informatics

MDPI

Article

Convolutional Neural Networks for Human Activity Recognition Using Body-Worn Sensors

Fernando Moya Rueda [1,*], René Grzeszick [1,2], Gernot A. Fink [1], Sascha Feldhorst [2] and Michael ten Hompel [2,3]

1 Department of Computer Science, TU Dortmund University, 44227 Dortmund, Germany; rene.grzeszick@tu-dortmund.de (R.G.); Gernot.Fink@tu-dortmund.de (G.A.F.)
2 Fraunhofer IML, 44227 Dortmund, Germany; sascha.feldhorst@iml.fraunhofer.de (S.F.); michael.ten.hompel@iml.fraunhofer.de (M.t.H.)
3 Department of Mechanical Engineering, TU Dortmund University, 44227 Dortmund, Germany
* Correspondence: fernando.moya@tu-dortmund.de; Tel.: +49-231-755-4626

Received: 7 March 2018; Accepted: 22 May 2018; Published: 25 May 2018

Abstract: Human activity recognition (HAR) is a classification task for recognizing human movements. Methods of HAR are of great interest as they have become tools for measuring occurrences and durations of human actions, which are the basis of smart assistive technologies and manual processes analysis. Recently, deep neural networks have been deployed for HAR in the context of activities of daily living using multichannel time-series. These time-series are acquired from body-worn devices, which are composed of different types of sensors. The deep architectures process these measurements for finding basic and complex features in human corporal movements, and for classifying them into a set of human actions. As the devices are worn at different parts of the human body, we propose a novel deep neural network for HAR. This network handles sequence measurements from different body-worn devices separately. An evaluation of the architecture is performed on three datasets, the Oportunity, Pamap2, and an industrial dataset, outperforming the state-of-the-art. In addition, different network configurations will also be evaluated. We find that applying convolutions per sensor channel and per body-worn device improves the capabilities of convolutional neural network (CNNs).

Keywords: human activity recognition; order picking; convolutional neural networks; multichannel time-series

1. Introduction

Methods of HAR have been developed for classifying human movements. HAR uses as inputs signals from videos [1] or from multichannel time-series, e.g., measurements from inertial measurement unit (IMUs) [2–5]. This work focuses on HAR from multichannel time-series. Capturing, evaluating and analyzing signal series for recognizing human actions are critical for many applications. For example, HAR is of special interest for developing accurate smart assistive technologies or carrying out optimizations in the industry where manual work remains dominant. HAR is a classification task. A traditional HAR pipeline segments sequences using a sliding-window approach, extracts relevant hand-crafted features from the segmented sequences, and it trains a classifier for assigning certain action labels to the sequences [6–8]. However, this classification is hard, due to large intra- and inter-class variability of human actions. Humans perform similar tasks differently. For example, in the context of order picking, workers might change the way how they pick a box depending on the size of the box, its texture or on their working fatigue. Besides, with respect to the amount of data, there is a strong unbalance of the number of samples per activity class; that is, there are more measurements for

frequent activities such as walking, than for less frequent ones such as grabbing a box [5,8]. Recently, CNNs have been used successfully for solving HAR problems in the context of gesture recognition and activities of daily life. In comparison with standard approaches, CNNs combine conveniently the feature extraction and classification in an end-to-end approach [2–5,9,10]. The features extractors are non-linear transformations, and they are learnt directly from raw data being more discriminative with respect to human action-classes. In contrast, relevant handcrafted-features are hard to compute and to scale. By stacking several filters and pooling operations, convolutional neural network (CNNs) extract hierarchically basic and complex human movements, learning their non-linear and temporal relations [2,9]. These architectures have in common convolution and pooling operations that are carried out along the time axis. In addition, the convolutions' filters are shared among the sensors, and their receptive fields are rather small. Therefore, same parameters are applied to all the sensors and on local signals, as those are likely to be correlated. These extracted temporal relations are invariant to distortions and small temporal-translations [2,4,9]. Time-series are acquired from multiple sensors of body-worn devices. These devices are located on different human body parts, e.g., the human limbs, head, and torso. As human movements might vary with respect to these human body parts, it is natural to process the body-worn sensors' measurements individually.

This paper is an extended version of the work initially published in iWOAR2017 [11]. This work introduced a novel CNN, which is applied to the order picking scenario. This architecture processes body-worn sensors individually. Moreover, this work compared the performance of the architecture with traditional methods based on statistical features, presented in [8], in the context of an order picking scenario. Here, we add novel contributions including an evaluation of the proposed architecture in two benchmark datasets for HAR, since the work in [11] concentrated on a private dataset. Besides, we included an evaluation of different configurations of the deep architecture presented in [11] and of the training procedure involving the number of layers, the usage of max-pooling operations and different learning rates, as well as, a deployment of the deep architectures

The remainder of the paper is organized as follows: Section 2 will discuss the related work in HAR using multichannel time-series. In Section 3, CNNs for HAR will be introduced and the novel IMU centered CNNs will be described. In Section 4, the datasets and implementation details will be explained. Experiments on the datasets, the Opportunity, Pamap2, and order picking datasets will be presented and discussed in Section 5. Finally, conclusions will be drawn in Section 6.

2. Related Work

Multichannel time-series HAR obtains sequence segments and assigns them an activity class label. In the past, HAR has been solved following the standard pipeline in pattern recognition: extracting segments out from an input sequence, computing hand-crafted features, and predicting class labels using classifiers. Statistical features, e.g., mean, median, max, min or the signal magnitude were computed from each of the sensor segments, which were segmented from the input signal by a sliding window approach [6–8]. Classifiers like Support Vector Machine (SVM), Naive Bayes, Random Forest or sequence based approaches such as Dynamic Time Warping (DTW) [12] or Hidden Markov Model (HMMs) [6,13] were frequently used. Following this standard approach, the authors in [8] analyzed the order picking process. They introduced, first, a new dataset for order picking, where recordings are taken from three IMUs worn by three workers in two warehouses. Second, they evaluated different classifiers, namely an SVM, a Bayes classifier and a Random Forest, on segments that were extracted with a sliding window approach using a set of handcrafted statistical features. These are often complemented by features computed on the derivative of the signal or in some cases by a correlation analysis between different signals [7]. However, even with the relatively good performance on small and unbalanced datasets [6], designing and selecting relevant features is time consuming and it is hardly scalable [5]. Besides, approaches using handcrafted features show overfitting to one of the activities in a specific HAR task; thus, different sets of these features must be selected per activity [6,14].

