 doi:10.20944/preprints202203.0363.v1

Context-Aware Complex Human Activity Recognition using Hybrid Deep Learning Models

Adebola Omolaja[1] and Abayomi Otebolaku[2]
Department of Computing, Sheffield Hallam University, Sheffield, United Kingdom
a.otebolaku@shu.ac.uk[2], debosky007@gmail.com[1]

Correspondence: a.otebolaku@shu.ac.uk

ABSTRACT

Smart devices such as smartphones, smartwatches, etc. are promising platforms that are being used for automatic recognition of human activities. However, it is difficult to accurately monitor complex human activities due to inter-class pattern similarity, which occurs when different human activities exhibit similar signal patterns or characteristics. Current smartphone-based recognition systems depend on the traditional sensors such as accelerometer and gyroscope, which are inbuilt in these devices. Therefore, apart from using information from the traditional sensors, these systems lack contextual information to support automatic activity recognition. In this article, we explore environment contexts such as illumination(light conditions) and noise level to support sensory data obtained from the traditional sensors using a hybrid of Convolutional Neural Networks and Long Short Time Memory(CNN_LSTM) learning models. The models performed sensor fusion by augmenting the low-level sensor signals with rich contextual data to improve recognition and generalisation ability of the proposed solution. Two sets of experiments were performed to validate the proposed solution. The first set of experiments used inertial sensing data whilst the second set of extensive experiments combined inertial signals with contextual information from environment sensing data. Obtained results demonstrate that contextual information such as environment noise level and illumination using hybrid deep learning models achieved better recognition accuracy than the traditional activity recognition models without contextual information.

Keywords: Contextual information; deep hybrid learning model; activity recognition; Sensors; Smart devices.

1. INTRODUCTION

In the last two decades, the study of Human Activity Recognition (HAR) has been significantly enhanced because of the emergence of smart computing devices, availability of huge datasets and unprecedented breakthrough in the development of machine learning/artificial intelligence algorithms that can provide real-time predictions of various daily human activities and movements[1-2]. Today, the number of smartphone users worldwide has surpassed the three billion mark and is projected to increase by several hundred million in the next few years[1]. This surge has resulted in rapid development of several intelligent application domains that leverage on the use of inbuilt sensing capabilities of smartphones. These domains among others include smart homes [3], healthcare [4-5] personalised content recommendations [6-7]manufacturing [8] and self-driving cars [9]. Whilst these domains may have certain peculiarities regarding the activities involved, each domain, within its own contexts, involves several combinations of simple and complex activities, particularly in the domain of human activity monitoring in ambient environments. Although significant progress has been achieved in the HAR domain, most works have focused on simple activities compared to complex activities that reflect people's real daily lives [2, 10-11]. Whilst complex activities are characterised by several different simple activities, they are also prone to inter-class similarity problems. According to [10], an inter-class similarity occurs when activities that are different exhibit similar patterns in their sensor signals. Because of these similar signals from sensing devices, discriminating between simple and complex activities becomes more difficult for HAR models. To address the inter-class similarity problem, not just the use of more sensory data is required but also the use of algorithms capable of discriminating patterns in these activity signals, considering the contexts of such activities[10] .

To emphasise the inter-class problem in HAR, let us consider the following scenarios that illustrate typical situations where a mix of simple and complex activities can be misclassified when they exhibit inter-class similarity

[1] https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

when contextual information is not taken into consideration. A driver, for example, may be distracted with a personalised mobile content recommendation because a model confuses driving with running. Both activities

lying in the bed with sun-bathing. These are two examples of activities that involve lying down but the latter is a complex activity typically involving substantial amount of sunshine(illumination) and sometimes noise in a specific location usually on the beach. A final example is walking and mountain-climbing. Again, these are two activities that are very similar but contextually different. Mountain-climbing involves walking but, at the same time, the contexts are quite different. Usually, factors such as high altitude are associated with low temperature and humidity. Therefore, to effectively discriminate between these similar activities, information such as spatial-temporal and environment contexts are required.

The examples above demonstrate clearly that simple and complex activity recognition from sensor data does pose several challenges. First, the underlying activities of a complex activity are mostly dependent [12]. Particularly, in one complex activity, the temporal relatedness of several simple activities often manifest itself in several different forms. As a result, the complicated temporal combinations often lead to semantic meanings for understanding a complex activity [13]. Second, multiple complex activities share one or more common and related simple activities. The driving and eating examples above clearly illustrate this. Another good example is making sandwiches which is expected to be more similar to making coffee than cycling in terms of activity patterns. Third, complex activities, compared to simple activities, contain higher level semantics that are often more complicated and are known to be more reflective of the daily lives e.g., driving, eating and relaxing. Finally, complex activities often take longer to detect compared to the simple ones [14-15].

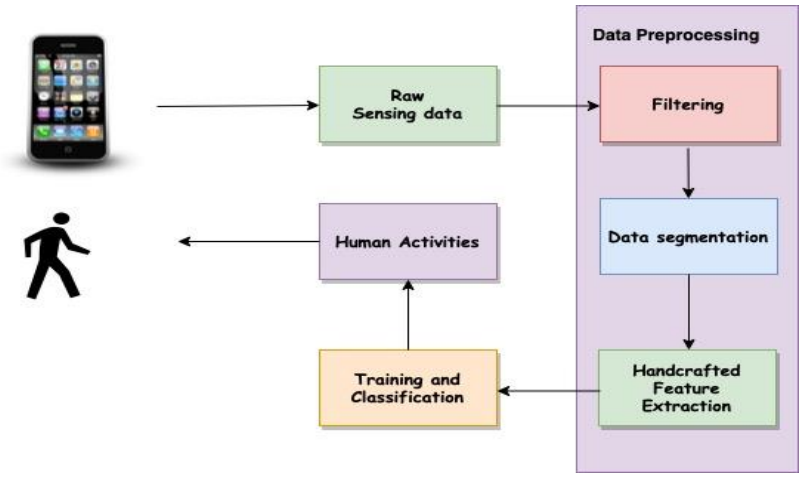


Figure 1. An Overview of Traditional Activity Recognition Processes

In the past, as illustrated in Figure 1, researchers have solved HAR using the traditional machine learning techniques and hand-crafted feature extraction methods[16-17]. Although this process is known to help reduce overfitting the model, reduce training time, and improve accuracy, the downside is that when features are hand-crafted, the model becomes susceptible to classification errors. Handcrafted feature extraction is also limited by human domain expertise, that is, a deep knowledge of the application domain is required. Lastly, because this approach only helps extract shallow features, the performance of the model is largely undermined resulting in poor generalisation [11].

Recently, Deep Convolutional Neural Networks (DNN) began to deliver on its promises of automatic feature extraction and achieve state-of-the-art results for HAR. When DNN is applied to raw time series data and because of its automatic feature extraction capabilities, it often outperforms models trained on heuristic hand-crafted features[18]. Deep learning models also have capabilities to extract high-level representation within the deep layer making it more applicable to complex activity recognition problems[11].The most common DNNs for HAR are the Convolutional Neural Network, Recurrent Neural Network (RNN) and the Long-Term Short Memory (LSTM) networks. CNNs are specifically designed to effectively process image data solving computer vision problems such as image classification, image captioning and object localisation[19]. Compared to CNNs, RNNs can learn and keep memories of temporal dependencies of time series data by relying on its dynamic temporal behaviours. LSTM networks are an extension of RNNs, possessing complex memory cells which help it avoid the long-term dependency problem of vanilla RNNs. Since DNNs can adaptively learn spatial hierarchies of features

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

to discover low- and high-level patterns in the dataset, new datasets and new sensor modalities can be quickly adopted with minimal configuration. Several authors have demonstrated the effectiveness of DNNs on HAR systems with state-of-the-art results and recently using hybrid deep learning models [20]as we discuss in section 2.

The aim of this study is to distinguish between simple and complex human activities of daily living and address the associated inter-class similarity problems using a combination of traditional motion and ambient sensing data with hybrid deep neural networks. The proposed deep hybrid feature model combines rich context-aware data from ambient sensors with low-level inertial sensor signals as shown in Figure 2. The motion sensors include the accelerometer, gyroscope, as well as other sensors such as magnetometer while the ambient signals, which serve as the context data include signals from the sound and light sensors of the smartphone. The hybrid model efficiently combines the diverse capabilities of CNNs and LSTM to form a CNN-LSTM hybrid model. This hybrid model has previously demonstrated very good accuracy in the speech recognition domain [21-22].

The key research question addressed in this article is "how do we determine the extent to which rich contextual information could improve the accuracy of deep hybrid learning models in generalising simple and complex activities of daily living while also addressing the inter-class similarity problems?". To answer this question, the contributions of this article are five-fold:

- The article proposes a deep hybrid feature model capable of discriminating between simple and complex activity signal patterns by augmenting low-level inertial signals with contextual signals to improve recognition and generalisation accuracy.
- An extensive review of state-of-the-art in human activity recognition, context-awareness, deep learning algorithms, and inter-class similarity problems in human activity recognition.
- We optimize the hybrid models' hyperparameters using grid search method.
- We demonstrate through extensive experiments the efficacy of CNN-LSTM model for realising state-of-the-art results using raw sensor signals without heuristic hand-crafted features.
- We demonstrate the efficacy of smartphone rich contextual data for solving inter-class similarity problem of simple and complex activities of daily living.

The remainder of the article is structured as follows: In section 2 we review related works in aspects covered by the proposed solution. In section 3, we present the proposed solution. Experimental validation of the proposed solution is presented in section 4. In section 5, we discuss the experimental results and their significance. Finally, section 6 draws conclusions and presents recommendations for future work.

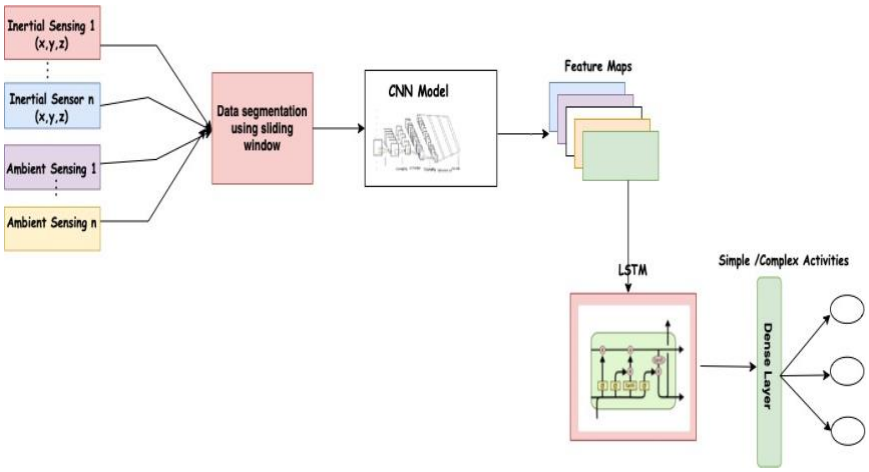


Figure 2.High-level view of the hybrid model

2. Literature Review and Related Work

In this section, the article examines existing literature in the area of simple and complex activities, machine learning, and context awareness for complex HAR systems.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

2.1 Simple and complex human activities

The HAR domain has witnessed huge attention by research studies involving simple activity recognition. However, there is limited support of studies focusing on complex activities. This is evident in several independent surveys conducted by other researchers [2, 10-11, 23-24]. They have suggested that lack of studies is attributed to the data associations inherent in complex activities. For instance, [2] categorised complex activities into three broad groups: composite activities comprising a series of simple activities; concurrent activities occurring when a user engages in more than one simple activities at the same time; and Multi-occupant activities suggesting more than one user engaging in a set of activities in a multi-resident environment. Moreover, using an approach-based tree-structured taxonomy,[23] categorised human activities into two broad groups: single-layered and hierarchical activities. The single-layered activities are likened to simple activities such as gestures and actions with sequential characteristics whilst the hierarchical activities are high-level activities with multiple single-layered activities. Another definition given by [25] suggested that the differentiating factor is the duration. In their paper, they referred to simple activities as single-person actions with short duration, while complex activities are regarded as a complex sequence of actions performed by several individuals over a long time. Although several authors have used different terminologies for their categorisation, [26] argue that the boundaries of these classes are still not clearly demarcated. One thing that these existing studies agree on is that human activities can be further categorised into simple and complex activities to better reflect activities of daily living. In this article, we have adopted the classification taxonomy developed by [2] and [23].

For HAR models to achieve state-of-the-art performance when discriminating between simple and complex activities, Blanke et al[27] suggested that collecting data from multiple sensors should be explored because of the hierarchical structures in determining the different levels of physical activities. Although [10] agrees with this heterogenous data collection approach, they also identify the prevalent inter-class similarity challenges associated with HAR, most especially when the dataset comprises both simple and complex activities. They define inter-class similarity as the potentials for activity classes that are fundamentally different to exhibit very similar patterns in the signal data. Finally, they recommend applying an Activity Recognition Chain (ARC) to mitigate against these challenges. ARC is a specific activity recognition system behavior that comprises signal processing, pattern recognition, and machine learning algorithms.

2.2 Context Awareness and Inter-similarity in Complex HAR

The role of context recognition in complex activity detection cannot be overemphasised. Therefore, understanding the context in which an activity is performed can enhance the ability of HAR models to identify the complexity of such activity[28]. Researchers have, over the years, developed models for activity contexts recognition using additional sensor signals other than the traditional inertial sensors (accelerometer, gyroscope, and magnetometer). The rationale is to acquire as much information as possible from integrated or nearby ambient sensing devices and use the same to train the deep neural networks. Using this rich contextual information, existing research works have been able to recognise different activities that are difficult to identify using fewer sensor modalities. In recent research conducted by [29] they used ambient signals from audio sensors combined with inertial signals to pre-train a CNN model for automatic human activity recognition. Similarly, [30] combines ambient signals from audio sensors and video cameras in addition to other inertial sensors to detect activities of the elderly in order to recommend early intervention and rehabilitation. To address inter-class similarity in HAR, [31] relied on camera signals from sports videos and developed a model that employs hierarchical matching using a Consistency-correlation driven Feature Selection. To address the same challenge, [32] combine sensing information from multiple sensor modalities including traditional sensors (accelerometer and gyroscope) and ambient sensors (atmospheric pressure, temperature and humidity sensors).

