DEVELOPMENT OF SMART ON-DEVICE BASED ELECTRICITY CONSUMPTION FORECAST MODEL FOR ENERGY MANAGEMENT

ALEXANDER NDIFE

A Thesis Submitted to the Graduate School of Naresuan University
in Partial Fulfillment of the Requirements
for the Doctor of Philosophy in Smart Grid Technology
2022

DEVELOPMENT OF SMART ON-DEVICE BASED ELECTRICITY CONSUMPTION FORECAST MODEL FOR ENERGY MANAGEMENT

ALEXANDER NDIFE

A Thesis Submitted to the Graduate School of Naresuan University
in Partial Fulfillment of the Requirements
for the Doctor of Philosophy in Smart Grid Technology
2022

Thesis entitled "Development of Smart On-device based Electricity Consumption Forecast Model for Energy Management"

By ALEXANDER NDIFE

has been approved by the Graduate School as partial fulfillment of the requirements for the Doctor of Philosophy in Smart Grid Technology of Naresuan University

Oral Defense Committee

..... Chair
(Associate Professor Boonyang Plangklang, Ph.D.)

..... Advisor
(Yodthong Mensin, Ph.D.)

..... Co Advisor
(Professor Paisarn Muneesawang, Ph.D.)

..... Co Advisor
(Associate Professor Wattanapong Rakwichian, Ph.D.)

..... Internal Examiner
(Associate Professor Nipon Ketjoy, Dr.Ing.)

Approved

.....
(Associate Professor Krongkarn Chootip, Ph.D.)

Dean of the Graduate School

Title	DEVELOPMENT OF SMART ON-DEVICE BASED ELECTRICITY CONSUMPTION FORECAST MODEL FOR ENERGY MANAGEMENT
Author	ALEXANDER NDIFE
Advisor	Yodthong Mensin, Ph.D.
Co-Advisor	Professor Paisarn Muneesawang, Ph.D. Associate Professor Wattanapong Rakwichian, Ph.D.
Academic Paper	Ph.D. Dissertation in Smart Grid Technology, Naresuan University, 2022
Keywords	Forecasting, Deep Learning, Ensemble method, Neural Networks, HEMS, Automation, Edge Computing, Real-Time Control, Power Consumption

ABSTRACT

This thesis developed a Deep Learning algorithm for power consumption forecasting implementable in low memory storage and energy systems like our smartphones, IPad and Tablets. Power forecasting is a multidisciplinary task that forms an important aspect of energy generation, distribution, and management. When forecasting is done in a smart way, it will help in both energy conservation and resource planning. Interestingly, the traditional means of estimating power usage through previous utility bills is nowadays being replaced with machine intelligence. This research is motivated by the quest to determine power consumption expectation for a medium-term (a week ahead) given the current load demand and the possibility of monitoring and controling energy usage at home or offcie in a real-time from anywhere in the world. This forecast model leveraged on multivariate dataset to make a multi-step time series (7 consecutive days ahead) forecast. It is split into: 1) Problem Framing - here, considerations were made on what type of forecast we really want: short, medium or long term; 2) Modelling - it entails finding the consumption behaviour that captures features of the medium-term forecast that was chosen; 3) Forecasting - predicting the power need for the chosen medium-term period of 7days; 4) Smart Home Energy Management System - finding a way to control appliances that consumes energy in order to limit the energy usage within the

estimated threshold of the forecast result.This forecast model is based on ConvLSTM-Encoder-Decoder algorithm explicitly designed to enhance the quality of spatiotemporal encodings throughout its feature learning process.

However, randomness and other challenges of training neural networks necessitated the ensemble approach used, where multiple models were trained but allowed contribution to prediction by each model to be weighted proportionally to their level of trust and estimated performance on the model. This architecture in principle investigated power consumption of manually operated home against a smart home and its performance tested on time-domain Household Electricity Power Consumption dataset from France; and further validated using a real time load profile collated from School of Renewable Energy and Smart Grid Technology (SGtech), Naresuan University Smart Office. RMSE of 361kWh was recorded compared with 465kWh on persistence model and an improved RMSE of 358kWh was achieved when validated using holdout validation data from the automated office. However, overall performance on error, forecast time and computational speed was later compared with research efforts in literature and the result obtained showed a significant improvement. And comparative analysis carried out between the energy consumption of a manually operated office and a smart office using the proposed SmartHEMS showed that this smart app saved about 24% of the energy normally consumed in the office.

ACKNOWLEDGEMENTS

Firstly, I give glory to Almighty God who gave me the opportunity to embark upon PhD and also made it possible for this academic journey I started about 3years ago to come to a joyful end with good health. My profound gratitude goes to my able advisor Dr Yodthong Mensin, a gentleman extraordinary. Sir, I thank you immensely for your supports in terms of funding and provision of research materials that made it possible for timely completion of this program.

I also want to appreciate my two fathers in Thailand: Assoc. Prof. Dr Wattanapong Rakwichian and Prof. Dr Paisarn Muneesawang. These two academic giants made me to believe that one man can actually have more than one father. Their fatherly support and advice kept my hope alive especially at those hard times in this academic journey that I nearly gave up hope. Sincerely speaking, I have not missed my biological father at home for once since I arrived Thailand because these amiable professors of repute made me to see Thailand as my home. I also want to thank in a special way Dr Sukrudee Sukchai for her motherly support and advice. She gave me a warm welcome on my arrival in SGtech and since then has continued to have my back at all times. As a matter of fact, I cannot remember asking her for any kind of assistance without getting it right away.

My appreciation cannot be completed without mentioning Assoc. Prof. Dr Ing-Nipon Ketjoy who is a strong pillar in rapid transformation ongoing in SGtech in terms of academic expansion. Sir, I am grateful to be a beneficiary of your academic contributions in SGtech. I also want to appreciate in a special way Assoc. Prof. Dr Sakda Somkun and Assoc. Prof. Dr Tawat Suriwong for all their efforts towards the growth of SGtech. My warm greetings also goes to Assist. Prof. Dr Pisit Maneechot, Assist. Prof. Dr Prapita Thanarak, and Dr Sahataya Thongsan for providing me the moral support and encouragement in the course of my study. I cannot forget to extend my appreciation to Dr Russamee Sitthikhankaew, Dr Pornthip Mensin, Dr Malinee Kaewpanha, and Dr Wisut Chramsa-ard and other academic staff who in one way or the other have supported me towards achieving this academic fit.

On a general note, I want to use this medium to thank all members of great SGtech family; her technical and administrative staff especially Alice, who has been

very proactive to my requests for assistance at all times. It is true I bothered her so much but she never complained for once rather she will always put up a smiling cheeks no matter the level of stress. I can confirm to anyone who cares to hear that SGtech staffers are more than team players and also very diligent in carrying out their assigned responsibilities. I am not forgetting my friend Pornnipa Nunocha (Mixx), who has been of great help to me in most of the challenging times ranging from difficulty in the use of iThesis platform and other online services including download of software and analysis tools.

Finally, I extend my love to my family especially my parents Mr. Joseph & Mrs. Julie Ndife, who have been supporting me in their daily prayers. Almost every week I receive numerous calls and text messages from my family members and relatives to confirm that I am in a good health physically and mentally. I am writing this acknowledgment with a fulfilled heart because I have kept my promise to my lovely mother that I will reach the peak of academic world, which has always been her wish for me. Mom I appreciate your surprise visit to me in the course of this doctoral program and the delicious meals your made during your those period. I am grateful to my siblings Ifenyinwa, Stanley, Oge and Ebuka, my aunt Ebele, and Uncles Peter, Edwin and Sunday, you guys are wonderful and has been my backbone. I also extend my warm heart and special love to my new born niece Obiageli-aku Ndife. I love you all and will remain indebted to all of you.

ALEXANDER NDIFE

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

PUBLICATIONS

The research in this thesis has resulted into 4 yet to be published publications in total (4 peer reviewed): 3 journal articles, 1 conference proceedings. The key publications are shown in bold.

I. Alexander N. Ndife[1], Wattanapong Rakwichian [2], Paisarn Muneesawang [3] and Yodthong Mensin [4] "Cyber-Security Audit for Smart Grid Networks**:** An Optimized Detection Technique Based on Bayesian Deep Learning" published in *Journal of Internet Services and Information Security (JISIS),* Volume 12, Issue 2, May 2022, pp. 95-114, 2022, DOI:10.22667/JISIS.2022.05.31.095.

II. Alexander N. Ndife[1], Wattanapong Rakwichian [2], Paisarn Muneesawang [3] and Yodthong Mensin [4] "Smart Power Consumption Forecast Model with Optimized Weighted Average Ensemble" *IAES International Journal of Artificial Intelligence (IJ-AI)* Vol. 11, No. 3, September 2022, pp. 1004~1018 ISSN: 2252-8938, DOI: 10.11591/ijai.v11.i3.pp1004-1018.

III. Alexander N. Ndife[1], Wattanapong Rakwichian [2], Paisarn Muneesawang [3] and Yodthong Mensin [4] "Consolidated Artificial Intelligence Method for A Real-Time Energy Management" *International Conference on Robotics and Artificial Intelligence holding in Las Vegas, United States of America 2nd to 3rd December 2021.*

IV. Alexander N. Ndife[1], Wattanapong Rakwichian [2], Paisarn Muneesawang [3] and Yodthong Mensin [4] "IoT Based Smart Home Energy Management System – a centrepiece of optimized real-time control" presented in *6th International Conference on Cloud Computing and Internet of Things (CCIOT 2021),* held in Japan between September 22 - 24, 2021, *http://www.cciot.org/.*

ABREVIATIONS

AR	Autoregressive Model
AI	Artificial Intelligence
ARMA	Autoregressive Moving average
ARIMA	Autoregressive Integrated Moving Average
CNN	Convolutional Neural Networks
ConvLSTM	Convolutional Long Short-Term Memory
DL	Deep Learning
EV	Electric Vehicle
FC-LSTM	Fully Connected-Long Short-Term Memory
GRU	Gated recurrent units
HEMS	Home Energy Management Systems
HPC	Household Power Consumption
ICT	Information and Communications Technology
IoE	Internet of Energy
IoT	Internet of Things
LOCF	Last Observation Carried Forward
LSTM	Long Short-Term Memory
LTLF	Long-Term Load Forecasting
MAD	Mean Absolute Deviation
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MQTT	Message Queuing Telemetry Transport
MTLF	Medium-Term Load Forecasting
ML	Machine Learning
NPL	Natural Language Processing
PCA	Principal Components Analysis
RMSE	Root Mean Square Error
RNN	Recurrent Neutral Networks
SARIMA	Seasonal Autoregressive Integrated Moving Average
STLF	Short-Term Load Forecasting

ABREVIATIONS (CONT.)

STS	Structural Time Series
TinyML	Tiny Machine Learning

CHAPTER I

INTRODUCTION

…“Let me start with a popular quote from Paul Satto “The goal of forecasting is not to predict the future but to tell you what you need to know to take meaningful action in the present”. In this context, times series forecasting as discussed here is about putting both electricity power consumption and its management challenges into a proper perspective beneficial for tomorrow.

1.1 Motivations for this study

This thesis is centered on applying machine technology in terms of Deep Learning in modeling of power consumption behavior of a typical smart home/office for the purposes of forecasting. Proper understanding of energy consumption behaviors and its forecasting has numerous advantages to all the participants in the smart grids, such as manufacturers, renewable energy generators, utility companies, prosumers, and consumers especially. Part of the advantages ranges from tracking the loads relative to proper balancing, real-time energy pricing opportunity, and overall energy management. Meanwhile, in stone age, the only means of estimating power usage is through analysis of previous utility bills.

However, the emergence of machine intelligence has today opened a fantastic vista of opportunity for a real-time energy forecasting. Nowadays, with smart meters’ historical data several activities such as load forecasting, real-time energy pricing, load control, as well as metering information and energy analysis etc. can easily be done. This study however is gravitated towards developing a computational forecast model with high-level accuracy that will be beneficial to both energy users and suppliers alike in terms of uninterruptible service provision, robust economic planning, and energy conservation etc.

As mentioned earlier, a reliable forecast has the potentials of creating a dynamic energy pricing and trading opportunities for users especially in this era of

Internet of Energy (IoE), just with the knowledge of their anticipated power needs. Load forecasting can as well be applied to scheduling of devices [1] and energy storage. Apparently, with latest technologies in energy industry energy trading is gradually becoming the centerpiece of today's energy revolution. However, this type of modeling can be either be based on linear or nonlinear methods but modeling a complex real-world problem like power forecasting with linear models like AR, ARMA, ARIMA, SARIMA etc. it is always difficult and yield less reliable results. Obviously, such types of models cannot determine non-linear relationship in a complex data like that of power consumption, therefore complex models like neural networks are advantageous in this type of problem.

In contrast to statistical model, neural networks models formulate a model based on features learned from existing data and this dependency makes it data-driven, self-adaptive and a chosen bride as far as time series forecasting and Big Data is involved. Neural networks are preferable though not without its own inherent limitations. Importantly, learning of arbitrary complex mapping from inputs to outputs has become research focus recently with significant performance improvement recorded, yet a huge gap still exists between the methods of deployment and implementation environment. Some of these gaps includes: the proper ways to capture dominant factors in the data to be learned, how to reduce the size of the model, and increase its computational speed, and finally how to determine model parameters selection etc. Basically, these are the major areas this proposed forecast model aims to optimize.

1.2 Statement of the Problems

As our society evolve, electricity demands for households and industrial productions purposes continues increasing in line with population growth. New businesses are emerging and there is steady increase in the number of personal electric appliances, and so on. However, the essence of demand forecasting is to create an efficient management and distribution platform for electricity resources which is viewed as temporally and quantitatively finite. Of course, forecasting the future demand of power resources within a distribution network is fundamental to the management of the availability of the limited resources. So, planning of viable

electricity industry and the operation of electric power systems requires an accurate forecast. With accurate forecast, there will reduction in both operational and maintenance costs of such energy industry, increase in reliability of power supply and delivery system, and an opportunity for future expansion.

Time series forecasting task especially for electricity consumption is very tedious for some reasons including multiple scales of time dependencies. For an instance, the load at a given hour is dependent on both the load at the previous hour at the same hour of the last day, and the same hour of the day in the previous week. It is also dependent on other exogenous variables, such as environmental conditions including variations of climate, human social activities etc. There are three major standard of load forecasting: short-term load forecasting (STLF), medium-term load forecasting (MTLF) and long-term load forecasting (LTLF). But this thesis is based on medium-term load forecasting (STLF), specifically 7days-ahead. There are three major standard of load forecasting: short-term load forecasting (STLF), medium-term load forecasting (MTLF) and long-term load forecasting (LTLF). But this thesis is based on medium-term load forecasting (STLF), specifically 7days-ahead.

Since we are interested in multi-step time series forecasting, multivariate input variables are used for a week ahead forecast carried out. Nonetheless, need for demand-side energy management, necessitated emphasis on infrastructures like smart meter and how best it can interact with human user to achieve optimized energy conservation [2]. Interestingly, emergence of smart meter has opened vista of opportunities for energy users to acquire actionable information about their energy usage, varying tariffs etc. for effective costs management. It has almost succeeded in eliminating estimated billing and has enabled dynamic pricing as well as provision of quick monitoring of electrical systems.

Today efforts are being made by industrial players in energy circles to usher in a flexible energy system infrastructure that can allow heterogeneous energy supply to or withdrawal from the grid as a way of guaranteeing a blackout free power generation and distribution. These efforts will be a mere thought if cyber-physical systems and the existing electric grid is not integrated into ‘Internet of Energy’ (IoE) applications in such as a way that it could be implemented within an On-device/embedded systems. Smart meter contributions in integration of cyber-physical

systems in current grid technology can be supplemented with AI in power forecasting in so many ways. Applying smart analytical model with cognitive intelligence capable of learning power consumption pattern/behavior for the purposes of making a forecast; automating meter control processes will help in appreciating the usefulness of smart meter in the global quest for transition to smart homes/cities.

1.3 Research Objectives

This research is geared towards two (2) major objectives:

1. To develop a high-performance dynamic power consumption model, capable of handling a generic data (i.e., Big Data) for the purposes of forecasting 'Total Active Power' and,

2. To build a bidirectional communication platform between IoE devices (i.e., smart meters/smart appliances) and human users for Home Energy Management as a way of integrating cyber-physical systems engineering/technology into smart grid technology.

It is imperative to state that these objectives cuts across both characterization, modeling, and forecasting of times series. The first objective aimed to optimize performance of standard forecasting algorithms and possibly beat the 'state-of-the-art'. While the second objective is centered on addressing the major research gap confronting Home Energy Management Systems (HEMS) like lack of interoperability services for real-time control via emerging smart/IoT based applications in smart cities or homes through a standalone mobile app etc. Honestly, global quest for transition to smart cities and homes will be a mirage if power consumption forecasting is not integrated into on-device systems (especially smartphones, smart devices, PCs, etc.) for seamless and spontaneous data transmission and control.

1.4 Scope of the Study

This research is carried out in School of Renewable Energy and Smart Grid Technology (SGtech), Naresuan University, Thailand due to its proximity to required resources and technical expertise needed for the experimentations. A real-time

experimentations and simulations using SGtech smart grid infrastructures under its Machine Learning Laboratory was performed.

Figure 1 Map view of SGtech, Naresuan University, Phitsanulok

However, the fact that atmospheric climate changes have significant effect on power consumption necessitated the collection of primary data from SGtech to augment the secondary from a household in France. This real-time data was used for validation of the forecast model after all deep learning algorithm used in building the forecast model requires a big data to function optimally.

Literature showed that research efforts has been made in the recent past in terms of developing a forecast model without recourse to real-time interface between human users and cyber-physical systems to guarantee spontaneous interactions. This research therefore focused on developing a lightweight multivariate multi-step time series forecast model with high performance/computational speed for on-device systems. It is important to note that on-device systems are low memory and low powered, hence can easily be used for a seamless human/smart meter interaction

interface. It is equally important to state that the second objective of this study is a continual project due to its capital-intensive nature and need to obtain various forms approvals or hosting the application in the public service domain.

Furthermore, this bidirectional communication path established ensures that at every given time, the real-time data and granular information from the smart meter be compared with values that is generated from this proposed forecast model (used as threshold value) for an effective energy management and automated control. Since the major research gap in this field has been identified as lack of real-time monitoring and control from remote and/or local physical location, this innovative concept will then bridge the gap by interfacing of smart meter with human user in a partial automation platform. The data transmission process between the smart meter and human user via on-device system for monitoring and control involved a minute sending of the granular data with the help of microcontroller through an IoT gateway. This accessibility and spontaneous control ability of the proposed system will help to improve the expected user convenience as well as optimize the computing resources usage. Machine-to-machine communication protocol called Message Queuing Telemetry Transport (MQTT) was used to publish/subscribe the 'total active power' data from smart meter to user interface (smart phone, web page etc.) via a cloud server. It is expected that IoT switch connected to the smart meter or other home appliances as the case maybe performs effectively ON/OFF functions using forecast result as threshold for its operation when need arises.

1.5 Contributions to Knowledge

This multidisciplinary PhD thesis seeks to make times series forecasting especially as it relates to electricity at homes and offices in such a way that it will be readily available in a very smart way. The contributions cuts across data science field in terms of big data analysis for the purpose of modeling power consumption behavior, to application of artificial intelligence in learning the data representations and finally to a partial automation implemented using a an embedded IoT device.

Data Science – big data-based research is one of the hottest research areas in recent time because of timely need for the utilization of such data stream for the growth of organizations. Almost on daily basis, large volume of data (both structured

and unstructured) inundates business. So, the important question is ‘what does the organizations do with those data’. Big data can be collected from a variety of sources, including business transactions, smart (IoT) devices, industrial equipment, videos, social media etc., and can be analyzed for insights leading to better decisions and strategic business moves. This thesis collated power consumption data through both primary and secondary means from the smart meters located in two different continents (Europe and Asia). Both primary and secondary data was collated in a real-time and scientific means, the only difference is location and secondly the primary data is collated from an automated office while the secondary is from a manually operated home in France.

Machine Intelligence: So many important models have been proposed in the past to improve the accuracy and efficiency of time series modeling and forecasting. The results so far showed that machine intelligence through machine and deep learning technologies have outperformed other methods deployed in forecasting, classification, and regression tasks. Therefore, this thesis did not only consider the ways to improve the accuracy of a time series forecast model but also seek for a robust way to make such models implementable is low-power and memory storage devices like on-device systems. So, one of the most significant contributions of this thesis is developing a miniature forecast model. Modeling the electricity users’ behavior is viewed from an automated point of view, to be able to get a true reflection the effect of the relationship between human users and their environment on power consumption. For an instance, when someone comes into a room or office with temperature above room temperature, an air conditioner is expected to ON once movement or motion is sensed otherwise it remain OFF.

IoT Device: The Internet of Things (IoT) is the network of physical devices, vehicles, home appliances and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these objects to connect and exchange data. It is characterized by real world and smart things, limited capacity, constrained devices etc. Interoperability of the existing Internet infrastructure and each uniquely identifiable system for the purposes of monitoring and control made this aspect of the thesis contribution a great one. It opened wider route for adequate

integration of cyber-physical systems into smart grid technology to deal with real world things in a more distributed and dynamic manner.

1.6 Thesis Organization

The first chapter of this thesis acts as the groundbreaking to the main content of the thesis. It started with motivations behind this research, presented the research problems and the defined objectives set to achieve including the research question. It also discussed the scope of the study, research methodology adopted towards achieving the set-out objectives and went further to state the contributions to body of knowledge. Finally, it gave a step-by-step guide to the rest of the content of the thesis.

The second chapter centered on the existing times series models and its relation to the research in this thesis. The literature in the following areas is reviewed: deep learning neural networks and their appropriateness in modeling of a stochastic system, encoder-decoder feature learning method, handling of inherent challenges of neural networks models, model improvement strategies and cyber-physical systems integration to smart grid technology.

Chapter three provided the historical background to artificial intelligence in forecasting and went further to discuss deep learning (DL) networks methods application to supervised learning. It also delved into the choice of different deep learning algorithms and showed how they can be used to provide an answer to the research question of this thesis.

Chapter four is about the modeling of power consumption characteristics. It discussed time series modeling as it relates to multi-step forecasting and detailed the use of neural networks method in modeling power consumption, choices of sizable dataset and how it was augmented so it could be used to model and even forecast the total active power for each day over the next seven days of future electricity consumption.

Chapter five is centered on the using the result of power consumption characteristics in making a forecast. It discussed how the forecasting problem was framed for multi-step forecasting. It also showed the feature extraction process, validation method used and how challenges of overfitting and underfitting during model training preparatory to forecasting is handled and finally provided information about the model evaluation and performance measurement strategies in this thesis.

Chapter six is about the results and its detailed discussions, and how the forecast result can be applied smartly in-home energy management system to achieve the second objective of this research.

Finally, the chapter seven which is the conclusion therefore summarizes the thesis and highlights the contributions made. Potential ideas for further work in the field are made, including the application of machine learning algorithmic improvement and expanding human machine interface web app to offline functionality.

CHAPTER II

BACKGROUND AND LITERATURE REVIEW

This section reviews some of the existing methodologies that were generally applied in modeling neural networks for different tasks such as classification and regression etc. Because of the stochastic nature of neural networks, it has some inherent problems. Therefore, this section peculiarity of the problems, methods applied, and other efforts towards containing the resultant effects and finally the aspect of cyber-physical integration of smart grid.

