Computers, Materials & Continua
DOI:10.32604/cmc.2021.015660
Article

Tech Science Press

Adapted Long Short-Term Memory (LSTM) for Concurrent Human Activity Recognition

Keshav Thapa, Zubaer Md. Abdhulla AI and Yang Sung-Hyun*

Department of Electronic Engineering, Kwangwoon University, Seoul, 139-701, Korea
*Corresponding Author: Yang Sung-Hyun. Email: shyang@kw.ac.kr
Received: 01 December 2020; Accepted: 14 April 2021

Abstract: In this era, deep learning methods offer a broad spectrum of efficient and original algorithms to recognize or predict an output when given a sequence of inputs. In current trends, deep learning methods using recent long short-term memory (LSTM) algorithms try to provide superior performance, but they still have limited effectiveness when detecting sequences of complex human activity. In this work, we adapted the LSTM algorithm into a synchronous algorithm (sync-LSTM), enabling the model to take multiple parallel input sequences to produce multiple parallel synchronized output sequences. The proposed method is implemented for simultaneous human activity recognition (HAR) using heterogeneous sensor data in a smart home. HAR assists artificial intelligence in providing services to users according to their preferences. The sync-LSTM algorithm improves learning performance and sees its potential for real-world applications in complex HAR, such as concurrent activity, with higher accuracy and satisfactory computational complexity. The adapted algorithm for HAR is also applicable in the fields of ambient assistive living, healthcare, robotics, pervasive computing, and astronomy. Extensive experimental evaluation with publicly available datasets demonstrates the competitive recognition capabilities of our approach. The sync-LSTM algorithm improves learning performance and has the potential for real-life applications in complex HAR. For concurrent activity recognition, our proposed method shows an accuracy of more than 97%.

Keywords: Concurrent HAR; deep learning; LSTM; sync-LSTM; smart home

1 Introduction

Activity recognition has been one of the core topics in the field of artificial intelligence (AI) research for decades. However, the various features and classifiers designed for specific activity recognition are still not satisfactory for measuring the effectiveness of detecting complex human activities. Activity recognition has started to adopt a more complex structure and inference procedures due to deep learning methods [1]. These deep learning approaches have demonstrated the potential to significantly improve the state of the art in human activity recognition (HAR) [2]. The

complete recognition process includes sensor data, pre-processing, feature extraction, and classifiers (model), where feature extraction and classifier algorithms play a crucial role in achieving high recognition accuracy [3]. Manual feature extraction or selection is often laborious and arbitrary and lacks generalizability [4]. Instead of using standard produced features, we applied automatic feature extraction or selection [5,6]. Recent activity recognition studies have often focused on single individuals or single activities rather than complex, concurrent activities. Our method addresses the challenge of concurrent activity detection.

The need to structure information in order to process a large amount of data is becoming a pervasive problem and gaining high research interest. Automatic sequence classification based on convolutional neural network (CNN) [7] and recurrent neural network (RNN) [8] methods are used to structure and process these sequences with a set of high-level representations. Deep learning methods such as long short-term memory (LSTM) [9] and bi-directional LSTM (BLSTM) [10] are used in various domains and tasks, including image processing [11], language recognition [12], and sentence processing [13].

Despite its popularity, LSTM is unable to take multiple inputs for synchronous data and cannot produce multiple outputs directly when used as a classifier. In [14], a multiple stream neural network called parallel LSTM is presented, which can process only the synchronous data stream to produce a single output. In this paper, we present an improved version of the architecture of a synchronous neural network called synchronous LSTM (sync-LSTM), which processes multiple data streams simultaneously to detect concurrent human activity. Two or more activities running simultaneously in parallel are called concurrent activities. Most importantly, we detect complex human activity using the improved deep learning platform. Multi-binary classifiers can detect parallel activity [15,16], but this fails for many activities. The hidden Markov model (HMM) [17], condition random field [18], and various other types of machine learning approaches [19] and probability inference algorithms [20] are widely used for parallel activity detection. However, they cannot handle a large number of spatio-temporal data sequences. The method proposed here is the first so far to address these issues.

Two requirements for human activity recognition drive this work: improving recognition accuracy and developing reliable algorithms to solve complex activity recognition problems. Therefore, our method promises to address the needs of activity recognition with heterogeneous sensors, primarily by improving performance over existing approaches. The highlights of our method are as follows.

- We present the improved LSTM known as sync-LSTM to recognize concurrent human activity.
- The presented concept is an adapted version of LSTM that supports parallel input and parallel output.
- The proposed approach can structure and learn spatio-temporal features directly and automatically from the raw sensor data, without the need for manual feature extraction or selection for concurrent activity recognition.
- This framework can likely be applied to different recognition platforms with different sensor modalities in different domains.
- The results obtained by our proposed method show that it outperforms existing methods on recognition of concurrent activity.

Activity recognition relies on a combination of different sensors: wearable, external, or both. We chose external sensors for the user's comfort. We evaluate and compare the performance of

our proposed approach using fully annotated real-world datasets generated by Kasteren and the Center for Advanced Studies in Adaptive Systems (CASAS). The rest of this paper is organized as follows. Related work on activity recognition and LSTM is described in Section 2. Section 3 illustrates our proposed method. The experiment setup, analysis, and evaluation are provided in Section 4. Finally, Section 5 concludes the paper.

2 Related Work

Early research trained shallow depth classifiers on sequences collected by a single sensor. Sensor-specific constraints overly controlled this single sensor-based system, as the data from this type of sensor was inherently insufficient for complex activity recognition. Therefore, a multi-sensor heterogeneous system was designed to address these challenges. Significantly, our system can work with multiple and heterogeneous sensors. A naive Bayes classifier is used for activity recognition with the longest sequences of sensor data sequences [21]. An incremental learning approach called the dynamic Bayesian network is tested to detect differences in activities by rebuilding the previously learned models [22]. Early generative deep learning methods used restricted Boltzmann machines to derive task-independent feature representations [23,24]. More complex models such as CNN have been successfully used for challenging HAR tasks [25]. Similarly, some decent algorithms, such as multilayer perceptron [26], support vector machine [27], decision tree [28], and an updated HMM [29], are used in classifying some types of activities.

The discriminative models have used the independence assumption, where we learn the model parameters by optimizing the conditional likelihood rather than the joint likelihood [30]. Audio [31] and video [32,33] activity recognition methods are also widely used and best suited for healthcare and remote monitoring. However, audio-visual methods have privacy issues and are complex and pervasive. In recent years, many deep learning methods have been used to work with CNN [34,35], RNN [36], and LSTM. The enhanced RNN and LSTM are widely applicable in language modeling, hand gesture recognition [37], machine translation [38], sound recognition [39], video analysis [40], and image captioning [41]. As mentioned earlier, they lag behind in parallel processing. Additionally, these algorithms are least common in activity recognition. However, a significant amount of research is concerned with single or regular activity recognition, and few researchers are interested in complex activity recognition such as concurrent activity recognition. Activity recognition with a CNN-LSTM [42] structure sheds light on concurrent activity detection with multimodal sensor data, but it suffers from data redundancy.

3 Materials and Method

The HAR process consisted of four phases: data acquisition, preprocessing, feature extraction, and training/testing, as shown in Fig. 1. In this work, concurrent activity is recognized by using Sync-LSTM. Therefore, the experimental process is divided into preprocessing and training/testing, as LSTM automatically acquires the feature vectors.