Deep architectures became the standard approach in different fields, for example in image classification [15], image segmentation [16,17], word spotting [18,19], and even in non-visual domains like acoustics [20]. Recently, they have been also used in HAR applications [2–5,9]. CNNs, applied to multichannel time series HAR following a sliding window approach, capture the complex human movements by combining basic ones. CNNs find hierarchical patterns and classify them taking directly raw data from time series. The authors in [3] introduced a CNN with convolutional layers applied along the time axis and over all the sensors simultaneously, called temporal convolution layers. They used two or three of these layers followed by a pooling layer and a softmax classifier. The authors in [4] proposed applying the temporal convolutions for sensors individually sharing the same filter. Besides, they deployed a deeper architecture with four temporal convolutions followed by a fully-connected and a softmax classifier. The fully-connected layer combines the convolved sensor sequences, modeling correlations among the sensors. The authors in [5] combined CNNs and recurrent neural network (RNNs), specifically long short-term memories (LSTMs); RNN are architectures that contain neurons with self-connections, being suitable for processing sequences; LSTM units are recurrent units that include memory cells and a gating system for learning long temporal dependencies in series without exploding or vanishing gradients at training [5,21,22]. The authors replaced the fully-connected layers that are usually part of a CNN by LSTM layers. Their architecture contains four temporal convolution layers followed by two LSTMs layers and a softmax classifier. In [3–5], the authors showed that the CNN exhibit a better performance in comparison with shallower methods using Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), K-Nearest Neighbor (KNN), SVMs. By learning features in conjunction with the classifier, these become more discriminative with respect to human actions. The authors in [9] compared deep neural network (DNNs), containing only fully-connected layers, CNNs, similar to [3] with temporal convolutions over all sensors, LSTMs and bi-directional long short-term memories (BLSTMs). B-LSTM are similar to LSTMs , however, they process sequences following forward and backward directions. They presented a deep LSTM architecture with three layers, and a B-LSTM architecture with a single layer. This last one presents the state-of-the-art performance. Nevertheless, CNNs show results close to the performance of the B-LSTM, being more robust against parameter changes. Following a sliding window approach, a single activity label is predicted per window assuming that all measurements of the window belong to the same activity class. However, this is hardly true for real applications. The authors in [10] replaced the fully-connected layers by fully-convolutional layers for dense prediction of sequences. They presented a deeper architecture in comparison to previous networks. In addition, they padded the convolutional and max-pooling layers for keeping the same size of the sequence input. Differently from using raw data directly, the authors in [23] implemented a compact neural network for mobile devices using spectogram of signals as inputs. This network uses temporal convolutions per frequency, and it performs a weighted summation over the convolved output for all the sensors and per frequency. The authors in [24] extended this approach combining the learnt features with handcrafted ones.

3. Convolutional Neural Networks for HAR

CNNs, first proposed in [25], are hierarchical structures that combine convolutional operations using learnable filters and non-linear activation functions, downsampling operations and classifiers [15]. They map their input into a more compact representation, or they classify their input into classes, depending on their objective function. Convolutional layers extract particular features at different locations from their inputs. By stacking them, and downsampling their outputs, CNNs extract more complex and abstract features, being at the same time invariant to distortions and translations [26]. In multichannel time-series HAR, the CNN's input consist of a stack of segmented sequences from different sensors for a certain temporal duration. This input is a $2D$ matrix of T measurements for each of the D sensors, see Figure 1. Additionally, convolution and downsampling operations are carried out along the time axis. In this way, CNNs extract hierarchical human body

movements, i.e., from basic and simple movements to complex ones. Besides, they learn the temporal dependencies among different movements.

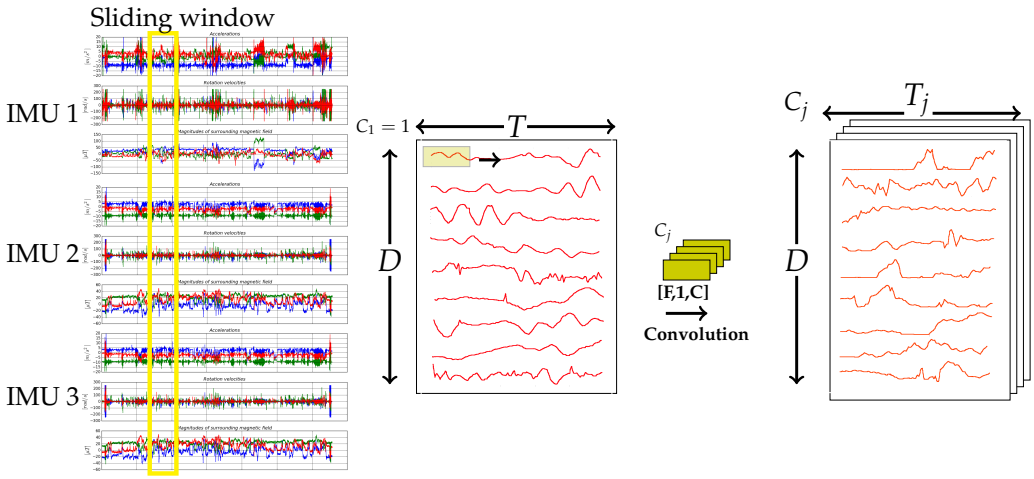

Figure 1. Example of IMUs' measurements, a CNN's input extracted following a sliding window approach, and a temporal convolution-operation in HAR.

3.1. Temporal Convolution and Pooling Operations

Having a sequence with $d = 1, 2, ..., D$ sensors, a sliding window of size T is moved forward with a frame-shift of s segmenting sequence inputs. These sequence inputs are then of size $[T, D]$. Using a small s, multiple windows representing the activity are extracted. Although the information in these is highly redundant, the small frame-shift allows to generate a large number of samples, which is important for training a CNN [4,11]. In CNNs, convolutional layers convolve their feature-map inputs with C filters along the temporal axis. Having a feature-map x^i of size $[T, D, C]$ in layer i, and a set of $c_j \in C_j$ filters w^{j,c_j} of size $[F, 1, C_i]$ and biases b^{c_j} connecting layers i and j, a temporal-convolution for each d sensor is:

$$x^j_{t,d,c_j} = \sigma\left(\sum_{c=0}^{C_i}\sum_{f=0}^{F-1} w^{c_j}_{f,1,c} \cdot x^i_{t+f,d,c} + b^{c_j}\right) \qquad \forall d = 1, ..., D \tag{1}$$

where σ is the activation function. The filters w^j are shared among all the D sensors. Figures 1 and 2 depict the temporal-convolution operation for the input and different layers of a CNN.

Pooling operations reduce the size of a feature map along the temporal axis inducing a temporal robustness. A max-pooling operation between layer i and j for a single channel c finds the maximum among a set of p values, and is given by:

$$x^{(j)}_{t,d,c_j} = \max_{0<p\leq P}\left(x^i_{t+p,d,c_i}\right) \quad \forall d = 1, ..., D \tag{2}$$

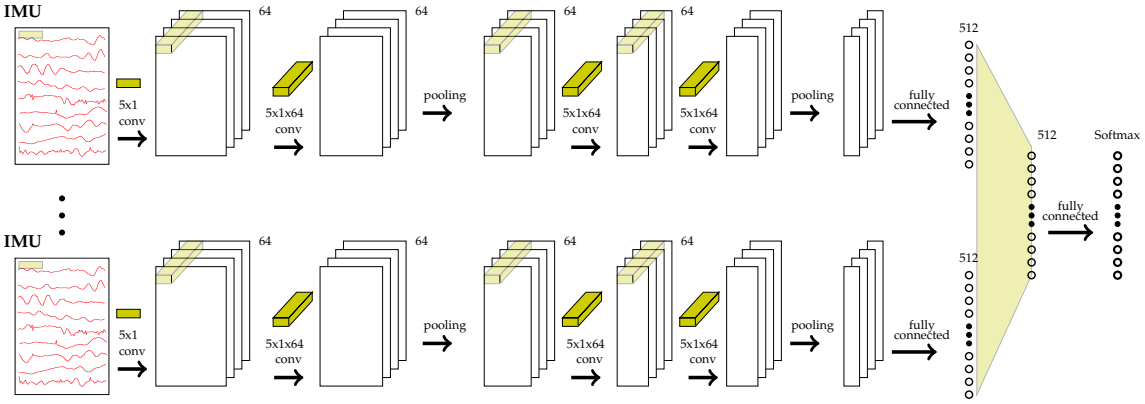

Figure 2. The CNN-IMU architecture contains m parallel branches, one per IMU. Branches are composed of B blocks, each with two $[5 \times 1]$ temporal convolutions and a $[2 \times 1]$ max-pooling. The outputs of the blocks are concatenated and forwarded to a fully connected layer. The output layer is the softmax function.

