# Deep Convolutional Neural Networks On Multichannel Time Series For Human Activity Recognition

**Jian Bo Yang, Minh Nhut Nguyen, Phyo Phyo San, Xiao Li Li, Shonali Krishnaswamy**
Data Analytics Department, Institute for Infocomm Research, A*STAR, Singapore 138632
{yang-j,mnnguyen,sanpp,xlli,spkrishna}@i2r.a-star.edu.sg

## Abstract

This paper focuses on human activity recognition (HAR) problem, in which inputs are multichannel time series signals acquired from a set of body-worn inertial sensors and outputs are predefined human activities. In this problem, extracting effective features for identifying activities is a critical but challenging task. Most existing work relies on heuristic hand-crafted feature design and shallow feature learning architectures, which cannot find those distinguishing features to accurately classify different activities. In this paper, we propose a systematic feature learning method for HAR problem. This method adopts a deep convolutional neural networks (CNN) to automate feature learning from the raw inputs in a systematic way. Through the deep architecture, the learned features are deemed as the higher level abstract representation of low level raw time series signals. By leveraging the labelled information via supervised learning, the learned features are endowed with more discriminative power. Unified in one model, feature learning and classification are mutually enhanced. All these unique advantages of the CNN make it outperform other HAR algorithms, as verified in the experiments on the Opportunity Activity Recognition Challenge and other benchmark datasets.

## 1 Introduction

Automatically recognizing human's physical activities (a.k.a. human activity recognition or HAR) has emerged as a key problem to ubiquitous computing, human-computer interaction and human behavior analysis [Bulling *et al.*, 2014; Plätz *et al.*, 2012; Reddy *et al.*, 2010]. In this problem, human's activity is recognized based upon the signals acquired (in real time) from multiple body-worn (or body-embedded) inertial sensors. For HAR, signals acquired by on-body sensors are arguably favorable over the signals acquired by video cameras, due to the following reasons: i) on-body sensors alleviate the limitations of environment constraints and stationary settings that cameras often suffer from [Bulling *et al.*, 2014; Ji *et al.*, 2010; Le *et al.*, 2011]; ii) multiple on-body sensors allow more accurate and more effective deployment of signal acquisition on human body; iii) on-body sensors enjoy the merits on information privacy, as their acquired signals are target-specific while the signals acquired by camera may also contain the information of other nontarget subjects in the scene. In the past few years, body-worn based HAR made promising applications, *e.g.* game consoles, personal fitness training, medication intake and health monitoring. An excellent survey on this topic can be found at [Bulling *et al.*, 2014].

The key factor attributed to the success of a HAR system is to find an effective representation of the time series collected from the on-body sensors. Though considerable research efforts have been made to investigate this issue, diminishing returns occurred. Conventionally, the HAR problem is often taken as one of specific applications of time series analysis. The widely-used features in HAR include basis transform coding (*e.g.* signals with wavelet transform and Fourier transform) [Huynh and Schiele, 2005], statistics of raw signals (e.g, mean and variance of time sequences) [Bulling *et al.*, 2014] and symbolic representation [Lin *et al.*, 2003]. Although these features are widely used in many time series problems, they are heuristic and not task-dependent. It is worth noting that the HAR task has its own challenges, such as intraclass variability, interclass similarity, the NULL-class dominance, and complexness and diversity of physical activities [Bulling *et al.*, 2014]. All these challenges make it highly desirable to develop a systematical feature representation approach to effectively characterize the nature of signals relative to the activity recognition task.

Recently, deep learning has emerged as a family of learning models that aim to model high-level abstractions in data [Bengio, 2009; Deng, 2014]. In deep learning, a deep architecture with multiple layers is built up for automating feature design. Specifically, each layer in deep architecture performs a nonlinear transformation on the outputs of the previous layer, so that through the deep learning models the data are represented by a hierarchy of features from low-level to high-level. The well-known deep learning models include convolutional neural network, deep belief network and autoencoders. Depending on the usage of label information, the deep learning models can be learned in either supervised or unsupervised manner. Though deep learning models achieve remarkable results in computer vision, natural language processing, and speech recognition, it has not been fully exploited in the field

of HAR.