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

In a way, these works are very similar to the current work, however, we did not only use inertial signals from smartphones, but we also investigated the significance of ambient sensing in conjunction with the inertial sensors to address the challenge of interclass similarity in the dataset. Thus, in addition to the traditional inertial sensors, we proposed two (2) extra ambient signals (ambient illumination and noise level signals) to augment the traditional inertial sensing signals.

2.3 Machine Learning for Complex HAR

Early 2012, [26] conducted one of the first studies to have focused on simple and complex activities recognition using data collected from smartphone accelerometers and gyroscope sensors. Although the model had a poor 50% accuracy at recognising complex activities, it achieved over 93% for simple activities suggesting that relying on smartphone sensors alone may be inadequate for detecting complex activities. However, other studies revealed that the size of the segmentation window could lead to poor performance of the model for classifying complex activities, suggesting that some efforts are required to determine the optimal window size [33-34]. The model is designed in a way that automatically detects the appropriate window size, thus, obviating the need for a pre-specified window length.

In addition, recognition of several different complex activities has contributed immensely to providing context-aware feedback in various well-being mobile applications [35]. Unlike the proposed model, most of these works focus on one complex activity and several simple activities. For instance, Ramos-[36] focused on eating while [37] concentrated on smoking. In addition, most of these studies used data from accelerometers and gyroscopes only, while the work explores a few more complex activities using additional sensors such as the smartphone magnetometer. Although another piece of work combined data from accelerometer, gyroscope and magnetometer on the wrist to detect smoking puffs, the models were only able to detect smoking activity [38]. The work explores the application of deep learning models to generalise more than one complex activity in human activities of daily living.

So, how has deep learning helped in generalising complex activities? There have been a good number of works, which explore the application of Deep Neural Networks (DNN) to HAR models. Early works carried out by some authors [19, 39] to examine the feasibility of deep learning in the HAR space paved the way to other related studies in this field. Motivated by this, other scholars [16-17] attempted activity classification by starting with a hand-crafted feature extraction process. The models performed very poorly simply because the Deep Convolutional Neural Network (CNN) was only explored as a classification model and certainly not used for feature extraction. On the other hand, Hammerla et al(2016) achieved a better result when they employed a 5-hidden-layer CNN to carry out automatic feature extraction and classification. Similarly, other studies have been able to replicate and achieve similar results[14, 40]. This new approach allows feature extraction and model building tasks to be done through the network at the same time, thus making it suitable for classifying composite activities. Although the work also leverages CNN's automatic feature extraction, we additionally investigated the impact of different window lengths on the recognition accuracy as suggested by [41]. Furthermore, we explored the recommendations from other studies [10, 42-44]. by incorporating temporal and environmental parameters such as noise level and illumination data respectively into the model. These authors have all suggested that contextual information, when combined in an optimal manner, can significantly improve HAR model's generalisation ability.

Several other works explored the Deep Recurrent Neural Networks (RNN), a deep learning model known to be widely applicable in natural language processing and speech recognition. However, these works have some limitations: consuming too a lot of resources and a long learning curve[45-48]. [49] applied the RNN successfully on a number of HAR datasets with impressive results. They commented on the limitation of CNNs to operate effectively on fixed-size windows of sensor data, a limitation that Long Short-Term Memory Networks (LSTM) does not suffer from. LSTM is a specific type of RNN widely used with time-series data because of its ability to learn temporal dependencies in a sequence. In a related study conducted to detect anomalies in time series, [50] demonstrated the ability of the LSTM

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

networks to maintain long-term memory and learn sequences of data containing long-term patterns of unspecified length. This capability of LSTM makes it a suitable candidate for the complex activity recognition system. More often than not, complex activities take longer to detect compared to the simple counterparts [14-15]. By splitting the window sequences into sub-sequences, we are able to reuse the same CNN model when reading in each sub-sequence of data separately. We achieved this

by wrapping the CNN layers in a Time Distributed wrapper and applied the entire model once per input subsequence. The extracted features are then flattened and provided to the LSTM model to read, extracting its own features before a final mapping to an activity is made.

Existing works rely on using an LSTM with a CNN to tackle the identified challenges resulting in a CNN-LSTM or ConvLSTM model[2]. This combination of CNNs and RNNs is an emerging methodology adopted lately by researchers to replace the previously conventional independent learning for multiple related tasks. One of such works by [51] implemented the hybrid model: a combination of CNNs and unidirectional RNN-based Long Short-Term Memory (LTSM) recurrent layers to classify activities from wearable sensors. Another work by [47] examined the impact of different deep learning models employing the deep feed-forward network, CNN, and bi-directional LSTM network on HAR problems. In the same vein, [52] implemented a hybrid CNN-LSTM model and recorded significant improvement in comparison to previous works using similar datasets.

Recently, some studies [14, 19, 53] focused on activity recognition by taking advantage of the hybrid multi-task learning models. In their work, [19] implemented a CNN model combined with a bi-directional LSTM network to extract task-dependent simple activities and classify complex activities. The model, tagged AROMA, utilises the CNN layer to extract deep features followed by a Softmax classifier for predicting simple activities. The deep features from the CNN layer are then concatenated together to form a deep feature layer which serves as input for the LSTM network. AROMA then implements another Softmax classifier to predict complex activities. Also, [53] adopted the multi-task learning approach to discriminate between simple and complex activity labels. The results are most impressive, suggesting that combining CNN with LSTM network can achieve higher generalisation accuracy than just using either of them alone. These works provide the basis for the research although not even one of these studies explores the combination of rich contextual information provided by the smartphone ambient sensors to augment the inertial sensing signals. The work is unique in that we took a step further to improve recognition accuracy significantly by exploring these contextual dependencies.

More specifically, the proposed solution only utilises sensor data from smartphones as opposed to other models that require multiple wearable and non-wearable sensors, which are far from ideal and intrusive. Although there are similar works that demonstrate the adequacy of smartphone sensor data in activity recognition [40, 54-58] to the best of the knowledge, the proposed work is the first to combine rich contextual information from ambient sensors with low-level sensors signals from the inertial sensors to distinguish between simple and complex activities and address associated inter-class similarity problems. The proposed solution also demonstrates significant improvement of recognition accuracy using raw sensor signals without heuristic hand-crafted features.

3. Proposed Methodology

In this section, we present the approach and methods of the proposed solution for generalising simple and complex activities using raw sensing signals from smartphone sensors.

3.1 Problem Definition

We start by clarifying a few terminologies. Figure 3 below represents the high-level workflow of the work. Experiments on smartphone-based activity recognition require activity signals being collected from the user via smartphone sensors [2]. The sensing signals are usually

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

multivariate time-series data. Multivariate refers to multiple inputs, for example, the accelerometer signals come in 3 axes (x, y, z). Moreover, the smartphone has several sensors ranging from inertial, ambient, and environmental sensors. When signals are collected from multiple sensors, such a dataset is referred to as multimodal i.e. multiple sensor modes. State-of-the-art HAR models are usually trained using a multimodal multivariate dataset [18].

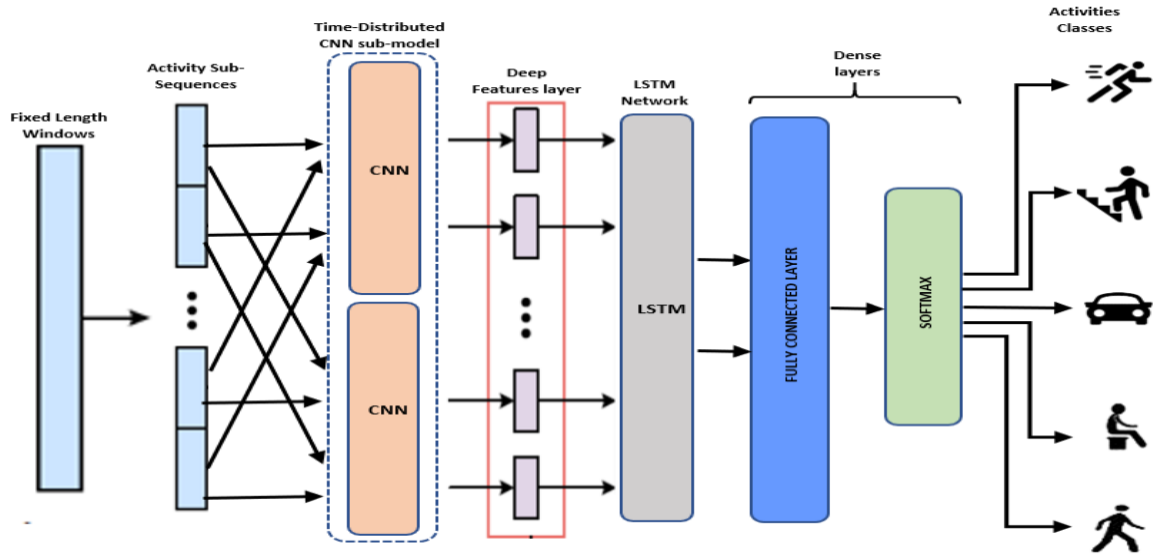


Figure 3. Overview of the proposed context-aware smartphone-based human activity recognition using Hybrid CNN-LSTM Model

As illustrated in Figure 2 in section 1, the input data consists of a combination of inertial and ambient data signals. In the next section, we describe the input data in detail.

3.2 Inertial Sensor Signals

According to a recent survey conducted by [53] the inertial sensors, also known as motion sensors, are the most commonly used for HAR systems. These set of sensors, compared to the videos-based sensors, capture body movements effectively without posing any privacy issues. The accelerometer, gyroscope and magnetometer are the most popular examples of inertial sensors which are now freely integrated into smartphones, smartwatches and sometimes on clothes. In this work, we utilised sensing signals from the 3 sensors to train the baseline model, just like many other studies[38, 51]. The raw signals from the inertial sensors are pre-processed and passed to a convolution filter for feature extraction.

The accelerometer, for instance, is used to measure acceleration, same as rate of change of velocity of an object. The unit of measurement is meters per second squared (m/s^2) along the 3 axes (x, y, and z). Therefore, each sample is made up of a tri-variate time series. The gyroscope is another motion sensor used for measuring orientation and angular velocity. The unit of measurement is degree per seconds ($^o/s$). In contrast to the accelerometer which measures change in linear velocity, the gyroscope is a more advanced sensor which captures angular and lateral orientation of an object. Lastly, the magnetometer is a motion sensor often installed together with an accelerometer and gyroscope into an inertial unit. It helps measure the change in magnetic field at specific locations. Recently, the magnetometer, often installed in aircrafts, has become a common sensing equipment in mobile devices to detect orientation

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

relative to the Earth's magnetic north [2]. Likewise, the magnetometer captures signals in three axes i.e. tri-variate as the accelerometer and gyroscope.

3.3 Environment sensor signals

Interactions between humans and the environment are usually captured using ambient sensors embedded on devices present in the same physical location. These sensors are good at detecting multi-occupant activities [2]. Examples of ambient sensors include WiFi, Bluetooth, sound, light, pressure sensors etc. In this work, the approach is to augment traditional inertial sensing data with rich contextual data captured by ambient sensors to train the classifier to distinguish between simple and complex activities. Several other works used a similar approach to achieve state-of-the-art results [29]. However, their work did not combine the audio and light sensors as proposed in this article. In addition to the inertial signals, we utilised ambient signals from the audio and light sensors to train and improve the baseline model to create a context-aware model.

The sound or audio sensors are usually made up of microphones and speakers. The microphone receives ultrasound signals while the speaker transmits same. The idea is that human movement activities and interactions create different level of environmental sounds, and these can be captured by the microphone as noise level contexts. Likewise, the light sensor captures the intensity of light i.e. illumination level as at the time these interactions occur. Together, these two sensors create rich contextual data that can be passed to the classifier to make it context aware.

3.4. The Proposed CNN-LSTM Hybrid Model

The proposed CNN-LSTM hybrid model uses the Time Distributed CNN sub-model for automatic learning and extraction of discriminatory features from raw input signals, and the LSTM networks for frequency and temporal modelling. First, a CNN as illustrated in Figure 4 is usually made up of the following components: an input layer, multiple hidden layers, and an output layer. The input layer is determined by the input signals. Each of the multiple hidden layers can either be a convolutional layer followed by an activation function, a pooling layer, and a fully connected layer (FC). The main difference between the two models is at the *TimeDistributed* input layer where one model trains on 9 input features as against 11 in the other.

a. **The Convolutional (Conv1D) layer:** between the convolutions, data processing happens layer by layer as the output of one layer is the input of another layer. Let us assume that $x_i^a = [x_1, x_2, \dots, x_N]$ are the inputs from the sensors, and a represents the axis of the sensor. Going by the literatures, given l number of **convolution layers**, the feature map of the l-th layer is derived using the following equation:

$$z_i^{l,j} = \sigma\left(\sum_{k=1}^{K} w_k^j x_{i+k-1}^{l-1,j} + b_j^l\right) \quad (1)$$

where w_k^j and b_j^l represents the weight and bias of the j-th term of the l-th layer. While l stands for the index of the current layer, σ is the activation function and $x_{i+k-1}^{l-1,j}$ is the input patch. K is the size of the CNN kernel or filter.

b. **The Pooling layer:** we added a pooling layer to the CNN sub-model to reduce the spatial size of representations i.e. a form non-linear down-sampling operation to help reduce the feature maps. It is a common practise to have a pooling layer between successive

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

convolutions in a CNN [14]. In this article, we use the max-pooling operation in mapping the output of the preceding layer.

$$f_i^{l+1} = max_{k=1}^{r}(f_{i+k}^{l}) \quad (2)$$

where f_i^l is the value of the i-th unit in layer l, and r is the length of the pooling.

c. **The Flatten layer:** we included a Flatten layer in the model just immediately after the pooling layer. The flatten layer, upon receiving the output of the pooling layer, converts the reduced feature maps into a single column vector. These are one-dimensional vector of features such that $f^l = [f_1, f_2 \ldots, f_l]$, where *l* represents the number of nodes in the last pooling layer.