3.2. Deep Architectures

We use an architecture that processes time-series of multiple IMUs separately. It is built based on a sensor setup where multiple IMUs are worn by an individual. This architecture was introduced in a previous work [11] based on the CNN and deepConvLSTM presented in [4,5,9]. It uses temporal-convolution layers for finding temporal-local features in the input data, and fully-connected layers for connecting all of these local features, creating a global representation of the data. However, this architecture contains multiple parallel processing branches, following the idea of wider rather than deeper networks. Each parallel branch has a logical meaning as it represents the data of a single IMU. In theory, this abstraction should also allow for more robustness against the IMUs being slightly asynchronous or having different characteristics [10,23]. Moreover, as these IMUs are located on different parts of the human body, branches process only signals coming from individual parts, increasing the descriptiveness [11].

This architecture is referred as CNN-IMU network. It contains parallel branches, each with temporal-convolutions, subsequent pooling operations, and an additional fully-connected layer, see Figure 2. These parallel branches process and merge input sequences from IMUs individually, creating an intermediate representation per IMU. Following [4], each sensor $d \in D$ is processed separately by a temporal convolution; that is, convolutions are carried out in the time axis, see Equation (1), and weights are shared among sensors. Each branch contains B blocks, each of them having two stacked 5×1 temporal convolutions followed by 2×1 max-pooling operations, and, eventually, concatenated by a fully-connected layer. Depending on the dataset, the number of temporal convolutions and max-pooling layers may change. Instead of scaling the network deeper, these layers are processed in parallel for each IMU, increasing the network's descriptiveness. The network combines these intermediate representations into a global one by means of a subsequent fully-connected layer. As only one activity is considered to be present at each sequence segment, a softmax activation is used for deriving pseudo-probabilities from the $k_i \in K$ class scores. Thus, for training, the cross-entropy loss between the estimated probabilities x_k^j and the target label $y_k \in Y$ is used. Dropout is applied to all fully connected layers, except the classification layer.

For comparison with the proposed CNN-IMU architecture, a CNN baseline is also evaluated following the architecture designs presented in [5,9]. This CNN has temporal convolutions over all sensor values. One can see it as a simpler CNN-IMU with a single branch for all of the IMUs' sequences. Similarly, it contains 5×1 temporal-convolutions and 2×1 max-pooling operations followed by three fully-connected layers. The number of neurons per layer is $C = 64$ and it is chosen according to [5], which is smaller than the one used in [11].

4. Materials and Methods

For the validation of the CNN-IMU architecture for HAR, we have selected three datasets: The Opportunity [27], Pamap2 [28] and the Order Picking dataset [8]. These datasets comprise multichannel time-series recordings from different number of IMUs on gestures, locomotion and industrial activities. We evaluate the performance under different changes in the architectures and training, precisely the number of branches, number of convolutional blocks, usage of max-pooling operations, base learning rates β and learning rate reductions γ.

4.1. Opportunity Dataset

The Opportunity dataset [29,30], publicly available in [27], comprises recordings of gestures and locomotion settings. In the gestures setting, the goal is to classify $K = 18$ right-arm gestures; for example, activities like opening and closing doors or drawers, and drinking coffee. In the locomotion setting, $K = 5$ classes of movements and postures of the body are recognized. Specifically, the tasks are walking, standing, sitting, and lying. Both settings contain the 'Null' class. This class covers all the human actions that are not relevant for a task. The recordings are taken from four participants performing activities of daily living (ADL). Each participant executed five ADLs sessions without a certain execution pattern. Additionally, participants carried out a drill session following a list of 17 activities. IMUs' are placed on both wrists, upper arms, feet, and back, providing recordings of different sensors. In total, the dataset provides measurements of $D = 113$ sensors from $m = 8$ IMUs. Sessions ADL3 of participants 2 and 3 are used as validation set, and sessions ADL4 and ADL5 of participants 2 and 3 as testing set, following [3,5,30]. The sample rate of the recordings is 30 Hz. A sliding window approach, with window size of 720ms or $T = 24$ and a step size of 360ms or $s = 12$, is used for segmenting the sequences.

4.2. Pamap2 Dataset

The Pamap2 dataset [31], publicly available in [28], consists of recordings taken from 9 participants carrying out locomotion activities. According to the dataset's protocol, the dataset contains $K = 12$ action classes. Specifically, the classes are *lying, sitting, standing, walking, running, cycling, nordic walking, ascending stairs, descending stairs, vacuum cleaning, and rope jumping*. We list the action classes as the dataset contains, in addition, a set of optional activities. It provides recordings from IMUs including accelerometers, gyroscope, magnetometer, and temperature sensors, and from a heart-rate monitor. It contains overall measurements of $D = 40$ sensors from $m = 4$ IMUs. The IMUs are placed on the hand, chest and ankle. Following [9], recordings from participants 5 and 6 are used as validation and testing sets respectively. IMUs' recordings are downsampled to 30 Hz, similar to the Opportunity dataset. A sliding window approach with a window size of 3 s or $T = 100$ and step size of 660 ms or $s = 22$ is used for segmenting the sequences. The window size is smaller than the one in [9] for generating a larger number of segments. The step size allows a 78% overlapping as in [9,31].

4.3. Order Picking Dataset

The Order Picking dataset consists of recordings taken from persons in a logistic scenario. In order picking, workers collect manually a list of items in a warehouse. The data consist of recordings of three persons in two different warehouses, denoted as A and B. In total, the dataset contains recordings of 10 min and 23.30 min for warehouse A and B respectively. While the proposed approach is applicable to an arbitrary number of IMUs, the number D of sensors has been restricted to not interfere with the actual work. Thus, the recording in the order picking setup has been restricted to $m = 3$ IMUs that were located at both wrists and at the torso. Each IMU captures data from an accelerometer, a gyroscope and a magnetometer with a sample rate of 100 Hz. Thus, there are, in total, $D = 3 \times 9 = 27$ sensor channels. There are seven action classes in the dataset, see Table 1, namely *walking*, *searching*, *picking* an article, *scanning*, *info*, *carrying* and *acknowledge* on a paper list. The dataset includes as

well two background classes *unknown* and a *sensor flip*, which has been used for synchronization and marking the beginning and end of an order line. The 'Null' class is not considered. Similar to the other two datasets, a sliding-window approach with window size of 1 s or $w = 100$ is used for segmenting the sequences. However, all possible windows are extracted from the sequences; that is, a step size of 10 ms or $s = 1$ is used. As a result for each event in a sequence multiple windows representing the activity are extracted. So, in general, there are 54.079 frames or approx. ≈9.01 min labeled data for warehouse A, and 99.941 frames or approx. ≈16.39 min labeled data for warehouse B. An overview is given in Table 1. The data is highly unbalanced, being the *walking* and *picking* the action classes with the highest number of samples. Besides, activities in both warehouses are disjoint, i.e., not all action classes are used in both warehouses. Workers in warehouse A use a paper list as guidance, annotated as *info* activity. They also acknowledge a picking by signing manually the list. On the other hand, workers in warehouse B use a handheld device as guidance and the *info* action represents the interaction with the handheld device. Instead of the *acknowledging* activity, workers use the handheld device for *scanning* an article.

Table 1. Overview of the Order Picking dataset and number of frames for each of the activity classes in the different parts of the dataset [11].

