sensors

MDPI

Article

Sign Language Recognition Using Wearable Electronics: Implementing k-Nearest Neighbors with Dynamic Time Warping and Convolutional Neural Network Algorithms

Giovanni Saggio [1], Pietro Cavallo [2], Mariachiara Ricci [1], Vito Errico [1,*], Jonathan Zea [3] and Marco E. Benalcázar [3]

[1] Department of Electronic Engineering, University of Rome "Tor Vergata", Via Politecnico 1, 00133 Rome, Italy; saggio@uniroma2.it (G.S.); maryclair_91@hotmail.it (M.R.)
[2] Data Analysis Group, MathWorks, Matrix House, Cambridge Business Park, Cambridge CB4 0HH, UK; p.cavallo85@gmail.com
[3] Department of Informatics and Computer Science, Escuela Politécnica Nacional, Quito 170517, Ecuador; marco.benalcazar@epn.edu.ec (M.E.B.); z_tjalezea@yahoo.com (J.Z.)
* Correspondence: vito.errico@uniroma2.it

Received: 27 April 2020; Accepted: 8 July 2020; Published: 11 July 2020

Abstract: We propose a sign language recognition system based on wearable electronics and two different classification algorithms. The wearable electronics were made of a sensory glove and inertial measurement units to gather fingers, wrist, and arm/forearm movements. The classifiers were k-Nearest Neighbors with Dynamic Time Warping (that is a non-parametric method) and Convolutional Neural Networks (that is a parametric method). Ten sign-words were considered from the Italian Sign Language: cose, grazie, maestra, together with words with international meaning such as google, internet, jogging, pizza, television, twitter, and ciao. The signs were repeated one-hundred times each by seven people, five male and two females, aged 29–54 y ± 10.34 (SD). The adopted classifiers performed with an accuracy of 96.6% ± 3.4 (SD) for the k-Nearest Neighbors plus the Dynamic Time Warping and of 98.0% ± 2.0 (SD) for the Convolutional Neural Networks. Our system was made of wearable electronics among the most complete ones, and the classifiers top performed in comparison with other relevant works reported in the literature.

Keywords: wearable electronics; gesture recognition; sign language; sensory glove; IMU; classifiers

1. Introduction

Generally speaking, standardized signs or gestures improve communication [1], as it occurs in army and aircraft security scenarios [2], or improve interaction, as it occurs in human–machine systems [3–5] and in tele-control [6], or improve efficiency, as it occurs in surgery [7], just to mention a few. Specifically, signs that are structured with syntax, semantics, grammar, pragmatics morphology, and phonology become "sign languages" [8], which allow us the expression of thoughts and feelings, in the same way as a "natural language" can do.

Sign languages spread worldwide. This occurs especially because of a large amount of hearing disabilities, suffered by more than 466 million people, as pointed out by the World Health Organization (WHO, 2018) [9]. Sign languages are used by muted, deaf, aurally challenged, and hear impaired people, along as with their relatives and educators. Unfortunately, most of the people are not used to, or do not know, sign languages so their communication with hearing and speaking impaired people is quite difficult. Fortunately, more and more technologies focus on solving this challenging issue, measuring signs and assigning to each gesture the appropriate meaning.

Signs consist of sequences of movements of the upper limbs [10]. Therefore, we have to focus on the measures of these movements, and rests, of each finger, of the wrist and of the arm and forearm too [11]. Such measures can be performed by means of two main approaches, namely image-based and sensor-based methods [8,12]. In particular, the image-based approach demands less-expensive technology, but appropriate light and unobstructed-view conditions, and usually demand of high computational resources. On the other hand, the sensor-based approach requires the signers to wear devices, which can be of some discomfort, but the user is not limited by the scene conditions, since there is no need to stay in front of a camera.

For this work, we decided the use of the sensor-based approach. In particular, we used wearable electronics composed of a homemade sensory glove, used for measuring the movements of the fingers, and inertial measurement units (IMUs), used for measuring the movements of the wrist, forearm, and arm. The potential discomfort of the signers wearing these sensors is minimized, because we built the sensory glove according to the needs and requests of the users, and applied the IMUs with simple Velcro self-gripping straps.

Our research focused on the Italian sign language (Lingua Italiana dei Segni, LIS), that is the Italian one, but our methodology applies to whatever sign language. Our dataset consisted of seven thousand instances of ten gestures measured on seven signers, who performed one hundred instances of each gesture.

For the sign recognition, we considered two classifiers: one based on the k-Nearest Neighbors (k-NN) plus the Dynamic Time Warping (DTW) algorithms (i.e., a non-parametric method), and the other based on Convolutional Neural Networks (CNNs) (i.e., a parametric method). In particular, the k-NN method is applied for the recognition of gestures that may vary in speed, but having similar patterns in the shape of the temporal sequences, whereas the CNN has been more and more proving state of the art accuracies for different kind of signals (in particular for 1D, images, and video).

From the side of measurements, our efforts were particularly devoted to implement a full set of wearable devices, measuring both hand/forearms/arms and fingers of the dominant hand, to realize a dataset with an important number of gesture repetitions (100), and to enroll non-native gesture-speakers, so as to rely on low-repeatability gestures in order to stress the classification algorithms. From the side of classification, we adopted two different approaches, competing or even better performing related works on sign language recognition.

2. Related Works

In the scientific literature, the works that adopt a sensor-based approach use a sensory glove with embedded flex sensors, or inertial measurement units (IMUs), or both [13]. Flex sensors are used for measuring flexion and extension of the fingers' joints [14,15]; IMUs are used for measuring the accelerations and rotations of the palm/wrist and/or the arm/forearm. The number of involved signers, the number of measured signs, and the number of sign repetitions can vary from work to work. Moreover, a number of different classification algorithms have been reported in the literature, underlying the challenging issues to be solved towards the optimal solution for sign language recognition. Within this frame, in the remainder of this section, we present some relevant works for comparison purposes.

Mohandes et al. [16] used the PowerGlove, which provides hand's space position, wrist roll, and four bending finger movements [17]. The glove was used to repeat 10 signs for 20 times. The adopted support vector machine (SVM) algorithm reached an accuracy greater than 90% ± 10(SD).

Mohandes and Deriche [18] measured finger movements using two CyberGloves, with 22 embedded sensors each, and hand movements using the so-called 6-Degrees of Freedom (DOF) "Flock of Birds" device. One signer performed 20 repetitions of 100 two-handed signs. The data dimensionality was reduced by means of Linear Discriminant Analysis (LDA), and the classification was based on the minimum distance (MD) algorithm, reaching an accuracy of 96.2% ± 0.78 (SD).

Two DG5-VHand data gloves, both equipped with five flex sensors and a three-axes accelerometer, were adopted by Tubaiz et al. [19]. Those two gloves measured gestures performed by one signer, who repeated ten times 40 sentences, later divided into 80 words by means of manual labeling separation. With the use of a Modified k-Nearest Neighbors classifier (Mk-NN), researchers obtained the best result as 82% ± 4.88 (SD).

Abualola et al. [20] used a glove equipped with six IMUs to measure fingers and palm movements of nine participants, while signing 24 static letters, one sample per letter per user. Linear Discriminant Analysis (LDA) was applied to reduce the data complexity later analyzed by the Euclidean distance-based classification algorithm, resulting with an accuracy of 85%.