The sensor data is pre-processed by applying filters and overlapping sliding windows of 128 time steps each. Our model operates with two fully connected and two LSTM layers of 64 units each, as shown in Fig. 2. We train the model and predict the real value by setting hyperparameters such as entropy, learning rate, weight decay, and optimizer. A k-fold cross-validation technique is performed to improve and validate the output before testing. During testing, a test sensor is added without affecting the learned parameters. The detection obtained during testing is then compared with the real values, and the accuracy is calculated using the F1 score determined from HAR. We present the sync-LSTM based on a standard LSTM and describe its implementation

for concurrent activity recognition. The main feature of our approach is that it can accept multiple inputs and produce multiple outputs by improving the LSTM algorithm to handle parallel structure and detect concurrent activity. The input sequences are independent and mapped into homogeneous subsets. A single-mode LSTM with concatenated input sequences is theoretically not suitable for mapping the heterogeneous input representation of parallel sequences.

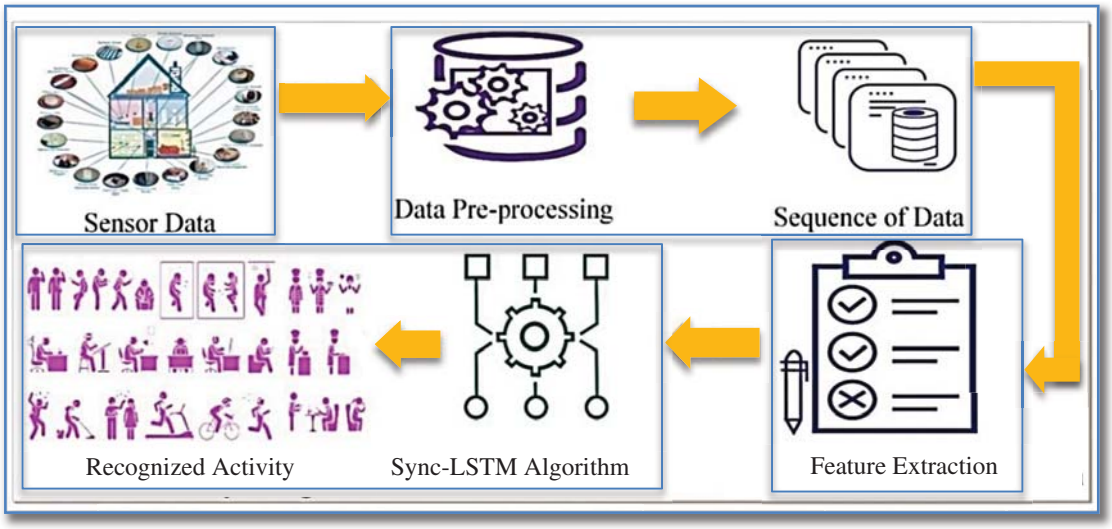


Figure 1: General system workflow of human activity recognition

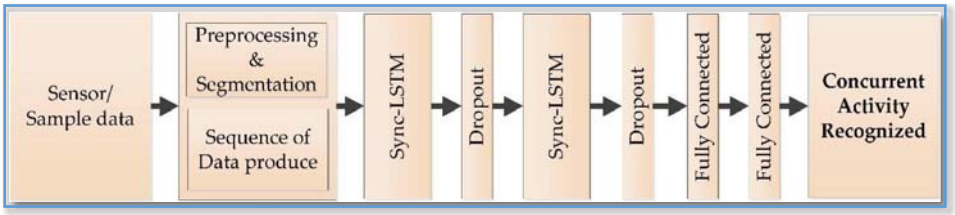


Figure 2: Proposed workflow for the recognition of concurrent human activity

3.1 Standard LSTM

RNNs use feedback to classify current data to neurons. This unique ability of RNN helps to find patterns with long-term dependencies. However, the vanishing gradient problem still occurs. To solve this problem, LSTM was introduced. LSTM outperforms RNN in finding long-term dependencies. Fig. 3a shows the internal architecture of standard LSTM. The cell in this LSTM unit of the network consists of an input gate i(t), a forgetting gate f(t), an output gate o(t), and a memory cell c(t) that stores the information and potentially updates the output over some periods.

The LSTM also has a peephole in the inner cells to the gates in the same cell to learn the timing of the outputs. The multiplicative equation commands each cell and gate to propagate forward.

$$i(t) = \sigma\left(w_{xi}x(t) + w_{hi}h(t-1) + w_{ci}c(t-1) + b_i\right) \tag{1}$$

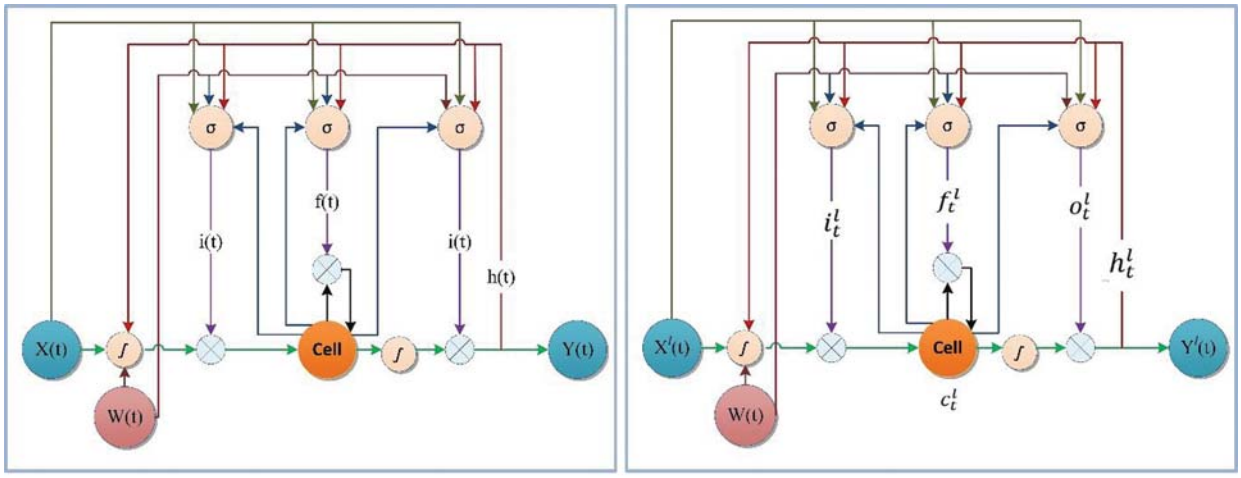
$$f(t) = \sigma(w_{xf}x(t) + w_{hf}h(t-1) + w_{cf}c(t-1) + b_f) \tag{2}$$

$$o(t) = \sigma(w_{xo}x(t) + w_{ho}h(t-1) + w_{co}c(t-1) + b_o) \tag{3}$$

$$c(t) = f_t c(t-1) + i(t)tanh(w_{xc}x(t) + w_{hc}h(t-1) + b_c) \tag{4}$$

$$h_f(t) = o(t)tanh_f c(t) \tag{5}$$

$x(t)$ is the input sequence. w_{xi}, w_{xf}, and w_{xo} are the input weights of the sequence associated with an input gate, a forgetting gate, and an output gate, respectively. $c(t)$ represents the memory cell at a corresponding time, and b represents the bias voltage of the corresponding gate and cell. The hidden layer $h(t)$ computes input $x(t)$ and provides output $y(t)$ at time t, as shown in Fig. 3a. σ is known as the logistic sigmoid activation function, which bounds the value between [0,1] and is mathematically expressed as $\sigma(x) = \frac{1}{1+e^{-x}}$. *tanh* is known as the hyperbolic tangent activation function.