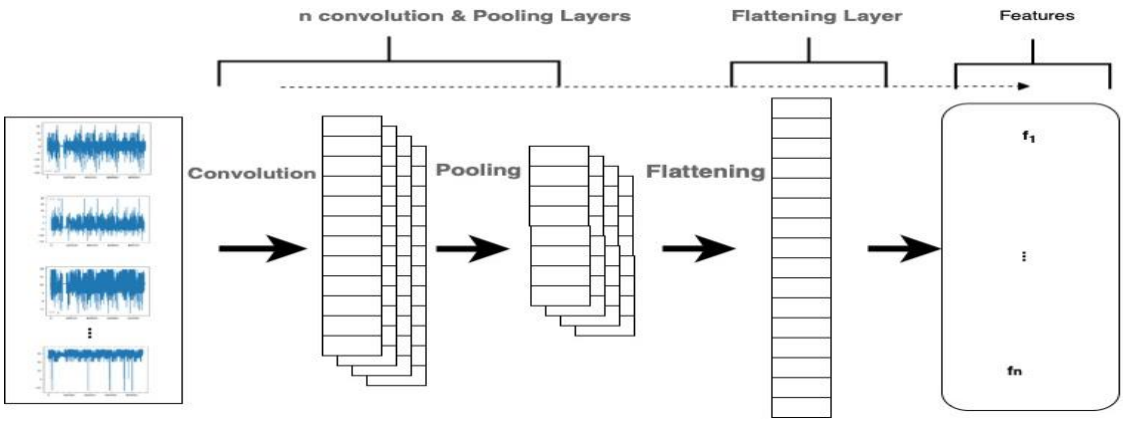


Figure 4.Automatic Feature Extraction Using CNN

d. **The LSTM Networks:** an interesting part of the model in the introduction of the LSTM networks immediately after the Flatten layer. This is where the current work differs significantly to that conducted recently by researcher in Otebolaku et al in [59]. According to [22] for models that need to learn the temporal patterns in input data, preceding an LSTM network with a CNN sub-model always outperforms models with only one of the two. LSTM, like any other recurrent neural network, supports forward and backward propagation within its networks. Made up of several LSTM units, the LSTM networks use its memory cells to master the temporal context in input data. Each of the units has a memory cell c_t which is readable, updatable, and erasable [14]. As mentioned earlier,

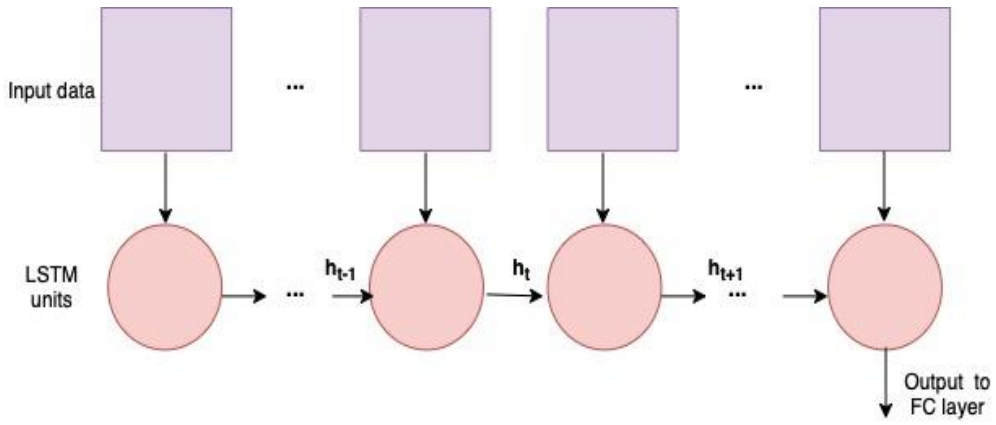


Figure 5. A single layer of LSTM Network

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

LSTMs comprises the input gate i_t, forget gate f_t, and output gate o_t which control reading, memory update, and writing operations, respectively.

A typical one-layer LSTM network as illustrated in Figure 5 comprises specified LSTM units, each of which takes input data x_t, cell state c_{t-1}, and hidden state h_{t-1} from the previous LSTM unit. At every time step t, the hidden state gets updated. This operation can be summarised as follows:

$$i_t = sigm(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \quad (3)$$
$$f_t = sigm(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \quad (4)$$
$$c_t = f_t \circ c_{t-1} + i_t \circ tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (5)$$
$$o_t = sigm(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_{t-1} + b_o) \quad (6)$$
$$h_t = o_t \circ c_t) \quad (7)$$

where c_t and h_t are the outputs of the LSTM unit that can be passed to the successive LSTM unit in the next time step and the iteration continues until the last LSTM unit in the network. Operator ° is the element-wise multiplication. W represents the weight matrix, with subscripts indicating from-to relationship. W_{xi} and W_{hf} are the input-input gate and hidden-forget gate matrices, respectively. Variables b_i, b_f, b_c, and b_o are the bias vectors.

As shown in Figures 5 & 6(Figure 6 representing, LSTM architecture) we implemented a single-layer LSTM network according to[18]. By passing the output of the previous LSTM unit into the next, the network can begin to perform frequency and temporal modelling of the activity contexts. The purple boxes are the inputs i.e. the one-dimensional vector of features from the Flatten layer of the TimeDistributed CNN sub-model. As suggested by[22] and [51], we passed the output of the single-layer LSTM network to a fully connected dense layer with a Softmax activation function.

e. **The Fully Connected (FC) layer:** this is a dense layer, known for producing higher-order feature representation that is easily interpreted into the different activity classes by a SoftMax classifier. Finally, we are ready to predict the activity labels. Before then, let us see how the dense layer implements the probability distribution. Given the output from the LSTM layer i.e. h_t, predictions can be performed and the activity probability distribution vector pc = $[pc_1, pc_2, \ldots, pc_m]$ is derived as follows:

$$pc = s(W_{hm}^T h_t + b_m) \quad (8)$$

where *m* is the number of activity classes, W_{hm} and b_m are the weight and bias, respectively, for the output layer. The $s()$ is the softmax function with the following equation:

$$f(x) = P(x) = \frac{e^{pc_y al}}{\sum_{j=1}^{n} e^{pc_j}} \quad (9)$$

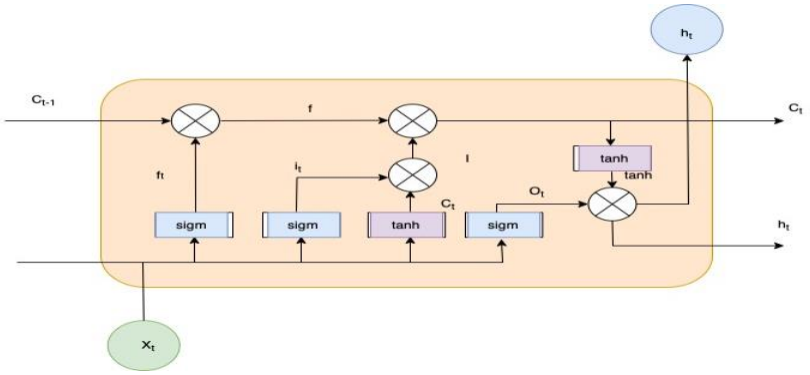


Figure 6. Architecture of LSTM Cell

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

where al is an activity label, y is the output of the activity classifier $f(x)$, n is the number of activity labels, and pc_j means the j-th element of unnormalized log probability vector pc. The predicted activity label is assigned to the one with the highest probability, i.e., $al \leftarrow argmax_{al=1}^{n} P(y = al|x)$. In the case, both the simple and complex activities are learnt concurrently since the dataset is single label. We used the classification methodology suggested by [23, 53] to categorise activities within the dataset into simple and complex labels.

3.4 Data Pre-processing and segmentation

To achieve the fixed length windows, one key requirement is to divide the standardized data into segments using a temporal sliding window algorithm. According to [18], the algorithm divides the data into windows of signals, where a given window is associated with a specific activity and usually have one to a few seconds of observation data. Two parameters are configured to achieve this: the size of the window and the shift [60]. The shift parameter helps create the overlap between windows as seen in Figure 7 below. In the case of a 50% shift, which is commonly used in many HAR literatures, an overlap is created such that the first half of a window contains the observations from the last half of the previous window [32, 47, 61]. In this work, we used a window length of 32 (circa 0.75 seconds of sensor signals) and a 50% shift. Detailed analysis of different window lengths and how we arrived at the choice of 32 can be found section 4.4.2.

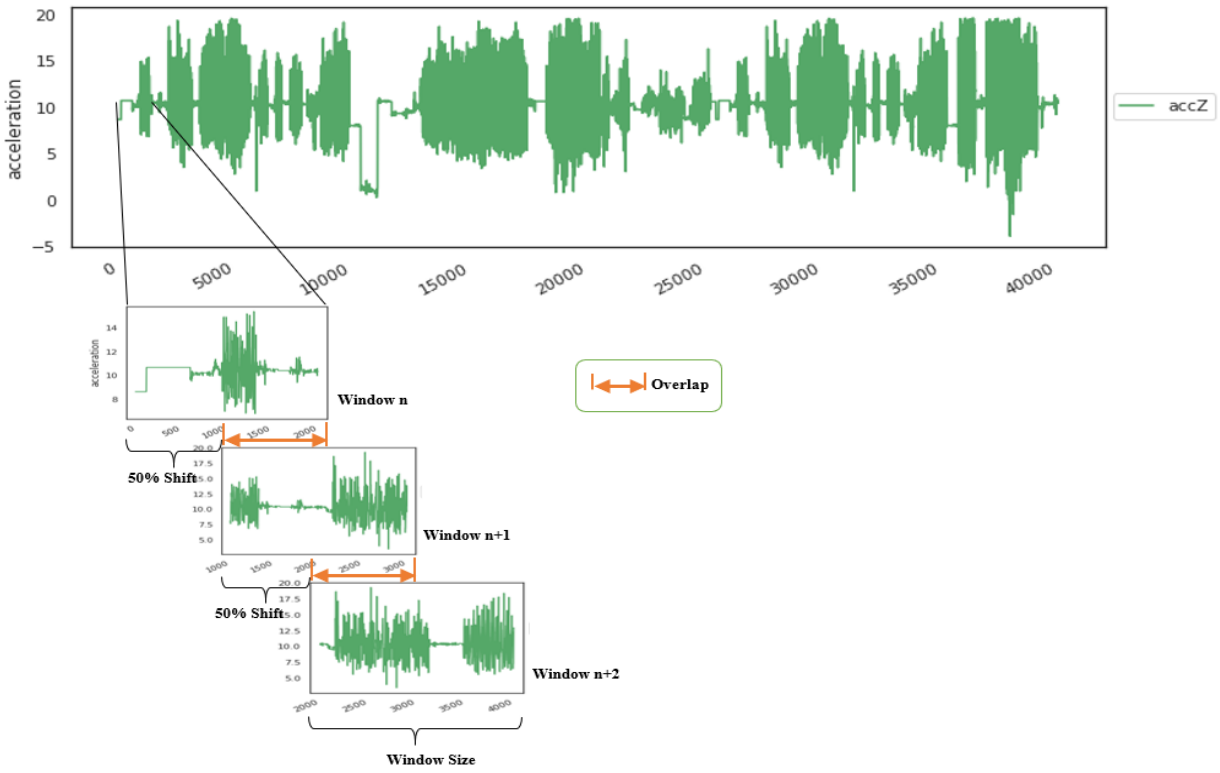


Figure 7 Sliding window with 50% overlap

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

3.5 Hyper-Parameter Tuning

To ensure optimal performance for the model, we evaluated the influence of several hyper-parameter settings on performance of the model. This process is referred to as Hyperparameter tuning. We adopted the grid search methodology, a tested and reliable method for tuning hyperparameters[16, 52]. Grid search methodology proved to be the optimal approach for setting the hyper-parameters by creating a grid of a certain range and then looping through the candidate elements to find the best values for each hyperparameter. There are several studies that provide sufficient guidance on how to approach grid search, especially when a hybrid CNN-LSTM model is involved. [22, 61]. Both works analysed the effect and shared the many benefits of adding the CNN layers before the LSTM to solve speech and activity recognition problems. We approached the hyper-parameter tuning by first optimising specific components of the model and then, we moved on to tuning the model for overfitting. For the sub-component tuning, a common practise is to have 2 consecutive CNN layers followed by a dropout and a max-pooling layer[18]. We adopted this simple structure for the CNN-LSTM model and train the 2 fully connected 1-Dimensional convolutional (Conv1D) layers with 9 feature maps representing the 3 (x, y, z) axes of the 3 inertial sensors (accelerometer, gyroscope and magnetometer). The number of nodes in the hidden layer was replaced by the number of filter maps and kernel size which we grid-search as well. The number of filters ranges between (16, 32, 64, 128, 256) while the kernels investigated are between 2 and 7 in the following order (2, 3, 4, 5, 7).
Moreover, for Conv1D layers, the input data must be in the form of [samples, timesteps, features] where features map onto the channels [18] and, in the case, it is 9 for the 9 variables we have from the inertial sensors. The approach to implementing this model is to split each window of 64 time-steps, for example, into sub-sequences for the CNN model to process. The 64 time-steps in each window can then be split into 4 sub-sequences of 16 time-steps. As a result, we defined a CNN model that reads in sequences with a length of 16 time-steps and 9 features. The entire CNN model is also wrapped in a TimeDistributed layer to allow the same CNN model to read in each of the sub-sequences in the window. The extracted features are then flattened and provided to the LSTM model to read, extracting its own features before a final mapping to an activity is made.

