Sensor Data for Human Activity Recognition: Feature Representation and Benchmarking

Flávia Alves*, Martin Gairing*, Frans A. Oliehoek†, Thanh-Toan Do*
**Department of Computer Science, University of Liverpool*, Liverpool, United Kingdom
{F.Alves, gairing, Thanh-Toan.Do}@liverpool.ac.uk
†*Department of Intelligent Systems, Delft University of Technology*, Delft, Netherlands
f.a.oliehoek@tudelft.nl

Abstract—**The field of Human Activity Recognition (HAR) focuses on obtaining and analysing data captured from monitoring devices (e.g. sensors). There is a wide range of applications within the field; for instance, assisted living, security surveillance, and intelligent transportation. In HAR, the development of Activity Recognition models is dependent upon the data captured by these devices and the methods used to analyse them, which directly affect performance metrics. In this work, we address the issue of accurately recognising human activities using different Machine Learning (ML) techniques. We propose a new feature representation based on consecutive occurring observations and compare it against previously used feature representations using a wide range of classification methods. Experimental results demonstrate that techniques based on the proposed representation outperform the baselines and a better accuracy was achieved for both highly and less frequent actions. We also investigate how the addition of further features and their pre-processing techniques affect performance results leading to state-of-the-art accuracy on a Human Activity Recognition dataset.**

Index Terms—**Machine Learning, Supervised learning, Neural networks, Human Activity Recognition**

I. Introduction

Over the past fifteen years, extensive research has been carried out in the field of Human Activity Recognition [1]. This has been largely motivated by the technological advancement in monitoring devices within several research areas. One example where this applies is the improvement of services in elderly care. As discussed in [2], any form of traditional methodology (e.g. in-person visits and telephone interviews) has its inherent limitations and a 24-hour continuous monitoring contributes towards mitigating the risks associated with them. Therefore, the potential that HAR has in order to detect physical and cognitive changes provides a great opportunity for the development of bespoke prevention plans.

HAR aims to infer the actions taken by an individual using monitoring sensors [3]. A generic activity recognition model takes as input the data collected by the sensors and aims to accurately classify the activities of the individual.

We use the van Kasteren dataset [4] which consists of binary sensor activity from three different houses (A, B and C) [4]. The binary sensors capture human activity by indicating, for instance, if a door or a cupboard is open or closed, if the toilet is being flushed, or if a person is sitting on a couch, lying in bed or moving in a specific area. The dataset provides sensor readings in 60 second intervals.

In this paper, we present a thorough study of ML techniques including probabilistic (Naïve Bayes, Hidden Markov Model, Hidden Semi-Markov Model and Conditional Random Field) and neural network based (Recurrent Neural Network, Long Short-Term Memory Network, Gated Recurrent Unit, Multi-Layer Perceptron and a Long Short-Term Memory Network with a Conditional Random Field layer) models to the classification task. The main contributions are: (i) A new feature representation (observation-based) is proposed and compared against the state-of-the-art results for other feature representations. The proposed representation outperforms the others and, in general, is able to produce a better accuracy for both dominant and minor classes; (ii) We provide an extensive evaluation and analysis of the aforementioned classification models. Our analysis shows that the Conditional Random Field model performs best using an observation-based representation; (iii) Our best method produces state-of-the-art accuracy on the van Kasteren dataset.

A. Related Work

A number of papers have proposed techniques for classifying the data in [4] and evaluated them using two evaluation metrics: the overall accuracy and the mean per class accuracy[1].

Both generative (e.g. Naïve Bayes (NB) [4], Hidden Markov Models (HMMs) [4], [5]) and discriminative (Support Vector Machines (SVMs) [6], Conditional Random Fields (CRFs) [4], [5]) methods have been evaluated against this dataset. The state-of-the-art methods are Hidden Semi-Markov models (HSMMs) and CRFs [4], [5] depending on which evaluation metric is being considered.

From the literature we are able to identify the state-of-the-art methods which provide the best accuracy and mean per class accuracy, in particular, CRFs and HSMMs, respectively. The previous best results for those metrics and their standard deviation and our improved results are summarised in Table I. Our results for HSMM and CRF differ from the ones that were published in [4], in particular, the values of the mean per class accuracy that we obtained for the CRF method are significantly higher. The improved results are most likely

[1]The accuracy calculates how often the predictions match the class labels and the mean per class accuracy calculates the average of the per-class accuracies.

TABLE I
ACCURACY AND MEAN PER CLASS ACCURACY RATES (%) AND THEIR STANDARD DEVIATION FOR STATE-OF-THE-ART METHODS OBTAINED BY REPRODUCING RESULTS PRESENTED IN [4] AND OUR BEST RESULTS FOR HOUSES A, B AND C

House	Model	Mean per class accuracy	Accuracy
A	HSMM	74.96 ± 12.1 (75.0 ± 12.1 [4])	91.81 ± 5.88 (91.8 ± 5.9 [4])
A	CRF	69.35 ± 12.07 (65.8 ± 14.0 [4])	96.93 ± 2.11 (96.4 ± 2.4 [4])
A	This paper	**88.40 ± 12.43**	**98.95 ± 1.62**
B	HSMM	65.18 ± 13.41 (65.2 ± 13.4 [4])	82.27 ± 13.51 (82.3 ± 13.5 [4])
B	CRF	58.06 ± 7.01 (51.5 ± 8.5 [4])	94.99 ± 5.71 (92.9 ± 6.2 [4])
B	This paper	**79.08 ± 22.35**	**96.07 ± 6.35**
C	HSMM	55.98 ± 15.4 (56.0 ± 15.4 [4])	84.48 ± 13.17 (84.5 ± 13.2 [4])
C	CRF	46.79 ± 15.63 (40.4 ± 16.0 [4])	90.69 ± 9.05 (89.7 ± 8.4 [4])
C	This paper	**76.54 ± 18.99**	**94.10 ± 15.27**

due to the enhancement of the MATLAB library L-BFGS[2] (Limited-memory Broyden–Fletcher–Goldfarb–Shanno [8]).