2.1 Determining the appropriate model

The first step towards modeling a time series problem is determining the appropriate model based on the peculiarities of the problem. Modeling a complex real-world problem like power forecasting with linear models like AR, ARMA, ARIMA, SARIMA etc. will surely yield a poor result. Obviously, these types of models cannot determine non-linear relationship in a complex data [3] like that of power consumption, therefore complex models like neural networks have a leverage. In contrast to statistical model, neural networks models formulate a model based on features learned from existing data and this dependency makes it data-driven, self-adaptive and a chosen bride as far as time series forecasting and Big Data is involved. Neural networks are preferable though not without its own limitations.

2.1.1 Deep Neural Networks Models

Deep Learning is a subset of Machine Learning within the big family of Artificial Intelligence. It that mimics human neurons and has since became a powerful tool for representation learning. Most of the early representation learning ideas revolve around linear models such as Principal Components Analysis (PCA), factor analysis, or sparse coding. In order to achieve state-of-art power consumption forecast model, numerous deep learning techniques [4-6] had been deployed in the processes of problem formulation, data transformation to supervised learning, and neural networks architecture building. [7] used feature encoding method on a deep reinforcement learning algorithm as shown in fig. 2 and 3, for inputting of data into

the neural network layers. Nonlinear transformation defined by this encoder and decoder method can be viewed as an advance feature extractor capable of preserving the hidden abstractions and invariant structures in input.

2.1.2 Encoding-Decoding

Encoding-Decoding process is specialized sequence prediction method that reads input sequence and encode it i.e., mapping the variable-length source sequence into a fixed-length, and decoding the fixed-length vector and outputting the predicted sequence i.e., mapping the vector representation back to a variable length target sequence. Basically, the idea of the proposed model was for Convolutional Neural Networks (CNN) to interpret the sequences of input provided to a Long Short-Term Memory (LSTM) network to process (i.e., learned) while LSTM processes the sequential input data of arbitrary length of inputs, and stores the information in the memory using feedback connection. The introduction of CNN onto the proposed recurrent architecture was due to the ignorance of LSTM to spatial connectivity in 3D data. Instead of having a fully connected layer as in the case of FC-LSTM, convolution layers were connected to achieve reduction of the input data feature to a small-learned feature that can be reconstructed to its original input by the second network.

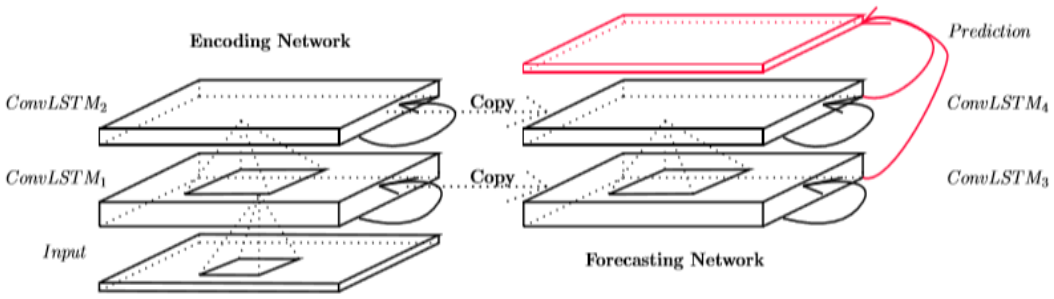


Figure 2 Encoding forecasting using ConvLSTM network

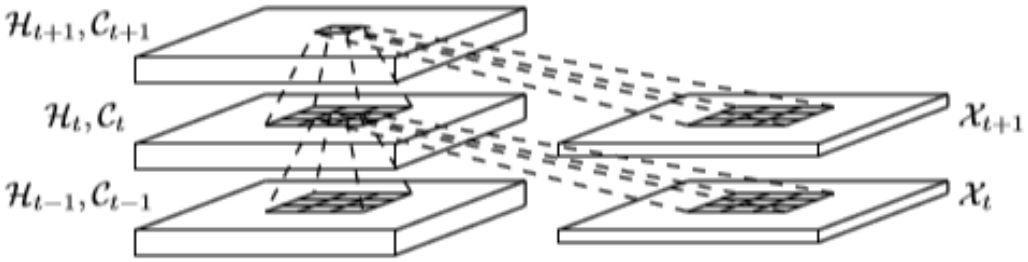


Figure 3 Inner structure of the ConvLSTM showing state-to-state and input-to-state transitions of feature extraction

2.1.3 Neural Networks Inherent Problems

One of the fundamental challenges in machine learning is determination of whether the model truly or accurately generalized, and the only sure test of this is its performance over unseen data [8]. The stochastic nature of neural networks problems made it susceptible to the following shortcomings:

1. Randomness in the output
2. Overfitting & underfitting
3. Sensitivity to statistical noise esp. in training data which in turn results to…
4. Generalization Error

However, well-articulated methods/techniques like model averaging (ensemble method), dropout, exponential decay etc. was adopted at different stages of development and deployment of this proposed model to countervail some of these neural networks' inherent problems.

2.2 Cyber-physical Systems Engineering into Smart Grid Technology.

Advent of smart grid technology has made integration of Internet of all Things (IoT) and Big Data a research focus, enabling the rapid growth of smart and self-configurable intelligent devices. These smart devices are often used as an energy optimization measures and sometimes movable, thereby prompting objective (I) towards 4 distinctive focuses: lightweight, high computational speed for prediction, less error and adaptive/deterministic. However, to successfully set up a communication interface for monitoring and control in attainment of objective (II),

the HEMS design architectures should involve components such as a smart meter, IoT switch containing microcontroller to transmit data, WIFI connectivity, IoT gateway, cloud server, mobile app, and user interface (smartphone or webpage).

In this era of smart grid technology, it is expected that energy packets are managed in such a similar way as data packets; transiting routers and gateways autonomously, deciding the best pathway to its destination based on integrity level [ref journal paper] hence justification for the proposed objective (II). Interestingly, with the augmentation of IoT technology with sensors and actuators, it becomes an instance of the more general class of cyber-physical system, which also encompasses technologies such as smart grids, virtual power plants, smart homes and smart cities [9]. This indicator shows that smart grid technology in power generation, transmission and distribution has become the engine of innovation for tomorrow's smart society, so there is need to design, implement and deploy more flexible methods in its system infrastructures to guarantee convenience, cost effectiveness and blackout free society.

In the proposed integration of cyber-physical system into smart grid, a Home Energy Management System (HEMS) is proposed. This system will not only help in proper budgeting/planning but will also play a vital role in prudent energy management (most often when they are outside home). This proposed HEMS can be operated from any location of the world without restriction to a limited coverage, complimentarily with Advanced Metering Infrastructure enablement for demand response controls. With the world intelligence and cutting-edge technology movement towards edge devices driven by AI, this thesis leveraged on this technological revolution (exemplified with high computational powers and sensory abilities) in achieving the objectives of this study for an increased efficiency of energy usage.

CHAPTER III

RESEARCH METHOD

In this research, a supervised learning approach using deep learning technology (Conv-LSTM encoder-decoder network) is proposed to build an end-to-end trainable model for multivariate multi time steps power consumption forecast. This architecture has 2 sub-models: one for reading the input sequence and encoding it into a fixed-length vector, and the other for decoding a fixed-length vector and outputting the predicted sequence. Quest for accurate and reliable forecast models has made training of 'Big Data' expedient to achieve a state-of-art model, adaptive to the dynamics of power consumption. In view of this, a good number of research efforts like [7] started by formulating complex regression problem as a spatiotemporal and time-frequency sequences forecasting problem in which both the input and the prediction target are viewed as spatiotemporal time-frequency transform sequences respectively. Various methods have been adopted in this regard using neural networks in extraction of features from input data and network training. Convolutional layers relatively inspire automatic temporal feature learning, and capacity to convolve input to reduce size and output multi-step time vector directly, while LSTM learn from past memory with its feedback potentiality.

3.1 Artificial Neural Networks

Artificial Neural Networks (ANN) are one of the intelligence tools that mimics human brain, and its basic algorithms and simplified methods is used in Deep Learning (DL). There are a lot of complicated and high-end models that leverage on Deep Learning approach both for regression and classification problems. ANN is specifically established to perform human brain activity, therefore it understood humans' neural system as a way of transmitting information via neurons in the brain. Human brain is housing millions of neurons interconnected with each other for different activities. From the neural anatomy of fig 4, the major function of a neuron is to receive and transmit impulses/information in various parts of our own body.

Neurons contain some elements such as dendrites which are regarded as a signal receiver and Axon as a signal transmitter.

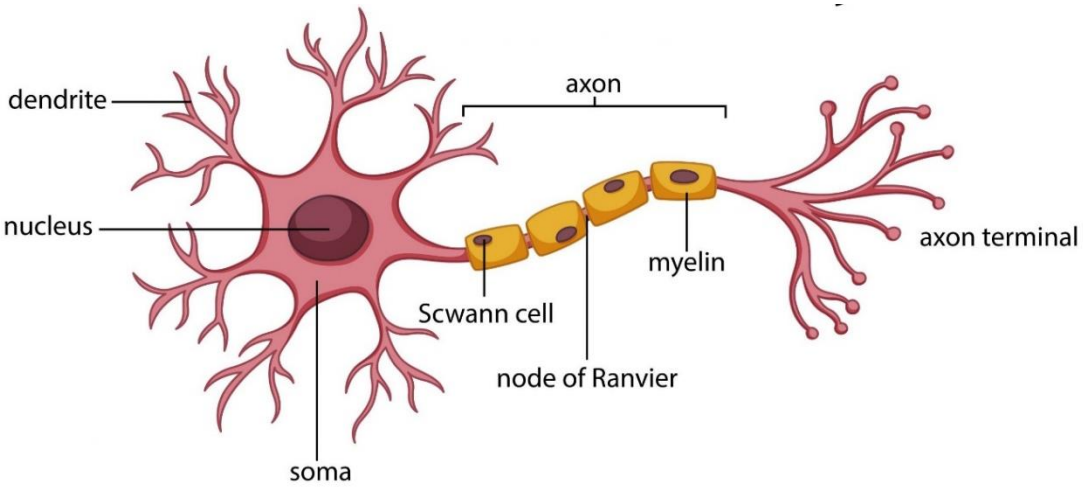


Figure 4 Human Neural Anatomy

Neural networks are basically the technological approach in the graphical representation of the biological notions of how neurons receive and transmits signals to and from one another. In neural networks, learning is done through adjustment of weights. In the process of learning the weights are modified and neural networks then decides which signal is adaptable in each case and which signal is not relevant for neurons, or which signal is passed, or which signal is not passed or to which intensity signal is passed.

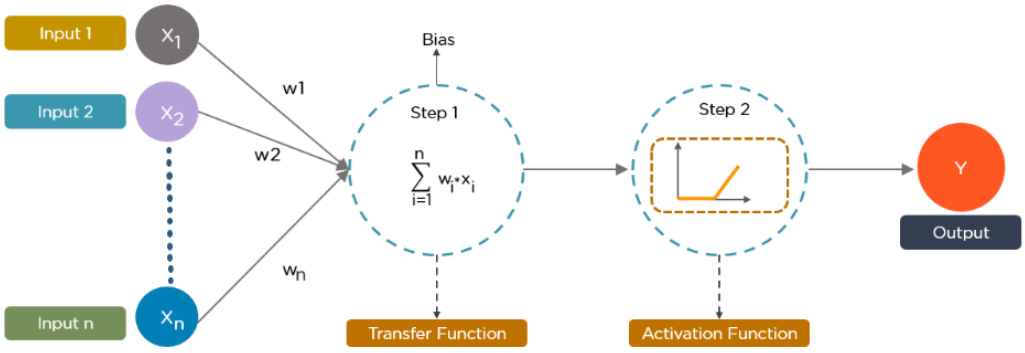


Figure 5 Artificial Neural Network

Image Source: superdatascience.com

3.2 Deep Learning Networks

Deep learning is an alternative to machine learning in terms of complex problems, having proved its worth in image classification and recognition tasks involving more than sequential data processing [10]. Deep learning and machine learning are subsets of artificial intelligence that deals with large, complex datasets (such as a power consumption dataset and image dataset with millions of images, each consisting of tens of thousands of pixels). They discover rules to execute a data-processing task, given examples of what is expected. As a means of consolidating the performance accuracy in deep learning, various techniques have been introduced including dropout for model regularization. [11] is the first regularization method that uses the stochastic model averaging technique to improve the performance of deep learning models.

This supervised learning approach proposed in this thesis seeks to clearly understand the power consumption behaviors of a typical household as a case study by utilizing the knowledge learnt from the previous power consumptions (historical data) to perform complex computational tasks that hitherto was difficult or almost impossible for human brain to make a reliable forecast. Deep learning is very popular in recent times because of the way it simulates the ways densely interconnected networks of neurons interact in the brain to perform specialized tasks better than the human brain [4]. It can learn a near infinite number of mapping functions which endeared it to large and complex neural networks problems. Deep learning algorithms are stacked in a hierarchy of increasing complexity and abstraction unlike the traditional machine learning algorithms which are linear. One of the major problems of deep learning models is that they are highly flexible nonlinear algorithms which often encounter high variance and generalization errors due to their stochastic nature. High variance in neural networks is mostly occasioned by noise and randomness in the learning algorithm. The more the bias in the learning algorithm the less the variance and vice versa, hence, need for trade-off via the hyperparameters tuning.

However, a technique referred to as ensemble method was adopted in addressing the problem of randomness in the model's output. It is noteworthy that deep Learning algorithms often require a larger amount of training data and processing power than machine learning [12]; the bigger the data samples to be

computed the better the model's performance. Deep learning common application is image recognition, natural language processing (NPL), speech recognition and several regression problems. It has also been applied in diverse big data analytics applications such as self-driving cars, medical diagnosis, stock market trading analysis, network security and language translation services.

3.2.1 Convolutional Neural Networks (CNN)

Convolutional neural networks are a class of deep learning algorithm that is mostly applied to image classification tasks. Its configuration has a shared-weight architecture of the convolution kernels or filters that slide along input features and provide exact representation known as feature maps. In convolutional Neural Networks, the central building block is at the convolutional operator. This operator enables networks to construct informative features by fusing both spatial and channel-wise information extracted from prior knowledge of the data within local receptive fields at each layer. Convolutional neural network learns to optimize the filters (or kernels) through automated learning, whereas in other traditional algorithms these filters are tuned. CNN input is a tensor with a shape: (number of inputs) x (input height) x (input width) x (input channels). This shape is passed through convolutional layers to extract the feature map, also called an activation map, with shape: (number of inputs) x (feature map height) x (feature map width) x (feature map channels).

Looking at fig… it will be noticed that input data is handled as image data of RGB color space and the image size even though the data is 1D (i.e., 1 channel type data). The CNN utilizes an image width of 28 pixels and height of 28 pixels for one image at [28 × 28 × 1], where 1 is the number of color channel. Convolution layer operates for the multiplication of each pixel with filter coefficients. It initializes operation at location (0,0) of the data and moves by one pixel (stride 1) at a time from left to right and top to bottom until all pixels are completed. This is exactly how the creation of an activation map is done in convolutional layer. For an instance, if the size of the image data is [224 × 224 × 3] with a total of 96 filters, each of which has a size of 3×3, the resulting activation map will be [111 × 111 × 3] when the filter moves by two pixels (stride 2) each time.

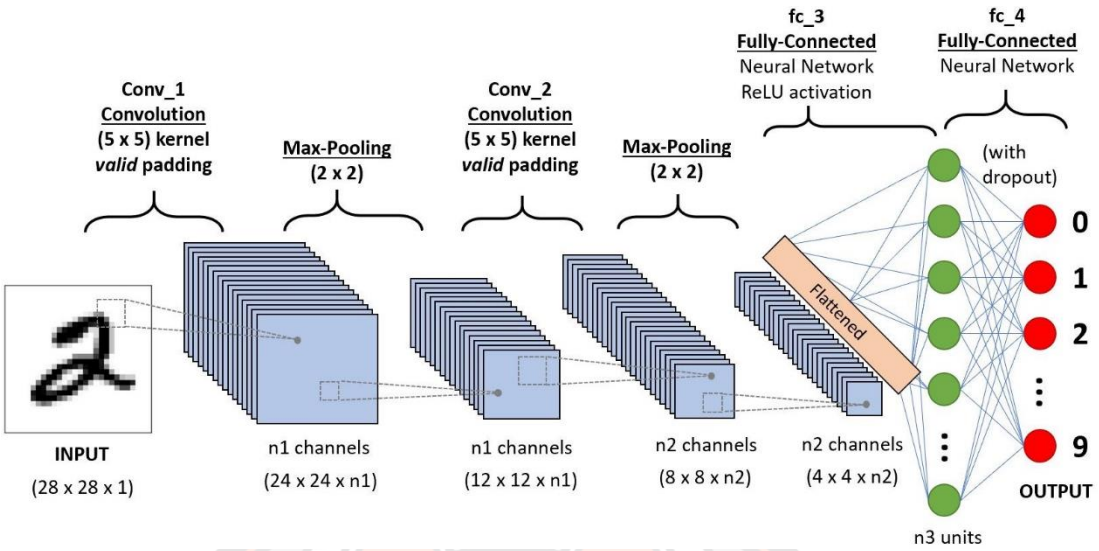


Figure 6 Convolutional Process in Neural Networks

Image Source: towardsdatascience.com

3.2.2 Long Short-Term Memory (LSTM)

Long short-term memory is a family of recurrent neural networks that can automatically learn features from sequence of data. It supports multiple-variate time series data and can output a variable length sequences that can be used for multi-step forecasting. LSTM likewise GRU has internal mechanisms called gates that regulates the flow of information.

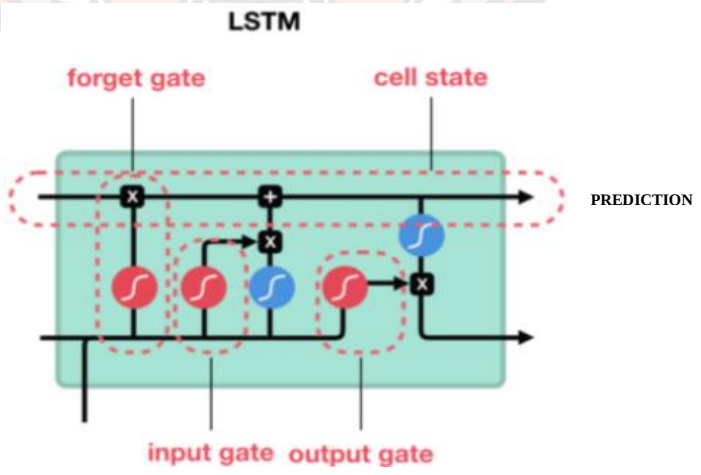


Figure 7 LSTM Architecture

Source: towardsdatascience.com

LSTM emergence is purposely to address the short-term memory and vanishing gradient problems suffered by Recurrent Neutral Networks (RNN) during backpropagation. During networks training in RNN, the gradient shrinks as it back propagates through time and once the gradient value becomes extremely small it contributes little to learning. Meanwhile, these gradients are the values used in updating neural networks weights during learning. Layers that get small gradient updates stops learning with time, making the network to forget what it seen in longer sequences, thus having a short-term memory. So, both LSTM and GRU were created to proffer solution to short-term memory problem. Core LSTM architecture comprises a cell state, and various gates. During learning, these gates as shown in fig…. determines which data in the sequence that is important to keep or throw away.

In LSTM, cell state transports relative information throughout the processing of the sequence. Even information from the earlier time steps can be made available in later time steps, thereby reducing the effects of short-term memory. As the cell state carries out this information transport process, information gets added or removed to the cell state via gates. The gates decide which information is allowed on the cell state.

3.2.3 Gated recurrent units (GRU)

Gated recurrent units (GRU) is the newer generation of Recurrent Neural Networks with gates like LSTM. The difference between both is that GRU has fewer parameters than LSTM, as it lacks an output gate. Due to fewer parameters, its training is a little speedier than LSTM. GRU has been used in tasks like polyphonic music modeling, natural language processing and speech signal modeling. Though it does not have forget gate like LSTM, but its update gate acts like the forget and input gates. The update gate decides what information to throw away and what new information to add. On the hand, reset gate decides how much past information to forget.

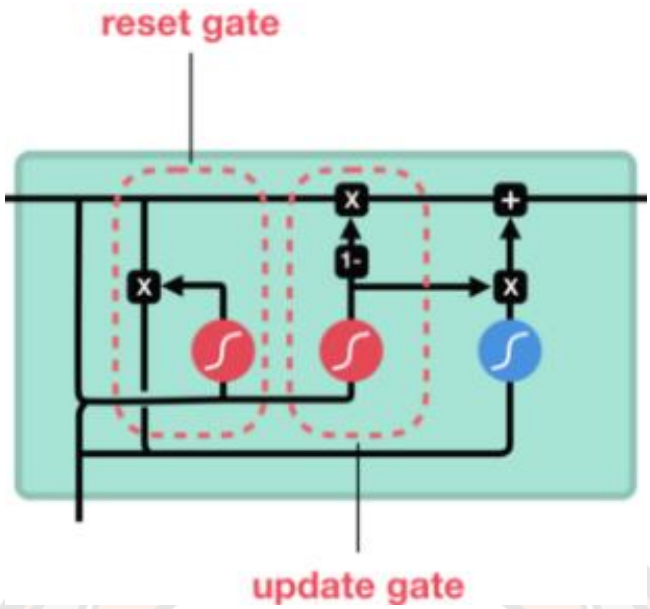


Figure 8 Gate Recurrent Units Architecture

Source: towardsdatascience.com

3.3 Dynamics of Power Consumption

"One of the basic concepts of the smart grid is the integration of information and communications technologies (ICT) into the power system to make it more cost effective, efficient, reliable, cleaner and, provide customers with actionable information about their energy use so they can control their costs" (George W. Arnold, 2011 cited by [1]). However, the resultant exponential growth of this technology (with intelligent and flexible electrical grid; capable of reacting to power fluctuations) has made energy management an interesting research area due to its insight in energy consumption of individual loads i.e., on a per-customer level. Therefore, for proper load balancing, research focus is now directed towards proper understanding of energy consumption behaviors; to keep track on consumers loads. Power load forecasting is oftentimes classified into short, medium, or long time depending on the choice and objectives of the forecast.

In this context, electricity demand behavior at any given instances is perceived to be dynamic due to peculiar nonlinearity nature, often difficult for simple statistical analytical tools. This behavior is determined mostly by either statistical or meteorological factors. However, accurate data on load demand will help in making the amount of energy delivered per unit generated to be easily managed (supply/load

distribution) since supply chain is usually demand driven. This will in turn reduce the fuel needs and carbon emissions in the part of energy suppliers. Interestingly, energy generation follows the time variant consumption, and one prerequisite of grid stability has always been the balance between energy consumption and generation.

Consequently, this research is focused on learning the real-time load data from demand side for the purposes of making projections. This load profile will be of enormous benefits to both energy users and suppliers alike including providing consumers the incentive to determine which appliances to be used at every given time based on its power consumption capacity (for energy conservation), and when to use them considering the varying tariffs over time (as a cost-effective measure in compliance with demand-response model). More so, knowledge of real-time power usage can as well create a dynamic energy pricing and trading opportunities for users. Today energy trading is on the front burner in energy market as far as smart grid technology is concerned. Having knowledge of the amount of energy expected to be consumed within a stipulated timeframe, no doubt will also help consumers to adjust their usage habit as a cost saving measure as well as in decision making as per whether to sell off the excess energy they have or store for future use. Such consumption profile will as well help in scheduling residential energy storage device, maintenance and as well as provide a guide for effective implementation of energy policies.