Hernandez et al. [21] adopted a technology based on the "AcceleGlove" and a two-link arm skeleton. Their database consisted of data coming from 17 signers who repeated 30 times one-hand gestures. The classification was based on Conditional Template Matching (CTM) and the resulting recognition rate was equal to 98%.

Lu et al. [22] used the "YoBu" sensory glove, made of eighteen low-cost inertial and magnetic measurement units. This sensor was used to acquire both arm and hand motions, simultaneously. The glove acquired data of 10 types of static gestures. The classification was based on Extreme Learning Machine kernel (ELM-kernel) and on Support Vector Machine (SVM) algorithms, with classification accuracies of 89.59% and 83.65%, respectively.

A work done by Saengsri et al. [23] measured signs by means of a sensory glove and a motion tracker. The first was the "5DT Data Glove 14 Ultra", with 14 sensors for measuring both flexures and abductions of the fingers. The latter provided 3D spatial coordinates of the hand. The dataset consisted of 16 signs, 4 samples for each sign made by a professional signer. The adopted Elman Back Propagation Neural Network (ENN) algorithm resulted with an accuracy of 94.44%.

Silva et al. [24] used a sensory glove, which was equipped with a flex sensor, a gyroscope and an accelerometer attached on the distal phalanx of each finger. The signs were alphabet characters, measured 100 times each. An Artificial Neural Network (ANN) reached 95.8% of recognition accuracy.

Although the aforementioned papers demonstrate important efforts to solve the problem of sign language recognition, the optimal solution remains a challenging problem. This is because there is a lack in defining the best measurement technology and the best classifier, all having advantages and drawbacks too.

We present two different classification algorithms, both competing or even better performing similar literature works. The algorithm based on convolutional neural networks is well suited for training sets from few to thousands, while the algorithm based on the k-nearest neighbors, combined with dynamic time warping, is a good option for scenarios with training sets made of examples in the order of dozens or hundreds.

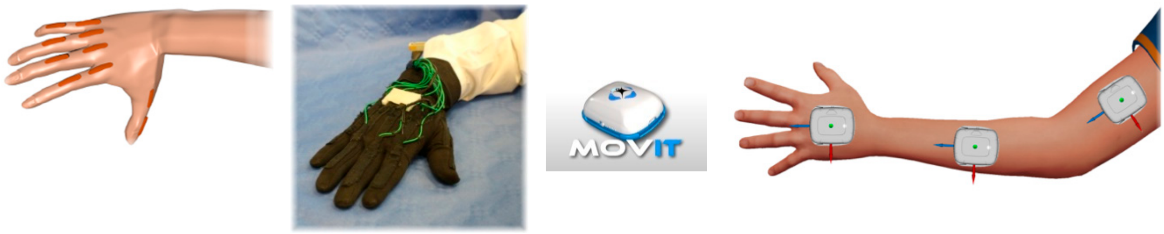
This work explores possibilities gathered from a double technology measurement approach (a sensory glove plus IMUs), from a huge number of repetitions (100) of each gesture, and from a two-model classification approach (k-NN with DTW and CNN algorithms). In addition, given that native signers sign in highly repeatable manner, we adopted language signs performed by trainees (with lower repetition ability) in order to evidence the robustness of the system, if any.

3. Materials and Methods

Our wearable electronics (simply called wearables hereafter) is composed of a sensory glove and IMUs. In particular, the sensory glove allows measuring the movements of the fingers' joints of the dominant hand (specifically the right one) for every signer (Figure 1a,b), and the IMUs allow measuring the spatial arrangements of the upper limbs (Figure 1c,d).

In general, for the most part of sign languages (and in particular for the here adopted Italian one), the meaning concerns the movements of the fingers of dominant hand, the non-dominant one acting symmetrically or to intensify the meaning only. Therefore, here we adopted a single glove for the dominant hand. Figure 2 shows the overall system.

Data, wireless send via Zigbee protocol to a receiving unit, are stored on a personal computer (Intel i5, 8GB RAM) and reproduced on a screen via avatar (Figure 2).



(**a**) (**b**) (**c**) (**d**)

Figure 1. (**a**) Arrangement of the ten flex sensors on top of the carpal–metacarpal and metacarpal–phalangeal joints of the fingers. (**b**) The sensory glove equipped with the ten flex sensors, singularly hosted in one pocket each. (**c**) The inertial measurement units termed Movit. (**d**) Arrangement of the inertial measurement units (IMUs) on the dorsal aspect of the hand, on the forearm and arm, for both of the upper limbs.

3.1. Sensory Glove

We adopted an indigenously developed sensory glove, termed Hiteg Glove, in order to take into account the requests of comfort and usability evidenced by the signers. Our sensory glove consists of a 70% Lycra and 30% cotton fabric with 10 slim pockets, each hosting one flex sensor [25,26]. An on-purpose hardware translates the analog data from the flex sensors into 12-bit digital values at a 40 Hz sampling frequency. The pockets were sewn onto the regions of the carpal–metacarpal and the metacarpal–phalangeal joints to measure the flexion/extension movements of all fingers.

3.2. IMUs

We used six inertial measurement units (IMUs), three for the left and three for the right upper limb, so as to measure the space arrangement and the movements of the wrists, arms, and forearms. The IMU, previously validated [27–29], is termed "Movit G1" (by Captiks Srl Rome, Italy) (Figure 1c), and hosts three-axial accelerometer, gyroscope and a compass (plus a barometer and a thermometer too, but not used for our purposes). The sampling rate for measurement can be set within 4–200 Hz, the acceleration within ±2 g to ±16 g (gravities), the angular velocity within ±250 dps to ±2000 dps (degree per seconds). Each IMU can store the measured data and/or provide the data to a unique receiver via a proprietary wireless protocol, or through a USB cable.

3.3. Calibration and Data Acquisition

The "Captiks Motion Studio" software store data and reproduce gestures by means of an avatar on a computer screen. The starting procedure consists of a calibration step both for the sensory glove and for the IMUs. In particular for the glove, on the hypothesis of a linear response of the bend sensors, it is requested to the wearer to place the hand completely open (flat) and completely closed (wrist), so that the software will interpolate all the intermediate angles of the joints of the fingers.

For the IMUs, a patented calibration procedure consists of placing the IMUs on a support turned on three orthogonal planes in sequence, and then wearer by the subject posed in a "T" condition (the arms parallel to the floor) to assign a starting plane to each sensor.

The software synchronizes all acquired data by adding timestamps. Thus, the LIS recognition can benefit from simultaneous, multiple sensors, multiple technologies, synchronized data collection, and real-time kinematic reconstruction. The open-source universal messaging ZeroMQ socket (http://zeromq.org/) is configured for transferring the streaming of the acquired data from Motion

Studio to MATLAB® and here saved in MATLAB tables. The proposed off-line study permits to identify the most appropriate method for real-time LIS word recognition, further improvable for the LIS consecutive sentence recognition. For this work, we selected 40 Hz sampling rate from the glove's flex sensors and 200 Hz sampling rate for the IMUs. Delays in the signal chain resulted in an effective average of 37.5 Hz data rate from IMUs. Moreover, we selected ±2 g and ±2000 dps for the acceleration and angular degree scales, respectively.

