Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 9356165, 21 pages
https://doi.org/10.1155/2020/9356165

Research Article

A Machine-Learning Based Nonintrusive Smart Home Appliance Status Recognition

Liston Matindife,[1] Yanxia Sun,[1] and Zenghui Wang[2]

[1]*Department of Electrical and Electronic Engineering Science, University of Johannesburg, Auckland Park 2006, South Africa*
[2]*Department of Electrical and Mining Engineering, University of South Africa, Florida 1710, South Africa*

Correspondence should be addressed to Zenghui Wang; wangzengh@gmail.com

Received 18 June 2020; Revised 1 September 2020; Accepted 7 September 2020; Published 18 September 2020

Academic Editor: Qian Zhang

In a smart home, the nonintrusive load monitoring recognition scheme normally achieves high appliance recognition performance in the case where the appliance signals have widely varying power levels and signature characteristics. However, it becomes more difficult to recognize appliances with equal or very close power specifications, often with almost identical signature characteristics. In literature, complex methods based on transient event detection and multiple classifiers that operate on different hand crafted features of the signal have been proposed to tackle this issue. In this paper, we propose a deep learning approach that dispenses with the complex transient event detection and hand crafting of signal features to provide high performance recognition of close tolerance appliances. The appliance classification is premised on the deep multilayer perceptron having three appliance signal parameters as input to increase the number of trainable samples and hence accuracy. In the case where we have limited data, we implement a transfer learning-based appliance classification strategy. With the view of obtaining an appropriate high performing disaggregation deep learning network for the said problem, we explore individually three deep learning disaggregation algorithms based on the multiple parallel structure convolutional neural networks, the recurrent neural network with parallel dense layers for a shared input, and the hybrid convolutional recurrent neural network. We disaggregate a total of three signal parameters per appliance in each case. To evaluate the performance of the proposed method, some simulations and comparisons have been carried out, and the results show that the proposed method can achieve promising performance.

1. Introduction

1.1. Background and Motivations. It is now common today to remotely monitor and control various appliances in the smart-home [1, 2]. The monitoring system is often integrated into the Internet of Things (IoTs). In addition to standalone appliances, the smart home is composed of security, air-conditioning personalised medical equipment, and plug-in-electrical-vehicles (PEVs) [3, 4] monitoring. In the smart home, a convenient way to automatically establish the on/off operational status and identity of an appliance is through the nonintrusive load monitoring (NILM) recognition method which was firstly proposed by Hart in 1992 [5–7]. The NILM method establishes the identity of an appliance through the intelligent extraction of that appliance's specific load signal information from an aggregate load profile acquired through a single signal sampling unit on the main power cable into the building. In contrast, sensors dedicated to each appliance define the intrusive load monitoring (ILM) [5] system. However, the ILM method involves a large number of sensors and extensive cabling in the house. Another recognition scheme known as the semi-intrusive load monitoring (SILM) [8] system only obtains part samples of the aggregate energy and guesses the remainder. SILM cannot give accurate specific load disaggregation but is appropriate for aggregate energy forecasting and needs some sensors and cabling.

The main thrust of the NILM systems is smart-home demand side energy management, whether it is based on single appliance or system based. Hence, we need to know

which and when the appliance/system is switched on or off. Load signal extraction and identification is achieved with high performance when the appliance component signals are due to large power appliances such as electric car charging that have widely varying power differences and whose signatures are very different from each other. The electric car charging in the smart home is now a prominent feature requiring consideration in the NILM recognition system design. The authors in [9] showed that the electric car charging can successfully be implemented into the NILM system using data from the Pecan Street Inc. Dataport. There are a number of challenges facing NILM recognition systems for achieving high recognition performance and they include the follows: (1) the system includes some equal or very close power specification electronic appliances (EVPSAs) during steady state operation and having basically identical signature characteristics, (2) the system has low power appliances that are difficult to recognize and are often interpreted as noise when the aggregate is composed of low and high power appliances (LHPAs), (3) the system includes continuously variable operating states' (CVOS) appliances, and (4) the same power appliances are switched on/off at the same time [5–7, 10]. However, in this paper, motivated by the need to differentiate and monitor the ever increasing array of EVPSAs in the smart home, we limit our research only to challenge (1) above. When summed up a large number of same specification laptops, televisions, refrigerators, light-emitting diode (LED) lamps, etc. will contribute significantly to the energy used in the smart home, and it becomes necessary to identify the operational status for each appliance through a deep learning NILM recognition system. Also, a high number of appliances in the house results in a higher overlap of their respective individual signals and switching events. A few studies often with complex detection algorithms [11, 12] have actively been involved in the NILM recognition of EVPSAs. In this paper, we fill the gap in the established literature by introducing less complex new deep learning model configurations with enhanced computation time and high accuracy for the NILM recognition of EVPSAs. By proposing three deep learning disaggregation algorithms, based on the multiple parallel structures convolutional neural networks (MPS-CNNs), the recurrent neural network (RNN) with parallel dense layers for a shared input, and the hybrid convolutional recurrent neural network (CNN-RNN), we aim to achieve a considerable improvement in the NILM recognition of EVPSAs. In this study, we propose to use in-house generated data from similar low power appliances such as light-emitting diode (LED) main lamps as opposed to the high energy consumption of the electric car charging since they are more difficult to be recognized.

1.2. Literature Survey. In the literature, we identify three approaches to detecting similar appliance signals in the NILM recognition systems. These are (1) event detection [13–15], (2) machine learning with hand crafted features, multiple classifiers, and complex algorithms [11, 12, 16–19], and (3) deep neural networks [3, 4, 10, 20–23]. Event detection algorithms are premised on being able to extract a large number of unique signature characteristics at the beginning, end, and during the transient period. The CUSUM and genetic algorithm have been implemented in solving the recognition challenge due to appliance disaggregated signals that are similar to each other [13]. With reference to the NILM system, the CUSUM adaptive filter is based on adaptive threshold (difference between maximum and minimum value of the parameter being measured within the transient period and the starting and ending of the transient detection [13]). By doing so the filter is capable of extracting the signal information during fast and slow transients. The Genetic Algorithm (GA) on the other hand obtains a fitness function that converges to zero for successful appliance signal recognition [13]. However, although it is capable of extracting a large number of appliance signatures, both the CUSUM adaptive filter and GA are complex requiring involved design. The authors in [14] proposed a high accuracy event detection algorithm (High Accuracy NILM Detector (HAND)) characterized by low complexity and better computation speeds. The HAND monitors the standard deviation of the current signal through the transient period and is capable of detecting unique signal magnitudes within the transient. However, this algorithm suffers suppressed recall value and the precision is sensitive to noise [14]. In [15], an unsupervised clustering event detection algorithm is proposed, which functions on noting the original signal state before and after an event. The approach in [15] is incapable of high recognition at low frequencies. Hence, requiring extra consideration of a large count of high frequency features adds to the complexity and cost of data acquisition.

Machine learning with hand crafted features, multiple classifiers, and complex algorithms seeks to avail a large number of signal features for discrimination between similar appliance signals often through carefully designed feature extraction algorithms for processing through various machine learning models. To date a large number of NILM systems have been developed around Hidden Markov models (HMMs), as HMMs achieve enhanced recognition and reduced computational capabilities. However, HMMs have limited discrete value modeling capability and the algorithms are complex [6, 16]. An emerging method, the NILM Graph Spectral Clustering aggregate energy forecasting method, mentioned in [17] assumes prior knowledge of the appliances' on/off states to provide future disaggregated signal duration of each appliance. This method has a deficiency in the conventional NILM system design as it assumes that appliance will in future always operate as in the past. In reality, appliances are randomly switched on/off at times for varying periods spanning from their minimum operational activation times to up to many hours, days, or weeks depending. Hence, it becomes difficult to implement the design for constantly changing on/off appliance states. The method in [17] is applicable where we have data acquisition of appliances' operating states over very long past periods, unlike in our case where we have limited data as it is the norm in many NILM systems. To this end, the authors in [17] acknowledge the need to enhance the forecasting

capability of this system. In [18], the authors proposed the disaggregation and classification of high power resistive and reactive appliances. They consider step change in implementing their disaggregation to include true and reactive powers of appliances with widely varying signatures. However, the NILM recognition system in [18] is incapable of disaggregating or classifying similar signatures due to its reliance on differentiating between active and reactive powers and on an appreciable level difference between like powers.

Still under machine learning with hand crafted features, multiple classifiers, and complex algorithms, the authors in [19] proposed to improve on the recognition of similar appliances from previous work based only on true power parameter level change by adding more features extracted in total from the true power, reactive power, and power factor of the respective signal. The authors in [19] went on further to propose the MinMaxSteady-State algorithm that constitutes hand crafting of the steady state features from the power and power factor signals. By hand crafting the steady state feature extraction, we increase the complexity of the system and at the same time we limit the system performance since it is difficult by trial and error to determine exactly the number of features required to provide absolute recognition of the appliance signals. In [18, 19], the performance of various classification algorithms that include the decision tree, 5-nearest neighbour, discriminant analysis, and support vector machine was investigated. The decision tree algorithm provided the highest identification rate of appliances for the said classifiers.

In [11], the generalized NILM algorithm provides a considerable improvement in the recognition of similar appliances here given by detecting between iron and kettle. In this algorithm, any machine learning classifier can be used in the recognition. However, different classifiers are assigned to a limited number of features out of the whole set of features under consideration. As in [18], the authors in [11] also consider a step change in the initiation of their disaggregation part of the NILM system. In the finality of the disaggregation, they consider an elaborate design to select an optimal number features out of possible nine features. In [11], the selected features are mean current, DC component, mean power, and for the first sixteen harmonics (active power, reactive power, real and imaginary current components, and conductance and susceptance values). Although the method in [11] gives good discrimination, among the various appliance signals, under consideration, the overall performance of the classifier on the identification between similar appliances requires further improvement as alluded to by the same authors in their conclusions. Furthermore, the number of hand-crafted features under consideration is very high, requiring a complex feature selection and extraction algorithm. In [12], the hierarchical support vector machine (HSVM) classifier is proposed for the classification of the disaggregated signals. However, the HSVM burdens the computational resources of the system. As in [11, 18], the authors in [12] also consider a step change in the formulation of their NILM disaggregation comprising a host of hand-crafted features that include average, peak value, root mean square, standard deviation, crest factor, and form factor for analysis per appliance. In addition to formulation of hand crafted event detection and hand crafted feature extraction, in [12], we observe a slightly suppressed average classification accuracy of 98.8% due to the HSVM.

The advent of deep learning algorithms has allowed for an accelerated increase in the development and performance of NILM recognition systems. In [20], the authors propose the following three deep learning neural networks for the NILM recognition: (1) recurrent neural network and (2) denoising autoencoder, and a model based on considering the steady state operation value and appliance activation start and end times. The experiments in [20] are performed using high power appliances that have widely varying signatures and result in acceptable average F-measures (F1 scores) that are however less than unity. The appliances considered in [20] are kettle, dish washer, fridge, microwave oven, and washing machine. The research here [20] forms one of the basis for application of deep learning to the NILM recognition, and as such requires further improvement as alluded to by the authors in their conclusions. In [20], networks (2) and (3) performed reasonably well for recognition of unseen appliance data, whilst network (1) did not perform well on unseen data. However, all the networks in [20] still need considerable improvement. In [21], the authors propose to predict the extent to which Parkinson's disease is manifest from gait generated data. Just like in NILM recognition, the system in [21] tries to infer an outcome from a composite input of gait information. An averaged output from the result of a parallel combination of a long-short-term memory (LSTM) network and convolutional neural network (CNN) model is obtained. The good results in [21] show that both LSTM and CNN models can be adopted for use in the NILM recognition system as the formats of the power series signals are the same in both cases.

