Passive RFID Module with LSTM Recurrent Neural Network Activity Classification Algorithm for Ambient Assisted Living

George Oguntala, *Senior Member, IEEE,* Yim-Fun Hu, *Senior Member, IEEE,* Ali A. S. Alabdullah, Raed Abd-Alhameed, *Senior Member, IEEE,* Muhammad Ali, Doanh Luong

Abstract—**Human activity recognition from sensor data is a fundamental research topic to achieve remote health monitoring and ambient assisted living (AAL). In AAL, sensors are integrated into conventional objects aimed at enabling people's capabilities through digital environments that are sensitive, responsive and adaptive to human activities. Moreover, new technological approaches to support AAL within the home or community setting offers people the prospect of more individually focused care and improved quality of living. In the present work, an ambient human activity classification framework that augments information from the received signal strength indicator (RSSI) of passive RFID tags to obtain detailed activity profiling is proposed. Key indices of position, orientation, mobility, and degree of activities which are critical to guide reliable clinical management decisions using 4 volunteers are employed to simulate the research objective. A two-layer, fully connected sequence long short-term memory recurrent neural network model (LSTM RNN) is implemented. The LSTM RNN model extracts the feature of RSS from the sensor data and classifies the sampled activities using SoftMax. The performance of the LSTM model is evaluated for different data size and the hyper-parameters of the RNN are adjusted to optimal states, which results in an accuracy of 98.18%. The proposed framework suits well for smart homes and smart health and offers a pervasive sensing environment for the elderly, persons with disability and chronic illness.**

Index Terms—Activity recognition, ambient assisted living, LSTM, recurrent neural network, RFID, smart homes

I. INTRODUCTION

WORLD demography shows that ageing and chronic diseases are on the increase. With the prevalence of chronic diseases among the elderly and younger generation, the demand for improved healthcare services is rapidly becoming an economic burden with healthcare budgets overstretched in most developed and developing countries. Recent reports from the world health organization (WHO) identified chronic diseases in both old and young people as a leading cause of disability, and if not successfully managed, would become the most expensive problem for most grown and growing economies [1, 2]. Moreover, with the global demographic trend towards the elderly, persons with chronic illness and immunocompromised patients, the demand for entirely new healthcare solutions using different indoor technologies [3] is ever-changing.

Human activity recognition (HAR) is a key aspect of ubiquitous computing and human-robot interaction with significant application in healthcare for tracking, rehabilitation assistance, elderly care support and cognitive disorder recognition in various environments. Understanding and recognising human activities is a cost-effective alternative healthcare approach to ameliorate the proliferating demand for independent and assistive living.

HAR from sensor data has proved to be an effective approach in pervasive computing due to its viable potentials in many wireless applications. Indoor human activities usually contain rich semantic information which can be applied as landmarks to infer the target's location and position. In addition, HAR is achieved via wearable and ambient or device-free sensing approach. Wearable sensing depends on the target to determine where the wearable device is worn and the device position with regards to the performed activity, which implies that the transition between each position has to be detected [4]. With the prevalence of smart sensor technology and the Internet of Things, wearable sensing approach has become more and more popular and can be implemented directly using the smartphone. However, several wearables are often found obtrusive, causing discomfort to sensitive patients.

G.A. Oguntala is with the Biomedical Engineering, Faculty of Health, Education and Life Sciences, Birmingham City University, West Midland, UK (e-mail: George.Oguntala@bcu.ac.uk)

Y. Fun Hu, Mohammad Ali, D. K. Luong are with the Future Ubiquitous Network (FUN) Research Group, Department of Biomedical and Electronics Engineering, Faculty of Engineering and Informatics, University of Bradford, West Yorkshire, UK (e-mail: Y.F.Hu@bradford.ac.uk, M.Ali70@bradford.ac.uk, K.D.Luong@bradford.ac.uk)

R.A. Abd-Alhameed, A.S. Alabdullah are with Antenna, Applied Computational Electromagnetics and Radio Frequency Research, Department of Biomedical and Electronics Engineering, University of Bradford, West Yorkshire, UK (e-mail: A.S.Alhabdalla@bradford.ac.uk, R.A.A.Abd@bradford.ac.uk).

Ambient sensing involves device-free approach where devices/sensors fixed at predetermined locations of interest are used to detect different human activities. Ambient sensing offers the advantage of wireless human activity detection. It is worth noting that traditional ambient systems require the use of camera and computer vision, which are effective for large coverage data acquisition, but are often found invasive in nature [5-7]. Nonetheless, the development of HAR systems for diverse applications typically follow key standard sequence of activities that involves sensor network, data acquisition, data analytics and classification techniques [8].

To this end, the main motivation of the present work is the extension of our previous work in [9] to employ a continuous-time classification approach using the LSTM RNN for AAL. The human activity classification framework augments data in the form of received signal strength (RSS) from passive RFID tags to obtain a more detailed activity profiling. The discriminative power of each RFID tag is sampled and processed for different sampled human activities. The physical characteristics of position, orientation, mobility, and degree of activities are key indices to support effective clinical management decisions, especially for the elderly, persons with disabilities and chronic illness and immunocompromised patients.

Therefore, the contributions of this paper are as follows:
1. We employed deep learning via LSTM RNN to model the sequential dataset from the RSS of passive RFID and learn high-level representations to improve time-dependency.
2. The LSTM RNN model is used to classify the experimental datasets to achieve human activity recognition.
3. The performance of the LSTM RNN is evaluated with on different data size to evaluate the complexity and accuracy of the classification model.

In the following section, we present a review of related works on different ambient approaches for human activity recognition in Section II. In Section III, the methodology of the LSTM model is presented. Section IV presents the experimental evaluation of the study using the SmartWall. In Section V, the training of the LSTM model is presented whilst the activity classification of the dataset is presented in Section VI. Performance evaluation of the LSTM model using different metrics is presented in Section VII. Finally, the conclusion drawn from the present work is presented in Section VIII.

II. RELATED WORKS

Ambient intelligence (AmI) in healthcare as a computing paradigm where sensors are integrated into conventional objects aimed at enabling people's capabilities through digital environments that are adaptive, sensitive, responsive and adaptive to human activities and emotions has been receiving intense attention in the subject of smart homes [10-13]. Readers interested in the subject of AmI for AAL and smart homes are referred to the following articles [14-19].

Different interesting AAL solutions using AmI have been proposed in the literature. Magherini et al. in [20] proposed the automated recogniser of activities of daily living (ADL) using the propositional temporal logic model. The proposed system is evaluated within the context of the smart kitchen. Jayatilaka et al. proposed the Home of Things for AAL (HoTAAL) solution [21]. In [21], appliances are developed to support unobtrusive, seamless social interactions with the elderly to assist them with ADL with a focus on meal preparation. Wan et al. in [22] proposed the Cummulately Overlapping windowing approach for amBient recognition of activities (COBRA) system. The proposed COBRA system employs the sliding window algorithm augmented with logistic regression to recognise real-time activities within a 60-secs window. Biagi et al. in [23] proposed a continuous-time AAL approach for online and offline ADL using the continuous-time Hidden Semi-Markov model and continuous-time Hidden Markov model. Garcia et al. in [24] proposed the Safe Neighborhood solution to crowd sense people outdoor using computer interferences from contextual and sensor data to provide safer community monitoring of the elderly. The above solutions employ different perspective of methodologies under various conditions to support AAL. However, the present work is based on the deep learning approach due to its high-level feature extraction and learning capabilities to achieve a more robust, real-time HAR solution.