Unfortunately, this data is not readily available, and where it exists it is being solely determined by suppliers posing a difficult situation for consumers especially in making projections for proper budgeting/planning, and energy management. Therefore, the research questions for this thesis are:

1. ***Given the current power consumption, what is the expectations for the week ahead?***
2. ***Can energy usage at home/office be monitored and controlled on a real-time from anywhere in the world?***

Addressing these research questions necessitated modeling power usage behavior using an intelligence system and human machine interface for energy management. It requires a forecast model applicable in demand-side energy management and control.

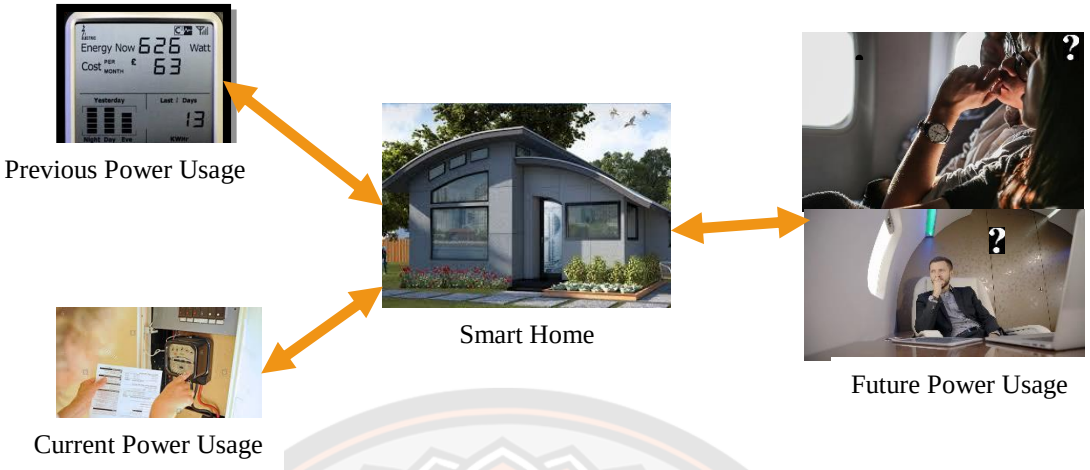


Figure 9 Schematics of a smart home power consumption problem

CHAPTER IV

MODELLING

Modeling in machine learning concept is training a primary or secondary data to recognize certain types of patterns in that data. A model can be trained over a set of data, by providing it an algorithm required to reason over and learn from the data. Once a model is trained with a specified data, it uses the knowledge it learned from the data to make predictions in the future. A good example is when build an application that can recognize a user's emotions based on their facial expressions. All that is required is to train such model with images of faces that are each tagged with a certain emotion, after the training process the next time, such model is used in the application for the purposes of recognize any user's emotion it will perform the task conveniently.

The essence of modeling power consumption behaviors of households or offices is to utilize the output in estimating the future energy needs and effect proper management strategies that can guarantee energy conservation. Modeling power consumption of a smart home is very challenging because of its stochastic nature and non-linear relations over time. Given a sequence by sequence nature of multivariate dataset used in this model, where an input sequence time series signal **X** = $(x_1, x_2, \cdots, x_T)$ with $x_t \in$ Rn, and n is the variable dimension. It is aim at predicting corresponding outputs **y** = $(y_1, y_2, \cdots, y_h)$ at each time. The target of this type of sequential modeling network is basically to obtain a nonlinear mapping to the prediction sequence from the current state as:

$$(y_2, \cdots, y_h) = f(x_1, x_2, \cdots, x_T) \quad (1)$$

And considering the neural network and its weights, the distinct forecast output gives:

$$y_i = f \sum_{i=1}^{n} w_i x_{1i} + b_i \quad (2)$$

Where x_i is the input to the neuron, w_i is the weight of the network, b_i is the bias in the network, $f()$ is the nonlinear function, while y_i is the output.

However, during data preprocessing stage, we noticed non-stationarity and seasonality due to spatiotemporal nature of power consumption dataset. This prompted the decision to apply different approach in modeling power consumption behavior for reliable forecasting. Statistical methods and neural network combinations [13-16] have been applied lately in regression problems of this nature with good results. Ordinarily, a stochastic method approach would have been the easiest particularly for power consumption forecasting if not for its error and inflexibility. [16-18] implemented different types of neural networks for time series problems. Because this study is interested in predicting a week ahead horizon, therefore the previous 7days power demand was used as the input vector x_i, and the next 7-steps ahead as y_i in the adaptive algorithm. The implementation of this forecast method is on deep learning encoder-decoder network. Dropout that has been a common technique in model regularization was used to block out random set of unit cells during model training. Eqn. 3 expressed the way this proposed model accepts multivariant times series input variables and output 7 distinct forecast ahead. The input parameters are the previously observed data at the scale times (t + p-1, t + p-2, …., t). Therefore, the answer to finding the relationship between the input and output data (for the purpose of predicting the future data at the time (t + p) is lies in nonlinear functional mapping from the past observation of the time series to the future value i.e., eqn. 3 using eqn.4.

$$y_t = \tilde{f}(y_{t-1}, y_{t-2}, \ldots, y_{t-p}, W) + \varepsilon_t \quad (3)$$

Where *w* is a vector of all parameters and *f* is function determined by network structure and connection weights.

Using a simple feed-forward neural networks architecture with 3-layers for example, the output of the model can be computed as:

$$y_i = \alpha_0 + \sum_{j=1}^{q} \alpha_j \, g\left(\beta_{0j} + \sum_{i=1}^{p} \beta_{ij} \; y_{t-1}\right) + \varepsilon_t, \forall \quad (4)$$

At the instances $y_{t-i}(i = 1, 2, 3, \ldots, p)$ are the p inputs and y_i is the output. P, q are the integers and the number of input and hidden nodes respectively, while $\alpha_j(j = 0, 1, 2, \ldots, q)$ and $\beta_{ij}(i = 0, 1, 2, \ldots, p; j = 0, 1, 2, .., q)$ are the connection weights. ε_t is the random shock, α_0 and β_{0j} are the bias terms. For activation of this type of

model, nonlinear activation function such as logistic sigmoid function or others like linear, Gaussian, hyperbolic tangent etc. can be used. However, the estimation of the connection weights as a measure to minimizing the error function in this network can be done using nonlinear least square method of eqn. 5.

$$F(\Psi) = \sum_{t} \mathrm{e_t}^{\,2} = \sum_{t} (y_t - \hat{y}_t)^2$$

This optimization technique for error minimization of eqn5. has Ψ as the space of all connection weights.

4.1 Predictive Time Series Modeling

In time series forecasting, series of strategies are normally tested and at the end a more reliable one with high level of consistency is chosen. Usually, predictive modeling like neural networks model proposed in this thesis involves the use of a historical data to make a prediction in an unseen data. It uses algorithm to find the best mapping function from the input variables (x) to discrete output variables (y) given the time and resources available to it. They are associated with machine learning, data mining and pattern recognition etc. which forms an integral piece of predictive modeling process. In practice, time series chronical ordered observations x_t at a specific time t and constructs a model a model of it as target and fit it on the observations. Time series can be discrete or continuous depending on if the observations were recorded continuously over time interval or not. Time series forecasting using neural networks is often expressed mathematically as approximating a mapping function (f) from input variable (x) to output variables (y) i.e., f_θ: $x \rightarrow y$ such that f predicts new or unseen data(x_i, y_i) whose behavior is parameterized by θ . Basically, the model is to generate a computational procedure that implements the function. These models have the capability to learn the regularities and patterns in the input data to use same to provide a generalized result based on known previous knowledge. This idea of dependency on formulating a model based on features learned from existing data made neural networks data-driven and self-adaptive. Some research like [19] utilized time-frequency feature representation analysis with neural network to capture dominant factors that was claimed to purportedly affects the pattern of electricity consumptions.

4.1.1 Modeling using Neural Networks

Consequence upon nonlinear time series pattern of power consumption behaviors as earlier stated, quest for real-time consumption data on per-customer level or loads has become the last straw that breaks the camel's back in energy load estimation forecasting. Time series forecasting can be achieved either using linear or nonlinear methods. Traditionally, linear methods are in dominance because of its simplicity. But considering the complex nature of time series forecasting problems these days (like power consumption forecasting), neural network approach (especially deep learning) has shown to outperform simple and classical models [18] in view of the fact that it can learn arbitrary complex mapping from inputs to outputs, as well as support multiple inputs and outputs. Obviously, time series forecasting is basically to discover changing patterns in time series data by analyzing both historical and current data, then establish model for prediction. The fact that this idea is pertinent to achieving the objectives of this research prompted the proposed ConvLSTM model.

4.1.2 Power Consumption Dependencies

Uniqueness of the characteristics of electricity that made it to be provided as soon it is demanded makes it dependent on both temporal and spatial factors unlike material products. Some of these time dependency factors include:

Weather: weather condition is a very important variable to be considered in electricity load forecasting. Temperature for an instance is a factor that strongly influences the load among other meteorological factors like humidity, rainfall, wind, cloud cover, thunderstorm and so forth. Hence, time series of the temperature can be considered as exogenous input of the STLF and MTLF models.

Calendar: electricity consumption also has a close relationship to human behavior, like calendar events (holidays, family social events, festival days and so on). These events can cause uncommon load of electricity. Therefore, they should be put into consideration during the modeling to improve the model accuracy. Accurate forecasting of trips during special events can increase efficiency of driver allocation, resulting in a decrease of waiting time for the riders.

Other Factors: factor such as geographical locations, human comfortable temperature, heating/cooling technology, type of consumers or purpose of electricity use (industrial or residential) also affects a load of electricity

consumption, for example, and so on. These various factors make electricity load patterns become more complex and irregular, that impedes the accuracy of the forecast.

Ultimately, objective 2 serves as a Home Energy Management System (HEMS) with distinct hybrid functions. Firstly, the predictive model will model the power consumption behavior of a home; leveraging on load profile to make a week ahead forecast of 'Total Active Power', and lastly on-the-spot interaction between human users and the smart meter and other smart appliances at home is established via IoT for the purposes of real-time control. The data communication structure to achieving the second objective as regards smart grid technology and IoT applications is depicted by fig. 45. As earlier stated, HEMS currently in use in most homes is limited by coverage (i.e., they only operate within line-of-sight), hence, need for a more robust system that can be operated outside home without coverage restrictions and implementable in on-device systems.

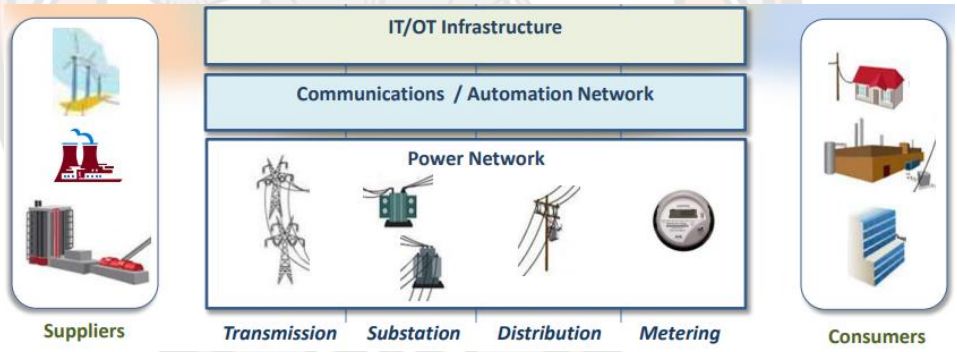


Figure 10 General drivers for smart grid

Source: https://www.engerati.com/sites/default/files/Andy%20Bae.pdf

4.2 Dataset

Based on the preliminary analysis carried out on various power consumption datasets to determine the most appropriate for the research problem, Household Power Consumption (HPC) dataset from MIT repository was selected as earlier stated. This is followed by data cleaning process (identifying and repairing issues like data corruption, error, or loss) using imputation method. Next stage is data preparation

process (transformation of time series data into a supervised machine learning format), where data shape and structure were changed making it suitable for chosen neural network algorithm.

4.2.1 Household Power Consumption (HPC) Dataset

This research leverage on secondary data (individual household electricity power consumption dataset) [20] that was augumented with remote-sensing data acquired from different SGtech buildings. This secondary dataset is a multivariate time series dataset containing 2, 075, 259 measurements gathered in a house located in Sceaux (7km of Paris, France) between December 2006 and November 2010 (47 months). The observations were made every minute in a sequential manner over the period of 4years though down sampled to hourly and subsequently daily, given the total power per day that is of interest here. The original data which is in time domain captured the consumption behavior over the earlier stated period irrespective of season of the years or weather conditions. The original dataset comprises of seven (7) independent variables described below though an addition was made in the design:

1. Global Active Power: the total active power consumed by individual loads measured in kilowatts.
2. Voltage: the average voltage measured in volts.
3. Global intensity: average current intensity (amps).
4. Global Reactive Power: the total reactive power (i.e., power resulting from inductive and capacitive loads) consumed by the household measured in Volt-Amps-Reactive (VAR).
5. Sub-metering 1: the active energy accruable to kitchen appliances (watt-hours of active energy).
6. Sub-metering 2: the active energy accruable to laundry (watt-hours of active energy).
7. Sub-metering 3: the active energy for climate control systems (watt-hours of active energy).
8. Sub-metering 4: active energy for EV charging (watt-hours of active energy).

Although other factors like weather, time, holidays, lagged load and load distribution in different time periods has been reported in [19] to be the most influential to

electricity consumption on a daily basis, further investigation by empirical tests and analysis will be carried out to justify the claims especially the meteorological factors. Meanwhile, a week (7days) ahead forecast is expected to be carried out in this research using range of previous day's data.

A B

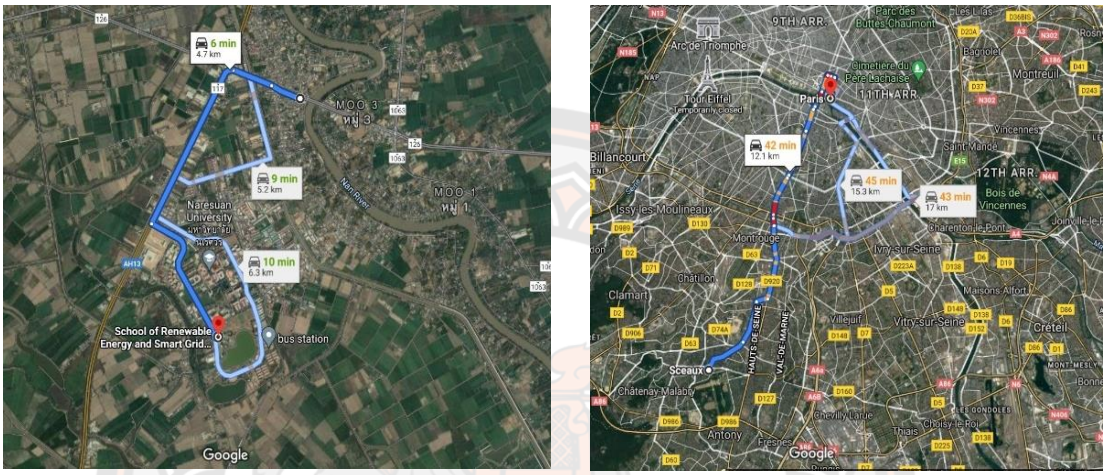


Figure 11 Data Collection Centers (A-SGtech Naresuan Uni., B-Sceaux France)

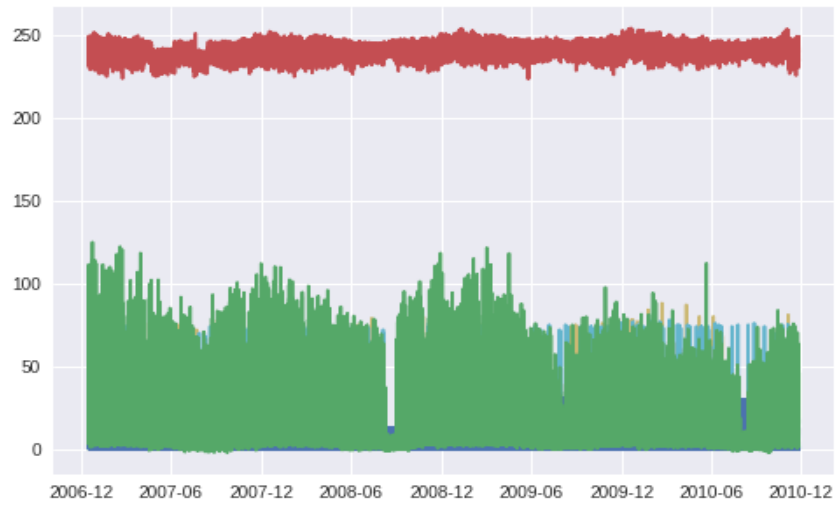


Figure 12 Household Power Consumption Data Structure (covering Dec. 2006 – Nov. 2010)

dt	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	Sub_metering_2	Sub_metering_3
2006-12-16 17:24:00	4.216	0.418	234.84	18.4	0.0	1.0	17.0
2006-12-16 17:25:00	5.360	0.436	233.63	23.0	0.0	1.0	16.0
2006-12-16 17:26:00	5.374	0.498	233.29	23.0	0.0	2.0	17.0
2006-12-16 17:27:00	5.388	0.502	233.74	23.0	0.0	1.0	17.0
2006-12-16 17:28:00	3.666	0.528	235.68	15.8	0.0	1.0	17.0

Figure 13 Data Structure of the secondary data from France showing 7 input variable

	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	Sub_metering_2	Sub_metering_3
count	2.075259e+06	2.075259e+06	2.075259e+06	2.075259e+06	2.075259e+06	2.075259e+06	2.075259e+06
mean	1.091615e+00	1.237145e-01	2.408399e+02	4.627759e+00	1.121923e+00	1.298520e+00	6.458447e+00
std	1.050655e+00	1.120142e-01	3.219643e+00	4.416490e+00	6.114397e+00	5.785470e+00	8.384178e+00
min	7.600000e-02	0.000000e+00	2.232000e+02	2.000000e-01	0.000000e+00	0.000000e+00	0.000000e+00
25%	3.100000e-01	4.800000e-02	2.390200e+02	1.400000e+00	0.000000e+00	0.000000e+00	0.000000e+00
50%	6.300000e-01	1.020000e-01	2.409600e+02	2.800000e+00	0.000000e+00	0.000000e+00	1.000000e+00
75%	1.520000e+00	1.920000e-01	2.428600e+02	6.400000e+00	0.000000e+00	1.000000e+00	1.700000e+01
max	1.112200e+01	1.390000e+00	2.541500e+02	4.840000e+01	8.800000e+01	8.000000e+01	3.100000e+01

Figure 14 Data Statistics of the secondary data

4.2.2 Dataset Augmentation

The data used in this study was augmented with a sensor-based data collated through an automated process. In this way, the manual and conventional method of putting ON a lightning bulb when it gets dark for an instance was eliminated. In a typical automated home/office it is expected that smart switch switches the bulb ON as it senses darkness and OFF when there is none. This also goes to motion sensor that senses the presence of home occupant as he or she moves into the home/office, whereby temperature and humidity sensor determines the coldness or hotness of the room temp for the purposes of switching ON the air-conditioner etc. With this procedure, the reliability of the load profile data will be significantly improved on, reflecting the true power consumption behavior in a smart environment.

	Day	Time	Power_Consumption	Power of PV	Energy_Imported	Energy_Exported	FLAG
0	Monday	0:00:06	8.3900	0.0100	25015.2600	3130.1800	True
1	Monday	0:01:06	8.3200	0.0100	25015.4200	3130.1800	True
2	Monday	0:02:06	8.2900	0.0100	25015.5400	3130.1800	True
3	Monday	0:03:06	3.5200	0.0100	25015.6700	3130.1800	True
4	Monday	0:04:06	3.5200	0.0100	25015.7400	3130.1800	True

Figure 15 SGtech Energy Data Structure showing Power Consumption and other Energy sources

```
            Time  Power_Consumption  ...  Energy_Exported  FLAG
Day                                  ...
Monday   0:00:06               8.39  ...         3130.18  True
Monday   0:01:06               8.32  ...         3130.18  True
Monday   0:02:06               8.29  ...         3130.18  True
Monday   0:03:06               3.52  ...         3130.18  True
Monday   0:04:06               3.52  ...         3130.18  True

[5 rows x 6 columns]
       Power_Consumption  Power of PV  Energy_Imported  Energy_Exported
count        7193.000000  7193.000000      7193.000000      7193.000000
mean            8.544948     5.577118     25478.074414      3137.549704
std             6.248696     7.823366       907.760126         3.962652
min            -9.870000    -0.120000         0.000000      3130.180000
25%             4.960000     0.010000     25220.280000      3131.930000
50%             6.890000     0.010000     25460.610000      3140.040000
75%            10.580000    12.270000     25818.080000      3140.140000
max            40.240000    22.820000     26063.560000      3140.230000
```

Figure 16 SGtech Energy Data Statistics

Before this real-time data was used for the validation of the proposed model, a confirmatory test was carried out to check if there are some missing or corrupted values in the data, but as shown in fig 18 there were none. 75% of the data was used for training of the model and the remaining 25% for testing to ascertain how well it performed.

```
Day                0
Time               0
Power_Consumption  0
Power of PV        0
Energy_Imported    0
Energy_Exported    0
FLAG               0
dtype: int64
```

Figure 17 Showing no missing or corrupted values in the data

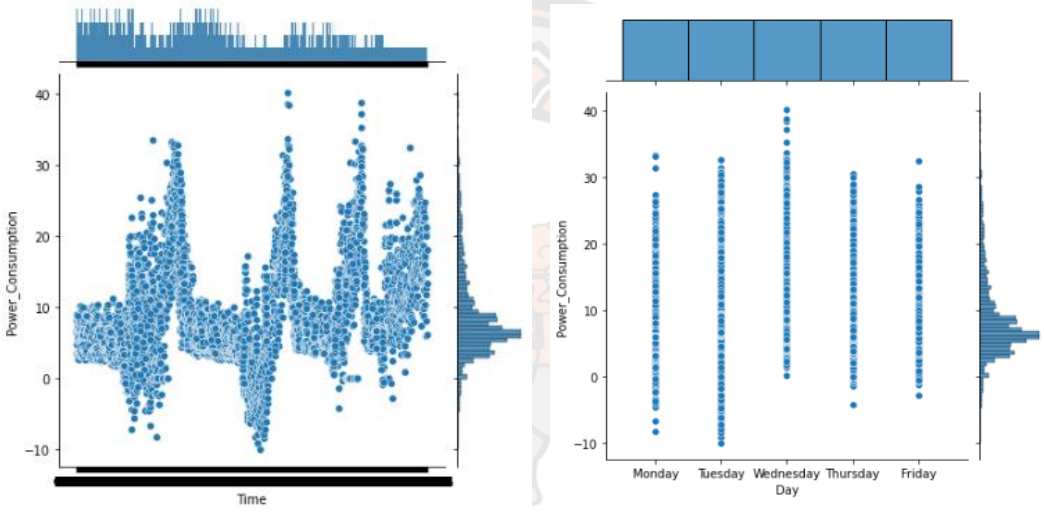


Figure 18 Comparison of power consumption over day & time

4.3 Exploratory Data Analysis

A good machine learning project involving a big data supposed to start with proper data exploratory to familiarize with the data and its features. Critical analysis is required most often through exploratory data visualization like sample variances, correlation coefficients, R-Squared and lines of best fit or other methods such automatic outliner detection and conditional summary statistics. These methods have proven to be effective as far as machine learning with high volume of high-dimensional data is concerned. Data summary statistics in the data frame of power consumption dataset is not enough to determine the discrepancies in the data distribution. Therefore, series of data visualization was carried to clearly understand the data trends, artifacts, outliners, and distributions. Since the dataset contains 7 variables originally including total active power plus one additional variable created

to serve as EV charging making it 8 variables, there is need to determine the relationship existing between those variables viz-a-viz total active power that is being forecasted. It became very important to determine how related the variable of targeted forecast to others in the data distributions.