Recently, Arifoglu et al. [6] applied SVMs and different types of Recurrent Neural Networks (RNNs) to the dataset. In their work, only a portion of the data is used for testing, which differs from the approach taken by van Kasteren et al. [4], where a full K-Fold cross validation is carried out. The results presented in [4] are therefore more trustworthy, hence we apply the same technique in this paper.

Singh et al. [9] applied an LSTM network to the dataset. Even though the results did not outperform state-of-the-art methods, this work demonstrated that LSTMs are capable of performing well given the temporal dependencies present in this dataset.

Other techniques such as stacked autoencoders [10] and modified weighted SVMs [11] have been considered in order to develop a classifier for the dataset. Furthermore, hybrid approaches have also been discussed and applied [12]–[16].

B. Roadmap

The rest of the paper is organised as follows. Section II introduces some of the ML models that were used, Section III presents the proposed feature representation and the pre-processing techniques utilised. Section IV demonstrates the effect that the feature representation as well as the combination of different features has on a model's performance. We also show how our best results improve the state-of-the-art. Section V concludes this paper with pointers to future directions.

II. PRELIMINARY

In this section, we present the task we aim to tackle and provide an overview of some of the ML models applied.

Given a dataset $\{(\mathbf{X}_t^i, \mathbf{y}_t)\}$, such that $t = 1...T$ and $i = 1...N$, where T is the number of data points and N the number of features, the task is to learn a function $f : \mathbb{S}^N \mapsto \{1, ..., c\}$, where S is some abstract space and c the number of activities. In this kind of task, both $\langle \mathbf{X}, \mathbf{y} \rangle$ need to be provided in order to perform supervised learning.

For our dataset, $\mathbf{X}$ represents the sensor data and $\mathbf{y}$ the corresponding labels of the activities performed.

A. Probabilistic models

Naïve Bayes, Hidden Markov Model, Hidden Semi-Markov Model and Conditional Random Field constitute the state-of-the-art probabilistic models for this dataset. A brief description of the Hidden Semi-Markov and Conditional Random Field models is presented in the following sections.

1) Hidden Semi-Markov Model: A Semi-Markov Model is a generalised Poisson [17] process where the holding times need not be independent and identically distributed. Although it is similar to a Markov renewal process [18], the Hidden Semi-Markov Model (HSMM) [19] is a stochastic process where a state has a corresponding length. The length of each state is determined by its duration. Therefore, this is a time-evolving process where the transition between states is made at jump times and dependent upon the corresponding probability distributions.

The main difference between HMMs and HSMMs is the relaxation of the Markov assumption. In particular, HSMMs are able to do this by modelling the duration of a state (e.g. activity). Therefore, a new variable d_t is introduced in this model and the joint probability is calculated as follows:

$$p(\mathbf{y}, \mathbf{X}, \mathbf{d}) = \prod_{t=1}^{T} p(\mathbf{x_t}|y_t)p(y_t|y_{t-1}, d_{t-1})p(d_t|d_{t-1}, y_t).$$

We will use maximum likelihood estimation (MLE) to estimate the parameters θ which maximises the likelihood of observing y, X and d given the model θ: $\hat{\theta} = \arg, \max_{\theta} P(\mathbf{y}, \mathbf{X}, \mathbf{d}|\theta)$.

2) Conditional Random Field: The Conditional Random Field model, which is the most structurally similar to the HMM model, is called a linear-chain CRF. This model relies on the same independence assumptions as the HMM:

(i) y_t is only dependent on y_{t-1} (first order Markov assumption);
(ii) $\mathbf{x_t}$ is only dependent on y_t (output independence assumption).

[2]The improvement of the L-BFGS library in 2011 [7] has likely resulted in a better learning process of the Conditional Random Field model and, consequently, in an improved algorithm that yields a better performance.

Unlike HSMMs, linear-chain CRF models do not explicitly model the duration of a state. The conditional distribution is calculated using the following expression:

$$p(\mathbf{y}|\mathbf{X}) = \frac{1}{Z(\mathbf{X})} \prod_{t=1}^{T} \exp \sum_{l=1}^{L} \lambda_l f_l(y_t, y_{t-1}, \mathbf{x_t}),$$

where $f_l(y_t, y_{t-1}, \mathbf{x_t})$ is a feature function, λ_l is a weight parameter and L is the number of feature functions. The potential function is the exponential representation of the product of $\lambda_l f_l(y_t, y_{t-1}, \mathbf{x_t})$, which can take any positive value, hence why $Z(\mathbf{X})$ is needed as a normalization term.

A CRF is also a stationary process and it uses CMLE (Conditional Maximum Likelihood Estimator), which finds the θ (CRF parameters) that maximises the conditional likelihood of observing y given the model θ: $\hat{\theta} = \arg, \max_{\theta} P(\mathbf{y}|\mathbf{X}, \theta)$. Therefore, unlike HMMs which assume that $\mathbf{x_j}$ are conditionally independent, CRFs make no assumptions about $p(\mathbf{X})$.