In this paper, we tackle the HAR problem by adapting one particular deep learning model —- the convolutional neural networks (CNN). The key attribute of the CNN is conducting different processing units (*e.g.* convolution, pooling, sigmoid/hyperbolic tangent squashing, rectifier and normalization ) alternatively. Such a variety of processing units can yield an effective representation of local salience of the signals. Then, the deep architecture allows multiple layers of these processing units to be stacked, so that this deep learning model can characterize the salience of signals in different scales. Therefore, the features extracted by the CNN are task dependent and non-handcrafted. Moreover, these features also own more discriminative power, since the CNN can be learned under the supervision of output labels. All these advantages of the CNN will be further elaborated in the following sections.

As detailed in the following sections, in the application on HAR, the convolution and pooling filters in the CNN are applied along the *temporal* dimension for each sensor, and all these feature maps for different sensors need to be unified as a common input for the neural network classifier. Therefore, a new architecture of the CNN is developed in this paper. In the experiments, we performed an extensive study on the comparison between the proposed method and the state-of-the-art methods on benchmark datasets. The results show that the proposed method is a very competitive algorithm for the HAR problems. We also investigate the efficiency of the CNN, and conclude that the CNN is fast enough for online human activity recognition.

## 2 Motivations and Related Work

It is highly desired to develop a systematical and task-dependent feature extraction approach for HAR. Though the signals collected from wearable sensors are time series, they are different from other time series like speech signals and financial signals. Specifically, in HAR, only a few parts of continuous signal stream are relevant to the concept of interest (*i.e.* human activities), and the dominant irrelevant part mostly corresponds to the Null activity. Furthermore, considering how human activity is performed in reality, we learn that every activity is a combination of several basic continuous movements. Typically, a human activity could last a few seconds in practice, and within one second a few basic movements could be involved. From the perspective of sensor signals, the basic continuous movements are more likely to correspond to the smooth signals, and the transitions among different basic continuous movements may cause significant change of signal values. These properties of signals in HAR require the feature extraction method to be effective enough to capture the nature of basic continuous movements as well as the salience of the combination of basic movements.

As such, we are motivated to build a deep architecture of a series of signal processing units for feature extraction. This deep architecture consists of multiple shallow architectures, and each shallow architecture is composed by a set of linear/nonlinear processing units on locally stationary signals. When all shallow architectures are cascaded, the salience of signals in different scales is captured. This deep architecture is not only for decomposing a large and complex problem into a series of small problems, but more importantly for obtaining specific "variance" of signals at different scales. Here, the "variances" of signals reflect the salient patterns of signals. As stated in [Bengio, 2009], what matters for generalization of a learning algorithm is the number of such "variance" of signals we wish to obtain after learning.

By contrast, the traditional features extraction methods such as basis transform coding (*e.g.* signals with wavelet transform and Fourier transform) [Huynh and Schiele, 2005], statistics of raw signals (e.g, mean and covariance of time sequences) [Bulling *et al.*, 2014] and symbolic representation [Lin *et al.*, 2003] are deemed to play a comparable role of transforming the data by one or a few of neurons in one layer of a deep learning model. Another type of deep learning models, called Deep Belief Network (DBN) [Hinton and Osindero, 2006; Le Roux and Bengio, 2008; Tieleman, 2008], was also investigated for HAR by [Plätz *et al.*, 2012]. However, this feature learning method does not employ the effective signal processing units (like convolution, pooling and rectifier) and also neglects the available label information in feature extraction. The primary use of the CNN mainly lies in 2D image [Krizhevsky *et al.*, 2012; Zeiler and Fergus, 2014], 3D videos [Ji *et al.*, 2010] and speech recognition [Deng *et al.*, 2013]. However, in this paper, we attempt to build a *new* architecture of the CNN to handle the unique challenges existed in HAR. The most related work is [Zeng *et al.*, 2014], in which a shallow CNN is used and the HAR problem is restricted to the accelerometer data.

## 3 Convolutional Neural Networks in HAR

Convolutional neural networks have great potential to identify the various salient patterns of HAR's signals. Specifically, the processing units in the lower layers obtain the local salience of the signals (to characterize the nature of each basic movement in a human activity). The processing units in the higher layers obtain the salient patterns of signals at high-level representation (to characterize the salience of a combination of several basic movements). Note that each layer may have a number of convolution or pooling operators (specified by different parameters) as described below, so multiple salient patterns learned from different aspects are jointly considered in the CNN. When these operators with the same parameters are applied on local signals (or their mapping) at different time segments, a form of translation invariance is obtained [Fukushima, 1980; Bengio, 2009; Deng, 2014]. Consequently, what matters is only the salient patterns of signals instead of their positions or scales. However, in HAR we confront with multiple channels of time series signals, in which the traditional CNN cannot be used directly. The challenges in our problem include (i) processing units in CNN need applied along temporal dimension and (ii) sharing or unifying the units in CNN among multiple sensors. In what follows, we will define the convolution and pooling operators along the temporal dimension, and then present the entire architecture of the CNN used in HAR.