(a) **(b)**

Figure 3: (a) Standard LSTM; (b) proposed sync-LSTM

3.2 Adapted LSTM: Sync-LSTM

Let a normal input sequence be $x \in \mathbb{R}^{S\times E\times I\times V\times L}$, where S and E are the start and end times, I is the sensor ID, V is the sensor value, and L is the location. Sync-LSTM takes the input of samples $\left(x_t^1, x_t^2, x_t^3 \ldots x_t^N\right)$ where each data point x_t^l is a set of individual samples l ($l = 1, 2, 3, \ldots N$) observed by the sensors at time t ($t = 1, 2, 3, \ldots N$) and processed as $x_t^1 \in \mathbb{R}^{S_1\times E_1\times I_1\times V_1\times L_1}$, $x_t^2 \in \mathbb{R}^{S_2\times E_2\times I_2\times V_2\times L_2} \ldots x_t^N \in \mathbb{R}^{S_N\times E_N\times I_N\times V_N\times L_N}$. $\left(h_t^1 + + h_t^3 + h_t^N\right)$ are the hidden states, while $Y_{t-1}^1, Y_{t-1}^2, Y_{t-1}^3 \ldots Y_{t-1}^N$ are concurrent activities detected at the corresponding time t. H is the composite function. The insight of the proposed sync-LSTM is shown in Fig. 3b, which contains the input gate i_t^l, the forget gate f_t^l, the output gate o_t^l, and the cell memory c_t^l. W(t) is the weight matrix. Each gate has its activation functions denoted by sigmoid (σ) and hyperbolic tangent ($\int$). Although synchronization is a more challenging prototype that may affect the convergence system,

this paradigm can accelerate the training model and provide a viable solution to facilitate the development and operation of AI applications.

$$i_t^l = \sigma(w_{xi} * x_t^l + w_{hi} * h_{t-1}^l + w_{ci} * c_{t-1}^l + b_i) \tag{6}$$

$$f_t^l = \sigma(w_{xf} * x_t^l + w_{hf} * h_{t-1}^l + w_{cf} * c_{t-1}^l + b_f) \tag{7}$$

$$o_t^l = \sigma(w_{xo} * x_t^l + w_{ho} * h_{t-1}^l + w_{co} * c_{t-1}^l + b_o) \tag{8}$$

$$c_t^l = f_t^l * c_{t-1}^l + i_t^l tanh * (w_{xc} x_t^l + w_{hc} h_{t-1}^l + b_c) \tag{9}$$

$$h_t^l = o_t^l * tanh * c_t^l \tag{10}$$

$$h_t^l = \mathrm{H}\left(w_{x^l h^l} x_t^l + w_{h^l h^l} h_{t-1}^l + b_h^l\right) \tag{11}$$

$$Y_t^l = (w_{y^l h^l} h_{t-1}^l + b_y^l) \tag{12}$$

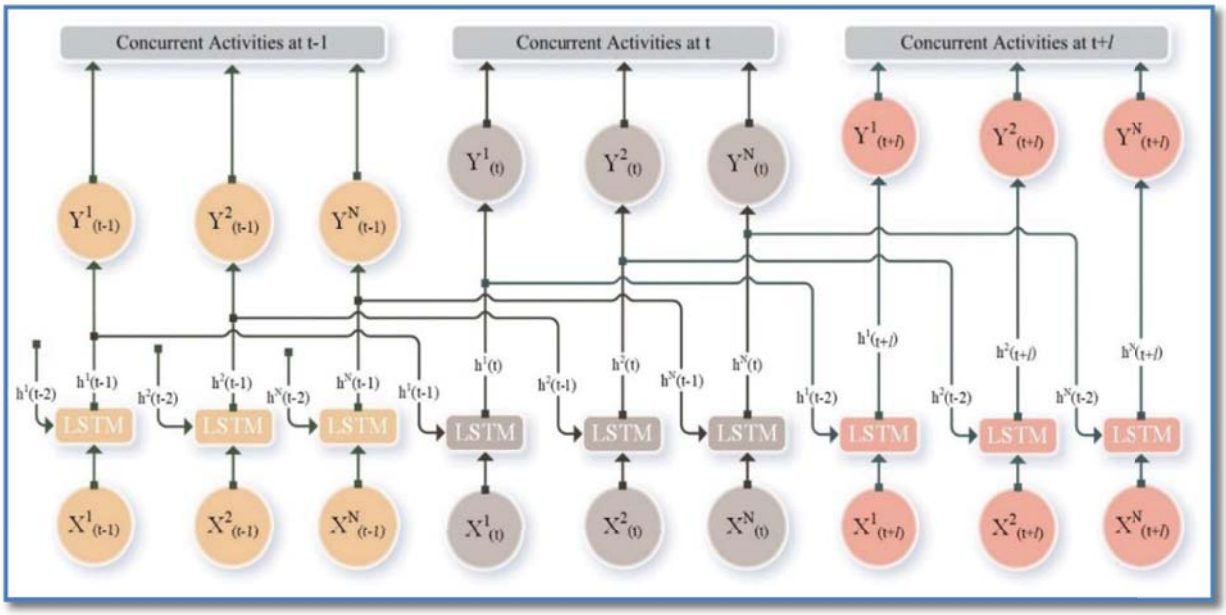


Figure 4: Unfolded architecture of sync-LSTM

Sync-LSTM forwards multiple activity sequences to detect and encode hidden internal structures between parallel hidden activity sequences. It is time-consuming to process a signal with long time steps by a standard LSTM neuron. Therefore, we used multiple LSTM units in parallel to process different parts of the information. Fig. 4 shows the unfolded architecture of the sync-LSTM network. It consists of the input layer, the parallel LSTM, the fully connected layers, and the outputs. The outputs in the final time step of each LSTM unit are summarized as n × h, where h is the number of hidden neurons of each LSTM unit. The LSTM layers adjust their internal state after each time step. The size of the weight, bias, cell, and hidden layers is assigned

to be 128. The final number of parameters depends on the number of activity classes in the classification.

Algorithm 1: Pseudo-code for concurrent activity recognition using sync-LSTM

1. *initialize network*
2. *reset: inputs = 0, activations = 0*
   forward propagation:
3. *initialize the inputs do*
4. *roll over: activations; cell states*
5. *loop over a cell, end for*
6. *do*

for t = 0 to n do
Calculate the gate values:
input gates: $i_t^l = \sigma\left(w_{xi} * x_t^l + w_{hi} * h_{t-1}^l + w_{ci} * c_{t-1}^l + b_i\right)$
forget gates: $f_t^l = \sigma\left(w_{xf} * x_t^l + w_{hf} * h_{t-1}^l + w_{cf} * c_{t-1}^l + b_f\right)$
loop over the cells in block now
output gates: $o_t^l = \sigma(w_{xo} * x_t^l + w_{ho} * h_{t-1}^l + w_{co} * c_{t-1}^l + b_o)$
update the cell: $c_t^l = f_t^l * c_{t-1}^l + i_t^l tanh * (w_{xc} x_t^l + w_{hc} h_{t-1}^l + b_c)$
final hidden state/final output: $h_t^l = o_t^l * \tan h * c_t^l$
$h_t^l = \mathrm{H}(w_{x^l h^l} x_t^l + w_{h^l h^l} h_{t-1}^l + b_h^l)$
$Y_t^l = (w_{y^l h^l} h_{t-1}^l + b_y^l)$
end for

7. *concurrent activity recognize*
8. *do*

Update the weight
end

4 Experimental Results and Analysis

This section presents the detailed results in both the training and recognition phases of the model. Several design parameters, such as the input data, the number of sync-LSTM layers, and the number of activities, are assigned and processed. The training dataset is used to train the classification, estimation, and evaluation of an individual activity for the best model parameters and for tuning the hyperparameters. Then the proposed model is trained, and the results are compared with the existing model outputs. For the experimental analysis of the proposed approach, the Kasteren and Kyoto 3 datasets are considered. The selected datasets have some limitations, such as the activity instances of different groups, some residents in each house, performance of the same activity in different ways, and availability of less learning data. Algorithm 1 presents the pseudo-code for concurrent activity recognition.