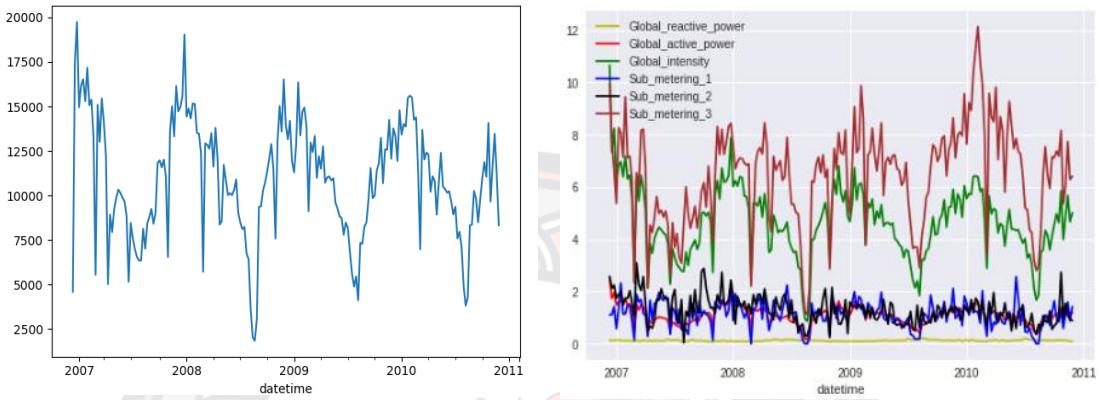


Figure 19 Global Active Power resampled over a week & Variables from the HPCD

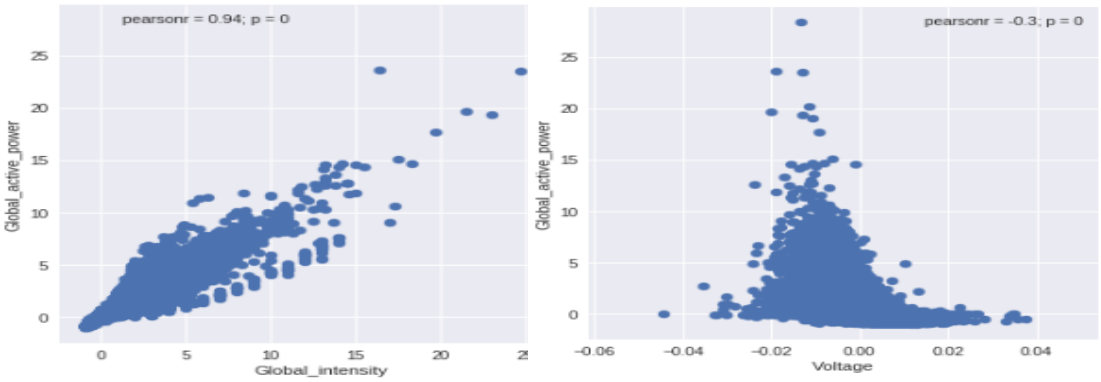


Figure 20 Correlation Coefficients of global intensity and voltage with Global Active Power

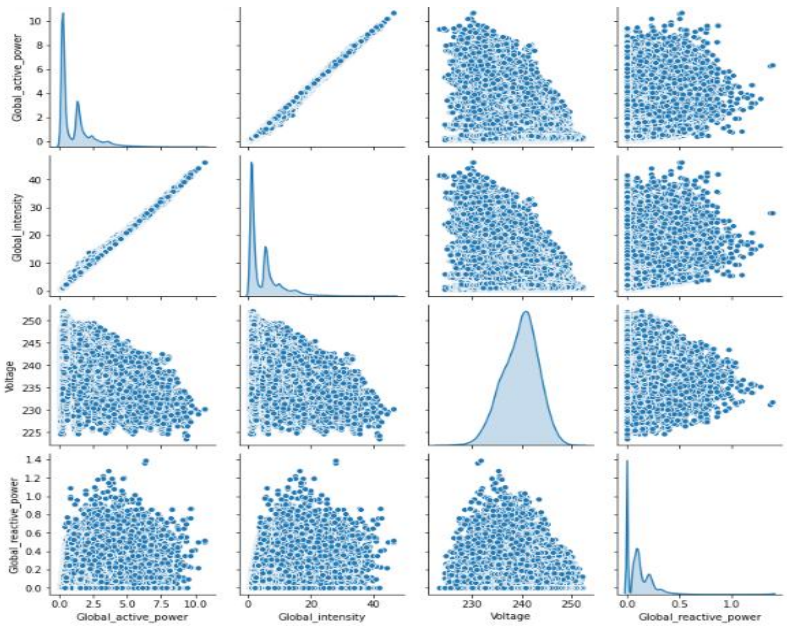


Figure 21 Plot of power consumption dataset variable distributions

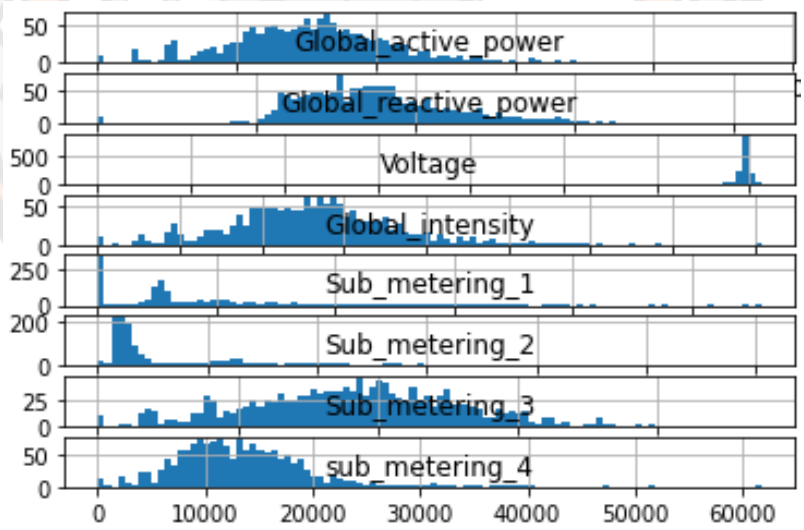


Figure 22 Individual distribution of the attributes

From fig. 21, it can be noticed that Voltage seems to have a gaussian distribution whereas rest of the data seems skewed (i.e., non-symmetric), necessitating power transformation of the data before modelling. Exploratory analysis further showed that Global Active Power expected to be predicted has strongest correlation (fig. 23) with Global Intensity with a factor of 1 followed by sub_metering_4 and

sub_metering_3 linked respectively to EV and climate-controlled systems like air-conditioner and heater. Therefore, this study further investigates the extent each input variable affects the outcome of the prediction result.

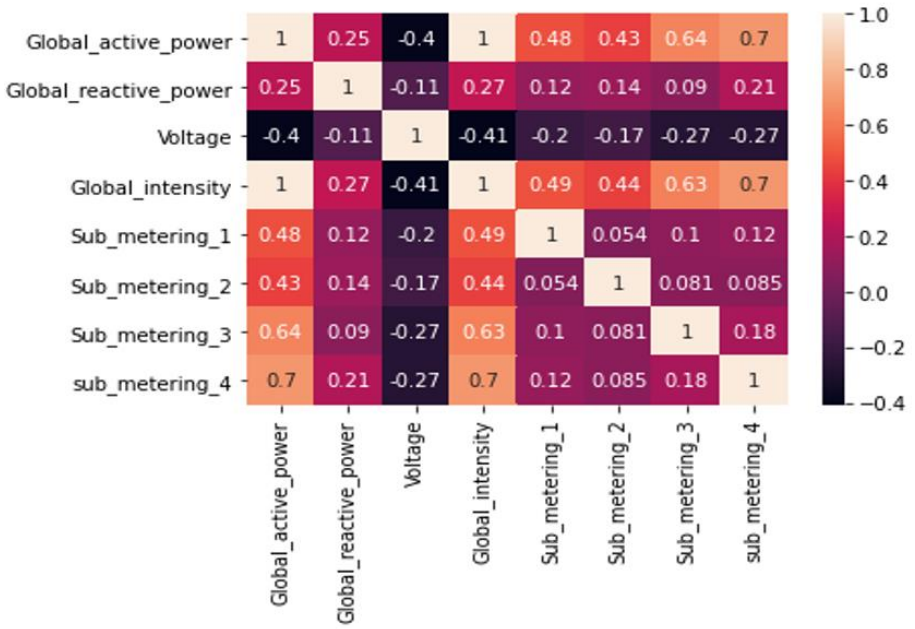


Figure 23 Correlation of 8 Input Variables

A statistical testing or confirmatory data analysis was carried out on the experimental results to draw an inference on whether the more the previous time-steps from the historical data (input data) applied to proposed model, the better the forecast performance (output). Hence, selection of the final model is based on this, to ensure only skills and predictions from the best is presented. It also compares the relationship between the data sets variables and total active power as an alternative to an idealized null hypothesis that proposes no relationship between the variables. This test is regarded as statistically significant once such relationship invalidates the null hypothesis in accordance with the threshold probability significance level.

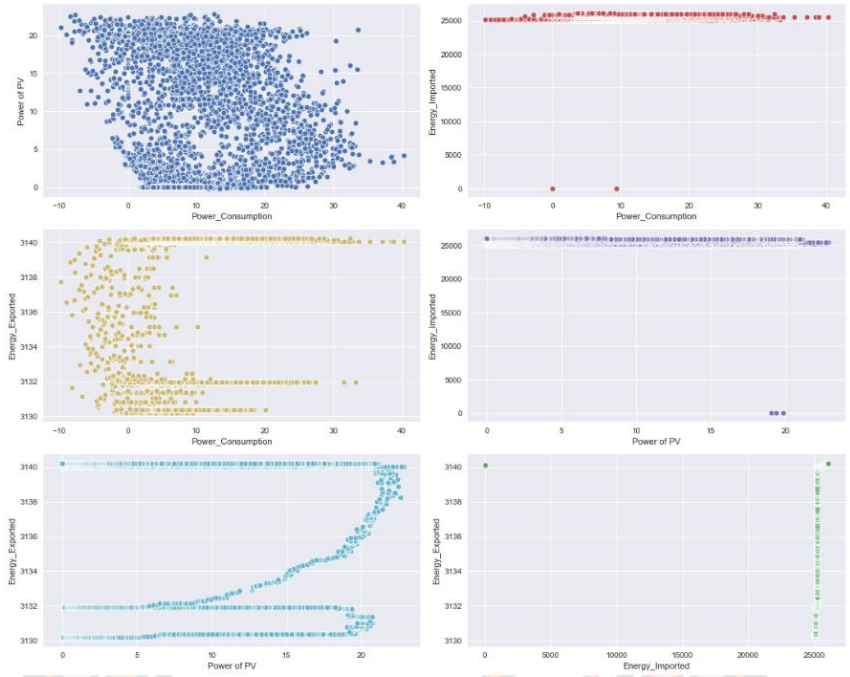


Figure 24 **Pair-wise Scatter Plot of all Continuous Variables from Smart Office**

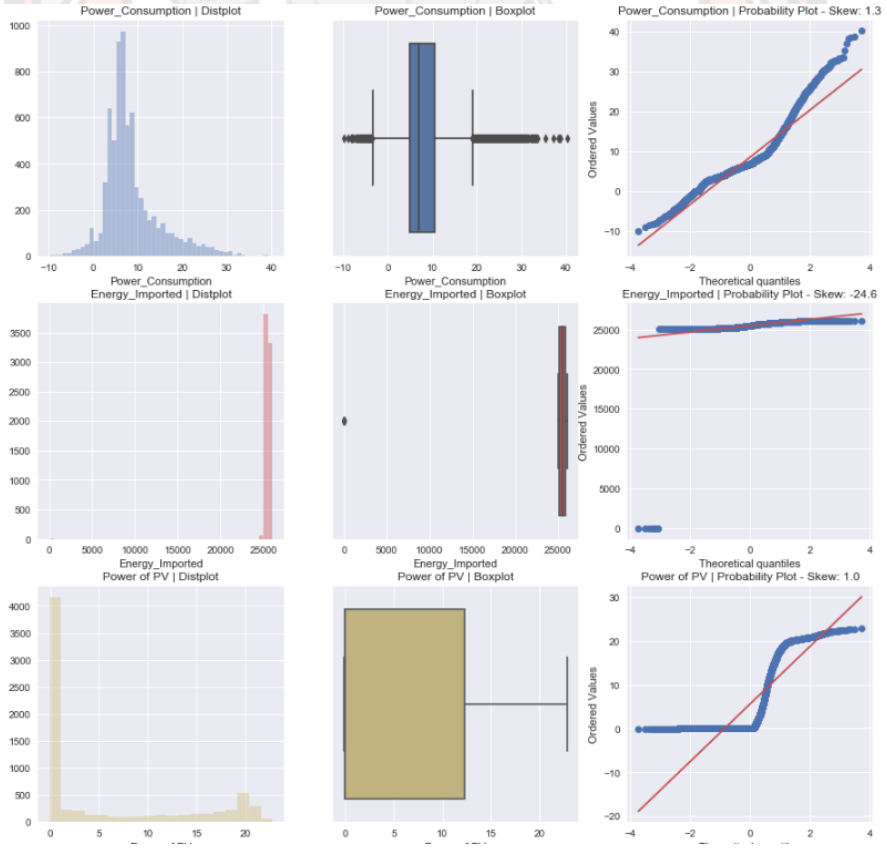


Figure 25 Pair-wise Scatter Plot of all Continuous Variables

Figure 26 Correlation Coefficient of Energy Sources

4.4 Data Pre-processing

Both primary and secondary datasets used in this work were collated using a real-time approach irrespective of season or weather conditions. Due to numerous appliances used at various homes and inability to capture their load estimation during the observation period (e.g., EV charging spot etc.), provision is made for additional sub-metering in the load profiling. Therefore, fourth sub-metering variable was created in this research by subtracting the sum of previous defined sub-metering variables from the total active energy.

Sub-metering_4 = (global-active-power * 1000/60) – (sub-metering-1 + sub-metering-2 + sub-metering-3). More so, down sampling of the observation period from minute to daily is later carried out for easy computational process since the study is interested in weekly forecast.

4.4.1 Data Cleaning

Data scientists, statisticians and computer programmers often encounter missing values in data points while constructing data for modeling or other form of statistical analysis. The secondary data used in this research was not exceptional; it contains some missing and corrupted values (nearly 1,25% of the rows). Usually, the first important step in data preprocessing prior to modeling is

cleaning of the data to ensure no error in the data. This preprocessing process also involves removal of noise and outliers from a dataset preparatory to its usage.

In this thesis, a secondary data coming from public repository used was thoroughly cleaned before it was used for the training of forecast model. However, the primary data collected in real-time from SGtech needs no further cleaning because it has no missing nor corrupted values from the result of the exploratory analysis carried out. There are two methods of cleaning: outliner detection and imputation method. While outliner detection identifies observations far from the expected value in the distribution, imputation method repairs or fill in corrupt or missing values in the observations. Therefore, the process for the cleaning of the corrupted parts, missing, and NaN values in the data adopted in this research is imputation method.

This method filled all missing and corrupted values using a day-wise Last Observation Carried Forward (LOCF) technique. This imputation method copied observation from the same time in the missing value from a day before (exactly 24hours ago). In a time-series data like power consumption with seasonality trends, other methods like linear interpolation, seasonal adjustment + linear interpolation could also be applied.

4.4.1.1 Last Observation Carried Forward (LOCF)

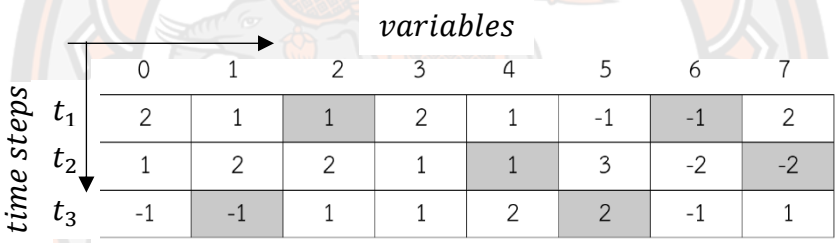
For regression and classification problems the task of estimating the values of missing data or substitution new values is always based on the traditional approach i.e., manual imputation. In dealing with missing time series data, characteristics should be taken into consideration, so an appropriate and efficient strategy of handling the missing data could be developed. From Table (1) for an instance, the multivariate data in the time series has some missing values. The vertical axis shows the time steps while horizontal axis represents each input variables.

Table 1 Multivariate Input data with missing values

time steps \ *variables*	0	1	2	3	4	5	6	7
t_1	2	1		2	1	-1		2
t_2	1	2	2	1		3	-2	
t_3	-1		1	1	2		-1	1

Last Observation Carried Forward method is simply implemented by carrying the last available value at the previous 24hrs time and impute it for the replacement of the missing value. This process continues is this order until all the existing missing values or corrupted ones are completely replaced with new values. Table (2) shows the complete multivariate time series data ready to proceed to the next stage. The method relied on the assumption that there will be no significant change within the missing period. However, if this assumption is unsatisfied, it then means that the dataset will be biased. Formal expression of last observation carried forward method is depicted in the Equation (4).

Table 2 Completed time series courtesy of last observation carried forward technique.



	0	1	2	3	4	5	6	7
t_1	2	1	1	2	1	-1	-1	2
t_2	1	2	2	1	1	3	-2	-2
t_3	-1	-1	1	1	2	2	-1	1

For x_t^v, a value of variable *v* at timestep *t*,

$$x_t^v = \begin{cases} x_t^v, \text{ if is Observed} \\ x_{t^\wedge}^v, \text{ Otherwise} \end{cases} \quad (4)$$

where $t^\wedge < t$ is regarded as the timestep of lastly observed value of the multivariate input variable *v*.

4.4.1.2 Linear Interpolation Technique

As the name suggests, linear interpolation is a very popular imputation technique that assumes a linear relationship between data points exists and hence replaces the missing values with results derived from non-missing, adjacent values to the missing points. Its implementing often requires a lag function, which

returns adjacent values previously stored in the lag queue. Some of the techniques in implementing this like Base SAS is very complex, requiring multiple sorts of data steps to identify and isolate the necessary components before merging the components back onto the source data set to perform the imputation

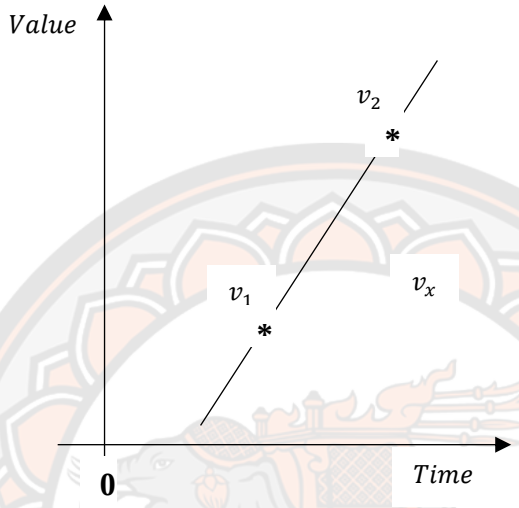


Figure 27 Shows the linear relationship existing between a missing data point Vx and, non-missing observations V1 and V2

In a circumstance where the value at point V_x is missing, both the last actual observation taken prior to point V_x, which is point V_1, and the first actual observation taken after point V_x, which is point V_2, will be used to compute the value at point V_x. Hence the coordinates at the three points are (V_1 time, V_1 value), (V_2 time, V_2 value), and (V_x time, V_x value), respectively. To calculate the missing values V_x value at V_x time for V_x observation the following equation can be used:

$$v_x\text{value} = v_1\text{value} + \left[\frac{v_2\text{value} - v_1\text{value} * v_x\text{time} - v_1\text{time}}{v_2\text{time} - v_1\text{time}}\right] \quad (5)$$

4.4.1.3 Seasonality Adjustment

Seasonality adjustment in time series data is the method used in removal of the seasonal variation from time series. It works perfectly well with data with trends and seasonality. Seasonal adjustment may be performed by some more or less conventional methods, such as Structural Time Series (STS) models, Bayesian seasonal adjustment, signal-extraction methods, different nonparametric (like spline-

based) methods etc. To implement seasonality adjustment with linear interpolation, firstly the time series needs to be decomposed according to the decomposition model (additive or multiplicative). Thereafter, the missing values are interpolated using the interpolation method (linear, cubic, or quadratic) on a series consisting of only the trend and irregular components. And finally, the seasonality is added back to the series.

4.5 Transformation of Time Series Data

Before proceeding with modeling of power consumption behavior, it is important to transform the time series data into a supervised learning format. Most of the practical machine learning uses supervised learning. In Supervised learning there are input variables (X) and an output variable (y), and the developed algorithm was used in learning the mapping function from the input to the output. The goal is to approximate the real underlying mapping so well that when it sees a new input data, it can be able to predict the output variables for that data. Time series data can be phrased as supervised learning. Given a sequence of numbers for a time series dataset, this data can be restructured to look like a supervised learning problem. This can be done by using previous time steps as input variables and use the next time step as the output variable. For example, the series:

X = 1, 2, 3, 4, 5, 6, ……

Can be transformed into samples with input and output components that can be used as part of a training set to train a supervised learning model like a deep learning neural network used here. This transformation pattern is referred to as a sliding window transformation as it is resembles sliding of a window across prior observations that are used as inputs to the model for the purposes of predicting the next value in the series. Looking at it, it showed the window width is 3 time-steps with 1 output.

X	y
[1, 2, 3]	[4]
[2, 3, 4]	[5]
...	

4.6 Model Configuration

An extension of Convolutional Neural Networks and Long Term-Short Memory (CNN-LSTM) algorithm called ConvLSTM is used in modeling the power consumption behavior for a forecast. ConvLSTM is a combination of Convolutional Neural Networks and Long Term-Short Memory and the difference between CNN-LSTM and ConvLSTM is that the later performs the convolutions of the CNN (especially how it reads the input sequence data) as part of the LSTM for each time step. A kernel size of (1, 3) was used, specifying the dimensions of the convolution window. The dimension of the output space (filter) is 64 i.e., the number of output filters in the convolution and rectified linear unit (ReLU) activation function used. LSTM with 100 neurons in the first visible layer, with dropout 20%. 1 vector neuron was used in the output layer for predicting Total-active-power. The input shape will be 1-time step with 7 features, using the Root Mean Square Error (RMSE) loss function for evaluation. The model is fit for 20 training epochs with a batch size of 8.