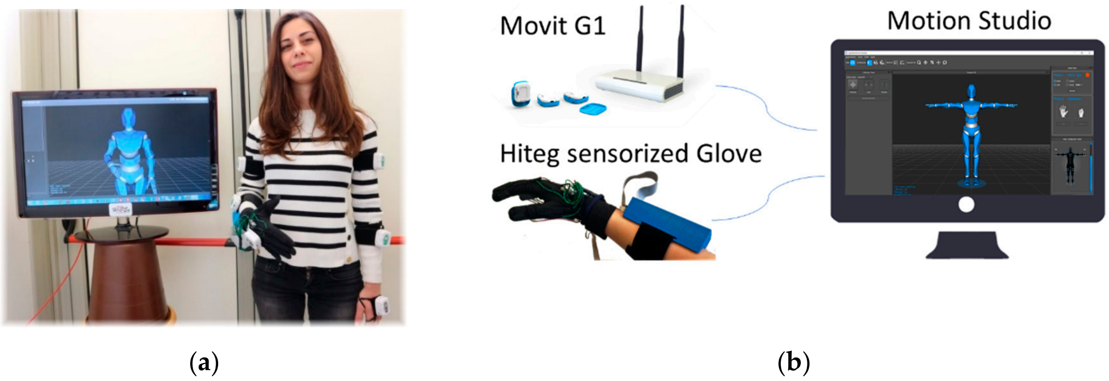


Figure 2. (**a**) A signer with the sensory glove and the six IMUs on the hand/forearm/arm. (**b**) Block diagram of the system: an avatar reproduces gestures on a computer screen, to visually control that the correct data flow from the sensors; the software manages the data stream and the synchronization of the two system.

3.4. Signers and Signs

Seven signers were recruited for the experiments, five males and two females aged 29–54 y ± 10.34 (SD). All signers were right-handed and with similar hand dimensions, so that a unique-size right-hand sensory glove was used.

The signers were asked to replicate, as accurately as possible, 10 common sign gestures, as showed by a professional language signer. The 10 signs were selected among the most used in LIS (the Italian Sign Language), such as ciao, cose (things), grazie (thanks), maestra (teacher), google, internet, jogging, pizza, TV, and twitter. Most of these 10 signs refers to words internationally known.

Each signer, in turn, wore the sensory glove and the six IMUs by means of Velcro straps, and performed signs when sitting on a chair in front of a table, with his/her right hand resting on the table. The time duration of each sign depended on the natural performance of the user, without any constraint. For each user and each sign, a continuous stream of data, generated by the electronic wearables, was stored on a personal computer. For a mere qualitative correct data flow control, an avatar replicated the signer's movements (Figure 2).

4. Classifiers

We applied two different classification models on the same dataset: the first is a non-parametric method based on the k-Nearest Neighbors (k-NN) classifier and the Dynamic Time Warping (DTW) algorithm, and the second is based on Convolutional Neural Networks (CNNs) [30].

4.1. k-NN with DTW

Data representation: The data acquired when a user performs a gesture, or class, is represented as a tuple H = ($\underline{\mathbf{F}}$, $\underline{\overline{\mathbf{A}}}$, $\underline{\overline{\mathbf{W}}}$, L). The matrix $\underline{\mathbf{F}}$ contains the data acquired by the flex sensors from the sensory glove, the hyper-matrices $\underline{\overline{\mathbf{A}}}$ and $\underline{\overline{\mathbf{W}}}$ contain the data from the accelerometers and the gyroscopes, respectively. The categorical variable $L = 1, 2, \ldots, c$, with $c \in \mathbb{Z}^+$, denotes the corresponding label of the signals $\underline{\mathbf{F}}$, $\underline{\overline{\mathbf{A}}}$, $\underline{\overline{\mathbf{W}}}$.

The matrix $\underline{\mathbf{F}} = (\mathbf{F}^{(1)}, \ldots, \mathbf{F}^{(i)}, \ldots, \mathbf{F}^{(10)})$ is formed by 10 column vectors of the form $\mathbf{F}^{(i)} = (f_1^{(i)}, \ldots, f_n^{(i)}, \ldots, f_N^{(i)})^{\mathrm{T}} \in \mathbb{R}^{N \times 1}$, with $i = 1, 2, \ldots, 10$ and $N \in \mathbb{Z}^+$. The value of $f_n^{(i)}$ represents the electric resistance, in ohms, measured at the nth instant in the ith flex sensor of the Hiteg Glove, at a sampling frequency of 40 Hz. The hypermatrix $\overline{\underline{\mathbf{A}}} = (\overline{\underline{\mathbf{A}}}^{(1)}, \ldots, \overline{\underline{\mathbf{A}}}^{(j)}, \ldots, \overline{\underline{\mathbf{A}}}^{(6)}) \in \mathbb{R}^{M \times 6 \times 3}$ is formed by the hypermatrices $\overline{\underline{\mathbf{A}}}^{(j)} = ({}^{x}\mathbf{A}^{(j)}, {}^{y}\mathbf{A}^{(j)}, {}^{z}\mathbf{A}^{(j)}) \in \mathbb{R}^{M \times 1 \times 3}$, with $j = 1, 2, \ldots, 6$ and $M \in \mathbb{Z}^+$. The vector ${}^{s}\mathbf{A}^{(j)} = ({}^{s}a_1^{(j)}, \ldots, {}^{s}a_m^{(j)}, \ldots, {}^{s}a_M^{(j)})^{\mathrm{T}} \in \mathbb{R}^{M \times 1}$ is the measurement returned by the jth inertial sensor of the Movit G1 in the $\underline{s}$th coordinate axis, with $s \in \{x,y,z\}$. The value of ${}^{s}a_m^{(j)}$ represents the acceleration, in fractions of gravity, measured at a sampling frequency of 37.5 Hz at the mth instant of time.

Similarly, the hypermatrix $\overline{\underline{\mathbf{W}}} = (\overline{\underline{\mathbf{W}}}^{(1)}, \ldots, \overline{\underline{\mathbf{W}}}^{(j)}, \ldots, \overline{\underline{\mathbf{W}}}^{(6)}) \in \mathbb{R}^{M \times 6 \times 3}$ corresponds to the angular velocity, in degrees per second, measured at a sampling frequency of 37.5 Hz, and it is formed by the hypermatrices $\overline{\underline{\mathbf{W}}}^{(j)} = ({}^{x}\mathbf{W}^{(j)}, {}^{y}\mathbf{W}^{(j)}, {}^{z}\mathbf{W}^{(j)}) \in \mathbb{R}^{M \times 1 \times 3}$. The column vector ${}^{s}\mathbf{W}^{(j)} = ({}^{s}w_1^{(j)}, \ldots, {}^{s}w_m^{(j)}, \ldots, {}^{s}w_M^{(j)})^{\mathrm{T}} \in \mathbb{R}^{M \times 1}$ contains the data of the jth inertial sensor in the sth coordinate axis, with $s \in \{\mathrm{x,y,z}\}$.

The length N of the column vectors $\mathbf{F}^{(i)}$ of the flex sensors and the length M of the vectors ${}^{s}\mathbf{A}^{(j)}$ of the accelerometer and the vectors ${}^{s}\mathbf{W}^{(j)}$ of the angular velocity are not equal. This is because of the difference in the sampling rate of each sensor: 40 Hz for the flex sensors and 37.5 Hz for the IMU sensors.

Datasets: The signs are grouped to form a dataset $\mathcal{H} = \{\mathcal{H}_1, \ldots, \mathcal{H}_1, \ldots, \mathcal{H}_U\}$, with $U \in \mathbb{Z}^+$, where the example $\mathcal{H}_u = (\underline{\mathbf{F}}_u, \overline{\underline{\mathbf{A}}}_u, \overline{\underline{\mathbf{W}}}_u, L_u)$ is the uth instance of a gesture labeled with L_u, where $u = 1, 2, \ldots, U$. Data were randomly split into a test set (with 20% of the original dataset) and a training set (with the remaining 80% of the original dataset).