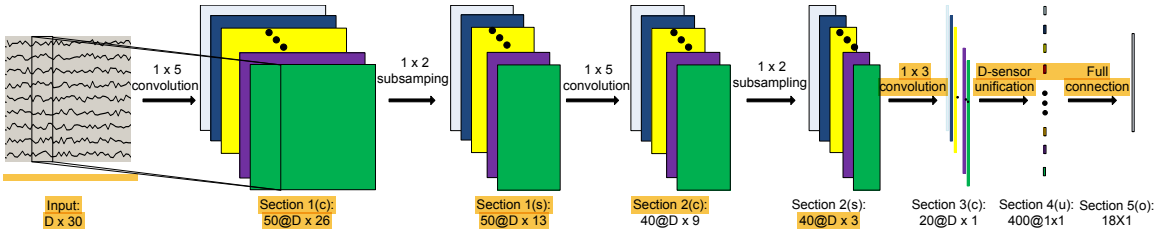


Figure 1: Illustration of the CNN architecture used for a multi-sensor based human activity recognition problems. We use the Opportunity Activity Recognition dataset presented in Section 4 as an illustrative example. The symbols "c", "s","u", "o" in the parentheses of the layer tags refer to convolution, subsampling, unification and output operations respectively. The numbers before and after "@" refer to the number of feature maps and the dimension of a feature map in this layer. Note that pooling, ReLU and normalization layers are not showed due to the limitation of space.

We start with the notations used in the CNN. A sliding window strategy is adopted to segment the time series signal into a collection of short pieces of signals. Specifically, an instance used by the CNN is a two-dimensional matrix containing $r$ raw samples (each sample with $D$ attributes). Here, $r$ is chosen to be as the sampling rate (*e.g.* 30 and 32 used in the experiments), and the step size of sliding a window is chosen to be 3. One may choose smaller step size to increase the amount of the instances while higher computational cost may be incurred. For training data, the true label of the matrix instance is determined by the most-frequently happened label for $r$ raw records. For the $j$th feature map in the $i$th layer of the CNN, it is also a matrix, and the value at the $x$th row for sensor $d$ is denoted as $v_{ij}^{x,d}$ for convenience.

### 3.1 Temporal Convolution and Pooling

In the convolution layers, the previous layer's feature maps are convolved with several convolutional kernels (to be learned in the training process). The output of the convolution operators added by a bias (to be learned) is put through the activation function to form the feature map for the next layer. Formally, the value $v_{ij}^{x,d}$ is given by

$$v_{ij}^{x,d} = \tanh\left(b_{ij} + \sum_{m}\sum_{p=0}^{P_i-1} w_{ijm}^{p} v_{(i-1)m}^{x+p,d}\right), \quad \forall d = 1, \cdots, D \tag{1}$$

where $\tanh(\cdot)$ is the hyperbolic tangent function, $b_{ij}$ is the bias for this feature map, $m$ indexes over the set of feature maps in the $(i-1)$th layer connected to the current feature map, $w_{ijm}^{p}$ is the value at the position $p$ of the convolutional kernel, and $P_i$ is the length of the convolutional kernel.

In the pooling layers, the resolution of feature maps is reduced to increase the invariance of features to distortions on the inputs. Specifically, feature maps in the previous layer are pooled over local *temporal* neighborhood by either max pooling function

$$v_{ij}^{x,d} = \max_{1\le q\le Q_i}\left(v_{(i-1)j}^{x+q,d}\right), \quad \forall d = 1, \cdots, D, \tag{2}$$

or a sum pooling function

$$v_{ij}^{x,d} = \frac{1}{Q_i}\sum_{1\le q\le Q_i}\left(v_{(i-1)j}^{x+q,d}\right), \quad \forall d = 1, \cdots, D. \tag{3}$$

where $Q_i$ is the length of the pooling region.

### 3.2 Architecture

Based on the above introduced operators, we construct a CNN shown in Figure 1. For convenience, all layers of the CNN can be grouped into five sections as detailed below.