III. METHODOLOGY

Deep learning is an effective high-level feature extraction and learning approach without *a prior* knowledge. Deep learning methods extract features in terms of complex ADL representations via different physical positions and orientations. However, since most human activities occur sequentially over a timeframe, the sequence model using the recurrent neural network is effective for activity classification.

A. RNN and LSTM

Recurrent neural networks (RNN) are a special type of artificial neural network capable of learning distinct patterns and long-term dependencies from time-series and sequential data including speech, language, genomes, text and video. Sequential data come in order of observations, rather than a set of observations.
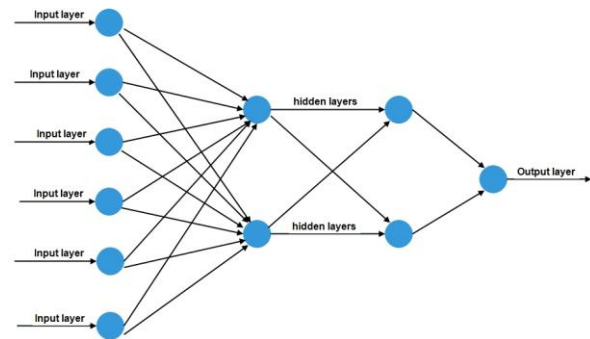

Fig 1. Topology of MLP

The RNN operates as a form of multi-layer perceptron (MLP) network with the addition of loops to the architecture [25]. MLP networks are made of one input layer, one or more hidden layers

and an output layer as illustrated in Fig. 1. In Fig. 1, the MLP consist of six distinct input nodes, two hidden layers with each hidden layer containing two nodes and output is referred to Class. Fig. 2 also illustrates the architecture of a basic RNN where nodes are connected in one direction and the nodes of the hidden layers are connected to the nodes of the input layer.
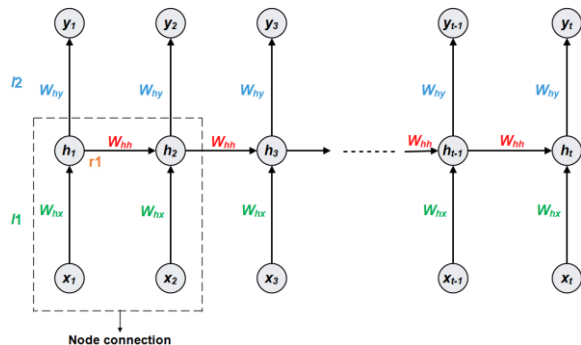

Fig. 2. RNN Architecture

From Fig. 2,

$$h_t = \tanh\left(l(x_t) + r(h_{t-1})\right) \quad (1)$$

$$y_{(t)} = l2(h_t) \quad (2)$$

Eq. (1) and (2) shows that the output at $t = n$ is dependent on the inputs at $t = n$, $t = n-1$ and $t = 1$. This implies that the predicted class depends on the sequence of sampled data rather than a single activity data and each hidden layer depends on the preceding hidden layer and the input layer. The structure of the hidden states operates as the memory of the network and the state of the hidden layer at each time step is conditioned on its previous state [26]. The RNN consists of hidden states of high dimensionality with nonlinear dynamics [27]. The performance of RNNs to learn long-term dependencies from time-series and sequential data is affected by the vanishing problem where the gradient of the loss function decays exponentially with time. However, to overcome the vanishing problem, and retain memory, the long short-term memory (LSTM) is a more effective method. The LSTM includes a set of gates that controls when information enters the memory and maintains it for an extended time. The LSTM modifies the structure of hidden units from *tanh* or *sigmoid* to memory cells where the inputs and outputs are controlled by gates [28]. LSTM cells are effective to learn temporal representations from statistical features for activity classification. The recurrent connections in LSTM add state or memory to the network, which allows the RNN to learn and harness the ordered nature of the observations within input sequences. It is worth noting that the LSTM employs two inputs, output from the lower layer and from a previous time step in the current layer. The configuration operates with the assumption that the current state depends on the state of the previous time step [29]. This assumption of time-dependency often constrains the modelling capability of the RNN. However, in the present work, the current state of the LSTM RNN is adjusted via the RNN learning process to depend on multiple states of different timestamps which results in improved time-dependency modeling capability.

B. LSTM Architecture

The LSTM model has evolved into several architectures with different improvements and limitations. A comparison of different LSTM model architectures is presented in Table 1. Moreover, readers are referred to [28] for detailed knowledge of the different LSTM model architectures. However, the present work focus on the LSTM model due to its effective computational cost for activity classification problem.

TABLE 1. COMPARISON OF VARIOUS LSTM ARCHITECTURES

LSTM Model Architecture	Features	Limitations
Bi-directional LSTM	support forward and backward leaning of the input sequence, which is an edge over standard and S-LSTM	forward and backward leaning often results in increased computational complexity than LSTM
Differential LSTM	effective on Spatio-temporal pattern and discriminates between non-salient and salient features in a sequence	use of differential operators in the model increases its computational complexity than LSTM
Frequency-Time LSTM	improved performance in both frequency and time-based information	number of parameters required to model frequency and time increases computational complexity in comparison with LSTM
Grid LSTM	model multi-dimensional sequence of increased grid dimension	the model requires higher memory and multiple recurrent connections increase computational complexity than LSTM
Local-Global LSTM	improved performance on local and global contextual pattern in a sequence	frequency of parameters for the local and global representations increases computational complexity in comparison with LSTM
LSTM	computational cost is low and effective for long-term dependencies. Robust to overcome the vanishing gradient problem than RNN	model requires higher memory with increasing memory cells
Matching LSTM	optimises LSTM for NLP	increased computational complexity due to word-to-word matching of hypothesis and premise
Multi-dimensional LSTM	effective on multidimensional sequence	the model becomes unstable with increasing grid size and depth, resulting in higher computational complexity than LSTM
S-LSTM	effective on complicated inputs that standard LSTM	the model requires higher computational cost than LSTM
Stacked LSTM	the model supports long-term sequential dependencies due to deeper architecture	Stacking of LSTM cells requires higher memory and subsequently increased computational complexity than LSTM