B. Recurrent Neural Network models

One of the main differences between statistical and neural network models is related to interpretability. Unlike statistical ML models, neural network models do not provide interpretation even though they do provide an effective representation of data properties [20].

In the following sections, two different recurrent neural network models are presented: RNN and LSTM.

1) Recurrent Neural Network: The RNN model considered is a fully-connected RNN, where the ouput is fed back to the input. Hence, RNNs contain loops in them which is what allows these type of networks to learn temporal dependencies.

Let $\mathbf{x} = (x_1, ..., x_T)$ be an input sequence and $\mathbf{h} = (h_1, ..., h_T)$ the hidden vector sequence computed by a recurrent neural network. In an RNN, the hidden vector $h^{(t)}$, at time step t, is computed as follows:

$$h^{(t)} = \phi(W_h h^{(t-1)} + W_x x^{(t)}) + b_h),$$

where ϕ is the activation function. The parameters W and b are the weight matrix and bias vector, respectively.

2) Long Short-Term Memory Network: In long-term dependencies, when there is a large time gap between where specific information is stored and where it is needed, RNNs do not perform well; and LSTMs [21] are a better and more robust solution. LSTMs are a type of RNNs which are able to detect dependencies across long time windows.

The LSTM architecture is composed of connected cells and each cell is constituted by three gates: the input ($i^{(t)}$), output ($o^{(t)}$) and forget ($f^{(t)}$) gates, which control the information that is added to or removed from the cell. Moreover, besides having an internal state $c^{(t)}$, a cell also contains a layer which produces the variable $\widetilde{c}^{(t)}$. This variable is representative of the candidate values which may potentially be added to the internal state.

This type of networks are able to learn the importance of features over time by storing information in the hidden layers. This is done by performing an optimisation of the weights that impacts the information flow. Consequently, LSTMs can lead to a better comprehension of data patterns, which makes them useful to be applied in the field of HAR.

The following equations are used, in an iterative manner, to obtain the scalar value $h^{(t)}$, at time step t, of the output vector of the cell. The symbol $\odot$ denotes element-wise multiplication.

$$\begin{aligned} i^{(t)} &= \sigma(W_{ih} h^{(t-1)} + W_{ix} x^{(t)} + b_i) \\ f^{(t)} &= \sigma(W_{fh} h^{(t-1)} + W_{fx} x^{(t)} + b_f) \\ o^{(t)} &= \sigma(W_{oh} h^{(t-1)} + W_{ox} x^{(t)} + b_o) \\ \widetilde{c}^{(t)} &= \phi(W_{ch} h^{(t-1)} + W_{cx} x^{(t)} + b_c) \\ c^{(t)} &= i^{(t)} \odot \widetilde{c}^{(t)} + f^t \odot c^{(t-1)} \\ h^{(t)} &= o^{(t)} \odot \phi(c^{(t)}), \end{aligned}$$

where σ and ϕ are the activation functions.

III. Learning from Observation-based Representations

A. The Dataset

The dataset which will be used in the experiments refers to sensor activity in three different houses (A, B and C) [4]. The data is representative of the activation and deactivation of binary sensors, where a reading is provided every minute for time spans ranging from 14 to 25 days. As a result, in the data there are long stretches where the sensor readings do not change. For example, for houses B and C, on average, the sensors change state only every one and a half hour.

Van Kasteren et al. [4] used various types of binary sensors (e.g. passive infrared; pressure mats; reed switches), which were placed in three different environments: houses A, B and C. In order to map the observations obtained from these sensors to activities, an annotation system was put in place [5].

Table II presents some information about this dataset, in particular, the number of sensors placed around the house, the number of activities, the age of the person who inhabited the house and how many days of data we have. The relative frequencies of activities in the three different houses are represented in the full version of the paper [22]. In general, the most frequent labels in the three houses are 'Idle', 'Leave house' and 'Go to bed'. A slight higher frequency of label 'Idle' is noticeable for house C. On the other hand, the label 'Leave the house' acquires a higher frequency in houses A and B.

B. Observation-based Representation

Since there are long periods of time where the sensors do not change, learning temporal dependencies on this type of data requires a long time history of previous data points, denoted as look-back window. We have observed that there is a gradual increase of training time with higher values for the look-back window. To overcome this, we propose a new representation for sensor data called observation-based (OB) representation, which combines consecutive data points with the same sensor readings into one data point. Hence, data points are merged if sensor readings remain unchanged.

TABLE II
DETAILS ABOUT THE DATASET

House	Sensors	Activities	Age	Duration (days)
A	14	10	26	25
B	23	13	28	14
C	21	16	57	19

Furthermore, three different feature representations were considered in [4]: raw, changepoint and last-fired. These were initially introduced in [5] and are a way of comparing how the data is given as an input and the impact that it has in the overall recognition performance. In the raw representation, the sensor takes value 1 when it is activated and 0 otherwise; with the changepoint representation, the sensor takes value 1 when it changes state and 0 otherwise; the last fired representation makes the last sensor that changed state to take value 1 until another sensor changes its state.

In comparison, our proposed representation is more expressive than the changepoint and last-fired representations, because it yields information about the current and/or most recent sensors that have changed its value, without having to provide a large number for the look-back window. The disadvantage of having a large number for the look-back window is that it may affect the classification of other activities which do not require all the information provided by the data that is fed into the network.