4.6.1 Encoder-decoder-network

An advanced ways of feature representation like encoder-decoder was considered to preserve the hidden abstractions and invariant structures in the time series input. This have been previously applied in both reinforcement [7], supervised [21] and unsupervised learning . This neural network is designed for adaptively learning of long-term dependency and hidden correlation features of multivariate temporal data. An encoder that extracts useful and representative features from the time series data was trained in such a way that the decoder can conveniently reconstruct those features from the encoded space. The output of the hidden layer of the network represents the learned features. Specifically, the output of convolutional layers is concatenated by ConvLSTM2D followed by LSTM layers just like in [22, 23] to capture all the inherent spatiotemporal correlations in the time series. This proposed ConvLSTM Encoder-Decoder architecture has 2-sub models: one for reading the input sequence and encoding (i.e., mapping the variable-length source sequence) it into a fixed length vector, while the second part decodes the fixed-length vector and output the predicted sequence (i.e., mapping the vector representation back to a variable length target sequence. Thereafter, a dense layer is used as the output for

the network, and it used same weights by wrapping the dense layer in a *TimeDistributed* wrapper function used in the network.

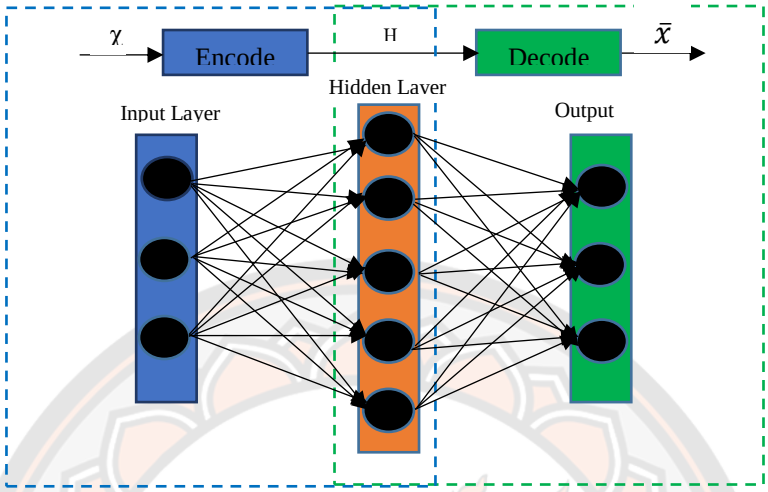


Figure 28 Schematics of Encoder-Decoder Structure

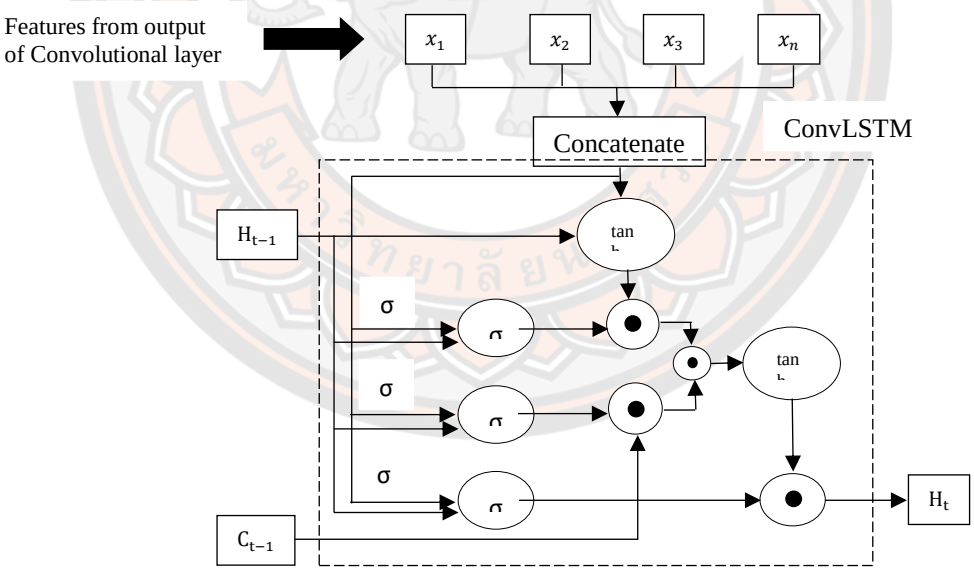


Figure 29 ConvLSTM Architecture

4.7 Model Improvement Techniques

Developing a state-of-art model requires some enhancement efforts. Some of these efforts could be through combination of algorithms, methods or simply reshaping of mapping function that is being learned.

4.7.1 Model Ensemble

Randomness associated with neural networks algorithms makes prediction results conducted on a particular dataset over different trainings inconsistence due to high invariance and low biases encountered during training. This implies that as the data is trained the model gets sensitive to specifics of the training data and the initial conditions and for the training. This results to finding different sets of weights each time the model is trained that makes it produce different predictions. Therefore, combining multiple best performing models and combining their predictions yields a reliable result. Through this method a bias which counters the variance of a single trained model is added resulting into predictions less sensitive to the specifics of the training data, choice of training scheme, and the serendipity of a single training.

However, this thesis leverage ensemble technique to improve the predictive model performance. This ensemble idea was firstly introduced in 2012 by AlexNet [8] in the famous image classification problem that won machine learning competition, and has since be deployed in other problems like complex regression etc. to achieve state-of-art results. AlexNet used model averaging method across multiple models with similar configuration and different initial random weights in data training on the same dataset. However, many other approaches has been used by researchers [24] to achieve model ensemble like K-fold cross validation [25], Random-Split, Bootstrap aggregation (or bragging) and stacking etc., through varying of training data (like [26] that ensembled the snapshots of a single model optimization path), model's configuration, or varying combinations. [4, 27] [28] used different combinations of algorithms in ensemble way to achieve high model performance.

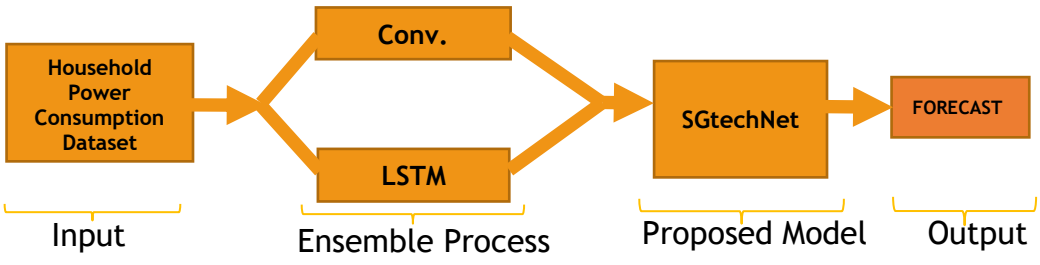


Figure 30 Architecture of the proposed model

Considering scarce computing resources and need to make a quick forecast, single model is trained instead of multiple models due to the success it recorded in [26, 29]. Model averaging method is used to address problems of high variance and low biases always encountered during training of neural networks. This method significantly makes the model less sensitive to specifics of the training data, choice of training scheme, and the serendipity of a single training run.

4.7.2 Ensemble Methods

Stochastic nature of power consumption (varying with season and time) necessitated the use of stochastic learning algorithm for dataset training. However, neural network algorithm has its inherent limitation of randomness which results in a different final model each time it is trained on the same dataset. To address this limitation, an ensemble method [27] with a weighted average of different trained models is used for prediction. Ensemble methods are aimed to improve the accuracy of results in machine learning models by combining multiple models instead of using a single model. Because judgment made by and an individual model cannot always be trusted, a combination of models helps to significantly increase the accuracy of the results. Ensemble can be achieved either by varying the training data (K-Fold Cross-Validation, Bootstrap Aggregation, and Random Training Subset), varying the models (Multiple Training Run Ensemble, Hyperparameter Tuning Ensemble, Snapshot Ensemble, Horizontal Epochs Ensemble, Vertical Representation Ensemble) or varying combinations (Model Averaging Ensemble, Weighted Average Ensemble, Stack Generalization Ensemble, Boosting Ensemble and Model Weighted Averaging Ensemble).

Ensemble can as well be implemented in form of bagging, boosting, and stacking through two main broad categories: sequential ensemble and parallel ensemble. Example of sequential ensemble technique is Adaptive Boosting (AdaBoost), that generates base learners in a sequence by assigning higher weights to previously misrepresented learners to improve performance. On the other hand, a typical example of parallel ensemble is Random Forest. This parallel ensemble encourage independence between the base learners through parallel generation of base learners.

4.7.3 Types of Ensemble Methods

Bagging: is used in describing bootstrapping and aggregation sampling technique where samples are derived using the replacement procedure and aggregated to incorporate all possible outcomes of the prediction and randomize the outcome. [30] combined several weak learners (base learners) to build a stronger model using the mean rule. Bagging is mainly applied in classification and regression tasks. Bagging increases the accuracy of models through reduction of variance and elimination of overfitting which poses a challenge for predictive models. It increases accuracy through decision trees, which reduces variance to a large extent. Even though bagging has the advantage of combining weak learners to form stronger learner, it is computationally expensive.

Boosting: this technique improves accuracy by learning from previous predictor mistakes and utilizing such to make better predictions in the future. The technique combines several weak base learners to form one strong learner, thus significantly improving the predictability of models. Some of examples of boosting includes gradient boosting, Adaptive Boosting (AdaBoost), and XGBoost (Extreme Gradient Boosting). Boosting works by arranging weak learners in a sequence, such that weak learners learn from the next learner in the sequence to create better predictive models. Boosting is a sophisticated way of ensemble where ensemble members are added one at a time to correct the mistakes of prior models. The added complexity means this approach is less often used with large neural network models. Boosting can be achieved in combination with other methods just like [30-33]. While AdaBoost uses weak learners in the form of decision trees, Gradient Boosting adds predictors sequentially to the ensemble, through which preceding predictors can correct their successors as way of increasing the model's accuracy. In gradient boosting, new predictors are fit to counter the effects of errors in the previous predictors. The gradient of descent helps the gradient booster identify problems in learners' predictions and counter them accordingly. XGBoost uses decision trees with boosted gradient for performance and speed improvement.

Stacking: this technique allows training algorithm to ensemble other similar learning algorithm predictions for accuracy improvement. [34] utilized different algorithms to perform ensemble training at various levels of stacking.

Stacking has been successfully implemented in regression, distance learning, density estimations, and classifications tasks. It can also be used to measure the error rate involved during bagging. Stacked generalization works by deducing the biases of the generalizer(s) with respect to a provided learning set [35]. This deduction proceeds by generalizing in a second space whose inputs are (for example) the guesses of the original generalizers when taught with part of the learning set and trying to guess the rest of it, and whose output is (for example) the correct guess. When used with a single generalizer, stacked generalization is a scheme for estimating (and then correcting for) the error of a generalizer which has been trained on a particular learning set and then asked a particular question.

4.8 Weighted Average Ensemble Technique

Some ensemble method uses a generalized committee prediction given by a weighted combination of the predictions of the members, but this could produce a bad result if there are bad learners amongst the committee members. Model averaging ensembles are limited because they require that each ensemble member contribute equally to predictions. Which means that a bad learner among the model can jeopardize the effort of good learners since they are allowed to contribute equally to the final model use for the prediction.

However, use of average or weighted average of model weights in the final model has become a common practice to ensure best performance results are achieved from the training run. This approach is the main brain behind Google Inception V2 and V3 deep convolutional neural network models wonderful performance in photo classification, a milestone in the field of deep learning. Ordinarily, model ensemble method allows each model an equal contribution to the final prediction which could sometimes be seen as a limitation because poorly performed model's contribution to the final model can jeopardize the efforts of well performed one.

Another combination that is a little bit different is to combine the weights of multiple neural networks with the same structure. The weights of multiple networks can be averaged, to hopefully result in a new single model that has better overall performance than any original model. This approach is called model weight averaging [36]. This approach of averaging these points in weight space, and use a network with

these averaged weights, instead of forming an ensemble by averaging the outputs of networks in model space has been found to be promising.

However, contribution to the final model in this proposed ensemble method in this thesis is purely dependent on model's trust and estimated performance, which expectedly offered the needed low variance and low bias that improved the final prediction result. To determine trustworthiness of models and estimate performance, there is need to find their individual weights. However, due to no analytical solution to estimation of values for the weights, a gradient descent optimization (with a unit norm weight constraint) was used on the holdout validation set rather than on the training set. Ordinarily, a simpler way of finding each ensemble member's weights would have been to grid search values but because the holdout validation data is large enough, but in this case gradient descent optimization is the best option. This optimization procedure sums up all the model vector of weights to 1 i.e. $\boldsymbol{w_1, w_2, \cdots, w_k}$ = 1, also constraint them to positive values to allow weights indicate the percentage of trust or expected performance of each model.

4.9 Model Tuning

Layers in Neural Networks is one of the determinants of the result of the prediction model. The deeper the layers the more robust is the model even though it makes the model more complex too. Fewer number of layers is enough for a simpler problem, but a larger number is more appropriate for a more complicated problem. The number of layers can be tuned using the "for loop" iteration. Inserting regularization layers in a neural network can help prevent model overfitting. And there are two regularization layers: Batch normalization layer and Dropout layer. Batch normalization layer is normally placed after the first hidden layers to normalize the values passed to it for every batch. This is like standard scaler in conventional Machine Learning.

On the other hand, dropout layer, as its name suggests, randomly drops a certain number of neurons in a layer as shown in fig. 41. The dropped neurons will no longer used during the training. The dropout rate sets the rate of how much percentage of neurons to be dropped.

4.9.1 Pruning Method

Pruning is model size reduction method that reduces the number of parameters through removal of redundant, unimportant connections that are insensitive to performance of the model. Fig 31 shows how most of the redundant connections are removed and it did not only helps reduce the overall model size but also saves on computation time and energy.

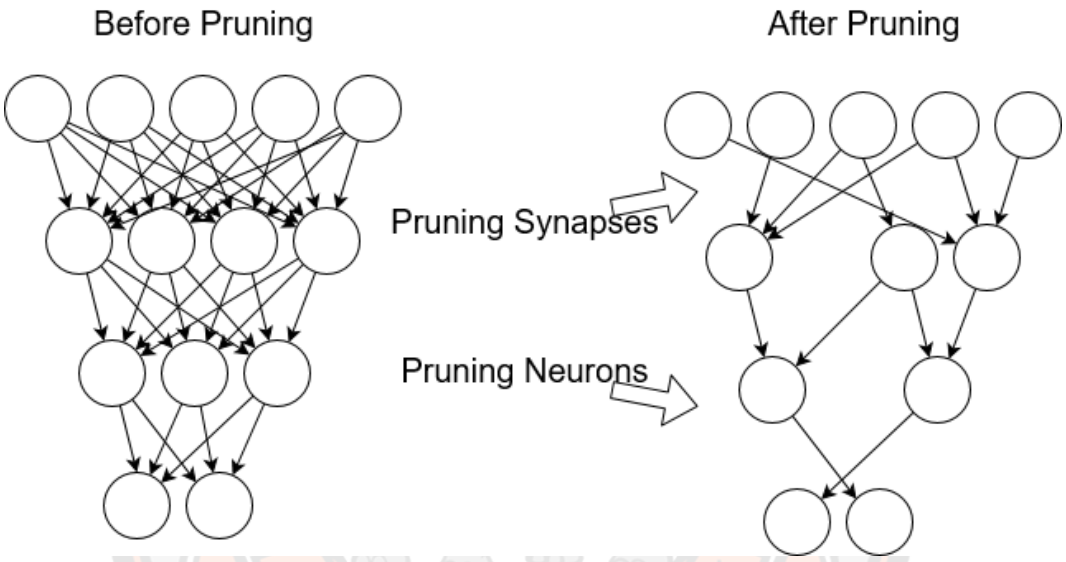


Figure 31 Parameter Pruning Technique

Source: towardsdatasciencience.com

4.9.2 Low-rank Factorization

In this method of model compression, matrix/tensor decomposition is used to estimate the informative parameters. As shown in fig 32 is a weight matrix with m x n dimension, where a rank r is replaced by smaller dimension matrices. By this method, a large matrix can be factorized into smaller matrices and can also reduce training time if applied during training.

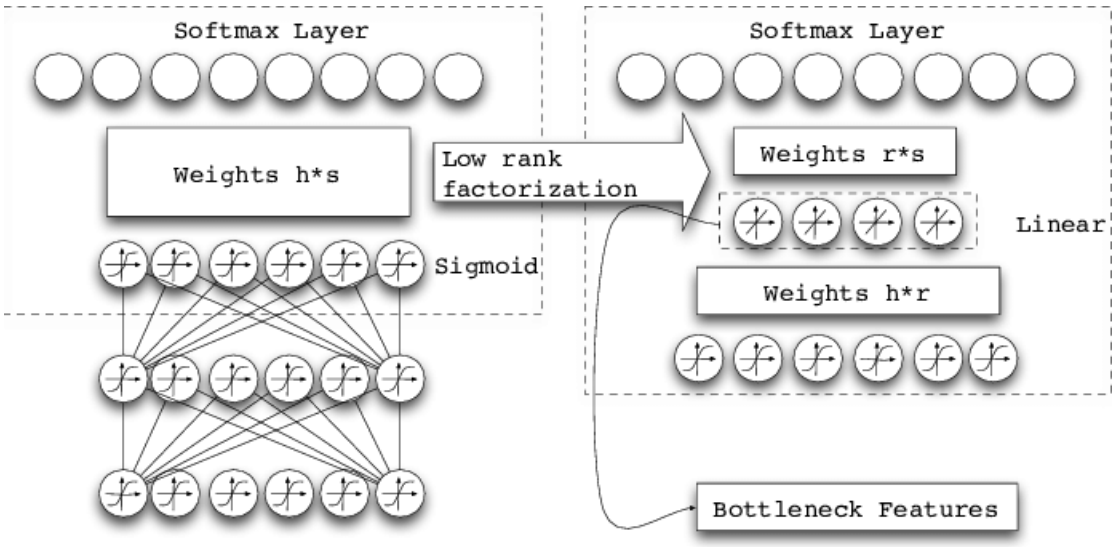


Figure 32 Matrix Decomposition Technique

Source: researchgate.net

4.9.3 Quantization Method

Quantization method came up with the idea of reducing the number of bits in Deep Neural Networks weights, which are stored as 32-bit floating-point numbers. The weights can be quantized to 16-bit, 8-bit, 4-bit or even with 1-bit. As the number of bits used in the networks is reduced, the size of the deep neural network itself is significantly reduced.

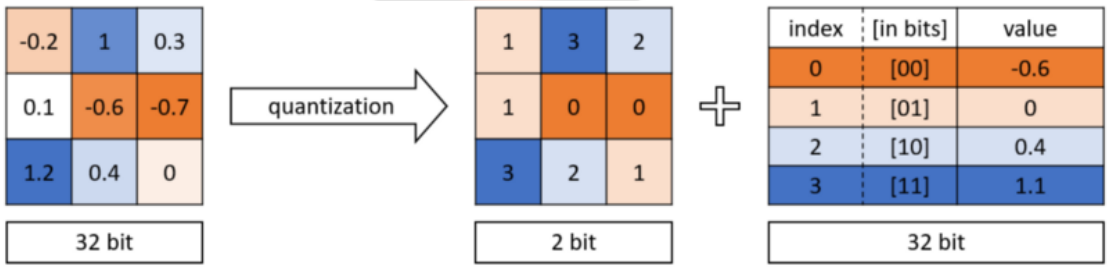


Figure 33 Weight Reduction Process through Binary Quantization

Source: software.intel.com

4.9.4 Knowledge Distillation Method

In this method a large, complex model is firstly trained on a large dataset. And when it is obvious this large model can generalize perfectly on an unseen data, it is then transferred to a smaller network. The larger model is also known as the teacher model and the smaller network is also known as the student network.

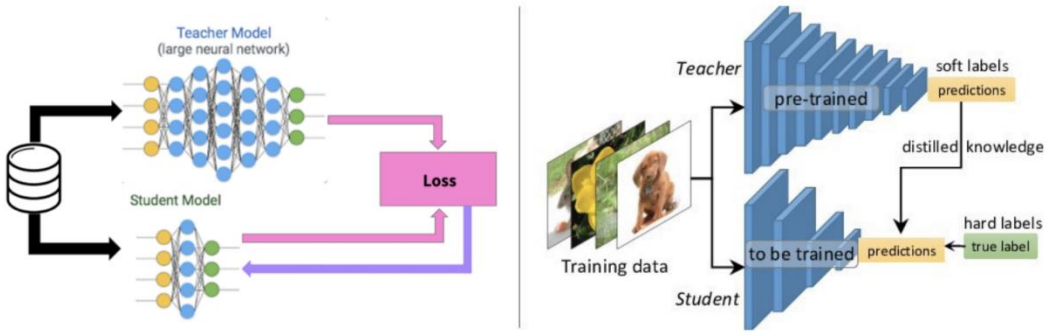


Figure 34 Distillation Process

Source: towardsdatascience.com

Most of these techniques can be applied to pre-trained models, as a post-processing step towards reduction of the model size and increase inference speed. Interestingly, they can be applied either during or after training. Quantization is currently gaining popularity and has now been baked into machine learning frameworks. In no distant time it is expected that pruning to be baked into one of the popular frameworks also.

4.10Model Compression Technique

Earlier deep learning models like AlexNet, VGGNet, SENet-154, ResNet-101 etc., have very large trainable parameters making it difficult to run in real-world applications requiring real-time processing like smartphones, self-driving car, virtual and augmented realities, smart cities etc. In present IoT era, there has been continual increase in the number of devices using Artificial Intelligence (AI). Consequently, research efforts are on to create moderate size models implementable in smart devices like IoT cameras including security cameras, cameras for smart vehicles to assist in driving, and cameras at parking lots, locks, smart light bulbs, kettles, and thermostats

etc. It is projected that by 2030, the number of IoT devices will be between 125 – 500 billion. Just imagine what would be the faith of these device when connected to the internet at the same time.

One major reason models with large trainable parameters are not good fit for on-device based devices is because of resource constrained; they limited memory and low computer power. The more the trainable parameters in a model the bigger the size, energy needs and space the model occupies. All these results to a higher cost burden in terms of training resources. As the name suggests, deep learning models have a more layers as it goes deeper, and its inference time increases along with the increase in number of trainable parameters. This means that model with a very big size is going to be difficult to deploy on resource-constrained devices. A lot of models that performs successfully in the lab could not replicate same in real-world applications due to their size. In the lab, it is operated with high-speed GPU which might not be in a real-world application making it encounter both high cost, power, heat, and other difficulties during use.

Table 3 Some Existing Models with their Parameters

Model	# Layers	# Parameters	# MACCs	Error-5(%)
AlexNet	8	60	650	19.7
ZefNet	8	60	650	11.2
VGG16	16	138	7800	10.4
SqueezeNet [24]	18	1.2	860	19.7
GoogleNet	22	5	750	6.7
Inception-v3	48	23.6	5700	5.6
Inception-v4	70	35	6250	5
ResNet-101	101	40	3800	6.8
ResNet-152	152	55	5650	6.7
ResNet-200	200	65	6850	5.8
ResNeXt [25]	101	68	4000	5.3
DenseNet-201	201	16.5	1500	6.3
SENet-154	154	100	10,500	4.5
MobileNet-v1	28	4.2	569	10
MobileNet-v2	28	3.5	300	9
ShuffleNet	11	5.3	260	10

However, deployed a model with large trainable parameters on the cloud would have been a solution instead of compressing it to reduce the size but network latency problem of cloud services is another challenge. Even though cloud has high computational and storage capabilities, poor response time sometimes makes it unacceptable in many real-time applications. So, the model size reduction technique implemented here is squeeze layer which is like AlexNet. There are few other methods like pruning, quantization, knowledge distillation, lo-rank factorization that complements each other in reducing the size of the model. These methods can work in isolation or in combinations with one another. For an instance, a three-stage pipeline; pruning, quantization, and Huffman coding was used in reducing the size of VGG16 model trained on the ImageNet dataset from 550 to 11.3 MB.