Another important parameter that requires tuning is the batch size. The training data is fed to the model in one or more batches. A batch represents the collection of samples that the model processes prior to its weights being updated. Depending on the dataset, adequate measures must be taken to ensure that the model is being fed with the right batch size for it to learn optimally [62]. Using the grid search, we passed batches ranging from 16 to 512 to train the model for each window length to inspect in what way the interaction between the two parameters affects model performance.
To prevent the model from overfitting, we optimised several parameters including weight decay, dropout regularisation, number of epochs and learning rate. Weight decay or weight regularisation provides an approach to reduce overfitting on the training data and improve generalisation capability on new data. We tuned the L2 weight regularisation for both the CNN layers and the LSTM cells. We confirmed from literature that the CNN layers usually require small L2 weight decay to perform optimally [59, 63].
Dropout regularisation is another way to prevent the model from overfitting. This is often referred to as Dropout rate, which can be specified for different layers of the model, and it defines the probability of setting each input to the layer to zero [18, 64]. In other words, a

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

dropout rate of 0.3 informs the hidden layer to set 30% of inputs to zero. For the hybrid model, we grid search between 0 and 0.9 dropout rates for each of the hidden layers i.e. the CNN layers and the LSTM cells.

Models often overfit or underfit when they train continuously to the end of a fixed number of epochs. Two things happen when a fixed number of epochs is set: either the model is interrupted while it is still learning i.e., underfit or it learns so much that it starts memorising the training data i.e. overfitting. The literature identifies two methods suitable for preventing this: early stopping and model checkpointing[61]. We invoked the Keras *EarlyStopping* call-back and specified the validation loss as the performance metric to monitor in order to end training. We also set a patience of 8 epochs to add a delay before stopping. Finally, we added a second call-back called *ModelCheckpoint* to save the best model achieved during the training process.

Finally, deep neural networks are trained using the stochastic gradient descent optimization algorithm which estimates the error gradient of the model being trained and then update the model's weights by passing this error back to it in a process known as back-propagation [16, 22, 47]. The amount of the model's weight that is being updated during this training is known as the learning rate. It is often regarded as the most important hyperparameter for deep neural networks [47]. The value is usually between 0.0 and 1.0 and choosing the sub-optimal rate is always a challenge. Masters and Luschi in [62], in one of their interesting papers, recommend tuning the learning rate after all other parameters. They suggested that even if the learning rate had been tuned earlier, effort should still be made to further re-optimise it at the end of all other parameter tuning because the learning rate tends to interact slightly with other parameters. In this article, we did tune the learning rate twice, one before tuning the sub-components of the model and the other at the very end of all other hyper-parameters.

4. Experiments and Evaluation

In this section, we present the evaluation experiments of the proposed solution. First, we discuss the main two types of data used in the experiments: the traditional inertial sensors data and the ambient contextual data.

4.1 Experimental Setup

The dataset for this experiment was collected exclusively from the built-in inertial and ambient sensors of a smartphone. A comprehensive description of the entire data collection process can be found in these papers [42, 65]. To the best of the knowledge, there is not one publicly available HAR dataset that is exclusively collected over the smartphone and comprises both traditional sensor signals and rich contextual information from the ambient sensors. Similarly, other authors who have used custom datasets in their work also corroborated this fact[32, 66].

To answer the research question, two experiments were conducted in this study. What differentiates the two experiments is the dataset used. In the first experiment, we trained the deep hybrid model using traditional inertial sensor signals, while in the second experiments, we used a richer dataset containing additional signals from the ambient sensors. The model was consistent for the two experiments, only the datasets changed. Next, we describe the two datasets. In the next section, we present details of the different experiments and the datasets used.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

4.2 Traditional inertial sensory data

For this experiment, we used data collected from the traditional inertial sensors i.e. accelerometer and gyroscope. We also used data from the magnetic sensors unlike previous studies that used data from one or two of these sensors [26, 33, 36-37, 67]. Subsequently, we refer to this experiment as *Non-contextual*. The dataset was collected using a mobile app developed by
Otebolaku and Andrade [65] and customised specifically for collecting context-aware human activities. Figure 8 highlights the dataset's distribution per activity class. Basic exploratory data analysis reveals that the dataset suffers from an extreme class imbalance with only about 4 activity classes representing approximately 77% of the entire dataset. The choice of evaluation metric thus takes this into consideration.

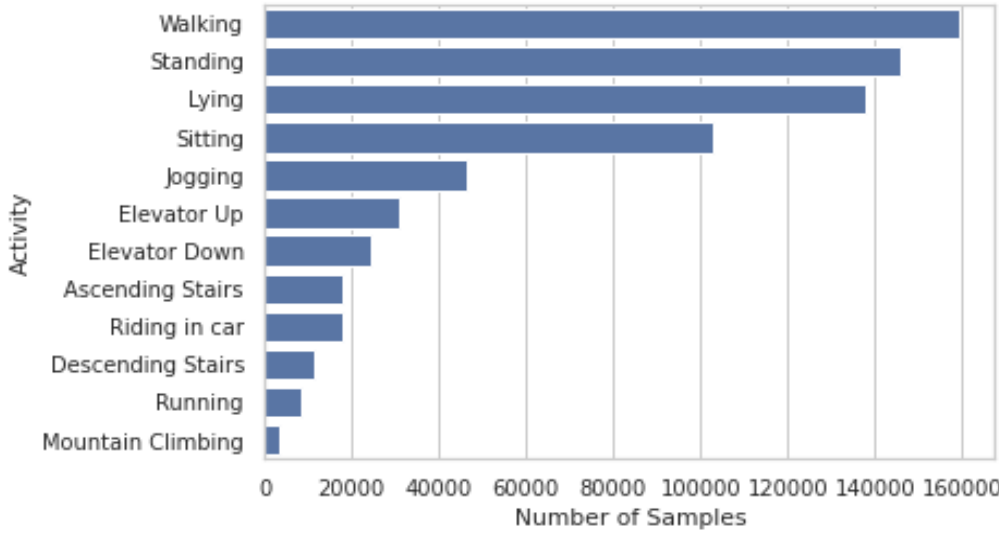


Figure 8: Activity classes by number of samples

4.3 Inertial and Environment Contextual Data

For the second experiment, which we refer to as *Contextual,* additional sensing data was introduced compared to the *Non-contextual* experiment. To increase the sensor modality and investigate the significance of rich contextual information on the model, we augment the inertial sensing data in the *Non-contextual* experiment with additional sensor signals from the ambient sensors. The ambient sensors used include the audio(microphones) and the light sensors in addition to the traditional inertia sensors used initially. This makes a total of five sensor modalities - accelerometer, gyroscope, magnetometer, audio and light – altogether collecting sensing signals for 12 activity classes. Further processing was carried out on the audio and light sensor signals to transform them into the representations of environmental noise level and illumination data.

The absence of other similar datasets relevant to the research is one drawback that we plan to address in the future research. Lastly, all the experiments were implemented in Python using the TensorFlow machine learning framework which comes bundled with Keras high-level API [68].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

4.4 Simple and Complex Activities Taxonomy

Given that the dataset is a mix of both simple and complex activities and following the classification approach suggested by other researchers [2, 23], we were able to identify the simple activities in the dataset to include Sitting, Standing and Walking. Others are Jogging, Lying and Running. The complex activities which fall under either of composite, concurrent or multi-occupant activities include Elevator Up, Elevator Down, Ascending Stairs, Descending Stairs, Mountain Climbing, and Riding in a car. We determined the simple activity category by analysing which activities are atomic i.e. cannot be further broken down, and those that can be broken down into other atomic activities are categorised as complex in nature. For instance, riding in a car can easily be broken down into sitting down and in motion (see Table 1 for the break-down).

Table 1. Simple and Complex Activities within dataset

#	Activity	Category	Sub-activity
1	Jogging	Simple	N/A
2	Lying	Simple	N/A
3	Running	Simple	N/A
4	Sitting	Simple	N/A
5	Standing	Simple	N/A
6	Walking	Simple	N/A
7	Ascending Stairs	Complex	Walking + in motion against gravity
8	Descending Stairs	Complex	Walking + in motion in line with gravity
9	Elevator Down	Complex	Standing + in motion in line with gravity
10	Elevator Up	Complex	Standing + in motion against gravity
11	Mountain Climbing	Complex	Walking + under high altitude
12	Riding in car	Complex	Sitting + in motion

4.5. Evaluation Metrics

We evaluated the classifiers from the two experiments using a hold-out test dataset. The hold-out test data came from users whose data were not used as part of the training and validation data. As part of the data pre-processing stage, the entire dataset was split into training, validation, and testing sets. The test data came from a user, validation data from 2 other users while the remaining data from 4 other users were used during training. This approach is often referred to as leave-one-subject-out validation in literature [10].

For both of the experiments, we used the following metrics: Precision, Recall, F-Score and the Confusion matrix. According to many authors, these are the most commonly used metrics in the field of activity recognition [10, 69-70]. As clearly illustrated in Figure 10, the dataset suffers from class imbalance. Accuracy as a performance measure, therefore, will be inappropriate for the experiments. This is a common position shared in several literatures. For instance, Peng et al.[14] explained that, when presented with a class imbalanced dataset, the number of samples from the majority class will certainly overwhelm the number of samples in the minority class. Suffice to say that even the worst model can achieve high accuracy above the 90% mark depending on the severity of the class imbalance. Given the highly imbalanced nature of the dataset, the choice of evaluation metrics - Precision, Recall, F1-Score and the Confusion matrix are well justified.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

Before discussing Precision, Recall and the F-Score, let us first define the confusion matrix. This is because the precision and recall values, as well as the harmonic mean of precision and recall i.e. the F-score can all be computed from the confusion matrix. The confusion matrix provides a summary of the number of instances of different activity classes that the model is confused about i.e. misclassified by the model[10]. For a more fine-grained analysis of predictions by the model, the confusion matrix provides insight into not only the performance of a predictive model, but also a breakdown of which classes are being predicted correctly (True Positive and True Negative), incorrectly (False Negative and False Positive) and the type of errors being made. However, the numbers can be strongly biased by dominant activity classes compared to the minority classes, more so when a class imbalance dataset is involved.

Precision is computed as the sum of True Positives (TP) predictions across all classes divided by the sum of True Positive predictions and False Positive (FP) predictions across all classes. Precision is defined as:

$$P = TP/(TP+FP).$$

Recall is another metric that computes the number of True Positive predictions divided by the sum of True Positive and False Negative predictions i.e.

$$R = TP/(TP+FN)$$

Finally, once the precision and the recall metrics are known, we then compute the F-score which is the harmonic mean between the recall and precision. The F-score provides a way to combine the precision and recall measures into a single measure that reflects both properties. The F-score remains the most commonly used metric for class imbalanced datasets (He & Ma, 2013, p. 27). Considering the imbalance nature of the dataset and the tendency of the classifier to skew towards the more frequent classes during training, we adopted the macro-average F-score previously used successfully by other scholars [47, 59]. This metric truly reflects the representation of small classes within the dataset and it is calculated as follows:

$$F\text{-}Score\ (R, P) = 2*RP/(R+P)$$

Just like the Precision and Recall measures, a perfect F-score is 1.0 and a poor F-score is 0.

4.6 Experiments and Performance Evaluation

We conducted the experiments using the Google Colab environment, an open-source python-based machine learning platform that leverages the power of Google hardware, including GPUs and TPUs. Google Colab is known for its tremendous support for training neural networks. Moreover, it requires zero configuration, free access to GPUs and much faster than any other easily accessible hardware. This environment also provides leverage on the open-source interfaces such as the TensorFlow machine learning framework which comes bundled with Keras high-level API [68].

In this work, two different models were built, one from the ***Non-contextual*** **experiment** and the other from the ***Contextual*** **experiment**. We refer to the *Non-contextual* model as the baseline model. The baseline model was trained using the inertial signals only. While the *Contextual* model was trained on both the inertial and ambient signals. We compare results from the two different models using the F-measure. We trained the models on the training data and validated the model with the validation dataset. Class prediction using test data that is totally new to the model is the approach adopted to test the performance of the final model.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

In a bid to ensure that all the different activity classes are well represented in each of the sets, we split the custom dataset into train and test sets using ratio 85:15. A further split on the train set to create train and validation sets was at ratio 80:20, giving us a 68:17:15 split between the train, validation and test sets. Figure 9 illustrates the splits showing training (blue signals), validation (orange signals) and test (green) samples of the accelerometer x-axis for *Ascending Stairs* activity.