For the first two sections, each section is constituted by (i) a convolution layer that convolves the input or the previous layer's output with a set of kernels to be learned; (ii) a rectified linear unit (ReLU) layer that maps the output of the previous layer by the function $\text{relu}(v) = \max(v, 0)$; (iii) a max pooling layer that finds the maximum feature map over a range of local *temporal* neighborhood (a subsampling operator is often involved); (iv) a normalization layer that normalizes the values of different feature maps in the previous layer $v_{ij} = v_{(i-1)j}\left(\kappa + \alpha\sum_{t\in G(j)} v_{(i-1)t}^2\right)^{-\beta}$, where $\kappa, \alpha, \beta$ are hyper-parameters and $G(j)$ is the set of feature maps involved in the normalization.

For the third section, it is only constituted by a convolution layer, ReLU layer and a normalization layer, as after the convolution layer the temporal dimension of an feature map becomes one (noting that the size of a feature map output by this layer is $D \times 1$) so the pooling layer is avoided here.

For the fourth section, we aim to unify the feature maps output by the third section among all $D$ sensors. Instead of simply concatenating these feature maps, we develop a fully connected layer to unify them to achieve the *parametric-concatenation* in this layer. An illustrative diagram is shown in Figure 2. Mathematically, the value of the $j$th feature map in this layer is computed by $v_{ij} = \tanh\left(b_{ij} + \sum_m\sum_{d=1}^{D} w_{ijm}^{d} v_{(i-1)m}^{d}\right)$, and this unification is also followed by the ReLU layer and normalization layer.

The fifth section is a fully-connected network layer. This layer is same as a standard multilayer perceptron neural network that maps the latent features into the output classes. The output of this layer is governed by the softmax function $v_{ij} = \frac{\exp(v_{(i-1)j})}{\sum_{j=1}^{C}\exp(v_{(i-1)j})}$, where $C$ is the number of output classes. This softmax function provides the posterior probability of the classification results. Then, an entropy cost function can be constituted based on the true labels of training instances and probabilistic outputs of softmax function.

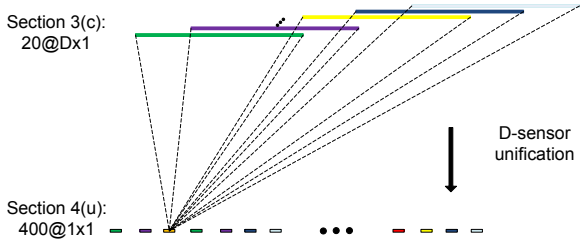


Figure 2: Illustration of unification layer (*i.e.* the fourth section in Figure 1).

To convert the matrix-level prediction given by the CNN to originally-desired sample-level predictions, the following two steps are used. First, all the samples in a matrix-level instance are labeled by the same predicted label for this matrix-level instance. Second, for a sample lying in the overlapped matrix-level instances, a voting method is used to determine the final predicted label of this sample.

Due to the temporal dependence of sensor signals, the labels of instances often have a smooth trend, as mentioned in Section 2. Recently, [Cao *et al.*, 2012] has proposed a simple but effective smoothing method to postprocess the predicted labels so as to enhance the prediction performance. The idea is to employ a low-pass filter to remove the impulse noise (potential wrong prediction) and maintain the edges, i.e., the position of activity transition. Specially, for the $i$th instance, a smoothing filter with a predefined length $u_i$ is applied on the sequence whose center is the $i$th instance. This filter finds the most frequent label in this sequence and assign it to the $i$th instance. We will investigate the prediction results with/without this smoothing method in the experiments.

### 3.3 Analysis

Note that the ReLU and normalization layers are optional in the first four layers of the CNN in Figure 1. In our experiments, we found that incorporating these two layers can lead to better results. Furthermore, to avoid the curse of dimensionality, dropout operation and regularization method might be employed in the CNN, though they are not used in our experiments due to the resultant minor performance difference.

**Remark 1.** *The conventional CNN [Krizhevsky* et al.*, 2012; Wan* et al.*, 2014; Ji* et al.*, 2010] used in the image/video case does not have the unification layer shown in Figure 2, because the image/video signal is considered to come from a single sensor channel. Thus, the proposed architecture of the CNN is a generalization of the conventional CNN by considering multiple channels of data.*

In the CNN, the parameters in all processing units and connection weights are jointly learned through a global objective function (*i.e.* entropy cost function) that is a function depending on all such unknown variables. This global objective function can be efficiently optimized by a so-called back-propagation algorithm [LeCun *et al.*, 1998].