4.1 Experimental Configuration and Training

The proposed neural network is implemented in the TensorFlow and scikit-learn libraries. In this paper, linear interpolation is used to fill the missing data and normalized to a zero mean with a standard deviation of 0.5. The sensor data is pre-processed and sampled in overlapping sliding windows with a fixed width of 200 ms and a window size ranging from 0.25 s to 7 s.

The data is sampled in a single window. The proposed method is trained and tested using the TensorFlow_GPU1.13.1 library. The computer deployment is best suited to run our algorithm using i7 CPU with 16 GB RAM and GTX Titan GPU with CUDA 9.0 and using the cuDDN 7.0 library. The CPU and GPU are used to avoid exceeding the memory limit during training. The dataset is split into a training set and a test set. The model uses 70% of the data for training, 10% for validation, and the remaining 20% for testing. We use a k-fold CV (cross-validation) to validate the data. In our experiment, we validate with k = 10, known as 10-fold cross-validation. The accuracy result is averaged across all 10 folds, and the error is calculated as the cross-validation error rate.

$$CV = \frac{1}{k}\sum_{k=1}^{10} Error \tag{13}$$

During training, the dropout rate is set to 0.5 to remove and avoid unused specific neurons from each hidden layer to solve the overfitting problem. The training loss function is minimized by random initialization and optimization of training parameters. The two-loss functions, named cross-entropy and L2 normalization, are inherited to avoid overfitting throughout the epoch to make the model stable.

$$L = -\frac{1}{m}\sum_{m=1}^{n} y_t^l.logy_t^{l'} + \Gamma.\|W\|, \tag{14}$$

where m is the number of samples per batch, and Γ is the weighting parameter. y_t^l is the predicted output, and $y_t^{l'}$ is the label from the dataset. L2-normalization limits the trainable weighting parameter to a smaller value, which avoids overfitting.

We try to tune the best hyperparameters as shown in Tab. 1 in networks so that the learning rate and L2 weight decrease to reduce the difference and thus achieve the possible optimal performance. We train the model with a learning rate of 0.005 and 0.006 for the Kyoto 3 and Kasteren House-B datasets by taking the batch value of 100 for each epoch. Learning starts at 0.001 for all data. The training is performed for more than 12,000 epochs and stops at stable outputs. The Adam optimizer is an adaptive moment estimator that obtains independent adaptive learning rates for different parameters, making them more stable. The dimension based on the input is set to 128. The gradient clipping is set to 4 and 5 to reduce the threshold for crossing the gradient to match the normalization. The batch size is set to 100 samples, and this is often referred to as a mini-batch gradient descent since our batch size is smaller than the training sample size.

4.2 Datasets

Our proposed method is evaluated using the Kasteren datasets [43] and CASAS [44]. An overview of the datasets is shown in Tab. 2. Both datasets were gathered in an apartment containing either four or two rooms. Seventy-six sensors are deployed in Kyoto 3, whereas 23 sensors are deployed in House B. Of these sensors, 51 were used for motion detection, 12 were used as cabinet sensors, 5 were used for cooking element detection, 3 were used as temperature sensors, and the remainder was used as a medicine container sensor, pot sensor, phone book sensor, water sensor, burner sensor, or phone sensor in Kyoto 3. The four residents in the Kyoto 3 dataset performed eight different activities for 15 days. There are 178 instances of activity recorded from Kyoto 3. The single resident in the Kasteren house performed 13 different types of activities for 14 days.

The sensors show the change of state according to the action of the occupant. Radiofrequency identification, a wireless sensor network, a pressure sensor, a reed switch, mercury contacts, a passive infrared-PIR, float sensors, and temperature sensors are used to record the data for the Kasteren house. The schematic diagram of the sensor deployment is shown in Fig. 5.

Table 1: Hyperparameter settings

Hyperparameters	Values	
	Kyoto 3	House-B
Time steps of input	128	128
Dropout rate	0.5	0.5
Initial learning rate	0.001	0.001
Learning rates	0.005	0.006
Optimizer (Bi-LSTM)	Adam	Adam
Batch size	100	100
Gradient clipping	4	5
Epochs	12000	12000

Table 2: Overview of Kyoto 3 and Kasteren House-B dataset

Description	Kyoto 3	House-B
Setting	Apartment	Apartment
Rooms	4	2
Senors	76	23
Activities	8	13
Residents	4	1
Period	15 days	14 days
Instances	178	135
Activities performed	Fill Medication Dispenser, Wash DVD, Water Plants, Answer the Phone, Prepare Birthday Card, Prepare Soup, Clean, Choose Outfit	Breakfast, Brushing Teeth, Dinner, Drinking, Dressing, Leaving House, Others, Preparing Breakfast, Preparing Dinner, Sleeping Showering, Toileting, Using Dishwasher

4.3 Evaluation Metrics

The metrics of the confusion matrix, accuracy, F1-score, and training time are used to evaluate the performance of the model. A confusion matrix shows the performance of the approach, where the row represents the predicted class, and the column represents the actual class and vice versa. It also gives information about the errors made. Generally, human activity recognition methods are evaluated according to their computational recognition accuracy. The accuracy is calculated using the confusion matrix, which is the result of the Precision and Recall parameters.

Precision is the proportion of correctly recognized instances out of the absolute perceived activity occurrences. Recall is the proportion of correctly recognized instances out of the total occurrences of the activity. An f-score is the weighted average of Precision and Recall whose value is between [0,1], with the best performance indicated if it is closer to 1.