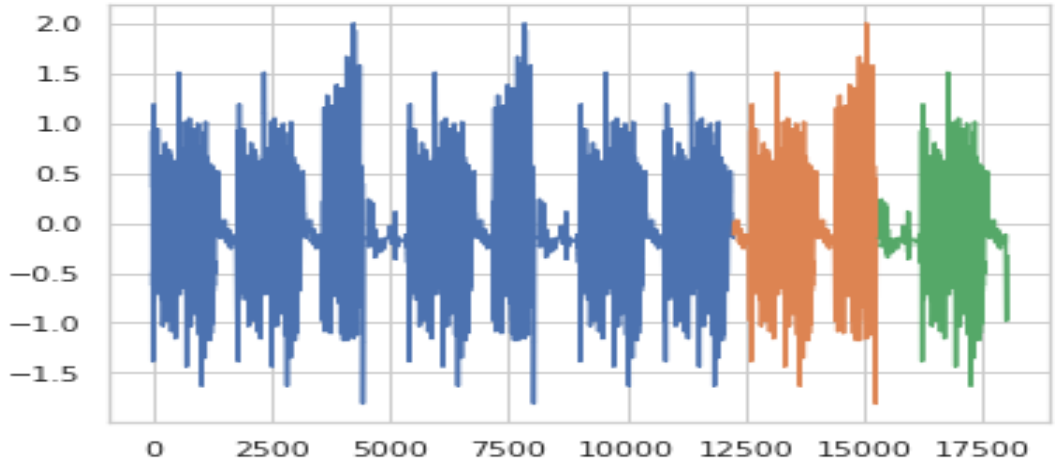


Figure 9. Accelerometer (x-axis) data samples for Ascending Stairs Activity

Prior to the two main experiments, a series of preliminary experiments was carried out to investigate the impact of several different hyper-parameters on the performance of the model. Moreover, we tuned the model appropriately to prevent it from overfitting. For the *Non-contextual* experiment, we investigated the performance of the model in distinguishing between simple and complex activities using a dataset with inter-class similarity. Lastly, the *Contextual* experiment helps us evaluate the extent to which rich contextual information can help improve the model's performance in recognising simple and complex activities.

4.6.1 Tuning for Conv1D filter and kernel

After conducting a grid search to determine the number of neurons for the Conv1D, Figures 10 and 11 show the model's performance by number of filters and kernel size respectively. Clearly, the model performance continues to improve as the number of filters increases. A decline is however noticed after the 256-mark suggesting this might be the peak. Accuracy was on a steady downward trend as the number of kernels increases also suggesting smaller kernels give better performance.

Table 2. Accuracy per kernel size for filter size 64

kernel=2: 97.674% (+/0.395)
kernel=3: 97.900% (+/0.307)
kernel=4: 97.703% (+/0.423)
kernel=5: 97.514% (+/0.273)
kernel=7: 97.099% (+/0.513)

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

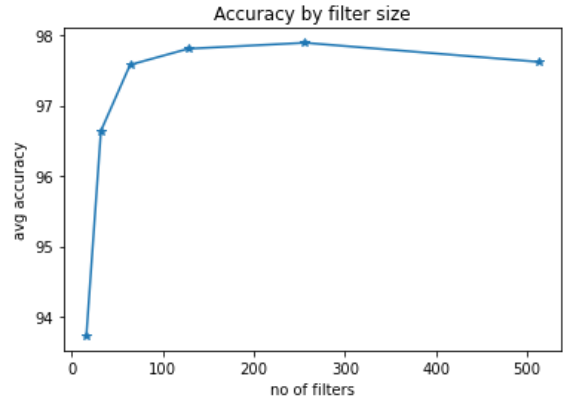


Figure10. Average Accuracy by Number of Filters

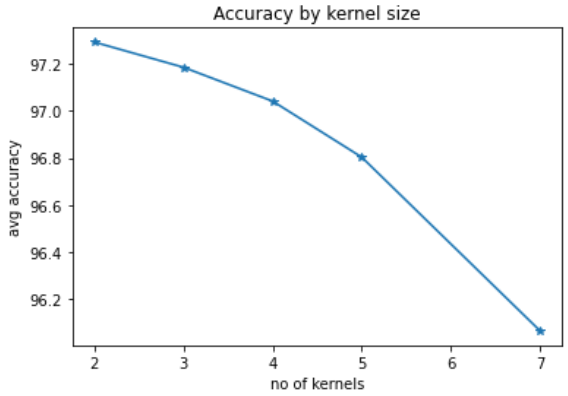


Figure 11. Average Accuracy by Kernel Size

We examined the interaction between the two and discovered that, once again, larger kernel size does not appear to yield better accuracy. As shown in the boxplot below (Figure 12), kernel

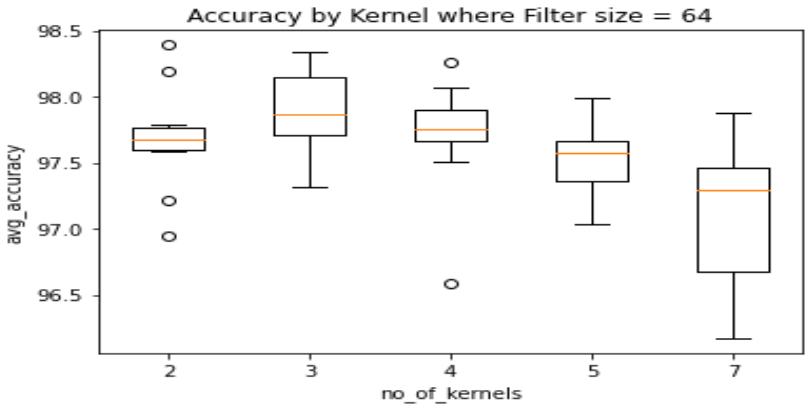


Figure 12. Box plot of different kernels for filter size 64

size 3 and filter size 64 provide a good balance between classification accuracy and computational efficiency.

Boxplots showing results of all other pairs of filter sizes (16, 32, 64, 128, 256, 512) and kernel sizes (2, 3, 4, 5, 7) can be found in the appendix. Although there are few other pairs with better average accuracy, for instance, filter size 256 and kernel size 2, this does come at the expense of more computational cost.

4.6.2 Tuning for Window Size segmentation

As part of the hyperparameter tuning, we investigated the optimal window size for the baseline model. For the dataset, about 128 samples are captured every 3 seconds from both the inertial and ambient sensors on the smartphone [7, 65]. Some scholars suggested that sensing a complex activity from the smartphone sensors may require a longer window length compared to the simple activities [14-15]. How true, and if true, how long is suitable for the dataset? It is crucial we experiment with different window lengths to ascertain the optimal window segmentation required to capture the complex activities in the dataset. Using the sliding window algorithm, we investigated several different window lengths ranging between 32 and 256. Concurrently, we examined the interaction between the window length and various batch sizes on a model's performance since the model learns from one batch of data at a time before its weight is being updated [18].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

The result (Figures 13 and 14) shows that increasing the window length degrades the performance drastically, while increase in batch size proves contrary. However, there is no significant improvement once the batch size passes the 64-mark. The drop in performance as window length increases may not have come as a surprise as some authors have suggested that models containing LSTM layers suffer from poor parallelism across longer window lengths [61]. Although we did not investigate further, Chambers and Yoder [61] suggest that increasing the LSTM layers can enhance the model's capacity to learn longer window lengths.

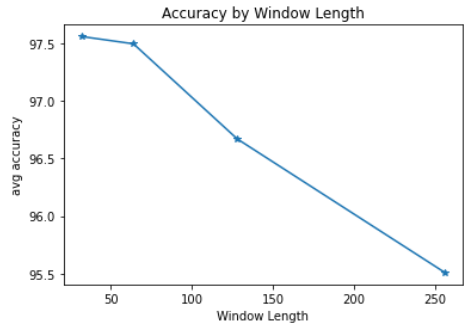


Figure 13. Performance by Window Length

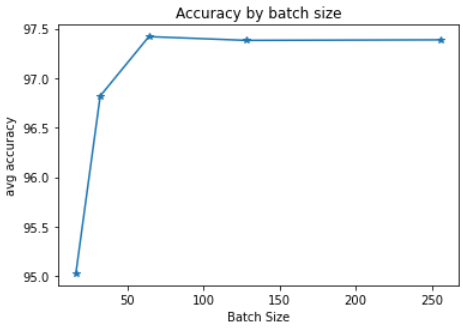


Figure 14. Performance by Batch Size

Instead, we investigated the interaction between the two parameters (Figure 13) and discovered that window length of 32 and any batch size between 64, 128 and 512 are good for the proposed model(Figures 13 & 14). We chose the smallest of the three i.e., 64 for the subsequent experiments following similar work of [62] suggesting that smaller batch sizes help improve accuracy and stable convergence(Figure 15). Although batch size 64 slowed down the learning process significantly, it does, in the final stages, converge to a more stable model characterized by lower variance in generalisation accuracy.

Table 3. Average % Accuracy per batch size for window length 32

Batch Size=16: 94.197% (+/0.908)
Batch Size=32: 97.625% (+/0.372)
Batch Size=64: 98.518% (+/0.216)
Batch Size=128: 98.767% (+/0.171)
Batch Size=256: 98.684% (+/0.175)

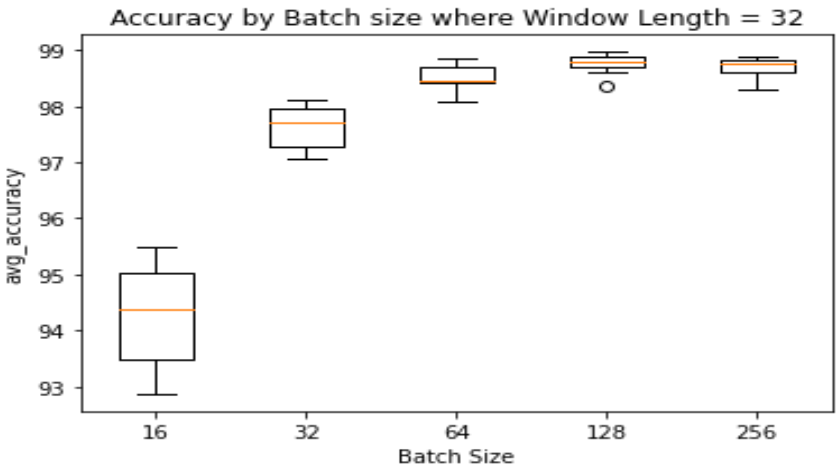


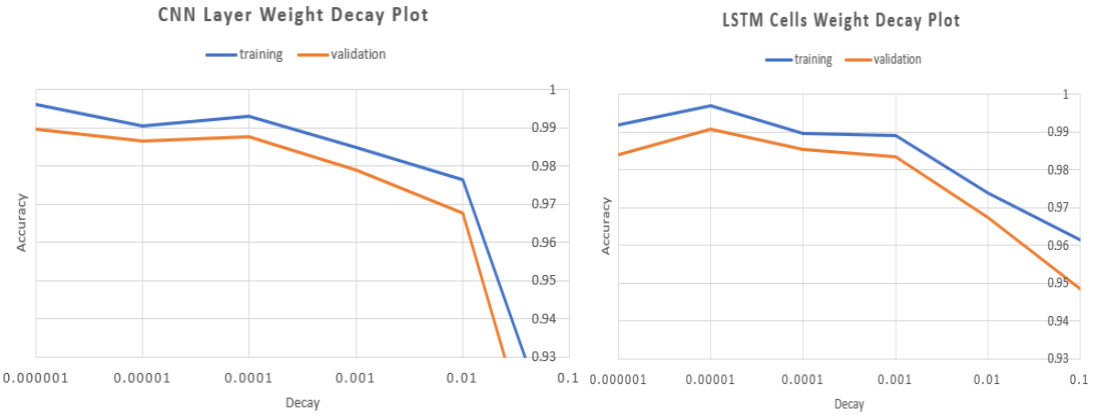
Figure 15. Box plot of different batch sizes for window length 32

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

4.6.3 Tuning for overfitting

a. Weight Decay

As part of measures to prevent overfitting, we benchmark model performance against weight decay ranging between $1e^{-6}$ and $1e^{-1}$. Setting the model's weight decay at $1e^{-6}$ and $1e^{-5}$ for the CNN and the LSTM respectively results in sub-optimal performance for the model (Figure 16).

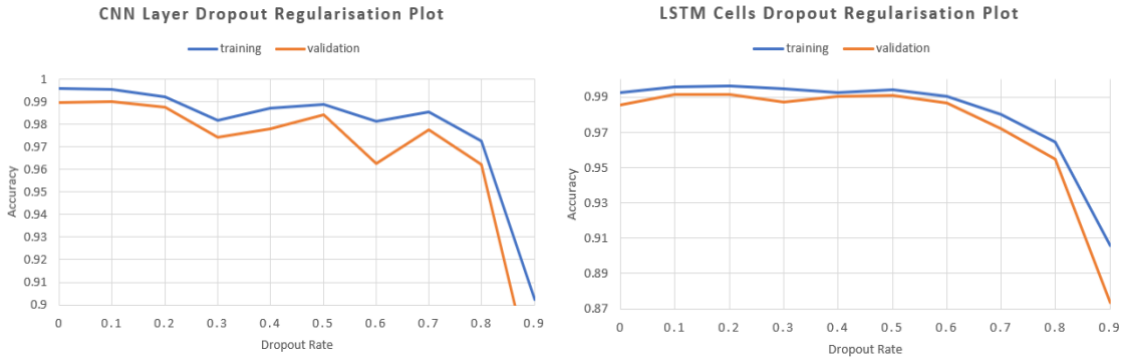


(a) (b)

Figure 16. Line plot showing Accuracy over the weight decay between $1e^{-1}$ and $1e^{-6}$ for **(a)** the CNN layer and **(b)** the LSTM cells.

b. Dropout Regularization

The line plots in Figure 17 highlight the learning performance of the model as we grid-search for dropout rate for the two main sub-models i.e. the CNN layer and the LSTM networks. We can see that performance continues to decline as we increase dropout for the CNN layer, likewise the LSTM networks. We, therefore, set the CNN dropout rate at 0.1 and the LSTM to none since the LSTM cells appear to perform optimally without dropout.