C. Time Related Features

When computing the OB representation, the variable Δt is obtained by calculating how long the sensor readings remain unchanged. Since the dataset provides sensor readings in 60 second intervals, Δt indicates the duration (in minutes) of no change for a sensor reading. We study the effect of using this variable as well as the $hour$ variable, which represents the hour of a sensor reading. By incorporating the latter, the information provided can be useful for classification purposes.

The frequency of each possible value for variable Δt in house A is presented in Figure 1 and we observe a similar distribution for houses B and C. For house A, this variable can take values from 1 to 2732. In order to keep the number of features small, we further discretize Δt into coarser bins. Hence, each bin will essentially represent an interval. Based on the relative frequency, we considered two different ways of splitting this variable into intervals: one results in a total of 48 intervals and the other one in a total of 7. The difference between the two lies on the importance of categorising smaller durations. Further details related to the discretization process are described in the full version of the paper [22]. We then encode each interval considering two different encoding processes: one-hot and unary-based encodings.

Regarding the one-hot encoding process, it will generate a squared matrix, where the number of rows is the same as the number of values. Therefore, it creates new binary columns, indicating the presence of each possible value. As for

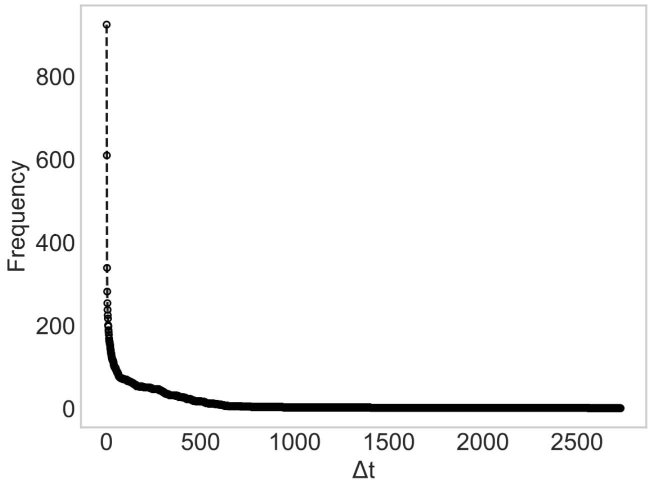

Fig. 1. Frequency of the Δt variable for house A

the unary-based encoding, this process also creates a squared matrix which has the same dimension as the matrix generated in the previous encoding process. The main difference between these two encoding processes lies on the interpretation of the binary columns. In the one-hot encoding, the binary columns indicate the presence of each possible value, therefore only one component in each column will take value one. On the other hand, for the unary-based coding, the binary columns indicate the presence of values that are less than or equal to each possible value; hence, without loss of generality, supposing the values are in ascending order, all the elements of the lower triangle of the matrix will take value one.

In regard to the $hour$ variable, which can take values from 0 to 23, we also encode this variable using the two processes aforementioned (one-hot and unary-based encodings) but we consider each number a category so, for this particular variable, we will have exactly 24 values. Hence, each category will be representative of the hour of the sensor reading. The reason why we encode this variable such that the values of the features are in the same range as the other features is because this makes training faster and reduces the chances of getting stuck in local optima.

IV. EXPERIMENTS

In the following experiments, an OB representation of the dataset is used in order to compare and evaluate against other feature representations. The OB representation is obtained by directly collecting information from the sensors, which corresponds to the data in its raw representation format. The raw representation gives the worst results, irrespective of the algorithm. Hence, a good performance by both generative and discriminant algorithms is always dependent on considering a changepoint or last-fired representation. The OB representation provides a generalisation of the changepoint and last-fired representations. Some further discussion of the proposed method and analysis of the results is presented in the following sections.

These experiments were run using a K-Fold cross validation approach, where we cycle through each one of the days using it for testing and the data corresponding to the remaining days is used for training. This is consistent with the technique applied by van Kasteren et al. [4]. The mean per class accuracy as well as the overall accuracy are presented as evaluation metrics; and the accuracy for each class is also calculated.

In regard to the neural network models - RNN, LSTM, GRU, MLP and LSTM with a CRF layer (LSTMCRF) - we considered the following set of hyper-parameters: 128 for the number of units, a learning rate of 0.0001, 100 for the number of epochs and a batch size of 512.

The optimisation algorithms that were used in order to minimise the error rates of the ML models were the Root Mean Square Propagation for training the RNN, LSTM and GRU models, and the Adaptive Moment Estimation optimiser was used for training the MLP and LSTMCRF models.

These parameters were selected after analysing the training losses and accuracies of the models applied by taking into consideration their performances across different sets of hyper-parameters. We also aimed at making a fair comparison among these methods and selected the same set of hyper-parameters for the NN-based models. Moreover, these particular parameters have shown to work relatively well for these methods irrespective of the feature representation.

A. Evaluation metrics

We will be using the mean per class accuracy and the *accuracy* as evaluation metrics for our experiments. The latter can be defined as follows. Let $pred$ and $true$ be the N-dimensional arrays which contain the model's predictions and the true labels of each data point, respectively. Then, the accuracy is the percentage of correctly predicted activities, i.e.:

$$\mathsf{accuracy} = \frac{|\{i \in \{1, ..., N\} \mid \mathsf{pred}(i) = \mathsf{true}(i)\}|}{N}.$$

Given the imbalance of the dataset, a classifier would not be properly evaluated if accuracy was the only metric utilised to assess its performance. Therefore, the accuracy for each class is also presented in order to analyse whether the models are being able to accurately classify not only highly frequent classes but also infrequent ones.