The architecture of the LSTM RNN model as a folded network, with two fully connected layers is illustrated in Fig. 3.
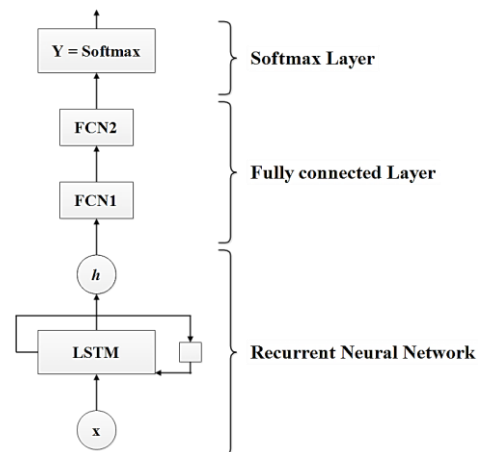

Fig. 3. LSTM Architecture (Folded Network)

The FCN layers and softmax layer are only applied at the end of the recurrence to generate sequence activity predictions. This implies that the intermediate outputs of the recurrent network are not used, as shown in the unrolled LSTM architecture in Fig. 4.
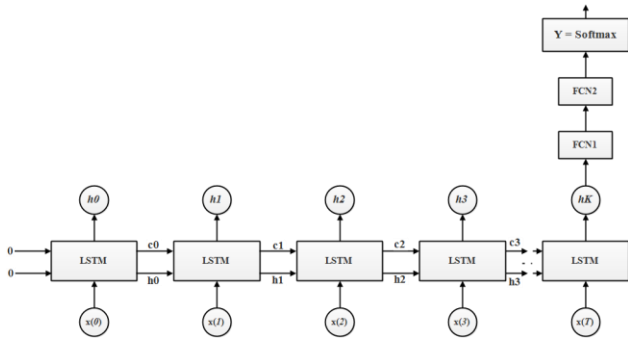
Fig. 4. Unrolled LSTM RNN where FCN layers with softmax are applied for activity prediction

C. LSTM layer

In Fig. 4, $x(T)$ represents the total sequence of input, o is the output whilst the LSTM cell is the computational cell that maps the input to the output. The computational activities that operate within each LSTM unit is illustrated in Fig. 5.
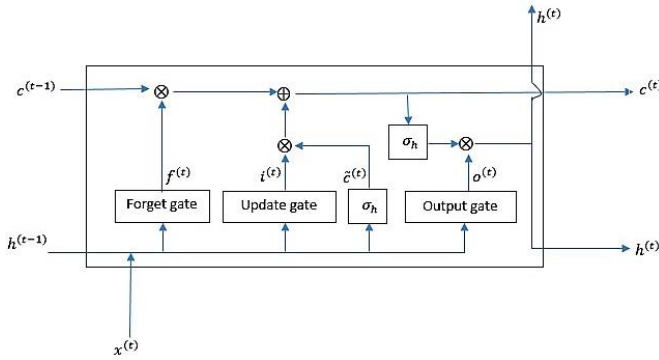

Fig. 5. Architecture of the LSTM unit

Fig. 5 illustrates the architecture of the LSTM unit. The LSTM unit consists of a self-connected memory cell $c^{(t)}$ and three distinct gates; the *forget* gate $f^{(t)}$ that controls the removal of the previous state, an *update* gate $i^{(t)}$ which controls the storage and update of input data and an output gate $o^{(t)}$ which generates the output results.

For any given timestamp t, of input data, $x^{(t)}$ and output $h^{(t)}$ generated, the LSTM activations are calculated from the expressions in Eq. (3) – (8)

$$f^{(t)} = \sigma_g\left(W_f x^{(t)} + U_f h^{(t-1)} + b_f\right) \tag{3}$$

$$i^{(t)} = \sigma_g\left(W_i x^{(t)} + U_i h^{(t-1)} + b_i\right) \tag{4}$$

$$o^{(t)} = \sigma_g\left(W_o x^{(t)} + U_o h^{(t-1)} + b_o\right) \tag{5}$$

$$\tilde{c}^{(t)} = \sigma_h\left(W_c x^{(t)} + U_c h^{(t-1)} + b_c\right) \tag{6}$$

$$c^{(t)} = i^{(t)} \otimes \tilde{c}^{(t)} + f^{(t)} \otimes c^{(t-1)} \tag{7}$$

$$h^{(t)} = o^{(t)} \otimes \sigma_h\left(c^{(t)}\right) \tag{8}$$

Therefore, the parameters to be learned in the LSTM cell are the weights and biases in the sampled data, which includes b_f, b_i, b_o, W_f, U_f, W_i, U_i, W_o, U_o.

D. Fully connected layers

In Fig. 5, the output from the LSTM is fed into two fully connected layers; FCN1 and FCN2. The two FCNs are used to reduce the dimensions of the features to a $1 \times m$ vector, where m denotes the number sampled activities. The computation within the two FCN layers are as follows:

$$a_{fcn1} = \sigma_{\text{Re}LU}\left(W_{fcn1}\ h^{(t)} + b_{fcn1}\right) \tag{9}$$

$$a_{fcn2} = W_{fcn2} \cdot a_{fcn1} + b_{fcn2} \tag{10}$$

It is worth noting that the parameters to learn in the FCN1 layer are W_{fcn1}, b_{fcn1} and W_{fcn2}, b_{fcn2} in the FCN2 layer. Moreover, the length of the vector a_{fcn2} is twelve, which represents the number of sampled classes. Each value in a_{fcn2} corresponds to the denormalized probability that the input data belongs to the class that has the same index as its identity. This show that the first value of a_{fcn2}, i.e. $a_{fcn2}(0)$, denotes the non-normalised probability that the input is from class_0 and similar conditions holds for all sampled activities.

E. SoftMax Layer

The output of the second fully connected layer, FCN2 layer is fed into the SoftMax layer (Y). The SoftMax layer converts the denormalized probabilities in a_{fcn2} to a probability distribution function expressed in the form:

$$Y = Softmax\left(a_{fcn2}\right) = \left[\frac{a_{fcn2}(0)}{\sum_{j=0}^{j=11} a_{fcn2}(j)}, \frac{a_{fcn2}(1)}{\sum_{j=0}^{j=11} a_{fcn2}(j)}, \ldots, \frac{a_{fcn2}(12)}{\sum_{j=0}^{j=11} a_{fcn2}(j)}\right]^T \tag{11}$$

Since the SoftMax acts on each element of a_{fcn2}, Y and is of equal size as a_{fcn2}. Therefore,

$$\sum_{j=0}^{j=11} Y(j) = 1 \tag{12}$$

From Eq. (12), the predicted class is expressed as an index in Y where the highest probability value and the observation is expressed as:

Observation: *Predicted Class = Argmax (Y)*

IV. EXPERIMENTAL EVALUATION

The ambient, experimental sampling approach for various ADLs is achieved via the SmartWall [9]. Fig. 6 illustrates the experimental setup for data acquisition using the SmartWall. The SmartWall is made from thick rubber cladding of dimension 1220×2440 mm^2. The rubber cladding is covered with fabricated passive RFID tags arranged in a grid of dimension 19×12 for the experiment. The RFID tags are mounted on the surface of the SmartWall. Each tag is positioned symmetrically on the surface of the wall 3m apart in both x and z-axis as shown in Fig. 6. The Impinj RAIN RFID multi-reader version 6.6.13.240 with octane firmware is used for the read the HID passive RFID tags. The experiments are conducted within two mock rooms B3.03 and B3.26 of the Biomedical and Electronics Department, University of Bradford. Each mock room layout is of dimensions 6.5m × 4.7m × 3.2m, with room B3.03 containing a kitchenette where some activities including prepares breakfast and brush teeth are performed.
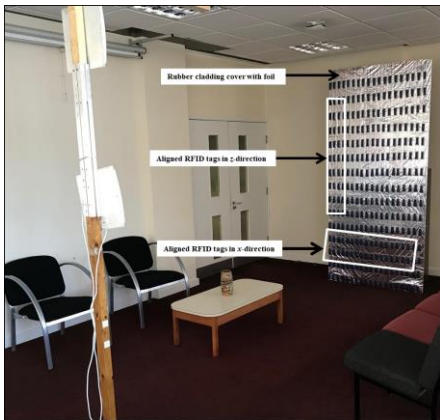
Fig. 6. Experimental Setup using the SmartWall

Two RFID reader antennas are vertically placed with the height of the upper antenna at 1.7m, whilst the lower antenna is mounted 1m to the ground. The transmitted power of the reader antenna is set to -31.5dBm. Each HID RFID tag consists of a meander antenna of dimension $88 \times 37 \times 15$ mm^3 integrated with Alien Monza 4E chip of impedance $14 + j135$ Ω. The distance between the SmartWall and the RFID reader antenna is set to 5m to achieve accurate readings. Different ADL is performed within and around the 5m radius of the SmartWall and the reader antenna and readings are recorded.

A. Dataset

The generated dataset involves a series of RSSI readings of passive RFID tags. Each activity was performed by the volunteers 2.5m in front of the SmartWall. The sampled activities were carried out in terms of height and width orientation. A description of each sampled human activity is presented in Table 2.

TABLE 2. DESCRIPTION OF SAMPLED HUMAN ACTIVITY FOR CLASSIFICATION

Activity_ID	Activity Description
S0	Target runs in and out of the room to the corridor
S1	Target sits 1m away from the wall
S2	Target uses the bathroom
S3	Target stands 1m away from the wall
S4	Target sits randomly at different locations in the room
S5	Target walks in and out of the room
S6	Target lay on the sofa
S7	Target falls
S8	Target brushes teeth
S9	Target sleeps in bed
S10	Target prepares food
S11	No target in the room

It is worth stating that the SmartWall is moved to different locations of the mock room to perform specific activities to achieve high precision. For example, the SmartWall is placed by the door of the mock to sample activities S0 and S5. For the more complex activity S2, the SmartWall is placed 2.5m from the predefined position of the activity. However, each activity is performed in ten iterations taken within a 60-secs window and a total of 120 data files is obtained.

B. Feature Extraction

To employ the deep learning technique using the LSTM model, features of the dataset is selected to produce refined information of each sampled activity. Three distinct features of EPC_values (tag_id), RSSI and timestamp are selected. The selected features of EPC values (tag id), RSSI and timestamp are used as each feature varies with each sampled activity. It is worth noting that timestamp is critical to developing HAR systems for both discrete and continuous-time human activities.

C. Feature Extraction and Data Cleanup

To achieve data reliability, the dataset is pre-processed to clean any corrupt data caused by possible hardware failures during data transfer. To clean the data for further processing, the following procedures are performed:

Step 1: The EPC_values are mapped to unique integer_ID such that the integer ranges from 1 to 228 (which corresponds to the total number of UHF passive RFID tags used in the experiment). Therefore, each tag ID is given a unique integer_ID whilst other tag IDs are similarly mapped.

Step 2: The timestamps are uniquely mapped to integer_ID. The integer_ID given to the timestamps are in ascending order i.e. for each activity dataset, the lowest timestamp value takes the id 0 and the next timestamps takes id 1 etc.

Step 3: Finally, the RSSI values for all tag IDs at each timestamp is collected to form an RSSI vector. Each vector is of length 228 (which corresponds to the total number of tags).

It is worth noting that $x^{(t)}$ represents an RSSI vector at timestamp t, where the vector length is 228, which corresponds to the total number of tag_ID in the array. This implies that each activity dataset is represented as a set of RSSI vectors. The number of RSSI vectors in each activity dataset depends on the total number of timestamps in the data file. However, the RSSI vector is not constant across each activity as it varies, with the maximum power found to be around -34dBm whilst is -25dBm across the entire dataset. Subsequently, each activity dataset represents a sequence of RSSI vectors $x^{(t)}$ and is denoted as X^K where K is the total number of timesteps. Therefore,

$$X^K = \{x^{(0)}, \ x^{(1)}, \ x^{(2)}, \ x^{(3)}, \ ...., \ x^{(K)}\} \quad (13)$$

Each X^K is padded with zero to have the same size of $K = 34$ and is expressed as:

$$X^T \in \mathbb{R}^{Q \ x \ K} \quad (14)$$

where $Q = 286$ and represents the size of the RSSI vector $x^{(t)}$

From Eq. (14), X^T represents a single activity training dataset. The activities are given unique integers from 0 to 11 and each integer is then converted to one-hot encoding as presented in Table 3. It is worth noting that the one-hot representation is used for the computation of the cross-entropy loss.

6

TABLE 3. ACTIVITY_ID WITH CORRESPONDING ONE-HOT REPRESENTATION

Activity_ID	One-Hot representation
S0	[**1**,0,0,0,0,0,0,0,0,0,0,0]
S1	[0,**1**,0,0,0,0,0,0,0,0,0,0]
S2	[0,0,**1**,0,0,0,0,0,0,0,0,0]
S3	[0,0,0,**1**,0,0,0,0,0,0,0,0]
S4	[0,0,0,0,**1**,0,0,0,0,0,0,0]
S5	[0,0,0,0,0,**1**,0,0,0,0,0,0]
S6	[0,0,0,0,0,0,**1**,0,0,0,0,0]
S7	[0,0,0,0,0,0,0,**1**,0,0,0,0]
S8	[0,0,0,0,0,0,0,0,**1**,0,0,0]
S9	[0,0,0,0,0,0,0,0,0,**1**,0,0]
S10	[0,0,0,0,0,0,0,0,0,0,**1**,0]
S11	[0,0,0,0,0,0,0,0,0,0,0,**1**]

V. TRAINING

A. Training Setup

To train the RNN model, the stochastic gradient descent optimiser is employed. The objective function used in the gradient descent algorithm is to minimise the loss function. The training loss function used in the present multi-class classification problem is the cross-entropy loss and is expressed in the form:

$$L(Y_{actual}, Y) = -\sum_{j=0}^{j=11} Y_{actual}(j) \cdot \log(Y(j))$$

(15)

where Y_{actual} represents the one-hot encoding label for the training dataset.