(a) (b)

Figure 17. CNN-LSTM Model Dropout regularization plots. (**a**) shows the CNN layer accuracy for training and validation over dropout rate between 0 and 0.9 (**b**) shows the LSTM cells accuracy for training and validation over dropout rate between 0 and 0.9

c. Number of epochs

Using the Keras early stopping of training via a call-back, we specify the validation loss as the performance metric to monitor in order to end training. We conducted several experiments and was able to gauge the range of epochs required for the model to prevent overfitting. Consequently, we set the number of epochs to 100 with a patience of 8 to add a delay before stopping. Finally, we added a

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

second call-back called ModelCheckpoint to save the best model achieved during the entire training process.

d. Tuning for Learning Rate

The learning rate plays a substantial role in the model's overall learning capability. Therefore, it is important to thoroughly investigate the best learning rate for the baseline model. Instead of choosing a fixed learning rate, we evaluated the learning rate between 10^{-8} and 10^{-1} using the learning rate scheduler. We can see from Figure 18 that the model records its lowest learning loss at the 10^{-3} mark. While it is recommended that the learning rate should be re-optimised at the end of all other hyper-parameter tuning [62] the sub-optimal learning rate remained the same before and after the hyper-parameter tuning. Consequently, we set the learning rate at 10^{-3} (same as 1e-4 if you prefer the exponential).

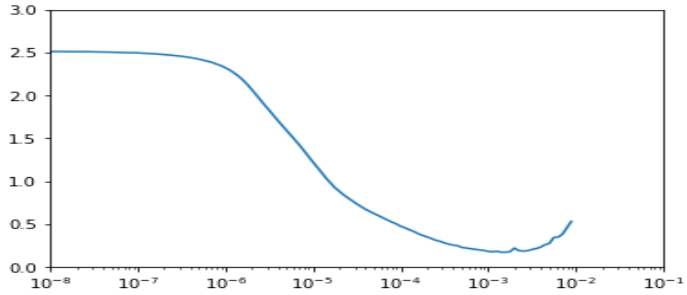


Figure 18. Line plot of loss over learning rate

As soon as we conclude the tuning for overfitting, we are left with the baseline model which doubles as the *Non-contextual* model. Table 4 below shows the optimised parameter values. In section 5 , we discuss the results of the two models i.e. *Non-contextual* and *Contextual* models.

Table 4. Training Parameters (tuned and default) used for baseline model

Parameter	Recommended	Feasible	Tuned
Conv1D layers	2		No
ConvID - Kernel Size	3	2	Yes
ConvID - Filter Size	64	256	Yes
LSTM Cells	100		No
Optimiser	Adam		Yes
Dropout	CNN=0.1		Yes
Weight Decay	CNN=$1e^{-6}$, LSTM=$1e^{-5}$		Yes
Window Length (size of input vector)	32	64	Yes
Number of input channels	9		
Batch Size	64	32	Yes
Subsequence steps	4		No
Learning Rate	$1e^{-4}$	$3e^{-4}$	Yes
Early Stopping	Patience: 10 epoch: 50-100	Patience: 8 epoch: 40-80	Yes
ModelCheckpoint	save_best_only: True		No

5. Result and Discussion

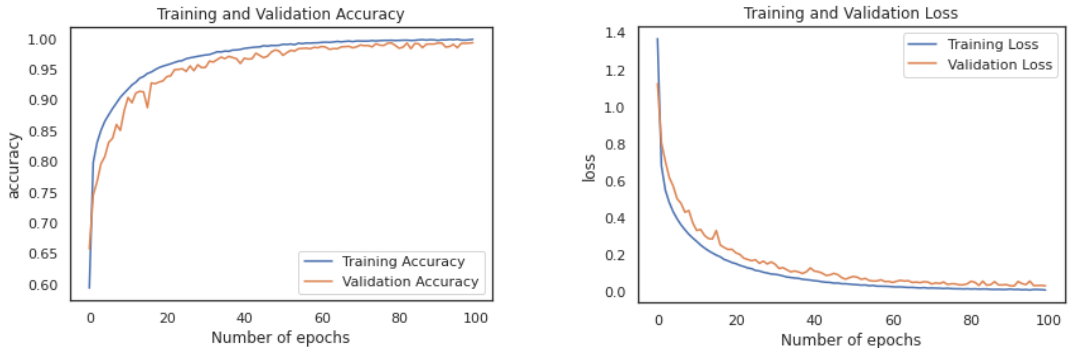
In the previous section, we analysed the impact of several different hyperparameters on the performance of the model. We also carried out a series of tuning to prevent the model from

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

overfitting, which finally led us to the baseline model. The baseline model which doubles as the ***Non-contextual*** model was trained using the traditional inertial sensing signals from 3 axes each of accelerometer, gyroscope, and magnetometer sensors, altogether making 9 channels of signals. In the second experiment, while keeping all parameters at optimal settings (as shown in Table 4), we trained another model using a combination of traditional inertial sensing signals and additional data from the ambient sensors, resulting in 11 channels of signals (2 additional from audio and light sensors). We tagged the model from the second experiment ***Contextual*** simply because it uses the ambient sensing as rich contextual information to create a context-aware model. In this section, using the hold-out test set, we analyse results from these models by measuring the significance of performance differences and investigate how accurate the models are in generalising simple and complex activities.

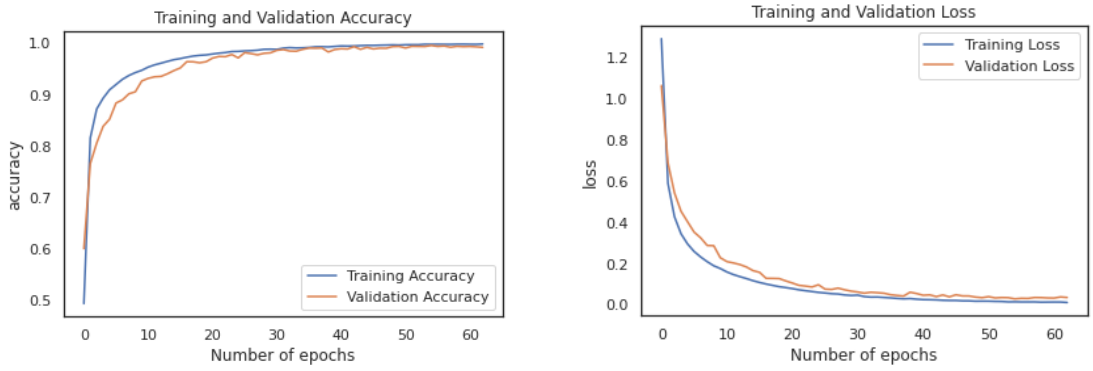
5.1 Analysis of training and validation data

The ***Non-contextual*** model started converging after about 60 epochs as indicated in the accuracy and loss plots in Figure 19. Compared to the ***Non-contextual***, the ***Contextual*** model (Figure 20) converged much earlier at about 30 epochs. Although it took the ***Contextual*** about 161 seconds to be interrupted by the early-stopping call-back, the ***Non-contextual*** model ran for another 100 seconds more before being interrupted. This is, however, surprising given that the ***Contextual model*** is exposed to more input signals and is expected to take longer time to train. This suggests to us that the ***Contextual*** model was faster in discovering the temporal relatedness between the signals using the additional ambient signals. Although given the stochastic nature of deep learning algorithms, convergence as a function of number of epochs sometimes varies[24].



(a) (b)

Figure 19. Line graph showing (a) accuracy plot and (b) loss plot as a function of number of epochs for the *Non-contextual* Model

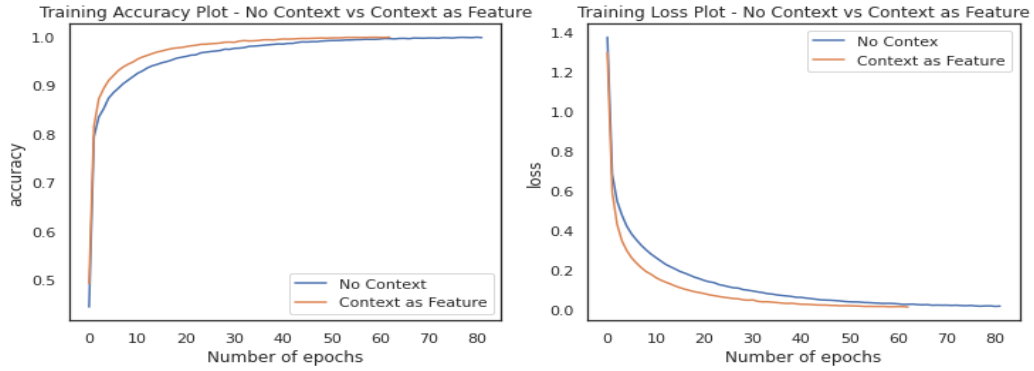


(c) (d)

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

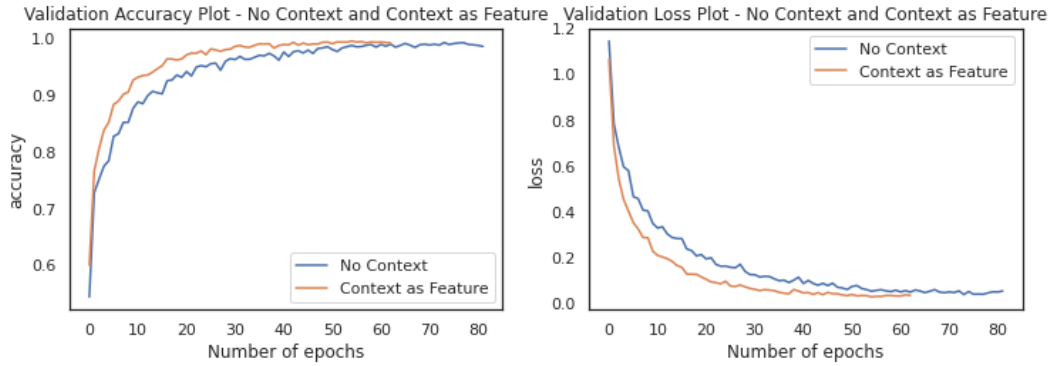
Figure 20. Line graph showing (c) accuracy plot and (d) loss plot as a function of number of epochs for the *Contextual* Model

Moreover, the training accuracy and loss plots (Figure 21) show how the two models compare during training. It is obvious that the *Contextual* model, for the most part of the training process, continues to outperform the *Non-contextual* model. Although the two models seemed to match one another in performance towards the tail end, at point in time the *Non-contextual* model trained better. In Figure 22, we compare the two models' performance with the validation data. Once more, the *Contextual* model quickly edges out the *Non-contextual* model in the first 10 epochs and even throughout the validation process suggesting that the additional input signals from the ambient sensors might be responsible. The line plot however appears wavy due to some noise in the input data. While neural networks are known to be robust to noise, model accuracy often improves when time series data are pre-processed for noise reduction and smoothing using median filters [32].



(a) (b)

Figure 21. Line plot comparing (a) Training Accuracy and (b) Training Loss between the two models during training



(a) (b)

Figure 22. Line plot comparing (a) Validation Accuracy and (b) Validation Loss between the two models during training

5.2 Recognition accuracy for simple and complex activities

The confusion matrices in Figures 23 and 24 give insight into the classification errors being made by the *Non-contextual* and *Contextual* models respectively. This result is based on the performance of the models' classifier after being evaluated on the hold-out test dataset that was never used during the training and validation phases. Table 6 provides the percentage accuracy by class using the macro average f-score. It is not surprising to see the weighted average or the micro average having higher percentages, this is often due to the skewness in the dataset. As earlier noted in section 4.2, the macro-average percentage ensures that,

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

despite the imbalanced dataset, each class has equal representation in the overall accuracy percentage however little the class samples. Overall, the *Contextual* model achieved 98.72% accuracy while the *Non-contextual* model trails behind at 96.19% indicating that the rich context data provided by the ambient sensors are instrumental to the improved performance. This results in about 2.62% overall model improvement by just increasing the sensor modalities.