Still under deep learning algorithms in [22] the authors propose a CNN NILM system based on differential input, with the aim of achieving higher performance than systems based on "raw" data. This is somewhat a form of signal preprocessing obtained by differentiating the raw data into power change signals. An auxiliary raw data feed is then applied in parallel to the differential input to provide additional mean, standard deviation, and max and min signal information. However, a well-constructed deep CNN network is capable of high performance internal signal differentiation and feature selection without the need for preprocessing the signals. Furthermore, the authors in [22] used a standard dataset that includes a dishwasher, fridge, and microwave oven without articulating the similar appliances signal issue. In [23], the authors propose a deep learning autoencoder-based NILM recognition system. Applying the concept of noise removal from speech signal, the authors in [23] are able to disaggregate the unique appliance signals from the aggregate with very high performance. However, in [23], the authors experiment on appliances that do not have similar signatures, and these are washing machine, desktop computer, electric stove, and electric heater. In [10], the authors approach the NILM

recognition through a convolutional neural network (CNN) applied to appliance voltage-current (V-I) trajectories. The V-I trajectories are transformed to the image form for input to the CNN. The features in [10] are attributed to slope, encapsulated area etc. of the V-I trajectory. The authors in [10] consider data acquired from high frequency measurements and does not sufficiently address low frequency (1 Hz) data acquisition. In [10], the authors are able to recognize a large pool of appliances from the WHITED and PAID datasets with macroaverage F1 scores of 75.46% and 77.60%, respectively. Poor recognition between similar appliances is a contribution to the low F1 score.

Analogous to detecting similar appliance signals is the modeling of travel behavior patterns for designing a charging strategy for plug-in electric vehicle [3, 4]. In [4], Plug-in Electric Vehicles (PEVs) travel pattern prediction accuracies of up to 97.6% were obtained through a hybrid classification approach. Similar travel patterns are grouped together and assigned to a particular forecasting network. Using stored previous PEVs data (departure time, arrival time, and travelled distance), the approach in [4] first runs an unsupervised model to establish those masked travel-behaviour patterns and assigns them to a specific group. The grouped travel-behaviour patterns are then channelled to the respective supervised model for final recognition. The unsupervised and supervised operations are both performed by LSTM networks that are characterized by enhanced feature extraction capabilities. The results in [4] show that deep learning as opposed to legacy scenario-based demand modeling achieves very high performance in PEV systems. In [3], PEVs travel pattern prediction was obtained through the use of the Rough Artificial Neural Network (R-ANN) with reference to the recurrent neural network system. R-ANNs are capable of enhanced forecasting of the masked travel-behaviour patterns of PEVs. In [3], the Conventional Error Back Propagation (CEBP) and LevenbergeMarquardt training approach was used with the LevenbergeMarquardt achieving higher performance in training Plug-in Electric Vehicles-Travel Behaviour (PEVs-TB). The outcome of the research in [3] shows that the Recurrent Rough Artificial Neural Network (RR-ANN) approach allows for better PEV-TB and PEVs load forecasting than the reference Monte Carlo Simulation (MCS). The overall result in [3] is a substantial saving in the use of electricity by the PEVS. In context of our research, we extend the application of the LSTM model to the NILM disaggregation part.

1.3. Paper Contribution. In this paper, we address the deficiencies mentioned in [11–23] of the NLM disaggregation and classification of EVPSAs with similar signatures by improving the deep learning approach. Deep learning neural networks are good at mastering the complex nonlinear connection between the source aggregate signal and the output target appliance signal. The success of the NILM recognition depends in principle on the feature extraction capabilities of the designed system. Hence, we propose NILM models that will attempt to extract as much feature information as possible from the experimental signals. Firstly, with the view of obtaining appropriate EVPSAs overall high performing disaggregation deep learning networks, we propose three deep learning disaggregation algorithms based on the multiple parallel structures convolutional neural networks (MPS-CNNs), the recurrent neural network (RNN) with parallel dense layers for a shared input, and the hybrid convolutional recurrent neural network (CNN-RNN). We then disaggregate a total of three signal parameters per appliance in each case for a limited number of similar signature appliances in the form of light-emitting diode (LED) main lamps. We propose CNN- and LSTM-based disaggregation networks. The CNN is a feed-forward neural network (FFNN) modelled on the naturally "vision perfect" biological visual cortex [24, 25] and has achieved extremely high levels of object recognition and classification. The LSTM network, on the other hand, which accurately models short and long term trends in the appliance signals [4], is characterized by enhanced feature extraction capabilities. Secondly, we propose an appliance classification strategy premised on the deep multilayer perceptron (MLP) having three appliance signal parameters as input to increase the number of trainable samples and hence accuracy. In the case where we have limited data, we implement a transfer learning- (TL-) based appliance classification strategy. In this paper, our first and second proposals attempt to fill the knowledge gap in the established literature by introducing less complex but powerful new deep learning model configurations with enhanced computation time and high accuracy for the NILM recognition of EVPSAs. The MLP feedforward neural network in its own right is an enhanced nonlinear problem solving deep neural network capable of high classification performance [26]. During data acquisition, we obtain three signal parameter values for both the aggregate and appliance target signals. We then perform a regression-based training of each disaggregation model based on the target parameters. Using the sliding window concept, we disaggregated the appliance signals through the trained disaggregation networks. We then use the mean summation of the part window disaggregated signals to obtain the overall disaggregated signals. We also train the classification network based on the three parameters of the ground truth signals and finally apply the disaggregated signal sums into the trained classification network for recognition. Our proposed NILM recognition system is tested on raw in-house generated data from similar LED main lamps. Disaggregation is carried out on all the appliances, and in the final analysis, we show the classification rates of all the appliances under test. To evaluate the performance of the proposed method, some simulations and comparisons are carried out. In summary, we make the following contributions in this study:

(i) Incorporate an all-encompassing disaggregation feature extraction capability that includes step change, transient, and steady state values deep learning framework based on three separate deep learning disaggregation algorithms: the multiple parallel structure convolutional neural networks, the recurrent neural network with parallel dense layers

for a shared input, and the hybrid convolutional recurrent neural network to substantially increase the disaggregation performance of the NILM system

(ii) Increase the classification accuracy by availing three parameters per signal into the classification network based on a simple deep learning multilayer-perceptron network

1.4. Organization of the Paper. The rest of this paper is structured as follows. Section 2 details the proposed methodology including the models, the proposed NILM recognition theory, aspects pertaining to data, performance metrics, verification of the proposed method performance to include proposed model description, pseudocode for proposed method, Keras model architectures, and the training framework and procedure. Section 3 gives a discussion of the experimental results, and Section 4 gives the conclusion.

2. Methodology

2.1. The Proposed Models. We propose our deep learning model structure based on the hybrid convolutional recurrent neural network (CNN-RNN). The CNN-RNN approach is referred to the GoogleNet model as done by the authors in [27]. However, we modify the concept and break it down into three possible networks for exploration in this paper. The first model in Figure 1 is premised on the multiple parallel structure convolutional neural networks (MPS-CNNs) disaggregation approach. In the GoogleNet model, we basically disaggregate one input parameter with a number of parallel feature extractors, whereas in our model, we disaggregate three independent input parameters, as shown in Figure 1. The second model in Figure 2 is a recurrent neural network in the form of an LSTM with parallel dense layers for a shared input for enhanced sequence prediction. The final model in Figure 3 is based on a hybrid convolutional recurrent neural network (RNN-CNN) that combines the enhanced feature extraction with ordered sequence prediction for CNN and RNN, respectively [28]. The authors in [29] use bidirectional LSTMs (BiLSTM or BLSTM) that preserve past and future information from combined hidden states for better interpretation of missing information. A BLSTM trained on the past and future information 12.17. . .12.175 will predict a 12.1725 instead of a likely 12.178 when trained on an LSTM. Notwithstanding the benefits of BLSTM, we will however base our LSTM models on forward pass ones only. Our models in this paper have three aggregate parameters separately disaggregated to give three individual mains lamp disaggregated signals. These three disaggregated signals become three (multivariate) signal inputs into the classification network with any one target signal of Watt, I_rms, or PF. Doing so may increase the appliance classification accuracy and improve on appliance generalization.

The idea for this research is to place a single measurement piece of equipment at the mains power cable input to the house, and to measure the current, power, and power factor parameters of four similar LED mains to find out which LED is on or not. The recognition module can be housed in a separate meter box next to the original one, or in the house just after the mains circuit breaker, as shown in Figure 4.

This system is meant to recognize similar LED mains lamps effectively connected to an alternating current main power supply cable, either supplied through the power grid, standby generator, or photovoltaic inverter system, to determine which area of the building is illuminated. This project includes the hardware design, signal processing, and signal recognition. The software and hardware can be implemented on microchip or arduino microcontrollers. Besides, the smart-home proposed project can find application in commercial and industrial installations, where there is a large count of similar LED main lamps. The recognition project concept can be extended to other similar electronic appliances such as laptops in a school or company and similar televisions in a hotel. In Figure 4, the NILM unit can then be combined with Internet of Things (IoT) premised on industry 4.0 standard platform for remote access.

2.2. The Proposed NILM Recognition Theory. The typical NILM appliance identification process is made up of (1) acquisition of the composite load profile, (2) obtaining of appliance state transitions (events), (3) feature extraction, and (4) with reference to supervised and unsupervised learning obtaining the disaggregated appliance signal and its class [6]. In supervised learning, the input aggregate is trained against each appliance signature target. In unsupervised learning, there is no target training but an intermediate disaggregated signal is produced which is compared with a known signature databank for pairing; if no pairing is possible, then the intermediate signal is labelled as a new appliance signature. Acquisition of the composite signal can be carried out at high sampling frequencies of 1 kHz to 100 MHz [6]. However, 1 Hz low sampling frequencies are the norm as sampling integrated into smart meters requires simple hardware [8]. The data in our study has been sampled at this low 1 Hz frequency for ease of acquisition. The feature extraction and disaggregated and classification appliance signatures can either be taken as steady state or transient state [5, 6, 8, 30]. Switching transients for each appliance are of different amplitudes, contain unique settling times, and harmonics thereby defining a unique signature for each appliance. On the contrary, steady state features define the normal operational unique signatures of appliances. The mathematical expressions of the load signatures and composite profiles have conveniently been represented in [31]. In our study, the disaggregation problem stated in [31] is tackled by implementing the "pattern recognition" approach that allows us to use the deep learning algorithms that we have proposed.

2.2.1. Deep Learning Algorithms. Well-configured deep learning neural networks are capable of extracting a large number of different features that define an input signal, whereas some deep learning algorithms are better at

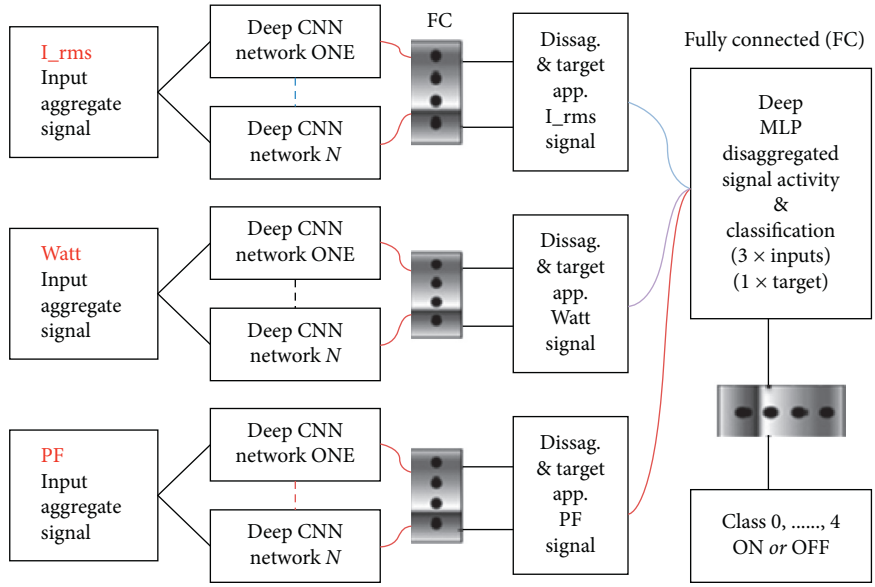

Figure 1: Multiple parallel structure convolutional neural networks. Incorporating the appliance disaggregation and classification.