Activity Class	Warehouse *A*	Warehouse *B*
walking	21,465	32,904
searching	344	1776
picking	9776	33,359
scanning	0	6473
info	4156	19,602
carrying	1984	0
acknowledge	5792	0
Unknown	1388	264
flip	1900	2933

4.4. Implementation Details

The proposed CNN-IMU and the CNN baseline are implemented in the the Caffe framework [32]. Depending on the dataset, the following configurations have been employed. The parameters of the networks are updated by minimizing the categorical cross entropy using the stochastic gradient descent with the RMSProp update rule as in [5]. Sequence segments, extracted using a sliding window approach, are fed to the networks. These segments are assigned the most frequent ground truth. We used the following hyperparameters: RMS decay of 0.95, base learning rates of $\beta = [10^{-2}, 10^{-3}]$ and batch size of 100 for the Gestures and Locomotion; base learning rates of $\beta = [10^{-2}, 10^{-3}]$ and a batch size of 50 for Pamap2, and base learning rates of $\beta = [10^{-5}, 10^{-6}]$ and a batch size of 100 for the Order Picking dataset. The number of epochs varies with respect to the dataset and the networks, see Table 2. The validation costs saturate or the (CNN) overfits after a certain number of epochs for the Opportunity and Pamap2 datasets. Figure 3 show examples of the training and validation costs using the CNN baseline on the Gestures, Locomotion and Pamap2 datasets. For the Order picking dataset, having a very limited number of original samples, more epochs lead to overfitting. In general, learning rates are decreased at a certain epoch or iteration during training. We present the performance of the architectures when learning rates were not decreased and when they were decreased with $\gamma = 0.1$, once at half the epochs, and twice after one-third of epochs in training. Additionally, we used dropout with probability of 50% on the inputs of the first and second fully-connected layer, and orthogonal initialization [5,33]. As suggested in [5,11], input sequences were normalized per sensor to the range $[0, 1]$. Moreover, a Gaussian noise of $\mu = 0$ and $\sigma = 0.01$ is added, simulating sensors' inaccuracies.

Table 2. Number of epochs for training the architectures on the four datasets.

Dataset	Epochs	
	Baseline CNN	CNN-IMU
Gestures	12	8
Locomotion	12	12
Pamap2	12	12
Order Picking	20	20

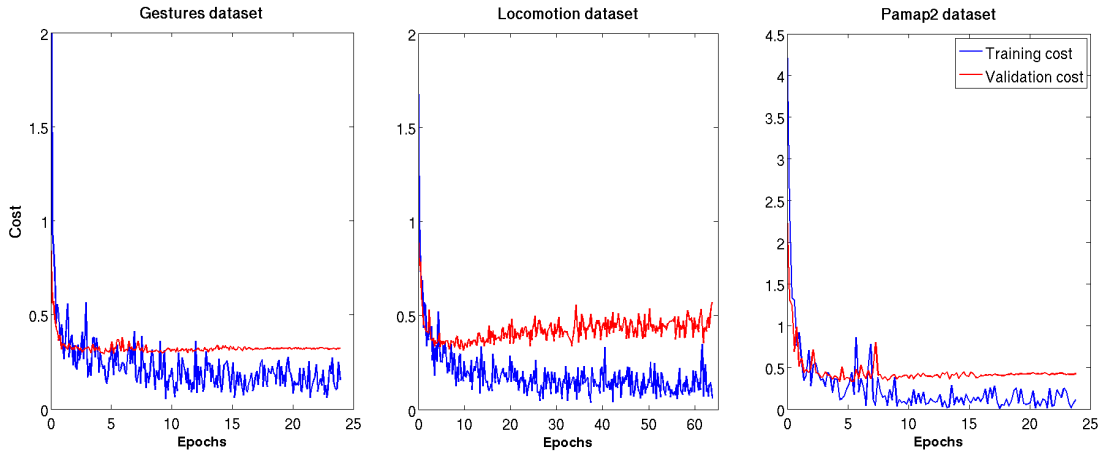

Figure 3. Training and Validation costs vs. epochs for the Opportunity-Gestures, -Locomotion and Pamap2 dataset using the CNN baseline.

As the evaluation metric, we report the the classification accuracy and the weighted F1 (wF_1). The weighted F1 (wF_1) considers the correct classification of each class equally, using the precision, recall and the proportion of class in the dataset:

$$wF_1 = \sum_i 2 \times \frac{n_i}{N} \times \frac{precision_i \times recall_i}{precision_i + recall_i}, \tag{3}$$

here, n_i is the number of segments for each class $k \in K$ and N is the total number of segments in the dataset.

Compared with the classification accuracy, the weighted F1 is more convenient for evaluating the performance of the networks on highly unbalanced datasets.

5. Results and Discussion

We evaluated two CNN-IMU architectures. A CNN-IMU containing $B = 2$ convolutional blocks per branch. A block consists of a stacking of two convolutional, a max-pooling, and at last a fully-connected layer. This architecture is refer as CNN-IMU-2, see Figure 2. The second CNN-IMU contains four convolutional layers and no max-pooling operations, following [5] as datasets contain short sequence segments and following [34] as max-pooling operations might not preserve information. In addition, two versions of the baseline CNN are evaluated: a baseline CNN with four convolutional layers and no max-pooling, as the architecture proposed in [5], and an additional baseline CNN, called CNN-2, including max-pooling layers, creating two blocks, similarly as CNN-IMU-2. All the networks contain $C = 64$ units per convolutional layer and $C = 512$ per fully connected layer, see Figure 2.

5.1. Opportunity-Gestures

We present the classification accuracy and the weighted F1 for the different architectures on the Opportunity-Gestures validation sets. The CNN-IMU has $m = 8$ branches, due to the number of IMUs. Table 3 shows the performances of the networks with max pooling, namely CNN-2 and CNN-IMU-2, and without max-pooling, the CNN and CNN-IMU architectures. For this datasets, max-pooling operations in CNN-2 and CNN-IMU-2 are carried out with a stride of 1, as the sequences are short $T = 24$. We used a randomization test for stating whether a certain performance is significant or not [35] (A performance improvement is considered significant if the p-value is less than 5.0×10^{-2} and highly significant if the p-value is less than 1.0×10^{-2}). The randomization tests are carried out comparing the classification accuracies between the architectures with and without max-pooling and the architecture with the best performance. In Table 3, the CNN-IMU without max-pooling trained with $\beta = 10^{-4}$ presents the best accuracy. For the randomization tests, one used the 59364 testing samples. As we are comparing the accuracies against the best one, the significant values will denote significant lower performances. These values are presented in *italic* in the table. Contrary to [5,34], using max-pooling layers do not influence negatively the networks' performances, even for short sequence inputs.

Table 3. Validation accuracies and Weighted F1 for architectures with and without max-pooling operations on the Opportunity-Gestures dataset. CNN & CNN-2 architectures with $\beta = 10^{-4}$ overfit to the "Null" class with zero predictions for the other action classes.

Architecture	**Max-Pooling**	**Acc**[%]	$\boldsymbol{wF_1}$[%]
$\beta = 10^{-3}$			
baseline CNN [5]	No	*90.96*	*91.11*
CNN-2	Yes	*91.49*	*91.52*
CNN-IMU	No	92.13	91.88
CNN-IMU-2	Yes	*91.80*	*91.57*
Architecture	**Max-Pooling**	**Acc**[%]	$\boldsymbol{wF_1}$[%]
$\beta = 10^{-4}$			
baseline CNN [5]	No	*83.43*	*75.89*
CNN-2	Yes	*83.43*	*75.89*
CNN-IMU	No	**92.24**	**92.01**
CNN-IMU-2	Yes	92.13	92.00

Table 4 presents the performances of the four architectures: the baseline CNN and CNN-IMU (without max-poolings); the CNN-2 and CNN-IMU-2 with two convolutional blocks CNN, when training using two different learning rates. Similarly to [5,9], the architectures with only one branch show better results when using a $\beta = 10^{-3}$ than a $\beta = 10^{-4}$. When training with this learning rate, both CNN networks seem overfitting to class "Null". Both CNN-IMU architectures present the best results when training using a $\beta = 10^{-4}$. Table 5 shows the extension of the evaluation for both learning rates when they are decreased by $\gamma = 0.1$ once and twice during training. Randomization tests were carried out per network and per β taking the best performance as the reference. Architectures benefit when the $\beta = 10^{-3}$ and it is decreased during training, contrary to using a smaller base $\beta = 10^{-4}$.