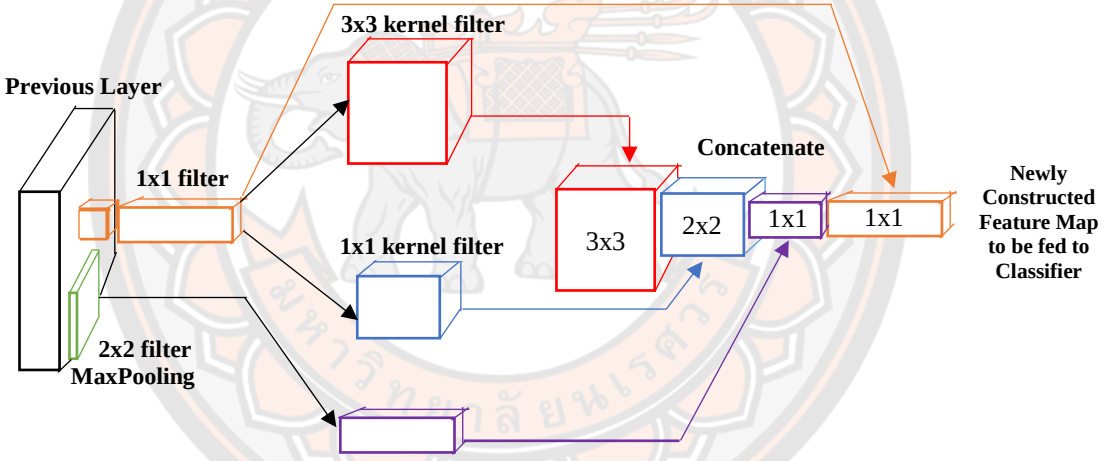


Figure 35 Inception Model Feature Extraction Module

4.11 Feature Extraction

As earlier mentioned, this SGtechNet leverage on SqueezeNet concept of fig 33, but instead of using 1x1 and 3x3 convolution filters for feature extraction, 1x1 and 1x3 were used. The kernel convolution layer was designed to extract features from the input time series data in a sequential way. As each kernel receives input time series, the corresponding outputs are concatenated and followed by convolutional-LSTM layers which captures the long-term spatial patterns in the data. This method achieved an improved result but not without marginal cost burden due to a slight increase in number of parameters. However, the choice of smaller filter is for the reduction of the

number of parameters as well as for reduction of processing time. Alternatively, large filters are preferable when it becomes necessary to detect the central position of an object like an image. SqueezeNet has almost similar accuracy of AlexNet but a little lower than GoogleNet. SeNet shown in fig 36 developed an architecture that recalibrates channel-wise feature responses and use it to determine the interdependencies existing between two channels. Channel-wise scale and element-wise summation operations were combined into a single layer “AXPY” using skip-connections. This resulted into considerable reduction in memory, cost, and computational burden.

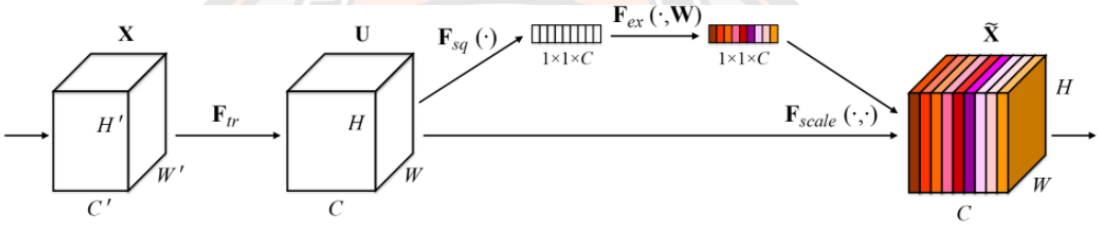


Figure 36 Squeeze and Excitation Building Block [37]

CHAPTER V

FORECASTING

Because this thesis is interested in predicting the per day power consumption for next 7 days, the data is therefore grouped on day level. This creates a data set on date level which can be converted to week level to predict the weekly consumption of power, but in this case, it is left at day level because the need is to predict it for next 7 days (day level). From fig. 37, some lagged features can be seen to be introduced into the modelling to assist in learning the data better since the prediction is for the Total Active Power consumption for the next day. And 6 additional features were created that has Total Active Power of last 6 days in each column as an independent attribute. For the fact, the prediction is multi step called for lead variables that will act as target variables representing Total Active Power consumption on each next day from day 1 to day 7. Similarly, some lead variables were introduced and new features like weekend and weekdays added. Energy user availability and usage behaviors makes the active power consumed per user changes very much for weekdays and weekend, consequently few additional features like maximum power consumed during week, minimum power consumed during week, rate of change of power consumed in a week etc. were further introduced.

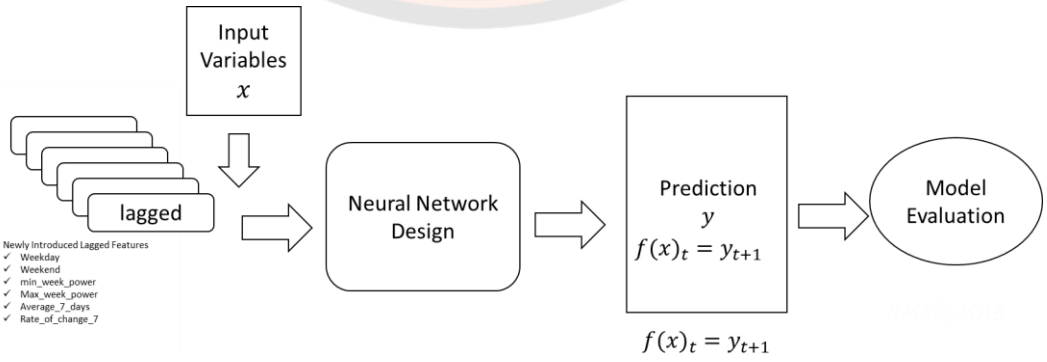


Figure 37 Proposed Neural Networks (SGtechNet) Model Architecture

In machine learning, modeling a system requires setting a target (label) from the attributes in the input training set. So, two different lists were created that will hold variables in independent variables (attributes) and that of dependent variables (label). All these steps are taken to ensure the feature extraction is perfectly done to ensure a reliable forecast result.

5.1 Model Configuration

The architecture of this network has 8 input dimensions with 1 output layer, 3 convolutional and hidden layers each. This architecture consists of combination of Convolutional Neural Network (CNN) and Long Sort-Term Memory (LSTM) deep networks. While the input transformations and feature map extraction take place in the convolutional layers, the resulting out is convolve and read into LSTM units. Because our input data is a 1-D sequence, it was easy for the interpretation over the number of time steps. The LSTM has 3 hidden layers with 4 gates that handles updates and memory functions of the network. As the gates receives both the input (output from the last convolutional layer) obtained at previous time step (h_{t-1}) and the related current time step (x_t).

5.2 Feature Extraction

Fitting of non-linear model in a neural network is as important as the modeler insight on how to represent the data. However, due to randomness in time domain problems, this study further analyzed the input data in frequency domain to ensure random signals are dully converted into different frequencies for stability and easily predictability. This concept can be likened to [19] that utilized Fast Fourier analysis to determine the dominant frequencies in electricity load values by designing a high pass filter to filter the dominant frequencies which was later converted to time domain. This approach considerably improved the model's performance. An average pooling was incorporated into the model configuration to fuse the features from all the time and summarize the features of the entire sequence [22]. As shown in fig 38, 2x2 filter was used in down sampling of the 1D input data so the model can learn the features properly. Average pooling function used after a convolutional layer simply calculates the average value for patches of a feature map and uses it to create a down sampled (pooled) feature map. This reduction in dimensionality of the input representation allows assumptions to be made about features contained in the sub-regions.

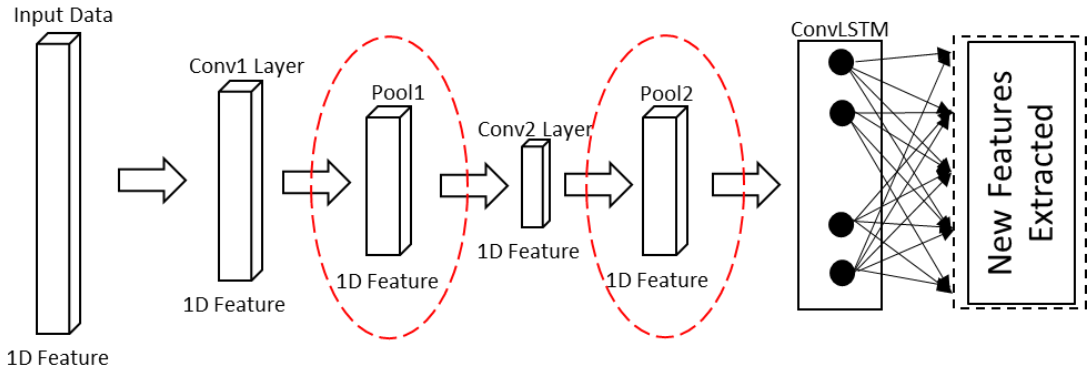


Figure 38 Feature Extraction Process.

In this work, the final model is made to make predictions for new data with unknown outcome. Walk-forward validation approach, (where the model makes a forecast for each observation in the test dataset one at a time by adding up the true observation for the current time step as part of the input for making prediction on the next time step) is used. Actual input data from previous 14days time steps was used to make a week ahead prediction. The model used the efficient Adam version of stochastic gradient descent to optimize the mean squared error ('MSE') loss function.

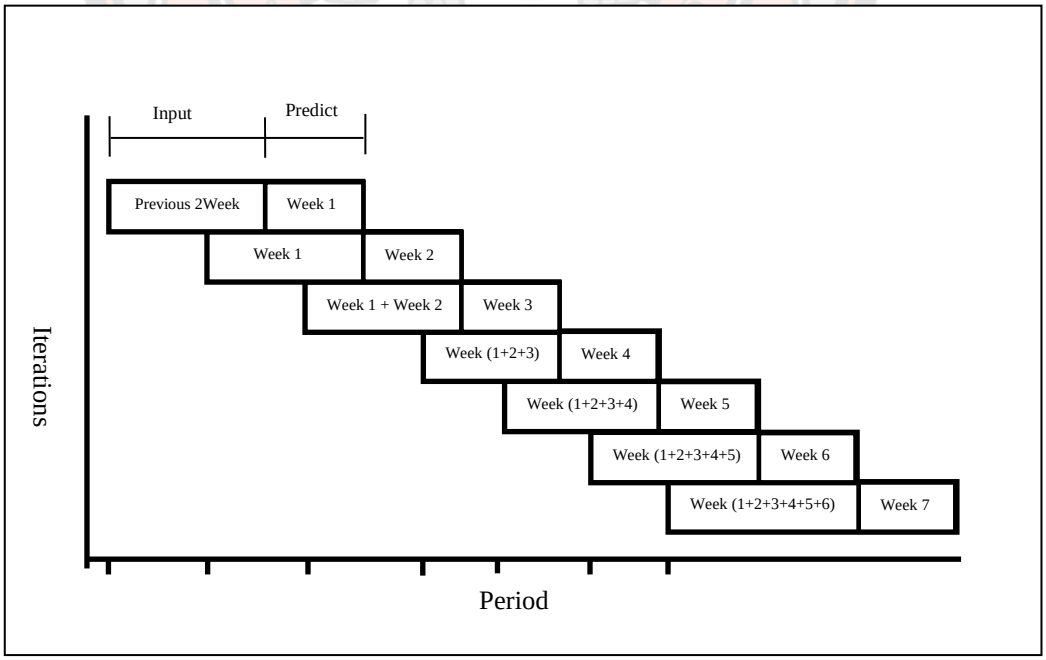


Figure 39 Walking-Forward-Validation Process

5.3 Model Parameters

This model is made up of a total of 201 neurons (100 neurons at each hidden layer) and 1 in the output layer used for predicting Total-Active-Power, the first layer of the network consists of a ConvLSTM with filter output size of 64, kernel size of (1, 3), Dropout of 50% of the layers, Input shape of 2 time-steps with 8 features, Mean Square Error loss function and Adam version of stochastic optimization gradient. The model was fit for 20 training epochs at batch size of 8.

5.4 Model Regularization Techniques and Hyperparameter Tuning

Training neural network models requires high level of prudency due to its sensitivity. There should be tradeoffs when training to avoid overfitting or underfitting problems. If a model is overtrained, it will become over familiarized with the training data that any other data (like a new data) not exactly as the training data will get the model confused. This situation is likely to lead to the model yielding unreasonable performance or poor prediction or classification. On the other hand, if a model is undertrained it leads to underfitting, which means it failed to learn the training data very well. This is the major cause of poor performance of neural networks models. Fig. 43 shows the different ways models behaves during data training.

Because many important parameters of neural networks model cannot be estimated directly from the training data calls for model tuning. Tuning is usual appropriate where there is no analytical formula available to calculate appropriate values for the parameters that controls the complexity of the model. In this situation, a poor choice of values will certainly result into either overfitting or underfitting. So, a strike of balance is highly recommended in this case during model training to avoid problems of overfitting and underfitting. There are so many methods of searching for the best parameters even though applying a defined set of candidate values that generates certain estimates of the model across candidates values as depicted in fig 38 is regarded as a general approach. Other approaches include Grid Search (where the data is not very much), Bayesian Optimization, Evolutionary Algorithms, Gradient-Based Optimization, Keras' Tuner, and Random Search etc.

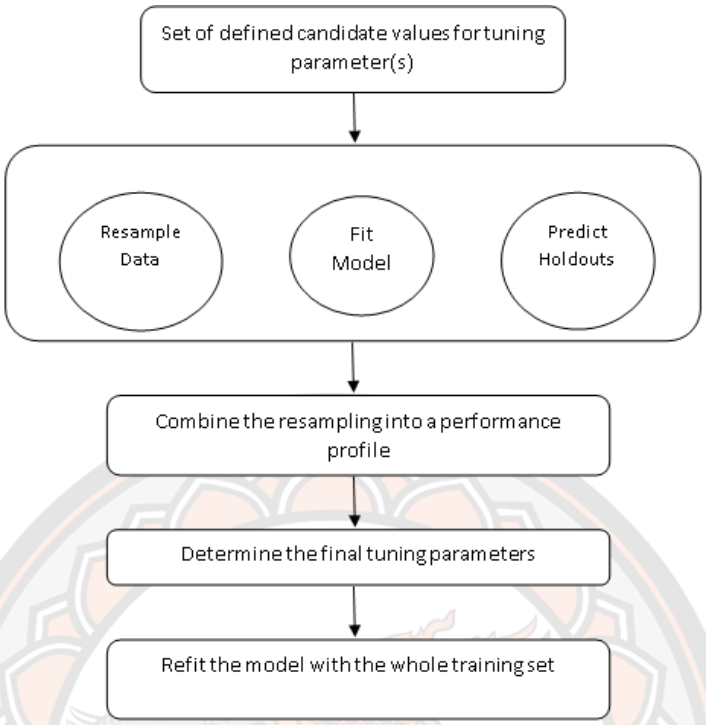


Figure 40 Parameters Tuning Process

The need for regularization is to address challenges of overfitting and error generalization inherent in time domain neural networks. Problem with learned features is that they can be too specialized to the training data or overfit, and not generalize well to new examples. Dropout which has been a common technique in deep learning helps to block out random set of unit cells during model training [38]. Network structure optimization and parameters tuning is carried out instead of grid searching because of the large dataset. And Rectified linear unit (ReLU) is used for activation. ReLU [39] is one of the most notable non-saturated activation functions. The ReLU activation function is defined as: ai, j, k = max (zi, j, k, 0), where zi, j, k is the input of the activation function at location (i, j) on the k-th channel. ReLU is a piecewise linear function that output the input directly if it is positive but prunes the negative part to zero if otherwise.

5.4 Other Methods to Prevent Overfitting

Overfitting occurs during model training at the point when the model performs well on training data but generalizes poorly to unseen data. This type of challenge is a

very common in Machine Learning and several research efforts have been made to finding solution to it. Some of the techniques include:

1. Hold-out: this can be achieved by simply splitting of the training data into two sets (training and testing) rather than using the entire data for training. A common split ratio in most machine learning projects is 80% for training and 20% for testing. But in this proposed model, 75% for training and 25% for testing ratio was adopted because of its better performance in training of the model. The model is trained until it performs well both on the training set but also for the testing set. This indicates good generalization capability since the testing set represents unseen data that were not used for training. However, this approach requires sufficient dataset to train on even after data splitting.
2. Feature selection: this has to do with careful determination of the most important features in a situation where there is limited amount of training samples with large number of features. By selecting the major features, the model does not need to learn for so many features that could lead to overfitting. With this method, the model can simply test out different features, train individual models for these features, and evaluate generalization capabilities, or use one of the various widely used feature selection methods.
3. Data augmentation: A larger dataset would reduce model overfitting. This method is applicable where enough data cannot be gathered and are constrained to the current dataset. Data augmentation is applied to artificially increase the size of the dataset. For example, in case of training for an image classification task, various image transformations to the image dataset (e.g., flipping, rotating, rescaling, shifting) can performed.
4. Dropout: this form of regularization is applied on the network layers by ignoring a subset of units of the network with a set probability. Using dropout, learning among units likely to led to overfitting can reduce interdependency. However, with dropout, we would need more epochs for our model to converge.

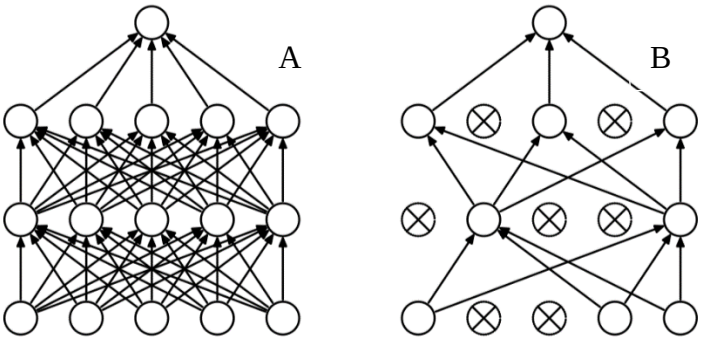
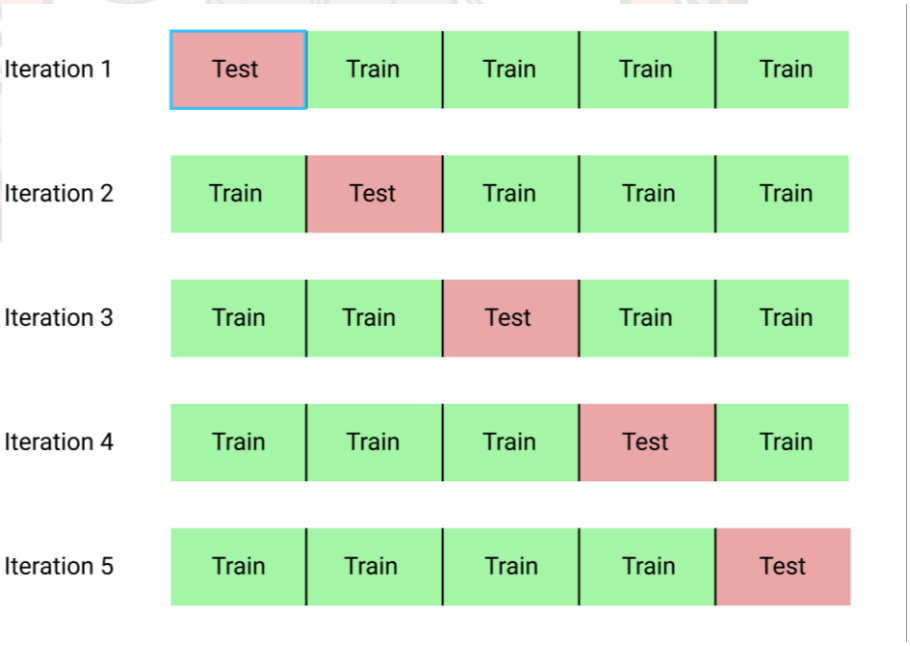


Figure 41 ((A) Neural Network before Dropout (B) After Applying Dropout)

5. Cross-validation: Dataset can be split into *k* groups (k-fold cross-validation) and setting one of the groups as the testing set and the others as the training set. This process is therefore repeated until each individual group has been used as the testing set (e.g., *k* repeats). Unlike hold-out, cross-validation allows all data to be used for model training even though it incurs more computationally expensive than hold-out.



6. L1 / L2 regularization: this type of regularization technique aims to constrain the network from learning a model that is too complex, which may lead to overfitting. In L1 or L2 regularization, a penalty term can be added on the cost function to push the estimated coefficients towards zero (and not take more extreme values). Also, L2 regularization allows

weights to decay towards zero but not to zero, while L1 regularization allows weights to decay to zero.

L1 Regularization	L2 Regularization
1. L1 penalizes sum of absolute values of weights.	1. L2 penalizes sum of square values of weights.
2. L1 generates model that is simple and interpretable.	2. L2 regularization is able to learn complex data patterns.
3. L1 is robust to outliers.	3. L2 is not robust to outliers.

7. Remove layers / number of units per layer: As mentioned in L1 or L2 regularization, there is high possibility of over-complex model overfitting. Therefore, model's complexity can directly be reduced by removing some layers which will result to reduction of the size of the model. Complexity can be further reduced by decreasing the number of neurons in the fully connected layers. However, to proffer solution to overfitting in a model, the model should have a complexity that sufficiently balances between underfitting and overfitting.

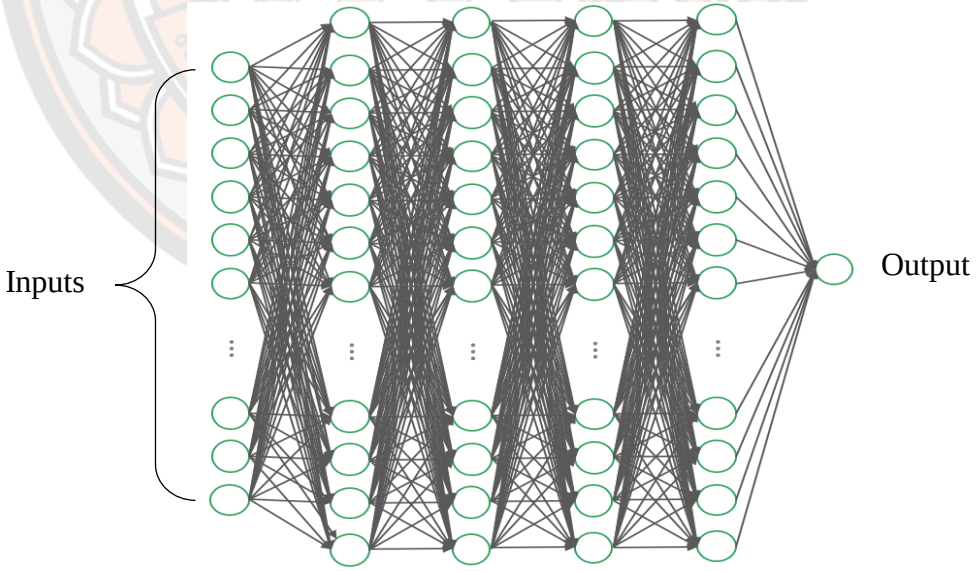


Figure 42 Densely Connected Neural Networks

Image Source: images.google.com

8. Early stopping: We can first train our model for an arbitrarily large number of epochs and plot the validation loss graph (e.g., using hold-out). Once the validation loss begins to degrade (e.g., stops decreasing but rather begins increasing), we stop the training and save the current model. We can implement this either by monitoring the loss graph or set an early stopping trigger. The saved model would be the optimal model for generalization among different training epoch values.