**Remark 2.** *The global objective function is related to the training error that is computed based on the ground truth labels as well as the outputs of the softmax function in the last layer of the CNN. This function's variables control the various feature maps of the signals. Consequently, through the optimization model, the two tasks of feature learning and classification are mutually enhanced, and the learned features by the CNN have more discriminative power w.r.t. the ultimate classification task.*

## 4 Experiments

### 4.1 Datasets

We consider two datasets for human activity recognition with different focuses. The first dataset is related to the whole-body's movement while the second dataset particularly focuses on the hand's movement.

**Opportunity Activity Recognition** The Opportunity Activity Recognition dataset[1] [Sagha, 2011; Roggen *et al.*, 2010; Cao *et al.*, 2012] is about the human activities related to a breakfast scenario. This dataset contains the data collected from the sensors configured on three subjects who perform Activities of Daily Living (ADL). There are 18 classes in this activity recognition task[2]. The Null class refers to the either non-relevant activities or non-activities. The used sensors include a wide variety of body-worn, object-based, and ambient sensors - in total, 72 sensors from 10 modalities- with 15 wireless and wired sensor network in home environment. The sampling rate of the sensor signals is 30 per second. Each record is comprised of 113 real valued sensory readings excluding the time information. With these sensors, each subject performed one drill session (Drill) which has 20 repetitions of some pre-defined actions in one sequence of sensory data, and five ADLs. Following [Cao *et al.*, 2012], we use Drill and first two sets of ADLs as the training data, and use the third set of ADL as the testing data.

**Hand Gesture** The hand gesture dataset [Bulling *et al.*, 2014][3] is about different types of the human's hand movements. In this dataset, two subjects perform hand movements with eight gestures in daily living and with three gestures in playing tennis. In total, there are 12 classes in this hand gesture recognition problem[4]. Similar to the first dataset, the Null class refers to the periods with no specific activity. The used body-worn sensors include a three axis accelerometer and a two-axis gyroscope, and the sampling rate is 32 samples per second. Then, each record has 15 real valued sensor readings in total. Every subject repeated all activities about

[1]The link is http://www.opportunity-project.eu/challenge.

[2]The 18 classes are `Null, open door 1, open door 2, close door 1, close door 2, open fridge, close fridge, open dishwasher, close dishwasher, open drawer 1, close drawer 1, open drawer 2, close drawer 2, open drawer 3, close drawer 3, clean table, drink cup and toggle switch`.

[3]The link is https://github.com/andyknownasabu/ActRecTut/.

[4]The 12 classes are `Null, open a window, close a window, water a plant, turn book page, drink from a bottle, cut with a knife, chop with a knife, stir in a bowl, forehand, backhand and smash`.

| | Subject 1 | | | Subject 2 | | | Subject 3 | | |
|---|---|---|---|---|---|---|---|---|---|
| | AF | NF | AC | AF | NF | AC | AF | NF | AC |
| | Without smoothing | | | | | | | | |
| SVM (quoted from [Cao *et al.*, 2012]) | 45.6 | 83.4 | 83.8 | 44.4 | 75.6 | 79.4 | 32.1 | 76.8 | 78.1 |
| 1NN (quoted from [Cao *et al.*, 2012]) | 42.7 | 80.3 | 79.3 | 41.1 | 73.5 | 73.9 | 28.5 | 67.5 | 63.8 |
| MV | 54.2 | 83.9 | 83.7 | 50.8 | 74.6 | 74.3 | 48.6 | 80.9 | 80.7 |
| DBN | 14.3 | 75.0 | 80.0 | 7.0 | 66.7 | 74.1 | 20.0 | 73.4 | 79.3 |
| CNN | **55.5** | **86.4** | **87.0** | **57.1** | **79.5** | **82.5** | **55.8** | **84.0** | **85.8** |
| | With smoothing | | | | | | | | |
| SVM | 48.6 | 84.7 | 85.9 | 43.8 | 75.9 | 80.4 | 27.6 | 76.7 | 79.2 |
| 1NN | 53.9 | 84.1 | 84.6 | 53.2 | 78.2 | 79.8 | 34.0 | 71.8 | 69.7 |
| MV | 47.9 | 83.3 | 84.3 | 54.3 | 75.7 | 75.7 | 49.6 | 81.9 | 82.1 |
| DBN | 12.9 | 74.5 | 79.5 | 7.3 | 66.9 | 74.9 | 21.1 | 73.0 | 79.7 |
| CNN | **51.6** | **86.5** | **87.7** | **60.0** | **79.7** | **83.0** | **52.6** | **84.7** | **86.7** |

Table 1: The average F-measure (AF), normalized F-measure (NF) and Accuracy (AC) results of the proposed CNN method and four baselines for the Opportunity Activity Recognition dataset. The best result for each metric is highlighted in bold.