Table 4. Validation accuracies and Weighted F1 for the architectures on the Opportunity-Gestures dataset using two learning rates during training. CNN & CNN-2 architectures with $\gamma = 10^{-4}$ overfit to the Null class with zero predictions of the other action classes.

Architecture	$\beta = 10^{-3}$		$\beta = 10^{-4}$	
	Acc[%]	wF_1[%]	**Acc**[%]	wF_1[%]
baseline CNN [5]	90.96	91.11	*83.43*	*75.89*
CNN-2	91.49	91.52	*83.43*	*75.89*
CNN-IMU	92.13	91.88	**92.24**	**92.01**
CNN-IMU-2	*91.8*	*91.57*	92.13	92.0

Table 5. Validation accuracies and Weighted F1 for the architectures on the Opportunity-Gestures dataset when reductions of the learning rate at training are used. CNN & CNN-2 architectures with $\gamma = 10^{-4}$ overfits to the Null class with zero predictions of the other action classes.

Architecture	$\gamma = 0.1$					
	No Decrease		@ $\frac{epochs}{2}$		@ $\frac{epochs}{3}$	
$\beta = 10^{-3}$	**Acc**[%]	wF_1[%]	**Acc**[%]	wF_1[%]	**Acc**[%]	wF_1[%]
baseline CNN [5]	*90.96*	*91.11*	91.68	91.94	91.72	91.88
CNN-2	91.49	91.52	**91.51**	**91.62**	91.22	91.45
CNN-IMU	*92.13*	*91.88*	92.46	92.38	92.22	92.14
CNN-IMU-2	*91.80*	*91.57*	92.40	92.29	**92.44**	**92.31**
Architecture	$\gamma = 0.1$					
	No Decrease		@ $\frac{epochs}{2}$		@ $\frac{epochs}{3}$	
$\beta = 10^{-4}$	**Acc**[%]	wF_1[%]	**Acc**[%]	wF_1[%]	**Acc**[%]	wF_1[%]
baseline CNN [5]	83.43	75.89	-	-	-	-
CNN-2	83.43	75.89	-	-	-	-
CNN-IMU	**92.24**	**92.01**	*91.52*	*91.38*	*91.25*	*90.81*
CNN-IMU-2	92.13	92.00	*91.64*	*91.29*	*91.31*	*90.93*

5.2. Opportunity-Locomotion

Likewise, Tables 6–8 show the performances for the different architectures on the Opportunity-Locomotion dataset. The CNN-IMUs have $m = 8$ branches. Randomization tests were carried out comparing the classification accuracies per network and β and the best performances as references using the 9894 testing samples. Contrary to the results on the Gestures, the networks with only one branch and two blocks present the best performances. Significant inferior accuracies are shown in italic. In general, the CNN's performances are superior than the CNN-IMUs' ones. Moreover, reducing the learning rate during training presents an unfavorable effect. Experiments using a low learning rate $\beta = 10^{-4}$ with no reductions show a significant performance' improving. Following the results on the Gestures, using the max-pooling layers does not affect positive nor negative the performance of the networks.

Table 6. Validation accuracies and Weighted F1 for the architectures with and without max-pooling operations on the Opportunity-Locomotion dataset.

Architecture	**Max-Pooling**	**Acc**[%]	wF_1[%]
$\beta = 10^{-3}$			
baseline CNN [5]	No	*88.05*	*88.02*
CNN-2	Yes	**89.66**	**89.57**
CNN-IMU	No	88.61	88.61
CNN-IMU-2	Yes	*88.11*	*88.05*
Architecture	**Max-Pooling**	**Acc**[%]	wF_1[%]
$\beta = 10^{-4}$			
baseline CNN [5]	No	**89.78**	**89.67**
CNN-2	Yes	*89.68*	*89.56*
CNN-IMU	No	88.76	88.59
CNN-IMU-2	Yes	*88.67*	*88.53*

Table 7. Validation accuracies and Weighted F1 for the architectures on the Opportunity-Locomotion dataset using two learning rates during training.

Architecture	$\beta = 10^{-3}$		$\beta = 10^{-4}$	
	Acc[%]	wF_1[%]	**Acc**[%]	wF_1[%]
baseline CNN [5]	*88.05*	*88.02*	**89.78**	**89.67**
CNN-2	**89.66**	**89.57**	89.68	89.56
CNN-IMU	88.61	88.61	88.76	88.59
CNN-IMU-2	88.11	88.05	88.67	88.53

Table 8. Validation accuracies and Weighted F1 for the architectures on the Opportunity-Locomotion dataset when reductions of the learning rate during training are used.

Architecture	$\gamma = 0.1$					
	No Decrease		@ $\frac{epochs}{2}$		@ $\frac{epochs}{3}$	
$\beta = 10^{-3}$	**Acc**[%]	wF_1[%]	**Acc**[%]	wF_1[%]	**Acc**[%]	wF_1[%]
baseline CNN [5]	88.05	88.02	88.06	87.92	*86.57*	*86.40*
CNN-2	**89.66**	**89.57**	*88.39*	*88.25*	89.29	88.17
CNN-IMU	88.61	88.61	88.48	88.39	*87.45*	*87.31*
CNN-IMU-2	88.11	88.05	88.65	88.55	*87.62*	*87.55*
Architecture	$\gamma = 0.1$					
	No Decrease		@ $\frac{epochs}{2}$		@ $\frac{epochs}{3}$	
$\beta = 10^{-4}$	**Acc**[%]	wF_1[%]	**Acc**[%]	wF_1[%]	**Acc**[%]	wF_1[%]
baseline CNN [5]	**89.78**	**89.67**	*87.14*	*89.02*	*88.19*	*88.07*
CNN-2	89.68	89.56	89.45	89.35	*87.75*	*87.67*
CNN-IMU	88.76	88.59	88.18	88.05	*87.50*	*87.35*
CNN-IMU-2	88.67	88.53	*87.76*	*87.59*	*86.85*	*86.69*

5.3. Pamap2

In Tables 9–11, the performances of the networks on the Pamap2 dataset are shown. The CNN-IMU architectures contain $m = 4$ branches. As the dataset consists of longer sequences than the Opportunity' ones, we also add an additional convolutional block to the networks; so that, the networks will have in total six convolutional layers and three max-pooling. We present then two more architectures; a CNN with three convolutional blocks, called CNN-3 and a CNN-IMU with three convolutional blocks, called CNN-IMU-3. Randomization tests were performed similar to the Opportunity ones with 3785 testing samples. Mixed results can be found, both configurations CNN and CNN-IMU

show the best performances starting from a base learning rate of 10^{-4}. The performances of the CNN and CNN-IMU improves adding a third convolutional block without learning rate reduction during training. However, the architectures with only one branch, the CNN-2 and CNN-3, show the best performances. In general for the performances on the three datasets and contrary to [34], CNNs benefit from max-pooling operations, even when sequences are short.

Table 9. Validation accuracies and Weighted F1 for the architectures with and without max-pooling operations on the Pamap2 dataset.

Architecture	**Max-Pooling**	**Acc**[%]	wF_1[%]
$\beta = 10^{-3}$			
baseline CNN [5]	No	*89.90*	*89.60*
CNN-2	Yes	**92.55**	**92.60**
CNN-IMU	No	90.12	89.94
CNN-IMU-2	Yes	90.78	90.76
Architecture	**Max-Pooling**	**Acc**[%]	wF_1[%]
$\beta = 10^{-4}$			
baseline CNN [5]	No	*84.75*	*84.99*
CNN-2	Yes	91.15	91.22
CNN-IMU	No	90.22	89.94
CNN-IMU-2	Yes	**91.22**	**91.25**

Table 10. Validation accuracies and Weighted F1 for the architectures on the Pamap2 dataset using two learning rates during training.