Training and testing sets: A subset $\mathcal{H}_1^v$ from the set $\boldsymbol{\mathcal{H}}$ was used for training and the remaining subset $\mathcal{H} - \mathcal{H}_1^v$ was used for testing, where $0 < v < U$, with $v \in \mathbb{Z}^+$. Since the k-NN with DTW algorithm does not need any training (but only the adjustment of the number of neighbors used for the classification), here the term training refers to differentiate data used to find the k-nearest neighbors to the signal being classified, which comes from the testing set used to estimate the accuracy of the classification model. Therefore, we should be clear that.

Pre-processing: In this stage, we first normalized the amplitude and then filtered the noise of the time series contained in the column vectors of the examples $(\underline{\mathbf{F}}, \overline{\underline{\mathbf{A}}}, \overline{\underline{\mathbf{W}}})$ from the set $\mathcal{H}_1^v$.

For the normalization, we used the function $\mathcal{L} : \mathbb{R}^{N \times 1} \rightarrow \mathbb{R}^{N \times 1}$ defined through the following equation:

$$\mathcal{L}(\mathbf{V}) = \frac{\mathbf{V} - \overline{\underline{\mathbf{V}}}_{\min}}{\overline{\underline{\mathbf{V}}}_{\max} - \overline{\underline{\mathbf{V}}}_{\min}}, \tag{1}$$

In this equation, $\mathbf{V} \in \{\underline{\mathbf{F}}, \overline{\underline{\mathbf{A}}}, \overline{\underline{\mathbf{W}}}\}$ denotes the vector to normalize and $\mathcal{L}(\mathbf{V})$ denotes the normalized vector. The values of $\overline{\underline{\mathbf{V}}}_{\max}$ and $\overline{\underline{\mathbf{V}}}_{\min}$ are computed among the values of the element from the set $\{\underline{\mathbf{F}}, \overline{\underline{\mathbf{A}}}, \overline{\underline{\mathbf{W}}}\}$ that $\mathbf{V}$ takes. For example, if we want to normalize the values of the vector $\mathbf{F}^{(i)} \in \underline{\mathbf{F}}$, being $\mathbf{V} = \mathbf{F}^{(i)}$, the $\overline{\underline{\mathbf{V}}}_{\max}$ and $\overline{\underline{\mathbf{V}}}_{\min}$ are the maximum and minimum values, respectively, computed among all the values of the column vectors that form the set $\{\underline{\mathbf{F}}_1, \ldots, \underline{\mathbf{F}}_v\}$. A similar process is applied for the case where $\mathbf{V} = {}^{s}\mathbf{A}^{(j)}$ and $\mathbf{V} = {}^{s}\mathbf{W}^{(j)}$. This normalization function $\mathcal{L}$ is applied over all the time series of each example from the training set $\mathcal{H}_1^v$, so that each component of the new vectors is in the range [0,1]. Applying this normalization over all the instances (i.e., tuples) of the training set, we obtain the new set $\mathcal{L}\left(\mathcal{H}_1^v\right) = \left\{\left(\mathcal{L}\left(\underline{\mathbf{F}}_u\right), \mathcal{L}\left(\overline{\underline{\mathbf{A}}}_u\right), \mathcal{L}\left(\overline{\underline{\mathbf{W}}}_u\right), L_u\right)_{u=1}^{v}\right\}$.

For filtering, we apply a low pass filter $\boldsymbol{\psi}$ to the normalized time series $\mathcal{L}\left(\overline{\underline{\mathbf{A}}}_u\right)$ and $\mathcal{L}\left(\overline{\underline{\mathbf{W}}}_u\right)$ of the inertial sensors only. The function $\boldsymbol{\psi}$ is a digital Butterworth filter [31] of 4th order, where the cutoff frequency is $f_c = 0.05\ \pi f_{\mathrm{s}}$, where f_{s} is the value of the sampling frequency in Hz. From this step, we obtain the set:

$$\psi\Big(\mathcal{L}\Big(\mathcal{H}_1^v\Big)\Big) = \Big\{\Big(\mathcal{L}\Big(\underline{\mathbf{F}}_u\Big), \psi\Big(\mathcal{L}\Big(\underline{\overline{\mathbf{A}}}_u\Big)\Big), \psi\Big(\mathcal{L}\Big(\underline{\overline{\mathbf{W}}}_u\Big)\Big), L_u\Big)_{u=1}^v\Big\} \tag{2}$$

Classification: The classification assigns a label to a gesture $\mathbf{X} \in \mathcal{H}_{v+1}^U$ of the testing set based on the k-Nearest Neighbors (k-NN) classifier and the Dynamic Time Warping (DTW) algorithm [32].

We computed a distance value $d \in \mathbb{R}^+$ that represents the similarity between the unlabeled preprocessed gesture $\psi(\mathcal{L}(\mathbf{X}))$, with $\mathbf{X} = (\underline{\mathbf{F}}, \underline{\overline{\mathbf{A}}}, \underline{\overline{\mathbf{W}}})$, and the 3 first elements of each tuple from the preprocessed training set $\psi\Big(\mathcal{L}\Big(\mathcal{H}_1^v\Big)\Big)$. We used the DTW algorithm [33] to optimally align the corresponding channels of $\psi(\mathcal{L}(\mathbf{X}))$ and each tuple from the set $\psi\Big(\mathcal{L}\Big(\mathcal{H}_1^v\Big)\Big)$. For computing the distance d between $\psi(\mathcal{L}(\mathbf{X}))$ and the three first elements of each tuple from the set $\psi\Big(\mathcal{L}\Big(\mathcal{H}_1^v\Big)\Big)$, we first computed the DTW distance $d(\mathbf{F}^{(i)})$ between the corresponding channels of the flex sensors (Equation (3)), the distance $d({}^{(s)}\mathbf{A}^{(j)})$ between the corresponding channels of the accelerometers (Equation (4)), and the distance $d({}^{(s)}\mathbf{W}^{(j)})$ between the corresponding channels of the angular velocity (Equation (5)).

$$d\Big(\mathbf{F}^{(i)}\Big) = DTW\Big(\mathcal{L}\Big(\mathbf{F}^{(i)}\Big), \mathcal{L}\Big(F_u^{(i)}\Big)\Big),\ \mathbf{F}^{(i)} \in \mathbf{X}, \tag{3}$$

$$d({}^{(s)}\mathbf{A}^{(j)}) = DTW\Big(\psi\Big(\mathcal{L}({}^{(s)}\mathbf{A}^{(j)})\Big), \psi\Big(\mathcal{L}({}^{(s)}\mathbf{A}_u^{(j)})\Big)\Big),\ {}^{(s)}\mathbf{A}^{(j)} \in \mathbf{X}, \tag{4}$$

$$d({}^{(s)}\mathbf{W}^{(j)}) = DTW\Big(\psi\Big(\mathcal{L}({}^{(s)}\mathbf{W}^{(j)})\Big), \psi\Big(\mathcal{L}({}^{(s)}\mathbf{W}_u^{(j)})\Big)\Big),\ {}^{(s)}\mathbf{W}^{(j)} \in \mathbf{X}, \tag{5}$$

with $u = 1, 2, \ldots, U$, where $U \in \mathbb{Z}^+$ is the size of the training set.