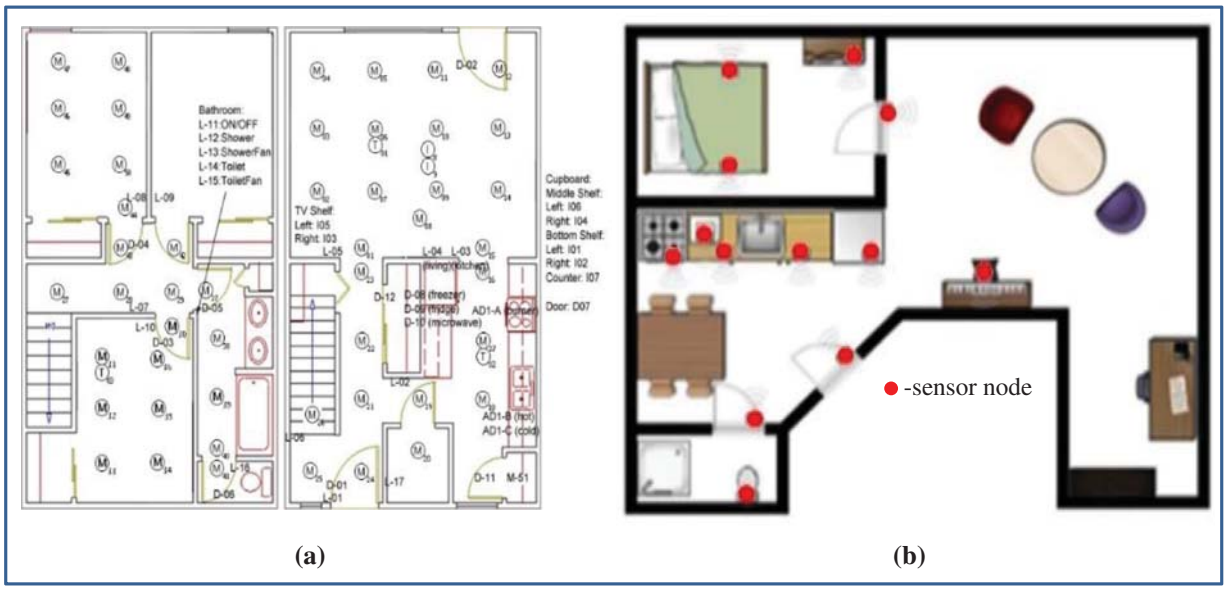


Figure 5: Schematic diagram of sensor deployed and layout (a) Kyoto 3 (b) Kasteren House-B

$$Precision = \frac{tp}{tp + fp} \times 100 \tag{15}$$

$$Recall = \frac{tp}{tp + fn} \times 100 \tag{16}$$

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \times 100 \tag{17}$$

$$f\text{-}score = \frac{2 \times Precision \times Recall}{Precision + Recall}, \tag{18}$$

where tp, tn, fp, and fn are true-positive, true-negative, false-positive, and false-negative, respectively. The true-positive score is defined as the number of true activities detected in positive instances, while a false-positive indicates the false activities detected in negative instances. The false-negative score indicates the exact number of false activities detected in positive instances, whereas the true-negative score reflects the correct non-detection of activities in the negative instances.

4.4 Recognition Analysis

In this section, a possible implementation of the platform for human activity detection in smart homes is explained. All activities are localized based on the dataset. The activities that

occur most frequently at the same time are considered the predominant activities in a smart home environment. Fig. 6 presents the graphical confusion matrix for Kyoto 3.

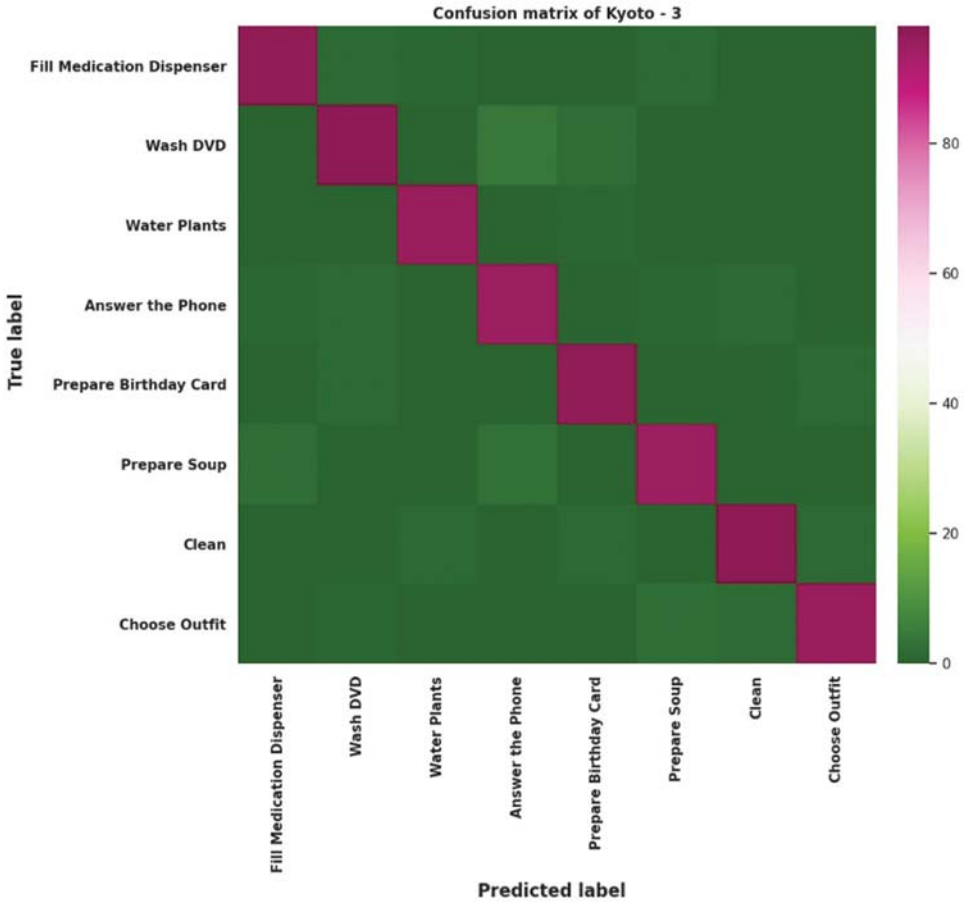


Figure 6: Confusion matrix for Kyoto 3

According to the confusion matrix, the *filling medication* activity was correctly detected with 98% accuracy but still has an activity error of 2% because the call may be made while other activities are being performed simultaneously. *Watching DVD* also has a 98% confusion accuracy, although 1.3%, 1.2%, and 0.4% of the *fill medication dispenser, answer the phone*, and *choose outfit* activities cause confusion since they can all be performed at the same location. The activities *water plants, answer the phone, prepare birthday card, prepare soup, clean*, and *choose outfit* have recognition accuracies of 98%, 96%, 97%, 99%, 98%, and 98%, respectively. The activity *answer the phone* has the lowest recognition accuracy compared to the rest of the activities, and this activity also leads to confusion with all other activities because a phone call can be performed with all other activities that share most sensor values. However, the co-occurrence of *answer the phone* is higher. The data from Kasteren and CASAS have the lowest number of records and instances; therefore, the actual distribution is easy to find and train. The recognition accuracy can increase if we train our proposed method with a large amount of data.