Architecture	$\beta = 10^{-3}$		$\beta = 10^{-4}$	
	Acc[%]	wF_1[%]	**Acc**[%]	wF_1[%]
baseline CNN [5]	89.90	89.60	*84.75*	*84.99*
CNN-2	**92.55**	**92.60**	*91.15*	*91.22*
CNN-3	92.54	92.62	93.0	93.15
CNN-IMU	90.12	89.94	90.22	89.84
CNN-IMU-2	90.78	90.76	91.22	91.25
CNN-IMU-3	*91.30*	*91.53*	92.52	92.62

Table 11. Validation accuracies and Weighted F1 for the architectures on the Pamap2 dataset when reductions of the learning rate during training are used.

Architecture	$\gamma = 0.1$					
	No Decrease		@ $\frac{epochs}{2}$		@ $\frac{epochs}{3}$	
$\beta = 10^{-3}$	**Acc**[%]	wF_1[%]	**Acc**[%]	wF_1[%]	**Acc**[%]	wF_1[%]
baseline CNN [5]	89.90	89.60	89.87	89.93	90.17	90.11
CNN-2	**92.55**	**92.60**	91.75	91.67	91.81	83.52
CNN-3	92.54	92.62	91.36	91.33	91.70	91.72
CNN-IMU	*90.12*	*89.94*	91.70	91.68	91.22	91.23
CNN-IMU-2	*90.78*	*90.76*	*88.27*	*88.84*	92.81	96.01
CNN-IMU-3	91.30	91.53	91.89	91.93	*89.93*	*89.95*
Architecture	$\gamma = 0.1$					
	No Decrease		@ $\frac{epochs}{2}$		@ $\frac{epochs}{3}$	
$\beta = 10^{-4}$	**Acc**[%]	wF_1[%]	**Acc**[%]	wF_1[%]	**Acc**[%]	wF_1[%]
baseline CNN [5]	*84.75*	*84.99*	89.27	89.13	88.35	88.18
CNN-2	91.15	91.22	*89.77*	*89.62*	*88.64*	*88.52*
CNN-3	93.0	93.15	92.23	92.10	91.81	91.49
CNN-IMU	90.22	89.84	89.72	89.58	90.06	90.09
CNN-IMU-2	*91.22*	*91.25*	91.94	92.04	**93.13**	**93.21**
CNN-IMU-3	92.52	92.62	*90.61*	*90.64*	92.92	92.99

5.4. Order Picking

Similarly, we evaluated the CNN-IMUs and the baseline CNN on the Order Picking Dataset. We follow a 3-fold validation as the dataset is scarce, in which one person is taken as the testing set and the other two as the training set. Tables 12 and 13 show the results for the warehouse A and B for each of the persons taken as the testing set. Tables present the CNN architectures: the *baseline CNN* from [5], the proposed *CNN-IMU* and *CNN-IMU*-2 architectures. There are $m = 3$ branches in the CNN-IMU architectures. The networks present overfitting to the "walking" action class when using $\beta = [10^{-3}, 10^{-4}]$. For this reason smaller βs were used. Randomization tests were performed similar to the previous experiments ones, comparing the classification accuracies with the best one. Thus, the significant values denote lower performances and are presented in *italic* in the tables. For these tests, we used $[12,567, 17,436, 16,799]$ samples for each of the three persons as test set. For the warehouse A, see Table 12, the CNN-IMU architectures obtain significantly better performances. Computing temporal dependencies and obtaining a feature representation per IMU show a favorable behaviour. The CNN-IMU-2, which contain max-pooling layers, show an advantage with respect to the CNN-IMU.

Table 12. Accuracies [%] and Weighted F1 [%] for warehouse A using CNN architectures.

Architecture		Person 1		Person 2		Person 3		Warehouse A	
		Acc	wF_1[%]	**Acc**	wF_1[%]	**Acc**	wF_1[%]	**Acc**	wF_1[%]
Baseline CNN	$\gamma = 10^{-5}$	*64.69*	*65.23*	*70.2*	*67.36*	*65.62*	*60.72*	66.84 ± 2.95	64.44 ± 3.39
Baseline CNN	$\gamma = 10^{-6}$	*63.72*	*62.71*	*69.81*	*65.32*	*66.34*	*61.5*	66.62 ± 3.06	63.18 ± 1.95
CNN-2	$\gamma = 10^{-5}$	*66.66*	*63.58*	*73.33*	*59.15*	*68.05*	*63.61*	69.35 ± 3.52	65.45 ± 3.21
CNN-2	$\gamma = 10^{-6}$	68.2	65.77	*69.53*	*66.46*	*67.74*	*61.97*	68.49 ± 0.93	64,73 ± 2.42
CNN-IMU	$\gamma = 10^{-5}$	*66.48*	*64.54*	*70.22*	*67.26*	*66.8*	*62.25*	67.83 ± 2.07	64.68 ± 2.51
CNN-IMU	$\gamma = 10^{-6}$	*67.36*	*65.20*	*73.34*	*68.67*	*67.24*	*63.22*	69.31 ± 3.49	65.22 ± 2.76
CNN-IMU-2	$\gamma = 10^{-5}$	**68.34**	**66.23**	**74.39**	**69.09**	69.68	63.97	**70.80 ± 3.18**	66.43 ± 2.56
CNN-IMU-2	$\gamma = 10^{-6}$	67.43	68.04	*72.05*	*69.69*	**70.60**	**66.20**	70.03 ± 2.36	67.97 ± 1.75

For the warehouse B, see Table 13, results are more diverse. The best results are obtained by the CNN-IMU without max-pooling operations when using a $\beta = 10^{-6}$. It can be observed that there is a big variance between person 1 and person 2. This is mainly due to the different activities those persons carried out in which person 1 does not perform the 'searching' activity. For the randomization tests, we used $[15,158, 47,044, 35,100]$ samples for each of the three persons as test set. For both warehouses, the CNN-IMU architectures performs better than the CNNs.

Table 13. Accuracies [%] and Weighted F1 [%] for warehouse B using CNN architectures.

Architecture		Person 1		Person 2		Person 3		Warehouse A	
		Acc	wF_1[%]	**Acc**	wF_1[%]	**Acc**	wF_1[%]	**Acc**	wF_1[%]
Baseline CNN	$\gamma = 10^{-5}$	*45.98*	*36.04*	*60.78*	*53.05*	*77.15*	*75.6*	61.30 ± 15.59	54.89 ± 19.84
Baseline CNN	$\gamma = 10^{-6}$	*43.9*	*35.53*	*59.66*	*52.27*	*77.68*	*76.35*	60.41 ± 16.9	54.72 ± 20.52
CNN-2	$\gamma = 10^{-5}$	*49.39*	*43.8*	*58.57*	*52.42*	*77.56*	*76.61*	61.84 ± 13.37	57.61 ± 17.01
CNN-2	$\gamma = 10^{-6}$	*47.42*	*40.16*	**62.64**	**56.53**	*77.97*	*76.70*	62.68 ± 15.27	57.80 ± 18.30
CNN-IMU	$\gamma = 10^{-5}$	*49.79*	*46.94*	*59.21*	*51.28*	*76.38*	*75.48*	61.79 ± 13.48	57.9 ± 15.37
CNN-IMU	$\gamma = 10^{-6}$	**67.23**	**63.21**	*60.87*	*53.99*	*80.0*	*77.76*	**69.36 ± 9.74**	**64.98 ± 11.98**
CNN-IMU-2	$\gamma = 10^{-5}$	*43.06*	*52.68*	*61.11*	*68.65*	*78.32*	*79.97*	60.83 ± 17.63	67.10 ± 13.71
CNN-IMU-2	$\gamma = 10^{-6}$	*54.31*	*45.66*	*61.71*	*53.78*	**89.93**	**78.13**	68.65 ± 18.79	59.86 ± 16.12