Eq. 15 gives the cross-entropy loss on a single training dataset. However, the total loss function over the entire training dataset is the sum of the cross-entropy loss for the entire training dataset expressed as:

$$J_{total} = \frac{1}{m}\sum_{j=0}^{j=m} L(Y_{actual}, Y)$$

(16)

where m is the number of sampled activities per training batch data, J_{total} represents the total loss function, that is, the function of all weights and biases, i.e. learnable parameters in the network. Therefore, Eq. 16 is further expressed as:

$$J_{total}(b_f, b_o, b_i, U_f, W_f, U_i, W_i, U_o, W_o, b_{fcn1}, W_{fcn1}, b_{fcn2}, W_{fcn2} = \sum_{j=0}^{j=nTrainingset} L(Y_{actual}, Y)$$

(17)

Eq. 17 is the objective/optimization function of the network and the value of the weights and biases minimises Eq. 16. Moreover, the gradient descent algorithm is used to minimise the objective function.

The dataset is split into training and test dataset according to the *iteration windows* approach. The first seven iteration windows for all sampled activities are selected for training the model and the last three iteration windows are used as test data. Since the RSSI value across the entire dataset is between 29dB and 31dB. The datasets are normalised and 30dB is selected as the normalised value. The normalised value is built into the batch loader to achieve accurate prediction. The datasets are loaded in batches and backpropagation is carried out on the batch dataset. The training is run on a Microsoft machine with Intel® Core™ i7-7500U CPU @ 2.90GHz and NVIDIA GeForce 930MX. The computational graph of the RNN model is defined in TensorFlow [30].

During training, two key metrics of minibatch loss and minibatch accuracy are observed. The minibatch loss gives the cross-entropy loss of the minibatch, whilst the minibatch accuracy gives the percentage accuracy the RNN model has on the batch dataset after backpropagation of its gradient, i.e. the RNN model is trained by optimising the total loss function. To overcome the overfitting problem, the hyperparameters are randomly tuned and optimised with the stochastic descent method to minimise the training loss. The dropout rate for each hidden layer is defined. The dropout allows a single dropout mask to be generated at the beginning of each training sequence and it adjusts over the duration of the sequence. The dropout rate is set to 0.5 at the FCN1 layer. Table 4 illustrates the hyperparameters used to train the RNN model.

TABLE 4. TESTING PARAMETERS

Hyperparameters	Value
Training iteration	700
Learning rate, α	0.0001
Max sequence length	34
Q	286
fcn1	100
fcn2	100
K	34
Num_Classes	12

The performance of the RNN model is evaluated on the dataset after each epoch of training. The training loss for training the datasets is presented in Fig. 7.
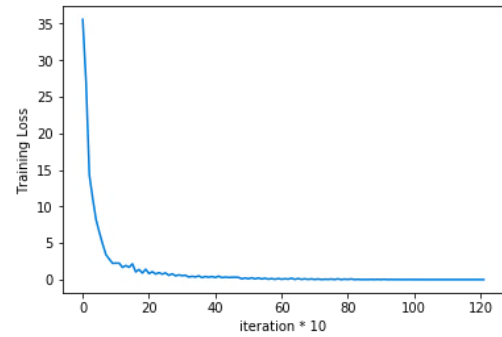

Fig. 7. Training loss curve

From Fig. 7, the LSTM RNN model achieves a training loss of 0.01.

VI. ACTIVITY CLASSIFICATION

To examine the generalisation of the LSTM RNN classifier for activity recognition, the model is evaluated on the selected test dataset. Six activity datasets were chosen randomly and fed into the model for classification. The classification of the selected activity test datasets is presented in Fig. 8. In Fig. 8, the LSTM RNN is illustrated to effectively recognise each selected activity class to their correct class with high probability.
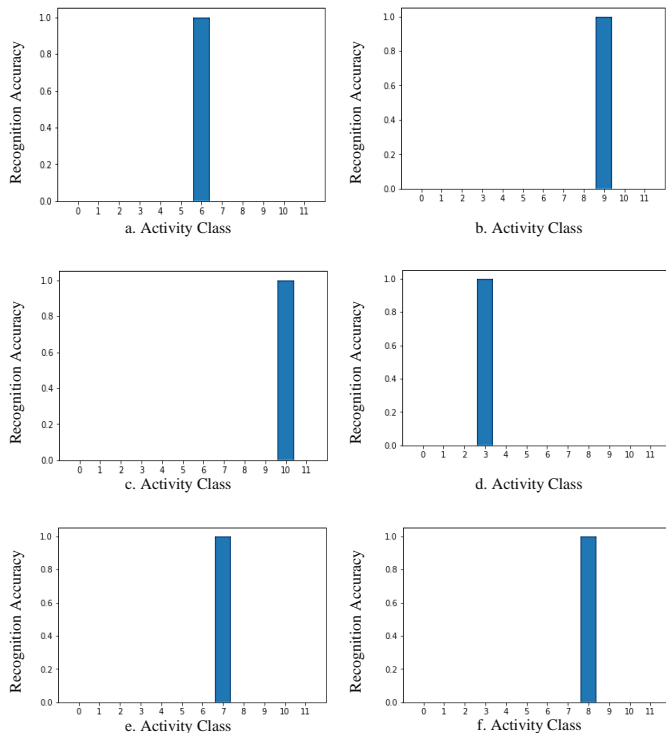
Fig. 8. Recognition accuracy of the LSTM RNN model for selected activities

Moreover, since accuracy as a performance metric expressed as the total correct classifications to the total number of instances, the recognition accuracy of the LSTM RNN classifier for all sampled activities is presented in Fig. 9.

Fig. 9. Confusion matrix of the LSTM RNN model of sampled activities

Fig. 9 illustrates the confusion matrix of the LSTM RNN model. The additional right column and bottom row indicate the accuracy of each actual and predicted activity class. The overall recognition accuracy of the LSTM RNN classifier for the sampled activities is found to be 98.18%, which is indicative that the model effectively learned each activity class. It is worth noting that the LSTM RNN classifier shows higher accuracy for stationary activities including *sitting, standing, laying on the sofa, sleep in bed and fall*. However, the LSTM classifier shows relatively low performance for concurrent and complex activities including *sitting randomly* and *bathing*, as the LSTM classifier, for example, misclassifies sequential random sitting with other stationary activities.