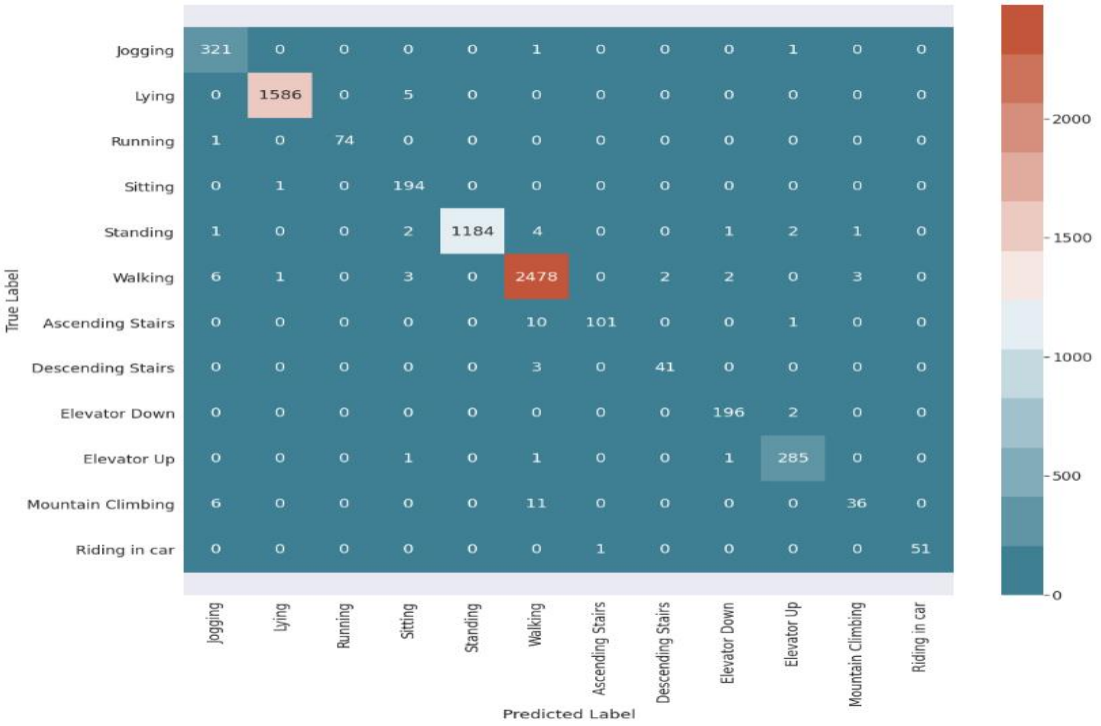


Figure 23. Confusion Matrix for the *Non-contextual* model

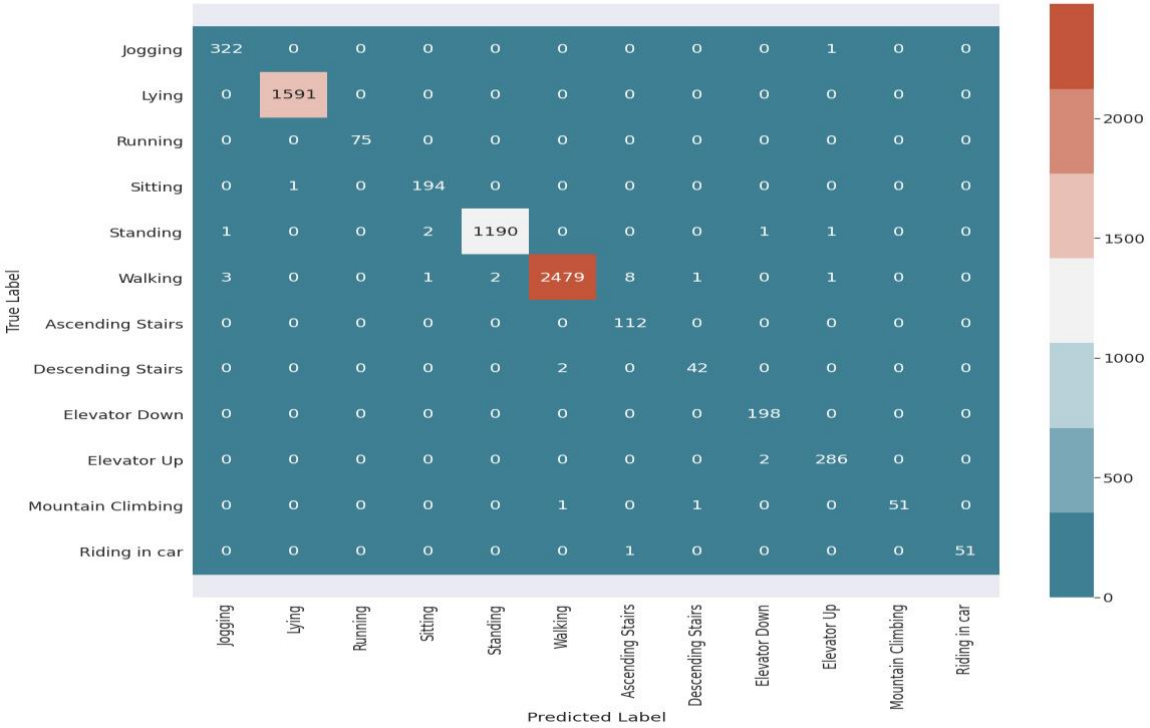


Figure 24. Confusion Matrix for the Contextual model

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

In Table 6, we also evaluated the performance based on simple and complex activities following the activity category table in section 4.4. The *Non-contextual* model achieved an outstanding performance of 98.7% on simple activities and a modest 93.7% on complex activities. However, the *Contextual* model was able to achieve a state-of-the-art performance in both simple and complex activities recording average performance of 99.6% and 97.8 respectively. Both models were able to distinguish between several different simple activities effectively. Although the two models recorded their individual lowest accuracies for simple activities in the Jogging and Sitting labels, the *Contextual* model still manages to record at least 99% in these two classes (Table 7). In the complex activity category, the Non-contextual *model* performed woefully generalising the Mountain Climbing activity with a 77.4% accuracy compared to 98.1% by the *Contextual* model. Interestingly, the two models performed equally well in recognising Driving activity. Nevertheless, the *Contextual* model did not perform less in any activity class under the simple and complex activity categories. Instead, it continues to prove that a model trained with rich contextual data from ambient sensors can consistently outperform the same model trained with only traditional inertia signals.

To answer the research question regarding the improvement observed, the *Contextual* model, going by Table 6, has demonstrated about 0.91% and 4.38% improvement on simple and complex activities respectively compared to the *Non-contextual* model.

Table 5. Overall Model Accuracy

Model	Accuracy
Non-contextual	96.19%
Contextual	98.72%

Table 6. Model Accuracy by Activity Category

	Activity category	
Model	Simple	Complex
Non-contextual	98.7%	93.7%
Contextual	99.6%	97.8%

Table 7. Model Accuracy by Activity Class

		F-Score	
#	Activity	*Non-contextual*	*Contextual*
1	Jogging	0.976	0.992
2	Lying	0.998	1.000
3	Running	0.993	1.000
4	Sitting	0.970	0.990
5	Standing	0.995	0.997
6	Walking	0.991	0.996
7	Ascending Stairs	0.944	0.961
8	Descending Stairs	0.943	0.955
9	Elevator Down	0.985	0.992
10	Elevator Up	0.984	0.991
11	Mountain Climbing	0.774	0.981
12	Riding in car	0.990	0.990
	accuracy	0.989	0.995
	macro avg	0.962	0.987
	weighted avg	0.989	0.995

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

To test the hypothesis that combining inertial and ambient sensing signals would produce a better performance for the hybrid model, we measure the significance of performance differences using a pairwise t-test at 5% significance level. For us to have adequate samples to carry out the hypothesis tests, we trained and evaluated the 2 models repeatedly for about 20 times. Each time, we noted the macro-average F-score for both models. Given the stochastic nature of deep learning algorithms, the specific results fluctuated for both models after each round of training. Nevertheless, in the end, we obtained an extremely small p-value which prompted us to reject the null hypothesis that the *Non-contextual* model is at par with the *Contextual* model in terms of average performance. In other words, with a 95% confidence interval, the *Non-contextual* model's mean performance F-score is significantly lower than that of the *Contextual* model. As a result, we conclude that the average generalisation ability of the *Contextual* model is superior to that of the *Non-contextual* model. Given that both models share the same architecture, matching parameters, and training environment, we can only attribute this improvement to the main difference between the two models which is, without doubt, the additional signal inputs. The ambient signals have proved to be the distinguishing factor in this study. Not only that, the *Contextual* model, despite the additional sensor modalities, was able to achieve convergence well before the *Non-contextual* model.

5.3 Interclass Similarity Analysis

we provide in this section experimental evidence that details performance results with respect to the prevalent interclass similarity issues between simple and complex activities. Mindful of the fact that it is not uncommon for models to experience poor classification capability in the presence of highly correlated input data [31], we once again, using Table 1 in section 4.4 as a guide, compare how the two models perform with activities that appear similar in nature. The *Non-contextual* model, for example, recorded the highest classification error with the Mountain Climbing activity. A quick look at the confusion matrix (Figure 23) for the *Non-contextual* model reveals that the model confuses Mountain Climbing activity with Walking at least once every three tries. The model also had some difficulties with Ascending and Descending Stairs, which it often confuses with Walking. This is not surprising since Walking is obviously a sub-activity of these other complex activities (Table 1). In the case of the *Contextual* model, there was a slight improvement on the classification error regarding the Ascending and Descending stairs being confused with Walking. With about 27% improvement compared to the *Non-contextual* model, the *Contextual* model recorded its single biggest generalisability improvement in the Mountain Climbing activity. Once again, the ambient signals have helped the model obviate inter-class similarity confusion during classification. Other suspected inter-class similar activities such as Standing, Elevator Up and Down, Driving and Sitting, do not seem to pose any challenge to either of the models. Following the recommendation from [32] to combine multiple sensor modalities was instrumental to addressing inter-class similarity challenge. Taken together, the indicate that models trained with both inertial and ambient data consistently outperforms models trained with only inertial data. in the next section we discuss the conclusion and recommendations for future work.

6. Conclusion and Future Work

In this article, we investigated the benefits of combining environment sensing as contextual data with traditional inertial sensing data to distinguish between simple and complex human activities of daily living. We designed a context-aware hybrid deep learning model that can

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 28 March 2022 doi:10.20944/preprints202203.0363.v1

distinguish between inter-similar class labels by adding rich contextual signals. To the best of the knowledge, this is the first study to have focused on utilising raw sensing data collected exclusively from smartphone sensors to generalise simple and complex human activities. The inertial sensors used in data collection include the accelerometer, gyroscope, and magnetometer. The environment sensors that provide the rich contextual information comprise the audio and light sensors.

The main goal was to demonstrate that, apart from the inertial sensing data, signals from the smartphone audio and light sensors, which represent environmental noise level and illumination respectively can serve as rich contextual information for determining fine-grained complex activities with high recognition accuracy. We proposed and designed two CNN-LSTM hybrid models: ***Non -Contextual*** and ***Contextual models***. The *Non-contextual model* represents the baseline model while the *Contextual* one is the improved model. Extensive experiments were conducted to benchmark the recognition capabilities of the two models.

First, using the grid-search approach, we executed a series of experiments tuning several different hyperparameters of the baseline model to ensure optimal performance and save it from overfitting. These initial experiments informed the decision to set the models' hyperparameters as summarised in Table 4. Next, we performed another round of experiments to benchmark the performance of the baseline model using only raw signals from the inertial sensors. Using the Keras API call-backs – early-stopping and model checkpointing – we were able to save the best performing model achieved during training to file. We then referred to this model as the *Non-contextual* model since it was trained with just inertial signals.

Expectedly, the *Non-contextual* model recorded state-of-the-art results generalising simple activities, achieving more than 99% F-score in 4 out of 6 simple activities with an average overall F-score of 98.7%. The *Non-contextual* model has a significant 2.77% improvement in generalisation accuracy compared to a similar work by [65] where they similarly to investigate the effect of imbalanced dataset on model classification accuracy. Their model architecture did not combine LSTM with CNN like we did, it only managed to achieve about 93.6% accuracy. The inclusion of LSTM networks did improve the performance of the model substantially. However, the results were not as high for complex activities with an average overall F-score of 93.7%. The *Non-contextual* model confused a few complex activities for simple activities, for instance, the model confused Walking for Ascending Stairs or Mountain Climbing. These three activities suffer from inter-class similarity [32]. As a result, this mix-up is understandable as both activities appear similar in a way. We described the inter-class similar activities in more detail in Table 1.

Subsequently, we trained a second model, in another set of experiments, using a combination of raw inertial and environment sensing signals. We introduced the environment signals to provide rich contextual information for the new model and then tagged it ***Contextual***. Experimental results confirmed that combining raw inertial and ambient signals from a single smartphone resulted in an improved model compared to the *Non-contextual* model. For simple and complex activities, the *Contextual* model achieved average overall F-scores of 99.6% and 97.8% respectively. This is a substantial 2.6% improvement in combined overall recognition capability compared to the *Non-contextual* model. Specifically, the *Contextual* model was able to effectively distinguish between Walking and Ascending Stairs or Mountain Climbing activities, overcoming the initial weak point of the *Non-contextual* model to inter-class

similarity issues. Moreover, a pairwise t-test statistics to compare the performance difference between the two models proved significant, further confirming that the generalisation ability of HAR models improves significantly when trained with a dataset containing contextual information collected from environment sensors.

It is also important to mention that the *Contextual* model is a CNN-LSTM hybrid model, which automatically extracts features from raw sensor signals using the CNN layers, and the LSTM layers for temporal modelling before class predictions are made. This further demonstrates the dominance of deep learning models over classical algorithms that rely heavily on heuristic hand-crafted features [11]. Interestingly, it also justified the applicability of the CNN-LSTM hybrid model to achieve state-of-the-art results with time series data, more importantly with activity context recognition [14, 51, 53, 61, 70] and speech recognition problems [22].

Finally, in the future, we plan to deploy the model on a mobile phone to evaluate how effective it is with helping people track and collect Time-Use Data (TUD). We also plan to tweak the architecture of the CNN-LSTM hybrid model to include bi-directional LSTM stacks, multi-headed CNNs and perhaps a deep multi-tasking learning model to improve the overall recognition accuracy for complex activities. This will require, but not limited to, expanding the dataset by collecting a multi-label classification dataset using the smartphone exclusively. In this dataset, we will have the opportunity to annotate several simple activities that make up a specific complex activity e.g. Sitting, Eating and Drinking to represent a complex activity like dining with friends. Given the lack of rich contextual datasets collected entirely over the smartphone sensors, we also plan to collect other wide-ranging environment sensing signals. We intend to collect more data on activities with significantly low sample size to create a dataset with balanced class labels. All this will allow the opportunity to test several different combinations of rich contextual data for developing a fine-grain complex activity recognition system in the nearest future.

References

1. Alawneh, L., Al-Ayyoub, M., Al-Sharif, Z.A. et al. Personalized human activity recognition using deep learning and edge-cloud architecture. J Ambient Intell Human Comput,202. https://doi.org/10.1007/s12652-022-03752-w
2. Chen, K.; Zhang, D.; Yao, L.; Guo, B.; Yu, Z.; Liu, Y. Deep Learning for Sensor-based Human Activity Recognition: Overview, Challenges, and Opportunities. ACM Comput. Surv. (CSUR) 2022, 54, 1–40.
3. Jalal, A., Kim, J. T., & Kim, T. Development of a life logging system via depth imaging-based human activity recognition for smart homes. Paper presented at the Proceedings of the International Symposium on Sustainable Healthy Buildings, Seoul, Korea, 2012 , 19
4. Foubert, N., McKee, A. M., Goubran, R. A., & Knoefel, F. Lying and sitting posture recognition and transition detection using a pressure sensor array. Paper presented at the 2012 IEEE International Symposium on Medical Measurements and Applications Proceedings, 2012, 1-6.
5. Gu, F. ; Chung, M.; Chignell, M.; Valaee, S.;Zhou, B.; Liu, X.; A Survey of on Deep Learning for Human Activity Recognition. ACM Comput. Surv. (CSUR) 2021, 54, No.8, Article 177, 1–40.
6. De Pessemier, T., Dooms, S., & Martens, L. Context-aware recommendations through context and activity recognition in a mobile environment. Multimedia Tools and Applications, 2014,72(3), 2925-2948.
7. Otebolaku, A., & Andrade, M. Context-aware media recommendations for smart devices. Journal of Ambient Intelligence and Humanized Computing, 2015, 6(1), 13-36
8. Grzeszick, R., Lenk, J. M., Rueda, F. M., Fink, G. A., Feldhorst, S., & ten Hompel, M. Deep neural network based human activity recognition for the order picking process. Paper presented at the Proceedings of the 4th International Workshop on Sensor-Based Activity Recognition and Interaction, 2017, 1-6.
9. Lu, D., Nguyen, D., Nguyen, T., & Nguyen, H. Vehicle mode and driving activity detection based on analyzing sensor data of smartphones. Sensors, 2018, 18(4), 1036.