Formally, the accuracy of a class c is given by

$$\mathsf{accuracy_c} = \frac{|\{i \in \{1, ..., N_c\} \mid \mathsf{pred_c}(i) = \mathsf{true_c}(i)\}|}{|\{i \in \{1, ..., N\} \mid \mathsf{true}(i) = c\}|},$$

where $pred_c$ and $true_c$ are the N_c-dimensional arrays which contain the model's predictions and the true labels of each data point belonging to a class c, respectively.

Lastly, we define the mean per class accuracy as follows. Let $c \in \{1, ..., C\}$, where C is the number of activities in a dataset. Then, the mean per class accuracy is calculated according to the following expression:

$$\mathsf{mean\_per\_class\_accuracy} = \frac{1}{C}\sum_{c=1}^{C} \mathsf{accuracy_c}.$$

The best values for the mean per class accuracy, overall accuracy and per-class accuracies are highlighted in bold.

B. Effect of the Feature Representation

All the experiments presented in this section do not take into consideration the features $hour$ (*NoToD*, where *ToD* stands for *Time of Day*) nor the Δt (*NoDeltaT*), i.e. the features $hour$ and Δt were not added to the dataset.

1) Raw Feature Representation: We evaluated 8 different methods using a raw feature representation: NB, HMM, HSMM, CRF, LSTM, GRU, RNN and LSTMCRF. We considered a look-back window of 1 and this serves as a baseline for the experiments run in the next subsections. In particular, the results provided by the methods NB, HMM, HSMM and CRF were obtained by reproducing the experiments done in [4]. The CRF model outperformed the other models for houses A and C. Specifically, the accuracy(mean per class accuracy) achieved for house A was 91.85±7.80(59.13 ± 15.66) and for house C 73.83±22.39(32.03±20.23). For house B, the RNN model achieved the best overall accuracy (87.16±11.12) in comparison to the other models; however, the CRF model provided the best value for the mean per-class accuracy (47.64±13.17).

2) Observation-based representation with RNN-based methods: We have also applied the LSTM, GRU, RNN and LSTMCRF methods to the raw and OB feature representations. We considered the following values for the look-back window: 2, 5 and 10. In Table III, we present the results achieved for house A considering the RNN model across different look-back window values. The full results are given in the full version of the paper [22].

In regard to house A, we observe that, for all methods, this dataset does not require a large value for the look-back window in order to be able to accurately classify highly frequent labels. LSTM is the method which provides the highest accuracy considering a look-back window of 2. Also, considering the mean per class accuracy, GRUs are able to perform better than any of the other RNN-based methods. We observe that the optimal value for the look-back window here was 5, which only differs 0.3 percent points from the result obtained for the same method with a look-back window of 2; therefore, since the difference between the mean per-class accuracies is not significant, a small look-back window provides enough knowledge in order to achieve a good performance in this classification task.

For house B, LSTMCRF is the method which provides highest accuracy considering a look-back window of 2. As for the mean per class accuracy, RNN with a look-back window of 5 is the method that performs the best, but we observe once again that there is not a significant difference between the mean per class accuracies for a look-back window of 2 and 5.

Lastly, for house C, the RNN method achieved the highest values for the evaluation metrics considered, where a look-back window of 5 and 2 gave the best results for the mean per-class accuracy and the accuracy, respectively.

TABLE III
ACCURACY (%) PER LABEL FOR RNN MODEL (RAW VS OB FEATURE REPRESENTATIONS) - HOUSE A (LOOK-BACK WINDOW: 2, 5 AND 10)

Label	2		5		10	
	Raw	OB	Raw	OB	Raw	OB
'Idle'	25.94	**86.0**	30.55	66.18	39.19	55.41
'Leave house'	96.45	**99.88**	96.37	98.89	96.17	97.62
'Use toilet'	44.66	**67.67**	52.88	63.84	47.95	55.62
'Take shower'	0.0	0.0	0.4	**22.31**	7.57	**22.31**
'Brush teeth'	0.0	0.0	0.0	0.0	0.0	0.0
'Go to bed'	95.21	**97.76**	95.21	94.33	95.44	91.89
'Prepare breakfast'	49.43	54.02	52.87	48.28	**55.17**	44.83
'Prepare dinner'	9.76	15.33	12.54	**26.13**	13.24	23.69
'Get snack'	0.0	4.76	0.0	4.76	2.38	**7.14**
'Get drink'	34.69	34.69	32.65	28.57	**38.78**	6.12
Mean per class accuracy	48.68	**62.58**	51.22	60.70	53.68	55.54
Standard deviation	17.42	13.81	16.93	17.50	16.42	17.87
Overall accuracy	85.39	**95.46**	86.0	91.77	87.0	88.86
Standard deviation	10.49	3.44	10.25	7.91	9.97	9.0

In general, we observe that for lower look-back window values, our proposed feature representation achieves significantly better results than the raw representation. Moreover, from the results obtained for the LSTM, GRU, RNN and LSTMCRF models, we conclude that the best accuracy for all houses was obtained by considering a look-back window of 2 and an OB feature representation of the data. In addition, neural network models seem not to benefit much from concatenating multiple data points for training as those techniques learn temporal dependencies differently.

We also note that, if considering the raw feature representation, a longer look-back window is required so that LSTM models are able to obtain reasonable results. In particular, it becomes hard to accurately predict labels due to the long-term dependencies inherent to the raw feature representation. Therefore, based on the results obtained, this implies that there is an advantage in using the proposed feature representation. The OB feature representation is shown to be beneficial not only in obtaining a higher accuracy but also in decreasing the training time given that a better performance is achieved when considering a low look-back window value.

