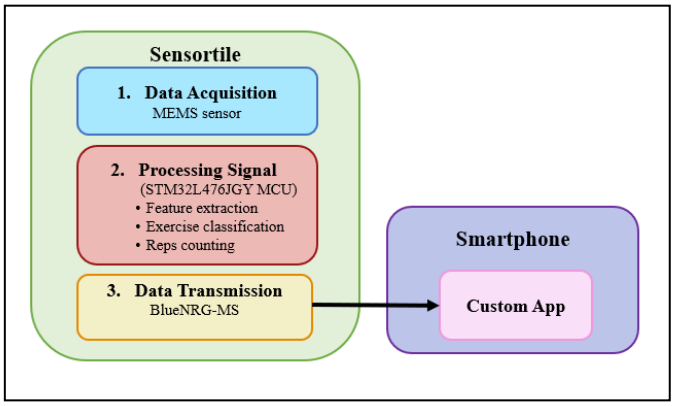

Fig. 1. Block Diagram of the system used in this work.

II. System Overview

In this work, the system provided aims to self-classify the exercises performed by users with a Deep Learning (DL) approach, using the data collected through a wearable device. The purpose is also to count the number of repetitions and exercises performed.

The block diagram, shown in Fig. 1, can be logically split in three parts:

- The first is responsible for data collection, i.e. the acquisition of raw data through the wearable device's sensors during the execution of an activity.
- The unprocessed values are sent as input to the MCU (block 2), in which the NN is running. It's

©2020 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works

DOI: 10.1109/MELECON48756.2020.9140602

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

responsible for the final recognition of the exercises and the reps counting.

- Lastly, the results provided by the MCU are sent via Bluetooth Low Energy (BLE) protocol (block 3) on a smartphone acting as an application interface layer.

III. HARDWARE

In order to capture data from all directions, a 3-axes accelerometer and a 3-axes gyroscope are used to create the dataset for both training and testing phase of the application. The new paradigm of edge machine learning requires a balanced use of Random Access Memory (RAM) and Flash of the MCU, due to the limited resources of the hardware. To this end, attention shall be devoted to the size of the NN, in terms of bias, weights and data structures. The SensorTile module by ST Microelectronics, shown in Fig. 2, is used as prototype both in dataset creation phase and in testing phase. This hardware module embeds the ultra-low-power STM32L476 microcontroller (80MHz clock, 1Mbyte Flash and 128Kbyte SRAM), a BLE module (with a peak current of 8.2mA at 0dBm) used for data transmission and the used sensors, namely the system-in-package LSM6DSM with accelerometer and gyroscope. Other sensors are assembled on the same board such as environment sensors and microphones, however they are not used within this application. Finally, the SensorTile module is powered by a 100 mAh Li-Ion battery. Ultra-low-power sleep modes for both BLE module and microcontroller allows low average current consumption, resulting in longer battery duration.

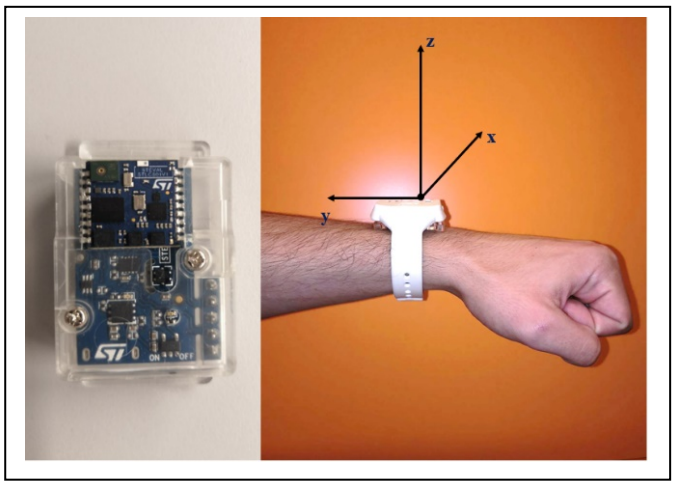

Fig. 2. SensorTile Hardware on the left. Prototype of the fitness tracker band and axes direction representation on the right.