For computing the total distance d_u between $\psi(\mathcal{L}(\mathbf{X}))$ and the uth tuple from the preprocessed training set $\psi\Big(\mathcal{L}\Big(\mathcal{H}_{u=1}^v\Big)\Big)$, we summed all the DTW distances obtained using the Equations (3)–(5):

$$d_u = \sum_{i=1}^{10} d(\mathbf{F}^{(i)}) + \sum_{j=1}^{6}\Big(\sum_{s\in\{x,y,z\}} d({}^{(s)}\mathbf{A}^{(j)}) + d({}^{(s)}\mathbf{W}^{(j)})\Big). \tag{6}$$

The label Y, with $Y \in \{1, 2, \ldots, c\}$, that the classifier returns for the unknown gesture $\mathbf{X}$ corresponds to the most voted label among the k-nearest neighbors in terms of the DTW distance, [34,35] found in the preprocessed training set $\psi\Big(\mathcal{L}\Big(\mathcal{H}_1^v\Big)\Big)$.

For the kNN and DTW classification algorithm, we evaluated several implementations of DTW. The options that we tested range from evaluating the whole matrix of distances between the two signals to find the best path of alignment and more efficient implementations based on dynamic programming and window constraints of the space of search for the best path of alignment of the signals. The criteria that we used for evaluating different implementations of DTW was the gain in the classification accuracy versus the increase in the time of processing of the classification algorithm. For all the implementations that we tested, there was not a significant difference in terms of the classification accuracy, but there was an important difference in the time of processing. Based on this analysis, we chose the option based on dynamic programming with a window constraint in the space of search space of length 50 [36].

4.2. CNN

Datasets: In the same way as for the k-NN with the DTW algorithm, for the CNN classifier data were randomly split into a test set (20% of the original dataset) and a training set (the remaining 80%). The training set was further split into two subsets to create a validation set. As common practice in literature, we used a holdout validation scheme considering 20% of the training samples for model selection and hyper-parameters tuning and the remaining 80% for training the model.

Pre-processing: All signals were re-sampled to a fix number of 120 samples by means of a cubic spline interpolation using not-a-knot end conditions. Signals were further normalized, dividing each dimension by its maximum across the training set, using the normalization function $\mathcal{L} : \mathbb{R}^{N\times 1} \rightarrow \mathbb{R}^{N\times 1}$:

$$\mathcal{L}(\mathbf{V}) = \frac{\mathbf{V}}{\underline{\overline{\mathbf{V}}}_{\max}}. \tag{7}$$

Network architecture: We tried several network architectures and hyper-parameter values for Convolutional Neural Networks (CNN) [30] including different numbers and complexity of the convolutional layers, various learning rates and mini-batch sizes. We also tried Long Short Term Memory (LSTM) networks [37]. Among all the architectures and the hyper-parameters values that we tried, the best validation accuracy was achieved by a CNN network with only one convolutional layer with 20 filters of size 16, ReLU activation and batch normalization [38], no pooling layers, and 16 × 16 sized kernel. A fully connected layer was then used to obtain the vector $\mathbf{Z} = (Z_1, \ldots, Z_c) \in \mathbb{R}^{c\times 1}$, where c is the number of classes, being c = 10 in our case (Figure 3).

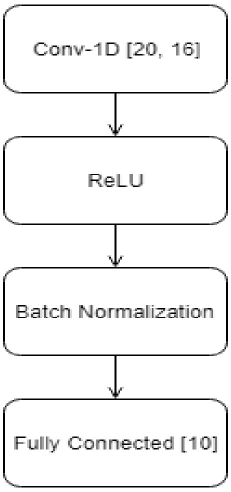


Figure 3. Convolutional Neural Network (CNN) architecture used in this work.

Training phase: We chose to optimize the cross-entropy loss function of the Matlab softmax function applied to the resulting vector **Z** together with L2 weights regularizes, where the jth component of the softmax is defined as follows:

$$p_j = \frac{e^{Z_j}}{\sum_{k=1}^{c} e^{Z_k}}, \tag{8}$$

with $j = 1, \ldots, c$. The cross-entropy between the resulting vector of probabilities $\mathbf{p} = (p_1, \ldots, p_c)$ predicted by the network and the expected one-hot encoding $\mathbf{Y} = (Y_1, \ldots, Y_c)$ of the actual label Y of the gesture **X** is defined as follows:

$$H(\mathbf{Y}, \mathbf{p}) = -\sum_{i=1}^{c} Y_i \times \log(p_i). \tag{9}$$

The CNN was trained using stochastic gradient descent with momentum, with a learning rate of 10^{-4} and a mini-batch size of 200. The training went on for 9600 iterations and was stopped when the validation loss did not increase for five consecutive times. This took a total time of 244 s using a Tesla K40 GPU and MATLAB® with its Neural Network Toolbox.

5. Results and Discussion

For this work, we used a dataset composed of 10 different gestures, repeated 100 times each by 7 users, obtaining a dataset of U = 7000 tuples. This dataset $\mathcal{H}_1^U$ was divided into two subsets: one subset $\mathcal{H}_1^{5600}$ contains 80% of the examples for designing and training the classification models, and the other subset $\mathcal{H}_{5601}^{7000}$, 20% of $\mathcal{H}_1^U$, was used for testing the models. Before performing this partition, the elements of the original dataset were randomly shuffled.

5.1. Results with the k-NN and DTW Algorithm

For defining the model composed of the *k*-NN classifier with DTW algorithm, we tested this model varying two parameters: the size *N* of the training set and the number *k* of neighbors. Each training set $\mathcal{H}_1^N$ was formed by selecting the first *N* elements from $\mathcal{H}_1^{5600}$, where N = 70, 140, 210, 280, 350, i.e., with an incremental step of 70 tuples, so that to balance the tuples added from each user. Moreover, the training set had the same number of tuples for each sign to recognize in order to be balanced. The number *k* of neighbors varies from 1 to *N*/7, where (*N*/7) $\in \mathbb{Z}^+$ is the number of tuples per gesture in the training set $\mathcal{H}_1^N$. It is worth noting that, for the classification using the k-NN with the DTW algorithm, we only used a small fraction of the training set in order to reduce the time of classification without reducing significantly the accuracy of the model. The accuracies reported in Figure 4a were computed by applying each model tested to all the tuples in the test set $\mathcal{H}_{5601}^{7000}$. Figure 4b shows the variation of the time of classification versus the size *N* of the training set. Each point of Figure 4b was obtained as the average of the times of classifying each tuple from the test set.

As Figure 4 shows, the accuracy of the classification model improves with the increase of the size *N* of the training set; however, the time required for training a model with a larger number of examples increases at a growth rate of approximately *O(N)*. Table 1 shows the confusion matrix related to the results obtained when the classification model was trained using N = 140 examples (2 repetitions per user for each gesture) and just one neighbor, k = 1, for the *k*-NN algorithm. It is worth mentioning that these settings correspond to a balance between a good accuracy and a low processing time. In the confusion matrix, the predicted gestures are in the rows, whereas the actual gestures are in the columns. The percentage shown in each cell, except for the cells of the last column and the last row, is obtained by diving the value that accompanies the percentage by the sum of the values of all the cells of the matrix, except for the last row and the last column. Therefore, the accuracy of each method (the last value of the main diagonal) is obtained by adding up the percentages along the main diagonal, except for the last value. The precisions (percentages are shown in the last column) are computed by dividing the value of the main diagonal, in each row, by the sum of the values of that row. The sensitivities (percentages shown in the last row) are computed by dividing the value of the main diagonal, in each column, by the sum of the values of that column.