5.5. Comparison with the State-of-the-Art

Table 14 compares the testing accuracies and Weighted F1 for best results on the Opportunity and Pamap2 datasets for each of the evaluated architectures and the reported performance in [5,9]. Randomization tests were performed per dataset comparing all architectures with the one portraying the best classification accuracies. The values that are significant lower than the best performance are

presented in *italic*. The proposed CNN-IMU's accuracies and weighted F1 on the Opportunity-Gestures and Pamap2 datasets are superior with respect to the other architectures and the benchmarking. The classification accuracies of the CNN-IMUs on the Opportunity-gestures are significant better in comparison with the baseline CNN. The CNN-IMU with $B = 2$ and $B = 3$ blocks and the CNN-3 present similar performances, which are significant better than the architectures without max-pooling operations (baseline CNN and CNN-IMU), showing the benefit of using max-pooling operations. However, the architectures with only one convolutional block, the baseline CNN and CNN-2, perform significantly better than the proposed networks on the Opportunity-locomotion dataset. As this setting contains annotations for movements and postures of the whole human body, it seems that processing measurements from individual IMUs does not provide any advantage.

Table 14. Comparison between the testing accuracies and Weighted F1 for the best performances using CNNs and state-of-the-art on the Opportunity and Pamap2 datasets. Results marked with '*' cannot be directly compared, as the authors in [10] predict action classes per sample and they do not follow a sliding window-approach.

Architecture	**Datasets**					
	Gestures		**Locomotion**		**Pamap2**	
	Acc[%]	wF_1[%]	**Acc**[%]	wF_1[%]	**Acc**[%]	wF_1[%]
baseline CNN [5]	*91.58*	*90.67*	**90.07**	**90.03**	*89.90*	*90.04*
CNN-2	*91.26*	*91.32*	89.71	89.64	91.09	90.97
CNN-3	-	-	-	-	91.94	91.82
CNN-IMU	**92.22**	**92.07**	*88.10*	*88.05*	*92.52*	*92.54*
CNN-IMU-2	91.85	91.58	*87.99*	*87.93*	**93.68**	**93.74**
CNN-IMU-3	-	-	-	-	93.53	93.62
CNN-Ordonez [5]	-	88.30	-	87.8	-	-
Hammerla [9]	-	89.40	-	-	-	87.8
DeepCNNLSTM Ordonez [5]	-	91.7	-	89.5	-	-
Dense labeling Yao [10] *	89.9	59.6	87.1	88.7	-	-

Table 15 compares the best results of the evaluated architectures with the traditional approaches' performances using statistical features, reported in [11], on the order picking dataset. In general for the warehouse A, the traditional approaches are outperformed by the CNN-IMU-2, however, not by a large margin. For example, the difference between the Bayes classifier and the CNN-IMU-2 is small for the person 3. This results are similar to different works in HAR and contrary to other recognition tasks. For the warehouse B, results are more diverse. In average, the CNN-IMU-2 performs better in comparison with the statistical features. Nonetheless, this mainly due to its significant better performance on the person 1.

Table 15. Comparison between the best performances using CNNs and statistical features on the order picking dataset for warehouses A and B.

	Architecture		**Person 1**		**Person 2**		**Person 3**		**Warehouse A**	
			Acc	wF_1[%]	**Acc**	wF_1[%]	**Acc**	wF_1[%]	**Acc**	wF_1[%]
Statistical features	Bayes		64.8	-	51.3	-	**69.9**	-	62.0 ± 7.8	-
	Random Forest		64.3	-	52.9	-	63.5	-	60.2 ± 5.2	-
	SVM linear		66.6	-	63.5	-	64.1	-	63.6 ± 2.6	-
CNN	CNN-IMU-2	$\gamma = 10^{-5}$	**68.34**	**66.23**	**74.39**	**69.09**	69.68	63.97	**70.80 ± 3.18**	66.43 ± 2.56
	Architecture		**Person 1**		**Person 2**		**Person 3**		**Warehouse B**	
			Acc	wF_1[%]	**Acc**	wF_1[%]	**Acc**	wF_1[%]	**Acc**	wF_1[%]
Statistical features	Bayes		58.0	-	62.4	-	**81.8**	-	67.4 ± 10.3	-
	Random Forest		49.5	-	**70.1**	-	79.0	-	66.2 ± 12.4	-
	SVM linear		39.7	-	62.8	-	77.2	-	59.9 ± 15.4	-
CNN	CNN-IMU	$\gamma = 10^{-6}$	**67.23**	**63.21**	60.87	53.99	80.0	77.76	**69.36 ± 9.74**	**64.98 ± 11.98**

6. Conclusions

We have evaluated a novel CNN architecture for HAR using multichannel time-series acquired from body-worn sensors, i.e., IMUs. The architecture is originally introduced in a previous work [11]. This architecture consist of parallel branches. A single branch contains temporal convolutions, max-pooling operations and a final fully-connected layer. Each of these branches finds temporal relations of time-series per IMU. They create an intermediate representation of the IMU's measurements. The architecture merges each of these representations by means of a subsequent fully-connected layer, which is then used for classification. Extending the work in [11], the proposed architecture is evaluated on two benchmark-datasets, the Opportunity with Gestures and Locomotion settings, and the Pamap2. The architecture is also deployed on a private dataset, the Order Picking dataset. In addition, we have investigated the effect of using max-pooling, as this operation might not preserve the information as suggested in [34]. For relatively long sequences, networks using max-pooling operations show better results. Furthermore, evaluations using different learning rates and their reduction during training were presented. These showed some mixed behaviours. The CNN-IMU benefits from relative small learning rate as compared to the baseline CNN. For the bigger learning rate, decreasing the learning rate during training is advantageous. The CNN-IMU and CNN-IMU-2 architectures' performances outperform the state-of-the-art on the Opportunity-gestures, Pamap2 and Order Picking datasets.

Implementation code and parameters of the CNN-IMU are available from [36].

Author Contributions: F.M.R. conceptualized and implemented the deep architectures, executed the experiments, analyzed the results, drafted the initial manuscript and revised the manuscript. R.G. conceptualized the deep architecture, provided the baseline results, provided feedback, revised the manuscript and approved the final manuscript. G.A.F. conceptualized the deep architecture, provided feedback, revised the manuscript, acquired the financial support for the project leading to this publication and approved the final manuscript. S.F. conceptualized the idea of automatically analyzing the order picking process, provided the baseline results, provided the Order Picking Dataset and approved the final manuscript. M.t.H. provided feedback to the conceptualization of automatically analyzing the order picking process and acquired the financial support for the project leading to this publication.

Acknowledgments: This work has been supported by the German Research Foundation (DFG) within project Fi799/10-1 ('Adaptive, Context-based Activity Recognition and Motion Classification to Analyze the Manual Order Picking Process').