IV. DATASET

A new dataset was created for this work using the hardware explained above. A group of 15 volunteers aged between 22 and 25 was selected for obtaining the activity recordings. Each person performed various series of simple exercises wearing the device around the wrist. Three exercises were chosen for this application (*Squat, Curl, Push-Up*), with the addition of another class named NAE (*Not An Exercise*), useful to detect rest and wrong execution of exercise. With NAE class is also possible to distinguish between an exercise and rest as well between series or repetitions. Labels are, then:

- Squat: quadriceps and glutes exercise (*label 1*)
- Curl: biceps exercise (*label 2*)
- Push-Up: pectoral and triceps exercise (*label 3*)
- NAE: not an exercise (*label 4*)

Accelerometer and gyroscopic values were collected with 20 Hz frequency. Subsequently, collected data were analyzed and processed with a MATLAB custom script to correctly label each dataset sample. An example of accelerometer data, related to curl exercise, is plotted in Fig. 5; as can be seen, repetitions are clearly discernable because inertial data vary during the execution of the exercise. The NN shall classify the exercises and, in order to count also repetitions, it was trained with single repetition. With respect to Fig. 5, the repetition duration is defined approximately as the time between $T_{i,n}$ (instant in which the movement starts) and $T_{f,n}$ (the instant in which the movements end). Single repetition duration is variable depending on the exercise type. For this reason, the correct definition of the acquisition window is critical. Short windows may enhance performances of the NN but may lead to data losses. On the other side, long windows may have negative impact during classifications and counting operations. A custom MATLAB script was used to collect repetition windows (yellow box in figure 5) of the same length to be able to label each window as a specific exercise repetition. So, each entry of the dataset is a fixed-length window of 68 sample for a total of 3.4 seconds (68/20Hz=3.4s) in which each sample is composed by 3 axes data acquired by accelerometer and 3 axes data acquired by gyroscope (6 values for each sample). Data between two successive windows are labeled as NAE together with other data acquired while the user is resting. Repetitions of Squat, for example, can be counted thank to a series of alternated NAE and Squat recognition of the NN. A total of 700 repetitions for the 3 exercise were collected of which 80% was used to train the NN and 20% for test.

V. NEURAL NETWORK

Among various architectures, CNN was chosen for its characteristics. Initially, a Recurrent Neural Network (RNN) was developed with the same dataset, showing not satisfactory performances with accuracy around 65%. Like Multi-Layer Perceptron (MLP), the CNN is a network where the main operation is convolution [11]. They are often used in Machine Learning applications with imaging recognition and clustering [12]. In the current state of the art they are also applied on HAR problems, e.g. running-walking or sitting-standing recognition [13]. Thanks to convolution, the CNN can reduce the input pattern in any layer, trying to keep the information content intact, with the use of special filters. After sliding the filter over all the locations, the output layer called Activation Map is passed through activation function and then for a pooling layer to reduce problem size. In the last, the actual classification happens; the output is composed by n positive value whom sum is equal to 1, with n the number of labels.

For this work, the CNN was chosen for three main reasons. First one, it doesn't require data pre-processing to extract features. Considering the edge machine learning applications, this is helpful because the introduction of pre-processing operations on the MCU can increase the complexity and the needed memory. Nevertheless, the reduction of the computation burden of the MCU has a positive impact on the power consumption, as the system is battery powered. The second benefit is the weight sharing, that bring to the network

less links between layer and a memory saving. Lastly, the CNN can exploit the concept of invariance, in fact, for the pattern discrimination purpose, response of the network remains almost the same despite simple transformations of input patterns (like small temporal or spatial alterations).