VII. PERFORMANCE EVALUATION

A. Effect of Training Data Size

The correlation between data size and the performance of machine learning and Deep learning algorithm has been a critical consideration to determine the complexity of the model [31-33]. With deep learning, there is ongoing research interest to determine how performance improves with increasing data size in the literature [34-36]. In the same context, the amount of datasets that would result in the optimal recognition performance of the LSTM RNN model and reduce overfitting is investigated. To achieve the evaluation objective, different data size, N is processed by the LSTM model to establish which range of N results in optimal recognition performance in terms of accuracy. The obtained results from the evaluation are presented in Fig. 10.
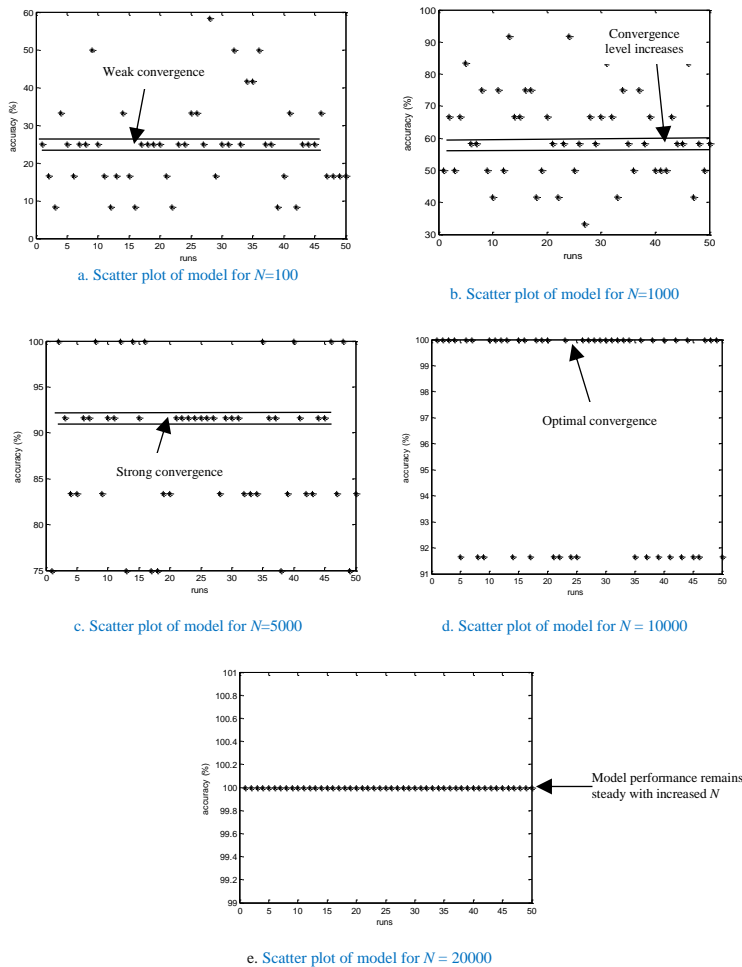
Fig. 10. Performance of LSTM RNN model for varying data size

In Fig. 10, it should be observed that the recognition performance of the LSTM RNN model improves as N increases. The convergence of the data indicates the recognition

accuracy of the LSTM model as N is increased from $N = 100$ (25%), for $N=1000$ (58%), for $N=5000$ (92%). However, the performance of the model reaches an optimal for N>=10000). This shows that the performance of the LSTM model performs better for a certain amount of datasets.

B. Performance Metrics

The performance of the LSTM model is further evaluated using other key metrics, including precision, recall and the F-score. Precision expresses the proportionality of TP to the total predicted true classes and is expressed as:

$$Precision \quad = \quad \frac{TP}{TP + FP} \qquad (18)$$

where TP represents all *true positives and FP* all the *false positive*.

Recall or sensitivity expresses the ratio of accurately predicted true classes to all activities in the true class.

$$Sensitivity \quad = \quad \frac{TP}{TP + FN} \qquad (19)$$

where FN represents *false-negative*.

F-score expresses the classification correctness of the LSTM RNN model. *F-score* estimates the weighted harmonic mean of precision and recall by taking both false positives, FP and FN into account and is usually expressed as:

$$F\text{-}score \quad = \quad 2 * \frac{Recall * Precision}{Recall + Precision} \qquad (20)$$

The overall performance of the LSTM using the three key metrics to achieve the research objective is presented in Table 5. From the results in Table 5, it is shown that the LSTM RNN model effectively classifies the dataset for human activity recognition.

TABLE 5. PERFORMANCE METRICS OF THE LSTM MODEL

Activity	Precision	Sensitivity	F-score
S0	0.984	0.986	0.9850
S1	0.976	0.976	0.9760
S2	0.972	0.978	0.9750
S3	0.990	0.976	0.9830
S4	0.968	0.978	0.9730
S5	0.982	0.963	0.9724
S6	0.986	0.986	0.9860
S7	0.982	0.992	0.9870
S8	0.976	0.984	0.9800
S9	0.988	0.982	0.9850
S10	0.978	0.983	0.9805
S11	0.996	0.996	0.9960

C. Comparison of the LSTM RNN model with other state-of-the-art probabilistic models

The performance of the LSTM RNN model in terms of accuracy is compared with other state-of-the-art algorithms, including random forest, Hidden Markov model (HMM), Naïve Bayes, Hidden Semi-Markov Model (HSMM) and Convolutional neural network (CNN) using the raw dataset. Python software is employed to implement the multiclass classifiers [37]. TensorFlow is used to evaluate the performance of both LSTM and CNN. The Scikit-learn library is used to implement the machine learning functions for Naïve Bayes and Random Forest. The HMM model is implemented using the *hmmlearn* and trained using the Baum-Welch algorithm, and the HSMM is evaluated using the factor adaptive training based on the expectation-maximization algorithm as described by Yamazaki *et al.* [38].

Each state-of-the-art algorithm is trained with an exact dataset. The obtained results are presented in Fig. 11. The obtained results show that the LSTM RNN matched and outperformed several state-of-the-art probabilistic models including random forest, HMM, Naïve Bayes, HSMM with an overall performance of 98.18%. However, the performance of the LSTM RNN and CNN exhibits a greater correlation, which shows both neural network models effectively learned and predicted the dataset. Therefore, the present work further builds on our previous work in [9] on the opportunistic deployment of passive RFID via the SmartWall for human activity recognition. However, compared to our previous work in [9], the present work is evaluated with increased data size to improve the complexity and robustness of the LSTM model for reliable real-time, human activity recognition.
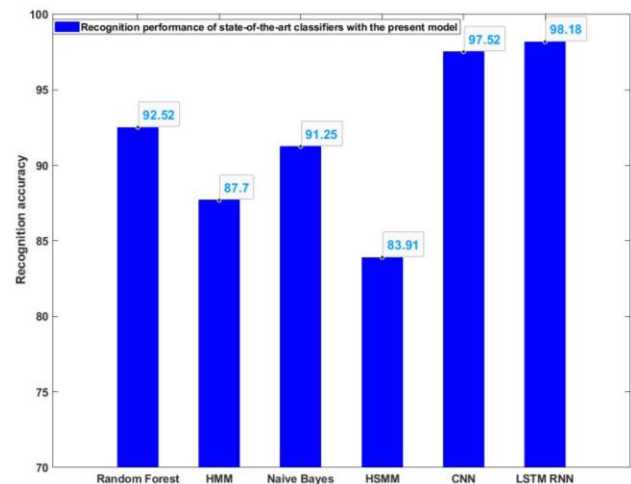

Fig. 11. Comparison of recognition performance of LSTM RNN model with standard state-of-the-art classifiers

VIII. CONCLUSION

In this paper, we have proposed an ambient, human activity classification framework using a two-layer fully connected sequence RNN. The proposed method employs LSTM RNN to extract the high-level features of EPC, timestamps, and RSS. The performance of the LSTM-RNN is evaluated for different data size and the hyper-parameters of the RNN are adjusted to achieve improved classification performance. Using the experimental dataset, the classification accuracy of the LSTM RNN model is found to be 98.18% and outperforms the standard state-of-the-art learning methods of random forest, HMM, Naïve Bayes, HSMM and CNN. With the outcome of the research, the study shows that RFID technology can be opportunistically employed to achieve smart health. Further, emphasis on data size in deep learning indicates that the performance of the LSTM model performs better for a certain

amount of datasets (i.e. $N \geq 10,000$). The future work will concentrate on exploring 3-D data sampling and improved development of the sensing methodology to achieve 3-D human activity profiling. The present activity recognition framework offers intelligent, pervasive sensing and classification approach to support active and assisted living for the elderly, vulnerable and persons with disabilities.