Table 1 shows two main results related to the confusion: on the lower row the sensitivity rates (i.e., percentage of gestures correctly classified over the total of gestures performed for a given class), and on the right column the precision rates (i.e., percentage of gestures correctly classified over the total of gestures predicted for a given class). Three gestures reach 100% of sensitivity rate: jogging, pizza, and twitter. On the other hand, the worst sensitivity is for the gesture cose with 87.5%, while its precision is 100%. This means that every time that cose was predicted, it was correct, but many cose signs were not identified as such. Interestingly, the sign google has an opposite behavior, because its precision is just 89.4% (the worst among all precisions), but its sensitivity is 96.4%. Finally, the overall recognition accuracy of the k-NN with DTW model is 96.6%.

5.2. Results with the CNN Algorithm

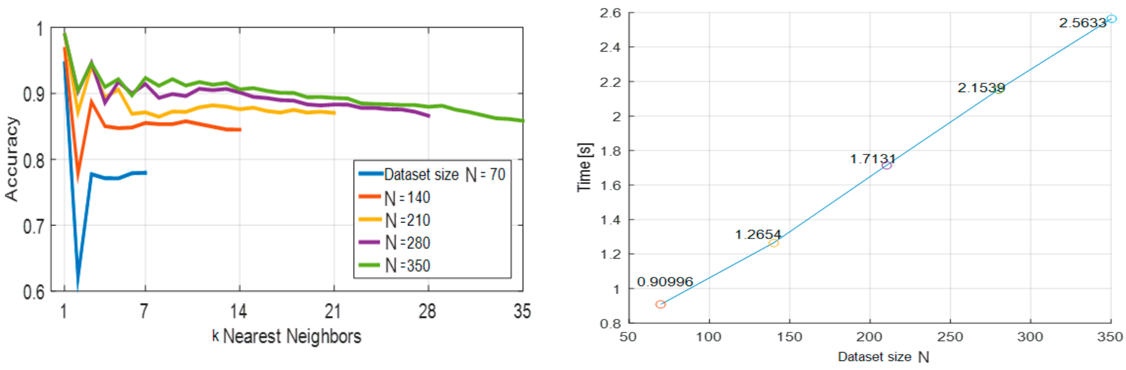
Table 2 shows the classification accuracy obtained by the CNN classifier. The experiments were repeated 10 times using different random seeds for hyperparameters initialization and the mean accuracy is reported with results equal to 98% on the test set, 98.04% on the validation set, and 99.8% on the training set. Deeper architectures with more than one convolutional layer and pooling layers led to comparable if not worse results. This was somehow expected, since the small number of classes. Among all the architectures that gave us comparable results, we picked the smallest since we envision this application to run in real time and a smaller network would lead to faster performances. It is worth mentioning that the time performance of the training phase on GPU is also very fast.

Table 1. Confusion matrix for the k-NN and DTW classification model.

Google	135; 9.6%	0; 0%	0; 0%	0; 0%	1; 0.1%	0; 0%	0; 0%	13; 0.9%	2; 0.1%	0; 0%	89.4%
Internet	0; 0%	132; 9.4%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	4; 0.3%	0; 0%	0; 0%	97.1%
Jogging	2; 0.1%	0; 0.0%	140; 10%	0; 0%	8; 0%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	93.3%
Pizza	0; 0%	0; 0%	0; 0%	140; 10%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	100%
Television	3; 0.2%	0; 0.0%	0; 0%	0; 0.0%	131; 9.4%	0; 0%	0; 0%	0; 0%	0; 0%	1; 0.1%	97.0%
Twitter	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	140; 10%	0; 0%	0; 0%	0; 0%	0; 0.0%	100%
Ciao	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	139; 9.9%	0; 0%	0; 0%	0; 0%	100%
Cose	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	120; 8.6%	0; 0%	0; 0%	100%
Grazie	0; 0%	8; 0.6%	0; 0%	0; 0%	0; 0%	0; 0%	1; 0.1%	3; 0.2%	138; 9.9%	1; 0.1%	91.4%
Maestra	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	138; 9.9%	100%;
SENSITIVITY	96.4%	94.3%	100%	100%	93.6%	100%	99.3%	85.7%	98.6%	98.6%	96.6%
	Google	**Internet**	**Jogging**	**Pizza**	**Television**	**Twitter**	**Ciao**	**Cose**	**Grazie**	**Maestra**	**PRECISION**

Table 2. Confusion matrix for the CNN classification model.

Google	139; 9.9%	0; 0%	1; 0.1%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	1; 0.1%	0; 0%	98.6%
Internet	0; 0%	139; 9.9%	0; 0%	0; 0%	0; 0%	0; 0%	1; 0.1%	0; 0%	0; 0%	1; 0.1%	98.6%
Jogging	0; 0%	0; 0%	135; 9.6%	1; 0.1%	0; 0%	1; 0.1%	2; 0.1%	2; 0.1%	1; 0.1%	0; 0%	95.1%
Pizza	0; 0%	0; 0%	0; 0%	138; 9.9%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	100%
Television	0; 0%	0; 0%	0; 0%	0; 0%	140; 10.0%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	100%
Twitter	0; 0%	0; 0%	0; 0%	1; 0.1%	0; 0%	139; 9.9%	0; 0%	0; 0%	0; 0%	0; 0%	99.3%
Ciao	0; 0%	0; 0%	2; 0.1%	0; 0%	0; 0%	0; 0%	133; 9.5%	0; 0%	0; 0%	0; 0%	98.5%
Cose	1; 0.1%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	137; 9.8%	1; 0.1%	2; 0.1%	97.2%
Grazie	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	0; 0%	1; 0.1%	135; 9.6%	0; 0%	99.3%
Maestra	0; 0%	1; 0.1%	2; 0.1%	0; 0%	0; 0%	0; 0%	4; 0.3%	0; 0%	2; 0.1%	137; 9.8%	93.8%
SENSITIVITY	99.3%	99.3%	96.4%	98.6%	100%	99.3%;	95.0%	97.9%	96.4%	97.9%	98.0%
	Google	**Internet**	**Jogging**	**Pizza**	**Television**	**Twitter**	**Ciao**	**Cose**	**Grazie**	**Maestra**	**PRECISION**



(**a**) (**b**)

Figure 4. Related to the *k*-Nearest Neighbors (*k*-NN) and Dynamic Time Warping (DTW) classification algorithm: (**a**) Accuracy for different dataset sizes *N* and number of neighbors *k*; (**b**) Average time of classification versus the size *N* of the training set.

5.3. Comparison of Results with Related Works

The relevant works about sign language recognition were already discussed in Section 2. Now, after reporting our results, we present in Table 3 a comparison of our work with the technologies and the classification accuracies of the works reviewed in this paper. According to this comparison, our method performed better than the other methods. As far as we know, we consider this as a very promising result.

Table 3. Confusion matrix for the CNN classification model.