3) Observation-based representation with probabilistic-based methods and a MLP network model: In this experiment, we use probabilistic models and a feed forward neural network model and considered an OB feature representation. Unlike recurrent neural networks, models such as NB, HMM, HSMM, CRF and MLP are limited to a single "time step" (i.e. a look-back window of 1). However, it is possible to provide look-back information to these models. We accomplish this by feeding in a sequence which contains concatenated data points. Specifically, we add the most recent data points as further features of the current single data point. We consider 2, 5 and 10 as the possible values for the number of recent data points to be concatenated with the current one.

Also, we do not consider the raw representation for these models as it would result in low information signals, where repeated information would be given as input to the models in the form of equal concatenated data points.

For both overall accuracy as well as per-class accuracies, CRFs were able to outperform all the experiments done thus far by using an OB feature representation. The best accuracy values were obtained by concatenating 5 data points for house A (97.14±5.89) and 10 data points for houses B (87.55±16.77) and C (90.43±14.85). Nevertheless, the experiments also show that a higher value for the number of concatenated data points significantly contributes towards a higher mean per class accuracy.

C. Adding the Time of Day as a further feature

From the results presented in the last section, it is possible to conclude that CRF is the algorithm which overwhelmingly is able to perform the best using an OB feature representation. In this section, we show the results obtained by adding the time of day ($hour$) as a further feature to the dataset. In total, we considered fifteen different feature combinations in our experiments.

In all the experiments presented in Section IV-B, the features $hour$ (*ToD*) and Δt (*DeltaT*) were not added to the dataset (*NoToD&NoDeltaT*). In order to test and evaluate the need to better distinguish duration intervals, we considered all the other feature combinations, which result from adding a one-hot(unary-based) encoding of i intervals of the feature Δt - $OneHotDeltaT_i(UnaryDeltaT_i)$ - and/or a one-hot(unary-based) encoding of the feature $hour$ - $OneHotToD(UnaryToD)$ - to the dataset.

The best performance for houses A, B and C resulted from the feature combinations *UnaryToD&UnaryDeltaT7* (5 data points concatenated), *UnaryToD&UnaryDeltaT48* (5 data points concatenated) and *OneHotToD&UnaryDeltaT48* (10 data points concatenated), respectively. The full results are given in the full version of the paper [22]. Furthermore, we observe that only house C significantly benefits from using more Δt values and generally, one-hot and unary-based encodings produce similar results for all houses.

Specifically, 98.95 ± 1.62 was the best result achieved for house A, where a unary-based encoding with 7 bins was considered. For house B, the best result achieved was 96.07 ± 6.35 by applying a unary-based encoding with 48 bins and the best result obtained for house C was 94.10 ± 15.27 by using a unary-based encoding with 48 bins.

D. Comparison with State-of-the-art methods

In this section, we present our best results and compare them against the state-of-the-art (Tables IV, V and VI). The state-of-the-art methods for this dataset are HSMM and CRF using changepoint and last-fired feature representations [4].

For house A (Table IV), we observe that the accuracy of every label increased by applying a CRF model with our proposed representation. In particular, the label whose accuracy benefited the most by using the OB representation was 'Get snack', which improved by 45%. Other labels that

TABLE IV

ACCURACY AND MEAN PER CLASS ACCURACY RATES (%) AND THEIR STANDARD DEVIATION FOR STATE-OF-THE-ART METHODS AND OUR BEST METHOD FOR HOUSE A - CRF USING OB FEATURE REPRESENTATION (UNARYTOD&UNARYDELTAT7 - DATA POINTS CONCATENATED: 5)

Label	HSMM (Change-point) [4]	CRF (Last-fired) [4]	This paper
'Idle'	50.75	86.62	**95.98**
'Leave house'	99.66	**99.92**	**99.92**
'Use toilet'	82.19	61.64	**82.74**
'Take shower'	64.94	27.89	**82.07**
'Brush teeth'	34.38	0.0	**40.62**
'Go to bed'	96.53	**99.76**	99.64
'Prepare breakfast'	68.97	68.97	**86.21**
'Prepare dinner'	51.57	88.85	**99.65**
'Get snack'	54.76	14.29	**100.0**
'Get drink'	67.35	44.9	**89.8**
Mean per class accuracy	74.96	69.35	**88.40**
Standard deviation	12.10	12.07	12.43
Accuracy	91.81	96.93	**98.95**
Standard deviation	5.88	2.11	1.62

TABLE V

ACCURACY AND MEAN PER CLASS ACCURACY RATES (%) AND THEIR STANDARD DEVIATION FOR STATE-OF-THE-ART METHODS AND OUR BEST METHOD FOR HOUSE B - CRF USING OB FEATURE REPRESENTATION (UNARYTOD&UNARYDELTAT48 - DATA POINTS CONCATENATED: 5)