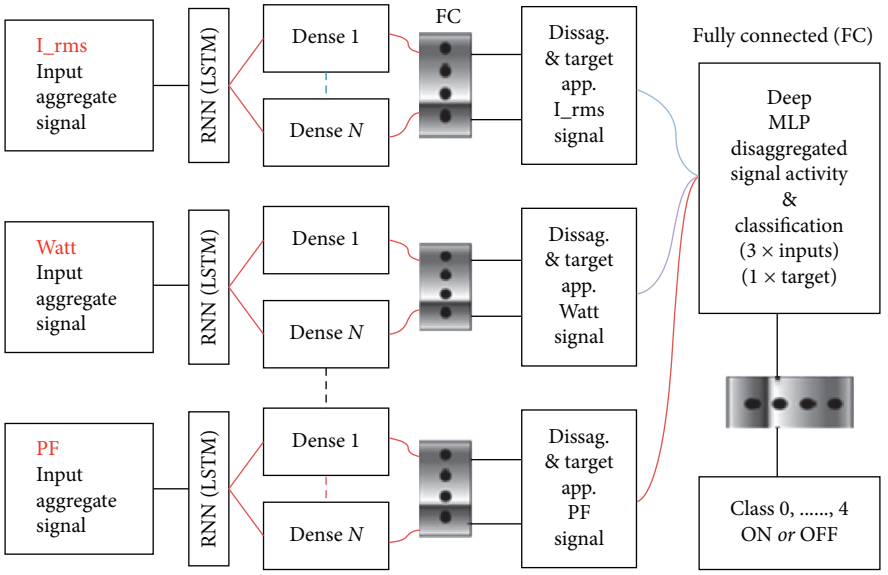

Figure 2: Recurrent neural network. Incorporating the appliance disaggregation and classification.

regression-based analysis some deep learning models such as the multilayer perceptron (MLP) feedforward neural network is more situated to classification [26]. However, the MLP normally forms the last stage of most CNN or RNN (LSTM) deep neural networks. According to [32], inputs bounded by convex polygon decision regions are sufficiently solved by two-layer feedforward networks where the inputs are continuous real and the outputs are discrete values. The underlying layers in a CNN are convolution, pooling or subsampling and fully connected or multilayer perceptron [24, 25]. The convolution through nonlinearity (ReLU) to pooling layers has feature extraction capabilities. Pooling effectively reduces the dimension of the preceding feature maps but maintaining all the important detail of the input, while the object recognition and classification is performed through the backpropagation algorithm in the fully connected layer. CNNs also require little data preprocessing. The image can be a three (red, green, and blue) channel or single (greyscale) channel matrix with pixel values 0 to 255.

In this paper, the CNN is adapted to 1D aggregate appliance signal inputs and targets. A matrix (the filter, kernel, or feature detector) of smaller dimension than the input matrix is used as the feature detector. Different filter matrix entries will extract different features of the input image. In appliance classification, the number of outputs is required to be equal to the number of appliances under test [10, 29]. CNNs have recently been incorporated into Capsule Networks (CapsNets) for significantly improved feature extraction and recognition based on dynamic routing by agreement rather than max pooling of image-based datasets

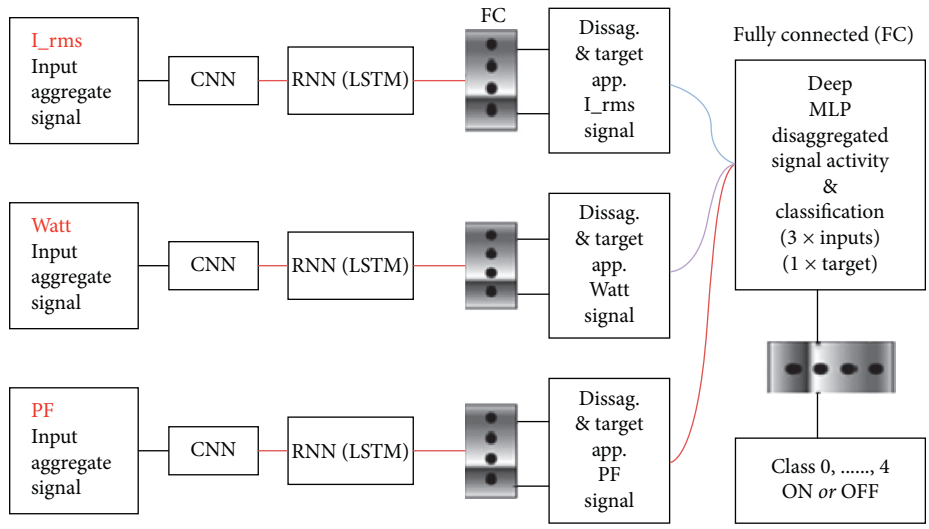

Figure 3: Hybrid convolutional recurrent neural network. Incorporating the appliance disaggregation and classification.

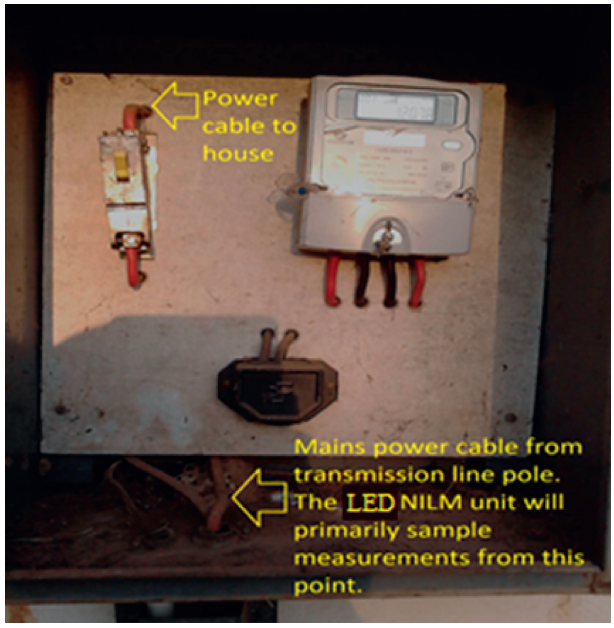

Figure 4: Power cables in the meter box. We show the provision for installing the LED main lamps NILM recognition module.

[33]. However, the application of CapsNets in the NILM scheme is not yet extensively documented and is not considered for application in this paper. Convolutional neural network training error can be significantly reduced by the use of a filter-based learning pooling (LEAP) CNN algorithm developed by the authors in [34]. However, in this paper, we use CNNs based on the traditional hand engineered average pooling scheme.

An RNN shown in Figure 5 is a neural network formulated to capture information from sequences and is based on considering immediate and just previous inputs in its calculations. As such the RNN has some memory attributes to easily enable it to decide the outcome of next input determined by the conditions of the stated present and just

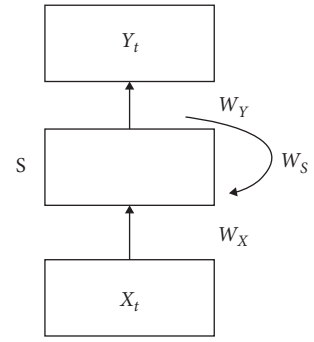

Figure 5: Recurrent neural network architecture. The outcome of next input is determined by the conditions of the stated present and just previous inputs.

previous inputs. A deep RNN is obtained by channelling consecutive S hidden layers from previous RNNs to subsequent RNN inputs. However, the RNN suffers a gradient problem which adversely affects model performance. To this end, the RNN-LSTM network is developed to solve the vanishing gradient issue by putting gating functions within its operation process [6, 10, 20, 35]. The RNN state expression is given infd1

$$S_t = F_W\left(W_X X_t + W_S S_{t-1}\right), \tag{1}$$

where S_t is hidden state at time step t; W_X is weights between hidden layer and input; W_S is weights between previous and current layers; X_t is input at time step t; F_W is a recursive function (tanh or ReLU); W_Y is weights between hidden and output layers; and S_{t-1} is previous hidden state at time $t-1$.

2.2.2. Disaggregation. As opposed to Hart's disaggregation framework that emphasizes event detection rather than individual appliance disaggregation from the composite signal [27], in this paper, we focus on the latter technique. The authors in [22, 36] use a sliding window on the aggregate. Sliding windows that partially overlap with each

other have dimensions that depend on the appliance activation sizes. A median filter is then used to add the intermediate outputs to get the final output. Kelly and Knottenbelt [20] propose, on the contrary, the constitution of the intermediate outputs by considering their mean values. In this particular case, the output is recognized by the start, end, and mean values of the target appliance from the aggregate. While disaggregation considers on all the data points on the target appliance, classification is based on assigning a label value that relates the disaggregated signal to the ground truth appliance signature. The authors in [27] base their disaggregation scheme on the parallel connection of CNN/RNN layers with varying filter sizes of 1×1, 3×3, 5×5, and 7×7 as in the GoogleLeNet structure. These CNN/RNN layers are then concatenated together after having extracted a large number of useful signal features from the aggregate signal. In this paper, the training to validation datasets are split in the ratio 7 : 3, respectively.

2.2.3. Transfer Learning-Based Classification. The method of using a model trained on a larger dataset which is similar to the new smaller dataset is known as transfer-based learning. Transfer learning allows for the speedy development of new models on constrained datasets and allows the application of these models in more varied situations [37, 38]. Transfer learning is more compactly defined as follows [37].

Definition 1. Given a set of source domains $DS = D_{s,1}, \ldots, D_{s,n}$, where $n > 0$, a target domain, D_t, a set of source tasks $TS = T_{s,1}, \ldots, T_{s,n}$, where $T_{s,i} \in TS$ corresponds with $D_{s,i} \in DS$, and a target task T_t which corresponds to D_t, transfer learning helps improve the learning of the target predictive function f_t in D_t, where $D_t \notin DS$ and $T_t \notin TS$.

2.3. Aspects Pertaining to Data. We use a set of mains lighting lamps in the form of light-emitting diodes (LEDs) in our experiments. Three of the lamps are shown in Figure 6. The measurement setup is performed in the laboratory where we use the same length of extension cables from the mains to the lamps. Hence, we do not consider the effect cable length contribution to our collected data. We obtain three aggregate signal parameters sampled at 1 sec intervals per mains lighting lamp using a Tektronix PA1000 Power Analyser [39]. The parameters that we measured for each light-emitting diode lamp are voltage current (I_rms), power (Watt), and power factor (PF). We create an appliance signature databank of all the individual mains lamps. These signals are our target data in the deep leaning training. We will not show the individual LED lamp signatures here, but in Section 3, when we compare these signatures (ground truths) with the reconstructed disaggregated signals as a way of accessing the performance of the disaggregation. Model simulation is performed in the Python 3.5 environment with Keras 2.2.2 TensorFlow 1.5.0 backend, Numpy, Pandas, and scikit-learn packages, on an Intel R CPU 1.60 GHz 4.00 GB Ram 64 bit HP laptop.

From the composite current (I_rms) signal, as shown in Figure 7, a recognition strategy is developed for a set of three

Figure 6: Experiment LED lamps. We show three of the lamps with the fourth being LED1-2 having the same power specification as LED1-1 and LED2-1.