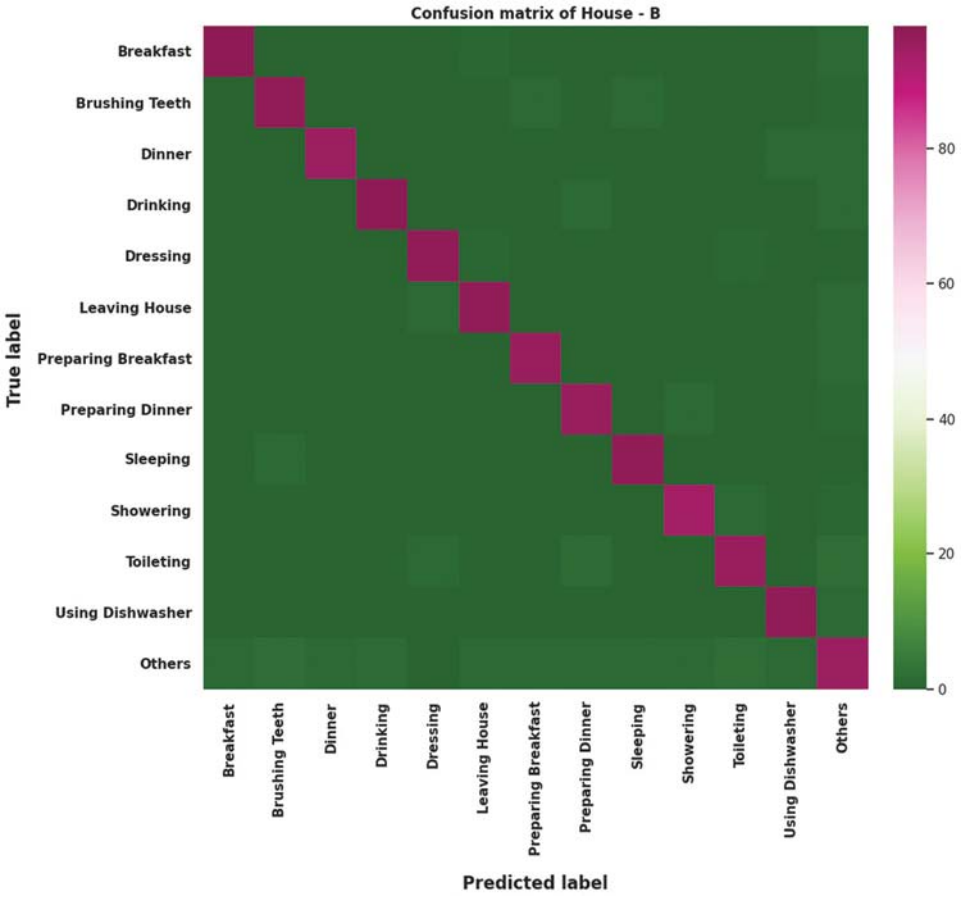


Figure 7: Confusion matrix for House-B

The confusion matrix for House-B is shown in Fig. 7. The number of activity instances is relatively small, so the occurrence of errors is relatively lower, and recognizing the concurrent activity with the highest true positive value results in 96.90% accuracy. The activity *drinking* in House-B is the most appropriate activity that occurs concurrently with the activities *preparing dinner, preparing breakfast, dinner, relaxing, using dishwasher*, and *leaving house*, with an accuracy of more than 96.90%. House-B achieved precision, recall, and an f1-score of 97.94%, 97.00%, and 0.97, respectively. *Brushing teeth* is also detected simultaneously with the activity *toileting*. Activities such as *brushing teeth, showering*, and *toileting* create confusion with some error because all the activities share the same location. However, the errors are comparatively very low and can be neglected. The *using dishwasher* activity shows concurrency with activities such as *preparing dinner, preparing breakfast, dinner*, and *breakfast*. The detection of *using dishwasher* concurrently with other activities is more than 97.935%. All of these concurrently recognized activities have high detection accuracies. *Sleeping* is a stand-alone activity: It cannot appear simultaneously with other listed activities. Sometimes it may create some confusion with the *dressing* activity, as it is performed in the bedroom. The accuracy of House-B is insufficient to fully establish the experimental concept. Although the accuracy is high, many datasets could be needed to find the proposed actual recognition. We confirm that our proposed approach for concurrent human activity recognition is feasible.

4.5 Performance Comparison

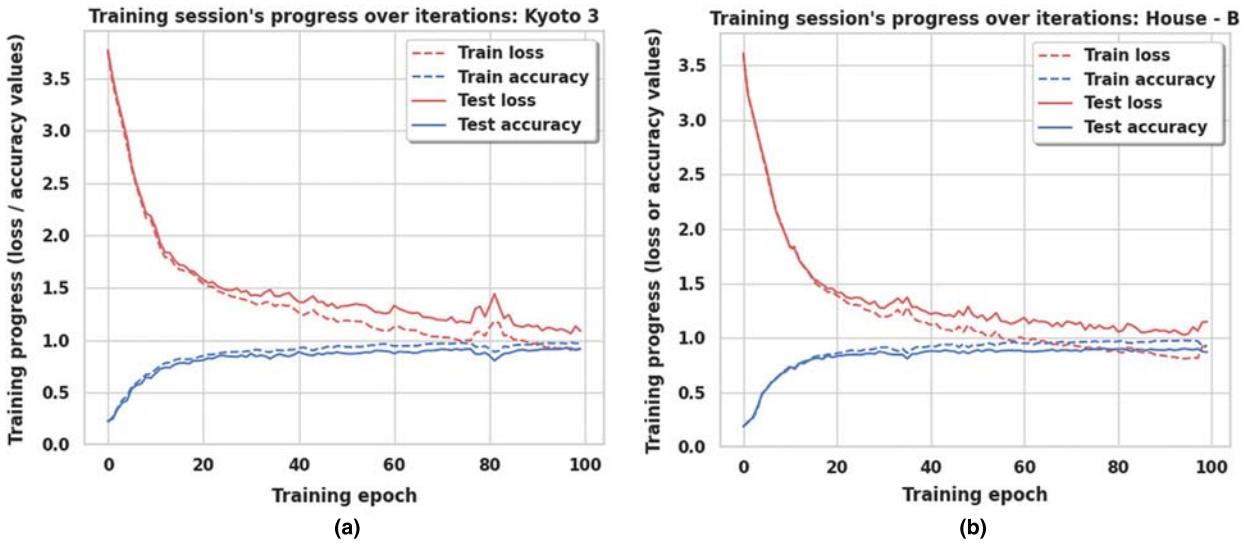


Figure 8: Accuracy and loss curve of (a) Kyoto 3 and (b) House-B

Figs. 8a and 8b present the accuracy and loss of training and testing procedures for Kyoto 3 and House-B, respectively. In the graphs, the gap between the training and testing accuracy is comparatively small, indicating the effectiveness of the model. The gap between training and test loss is also very narrow, which explains that the dropout techniques are beneficial and resistant to overfitting.

The average accuracy was found to be 97.374%, and the average error was 0.1637. The performance of the proposed approach was compared with the existing framework, which uses CNN, LSTM, and Bi-LSTM methods (algorithms) for recognition by measuring the average precision, recall, f-score, and accuracy, which are shown in Tab. 4. The accuracy of our method is more than 97%, and the f1-score is more than 0.97. The Bi-LSTM also has competitive accuracy with our method, but it can only process two inputs simultaneously. The cross-validation process is performed before the test to validate the input sequences; therefore, the accuracy of the test increases. This is a natural phenomenon in AI. The mean and standard deviation of the accuracy and error using 10-fold cross-validation are shown in Tab. 3. The given analysis shows that the proposed method can detect concurrent activities with higher accuracy than the existing approaches.

Table 3: Average accuracy and standard deviation (SD) over the 10-fold CV

	Mean (μ) $\pm$ SD (σ) Accuracy	Mean (μ) $\pm$ SD (σ) Error
Kyoto 3	0.9748 $\pm$ 0.0448	0.1761 $\pm$ 0.0160
House-B	0.9690 $\pm$ 0.0455	0.1513 $\pm$ 0.0194

Table 4: Accuracy comparison with existing approaches

Metric	Model	Dataset	
		Kyoto 3	House-B
Precision (%)	CNN	96.38	94.32
	LSTM	97.72	96.67
	Bi-LSTM	98.01	96.78
	Proposed method	**98.8**	**97.36**
Recall (%)	CNN	96.22	94.38
	LSTM	97.23	96.67
	Bi-LSTM	98.06	96.83
	Proposed method	**98.48**	**97.35**
f1-score	CNN	0.95	0.94
	LSTM	0.96	0.96
	Bi-LSTM	0.97	0.96
	Proposed method	**0.98**	**0.97**
Accuracy (%)	CNN	96.66	95.21
	LSTM	96.88	96.42
	Bi-LSTM	97.09	96.99
	Proposed method	**98.38**	**97.42**

4.6 Computational Complexity

The computational complexity depends on the number of weights and is given as $O(W)$, where W is the weight. The weight depends on the number of output units, the cell storage unit, the size of the memory, and the number of hidden units. It is also affected by the number of units associated with forwarding neurons, memory cells, gate units, and hidden units. The computational complexity does not depend on the length of the input sequence. Although using an LSTM framework increases time complexity, our approach has an acceptable computation time. Unlike concurrent activity detection, Bi-LSTM has lower complexity, but it can only handle two parallel activity detection processes and apply delay or other chaining functions to recognize more than two activities.