Label	HSMM (Change-point) [4]	CRF (Change-point) [4]	This paper
'Idle'	59.86	72.62	**75.24**
'Leaving the house'	93.7	**99.69**	99.21
'Use toilet'	**71.43**	31.17	70.13
'Take shower'	**92.79**	87.39	72.07
'Brush teeth'	33.33	19.44	**63.89**
'Go to bed'	68.65	96.15	**97.06**
'Get dressed'	69.57	69.57	**86.96**
'Prepare brunch'	59.52	71.43	**82.14**
'Prepare dinner'	38.03	**97.18**	95.77
'Get a drink'	**42.86**	14.29	28.57
'Wash dishes'	23.81	42.86	**71.43**
'Eat dinner'	42.86	0.0	**100.0**
'Eat brunch'	39.04	0.0	**63.7**
Mean per class accuracy	65.18	58.06	**79.08**
Standard deviation	13.41	7.01	22.35
Accuracy	82.27	94.99	**96.07**
Standard deviation	13.51	5.71	6.35

had significant improvements were 'Take shower' (17%), 'Prepare breakfast' (17%) and 'Get drink' (23%). On average, considering the label accuracies, we observe an improvement of 13% between the best value obtained from the state-of-the-art methods (HSMM (Changepoint) and CRF (Last-fired)) and the CRF model with our proposed representation.

In regard to house B (Table V), we observe that the accuracy of most of the labels improves, but the labels 'Take shower' and 'Get a drink' decrease by 21% and 14%, respectively. In particular, the accuracy of label 'Take shower' decreases due to being misclassified as 'Going to bed' and 'Prepare brunch'. As for label 'Get a drink', it is classified 63% of the times as 'Idle', 'Brush teeth' and 'Prepare brunch'. Nevertheless, on average, we obtain an improvement of 10.3% between the best value obtained from the state-of-the-art methods and the CRF model with the OB representation.

TABLE VI

ACCURACY AND MEAN PER CLASS ACCURACY RATES (%) AND THEIR STANDARD DEVIATION FOR STATE-OF-THE-ART METHODS AND OUR BEST METHOD FOR HOUSE C - CRF USING OB FEATURE REPRESENTATION (ONEHOTTOD&UNARYDELTAT48 - DATA POINTS CONCATENATED: 10)

Label	HSMM (Last-fired) [4]	CRF (Last-fired) [4]	This paper
'Idle'	68.57	82.6	**85.81**
'Leave house'	86.19	95.96	**98.14**
'Eating'	22.19	6.73	**72.07**
'Use toilet downstairs'	**63.29**	21.52	27.85
'Take shower'	60.0	36.32	**81.58**
'Brush teeth'	26.73	4.95	**78.22**
'Use toilet upstairs'	45.0	13.75	**52.5**
'Shave'	43.48	31.88	**97.1**
'Go to bed'	98.03	**99.37**	96.76
'Get dressed'	69.64	56.25	**81.25**
'Take medication'	26.67	0.0	**40.0**
'Prepare breakfast'	33.8	49.3	**76.06**
'Prepare lunch'	48.33	41.67	**83.33**
'Prepare dinner'	69.31	55.86	**90.69**
'Get snack'	20.83	4.17	**66.67**
'Get drink'	0.0	6.45	**51.61**
Mean per class accuracy	55.98	46.79	**76.54**
Standard deviation	15.4	15.63	18.99
Accuracy	84.48	90.69	**94.10**
Standard deviation	13.17	9.05	15.27

We observe that the largest improvement regarding label accuracy was given by house C (Table VI): on average, there was an improvement of 22% between the best value obtained from the state-of-the-art methods (HSMM (Changepoint) and CRF (Last-fired)) and the CRF model with our proposed representation. One exception we observe is the label 'Use toilet downstairs'. The highest accuracy for this label is obtained with the HSMM method and a last-fired representation. This occurs because, most of the times, the other two feature representations misclassify this highly infrequent label as 'Idle'.

From the experiments above, we conclude that the OB representation outperformed the state-of-the-art feature representations and, in general, there is not only a significant improvement in the accuracies for each class but also in the overall accuracy. The confusion matrices for each method applied to each dataset are given in the full version of the paper [22].

Even though CRFs outperform HSMM from an overall accuracy standpoint, when considering the per-class accuracy, HSMMs are sometimes able to better classify infrequent classes in comparison to CRFs. This results from the learning process each method is undertaking. Specifically, HSMMs build a model $p(\mathbf{x_t}|\mathbf{y_t})$ for each class, whereas CRFs use the same model for all classes by computing $p(\mathbf{y}|\mathbf{X})$, which causes competition among classes. Consequently, if a dataset is imbalanced, a higher likelihood may be obtained if the data points are classified as the dominant class(es) than if the low frequent classes are considered and some of the dominant ones are misclassified [5].

V. Conclusion

In this paper, we have presented a thorough study of different ML techniques for a standard HAR dataset. Our experiments show that a significant improvement was made in comparison to state-of-the-art methods in the HAR field.

A new representation for data that is to be given as input to a model was presented. The results have shown that, by applying such a representation, models are better able to learn data patterns and, consequently, successfully perform a classification task in the HAR domain for both dominant and minor classes.

By using an OB representation, we improved the mean per-class accuracy and the accuracy for house A by 13.44% and 2.02%, respectively, in comparison with the state-of-the-art results. Moreover, for house B, the aforementioned evaluation metrics increased 13.9% and 1.08%, respectively. As for house C, results improved 20.56% and 3.41% for the respective evaluation metrics considered.

Given the results obtained with an observation-based representation, its usage may also be suitable and advantageous in other domains. Moreover, using adversarial zero-shot learning [23], [24] to recognise abnormal human activity is an interesting direction for future work.

Acknowledgment

The support of NVIDIA Corporation is gratefully acknowledged with the donation of the Quadro P6000 GPU used for this research.