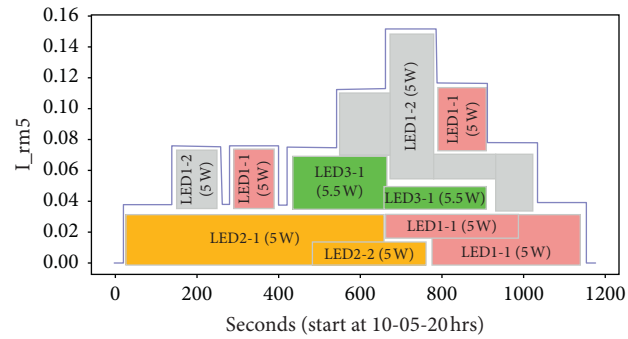

Figure 7: Equal power appliances (EPAs) aggregate current (I_rms) signal acquired on 09 Nov. 2018, showing the various combinations of the main lamp activations.

5 W and one 5.5 W light-emitting diode (LED) lamps numbered as LED1-1 (Philips 5 W (60 W) 100 V–240 V), LED1-2 (Philips 5 W (60 W) 100 V–240 V), LED2-1 (Philips 5 W (60 W) 170–250 V), and LED3-1 (Radiant 5.5 W B22 Candle 230 V, 50 Hz, 5000 K). For example, we aim to disaggregate LED1-2 from LED1-2 and LED2-1 aggregate. The aggregate power (Watt) and power factor (PF) signals equally valid also are not shown. As can be seen in a 600 seconds window in Figure 8, from the dynamics of the four LEDs, there is an order of less than ten to the power minus 4 difference in current magnitude for three LEDs and very close relationships in the steady state profiles of all the LEDs. This shows close tolerance of the LED characteristics especially for LED1-2 and LED1-1 as expected from the specifications.

The aspects pertaining to the selection of the training signal points are

(i) The overall length of the target series (T) defines the input and output series lengths into and out of the network, respectively (regression training)

(ii) The target series data should not be too long but enough to sufficiently define the ground truth signal

(iii) The on/off points should be captured in the target and aggregate data, with the training period chosen to be longer than the appliance activation window that incorporates appliances' start and end

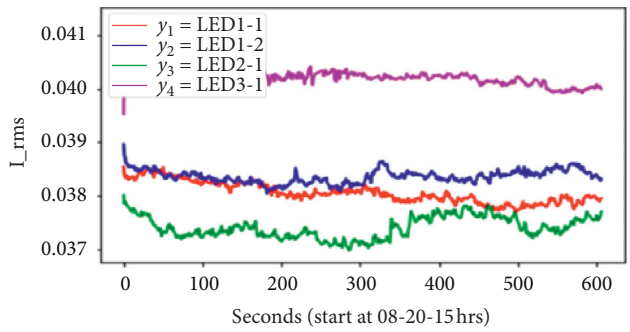

Figure 8: I_rms waveforms for the LED lamps (LED1-1, LED1-2, LED2-1, and LED3-1), showing the close tolerance of the parameter magnitude. For the first 300 seconds, LED1-1 and LED1-2 are hardly discernable from each other.

(iv) The disaggregation algorithm should align the target and the point at which target becomes active in the aggregate signal

The overall length of the aggregate signal should contain all the information about the specific target appliance. We consider the shape of the aggregate data and accordingly reshape our input data into the DL network. We can generate artificial data where our raw data is too limited for deep learning. CNN and LSTM are both premised on a three-dimensional input whose shape is [number of samples, timesteps length, and number of features]. The hybrid CNN-LSTM system requires that we further obtain subsequences from each sample. The CNN works on the subsequences with the LSTM working, summarizing the CNN results from the subsequences.

The aggregate data is normalized and then standardized (zero mean and unit standard deviation) to improve on deep learning (DL) gradient convergence. DL algorithms require a large training dataset and as a result before the normalization and standardization the acquired dataset (only input training data) size is increased by considering all sections of the entire aggregate signal where the target appliance appears. For example, the input training set for LED1-2 is enlarged from 121 sample points to 614 (spanning 5 LED1-2 activations) sample points by considering the total aggregate data length covered by the grey areas in Figure 7. Likewise, for LED2-1, the total aggregate signal length is obtained by considering the orange areas, an increase from 119 sample points to 714 (spanning six LED2-1 activations) sample points. The further addition of artificially generated data as done by Kelly and Knottenbelt [20] in their 50 : 50 ratio of real aggregate data to artificially generated data will improve the ability of our network to generalize to "unfamiliar" appliances not involved in the training.

As in [20], we created additional artificial data by synthesizing random values between the maximum and minimum readings of the aggregate signal from the RANDBETWEEN function in excel. Although there is a further possibility of increasing the aggregate length by adding generated delayed versions of the total real aggregate signal where that appliance appears, we experimented with only these increased real sample points plus synthesized samples to give respective total aggregate lengths of (614 real + 614 synthetic) for LED1-2 and (714 real + 714 synthetic) for LED2-1. The validation aggregate signal in Figure 9 is only real data without synthetic additions; however, this data is normalized and standardized. The validation dataset (containing the appliance activations) length is 441 samples in total with, for example, 121 to 363 samples for LED1-2 and 119 to 238 samples for LED2-1.

Data trains for Watt and PF are also available and applicable to the developed algorithm evaluation.

In this paper, using the prepared data, we first train the model in Figure 1 using only one network with varying filter sizes, and we obtain its performance, reconstruct the disaggregated signal, and compare it with the ground truth signal. We go on to add subsequent parallel networks and perform the overall networks' performance evaluation until there is no more appreciable change as we add extra parallel arms. It is only after this do we employ the disaggregated signal for an absolute classification test. Like other researchers [22, 36], we also employ the sliding window shown in Figure 10 based on the appliances activation size in the disaggregation. During training and using data prepared from Figure 5, we go on to add another network to have a model with two parallel networks. For the second added network, we again vary the filter sizes and evaluate the resultant parallel networks' performance and how good the reconstructed disaggregated signal is compared with the ground truth signal. In the second and final models, we gradually vary the RNN/LSTM memory cells while noting the performance.

We develop our recognition models in the random order of LED1-2, LED2-1, LED1-1, and LED3-1. For LED1-2, the actual target sample (divided by the largest value in that sample) length is 76 with four zeroes at start and end of series broken down as $((68 \times 1) + (8 \times 1))$ features. The actual aggregate length is 1224 including four aggregate signal samples that have no information about LED1-2 at both ends of the series, broken down as $((68 \times 18) + (8 \times 18))$ features. It should be noted that only one parameter is disaggregated at time, but three parameters are used in the classification.

The resultant disaggregated signal is obtained by finding the mean values of the window disaggregated parts. In some cases, the aggregate signals in Figures 7 and 9 span as little as 120 sample points with the disaggregated signal represented by as little as 68 sample points of data after the removal of redundancies. This represents limited data for use during the classification stage. Hence, we propose to use pretrained classification networks that use data spanning as much as 600 sample points for each ground truth signal obtained from an independent but related measurements, as shown in Figure 11. We then train the classification using this extended time series and implement transfer learning to test and classify the shorter disaggregated signals that are based on shorter initial target lengths. The disaggregation task is given by Pseudocode 1.

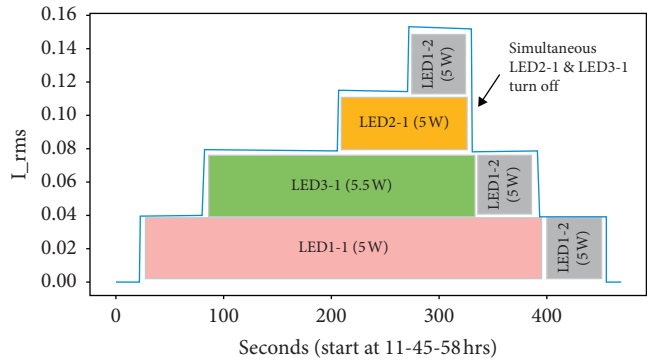

Figure 9: Disaggregation testing/validation data train acquired on 09 Nov. 2019 for the equal power appliances' (EPAs) LEDs. This signal is input into the trained disaggregation models to extract the appliance activations and hence their signatures.

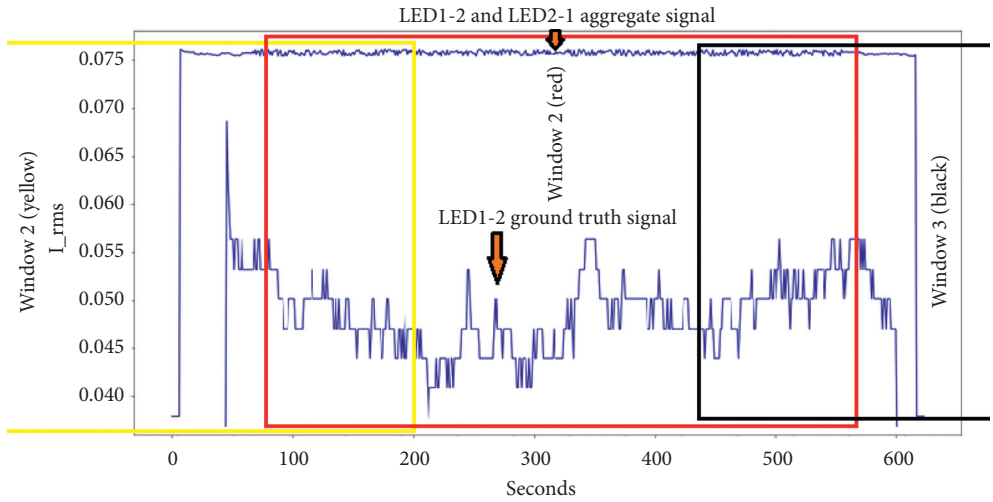

Figure 10: Sliding window in appliance disaggregation. The aggregate is a composite of LED1-2 and LED2-1. We disaggregate LED1-2 based on sliding window length equal to ground truth signal length.

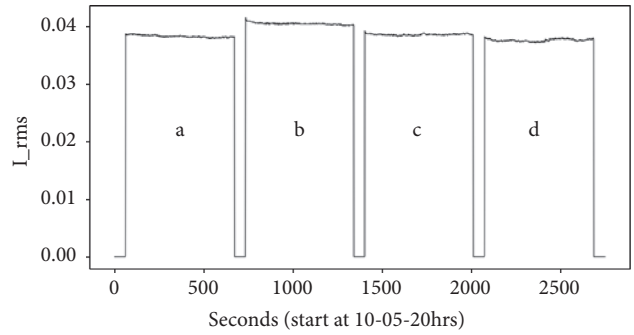

Figure 11: Independent classification data for LED lamps. This shows the LEDs series lengths for transfer learning acquired on 09 Nov. 2018. (a) LED1-1, (b) LED3-1, (c) LED1-2, and (d) LED2-1.