For the purposes of this model, collection of different methods such as dropout, hold-out, data augmentation, and early dropping were used for eliminating or minimizing overfitting and underfitting during model training.

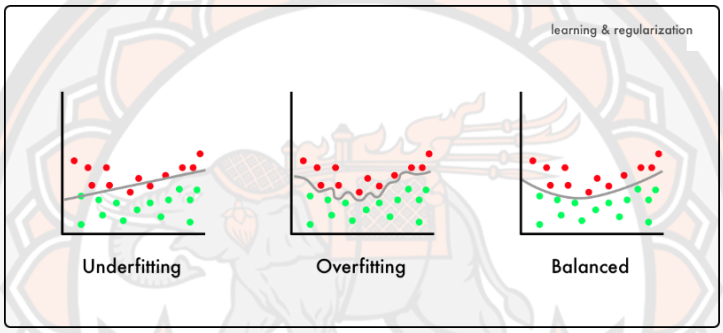


Figure 43 Different model learning behaviors during data training

5.5 Model Evaluation and Analysis

Firstly, an exploratory data analysis was carried out on the dependent variables in the dataset as shown in fig. 23 to determine the level of their correlation with the total active power that is forecasted. This statistical process is required because of the need to consider only relevant factors or variables that is directly related to total active power to be forecasted. Nevertheless, this relationship can as well be ascertained using normal data mining. Thereafter, the model is validated using Walk-forward validation approach to ascertain its skillfulness.

5.5.1 Performance Metrics

As stated earlier, the performance evaluation metric for assessing the 7-days ahead forecast of this model is based on three major standard error measurements: Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE) with special emphasis to prediction and

evaluation times. The reason for opting for these standard error measurements is because of the dynamism power forecasting and stochastic nature of neural networks models. Secondly, both error metrics were chosen because they use the same scale as the measured data; the error is of the same unit with the predictions and error can range from 0 to ∞. These metrics are negatively oriented scores, which means that lower values are better and are indifferent to the direction of errors.

5.5.2 Root Mean Square Error

Root Mean Square Error (RMSE) is a popular quadratic scoring rule that measures the average magnitude of the error. It is the square root of the average of squared differences between prediction and actual observation.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum\frac{n}{i=1}(x_i - y_i)^2}, \quad (1)$$

where x_i is the observed data, y_i denotes the predicted data, e_i is the arithmetic average of the absolute error, and n is the number of observations. This takes the variation between the predicted and actual values, square it up due to positive and negative difference that may arise, and obtain the mean to aggregate all the unseen data and finally square root it to counterbalance the square operations.

5.5.3 Mean Absolute Error

This type of error metrics also measures the average magnitude of errors in each set of predictions, without necessarily considering their direction. MAE is sometimes termed Mean Absolute Deviation (MAD) and it shows the magnitude of overall error in data points, in the cause of the forecast. It is the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight.

$$MAE = \frac{\sum_i^n \mid x_i - y_i \mid}{n} = \frac{\sum_i^n \mid e_i \mid}{n} \quad (2)$$

where e_i denotes the arithmetic average of the absolute error. This is also known as a scale-dependent accuracy measure, so it will be illogical to make comparisons between this metric and other series using different scales. The mean absolute error is commonly used for the measurement of forecast error in a time series analysis. It is less susceptible to outliers compared to MAPE and RMSE.

5.5.4 Mean Absolute Percentage Error (MAPE)

The mean absolute percentage error is the measure of the level of accuracy of a forecast system is in terms of percentage. Mean absolute percentage error (MAPE) is commonly used as a loss function for regression tasks and model evaluation because of how its interpretation in terms of relative error. It measures the forecast error and perform optimally if there are no extremes to the data (and no zeros). It can be calculated as the average absolute percent error for each time-period minus actual values divided by actual values.

$$MAPE = \frac{1}{n}\sum_{t=1}^{n} \left|\frac{A_t - F_t}{A_t}\right| \text{ x } 100 \quad (3)$$

where n is the number of fitted points, and A_t is the actual value, while F_t is the forecast value. The summation for the absolute value is done for every forecasted point in time and divided by the number of fitted points n. This performance metric is independent of the scale of measurement, yet it is affected by data transformation.

5.6 Home Energy Management System

Achieving the second objective of this thesis requires Human Machine Interface (HMI) like Home Energy Management System (HEMS) for real-time interaction between human users and their smart environment. This concept will go a long way in integrating cyber-physical system into smart grid technology. This HEMS expanded beyond energy savings to include convenience through remote controlling of appliances in a very smart way. However, most of the existing building/home automation systems are based on cloud computing. Because most cloud-based applications require fast processing and quick response to function effectively puts a little stunt in their applicability in energy management, considering latency. AI functionality inclusion in these systems require large amounts of processing power necessitating cloud computation, hence, incurring a huge cost burden in terms of computer resources, encounters latency and bandwidth problems.

To address this challenge, Google and Apple are now investing to run their AI algorithms on a user's device instead of cloud. Google recently came up with Progressive Web Apps with offline functionality, where works can be done without internet connections. All you are required to do is to save your work locally after

completion and synchronize with cloud at convenient time. But like I earlier mentioned, all these cloud-based devices have one challenge in common: network latency. Therefore, a smaller system that can run under the constraints of the edge-devices without compromising accuracy is key to network latency challenge. Hence, the justification to develop a Human Machine Interface (HMI) for energy management over an edge network, where computation is done locally either in an IoT device or edge sever or gateway.

In principle, the proposed HEMS is based on partial automation, where the real-time control decisions are taken by human users. And energy savings decisions are based on feedback from power demand forecast model just like [40]. Instead of using statistical method [41] to estimate the power usage for energy conservation and management, this thesis leveraged machine intelligence. For an instance, with the result of supervised deep learning algorithm [42], the residents already have the prior knowledge of the amount of energy expected to be used. This forecast result is therefore set as threshold for power consumption in the HEMS. As soon as energy consumption gets to a certain level close to this forecast value set as threshold, a notification is sent to a registered human user's mobile phone for proper decision making. The user will then decide whether to upwardly review the threshold or not. When nothing is done after the notification the system will start shooting down the appliances and this is where automation is achieved. Edge framework [43] is implemented locally to control the devices with the support of wireless communication techniques.

For performance evaluation, a case study of Thailand with her most recent energy statistics [44] as shown in table 4 is used. Considering factors such as weather (which is very hot almost all through the year in Thailand) and habit of the residents as regards to power usage; testing the potency of the proposed HEMS in energy saving in this environment is a good choice. From fig. 41 it can be noticed that load factor surged above generation in months of march and September; air-conditioner and refrigerator accounting for 26.50% and 19.33% of total load factor respectively, according to [41]. This prompted the expansion of the test period of this system to cover 12 months.

This proposed home energy management system expanded beyond energy savings to include convenience through remote controlling of appliances in a very smart way. However, most of the building/home automation systems in literature likewise similar applications such as self-driving car, virtual and augmented reality, smart cities etc. are based on cloud computing. The fact that these applications require fast processing and quick response to function effectively puts a little stunt in their applicability in energy management, considering latency. AI functionality inclusion in these systems require large amounts of processing power necessitating cloud computation, hence, incurring a huge cost burden in terms of computer resources, encounters latency and bandwidth problems. For the purposes of addressing this challenge, Google and Apple are currently focusing in running their AI algorithms on a user's device instead of cloud. Google has come up with Progressive Web Apps with offline functionality, where works can be done without internet connections. All that is required in this situation is to save the work locally after completion and synchronize with cloud at convenient time. Therefore, the objective of this study is to develop a HMI for energy management over an edge network, where computation will be done locally either in an IoT device or edge sever or gateway.

Table 4 Thailand Power Statistics, 2020

MONTH	PEAK (MW)	GENERATION (GWh)	LOAD FACTOR (%)
JAN	27,423	16,138	79.1
FEB	27,112	15,477	82.0
MAR	28,637	17,618	82.7
APR	27,747	15,715	78.7
MAY	28,328	16,899	80.2
JUN	27,240	15,887	81.0
JUL	26,855	16,390	82.0
AUG	27,235	16,348	80.7
SEP	27,159	16,195	82.8
OCT	26,162	15,475	79.5
NOV	27,433	15,292	77.4
DEC	25,924	14,500	75.2
TOTAL	28,637	191,935	76.3

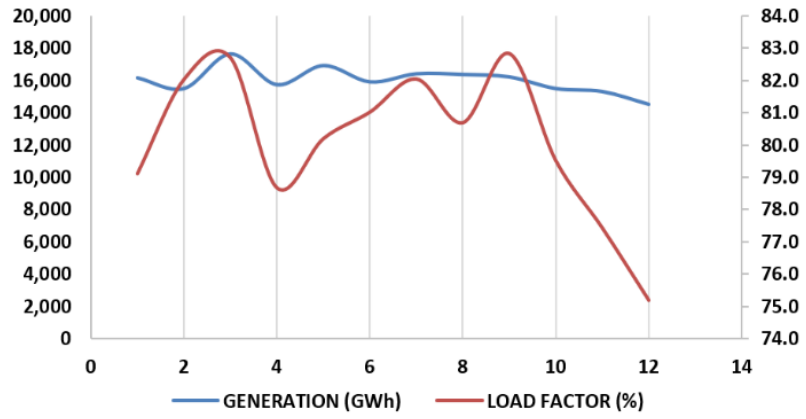


Figure 44 Plot of Power Statistics of Thailand, 2020

Figure 45 A typical Smart Home

5.6.1 HEMS Data Communication

Ordinarily, IoT switch has the capacity to perform the function of ON/FF remotely including ON/OFF schedules by timer or loop timer upon setting. Here, IoT switch is connected to all home appliances including the smart meter to enable real-time data transmission, monitoring and control. The real time ´Total Active Power´ transmitted to the cloud server can be viewed either through a mobile app or webpage as shown HEMS schematics of fig. 46. The process of data communication is based on Pub/Sub technique using machine-to-machine protocol; where real time data is published or subscribed simultaneously in a seamless way.

Data is published or subscribed in a bidirectional way as “Topic” just as described in fig. 48 with acknowledgment of its receipt at the completion of each process.

In the mobile app, the forecast value will be used as control benchmark, and be compared with the real time power value emanating from the smart meter. There is an option of sending a notification via text SMS within a set threshold to the user registered mobile phone. Ultimately, a notification will be sent as soon as the 90% of benchmark is about to be attended to enable the human user to determine whether to do upward review of the benchmark or simply allow the HEMS to shutdown automatically all appliances consuming significant amount of power. This is important because we do not want our forecast value for the day to be exceeded, therefore, power consumption rate must be managed with high level of prudency.

5.7 Cyber-physical System Integration into Smart Grid

One can insinuate that mass proliferation of AI-powered IoT devices today serving different purposes was enabled because of the pervasiveness of ultra-low-power embedded devices, coupled with the emergence of embedded machine learning frameworks like TensorFlow Lite for Microcontrollers. In this work, some of these emerging technologies were leveraged for the purposed of connectivity between the proposed forecast model, smart meter and on-device based system for data transmission for real-time monitoring and control.

5.7.1 Application of Tiny Machine Learning (tinyML)

Tiny machine learning (tinyML) is an evolving technology that acts as an intersection between machine learning and embedded internet of things (IoT) devices. The cost, bandwidth, and power constraints of deep learning models implementation necessitated running of models in microcontroller. Therefore, tinyML is very useful to systems that require direct connection to the power grid, frequent charges, or replacement of the battery. It is projected that this emerging engineering field can revolutionize many industries and it is most beneficial to edge-based computation devices. Looking at the growth trajectory of the number of connected devices to internet. [45] suggested that applying tinyML and ML in embedded devices will widen access since both leverage on low-cost and globally accessible hardware, and with complete, self-contained applications development, from data collection to

deployment stage. To reduce the model size, quantization of weights and biases can be done to reduce their default floating size (32-bit), so that they fit into 8-bit integers—a four times reduction in size. [46] quantized weights and biases precision by 224 bytes smaller than the original version. TinyML projects can be built just with Arduino and ultra-low-power microcontrollers. This emerging technology has potentials to optimize latency, energy usage, and model and binary size.

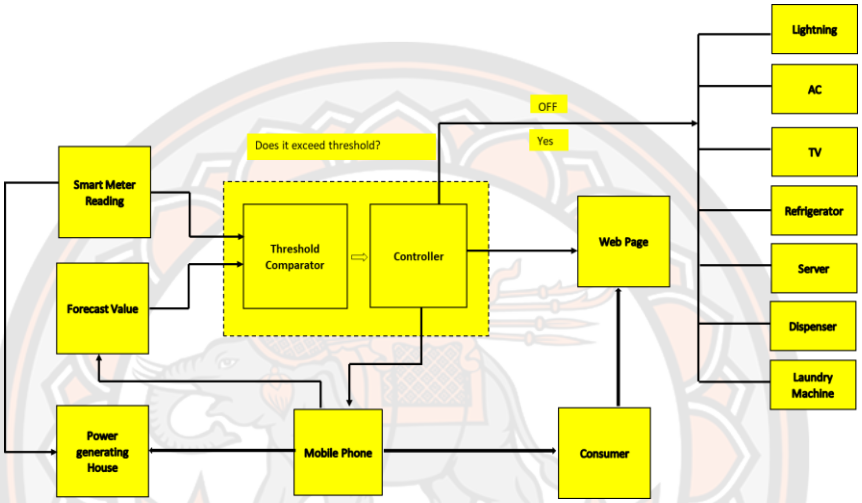


Figure 46 Schematics of the proposed HEMS (arrows show the data transmission direction)

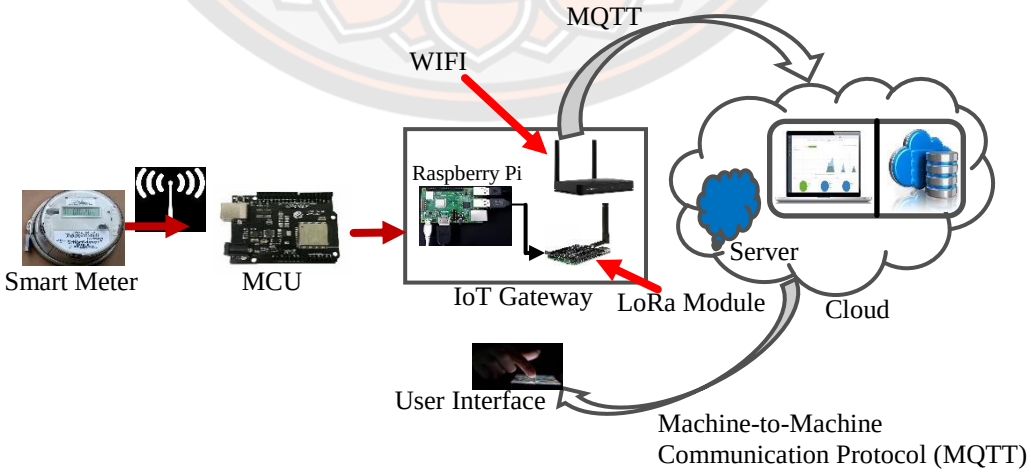


Figure 47 Data Transmission Process

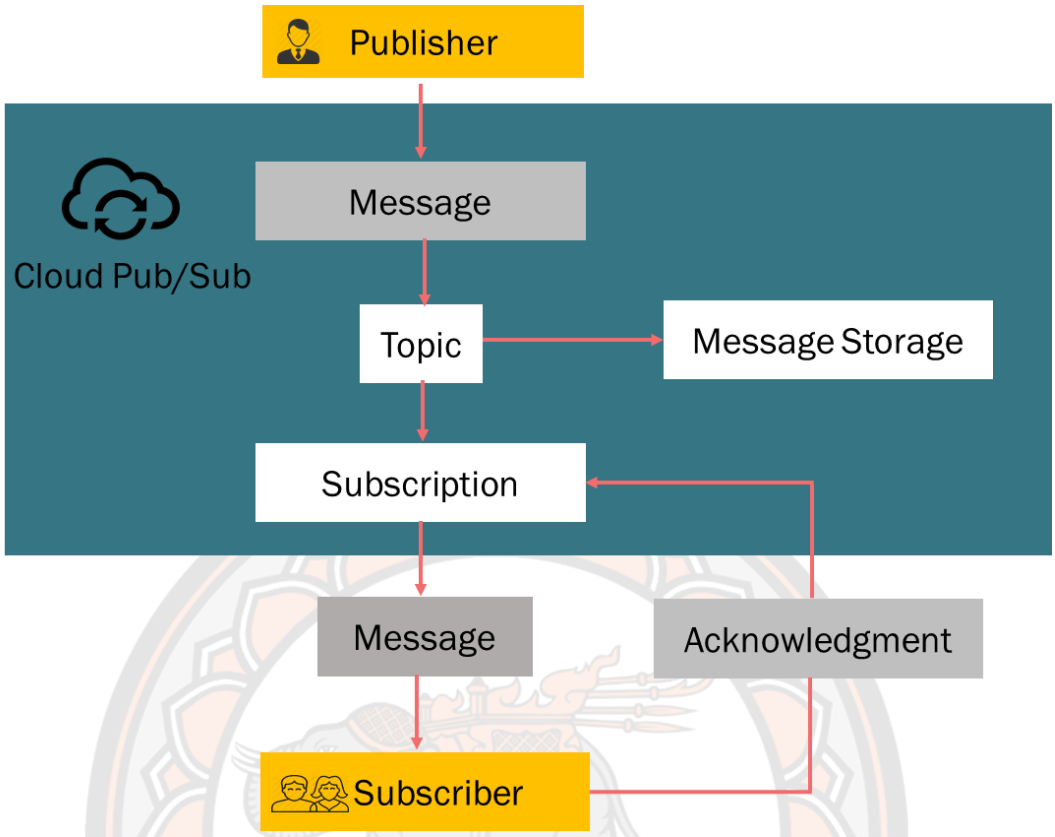


Figure 48 Communication configuration for Messaging/queueing in Cloud Pub-Sub

5.7.2 Pub/Sub for Data Transmission Framework between Smart Meter and HEMS

Appendix C demonstrates the capabilities of the pubsub library in combination with the ESP32 board/library. This bidirectional communication concept connects to an MQTT server then: publishes "Real-Time Power Consumption Values" from the smart meter to the topic "outTopic" every two seconds as well subscribes to the topic "inTopic" and printing out any messages it receives. This is done in assumption that the received payloads are strings rather than binary. Implementation of this is on ESP32 Board using Arduino 1.8.10 version with all the dependent libraries. It was made to switch ON the ESP LED if the first character of the topic "inTopic" is a 1, else it switches it OFF. And it also has the capacity to reconnect to the server if the connection is lost by using a “blocking reconnect function”.

5.8 MQTT

This is a lightweight communication protocol for sensors and mobile devices that is based on Transmission Control Protocol (TCP). It is chosen as the ideal protocol because of some factors which include low bandwidth, high reliability, and usefulness in an unstable network. Also, for the purposes of energy saving in the sensors and other IoT devices, data is sent every minute even though they generate data every second. In the initial design, Bluetooth module was used for communication between the HEMS and the IoT server. Though it performed very well, but unfortunately limited to 100meters coverage which does not satisfy the complete objectives of our study. Even though residents can easily maintain social distancing within this coverage, but we are looking at bigger pictures of monitoring and controlling our appliances from any part of the world.

Raspberry Pi was used in hosting the MQTT server (named Broker), where the microcontroller (ESP32) subscribes to MQTT event via MQTT Broker and publishes the granular data to the HEMS, which can as well be viewed via MQTT Dash. Data received from the sensor over time is stored by Raspberry Pi in a dedicated cloud time series database and hard drive persistently. From the messaging/queuing of fig. 48, every published message’s values are automatically persisted to database.

CHAPTER VI

EXPERIMENTAL RESULTS AND DISCUSSIONS

6.1 Experimental Results

A real-time experimentation using Google Colab TPU and one of the finest neural network APIs, contained in Keras® with its backend TensorFlow produced the results of table 5. Based on validation accuracy result, this model significantly outperformed the baseline model. Though unstable training trajectory was experienced during training, which could be likened to overfitting in the training data, but the overall performance is good based on validation. Fig. 50 showed that training error decreases sharply after commencement of training before it became linear because of the model's complexity, likewise the validation error. A squeeze layer technique [47] adopted reduce the size of the model to 4.9M without affecting its performance, making it implementable in a low-power-low-memory device like smartphone. One of the limitations observed from this technique is that as the model size is reduced, the number of parameters slightly increased, resulting to marginal increase in computer resources usage as relates to implementation. Therefore, further work is proposed to develop a systemic method of reducing the model size without necessarily increasing the number of model parameters.