This last benefit makes the application generalizable for any user. In one repetition's movement of an exercise there are 68 samples (each with 6 values). So even if they are slightly different from data with which the network has been trained, it is still possible to classify the exercise correctly.

In this project a CNN-1D was used. It has a 3D array input of [68, 6, 700] size and an array output containing 4 values corresponding to the probability related for each class depending on the specific input pattern. This CNN has two hidden layers, with a kernel size equal to 3. Activation functions are *relu* in the first layer and *sigmoid* in the second. Filters have both the same dimension. *MaxPooling* was used to further reduce dimension and redundancy without altering information content. Training (80% of dataset entries) and test remaining 20%) were performed 10 times with a number of epochs equal to 200 and the accuracy was calculated evaluating the cross-entropy; when this has reached a value greater than 95%, the testing phase started, both on desktop and on target (MCU).

VI. TESTING AND RESULTS

After generating model, the NN was tested on Keras with data not present in the input space. For all input patterns (about 50), the CNN gave the right output every time. After this phase, the challenge was to convert this CNN in C-code compatible with the MCU firmware. For this reason, a free software released by STMicroelectronics was used, namely STM32CubeMX, and his tool STM32CubeAI. This tool can translate the NN from .h5 saved format to C library, automatically generating the APIs to use in the firmware. In this tool, there are three different bundle about code generation: Validation, System Performance and Application Template. The first and the second have been used during the validation and the testing of device using the NN, the last was selected to write the firmware about this application. This tool allows a further compression of the NN to reduce the occupied memory. Subsequently, the CNN was desktop tested with STM32CubeAI and then it was tested on target, with NN in run-time on the MCU. In both tests, random and unknown data input are used, for a total of 70 input. In each of these, the CNN gives the right output and the confusion matrix is consistent with the obtained results, having only main diagonal elements different from 0 as can be seen in Fig. 5.

In this part of the work, application performances have been analyzed, such as occupied memory from NN (both RAM and Flash), or inference time needed to produce an output from the network (which was estimated to be around 80ms). The Table 1 shows the values of RAM and Flash memory, represented for any configuration of compression. In this work the selected compression is 8x to obtain more memory for future developments, as the compression doesn't reduce the NN performances.

TABLE I. COMPRESSION AND MEMORY

Compression	*RAM*	*FLASH*
-	16,25 Kbytes	416,34 Kbytes
4x	16,25 Kbytes	117,47 Kbytes
8x	16,25 Kbytes	66,53 Kbytes

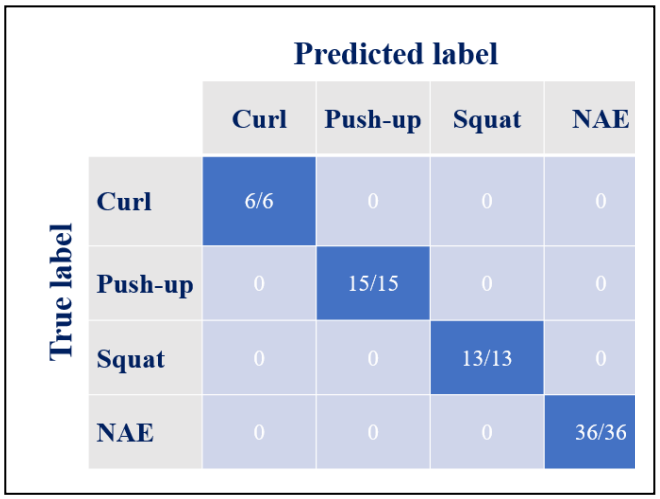

Fig. 3. Confusion matrix in both testing activities on desktop and on target.