References

[1] E. De la Hoz, P. Ariza, J. Medina, and M. Espinilla, "Sensor-Based Datasets for Human Activity Recognition – A Systematic Review of Literature," *IEEE Access*, vol. 6, pp. 59 192–59 210, 2018.

[2] J. Kaye, S. Maxwell, N. Mattek, T. Hayes, H. Dodge, M. Pavel, H. Jimison, K. Wild, L. Boise, and T. Zitzelberger, "Intelligent Systems for Assessing Aging Changes: Home-Based, Unobtrusive, and Continuous Assessment of Aging," *The journals of gerontology. Series B, Psychological sciences and social sciences*, vol. 66 Suppl 1, pp. i180–i190, 2011.

[3] L. Chen, J. Hoey, C. Nugent, D. Cook, and Z. Yu, "Sensor-Based Activity Recognition," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 790–808, 2012.

[4] T. van Kasteren, G. Englebienne, and B. Krose, "Human Activity Recognition from Wireless Sensor Network Data: Benchmark and Software," in *Activity recognition in pervasive intelligent environments*, L. Chen, C. D. Nugent, J. Biswas, and J. Hoey, Eds. Atlantis Press, 2011, vol. 4, pp. 165–186.

[5] T. van Kasteren, A. Noulas, G. Englebienne, and B. Kröse, "Accurate activity recognition in a home setting," in *Proceedings of the 10th International Conference on Ubiquitous Computing*, 2008, pp. 1–9.

[6] D. Arifoglu and H. Bouchachia, "Activity Recognition and Abnormal Behaviour Detection with Recurrent Neural Networks," *Procedia Computer Science*, vol. 110, pp. 86–93, 2017.

[7] S. Becker, "LBFGSB (L-BFGS-B) mex wrapper," https://www.mathworks.com/matlabcentral/fileexchange/35104-lbfgsb-l-bfgs-b-mex-wrapper, accessed: 2019-12-05.

[8] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A Limited Memory Algorithm for Bound Constrained Optimization," *SIAM Journal on Scientific Computing*, vol. 16, pp. 1190–1208, 1995.

[9] D. Singh, E. Merdivan, I. Psychoula, J. Kropf, S. Hanke, M. Geist, and A. Holzinger, "Human Activity Recognition Using Recurrent Neural Networks," in *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, 2017, pp. 267–274.

[10] G. Chen, A. Wang, S. Zhao, L. Liu, and C.-Y. Chang, "Latent feature learning for activity recognition using simple sensors in smart homes," *Multimedia Tools and Applications*, vol. 77, p. 15201–15219, 2018.

[11] M. Abidine, L. Fergani, B. Fergani, and M. Oussalah, "The joint use of sequence features combination and modified weighted SVM for improving daily activity recognition," *Pattern Analysis and Applications*, vol. 21, p. 119–138, 2018.

[12] K. Guo, Y. Li, Y. Lu, X. Sun, S. Wang, and R. Cao, "An Activity Recognition-Assistance Algorithm Based on Hybrid Semantic Model in Smart Home," *International Journal of Distributed Sensor Networks*, vol. 12, 2016.

[13] I. Ihianle, "A Hybrid Approach to Recognising Activities of Daily Living from Patterns of Objects Use," Ph.D. dissertation, University of East London Architecture Computing and Engineering, 2018.

[14] G. Okeyo, L. Chen, H. Wang, and R. Sterritt, "A Hybrid Ontological and Temporal Approach for Composite Activity Modelling," in *IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, 2012, pp. 1763–1770.

[15] F. J. Ordóñez, P. De Toledo, and A. Sanchis, "Activity Recognition Using Hybrid Generative/Discriminative Models on Home Environments Using Binary Sensors," *Sensors*, vol. 13, pp. 5460–5477, 2013.

[16] D. Riboni, L. Pareschi, L. Radaelli, and C. Bettini, "Is ontology-based activity recognition really effective?" in *IEEE International Conference on Pervasive Computing and Communications Workshops*, 2011, pp. 427–431.

[17] P. C. Consul and F. Famoye, "Generalized Poisson Distribution," in *Lagrangian Probability Distributions*. Birkhäuser, 2006, pp. 165–190.

[18] N. Limnios and G. Oprişan, "Markov Renewal Processes," in *Semi-Markov Processes and Reliability*. Birkhäuser, 2001, pp. 31–49.

[19] S.-Z. Yu, "Hidden semi-Markov models," *Artificial Intelligence*, vol. 174, pp. 215–243, 2010, special Review Issue.

[20] M. Karlaftis and E. Vlahogianni, "Statistical methods versus neural networks in transportation research: Differences, similarities and some insights," *Transportation Research Part C: Emerging Technologies*, vol. 19, pp. 387–399, 2011.

[21] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, pp. 1735–1780, 1997.

[22] F. Alves, M. Gairing, F. A. Oliehoek, and T.-T. Do, "Sensor Data for Human Activity Recognition: Feature Representation and Benchmarking," *CoRR*, vol. abs/2005.07308, 2020. [Online]. Available: http://arxiv.org/abs/2005.07308

[23] C. Lampert, H. Nickisch, and S. Harmeling, "Learning To Detect Unseen Object Classes by Between-Class Attribute Transfer," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2009, pp. 951–958.

[24] B. Tong, M. Klinkigt, J. Chen, X. Cui, Q. Kong, T. Murakami, and Y. Kobayashi, "Adversarial Zero-shot Learning With Semantic Augmentation," in *AAAI Conference on Artificial Intelligence*, 2018, pp. 2476–2483.