APPENDIX

$\otimes$	Element-wise multiplication
b_f	Forget gate bias vector, i.e. $b_f \in \mathbb{R}^{nhidden \times 1}$
b_i	Update gate bias vector, i.e. $b_i \in \mathbb{R}^{nhidden \times 1}$
b_o	Output gate bias vector, i.e. $b_o \in \mathbb{R}^{nhidden \times 1}$
b_c	Input gate bias vector, $b_c \in \mathbb{R}^{nhidden \times 1}$
$f^{(t)}$	Forget gate activation vector, $f^{(t)} \in \mathbb{R}^{nhidden \times 1}$
$i^{(t)}$	Input gate activation vector, $i^{(t)} \in \mathbb{R}^{nhidden \times 1}$
$o^{(t)}$	Output gate activation vector, $o^{(t)} \in \mathbb{R}^{nhidden \times 1}$
W_f, U_f	Forget gate weight matrix, $W_f, U_f \in \mathbb{R}^{nhidden \times Q}$
W_i, U_i	Update gate weight matrix, $W_i, U_i \in \mathbb{R}^{nhidden \times Q}$
W_o, U_o	Output gate weight matrix, $W_o, U_o \in \mathbb{R}^{nhidden \times Q}$
W_c, U_c	Input gate weight matrix, $W_c, U_c \in \mathbb{R}^{nhidden \times Q}$
σ_h	Hyperbolic tangent activation
σ_g	Sigmoid activation
$h^{(t)}$	Output vector from LSTM, $h^{(t)} \in \mathbb{R}^{nhidden \times 1}$
$c^{(t)}$	LSTM state vector, $c^{(t)} \in \mathbb{R}^{nhidden \times 1}$
a_{fcn1}	Activation vector of FCN1, i.e. $a_{fcn1} \in \mathbb{R}^{nfcn1 \times 1}$
a_{fcn2}	Activation vector of FCN2, i.e. $a_{fcn2} \in \mathbb{R}^{nfcn2 \times 1}$
W_{fcn1}	Weight matrix of FCN1 i.e. $W_{fcn1} \in \mathbb{R}^{nfnc1 \times nhidden}$
W_{fcn2}	Weight matrix of FCN2 i.e. $W_{fcn2} \in \mathbb{R}^{nfnc2 \times nhidden}$
b_{fcn1}	Bias vector of FCN1 i.e. $b_{fcn1} \in \mathbb{R}^{nfcn1 \times 1}$
b_{fcn2}	Bias vector of FCN2 i.e. $b_{fcn2} \in \mathbb{R}^{nfcn2 \times 1}$
σ_{ReLU}	Rectified linear unit activation function
a_{fcn1}	Activation vector of FCN1, i.e. $a_{fcn1} \in \mathbb{R}^{nfcn1 \times 1}$
a_{fcn2}	Activation vector of FCN2, i.e. $a_{fcn2} \in \mathbb{R}^{nfcn2 \times 1}$
W_{fcn1}	Weight matrix of FCN1 i.e. $W_{fcn1} \in \mathbb{R}^{nfnc1 \times nhidden}$

REFERENCES

[1] U. Nations, "World Population Ageing 2015," United Nations, New York, (ST/ESA/SER.A/390), 2015.

[2] M. Orchard, E. Green, T. Sullivan, A. Greenberg, and V. Mai, "Chronic disease prevention and management: implications for health human resources in 2020," (in eng), *Healthcare quarterly (Toronto, Ont.),* vol. 11, no. 1, pp. 38-43, 2008.

[3] G. Oguntala, R. Abd-Alhameed, S. Jones, J. Noras, M. Patwary, and J. Rodriguez, "Indoor location identification technologies for real-time IoT-based applications: An inclusive survey," *Computer Science Review,* vol. 30, pp. 55-79, 2018, doi: 10.1016/j.cosrev.2018.09.001.

[4] C. Lin, Y. C. Yang, J. Wang, and Y. Yang, "A Wearable Sensor Module With a Neural-Network-Based Activity Classification Algorithm for Daily Energy Expenditure Estimation," *IEEE Transactions on Information Technology in Biomedicine,* vol. 16, no. 5, pp. 991-998, 2012, doi: 10.1109/TITB.2012.2206602.

[5] F. Attal, S. Mohammed, M. Dedabrishvili, F. Chamroukhi, L. Oukhellou, and Y. Amirat, "Physical Human Activity Recognition Using Wearable Sensors," *Sensors,* vol. 15, no. 12, pp. 31314-31338, 2015. [Online]. Available: http://www.mdpi.com/1424-8220/15/12/29858.

[6] N. Shibuya *et al.*, "A real-time fall detection system using a wearable gait analysis sensor and a Support Vector Machine (SVM) classifier," in *2015 Eighth International Conference on Mobile Computing and Ubiquitous Networking (ICMU)*, 2015, pp. 66-67, doi: 10.1109/ICMU.2015.7061032.

[7] R. Damasevicius, M. Vasiljevas, J. Salkevicius, and M. Wozniak, "Human Activity Recognition in AAL Environments Using Random Projections," *Computational and Mathematical Methods in Medicine,* vol. 2016, p. 17, 2016, Art no. 4073584, doi: 10.1155/2016/4073584.

[8] F. Li, K. Shirahama, M. A. Nisar, L. Köping, and M. Grzegorzek, "Comparison of Feature Learning Methods for Human Activity Recognition Using Wearable Sensors," *Sensors,* vol. 18, no. 2, p. 679, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/2/679.

[9] G. A. Oguntala *et al.*, "SmartWall: Novel RFID-Enabled Ambient Human Activity Recognition Using Machine Learning for Unobtrusive Health Monitoring," *IEEE Access,* vol. 7, pp. 68022-68033, 2019, doi: 10.1109/ACCESS.2019.2917125.

[10] M. R. Alam, M. B. I. Reaz, and M. A. M. Ali, "A Review of Smart Homes—Past, Present, and Future," *Trans. Sys. Man Cyber Part C,* vol. 42, no. 6, pp. 1190–1203, 2012, doi: 10.1109/tsmcc.2012.2189204.