Reference	Sensor(s)	Signers, Signs, Repetitions	Classifier	Accuracy m ± s [%]
Mohandes et al., 1996 [16]	PowerGlove	n/a, 10, 20	SVM	90 ± 10
Mohandes and Deriche, 2013 [18]	CyberGloves	1, 100, 20	LDA + MD	96.2 ± 0.78
Tubaiz et al., 2015 [19]	DG5 - VHand	1, 40, 10	MKNN	82 ± 4.88
Abualola et al., 2016 [20]	AcceleGlove + skeleton	17, 1, 30	CTM	98 ± n/a
Lu et al., 2016 [22]	YoBuGlove	n/a, 10, n/a	ELM-kernel SVM	89.59 ± n/a 83.65 ± n/a
Saengsri et al., 2012 [23]	5DTGlove + tracker	1, 16, 4	ENN	94.44 ± n/a
Silva et al., 2017 [24]	Glove + IMU	1, 26, 100	ANN	95.8 ± n/a
Our work	HitegGlove + Movit G1 IMU	7, 10, 100	kNN + DTW CNN	96.6 ± 3.4 98 ± 2.0

6. Conclusions

The non-parametric model that combines the *k*-NN classifier with the DTW algorithm has good classification accuracy (96.6%) and its time complexity has a linear growth with the number of examples used to find the *k*-nearest neighbors for classification. The classification accuracy of this model improves with the increase of the number of training examples, but at the cost of increasing also the time of classification. Therefore, practical applications of sign language classification, especially in real time, using the *k*-NN and DTW model demand a tradeoff between accuracy and time of processing. Even though the time of classification is a big drawback for this model when we have a large dataset, its high accuracy has to be considered, especially for scenarios where the number of examples for the classification is limited. As it is usual, the computing time of this and other models depends highly on the hardware used to run the model. The *k*-NN and DTW model was tested on a desktop computer with an Intel® Core™ i7-3770S processor (Intel Corporation, Santa Clara, CA, USA) and 4 GB of RAM.

The relatively high classification accuracy and linear time complexity of the *k*-NN and DTW model suggest that it can be a good option for applications where the processing and storage capabilities are limited and the number of available data is in the order of the dozens.

The CNN parametric classifier performed with a very high accuracy value, quite close to 100%. We also tried to run inference of this model using a CPU, and we got up to 500 gestures per second on computer with an Intel® Core™ i7-7700 processor ((Intel Corporation, Santa Clara, CA, USA) and 16 GB of RAM. This speed of processing means that inference in real time is possible using the CNN classifier.

The difference between the accuracies of both models tested in this work might occur because the CNN model captures better the distribution underlying the data of the ten gestures that we used here. However, with the increase of the number of training examples, the *k*-NN and DTW model might lead to similar or even better results than the CNN model at the cost of a higher time of processing. However, increasing the number of training examples will increase the time of training of the CNN, but its time of prediction will be fixed since the complexity of the CNN does not change with the increase of the size of the training set. This last feature of the CNN and the results obtained in this work evidence that this model is an interesting option for sign language classification in scenarios with many training examples.

We reported comparisons of our results with those from other relevant works published in the scientific literature. Although a perfect fit among all the works is almost impossible, since there are differences in terms of the types of wearables and signs, and number of signers and signs, the higher accuracies obtained in this work suggest how our approach can be valuable for sign recognition purposes. There are some major differences with respect to other studies. In particular, the number of used sensors, that constitute a complete set for full fingers, hands/arms/forearms joints analysis; the 100 gesture repetitions for each LIS sign from each signer, that constitute a comprehensive set of valuable data, permit us a fair and reliable comparison between two different data analysis approach. Moreover, the data from the sensory glove and the IMUs are collected and synchronized, providing real-time data-fusion from both sensory glove and IMUs.

As a meaningful further consideration, our database was constructed with signs performed by trainees rather than expert native signers, which adds a sort of "robustness" to the tested classification algorithms. Indeed, a non-native speaker typically repeats a specific gesture with minor precision with respect to a native speaker. Thus, the analysis applies to more dispersed values, thus enabling an expected better performance in case of native-speaker better-accuracy gestures (to be demonstrated in a future work).

Further research includes improvements on the *k*-NN and DTW model using an efficient reduction of the data dimensionality. In this way, better and faster results might be obtained, which in turn may also allow us to recognize more gestures for real-time applications. Finally, both models need to be tested at classifying more classes for different sign languages and for the recognition of consecutive sentences.

Author Contributions: Conceptualization, G.S. and M.E.B.; methodology, G.S., P.C., J.Z., and M.E.B.; software, P.C., M.R., and J.Z.; validation, G.S., P.C., and M.E.B.; formal analysis, P.C. and J.Z.; investigation, G.S., M.R., V.E., J.Z., and M.E.B.; resources, G.S. and M.E.B.; data curation, G.S., P.C., M.R., V.E., J.Z., and M.E.B.; writing—original draft preparation, G.S., P.C., and M.E.B.; writing—review and editing, G.S. and M.E.B.; visualization, G.S. and M.E.B.; supervision, G.S. and M.E.B.; project administration, G.S. and M.E.B.; funding acquisition, G.S. and M.E.B. All authors have read and agreed to the published version of the manuscript.

Funding: Authors gratefully acknowledge the financial support provided by Escuela Politécnica Nacional for the development of the research project PIJ-16-13 "Clasificación de señales electromiográficas del brazo humano usando técnicas de reconocimiento de patrones y machine learning".