26 times. We randomly select one repetition as the testing data and the rest repetitions as the training data.

### 4.2 Experimental Settings

The architecture of the CNN used in Opportunity Activity Recognition dataset is shown in Figure 1. The same architecture of the CNN is used for Hand Gesture dataset with the only differences on the number of feature maps and the sizes of convolution kernels, since the dimensions of the input and output of the datasets are different. In the normalization operator of the CNN, the parameters are chosen as $\kappa = 1, \alpha = 2 \times 10^{-4}, \beta = 0.75$ and the size of $G(\cdot)$ is 5 in all experiments. We follow the rules of thumb shown in [LeCun *et al.*, 1998] to choose other parameters, as how to find the optimal parameters in CNN is still an open question.

We compare the proposed method with the following four baselines, namely SVM, KNN, MV and DBN. Among them, the first two methods and the third method show the state-of-the-arts results on the Opportunity Activity Recognition dataset and Hand Gesture datasets respectively. The fourth method is a recently-developed deep learning method for HAR.

- **SVM** [Cao *et al.*, 2012]. The support vector machine (SVM) with radial basis function (RBF) kernel is used as the classifier. In this baseline, the raw time series samples are directly used as the input of SVM. The cross validation procedure is used to tune the parameters of SVM.
- **KNN** [Cao *et al.*, 2012]. [Keogh and Kasetty, 2002] performed a comprehensive empirical evaluation on time series classification problems. Interestingly, the simple technique KNN (specifically, 1NN, *i.e.* classification based on the top one nearest neighbor) with Euclidean distance was shown to be the best technique. Therefore, we incorporate the KNN with $K = 1$ as the classifier. Same as the SVM baseline, the raw time series samples are directly used as the input of KNN.
- **Means and variance (MV)** [Bulling *et al.*, 2014] Same as the proposed CNN method, the sliding window strategy is used to generate a set of $r \times D$ matrix-level instances first. Then the mean and the variance of the signals over the $r$ samples in every $r \times D$ matrix are extracted to constitute the features of the input data for the classifier. The classifier used is the KNN with $K = 1$.
- **Deep belief network (DBN)** [Plätz *et al.*, 2012] Same as the CNN and MV methods, a set of $r \times D$ matrix-level instances are generated first. Then, the mean of the signals over the $r$ samples in every $r \times D$ matrix is used as the input of the DBN[5]. The classifier used in this method is chosen between KNN with $K = 1$ and a multilayer perceptron neural network, and the one with better performance is reported.

For MV and DBN methods, matrix-level predictions are converted to the sample-level predictions based on the same strategy used in the CNN method as introduced in Section 3.2. We evaluate all methods' performance under the both settings of with/without the smoothing method mentioned in Section 3.2. As suggested in [Cao *et al.*, 2012], the parameter $u_i$ in the smoothing method is recommended to be chosen in the range of $[60, 100]$.

### 4.3 Experimental Results

The results of the proposed CNN method and the four baseline methods on Opportunity Activity Recognition dataset and Hand Gesture dataset are shown in Table 1 and Table 2 respectively. Following [Cao *et al.*, 2012], average F-measure (AF), normalized F-measure (NF) and Accuracy (AC) are used to evaluate the performance of different methods in all experiments. The best performance for each evaluation metric is highlighted in bold.

From the results, we can see that the proposed CNN method consistently performs better than all four baselines in both settings of with/without the smoothing strategy on both datasets in terms of all three evaluation metrics. Remarkably, for Subject 3 in the first dataset and Subject 2 in the second

[5]The experiment of using the raw inputs as the features in DBN is also performed, but the results are substantially worse than the reported one.

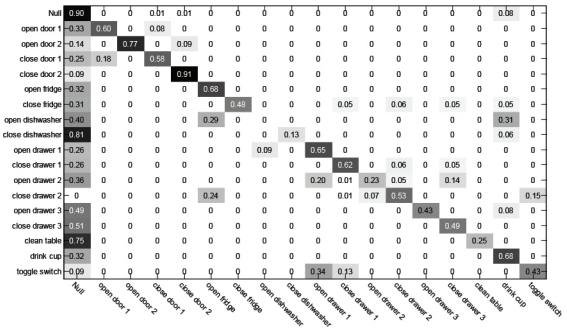


Figure 3: Confusion matrix yielded by the proposed CNN method (without temporal smoothing) on the Opportunity Activity Recognition dataset for Subject 1 (the larger the value the darker the background).