2.4. Performance Metrics. In this paper, for disaggregation performance, we consider the logcosh, root-mean-square-error (RMSE), mean_squared_error (MSE), and mean_absolute_error (MAE), and Coefficient of Determination (CD) (R^2) for the model evaluation. To evaluate our regression models, the R^2 shows the close relationships between the predicted and training values, with a good $R^2 \longrightarrow 1$. Logcosh is not easily affected by spurious predictions. Whilst, we consider the accuracy (Acc), recall (R), precision (P), F-measure (f_1), and confusion matrix for the classification [6, 7, 40]. We can also compare a plot of the reconstructed signal with the ground truth signal plot of each appliance through superimposition of these plots to physically see the relationship of these two signals:

(1) Begin: obtain preprocessed, formatted, and transformed training input aggregate data of series length TT_a secs according to Figure 7
(2) Obtain training target data of series length TT_t secs with redundancies removed
(3) Train the network
(4) Obtain preprocessed, formatted, and transformed validation/test input aggregate data of series length TV_a secs according to Figure 9
(5) Specify disaggregation window, $T_W = TT_t < TV_a < TT_a$
(6) Slide trained network input through validation/test aggregate data by amount equal to disaggregation window
(7) Repeat 6, until end of validation/test aggregate data series length
(8) Use mean method sum up all results of disaggregation window movement to obtain disaggregated signal of series length TT_t TTt secs
(9) Input 7 into trained classification network for appliance recognition
(10) Repeat 1 to 9, until performance $\longrightarrow$ 100%

Pseudocode 1: The disaggregation task

$$\begin{aligned}
\text{MSE} &= \frac{1}{T}\sum_{i=1}^{n} |\overline{y} + \widehat{\overline{y}}|^2, \\
\text{RMSE} &= \sqrt{\frac{1}{T}\sum_{i=1}^{n} |\overline{y} + \widehat{\overline{y}}|^2}, \\
\text{MAE} &= \frac{1}{T}\sum_{i=1}^{n} |\overline{y} + \widehat{\overline{y}}|, \\
\text{Logcosh}\,(t) &= \sum_{p\in^{p}} \log\left(\cosh\left(p - t\right)\right), \\
R^2 &= 1 - \frac{\sum \left(\text{Original} - \text{Predicted}\right)^2}{\left(\text{Original} - \text{Originalmean}\right)^2}, \\
\text{Acc} &= \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}, \\
R &= \frac{\text{TP}}{\text{TP} + \text{TN}}, \\
P &= \frac{\text{TP}}{\text{TP} + \text{FP}}, \\
f_1 &= \frac{2 * P * R}{P + R},
\end{aligned} \tag{2}$$

where T is activation time (time-series) for each appliance, $i = 1, \ldots, n$ is number of appliances, $\widehat{\overline{y}}$ is disaggregated power signal, $\overline{y}$ is aggregate actual power at time t, Original is target signal and Predicted is the disaggregated signal, TP is true positives, FP is false positives, FN is false negatives, and TN is true negatives [6, 7].

The results and discussion may be presented separately, or in one combined section, and may optionally be divided into headed sections.

2.5. Verification of the Proposed Method Performance

2.5.1. Proposed Model Description. Disaggregation is performed by using a sliding window on real test/validation data. Training is performed by using a combination of real and synthesized data to improve on the recognition generalization of the NILM system. The disaggregation is performed on three parameters one at a time using the three proposed models separately. Each model goes through three training and disaggregation processes for the disaggregation part, excluding the classification part. Hence, we assign the three trained and disaggregating model outputs for model 1 in Figure 1 as *mdl1I_rms, mdl1Watt*, and *mdl1PF*. Likewise for model 2 in Figure 2 and model 3 in Figure 3 we have *mdl2I_rms, mdl2Watt, mdl2PF, mdl3I_rms, mdl3Watt*, and *mdl3PF*, respectively. In summary, the number of disaggregating trained model outputs are nine (three per model), and the total number of disaggregating signals is nine. Of the three models, under consideration, we note the one with the higher or better disaggregation (regression performance plot) performance and exclude the results of the other models for further processing. This effectively leaves us with only three better disaggregated signals at any one time represented by *mdlbI_rms, mdlbWatt*, and *mdlbPF, where mdlb* is model better output.

The classification model is trained based on tuning the MLP hyperparameters to provide the best performance on the ground truth signal parameters of I_rms, Watt, and PF for four input LED similar signature appliances. The total number of parameters input into the classification network is twelve during the training stage. However, in the recognition stage, the total number of signal parameters input into the trained MLP is three, obtained from the best disaggregating model (that is, the *mdlb* model output). Due to the limited data for training the MLP deep network, we implement transfer-based learning where we train the classification network on a larger training dataset of the four LEDs than the one we have acquired that is directly related to the experiment.

2.5.2. Pseudocode for Proposed Method. The proposed method evaluates the performance of the disaggregation algorithm on three models and carries out the classification only on one model. Although we have the same disaggregation task, we have in actual fact three disaggregation

algorithms due to the different model structures. Hence, we show the pseudocodes of the training of the three disaggregation algorithms one for each model as Pseudocodes 2–4. Pseudocode 1, which shows the actual sliding window disaggregation, is a common operation in the three different disaggregation algorithms. We then add Pseudocode 5 which shows how the classification is performed.

2.5.3. Keras Model Architectures. The architectures for the models we used in the disaggregation and classification are given as follows.

(1) Disaggregation. For Model 1 (MPS-CNN), the architecture we used is detailed as follows:

(i) Input of length equal to T of target series.

(ii) Three parallel double layer 1D convolutional networks filter sizes 64 and 128, 64 and 28, and 64 and 28 but having kernel size = 1, 3, and 7 each and activation = relu. Each network has a single MaxPooling1D(pool_size = (2)) layer

(iii) A merge layer.

(iv) Three hidden dense layers with 50, 100, and 200 neurons, and activation = relu.

(v) Output dense layer of length equal to T of target series.

For Model 2 (RNN), the architecture we used is detailed as follows:

(i) Input of length equal to T of target series.

(ii) An LSTM layer with 500 memory cells and two parallel dense layer networks, one with 1024 neurons and the other with three layers have

  (i) LSTM(500)
  (ii) Dense(1024, activation = "relu") first parallel dense network
  (iii) Dense(500, activation = "relu") second parallel dense network in series with two dense layers comprising a Dense(1024, activation = "relu"), and a Dense(500, activation = "relu") layer

(iii) A merge layer.

(iv) An output dense layer of length equal to T of target series.

For Model 3 (CNN-RNN(LSTM)), the architecture we used is detailed as follows:

(i) A TimeDistributed 1D convolutional network with 128 of filter sizes 1, followed by another 1D convolutional layer with 256 filters and filter size 1, activation = relu, and a single TimeDistributed(MaxPooling1D(pool_size = 2)) layer

(ii) A flatten layer

(iii) Three hidden LSTM hidden layers with memory cells of lengths 1024, 4096, and 1024, respectively

(iv) A hidden dense layers with 512 neurons, and activation = relu

(v) Output dense layer of length equal to T of target series.

We experimented with learning rates of the Adam optimizer from 0.0000001 to 0.1 and found a good compromise for a value of 0.01. We used the logcosh to evaluate all regression-based experiments and also included and evaluated other regression metrics as given in the results.

(2) Classification. We have developed the classification algorithm using transfer learning and have adopted the weights from the large dataset given in Figure 11 to our constrained dataset. The MLP transfer learning model used is shown below. The CNN is more appropriate when the classification input dimension is very large. However, in our case, for training, we format the data as a matrix of three parameter values (multivariate time series of thirty columns (points) per parameter for current (I_rms), power (Watt), and power factor (PF).

The MLP transfer learning-based classification architecture is

(i) Input into Dense layer with 8 units, activation = "relu," and input_dimension = 3

(ii) A hidden Dense layer with 10 units and activation = "relu"

(iii) A hidden Dense layer with 16 units and activation = "relu

(iv) An output Dense layer (Dense(3, activation = "softmax"))

The model used the Adam optimizers with a validation split = 0.3, one hot encoded labels, and only 50 epochs to achieve high performance. In the architecture shown, we use only 3 classes instead of 4, and the reason is explained in detail in Section 3. Although the classification model above achieved good performance, we are able to reach high validation accuracy faster by changing the input Dense layer to 500 units.

2.5.4. Training Framework and Procedure. The classification training framework is based on the Rectified Linear Unit (ReLU) activation function, the softmax function, selecting maximum number of epochs of 50, the Adam optimizer, and a validation split of 0.3. We initially provisionally include the training dropout regularization in the classification model. The ReLU shown in Figure 12 is an operation meant to introduce nonlinearity in the network, and it replaces all negative values with zero. Nonlinearity network characteristics are required to solve complex nonlinear situations. All the disaggregation networks are also based on this ReLU [24] activation function.

Furthermore, CNN networks inherently perform linear operations, and as such to consider nonlinearity, we incorporate the ReLU activation.

The basic training procedure of the MLP is defined by

$$w_{i+1} = w_i + \Delta w = w_i + \eta y_i (x_i - w_i y_i), \tag{3}$$

where η is the learning rate, x_i is an m-dimensional input vector (input neuron), and $y_i = w_i^T x_i y_i = x_i^T w_i$ (output

(i) Acquire the aggregate and ground truth signals for the selected parameter (p)
(ii) Define: no. of parallel CNN networks (pn), hidden layers in each pn, filters in each pn, dense layers (ds), epochs, and early stopping condition
(iii) Initialize all weights
While parameter (p) is True: **do**

(1) Epochs = 0 (train)

(i) Iterate through each pn
(ii) Concatenate all pn
(iii) Iterate through each ds
(iv) *mdl1* converged ⟶ stop

(2) Disaggregation

(i) Pseudocode 1

(3) *mdl1* not converged

(i) epochs = epochs + 1
(ii) Return [(**1**) (i)]

Otherwise: **do**

(4) $p = p + 1$ (next parameter): **do**
(**1**) to (**3**).
(5) Train and disaggregate all parameters:

(i) $p \leq 3$: return [(**4**)]
(ii) $p > 3$: Exit

Pseudocode 2: Training and disaggregation process of the MPS-CNN (*mdl1*)

(i) Acquire the aggregate and ground truth signals for the selected parameter (p)
(ii) Define: no. of LSTM memory cells, parallel MLP networks (MLPn) connected in series with LSTM layer, hidden layers in dense parallel layers (dsp), dense layers (ds) connected to series combination of LSTM + MLPp, epochs, and early stopping condition
(iii) Initialize all weights
While parameter (p) is True: **do**

(1) Epochs = 0 (train)

(i) Iterate through LSTM
(ii) Iterate through LSTM + each MLPn arm
(iii) Concatenate all MLPn layers
(iv) Iterate through each ds
(v) *mdl2* converged ⟶ stop

(2) Disaggregation

(i) Pseudocode 1

(3) *mdl2* not converged

(i) epochs = epochs + 1
(ii) Return [(**1**) (i)]

Otherwise: **do**

(4) $p = p + 1$ (next parameter): **do**
(**2**) to (**3**).
(5) Train and disaggregate all parameters:

(i) $p \leq 3$: return [(**4**)]
(ii) $p > 3$: Exit

Pseudocode 3: Training and disaggregation process of the RNN (LSTM) (*mdl2*)

(i) Acquire the aggregate and ground truth signals for the selected parameter (p)
(ii) Define: no. of CNN networks including hidden (CNNn), filters, LSTM networks including hidden and memory cells (LSTMn), dense layers (ds), epochs, and early stopping condition
(iii) Initialize all weights
While parameter (p) is True: **do**

(1) Epochs = 0 (train)

(i) Iterate through each CNNn
(ii) Iterate through LSTMn
(iii) Iterate through each ds
(iv) *mdl3* converged ⟶ stop

(2) Disaggregation

(i) Pseudocode 1

(3) *mdl3* not converged

(i) epochs = epochs + 1
(ii) Return [(**1**) (i)]

Otherwise: **do**

(4) $p = p + 1$ (next parameter): **do**
(**3**) to (**3**).
(5) Train and disaggregate all parameters:

(i) $p \leq 3$: return [(**4**)]
(ii) $p > 3$: Exit

Pseudocode 4: Training and disaggregation process of the CNN-RNN (LSTM (*mdl3*))

(i) Acquire four LED ground truth extended time series signals for transfer learning (TL)
(ii) Acquire three disaggregated signals of *mdlbI_rms*, *mdlbWatt* and *mdlbPF* from the best performing disaggregation model (*mdlb*), no. of neurons of TL model per dense layer, and early stopping condition
(iii) Initialize all weights
(1) Epochs = 0 (train TL network)

(i) Iterate through each MLPn
(ii) TL model converged ⟶ stop

(2) TL model not converged

(i) epochs = epochs + 1
(ii) Return [(**1**) (i)]