This feature causes the system to wait for a complete process, which increases the computational complexity. Fig. 9 shows the computational time for testing our method compared to existing frameworks, such as CNN, LSTM, and Bi-LSTM, for Kasteren house and Kyoto 3. In the CNN comparison, it processes too many hidden layers and pooling as it segments the activity into other sub-activities, which causes higher computational complexity. The computation time is slightly higher but satisfactory and executable.

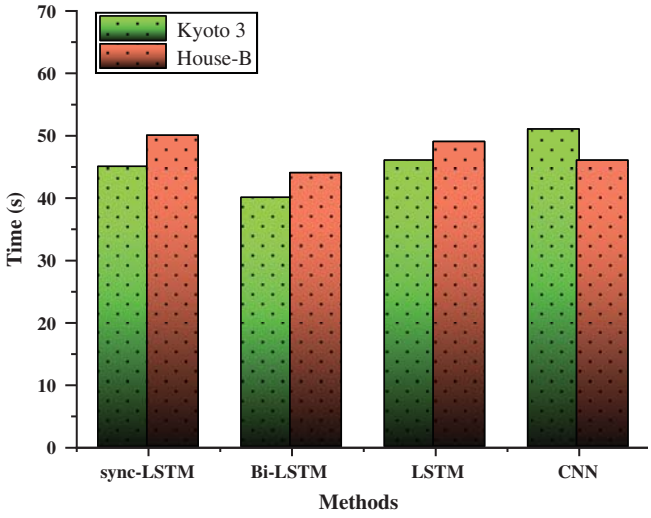


Figure 9: Computation time comparison with existing approaches

5 Conclusion

The framework presented in this paper shows that sync-LSTM can lead to a feasible solution for detecting concurrent human activities in the smart home scenario. This claim is supported by a comprehensive comparison with recently utilized activity recognition techniques, such as CNN, LSTM, and bi-LSTM. LSTM can work with single data sequences and bi-LSTM with a maximum of two data sequences, but our sync-LSTM can accept multiple inputs and generate multiple outputs, synchronized and in parallel. That is, for any environment or domain for concurrent processing and recognition, this sync-LSTM would be an effective solution. Many approaches focus on single and regular activity detection. Few of them have tried to detect complex activity. Starting from the standard LSTM formulation, we have improvised a more efficient LSTM-based approach to recognize complex human activity.

However, accuracy, processing complexity, and complex activity recognition are still significant challenges in human activity recognition. The proposed method has an f1 score of more than 0.97, along with an accuracy of more than 97%. This proves its effectiveness for concurrent human activity detection with successful training and testing. Nevertheless, the accuracy is limited due to some error factors, such as same location errors, sensor distance, noise interference, and limited data. The unique best-performing model also suffers from several real-time challenges across different datasets. The parameters, like number of activities, type of sensors, sensor distribution, number of occupants, and duration of test periods, also affect the performance. The window size plays a significant role in performance, as a small size may not contain all of the information, and large size may lead to resident detection errors. The proposed method processes parallel data, which is beneficial and consistent with the setting of highly imbalanced datasets. Therefore, data augmentation techniques are not required.

Besides, sync-LSTM can automatically extract spatio-temporal information by reducing the time-consuming effort for pre-processing data and manual feature extraction. External sensors were used instead of wearables and camera or video sensors to avoid the unnecessary burden and protect the privacy of the resident. In the future, more complex activities, such as interleaved activity, will be recognized by improving and updating the proposed method. Furthermore, we

can take advantage of cloud computing by using Google Colab and Amazon Web Services. These technologies provide the opportunity to use their servers and also experiment with tensor processing units. Using these techniques and technologies will also reduce the time complexity for faster and better performance. We will also explore a transfer learning approach for this model in other domains, environments, and sectors on big data and cloud infrastructures. In summary, our proposed method (i.e., a sync-LSTM-based model that provides fewer parallel and synchronized recognition and prediction paradigms) is preferable to its competitors.

Funding Statement: This research was supported by the Ministry of Trade, Industry & Energy of the Republic of Korea as an AI Home Platform Development Project (20009496) and conducted under a research grant from Kwangwoon University in 2021.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] S. Wan, L. Qi, X. Xu, C. Tong and Z. Gu, "Deep learning models for real-time human activity recognition with smartphones," *Mobile Networks Application*, vol. 25, no. 2, pp. 743–755, 2020.

[2] F. J. Ordóñez and D. Roggen, "Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, Article no. 115, 2016.

[3] A. Graves, N. Jaitly and A. Mahamed, "Hybrid speech recognition with deep bidirectional LSTM," in *Automatic Speech Recognition and Understanding, 2013 IEEE Workshop*, Olomouc, Czech Republic, pp. 273–278, 2013.

[4] A. Karpathy, A. Joulin and F. F. Li, "Deep fragment embeddings for bidirectional image sentence mapping," *Advance Neural Information Process System*, vol. 3, no. January, pp. 1889–1897, 2014.

[5] T. L. M. Van Kasteren, G. Englebienne and B. J. A. Kröse, "An activity monitoring system for elderly care using generative and discriminative models," *Personal and Ubiquitous Computing*, vol. 14, no. 6, pp. 489–498, 2010.

[6] P. Rashidi and D. J. Cook, "Keeping the resident in the loop: Adapting the smart home to the user," *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans*, vol. 39, no. 5, pp. 949–959, 2009.

[7] A. Bevilacqua, K. MacDonald, A. Rangarej, V. Widjaya, T. Kechadi *et al.,* "Human activity recognition with convolutional neural networks," *Machine Learning and Knowledge Discovery in Databases*, vol. 11053, no. September, pp. 541–552, 2019.

[8] A. Holzinger, "Introduction to machine learning & knowledge extraction (MAKE)," *Machine Learning and Knowledge Extraction*, vol. 1, no. 1, pp. 1–20, 2017.

[9] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computing*, vol. 9, no. 8, pp. 1735–1780, 1997.

[10] K. Thapa, Z. M. Al Abdullah, B. Lamichhane and S. H. Yang, "A deep machine learning method for concurrent and interleaved human activity recognition," *Sensors*, vol. 20, no. 20, Article no. 5770, 2020.

[11] L. Theis and M. Bethge, "Generative image modeling using spatial LSTMs," *Advance in Neural Information Processing Systems*, vol. 2015, no. January, pp. 1927–1935, 2015.

[12] E. Nazerfard, B. Das, L. B. Holder and D. J. Cook, "Conditional random fields for activity recognition in smart environments," in *IHI'10—Proc. of the 1st ACM Int. Health Informatics Symp.*, Arlington, Virginia, USA, pp. 282–286, 2010.