[11] F. K. Aldrich, "Smart Homes: Past, Present and Future," in *Inside the Smart Home*, R. Harper Ed. London: Springer London, 2003, pp. 17-39.

[12] J. Bennett, O. Rokas, and L. Chen, "Healthcare in the Smart Home: A Study of Past, Present and Future," *Sustainability,* vol. 9, no. 5, p. 840, 2017. [Online]. Available: https://www.mdpi.com/2071-1050/9/5/840.

[13] G. Oguntala *et al.*, "Design framework for unobtrusive patient location recognition using passive RFID and particle filtering," in *2017 Internet Technologies and Applications, ITA 2017 - Proceedings of the 7th International Conference*, 2017 2017, pp. 212-217, doi: 10.1109/ITECHA.2017.8101941. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-85041745724&partnerID=MN8TOARS

[14] J. Esch, "A Survey on Ambient Intelligence in Healthcare," *Proceedings of the IEEE,* vol. 101, no. 12, pp. 2467-2469, 2013, doi: 10.1109/JPROC.2013.2286654.

[15] G. Acampora, D. J. Cook, P. Rashidi, and A. V. Vasilakos, "A Survey on Ambient Intelligence in Healthcare," *Proceedings of the IEEE,* vol. 101, no. 12, pp. 2470-2494, 2013, doi: 10.1109/JPROC.2013.2262913.

[16] M. Ramkumar, S. S. Catharin, and D. Nivetha, "Survey of Cognitive Assisted Living Ambient System Using Ambient intelligence as a Companion," in *2019 IEEE International Conference on System, Computation, Automation and Networking (ICSCAN)*, 29-30 March 2019 2019, pp. 1-5, doi: 10.1109/ICSCAN.2019.8878707.

[17] P. Rashidi and A. Mihailidis, "A Survey on Ambient-Assisted Living Tools for Older Adults," *IEEE Journal of Biomedical and Health Informatics,* vol. 17, no. 3, pp. 579-590, 2013, doi: 10.1109/JBHI.2012.2234129.

[18] J. C. Augusto, "Ambient intelligence: opportunities and consequences of its use in smart Classrooms," *Innovation in Teaching and Learning in Information and Computer Sciences,* vol. 8, no. 2, pp. 53-63, 2009/06/01 2009, doi: 10.11120/ital.2009.08020053.

[19] F. Sadri, "Ambient intelligence: A survey," *ACM Comput. Surv.,* vol. 43, no. 4, p. Article 36, 2011, doi: 10.1145/1978802.1978815.

[20] T. Magherini, A. Fantechi, C. D. Nugent, and E. Vicario, "Using Temporal Logic and Model Checking in Automated Recognition of Human Activities for Ambient-Assisted Living," *IEEE Transactions on Human-Machine Systems,* vol. 43, no. 6, pp. 509-521, 2013, doi: 10.1109/TSMC.2013.2283661.

[21] A. Jayatilaka, Y. Su, and D. C. Ranasinghe, "HoTAAL: Home of social things meet ambient assisted living," in *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, 14-18 March 2016 2016, pp. 1-3, doi: 10.1109/PERCOMW.2016.7457073.

[22] J. Wan, M. Li, M. O. Grady, X. Gu, M. Alawlaqi, and G. O. Hare, "Time-bounded Activity Recognition for Ambient Assisted Living,"

IEEE Transactions on Emerging Topics in Computing, pp. 1-1, 2018, doi: 10.1109/TETC.2018.2870047.

[23] M. Biagi, L. Carnevali, M. Paolieri, F. Patara, and E. Vicario, "A Continuous-Time Model-Based Approach for Activity Recognition in Pervasive Environments," *IEEE Transactions on Human-Machine Systems,* vol. 49, no. 4, pp. 293-303, 2019.

[24] A. C. B. Garcia, A. S. Vivacqua, N. Sánchez-Pi, L. Martí, and J. M. Molina, "Crowd-Based Ambient Assisted Living to Monitor the Elderly's Health Outdoors," *IEEE Software,* vol. 34, no. 6, pp. 53-57, 2017, doi: 10.1109/MS.2017.4121217.

[25] N. D. Lewis, "MLP in a Nutshell," in *Build Your Own Neural Network Today!*: CreateSpace Independent Publishing Platform, 2015, ch. 1, p. 224.

[26] T. Mikolov, A. Joulin, S. Chopra, M. Mathieu, and M. A. Ranzato, "Learning Longer Memory in Recurrent Neural Networks," presented at the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, 2015. [Online]. Available: http://arxiv.org/abs/1412.7753.

[27] I. Sutskever, J. Martens, and G. Hinton, "Generating text with recurrent neural networks," presented at the Proceedings of the 28th International Conference on International Conference on Machine Learning, Bellevue, Washington, USA, 2011.

[28] H. Salehinejad, S. Sankar, J. Barfett, E. Colak, and S. Valaee, "Recent advances in recurrent neural networks," *arXiv preprint arXiv:1801.01078,* 2017.

[29] F. Tao and G. Liu, "Advanced LSTM: A Study About Better Time Dependency Modeling in Emotion Recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 15-20 April 2018 2018, pp. 2906-2910, doi: 10.1109/ICASSP.2018.8461750.

[30] *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*, White Paper Martín Abadi *et al.*, USA, 2016.

[31] B. Juba and H. S. Le, "Precision-Recall versus Accuracy and the Role of Large Data Sets," in *Thirty-Third AAAI Conference on Artificial Intelligence: AAAI Technical Track: Machine Learning*, Honolulu, Hawaii, USA, 2019, vol. 33: AAAI Press, doi: https://doi.org/10.1609/aaai.v33i01.33014039.

[32] X. Zhu, C. Vondrick, C. C. Fowlkes, and D. Ramanan, "Do We Need More Training Data?," *International Journal of Computer Vision,* vol. 119, no. 1, pp. 76-92, 2016/08/01 2016, doi: 10.1007/s11263-015-0812-2.

[33] A. K. Jain and B. Chandrasekaran, "39 Dimensionality and sample size considerations in pattern recognition practice," in *Handbook of Statistics*, vol. 2: Elsevier, 1982, pp. 835-855.

[34] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting Unreasonable Effectiveness of Data in Deep Learning Era," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 22-29 Oct. 2017 2017, pp. 843-852, doi: 10.1109/ICCV.2017.97.

[35] A. Joulin, L. v. d. Maaten, A. Jabri, and N. Vasilache, "Learning Visual Features from Large Weakly Supervised Data," in *ECCV*, 2016.

[36] M. Olson, A. J. Wyner, and R. Berk, "Modern neural networks generalize on small data sets," presented at the Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montréal, Canada, 2018.

[37] *About Python*. Accessed: 20 July 2020. [Online]. Available: https://www.python.org/

[38] T. Yamazaki, N. Niwase, J. Yamagishi, and T. Kobayashi, "Human walking motion synthesis based on multiple regression hidden semi-Markov model," in *2005 International Conference on Cyberworlds (CW'05)*, 23-25 Nov. 2005 2005, pp. 8 pp.-452.