(3) Classification of disaggregated signals: **do**

(i) Input into TL
(ii) Recognize LED ⟶ stop

Pseudocode 5: Transfer learning-based deep MLP classification

neuron) is the output. The new and old synapse weights are w_{i+1} and w_i, respectively, and the weight change is given as Δw [41]. The backpropagation (Error-correction (ECL) (supervised learning is typically used to update the learning weights Δw)). The error change can be used to increase or decrease the magnitude of the weight update component given in equation (4). A change in the weight results in a change in the error. Minimum error point is achieved through gradient descent. However, gradient descent may converge to local instead of global minima. Hence, there is need to mitigate this shortcoming by continually randomly selecting the initial weights during the training process [42]:

$$\Delta w_{kj} = -\eta \frac{\partial E_k}{\partial w_{kj}}, \tag{4}$$

where E_k with respect to node k is the resultant sum of squares of errors (cost or loss function) between the target output (y_t) and the network output (y_o). For all the weights in the network, an update of these weights is achievable through the backward propagation (BP) of this error through the said network [42]. The backpropagation algorithm is more efficient than the normal feedforward algorithm. This is so because there are more passes to achieve significant weight change in a normal feedforward network

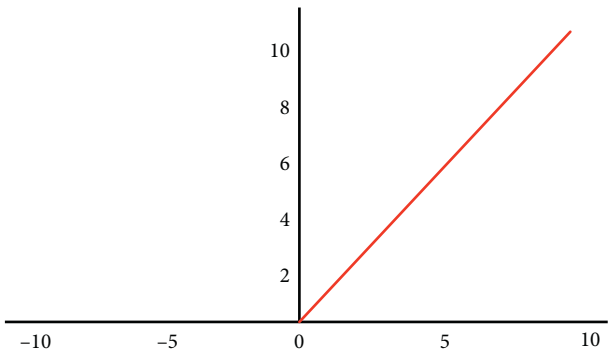

Figure 12: ReLU nonlinear activation function.

than there are in backpropagation. The algorithms of the standard and improved backpropagation methods referenced to Figure 13 are given below.

Let the same error backpropagated through the network be $E = E_k$ and the activation function be the sigmoidal $(\sigma(\cdot))$ one as given in

$$z_i = \sigma(q_i) = \frac{1}{1 + e^{-q_i}}. \tag{5}$$

Then, the standard backpropagation error derivative between nodes i and j is

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial q_j} \cdot \frac{\partial q_j}{\partial w_{ji}} = \delta_j \cdot z_i, \tag{6}$$

where

$$\delta_j = \frac{\partial E}{\partial q_j} = \sum_k \frac{\partial E}{\partial q_k} \cdot \frac{\partial q_k}{\partial q_j} = \sum_k \delta_k \cdot \frac{\partial q_k}{\partial q_j},$$
$$\frac{\partial q_k}{\partial q_j} = \frac{\partial q_k}{\partial z_j} \cdot \frac{\partial z_j}{\partial q_j} = w_{kj} \cdot \sigma'(q_j). \tag{7}$$

Node 2 delta is given as

$$\delta_j = \sigma'(q_j) \cdot \sum_k \delta_k w_{kj}. \tag{8}$$

The error derivative between nodes j and k is

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial q_k} \cdot \frac{\partial q_k}{\partial w_{kj}} = \delta_k \cdot z_j. \tag{9}$$

Equation (9) can be written as

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial (y_t - y_o)} \cdot \frac{\partial (y_t - y_o)}{\partial w_{kj}},$$
$$\frac{\partial E}{\partial w_{kj}} = \sum (y_t - y_o) \cdot -\frac{\partial (\sigma(q_k))}{\partial w_{kj}}, \tag{10}$$
$$\frac{\partial E}{\partial w_{kj}} = -\sum (y_t - y_o) \cdot y_o (1 - y_o) \cdot z_j.$$

Comparing equations (9) and (10), the magnitude of δ_k is given by $\sum (y_t - y_o) \cdot y_o (1 - y_o)$:

(i) Node 1 layer error is given by $\sum_j \delta_j w_{ji}$ (where this node is another hidden layer node in a two hidden layer network)

(ii) Node 2 layer error is given by $\sum_k \delta_k w_{kj}$

(iii) Node 3 layer error is given by $(y_{\downarrow t} - y_{\downarrow o})$

The standard backpropagation algorithm is given below:

(1) Obtain initial values of weights and offsets.

(2) Establish the input vector x and target output. Also, determine the number of hidden and output units.

(3) Find the deltas (δ) for all the output nodes.

(4) Backpropagate the deltas using $\delta_j = \sigma'(q_j) \cdot \sum_k \delta_k w_{kj}$.

(5) Evaluate the derivatives $(\partial E / \partial w_{kj}) = \delta_k \cdot z_j$ for all synapses.

(6) Update the weights according to w_{kj} (new) $\longleftarrow w_{kj}$ (old) $- \eta$ $(\partial E / \partial w_{ji})$.

(7) Repeat (2).

In the recognition training, we experimented with various optimizers that included the Adam, rmsprop, and sgd. The sgd was set to optimizers SGD((lr = 0.000001 to 0.1), decay = $1e-6$, momentum = 0.9, netrov = True). The Adam and rmsprop were set to a learning rate that varied between 0.000001 and 0.1. Both the Adam and sgd optimizers performed well with a learning rate of 0.01 and 0.001 for the disaggregation and classification algorithms, respectively. The categorical_crossentropy cost function was used in the classification model training. We also experimented with various activation functions that included the tanh (sigmoid) (mainly used in artificial neural networks (ANNs) since its characteristics can accommodate both linear and nonlinear situations), relu, and the leaky_relu (an improvement over the normal relu). We settled on the relu which achieved acceptable performance. In the output stages of the disaggregation and classification models, we implanted the linear and softmax activation functions, respectively. We also experimented with the l_1 and dropout regularizers, but found out that due to the relatively simpler designed models the regularization did not affect the performance of the algorithms. Hence, there was no need to implement regularization in all the models. The choice of the number of hidden layers, neurons (units), number and size of CNN filters, and memory units in the LSTM was achieved through trial and error.

With respect to the CNN and LSTM disaggregation networks, we invoke the training procedure after specifying the Keras model architectures. The input aggregate power series of length (T) is trained against another power series represented by the target series also on length (T), $X = \{x1, x2, x3, \ldots, xT\}$. The objective of the training procedure is to minimize the regression cost functions

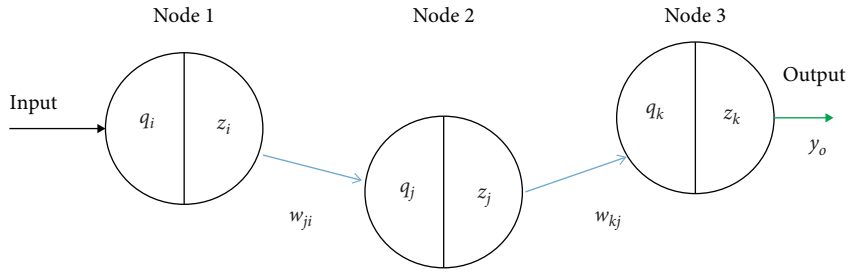

Figure 13: Cross-section of multilayer neural network.

represented by logcosh, root-mean-square-error (RMSE), mean_squared_error (MSE), and mean_absolute_error (MAE). However, another regression function, the Coefficient of Determination (CD) (R^2) for the model evaluation is required to be high. We also evaluate the training computation times of the proposed models.

3. Results and Discussion

3.1. Regression Training and Disaggregation. We compare our proposed models to each other and only use the output from the most accurate model as input into the classifier. Although disaggregation was carried out on all the LEDs, we limit our analysis to one LED lamp; however, we show the classification rates of all the LEDs. If we can achieve good performance for one LED, then we can also achieve good performance for other LED lamps since the features and their relative magnitudes are almost similar. Figures 14–16 show the relative performance of the regression models for LED1-2 I_rms signal using the data in Figure 7. The ground truth signal for this LED1-2 lamp is shown in Figure 17.

We did achieve comparable results for the power and PF signals. We experimented with different LSTM memory lengths and we found lengths above 500 provided good results. Furthermore, when we tried paralleling the LSTM networks by using the API Keras structure, we did not get an improvement in the LSTM model results. However, the network based on a single LSTM network provided acceptable results. The model based on the CNN-RNN also provided good regression results. It is, however, the MPS-CNN structure that achieved top disaggregation in this paper. The MPS-CNN structure allows us to capture a wide range of features and detail that include the on/off edge detection.

3.2. Classification. For the LED1-2 recognition, we apply three disaggregated input parameters into a deep MLP classification network. We first train the network on a larger dataset depicted in Figure 11 than the one obtained from the disaggregated signal in the transfer learning-based classification scheme. We fine tune the network on the larger dataset and when we have obtained satisfactory results, as shown by the training Figures 18 and 19, we apply the model on our disaggregated dataset. In Figure 18, we show that the model accuracy achieves high value early in the training of the TL model. From the training and validation loss characteristics in Figure 19, we show that our MLP TL model is very stable and the characteristics converge well. We tried six different classification MLP models using the larger dataset and all models misclassified LED1-1 and LED1-2 that have exactly the same specifications and identical parameter values. Also, where LED1-1 and LED1-2 appear in the disaggregation algorithm, we were not able to separate the two from each other.

Hence, we eliminate one of the LEDs, LED1-1, in our analysis as there is no added useful recognition information. So, in the whole recognition process, LED1-1 is taken as LED1-2. This explains why the classification model under Keras Models' Architectures is based on three classes. In future, we can detect LED1-1 and LED1-2 by considering the actual cable lengths that are different from each other from the main supply in a typical building installation. In the laboratory measurement setup, we did not factor in this issue and we just measured the appliance parameters using the same extension cables from the mains distribution point. We can also use deeper learning which is not possible in our experimental CPU platform. In addition, recognition can be based on parameter phase change and some advanced event detection schemes. Due to the initial experimental results, we modify our recognition strategy to only consider LED1-2, LED3-1, and LED2-1. In this case, for LED1-2, the class is 0, for LED3-1 the class is 1, and for LED2-1, the class is 3.

Table 1 and Figure 20 show the classification report and the classification matrix, respectively, of the model trained using a larger dataset in Figure 11. We see that all the three achieve one hundred percent classification. In Figure 20, the history parameters are batch size-1, epochs-50, steps-None, samples-892, verbose-2, do_validation-True, and metrics [loss, acc, val_loss, val_acc]. The classification model in Figure 20 achieved the following: Evaluation: loss-0.010676, accuracy-1.0, Test score-0.0173, and Test accuracy-1.0. We transfer this model without modification to the smaller disaggregated dataset in transfer learning, where we maintain the same class labels, as shown in the confusion matrix in Figure 21. Table 2 shows the classification report for Figure 21.

Table 3 gives the regression-based metrics during the training of the disaggregation algorithms.

It is necessary to evaluate the relative computation times of the models, especially those for the disaggregation algorithms. A fast computation time allows for fast turnaround of program development and indirectly implies less stress on the computation processor. The code for evaluating the computation time of each model is given as

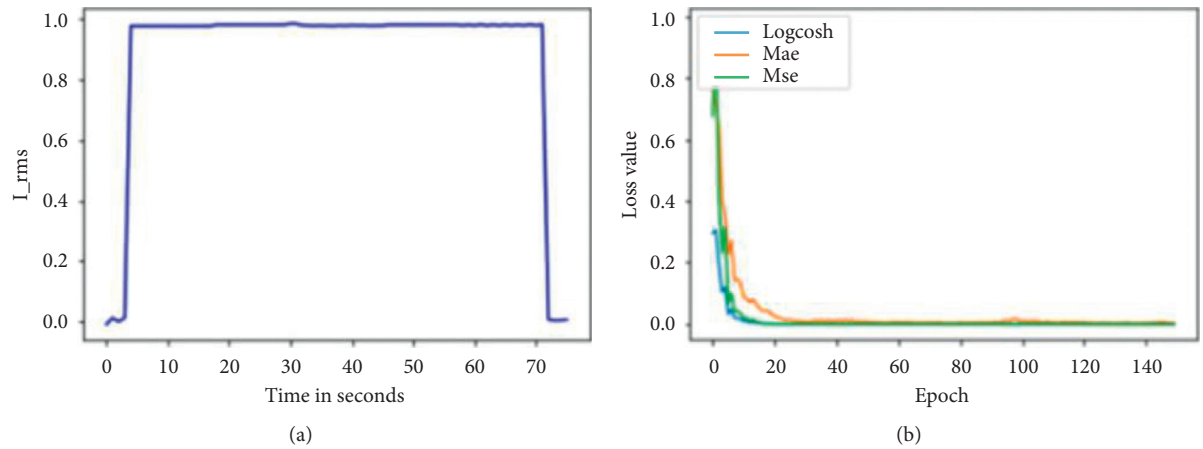

Figure 14: LSTM disaggregation model plots. (a) Reconstructed signal and (b) loss.