[13] X. Li, Y. Zhang, I. Marsic, A. Sarcevic and R. S. Burd, "Deep learning for RFID-based activity recognition," in *Proc. of the 14th ACM Conf. on Embeded Network Sensor Systems*, Stanford, CA, USA, pp. 164–175, 2016.

[14] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs and H. Lipson, "Understanding neural networks through deep visualization," *Computer Vision and Pattern Recognition*, vol. 12, no. June, pp. 12–22, 2015.

[15] S. Hochreiter and P. Frasconi, "Gradient flow in recurrent nets: The difficulty of learning long-term dependencies," in *A Field Guide to Dynamical Recurrent Neural Networks*, NY, USA, IEEE Press, 2001.

[16] L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, J. Donahue *et al.,* "Long-term recurrent convolutional networks for visual recognition and description," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 677–691, 2017.

[17] M. H. Kabir, M. R. Hoque, K. Thapa and S.-H. Yang, "Two-layer hidden Markov model for human activity recognition in home environments," *International Journal of Distributed Sensor Networks*, vol. 2012, no. 1, Article no. 4560365, 2016.

[18] D. Hu and Q. Yang, "CIGAR: Concurrent and interleaving goal and activity recognition," in *Proc. of the 23rd National Conf. on Artificial Intelligence*, vol. 3, pp. 1363–1368, 2008.

[19] T. Choudhury, S. Consolvo, B. Harrison, G. Bordello, B. Hemingway *et al.,* "The mobile sensing platform: An embedded activity recognition system," *IEEE Pervasive Computing*, vol. 7, no. 2, pp. 32–41, 2008.

[20] M. R. Hoque, M. H. Kabir, H. Seo and S. H. Yang, "PARE: Profile-applied reasoning engine for context-aware system," *International Journal of Distributed Sensor Networks*, vol. 12, no. 7, Article no. 5389091, 2016.

[21] P. G. Lavanya and S. Mallappa, "Activity recognition from accelerometer data using symbolic data approach," *Lecture Notes in Networks and Systems*, vol. 43, pp. 317–329, 2019.

[22] L. Gao, A. K. Bourke and J. Nelson, "Evaluation of accelerometer based multi-sensor versus single-sensor activity recognition systems," *Medical Engineering and Physics*, vol. 36, no. 6, pp. 779–785, 2014.

[23] M. L. Rohling, L. M. Binder, G. J. Demakis, D. Ploetz, J. L. Rohling *et al.,* "A meta-analysis of neuropsychological outcome after mild traumatic brain injury: Re-analyses and reconsiderations of Binder et al. (1997), Frencham et al. (2005) and Pertab et al. (2009)," *The Clinical Neuropsychologist*, vol. 25, no. 4, pp. 608–623, 2011.

[24] N. Y. Hammerla, S. Halloran and T. Plötz, "Deep, convolutional and recurrent models for human activity recognition using wearables," *Int. Joint Conf. on Artificial Intelligence*, New York, USA, vol. 2016, no. January, pp. 1533–1540, 2016.

[25] T. Y. Wu, C. C. Lian and J. Y. J. Hsu, "Joint recognition of multiple concurrent activities using factorial conditional random fields," in *AAAI Workshop—Techical Report*, Vancouver, British Columbia, vol. WS-07-09, pp. 82–87, 2007.

[26] J. Talukdar and B. Mehta, "Human action recognition system using good features and multilayer perceptron network," in *IEEE Int. Conf. on Communication and Signal Processing 2017*, Tamilnadu, India, pp. 317–323, 2017.

[27] C. Schuldt, I. Laptev and B. Caputo, "Recognizing human actions: A local SVM approach," in *17th International Conf. on Pattern Recognition*, Cambridge, vol. 3, pp. 32–36, 2004.

[28] L. Fan, Z. Wang and H. Wang, "Human activity recognition model based on decision tree," in *Proc. Int. Conf. on Advanced Cloud and Big Data*, Nanjing, China, pp. 64–68, 2013.

[29] M. H. Kabir, K. Thapa, J. Y. Yang and S. H. Yang, "State-space based linear modeling for human activity recognition in smart space," *Intelligent Automation and Soft Computing*, vol. 25, no. 4, pp. 673–681, 2019.

[30] Y. Sung-Hyun, K. Thapa, M. Humayun Kabir and L. Hee-Chan, "Log-viterbi algorithm applied on second-order hidden Markov model for human activity recognition," *International Journal of Distributed Sensor Networks*, vol. 14, no. 4, Article no. 1550147718772541, 2018.

[31] F. A. Gers, J. Schmidhuber and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[32] H. Yalcin, "Human activity recognition using deep belief networks," in *24th Signal Processing and Communication Application Conf.*, Zonguldak, Turkey, pp. 1649–1652, 2016.

[33] T. Ogawa, Y. Sasaka, K. Maeda and M. Haseyama, "Favorite video classification based on multimodal bidirectional LSTM," *IEEE Access*, vol. 6, pp. 61401–61409, 2018.

[34] S. Ji, W. Xu, M. Yang and K. Yu, "3D Convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[35] A. Gumaei, M. Al-Rakhami, H. Al Salman, S. M. Mizanur Rahman and A. Alamri, "Deep learning-based human activity recognition framework for edge computing," *Computer, Materials and Continua*, vol. 65, no. 2, pp. 1033–1057, 2020.

[36] X. Liu, Y. Si and D. Wang, "LSTM neural network for beat classification in ECG identity recognition," *Intelligent Automation and Soft Computing*, vol. 26, no. 2, pp. 341–351, 2020.

[37] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[38] M. Sundermeyer, R. Schlüter and H. Ney, "LSTM neural networks for language modeling," in *13th Annual Conf. on Int. Speech Communication Association, INTERSPEECH 2012*, Portland, OR, USA, vol. 1, pp. 194–197, 2012.

[39] H. H. A. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*. vol. 247. USA: Springer Science & Business Media, 1994.

[40] M. Bouaziz, M. Morchid, R. Dufour and G. Linares, "Improving multistream classification by mapping sequence-embedding in a high dimensional space," in *IEEE Workshop on Spoken Language Technology—SLT*, San Diego, CA, USA, pp. 224–231, 2017.

[41] J. H. Bappy, C. Simons, L. Nataraj, B. S. Manjunath and A. K. Roy-Chowdhury, "Hybrid LSTM and encoder-decoder architecture for detection of image forgeries," *IEEE Transactions on Image Processing*, vol. 28, no. 7, pp. 3286–3300, 2019.

[42] Y. Zhang, J. Zhang, S. Chen, I. Marsic, R. A. Farneth *et al.,* "Concurrent activity recognition with multimodal CNN-LSTM structure," *Computer Vision and Pattern Recognition, arXiv*, 2017, https://arxiv.org/abs/1702.01638.

[43] T. L. M. van Kasteren, G. Englebienne and B. J. A. Kröse, "Human activity recognition from wireless sensor network data: Benchmark and software," in *Activity Recognition in Pervasive Intelligent Environments*, Paris, France: Atlantis Press, pp. 165–186, 2011.

[44] D. J. Cook, M. Schmitter-Edgecombe, A. Crandall, C. Sanders and B. Thomas, "Collecting and disseminating smart home sensor data in the CASAS project," in *Proc. of the CHI Workshop on Developing Shared Home Behavior Dataset to Advance HCI and Ubiquitous Computing Research*, Matsue-city, Shimane, Japan, pp. 1–7, 2009.