| | Subject 1 | | | Subject 2 | | |
|---|---|---|---|---|---|---|
| | AF | NF | AC | AF | NF | AC |
| | Without smoothing | | | | | |
| SVM | 76.0 | 85.0 | 85.6 | 71.1 | 83.5 | 82.6 |
| 1NN | 64.8 | 73.2 | 71.8 | 66.2 | 79.3 | 77.9 |
| MV | 87.5 | 91.3 | 91.2 | 84.1 | 90.1 | 89.3 |
| DBN | 71.8 | 82.1 | 82.8 | 69.0 | 81.4 | 80.1 |
| CNN | **89.2** | **92.0** | **92.2** | **90.7** | **95.0** | **95.0** |
| | With smoothing | | | | | |
| SVM | 85.1 | 89.2 | 89.6 | 86.0 | 89.3 | 88.5 |
| 1NN | **92.2** | 93.3 | 93.2 | 86.1 | 89.8 | 89.2 |
| MV | 91.5 | 93.3 | 93.3 | 84.4 | 90.5 | 89.6 |
| DBN | 78.5 | 84.9 | 85.8 | 73.2 | 83.4 | 82.0 |
| CNN | **92.2** | **93.9** | **94.1** | **87.0** | **95.5** | **96.0** |

Table 2: The AF, NF and AC results of the proposed CNN method and four baselines for the Hand Gesture dataset.

dataset, the proposed method outperforms the best baseline by 5% or so in terms of accuracy for both of with/without the smoothing settings. When the smoothing strategy is used, the performance of all methods generally is improved, but the performance ranking of all methods almost keeps invariant. The class imbalance issue is a main challenge for all methods. This can be seen from the confusion matrix generated by the proposed CNN method shown in Figure 3. Due to the dominant `Null` class, all signals samples, except for the ones in class `close drawer 2`, tend to be classified into the `Null` class. The similar phenomena caused by the class imbalance issue exist in all methods but they are more severe for other baseline methods.

The better performance of the CNN over DBN demonstrates that the supervised deep learning outperforms the unsupervised ones for HAR. This observation has also been seen in other applications like image classification and speech recognition. Note that SVM and KNN use the raw instances in this paper while in [Plätz *et al.*, 2012] they use the matrix-level instances whose amount is smaller than that of the raw instances. This may explain why DBN is a bit worse than SVM and KNN in our experiments while it is slightly better than SVM and KNN in the experiments shown in [Plätz *et al.*, 2012]. The evidence that the CNN has the better performance than SVM, KNN and MV suggests that the CNN is more close to find the nature of signals in feature representation than the methods with shallowing learning architecture and heuristic feature design for the HAR problems.

We also conducted the experiments that the magnitudes of Fourier Transform of the raw data are taken as the inputs for all methods. However, no performance improvement can be made. The similar observation has been observed in [Cao *et al.*, 2012].

All experiments are conducted on nonoptimized Matlab codes on a PC, which has an Intel i5-2500 3.30 GHz CPU and 8 GB RAM. Due to the space limitation, we report the timing results of the CNN on the Opportunity Activity Recognition dataset for Subject 1 as this dataset is the largest one in all experiments. The training and testing raw samples for this dataset are 136,869 and 32,466 respectively, and the input dimension is 107. The training time of the CNN is around 1 hour, while the testing time is 8 minutes. On average, within a second the CNN can predict 56 raw instances' labels. Thus, the efficiency of the CNN is good enough for the online HAR. Note that the training and testing time can be significantly reduced when the parallel computation of the CNN [Jia *et al.*, 2014; Donahue *et al.*, 2014] is implemented. This research topic will be fully investigated in our future work.

## 5 Conclusions

In this paper, we proposed a new method to automate feature extraction for the human activity recognition task. The proposed method builds a new deep architecture for the CNN to investigate the multichannel time series data. This deep architecture mainly employs the convolution and pooling operations to capture the salient patterns of the sensor signals at different time scales. All identified salient patterns are systematically unified among multiple channels and finally mapped into the different classes of human activities. The key advantages of the proposed method are: i) feature extraction is performed in task dependent and non hand-crafted manners; ii) extracted features have more discriminative power w.r.t. the classes of human activities; iii) feature extraction and classification are unified in one model so their performances are mutually enhanced. In the experiments, we demonstrate that the proposed CNN method outperforms other state-of-the-art methods, and we therefore believe that the proposed method can serve as a competitive tool of feature learning and classification for the HAR problems.