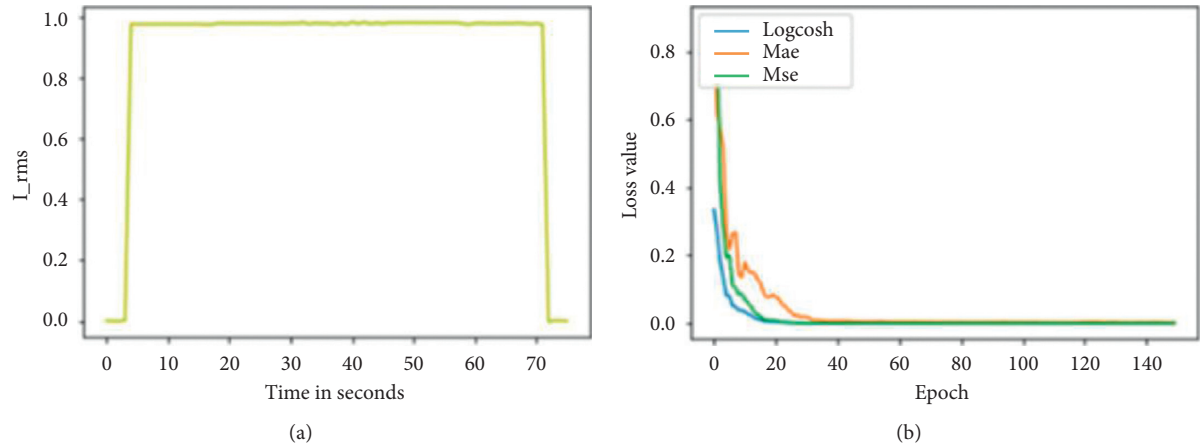

Figure 15: CNN-LSTM disaggregation model plots. (a) Reconstructed signal and (b) loss.

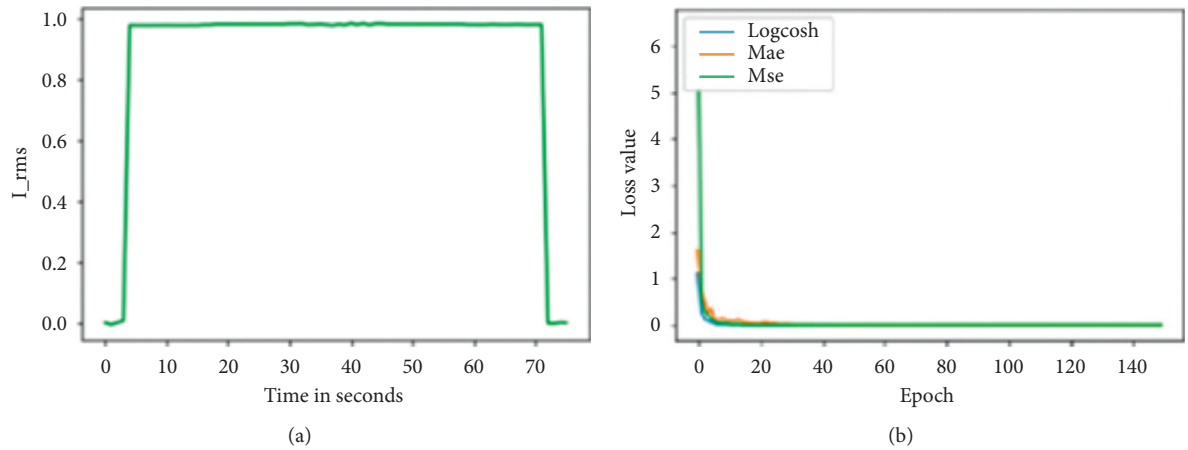

Figure 16: MPS-CNN disaggregation model. (a) Reconstructed signal and (b) loss.

```
from timeit import default_timer as timer
from datetime import timedelta
start = timer()
history = model.fit(X, Y, epochs = 150, verbose=, validation_split = 0.3) #Any model training.
end = timer()
```

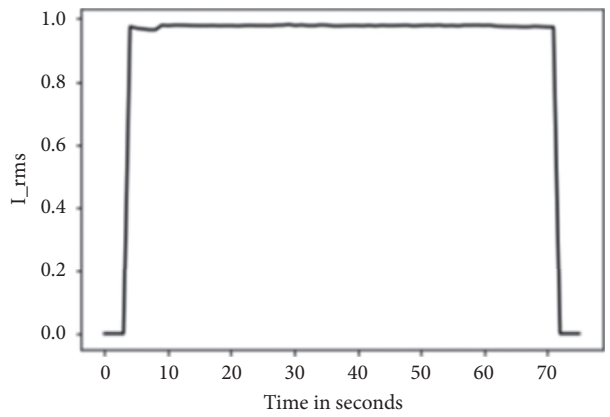

Figure 17: Ground truth signal for LED1-2.

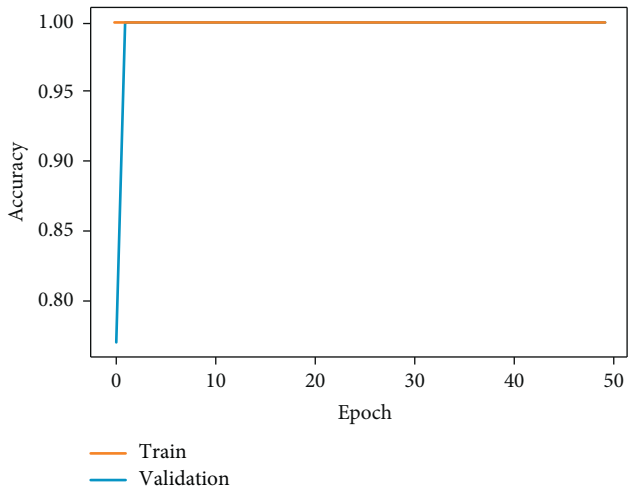

Figure 18: Model accuracy for the classification TL model training.

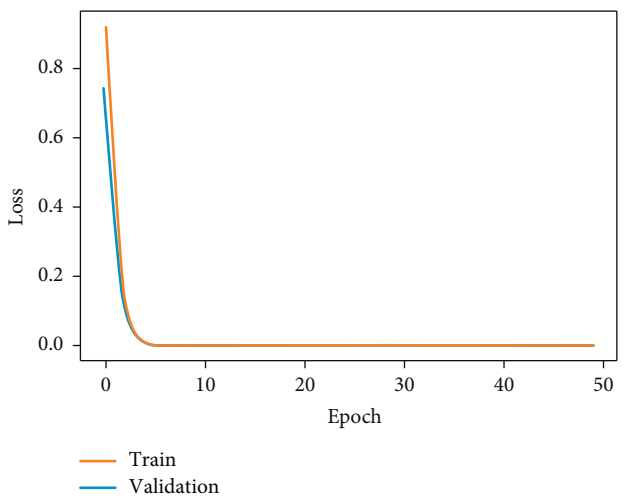

Figure 19: Model loss for the classification TL model training.

Table 1: Transfer learning model classification report.

Class	Precision	Recall	*F*-score	Support
0	1.00	1.00	1.00	175
1	1.00	1.00	1.00	175
2	1.00	1.00	1.00	175
Weighted avg	1.00	1.00	1.00	525

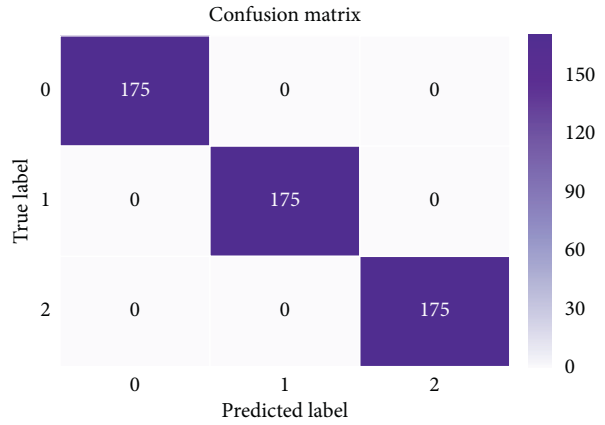

Figure 20: Classification matrix for the model trained on disaggregated dataset. This is the transfer learning stage. All base lamp signals are accurately classified.

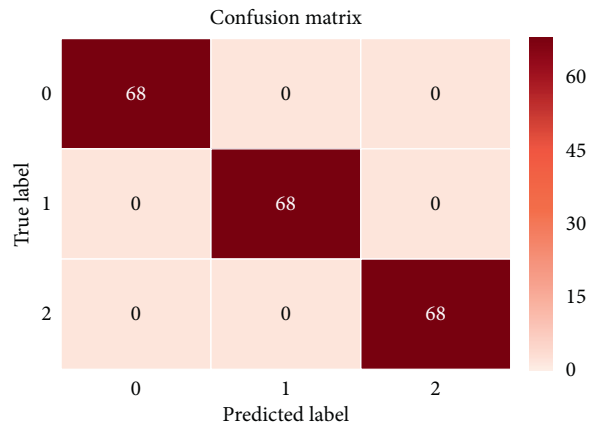

Figure 21: Classification matrix for the disaggregated lamp signals. All lamps are accurately classified.

Table 2: Disaggregated signal recognition classification report.

Class	Precision	Recall	*F*-score	Support
0	1.00	1.00	1.00	68
1	1.00	1.00	1.00	68
2	1.00	1.00	1.00	68
Weighted avg	1.00	1.00	1.00	204

print(timedelta(seconds = end-start))

Table 4 shows the computation times of the models in relation to the total trainable paramaters. The computation times of the models increase with an increase in the number of trainable parameters. The MLP-TL classification process is the fastest due to its simpler network structure and the fewer number of output labels required as compared to the

Table 3: Disaggregation regression-based model metrics.

Metric	MPS-CNN	LSTM	CNN-LSTM
MAE	0.0036	0.0025	0.0022
MSE	$2.275e-05$	$1.254e-05$	$9.420e-06$
RMSE	0.00477	0.00354	0.00307
CD	1.00	1.00	1.00

Table 4: Training computation times of the NILM models.

Model name	Number of trainable parameters	Number of epochs	Computation time (hh:mm:ss)
MLP TL	3,503	50	0:07:38.034314
MPS-CNN	238,543	150	0:00:57.302319
LSTM	2,794,573	150	01:6:46.786972
CNN-LSTM	8,429,057	150	0:38:21.626814

longer output power series signal samples required the disaggregation algorithms. The MPS-CNN model is faster than the LSTM-based models that have a larger number of total trainable parameters. LSTM RNN networks are adapted to capturing of information from power series data or sequences. However, LSTM RNNs suffer degraded performance [43] when the information is available in very long power series such as the ones we have in the NILM recognition. As such, this slows down their training computation times. Large LSTM RNN blocks also have a large number of gating functions which increases the number of trainable parameters, hence computation time.