Conflicts of Interest: The founding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

References

1. Feichtenhofer, C.; Pinz, A.; Zisserman, A. Convolutional two-stream network fusion for video action recognition. *arXiv* **2016**, arXiv:1604.06573.
2. Zeng, M.; Nguyen, L.T.; Yu, B.; Mengshoel, O.J.; Zhu, J.; Wu, P.; Zhang, J. Convolutional neural networks for human activity recognition using mobile sensors. In Proceedings of the 2014 6th International Conference on Mobile Computing, Applications and Services (MobiCASE), Austin, TX, USA, 6–7 November 2014; IEEE: New York, NY, USA, 2014; pp. 197–205.
3. Ronao, C.A.; Cho, S.B. Deep convolutional neural networks for human activity recognition with smartphone sensors. In Proceedings of the International Conference on Neural Information Processing, Istanbul, Turkey, 9–12 November 2015; Springer: New York, NY, USA, 2015; pp. 46–53.
4. Yang, J.; Nguyen, M.N.; San, P.P.; Li, X.; Krishnaswamy, S. Deep Convolutional Neural Networks on Multichannel Time Series for Human Activity Recognition. In Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI'15), Buenos Aires, Argentina, 25–31 July 2015; pp. 3995–4001.
5. Ordóñez, F.J.; Roggen, D. Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition. *Sensors* **2016**, *16*, 115. [CrossRef] [PubMed]
6. Bulling, A.; Blanke, U.; Schiele, B. A tutorial on human activity recognition using body-worn inertial sensors. *ACM Comput. Surv. (CSUR)* **2014**, *46*, 33. [CrossRef]
7. Lara, O.D.; Labrador, M.A. A Survey on Human Activity Recognition Using Wearable Sensors. *IEEE Commun. Surv. Tutor.* **2013**, *15*, 1192–1209. [CrossRef]

8. Feldhorst, S.; Masoudenijad, M.; ten Hompel, M.; Fink, G.A. Motion Classification for Analyzing the Order Picking Process using Mobile Sensors. In Proceedings of the 5th International Conference on Pattern Recognition Applications and Methods, Rome, Italy, 24–26 February 2016; SCITEPRESS-Science and Technology Publications, Lda: Setúbal, Portugal, 2016; pp. 706–713.
9. Hammerla, N.Y.; Halloran, S.; Ploetz, T. Deep, convolutional, and recurrent models for human activity recognition using wearables. *arXiv* **2016**, arXiv:1604.08880.
10. Yao, R.; Lin, G.; Shi, Q.; Ranasinghe, D.C. Efficient Dense Labelling of Human Activity Sequences from Wearables using Fully Convolutional Networks. *Pattern Recogn.* **2018**, *78*, 252–266. [CrossRef]
11. Grzeszick, R.; Lenk, J.M.; Moya Rueda, F.; Fink, G.A.; Feldhorst, S.; ten Hompel, M. Deep Neural Network based Human Activity Recognition for the Order Picking Process. In Proceedings of the 4th International Workshop on Sensor-based Activity Recognition and Interaction, Rostock, Germany, 21–22 September 2017; ACM: New York, NY, USA, 2017.
12. Roggen, D.; Cuspinera, L.P.; Pombo, G.; Ali, F.; Nguyen-Dinh, L.V. Limited-memory warping LCSS for real-time low-power pattern recognition in wireless nodes. In Proceedings of the European Conference on Wireless Sensor Networks, Porto, Portugal, 9–11 February 2015; Springer: New York, NY, USA, 2015; pp. 151–167.
13. Ordonez, F.J.; Englebienne, G.; De Toledo, P.; Van Kasteren, T.; Sanchis, A.; Krose, B. In-home activity recognition: Bayesian inference for hidden Markov models. *IEEE Pervasive Comput.* **2014**, *13*, 67–75. [CrossRef]
14. Huynh, T.; Schiele, B. Analyzing features for activity recognition. In Proceedings of the 2005 Joint Conference on Smart Objects and Ambient Intelligence: Innovative Context-Aware Services: Usages and Technologies, Grenoble, France, 12–14 October 2005; ACM: New York, NY, USA, 2005; pp. 159–163.
15. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. Proceedings of the 25th International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
16. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
17. Badrinarayanan, V.; Handa, A.; Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling. *arXiv* **2015**, arXiv:1505.07293.
18. Almazán, J.; Gordo, A.; Fornés, A.; Valveny, E. Word spotting and recognition with embedded attributes. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 2552–2566. [CrossRef] [PubMed]
19. Sudholt, S.; Fink, G.A. PHOCNet: A Deep Convolutional Neural Network for Word Spotting in Handwritten Documents. In Proceedings of the International Conference on Frontiers in Handwriting Recognition, Shenzhen, China, 23–26 October 2016.
20. Wu, C.; Ng, R.W.; Torralba, O.S.; Hain, T. Analysing acoustic model changes for active learning in automatic speech recognition. In Proceedings of the 2017 International Conference on Systems, Signals and Image Processing (IWSSIP), Poznan, Poland, 22–24 May 2017; IEEE: New York, NY, USA, 2017; pp. 1–5.
21. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
22. Graves, A. Generating sequences with recurrent neural networks. *arXiv* **2013**, arXiv:1308.0850.
23. Ravi, D.; Wong, C.; Lo, B.; Yang, G.Z. Deep learning for human activity recognition: A resource efficient implementation on low-power devices. In Proceedings of the 2016 IEEE 13th International Conference on Wearable and Implantable Body Sensor Networks (BSN), San Francisco, CA, USA, 14–17 June 2016; IEEE: New York, NY, USA, 2016; pp. 71–76.
24. Ravi, D.; Wong, C.; Lo, B.; Yang, G.Z. A deep learning approach to on-node sensor data analytics for mobile or wearable devices. *IEEE J. Biomed. Health Inform.* **2017**, *21*, 56–64. [CrossRef] [PubMed]
25. Fukushima, K.; Miyake, S. Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position. *Pattern Recogn.* **1982**, *15*, 455–469. [CrossRef]
26. LeCun, Y.; Kavukcuoglu, K.; Farabet, C. Convolutional networks and applications in vision. In Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS), Paris, France, 30 May–2 June 2010; IEEE: New York, NY, USA, 2010; pp. 253–256.

27. OPPORTUNITY Activity Recognition Data Set. Available online: https://archive.ics.uci.edu/ml/datasets/opportunity+activity+recognition (accessed on 6 March 2018).
28. PAMAP2 Physical Activity Monitoring Data Set. Available online: http://archive.ics.uci.edu/ml/datasets/pamap2+physical+activity+monitoring (accessed on 30 November 2017).
29. Roggen, D.; Calatroni, A.; Rossi, M.; Holleczek, T.; Förster, K.; Tröster, G.; Lukowicz, P.; Bannach, D.; Pirkl, G.; Ferscha, A.; et al. Collecting complex activity datasets in highly rich networked sensor environments. In Proceedings of the 2010 Seventh International Conference on Networked Sensing Systems (INSS), Kassel, Germany, 15–18 June 2010; IEEE: New York, NY, USA, 2010; pp. 233–240.
30. Chavarriaga, R.; Sagha, H.; Calatroni, A.; Digumarti, S.T.; Tröster, G.; Millán, J.d.R.; Roggen, D. The Opportunity challenge: A benchmark database for on-body sensor-based activity recognition. *Pattern Recogn. Lett.* **2013**, *34*, 2033–2042. [CrossRef]
31. Reiss, A.; Stricker, D. Introducing a new benchmarked dataset for activity monitoring. In Proceedings of the 2012 16th International Symposium on Wearable Computers (ISWC), Newcastle, UK, 18–22 June 2012; IEEE: New York, NY, USA, 2012; pp. 108–109.
32. Jia, Y.; Shelhamer, E.; Donahue, J.; Karayev, S.; Long, J.; Girshick, R.; Guadarrama, S.; Darrell, T. Caffe: Convolutional Architecture for Fast Feature Embedding. *arXiv* **2014**, arXiv:1408.5093.
33. Saxe, A.M.; McClelland, J.L.; Ganguli, S. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv* **2013**, arXiv:1312.6120.
34. Rippel, O.; Snoek, J.; Adams, R.P. Spectral representations for convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Montréal, QC, Canada, 7–12 December 2015; pp. 2449–2457.
35. Ojala, M.; Garriga, G.C. Permutation Tests for Studying Classifier Performance. *J. Mach. Learn. Res.* **2010**, *11*, 1833–1863.
36. Moya Rueda, F. CNN IMU. Available online: https://github.com/wilfer9008/CNN_IMU (accessed on 16 April 2018).