## References

[Bengio, 2009] Yoshua Bengio. Learning deep architectures for AI. *Found. Trends Mach. Learn.*, 2(1):1–127, January 2009.

[Bulling *et al.*, 2014] Andreas Bulling, Ulf Blanke, and Bernt Schiele. A tutorial on human activity recognition using body-worn inertial sensors. *ACM Computing Surveys.*, 46(3):33:1–33:33, 2014.

[Cao *et al.*, 2012] Hong Cao, Minh Nhut Nguyen, Clifton Phua, Shonali Krishnaswamy, and Xiao Li Li. An integrated framework for human activity classific. In *ACM International Conference on Ubiquitous Computing*, 2012.

[Deng *et al.*, 2013] Li Deng, Jinyu Li, Jui-Ting Huang, Kaisheng Yao, Dong Yu, Frank Seide, Michael Seltzer, Geoff Zweig, Xiaodong He, Jason Williams, Yifan Gong, and Alex Acero. Recent advances in deep learning for speech research at microsoft. ICASSP, 2013.

[Deng, 2014] Li Deng. A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Transactions on Signal and Information Processing*, 2014.

[Donahue *et al.*, 2014] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014.

[Fukushima, 1980] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, 1980.

[Hinton and Osindero, 2006] Geoffrey E. Hinton and Simon Osindero. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[Huynh and Schiele, 2005] Tâm Huynh and Bernt Schiele. 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*, 2005.

[Ji *et al.*, 2010] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. In *ICML*, 2010.

[Jia *et al.*, 2014] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM MM*, 2014.

[Keogh and Kasetty, 2002] Eamonn Keogh and Shruti Kasetty. On the need for time series data mining benchmarks: A survey and empirical demonstration. In *SIGKDD*, 2002.

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[Le *et al.*, 2011] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. CVPR, 2011.

[Le Roux and Bengio, 2008] Nicolas Le Roux and Yoshua Bengio. Representational power of restricted Boltzmann machines and deep belief networks. *Neural Computation*, 20(6):1631–1649, 2008.

[LeCun *et al.*, 1998] Y. LeCun, L. Bottou, G. Orr, and K. Muller. Efficient backprop. *Neural Networks: Tricks of the trade*, pages 9 – 50, 1998.

[Lin *et al.*, 2003] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2003.

[Plätz *et al.*, 2012] Thomas Plätz, Nils Y. Hammerla, and Patrick Olivier. Feature learning for activity recognition in ubiquitous computing. In *IJCAI*, 2012.

[Reddy *et al.*, 2010] Sasank Reddy, Min Mun, Jeff Burke, Deborah Estrin, Mark Hansen, and Mani Srivastava. Using mobile phones to determine transportation modes. *ACM Transactions on Sensor Networks*, 6(2):13:1–13:27, March 2010.

[Roggen *et al.*, 2010] Daniel Roggen, Alberto Calatroni, Mirco Rossi, Thomas Holleczek, Kilian Förster, Gerhard Tröster, Paul Lukowicz, David Bannach, Gerald Pirkl, Alois Ferscha, Jakob Doppler, Clemens Holzmann, Marc Kurz, Gerald Holl, Ricardo Chavarriaga, Marco Creatura, and José del R. Milln. Collecting complex activity data sets in highly rich networked sensor environments. In *Proceedings of the Seventh International Conference on Networked Sensing Systems (INSS), Kassel, Germany*, 2010.

[Sagha, 2011] Hesam *et al.* Sagha. Benchmarking classification techniques using the opportunity human activity dataset. In *IEEE International Conference on Systems, Man, and Cybernetics,*, 2011.

[Tieleman, 2008] Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *ICML*, 2008.

[Wan *et al.*, 2014] Ji Wan, Dayong Wang, Steven Chu Hong Hoi, Pengcheng Wu, Jianke Zhu, Yongdong Zhang, and Jintao Li. Deep learning for content-based image retrieval: A comprehensive study. In *ACM MM*, 2014.

[Zeiler and Fergus, 2014] Matt Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. ECCV, 2014.

[Zeng *et al.*, 2014] Ming Zeng, Le T. Nguyen, Bo Yu, Ole J. Mengshoel, Jiang Zhu, Pang Wu, and Joy Zhang. Convolutional neural networks for human activity recognition using mobile sensors. In *MobiCASE*, 2014.