The results show the ability of our proposed models to achieve high disaggregation and classification accuracy of the LED lamps in our experiment. It is important to take cognizance of the fact that state-of-the-art [20, 32] systems tested on a variety of widely deferring appliance specifications using more or less the same types of models might outperform our recognition in accurate classification of all test samples. In our case, we had to eliminate one highly misclassified LED1-1 in the final analysis. However, this paper is biased towards developing algorithms to recognize relatively low power appliances having the same specifications. Our argument has here been that if we can accurately classify and disaggregate low power same specification appliances, then naturally it should be a matter of fact to achieve the same for the widely varying power levels different specification appliances.

4. Conclusions

This paper evaluated three NILM disaggregation and one classification algorithm for equal power appliances with almost similar signatures, in the form of three 5 W and one 5.5 W mains LED lamps. We used the following labelled LED lamps in our experiments: LED1-1 (Philips 5 W (60 W)), LED1-2 (Philips 5 W (60 W)) and LED2-1 (Philips 5 W (60 W)), and LED3-1 (Radiant 5.5 W). We show that same specification appliances can indeed be recognized from each other. However, we need a cautious and elaborate approach in developing a holistic NILM recognition for appliances that have identical specifications. In our study, we had to eliminate in the final analysis from our experiments LED1-1 as it grossly misclassified as LED1-2 since its characteristics were almost identical to those of LED1-2. The point of divergence from the normal approaches was the disaggregation and classification based on three appliance parameters to substantially increase the accuracy. This in itself did not cure the problem. As no two appliances are exactly the same from manufacture, developing deeper learning algorithms is one possible way of solving this problem; however, the CPU platform we operated from has limitations both in speed and processing power. The results also show that equal power specification appliances should have parameters measured whilst in the actual installation and not in laboratory to take advantage of such issues as contributions due to wiring where we can measure phase change, time lag, wiring resistance etc. from the sampling point. However, our NILM recognition strategy is promising as we did obtain accurate recognition for some of the lamps.

Data Availability

All the data and codes used in this paper are available from the authors at the University of Johannesburg.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported partially by South African National Research Foundation Grants (nos. 112108 and 112142) and South African National Research Foundation Incentive Grants (nos. 95687 and 114911), Eskom Tertiary Education Support Programme Grants, Research grant from URC of University of Johannesburg.

References

[1] D. Ding, R. A. Cooper, P. F. Pasquina, and L. Fici-Pasquina, "Sensor technology for smart homes," *Maturitas*, vol. 69, no. 2, pp. 131–136, 2011.

[2] V. Fabi, G. Spigliantini, and S. P. Corgnati, "Insights on smart home concept and occupants' interaction with building controls," *Energy Procedia*, vol. 111, pp. 759–769, 2017.

[3] H. Jahangir, H. Tayarani, A. Ahmadian et al., "Charging demand of plug-in electric vehicles: forecasting travel behavior based on a novel rough artificial neural network approach," *Journal of Cleaner Production*, vol. 229, pp. 1029–1044, 2019.

[4] H. Jahangir, S. Sadeghi Gougheri, B. Vatandoust, M. Aliakbar Golkar, A. Ahmadian, and A. Hajizadeh, "Plug-in electric vehicle behavior modeling in energy market: a novel deep learning-based approach with clustering technique," *IEEE Transactions on Smart Grid*, 2020.

[5] N. F. Esa, M. P. Abdullah, and M. Y. Hassan, "A review disaggregation method in non-intrusive appliance load monitoring," *Renewable and Sustainable Energy Reviews*, vol. 66, pp. 163–173, 2016.

[6] C. Nalmpantis and D. Vrakas, "Machine learning approaches for non-intrusive load monitoring: from qualitative to quantitative comparation," *Artificial Intelligence Review*, vol. 52, no. 1, pp. 217–243, 2019.

[7] R. Bonfigli, S. Squartini, M. Fagiani, and F. Piazza, "Unsupervised algorithms for non-intrusive load monitoring: an up-to-date overview," in *Proceedings of the IEEE 15th International Conference on Environment and Electrical Engineering (EEEIC)*, pp. 1175–1180, Rome, Italy, June 2015.

[8] Q. Liu, K. M. Kamoto, X. Liu, M. Sun, and N. Linge, "Low-complexity non-intrusive load monitoring using unsupervised learning and generalized appliance models," *IEEE Transactions on Consumer Electronics*, vol. 65, no. 1, pp. 28–37, 2019.

[9] A. U. Rehman, L. T. Lie, B. Valles, and S. R. Tito, "low complexity event detection algorithm for nonintrusive load monitoring systems," in *Proceedings of the 2018 IEEE Innovative Smart Grid Technologies-Asia (ISGT Asia)*, pp. 746–751, Singapore, May 2018.

[10] L. De Baets, J. Ruyssinck, C. Develder, T. Dhaene, and D. Deschrijver, "Appliance classification using VI trajectories and convolutional neural networks," *Energy and Buildings*, vol. 58, pp. 32–36, 2018.

[11] P. Bilski and W. Winiecki, "Generalized algorithm for the non-intrusive identification of electrical appliances in the household," in *Proceedings of the 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*, pp. 730–735, Bucharest, Romania, September 2017.

[12] G. Hoogsteen, J. O. Krist, V. Bakker, and G. J. M. Smit, "Non-intrusive appliance recognition," in *Proceedings of the 3rd IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*, pp. 1–7, Berlin, Germany, October 2012.

[13] K. N. Trung, E. Dekneuvel, O. Zammil et al., "Event detection and disaggregation algorithms for NIALM system," in *Proceedings of the International Symposium on Industrial Electronics (ISIE)*, pp. 28–31, Taipei, Taiwan, May 2013.

[14] M. N. Meziane, P. Ravier, G. Lamarque, J.-C. L. Bunetelt, and Y. Raingeaud, "High accuracy event detection for non-intrusive load monitoring," in *Proceedings of the 2017 IEEE International Conference on Accoustics, Speech and Signal Processing (ICASSP)*, pp. 2452–2456, New Orleans, LA, USA, March 2017.

[15] T. Bernard and M. Marx, "Unsupervised learning algorithm using multiple electrical low and high frequency features for the task of load disaggregation," in *3rd International Workshop on Non-Intrusive Load Monitoring, Electric Power Research Institute—EPRI-, NDE Center NILM 2016*, pp. 1–5, Vancouver, Canada, May 2016.

[16] A. S. Bouhouras, P. A. Gkaidatzis, E. Panagiotou, N. Poulakis, and G. C. Christoforidis, "A NILM algorithm with enhanced disaggregation scheme under harmonic current vectors," *Energy and Buildings*, vol. 183, pp. 392–407, 2019.

[17] C. Dinesh, S. Makonin, and I. V. Bajić, "Residential power forecasting using load identification and graph spectral 25 clustering," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 6, no. 11, pp. 1900–1904, 2019.

[18] O. Hamid, M. Barbarosou, P. Papageorgas, K. Prekas, and C. Salame, "Automatic recognition of electric loads analyzing the characteristic parameters of the consumed electric power through a non-intrusive monitoring methodology," *Energy Procedia*, vol. 119, pp. 724–751, 2017.

[19] M. Figueiredo, A. de Almeida, and B. Ribeiro, "Home electrical signal disaggregation for non-intrusive load monitoring (NILM) systems," *Neurocomputing*, vol. 96, pp. 66–73, 2012.

[20] J. Kelly and W. Knottenbelt, "Neural NILM: deep neural networks applied to energy disaggregation," in *Proceedings of the 2nd 31 ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, pp. 55–64, Seoul, South Korea, November 2015.

[21] A. Zhao, L. Qi, J. Li, J. Dong, and H. Yu, "A hybrid spatio-temporal model for detection and severity rating of Parkinson's disease from gait data," *Neurocomputing*, vol. 315, pp. 1–8, 2018.

[22] Y. Zhang, G. Yang, and S. Ma, "Non-intrusive load monitoring based on convolutional neural network with differential input," *Procedia CIRP*, vol. 83, pp. 670–674, 2019.

[23] F. C. C. Garcia, C. M. C. Creayla, and E. Q. B. Macabebe, "Development of an intelligent system for smart home energy disaggregation using stacked denoising autoencoders," *Procedia Computer Science*, vol. 105, pp. 248–255, 2017.

[24] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights Into Imaging*, vol. 9, no. 4, pp. 611–629, 2018.

[25] https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/, June 2020.

[26] P. Lacko, "From perceptrons to deep neural networks," in *Proceedings of the IEEE 15th International Symposium on Applied Machine Intelligence and Informatics*, pp. 169–172, Herl'any, Slovakia,, January 2017.

[27] W. He and Y. Chai, "An empirical study on energy disaggregation via deep learning," in *Proceedings of the 2nd International Conference on Artificial Intelligence and Industrial Engineering (AIIE 2016)*, vol. 133, pp. 338–342Advances in Intelligent Systems Research, Beijing, China, November 2016.

[28] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4580–4584, Brisbane, Australia, April 2015.

[29] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, 2017.

[30] D. de Paiva Penha and A. R. C. Castro, "Home appliance identification for NILM systems based on deep neural network," *International Journal of Artificial Intelligence and Applications (IJAIA)*, vol. 9, no. 2, pp. 69–80, 2018.

[31] J. Liang, S. K. K. Ng, G. Kendall, and J. W. M. Cheng, "Load signature study—part I: basic concept, structure, and methodology," *IEEE Transactions on Power Delivery*, vol. 25, no. 2, pp. 551–560, 2010.

[32] R. P. Lippmann, "An introduction to computing with neural nets," *IEEE Acoustics Speech and Signal Processing Magazine*, vol. 4, no. 2, pp. 4–22, 1987.

[33] S. Sabour, N. Frost, and G. E. Hinton, "Dynamic routing between capsules," in *Advances in Neural Information Processing*, pp. 3856–3866, 2017.

[34] M. Sun, Z. Song, X. Jiang, J. Pan, and Y. Pang, "Learning pooling for convolutional neural network," *Neurocomputing*, vol. 224, p. 96, 2017.

[35] H. Patel, A. Thakkar, M. Pandya, and K. Makwana, "Neural network with deep learning architectures," *Journal of*

Information and Optimization Sciences, vol. 39, no. 1, pp. 31–38, 2017.

[36] R. Bonfigli, A. Felicetti, E. Principi, M. Fagiani, S. Squartini, and F. Piazza, "Denoising autoencoders for non-intrusive load monitoring: improvements and comparative evaluation," *Energy and Buildings*, vol. 158, pp. 1461–1474, 2018.

[37] D. Cook, K. D. Feuz, and N. C. Krishnan, "Transfer learning for activity recognition: a survey," *Knowledge and Information Systems*, vol. 36, no. 3, pp. 537–556, 2013.

[38] S. Al-Stouhi and C. K. Reddy, "Transfer learning for class imbalance problems with inadequate data," *Knowledge and Information Systems*, vol. 48, no. 1, pp. 201–228, 2016.

[39] PA1000, *Power Analyser User Manual*, Tektronix, Inc., Beaverton, OR, USA, 2018.

[40] https://www.tensorflow.org/opi_doc/python/tf/keras/losses/logcosh, June 2020.

[41] K. Samiee, A. Iosifidis, and M. Gabbouj, "On the comparison of random and hebbian weights for the training of single-hidden layer feedforward neural networks," *Expert Systems with Applications*, vol. 83, pp. 177–186, 2017.

[42] M. Bascema and W. J. Tastle, *Intelligent Data Mining in Law Enforcement Analysis: New Neural Networks Applied to Real Problems*, Springer, Berlin, Germany, 2013.

[43] F. Karim, S. Majumdar, H. Darabi, and S. Chen, "LSTM fully convolutional networks for time series classification," *IEEE Access*, vol. 6, pp. 1662–1669, 2018.