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Saggio, G.; Cavrini, F.; Di Paolo, F. Inedited SVM application to automatically tracking and recognizing arm-and-hand visual signals to aircraft. In Proceedings of the 7th International Joint Conference on Computational Intelligence 3, Lisbon, Portugal, 12–14 November 2015; pp. 157–162.
2. Saggio, G.; Cavrini, F.; Pinto, C.A. Recognition of arm-and-hand visual signals by means of SVM to increase aircraft security. *Stud. Comput. Intell.* **2017**, *669*, 444–461.
3. León, M.; Romero, P.; Quevedo, W.; Arteaga, O.; Terán, C.; Benalcázar, M.E.; Andaluz, V.H. Virtual rehabilitation system for fine motor skills using a functional hand orthosis. In Proceedings of the International Conference on Augmented Reality, Virtual Reality and Computer Graphics, Otranto, Italy, 24–27 June 2018; pp. 78–94.
4. Ramírez, F.; Segura-Morales, M.; Benalcázar, M.E. Design of a software architecture and practical applications to exploit the capabilities of a human arm gesture recognition system. In Proceedings of the 3rd IEEE Ecuador Technical Chapters Meeting (ETCM), Cuenca, Ecuador, 15–19 October 2018; pp. 1–6.
5. Tsironi, E.; Barros, P.; Weber, C.; Wermter, S. An analysis of convolutional long short-term memory recurrent neural networks for gesture recognition. *Neurocomputing* **2017**, *268*, 76–86. [CrossRef]
6. Saggio, G.; Bizzarri, M. Feasibility of teleoperations with multi-fingered robotic hand for safe extravehicular manipulations. *Aerosp. Sci. Tech.* **2014**, *39*, 666–674. [CrossRef]
7. Saggio, G.; Lazzaro, A.; Sbernini, L.; Carrano, F.M.; Passi, D.; Corona, A.; Panetta, V.; Gaspari, A.L.; Di Lorenzo, N. Objective surgical skill assessment: An initial experience by means of a sensory glove paving the way to open surgery simulation? *J. Surg. Educ.* **2015**, *72*, 910–917. [CrossRef] [PubMed]
8. Mohandes, M.; Deriche, M.; Liu, J. Image-Based and sensor-based approaches to Arabic sign language recognition. *IEEE Trans. Hum. Mach. Syst.* **2014**, *44*, 551–557. [CrossRef]
9. Who.int. Available online: http://www.who.int/mediacentre/factsheets/fs300/en/ (accessed on 2 May 2018).
10. Valli, C.; Lucas, C. *Linguistics of American Sign Language: An Introduction*, 4th ed.; University Press: Washington, DC, USA, 2000.
11. Yi, B.; Wang, X.; Harris, F.C.; Dascalu, S.M. sEditor: A prototype for a sign language interfacing system. *IEEE Trans. Hum. Mach. Syst.* **2014**, *44*, 499–510. [CrossRef]
12. Saggio, G.; Sbernini, L. New scenarios in human trunk posture measurements for clinical applications. In Proceedings of the IEEE International Symposium on Medical Measurements and Applications (MeMeA 2011), Bari, Italy, 30–31 May 2011.
13. Estrada, L.; Benalcázar, M.E.; Sotomayor, N. Gesture recognition and machine learning applied to sign language translation. In Proceedings of the 7th Latin American Congress on Biomedical Engineering CLAIB 2016, Bucaramanga, Colombia, 26–28 October 2016; pp. 233–236.
14. Orengo, G.; Lagati, A.; Saggio, G. Modeling wearable bend sensor behavior for human motion capture. *IEEE Sens. J.* **2014**, *14*, 2307–2316. [CrossRef]
15. Saggio, G.; Bocchetti, S.; Pinto, C.A.; Orengo, G.; Giannini, F. A novel application method for wearable bend sensors. In Proceedings of the 2nd IEEE International Symposium on Applied Sciences in Biomedical and Communication Technologies, Bratislava, Slovakia, 24–27 November 2009; pp. 1–39.
16. Mohandes, M.; A-Buraiky, S.; Halawani, T.; Al-Baiyat, S. Automation of the Arabic sign language recognition. In Proceedings of the IEEE International Conference on Information and Communication Technologies: From Theory to Applications, Damascus, Syria, 19–23 April 2004; pp. 479–480.
17. Kadous, M.W. Machine recognition of auslan signs using powergloves: Towards large-lexicon recognition of sign language. In Proceedings of the Workshop on the Integration of Gesture in Language and Speech, Wilmington, DE, USA, 7–8 October 1996.
18. Mohandes, M.; Deriche, M. Arabic sign language recognition by decisions fusion using Dempster-Shafer theory of evidence. In Proceedings of the IEEE Computing, Communications and IT Applications Conference, Hong Kong, China, 2–3 April 2013; pp. 90–94.
19. Tubaiz, N.; Shanableh, T.; Assaleh, K. Glove-Based continuous arabic sign language recognition in user-dependent mode. *IEEE Trans. Hum. Mach. Syst.* **2015**, *45*, 526–533. [CrossRef]
20. Abualola, H.; Al Ghothani, H.; Eddin, A.N.; Almoosa, N.; Poon, K. Flexible gesture recognition using wearable inertial sensors. In Proceedings of the 59th IEEE International Midwest Symposium on Circuits and Systems, Abu Dhabi, UAE, 16–19 October 2016; pp. 1–4.

21. Hernandez-Rebollar, J.L.; Kyriakopoulos, N.; Lindeman, R.W. A new instrumented approach for translating American sign language into sound and text. In Proceedings of the 6th IEEE International Conference on Automatic Face and Gesture Recognition, Seoul, Korea, 17–19 May 2004; pp. 547–552.
22. Lu, D.; Yu, Y.; Liu, H. Gesture recognition using data glove: An extreme learning machine method. In Proceedings of the IEEE International Conference on Robotics and Biomimetics, Qingdao, China, 3–7 December 2016; pp. 1349–1354.
23. Saengsri, S.; Niennattrakul, V.; Ratanamahatana, C.A. TFRS: Thai finger-spelling sign language recognition system. In Proceedings of the 2nd IEEE International Conference on Digital Information and Communication Technology and its Applications, Bangkok, Thailand, 16–18 May 2012; pp. 457–462.
24. Silva, B.C.R.; Furriel, G.P.; Pacheco, W.C.; Bulhoes, J.S. Methodology and comparison of devices for recognition of sign language characters. In Proceedings of the 18th IEEE International Scientific Conference on Electric Power Engineering, Kouty nad Desnou, Czech Republic, 17–19 May 2017; pp. 1–6.
25. Saggio, G.; Cavallo, P.; Fabrizio, A.; Ibe, S.O. Gesture recognition through HITEG data glove to provide a new way of communication. In Proceedings of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies (ISABEL'11), Barcelona, Spain, 26 October 2011.
26. Saggio, G.; Orengo, G. Flex sensor characterization against shape and curvature changes. *Sens. Actuators A Phys.* **2018**, *273*, 221–223. [CrossRef]
27. Costantini, G.; Casali, D.; Paolizzo, F.; Alessandrini, M.; Micarelli, A.; Viziano, A.; Saggio, G. Towards the enhancement of body standing balance recovery by means of a wireless audio-biofeedback system. *Med. Eng. Phys.* **2018**, *54*, 74–81. [CrossRef] [PubMed]
28. Ricci, M.; Terribili, M.; Giannini, F.; Errico, V.; Pallotti, A.; Galasso, C.; Tomasello, L.; Sias, S.; Saggio, G. Wearable-Based electronics to objectively support diagnosis of motor impairments in school-aged children. *J. Biomech.* **2019**, *83*, 243–252. [CrossRef] [PubMed]
29. Ricci, M.; Di Lazzaro, G.; Pisani, A.; Mercuri, N.B.; Giannini, F.; Saggio, G. Assessment of Motor Impairments in Early Untreated Parkinson's Disease Patients: The Wearable Electronics Impact. *IEEE J. Biomed. Health Inform.* **2020**, *24*, 120–130. [CrossRef] [PubMed]
30. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
31. Oppenheim, A.V.; Schafer, R.W. *Digital Filter Design Techniques of Digital Signal Processing*, 1st ed.; Prentice Hall: Upper Saddle River, NJ, USA, 1978.
32. Benalcázar, M.E.; Jaramillo, A.; Zea, J.; Páez, A.; Andaluz, V.H. Hand gesture recognition using machine learning and the Myo armband. In Proceedings of the 25th European Signal Processing Conference (EUSIPCO), Kos, Greece, 28 August–2 September 2017; pp. 1040–1044.
33. Müller, M. *Information Retrieval for Music and Motion*; Springer: Berlin, Germany, 2007.
34. Devroye, L.; Györfi, L.; Lugosi, G. *A Probabilistic Theory of Pattern Recognition. Applications of Mathematics. Stochastic Modelling and Applied Probability*; Springer: Berlin, Germany, 1991; Volume 31, pp. 61–81.
35. Benalcázar, M.E.; Motoche, C.; Zea, J.; Jaramillo, A.; Anchundia, C.; Zambrano, P.; Segura, M.; Benalcázar, F.; Pérez, M. Real-Time hand gesture recognition using the myo armband and muscle activity detection. In Proceedings of the 2nd IEEE Ecuador Technical Chapters Meeting (ETCM), Salinas, Ecuador, 18–20 October 2017; pp. 1–6.
36. Wang, Q. Dynamic Time Warping. MATLAB Central File Exchange. Available online: https://www.mathworks.com/matlabcentral/fileexchange/43156-dynamic-time-warping-dtw (accessed on 22 June 2020).
37. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
38. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6–11 July 2015.