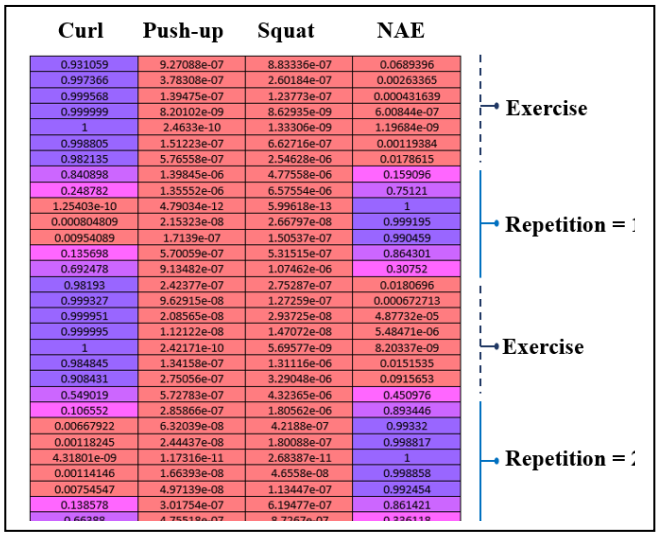

Fig. 4. Visual representation of the counting reps algorithm. When the NN recognises an exercise followed by a NAE, the rep count for that exercise is increased by one.

After checking that MCU's memory was enough, the application firmware was conceived. For exercise recognition, overlapping windows of raw sensor data are provided to a CNN that predicts the type of exercise for each window. For repetition counting, MCU analyzes outputs of CNN. During the execution of the exercise, an example of CNN output is showed in Fig. 4. The NN recognizes and classifies "NAE" when the user starts/finishes a single repetition. In this way, MCU processes this information and identifies how many times that condition has occurred. After the execution of the ML algorithm, the output is transmitted via BLE channel on the smartphone application, under development. Further details will be provided during the conference.