10. Bulling, A., Blanke, U., & Schiele, B. A tutorial on human activity recognition using body-worn inertial sensors. ACM Computing Surveys (CSUR),2014, 46(3), 1-33.
11. Wang, J., Chen, Y., Hao, S., Peng, X., & Hu, L. (2019). Deep learning for sensor-based activity recognition: A survey. Pattern Recognition Letters, 119, 3-11.
12. Hasan, M., & Roy-Chowdhury, A. Context aware active learning of activity recognition models IEEE, 2015, doi:10.1109/ICCV.2015.516
13. Liu, Y., Nie, L., Han, L., Zhang, L., & Rosenblum, D. S. Action2Activity: Recognizing complex activities from sensor data. Paper presented at the Twenty-Fifth International Joint Conference on Artificial Intelligence, July 2015, pp 1617-1623
14. Peng, L., Chen, L., Ye, Z., & Zhang, Y. Aroma: A deep multi-task learning based simple and complex human activity recognition method using wearable sensors. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2018, 2(2), 1-16.
15. Cheng, W., Erfani, S. M., Zhang, R., & Kotagiri, R. Predicting complex activities from ongoing multivariate time series. Paper presented at the Ijcai, 2018, 3322-3328.
16. Vepakomma, P., De, D., Das, S. K., & Bhansali, S. A-wristocracy: Deep learning on wrist-worn sensing for recognition of user complex activities. Paper presented at the 2015 IEEE 12th International Conference on Wearable and Implantable Body Sensor Networks (BSN), 2015, 1-6.
17. Walse, K. H., Dharaskar, R. V., & Thakare, V. M. PCA-based optimal ann classifiers for human activity recognition using mobile sensors data. Paper presented at the Proceedings of First International Conference on Information and Communication Technology for Intelligent Systems, 2016.1, 429-436.
18. Brownlee, J. (2018). Deep learning for time series forecasting: Predict the future with MLPs, CNNs and LSTMs in python Machine Learning Mastery.
19. Yang, J., Nguyen, M. N., San, P. P., Li, X. L., & Krishnaswamy, S. Deep convolutional neural networks on multichannel time series for human activity recognition. Paper presented at the Twenty-Fifth International Joint Conference on Artificial Intelligence, 2015.
20. Irfan, S., Anjum, N., Masood, N., Khattak, A. S., & Ramzan, N. A Novel Hybrid Deep Learning Model for Human Activity Recognition Based on Transitional Activities. Sensors (Basel, Switzerland), 2021, 21(24), 8227. https://doi.org/10.3390/s21248227
21. Khan, I.U., Afzal. S., Lee, J.W. Human Activity Recognition via Hybrid Deep Learning Based Model. Sensors. 2022, 22(1):323. https://doi.org/10.3390/s22010323.
22. Sainath, T. N., Vinyals, O., Senior, A., & Sak, H. Convolutional, long short-term memory fully connected deep neural networks. Paper presented at the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015,4580-4584.
23. Aggarwal, J. K., & Ryoo, M. S. Human activity analysis: A review. ACM Computing Surveys (CSUR), 2011, 43(3), 1-43.
24. Vrigkas, M., Nikou, C., & Kakadiaris, I. A. A review of human activity recognition methods. Frontiers in Robotics and AI, 2015,2, 28.
25. Turaga, P., Chellappa, R., Subrahmanian, V. S., & Udrea, O. Machine recognition of human activities: A survey. IEEE Transactions on Circuits and Systems for Video Technology, 2008, 18(11), 1473-1488.
26. Dernbach, S., Das, B., Krishnan, N. C., Thomas, B. L., & Cook, D. J. Simple and complex activity recognition through smart phones. Paper presented at the 2012 Eighth International Conference on Intelligent Environments, 2012, 214-221.
27. Blanke, U., Schiele, B., Kreil, M., Lukowicz, P., Sick, B., & Gruber, T. All for one or one for all? combining heterogeneous features for activity spotting. Paper presented at the 2010 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012, 18-24.
28. Vaizman, Y., Ellis, K., & Lanckriet, G. Recognizing detailed human context in the wild from smartphones and smartwatches. IEEE Pervasive Computing, 2017, 16(4), 62-74.
29. Cruciani, F., Vafeiadis, A., Nugent, C., Cleland, I., McCullagh, P., Votis, K., Hamzaoui, R. Feature learning for human activity recognition using convolutional neural networks. CCF Transactions on Pervasive Computing and Interaction, 2020, 2(1), 18-32.
30. Schrader, L., Toro, A. V., Konietzny, S., Rüping, S., Schäpers, B., Steinböck, M., . . . Bock, T. Advanced sensing and human activity recognition in early intervention and rehabilitation of elderly people. Journal of Population Ageing, 2020, 1-27.
31. Akila, K., & Chitrakala, S. Highly refined human action recognition model to handle intraclass variability & interclass similarity. Multimedia Tools and Applications, 2019, 78(15), 20877-20894.

32. Bharti, P., De, D., Chellappan, S., & Das, S. K. HuMAn: Complex activity recognition with multi-modal multi-positional body sensing. IEEE Transactions on Mobile Computing, 2018, 18(4), 857-870.
33. Shoaib, M., Scholten, H., & Havinga, P. J. (2013). Towards physical activity recognition using smartphone sensors. Paper presented at the 2013 IEEE 10th International Conference on Ubiquitous Intelligence and Computing and 2013 IEEE 10th International Conference on Autonomic and Trusted Computing, 2013, 80-87.
34. Shoaib, M., Bosch, S., Incel, O. D., Scholten, H., & Havinga, P. J. Fusion of smartphone motion sensors for physical activity recognition. Sensors, 2014, 14(6), 10146-10176.
35. Chen, G., Ding, X., Huang, K., Ye, X., & Zhang, C. Changing health behaviours through social and physical context awareness. Paper presented at the 2015 International Conference on Computing, Networking and Communications (ICNC), 2015, 663-667.
36. Ramos-Garcia, R. I., & Hoover, A. W. A study of temporal action sequencing during consumption of a meal. Paper presented at the Proceedings of the International Conference on Bioinformatics, Computational Biology and Biomedical Informatics, 2013, 68-75.
37. Scholl, P. M., & Van Laerhoven, K. A feasibility study of wrist-worn accelerometer-based detection of smoking habits. Paper presented at the 2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, 2012, 886-891.
38. Parate, A., Chiu, M., Chadowitz, C., Ganesan, D., & Kalogerakis, E. Risq: Recognizing smoking gestures with inertial sensors on a wristband. Paper presented at the Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services, 2014,149-161.
39. Ha, S., Yun, J., & Choi, S. Multi-modal convolutional neural networks for activity recognition. Paper presented at the 2015 IEEE International Conference on Systems, Man, and Cybernetics, 2015, 3017-3022.
40. Sousa Lima, W., Souto, E., El-Khatib, K., Jalali, R., & Gama, J. Human activity recognition using inertial sensors in a smartphone: An overview. Sensors, 2019, 19(14), 3213.
41. Ignatov, A. Real-time human activity recognition from accelerometer data using convolutional neural networks. Applied Soft Computing Journal,2018, 62, 915-922. doi:10.1016/j.asoc.2017.09.027.
42. Otebolaku, A., & Andrade, M. User context recognition using smartphone sensors and classification models. Journal of Network and Computer Applications, 2016, 66, 33-51.
43. Das, B., Seelye, A. M., Thomas, B. L., Cook, D. J., Holder, L. B., & Schmitter-Edgecombe, M. Using smart phones for context-aware prompting in smart environments. Paper presented at the 2012 IEEE Consumer Communications and Networking Conference (CCNC), 2012, 399-403.
44. Vaizman, Y., Weibel, N., & Lanckriet, G. Context recognition in-the-wild: Unified model for multi-modal sensors and multi-label classification. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies,2018, 1(4), 1-22.
45. Edel, M., & Köppe, E. Binarized-blstm-rnn based human activity recognition. Paper presented at the 2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2016, 1-7.
46. Inoue, M., Inoue, S., & Nishida, T. Deep recurrent neural network for mobile human activity recognition with high throughput. Artificial Life and Robotics, 2018, 23(2), 173-185.
47. Hammerla, Nils Y., et al. Deep, convolutional, and recurrent models for human activity recognition using wearables. Proceedings of the 25th International Joint Conference on Artificial Intelligence, 2016.
48. Guan, Y., & Plötz, T. Ensembles of deep lstm learners for activity recognition using wearables. proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2017, 1(2), 1-28.
49. Murad, A., & Pyun, J. Deep recurrent neural networks for human activity recognition. Sensors, 2017, 17(11), 2556.
50. Malhotra, P., Vig, L., Shroff, G., & Agarwal, P. Long short-term memory networks for anomaly detection in time series. Paper presented at the Proceedings, 2015, 89 89-94.
51. Ordóñez, F. J., & Roggen, D. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. Sensors, 2016, 16(1), 115.
52. Zhao, Y., , R., Chevalier, G., Xu, X., & Zhang, Z. Deep residual bidir-LSTM for human activity recognition using wearable sensors. Mathematical Problems in Engineering, 2018
53. Chen, L., Zhang, Y., & Peng, L. METIER: A deep multi-task learning based activity and user recognition model using wearable sensors. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2020, 4(1), 1-18.
54. Agarwal, P., & Alam, M. A lightweight deep learning model for human activity recognition on edge devices. Procedia Computer Science, 2020, 167, 2364-2373.
55. Almaslukh, B., Abdel, M. A., & Jalal Al-Muhtadi. A robust deep learning approach for position-independent smartphone-based human activity recognition. Sensors, 2018, 18(11), 3726. doi:10.3390/s18113726

56. Bragança, H., Colonna, J., & Souto, E. A smartphone lightweight method for human activity recognition based on information theory. Sensors, 2020, 20(7), 1856. doi:10.3390/s20071856
57. Gani, M. O., Fayezeen, T., Povinelli, R. J., Smith, R. O., Arif, M., Kattan, A. J., & Ahamed, S. I. A light weight smartphone based human activity recognition system with high accuracy. Journal of Network and Computer Applications, 141, 59-72. doi:10.1016/j.jnca.2019.05.001
58. Hassan, M. M., Uddin, M. Z., Mohamed, A., & Almogren, A. A robust human activity recognition system using smartphone sensors and deep learning. Future Generation Computer Systems, 2018, 81, 307-313. doi:10.1016/j.future.2017.11.029.
59. Otebolaku, A., Enamamu, T., Alfouldi, A., Ikpehai, A., & Marchang, J. Deep sensing: Inertia and ambient sensing for activity context recognition using deep convolutional neural networks. Sensors 2020, 20, 3803: https://doi.org/10.3390/s20133803
60. Laguna, J. O., Olaya, A. G., & Borrajo, D. A dynamic sliding window approach for activity recognition. Paper presented at the International Conference on User Modeling, Adaptation, and Personalization, 2011, 219-230.
61. Chambers, R. D., & Yoder, N. C. FilterNet: A many-to-many deep learning architecture for time series classification. Sensors, 2020, 20(9), 2498.
62. Masters, D., & Luschi, C. Revisiting small batch training for deep neural networks. arXiv Preprint, 2018, arXiv:1804.07612.
63. Chollet, F. Xception: Deep learning with depthwise separable convolutions. Paper presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, 1251-1258.
64. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 2014, 15(1), 1929-1958.
65. Otebolaku, A., & Andrade, M. Recognizing high-level contexts from smartphone built-in sensors for mobile media content recommendation. Paper presented at the 2013 IEEE 14th International Conference on Mobile Data Management, 2013, 2 142-147.
66. Bettini,C., Gabriele Civitarese,G., Presotto, R.CAVIAR: Context-driven active and incremental activity recognition, Knowl.-Based Sys, 2020, 196, 105816.
67. Weiss, G. M., Yoneda, K., & Hayajneh, T. Smartphone and smartwatch-based biometrics using activities of daily living IEEE, 2019. doi:10.1109/ACCESS.2019.2940729
68. Carneiro, T., Medeiros Da Nobrega, Raul Victor, Nepomuceno, T., Bian, G., De Albuquerque, V. H.,C., & Filho, P. P. R. Performance analysis of google colaboratory as a tool for accelerating deep learning applications. IEEE Access, 2018, 6, 61677-61685. doi:10.1109/ACCESS.2018.2874767
69. Ward, J. A., Lukowicz, P., & Gellersen, H. W. Performance metrics for activity recognition. ACM Transactions on Intelligent Systems and Technology (TIST), 2011, 2(1), 1-23.
70 Thakur, D., Biswas, S., Ho, E.S.L., and Chattopadhyay,S. ConvAE-LSTM: Convolutional Autoencoder Long Short-Term Memory Network for Smartphone-Based Human Activity Recognition, in IEEE Access, 2022,vol. 10, pp. 4137-4156, 2022, doi: 10.1109/ACCESS.2022.3140373.