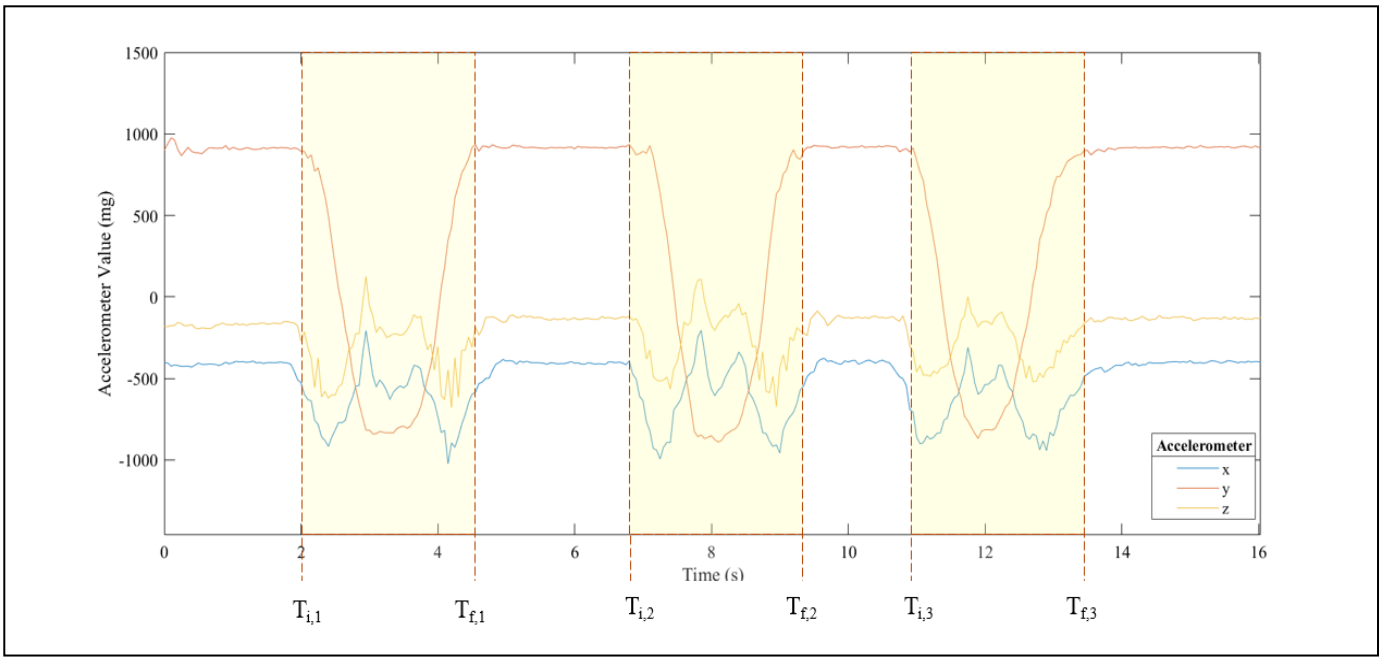

Fig. 5. Accelerometer data captured while executing Curl Exercise. Highlighted areas refers to the execution of a single repetitions.

REFERENCES

[1] Ó. D. Lara and M. A. Labrador, 'A survey on human activity recognition using wearable sensors', *IEEE Commun. Surv. Tutorials*, vol. 15, no. 3, pp. 1192–1209, 2013.

[2] M. Merenda, D. Laurendi, D. Iero, D. M. D'Addona, and F. G. Della Corte, 'Wireless Sensors for Intraoral Force Monitoring', in *Lecture Notes in Electrical Engineering*, 2020, vol. 627, pp. 267–273.

[3] M. Merenda, D. Iero, and F. G. D. Corte, 'Cmos rf transmitters with on-chip antenna for passive RFID and iot nodes', *Electron.*, vol. 8, no. 12, 2019.

[4] R. Khurana, K. Ahuja, Z. Yu, J. Mankoff, C. Harrison, and M. Goel, 'GymCam: Detecting, Recognizing and Tracking Simultaneous Exercises in Unconstrained Scenes', in *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2018, vol. 2, no. 4, pp. 1–17.

[5] O. Levy and L. Wolf, 'Live repetition counting', in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, vol. 2015 International Conference on Computer Vision, ICCV 2015, pp. 3020–3028.

[6] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, 'Activity recognition using cell phone accelerometers', in *ACM SIGKDD Explorations Newsletter*, 2011, vol. 12, no. 2, pp. 74–82.

[7] D. Morris, T. S. Saponas, A. Guillory, and I. Kelner, 'RecoFit: Using a wearable sensor to find, recognize, and count repetitive exercises', in *Conference on Human Factors in Computing Systems - Proceedings*, 2014, pp. 3225–3234.

[8] C. Seeger, A. Buchmann, and K. Van Laerhoven, 'MyHealthAssistant: A Phone-based body sensor network that captures the wearer's exercises throughout the day', in *BODYNETS 2011 - 6th International ICST Conference on Body Area Networks*, 2012, pp. 1–7.

[9] M. Merenda, C. Porcaro, and D. Iero, 'Edge Machine Learning for AI-enabled IoT devices: a review', *Sensors (Switzerland)*, 2020.

[10] P. Y. Simard, D. Steinkraus, and J. C. Platt, 'Best practices for convolutional neural networks applied to visual document analysis', *Proc. Int. Conf. Doc. Anal. Recognition, ICDAR*, vol. 2003-January, pp. 958–963, 2003.

[11] M. Xin and Y. Wang, 'Research on image classification model based on deep convolution neural network', *Eurasip J. Image Video Process.*, vol. 2019, no. 1, 2019.

[12] M. Merenda, F. G. Praticò, R. Fedele, R. Carotenuto, and F. G. D. Corte, 'A real-time decision platform for the management of structures and infrastructures', *Electronics (Switzerland)*, vol. 8, no. 10. 2019.

[13] M. Panwar *et al.*, 'CNN based approach for activity recognition using a wrist-worn accelerometer', *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS*, pp. 2438–2441, 2017.