Table 5 Experimental Results

Model Statistics	**Training on Simulated HHPC Dataset France**		**Training on Real-Time Dataset, SGtech, Thailand**			
	Propose Model	**Baseline Model**	**Propose Model**	**Persistence Model**		
				Hourly	**Daily**	**Weekly**
Training Time	114.109s	-	78.916	-	-	-
Prediction Time	2.282s	-	2.053	-	-	-
RMSE	***3.61.885kmh***	465.294kmh	***358kmh***	480.246	469.389	465.294

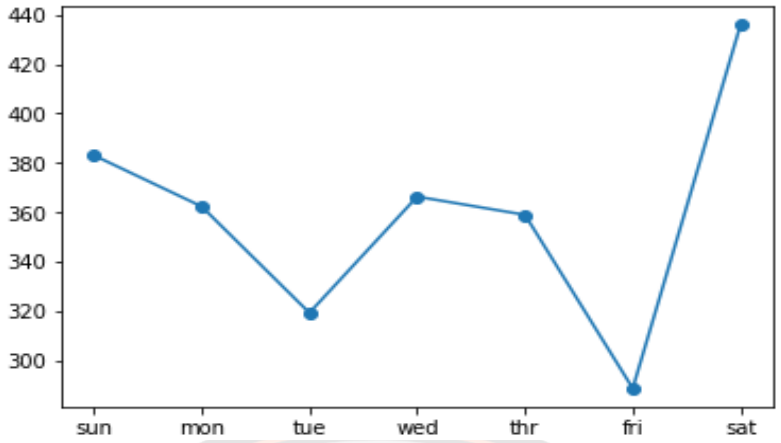


Figure 49 A 7-days power consumption forecast result

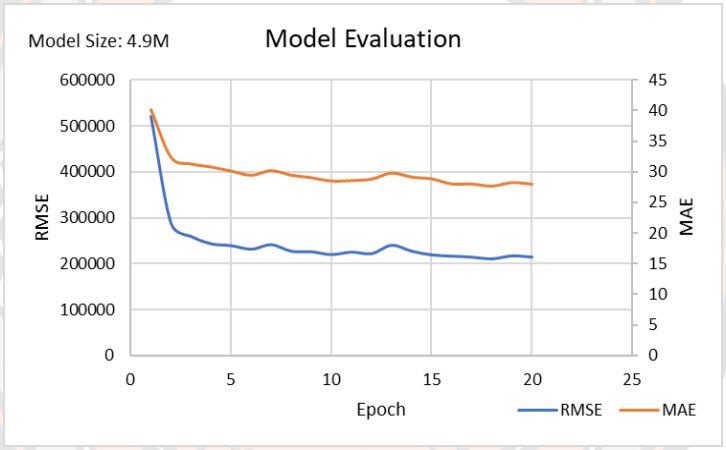


Figure 50 Plot of the model evaluation using RMSE and MAE

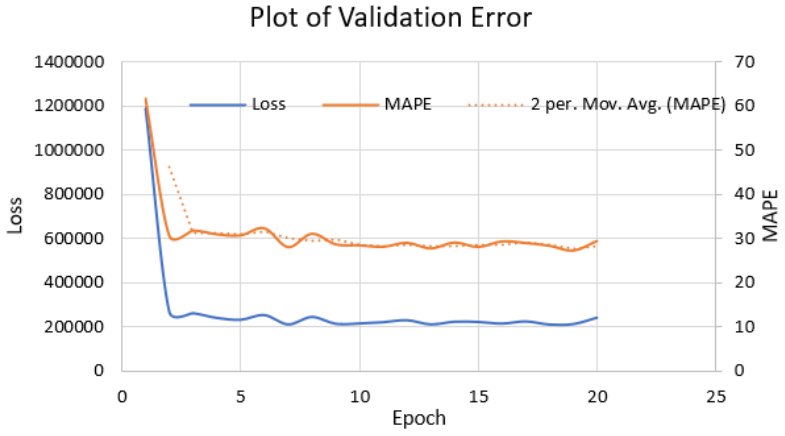


Figure 51 Plot of the model evaluation based on MAPE

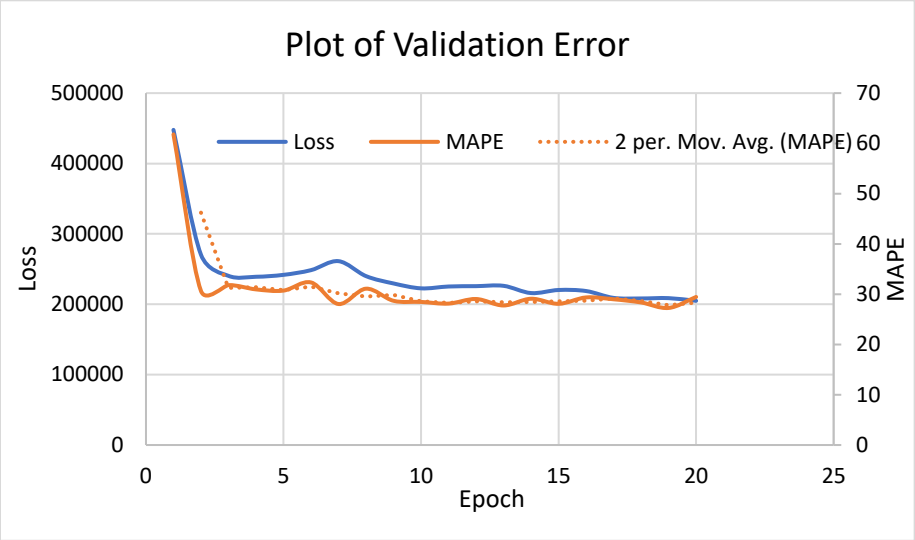


Figure 52 Plot of the model evaluation of MAPE against Model Loss

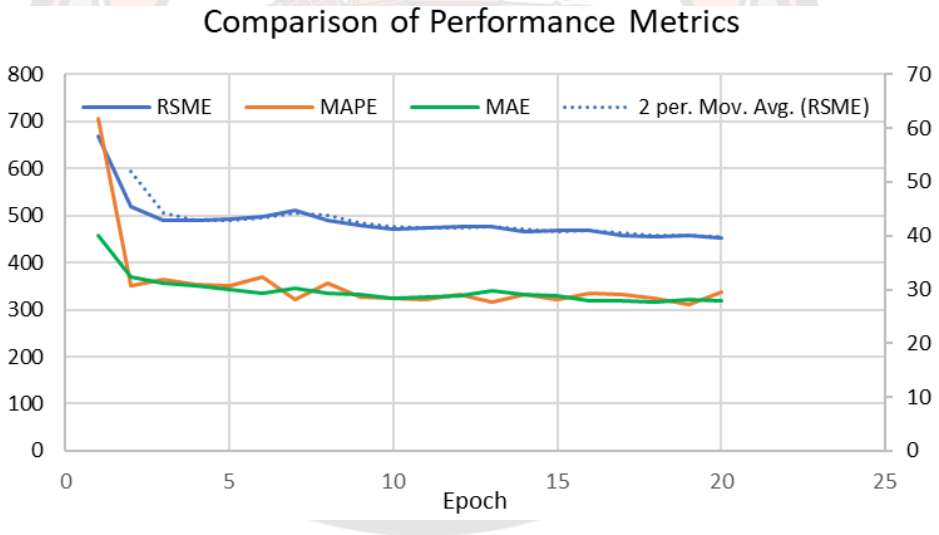


Figure 53 Plot of the model evaluation Comparing RMSE, MAPE and MAE

6.2 Discussions

A statistical or hypothesis testing was carried out on the experimental results to draw an inference on whether the more the previous time-steps from the historical data (input data) is applied to forecast model, the better the model performance. Based on this, the first validation test was ran using two (2) number of subsequences (i.e., n_steps = 2) and a length of subsequence n_length = 1. The length of sequence in this case is the number of prior days used as input. So, the total days used as input is

calculated as n_input = n_length * n_steps. This process was progressed until n_length = 14. As the length of sequences is varied upwardly the error values kept reducing downwardly, hence the 14days previous time steps used for model validation. This process of selecting the best parameters for the model is referred to as model tuning. Ordinarily, a more automatic way of selecting the best model parameters would have been “Grid Searching” but for the fact that the dataset used is very big, grid searching is inappropriate, hence, the trial-and-error method which is very time consuming. It is time consuming in the sense that it requires so many experiments with different hyperparameters.

Fig 49 summarizes the model performance using a single score, that is RMSE over the forecast days and comparison was later made against other evaluation metrics such MAE and MAPE shown in fig 53. All the errors are measure in Kilowatts because of the unit of Total-Active-Power being forecasted. Fig 49 showed that the model performed a better forecast on Tuesdays and Fridays than other days. Perhaps, Saturday being the end of the normal standard week seems to be the hardest day for the forecast.

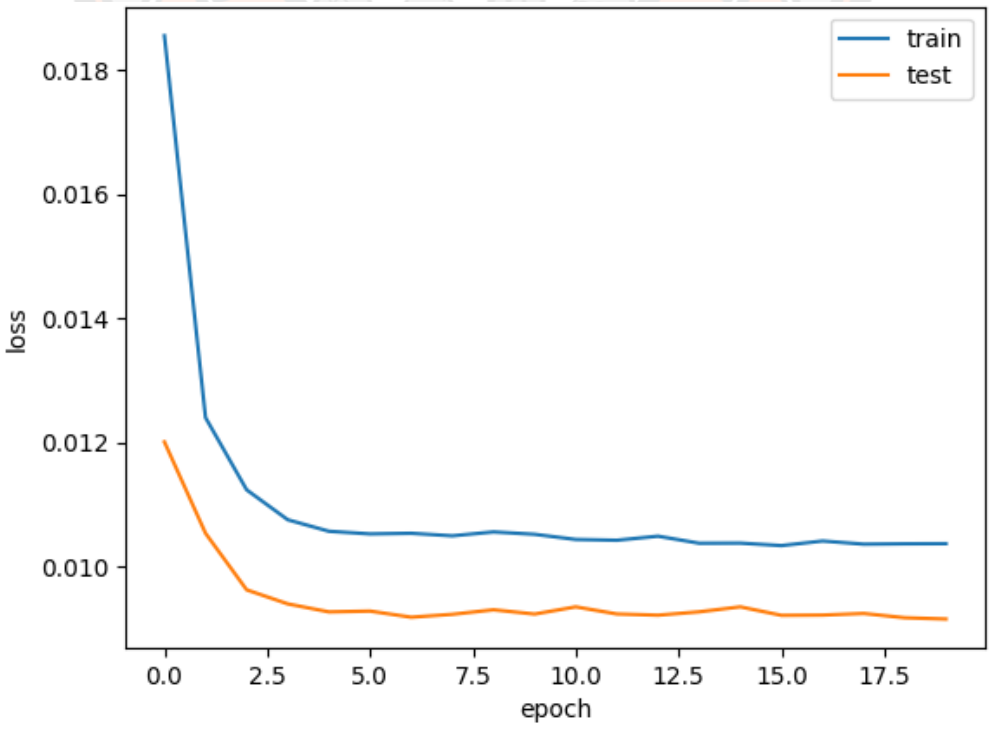


Figure 54 Model Training/Validation Loss

6.2.1 Appropriateness of the Model for On-device System

This model size is 4.93MB making is acceptable to every app stores. [48] expected Mobile Apps published on the app stores to have an average Android App file of 11.5MB and average iOS app file of 34.3MB. Also, Google Play Store recommended a file size as small as 5MB for easy download and optimal functionality. One of the major disadvantages of large mobile app file size, is that it may be difficult or expensive for users to download it. Perhaps, a user may have an extremely small amount of disk space in his or her device.

6.3 HEMS Human-Machine-Interface Assessment

This HEMS app displays very greatly on all different android phones used during experimentation and is likely to be the same on different models of iPhone and iPad. In case it is not same, features like Auto Layout, Xcode storyboards, and SwiftUI, can automatically fit the interface elements and layouts display. However, some evaluation indices were considered in determining the likely user experience:

Latency: Several experimentations was conducted using different sensors connected over different network technologies to determine the best throughput and coverage for this energy saving scheme. Latency of the HEMS was tested by initiating a control operation from different smartphones (installed with the proposed HEMS) simultaneously and result compare against only 1 smartphone connected at a time. However, the result showed no significant delay from the point of sending the control command to the point of implementing it. It proves that latency did not affect the system when different connections were made simultaneously as against when it was made one-after-the-other.

Response Time: on the other hand, It was also discovered there was no significant difference in the processing time. The simulation result satisfies both the convenience need and energy efficiency of a typical smart home. It provides a greater graphic user interface (GUI) for accessing appliances from any location through a WI-FI or 3G/4G enabled on-device systems. Arduino Ethernet Shield was used to connect Arduino board to the internet. And its Wiznet W5100/W5200 ethernet chip technology provided the needed network (IP) stack for both TCP and UDP. All the

interconnected devices including the smart meter has a specific sketch for communication written with different Arduino libraries and linked to the IoT server hosted in smart meter and cloud server via the internet.

As earlier mentioned, the edge computing takes place in the IoT server. As individual load data is persistently transmitted to the server, the aggregated power consumption is compared with the threshold value set for daily energy management. The moment 90 percent value of the threshold is attended, a notification is sent to the registered user's cellphone. At this point, human decision-making opportunity comes to play. A choice to allow the system shutdown appliances consuming much energy to reduce the rate of consumption by itself or upward review of the threshold is made. When nothing is done after the notification the system will start shooting down the appliances and this is where automation is achieved. Another interesting aspect of our HEMS is the incorporation of AI based forecast model whose result is used as threshold value. This supervised learning model leverage on few trainable parameters (4.9M) to make a daily forecast of Total Active Power over a period of 1week. And finally, its computation is on edge alienating it from the troubles of network latency.

6.4 HEMS Comparative Analysis

The performance of this system was analyzed against the simulated results of the benchmark method [1] that has the same concept though with different methodology that used four (4) major performance evaluation metrics such as cost effectiveness, energy conservation potency, efficiency, and processing speed.

Power Analysis: Using a daily power demand profile of a typical household [49], and that of our staging environment (Thailand) [41] as standard rated daily power consumption. 8 appliances and 1 EV charging spot in the smart office were selected and used for the evaluation of the HEMS as shown in Table 6. Their results were compared based on performance before and after application of HEMS.

Table 6 Energy saving and cost consideration between the use of proposed HEMS and manually operated option

Appliance	Rated daily Power Consumption (W) without HEMS	Cost (Baht) Unit per kWh x Tariff (4.2 Baht)	Daily Power Consumption (W) with HEMS	Cost (Baht) Unit per kWh x Tariff (4.2 Baht)	Priority
Lighting	100	0.42	76	0.3192	1
Air-conditioner	1025	4.305	779	3.2718	2
Toilet seat	450	1.89	342	1.4364	3
Dish washer	1100	4.62	836	3.5112	4
Television	200	0.84	152	0.6384	5
Laptop	20	0.084	15.2	0.06384	6
Refrigerator	272.93	1.146	207	0.8694	7
Electric Kettle	2000	8.4	1520	6.384	8
PEV	1200	5.04	912	3.8304	9
Total	**6, 347.93**	**26.745**	**4, 839.2**	**20.3246**	

Average household power demand of a typical home in a normal scenario was further calculated and compared against the power consumption from SGtech smart office. And the total energy after application of the proposed HEMS shown. Table 6 showed the comparison of power consumption and energy cost over appliances listed in their order of priority (in an office setting) using proposed HEMS against manually operated option. A use case of Thailand [41] projected Thailand energy saving potential at 13.97% on the condition that all old appliances is replaced with the higher efficiency ones. Therefore, the comparison of the result of the energy saving potentials of this proposed HEMS with this use case at the same location even though their methodologies slightly differ is very important.

Cost Analysis: Monthly load profile of School of Renewable Energy and Smart Grid Technology (SGtech), Naresuan University (test/staging environment) for an instance is estimated at about 250 – 400kWh corresponding to 4.2 – 4.4 baht/kWh tariff of provincial energy agency of Thailand. Time of Use Rate (ToU) is stipulated at 5.1135 and 2.6037Baht/kWh for peak and off-peak hours respectively at voltage level of 22 – 33kv, and 5.7982 and 2.6369Baht/kWh for peak and off-peak hours respectively at voltage level lower than 22kv. Therefore, to justify the cost effectiveness of the proposed system, the cost of energy in the smart office is calculated based on the minimum tariff level i.e., 4.2 baht/kWh on a normal monthly

rate. If electricity cost computation were to be based on Time of Use, then peak and off-peak energy usage period would be used for the calculation, which will result into slight variation in the total cost.

Generally, on-peak hours are between 7am until 11pm on weekdays, and off-peak hours between 11pm to 7am on weekdays. It is all day on Saturdays and Sundays even though utilities definitions from one environment to another can vary. From fig. 55 it was shown that peak demand of SGtech (i.e., when energy demand is highest and price higher) is between 3 – 5pm during the weekdays. It also showed that power demand between the office working hours of 9am – 5pm; amounts to 75% of the total energy usage which is within the acceptable level except for some few hours to work closing time. However, power demand of most of the remaining hours of the day is at off-peak hours.

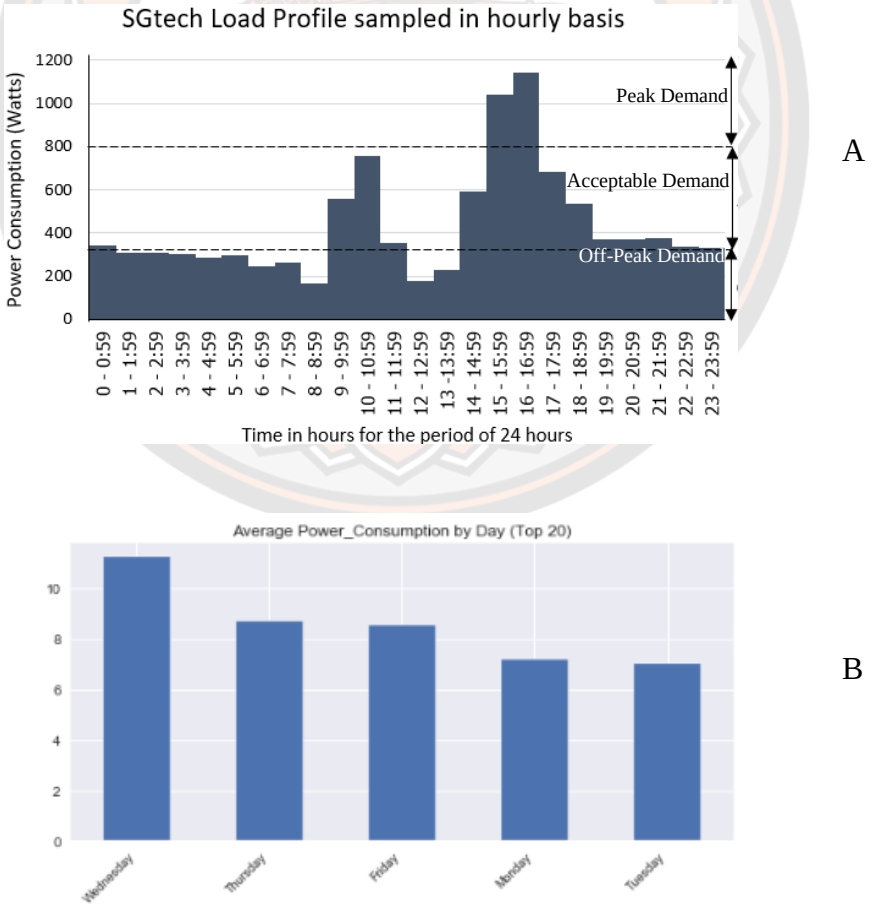


Figure 55 Energy Consumption of SGtech, Naresuan University, Thailand: (A) Showing Day Load Demand, (B) Average Consumption by Day

These statistics helps to determine when residents are likely to incur huge electricity cost in smart homes or cities, hence, adjust their usage behavior either through shifting of appliances that consume much energy to off-peak periods or ensure that appliances that are not in use at any given time is shutdown to avoid unaccounted power drain.

6.5 Research Instruments and Development Environment

The implementation of this forecast model was achieved using the finest neural network APIs contained in Keras® with its backend TensorFlow designed purposely for fast experimentation with deep neural networks. TensorFlow is an end-to-end open-source software published by Google for development and training of machine learning models. It is currently the most widely used deep learning library due to its fast-computational speed. The model training and validation in this research was done on Google Colab (a free Jupyter notebooks based computational cloud server for AI developers and researchers) using Intel(R) Core (TM) i3 CPU M 330 @ 2.13GHz, 2133 MHz, 2 Core(s), 4 Logical Processor(s).

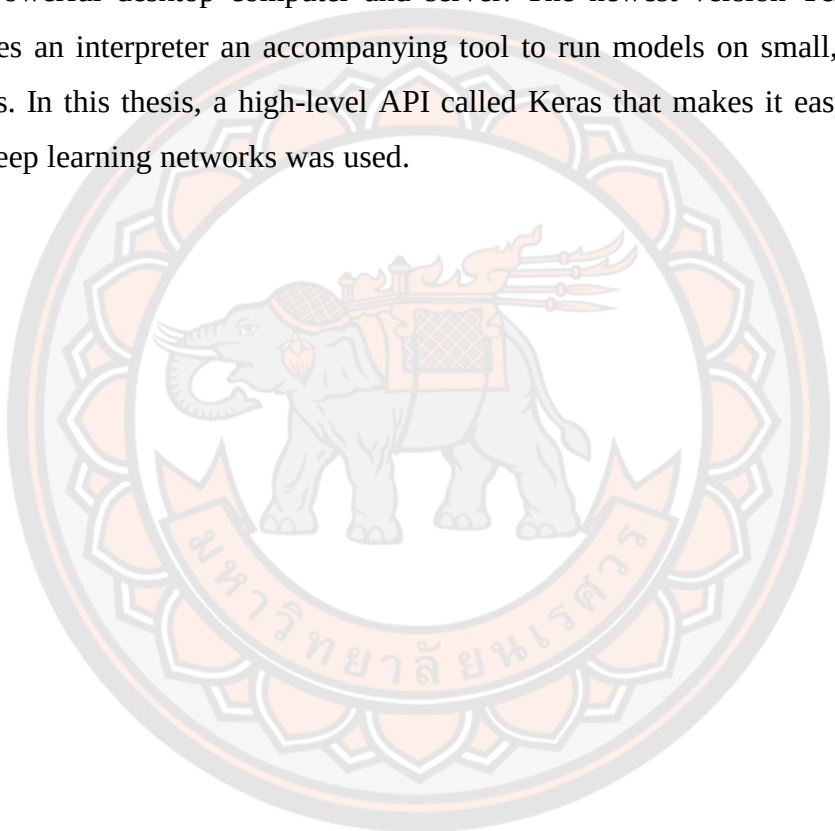
Google Colab is this machine learning hardware that supports free Graphics Processing Unit (GPU) and Tensor Processing Unit (TPU) networks computations and, it works perfectly well in python version used in the implementation of this forecast model. The recent achievement recorded through a cloud computing service competition by NVIDIA and Google showed TPU has a more speed than GPU and that informed the choice of TPU used here unlike commonly used GPU [50] etc. however, the fact that this custom-designed machine learning ASIC is integrated in already running server makes its implementation inexpensive, and readily available for rental on the cloud.

6.5.1 TensorFlow

TensorFlow is coined from an important physics terminology called tensor. Tensor is an important tool that provides mathematical framework for formulating and solving electrodynamics, mechanics, and general relativity problems. In mathematics, tensor describes multilinear relationship existing between sets of algebraic objects relative to vector space. It can map between scalar and vector objects, multilinear objects and even between operations such as dot product. As

vector spaces changes during transformation, its components also change. This describes its role in mapping input and targeted output relationships in deep learning models. However, these components can respond distinctively into covariance and contravariance because of change of basis.

However, TensorFlow used in this forecast model is essentially a set of instructions that tell the interpreter how to transform data to produce an output. When model is loaded into memory for execute, the TensorFlow interpreter runs this model on a powerful desktop computer and server. The newest version TensorFlow Lite provides an interpreter an accompanying tool to run models on small, low powered devices. In this thesis, a high-level API called Keras that makes it easy to build and train deep learning networks was used.



CHAPTER VII

CONCLUSION AND FUTURE

This chapter summarizes the contributions made in both the building of a power consumption forecast model for demand side energy users and development of home energy management system to complement the efforts made towards energy conservation and resource planning. It went further to provide ideas that could extend the current research work and finalizes with some suggestions for future work.

7.1 Conclusion

In this thesis, multi-step time series architecture for power consumption forecast was developed and tested on household electricity consumption dataset which is a secondary data with time domain dominant features. The multivariate time series dataset used in this work contain power-related variables useful in modeling future electricity consumption. This informed its preference even though it was augmented with a real-time data from an automated office. This study demonstrates the effectiveness of combining atmospheric climate domain knowledge of factors determinant to power consumption and associated empirical data through a spatiotemporal feature engineering method. Real-time data collated from School of Renewable Energy and Smart Grid Technology (SGtech) was used for model validation. It utilized a Deep Learning technology to model power consumption behaviors over different atmospheric environments and used the learned knowledge to make a week ahead power consumption forecast as a foundational step towards future advancement and dynamic forecast modeling.

Power Forecast Model: Interestingly, the traditional means of estimating power usage through previous utility bills is nowadays being replaced with machine intelligence. Therefore, this work proposed a neural networks architecture for demand side power consumption forecasting. This forecast model leveraged on multivariate dataset to make a multi-step time series (7days ahead) forecast. The forecasting model is based on ConvLSTM-Encoder-Decoder algorithm explicitly designed to enhance the quality of spatiotemporal encodings throughout the feature extraction process.

Randomness and other challenges of training neural networks necessitated the ensemble approach used, where multiple models were trained but only allow each model's contribution to prediction to be weighted proportionally to its level of trust and estimated performance. The result from the validation report showed a significant improvement on the forecast result when a real-time data from an automated office was used for model training against a manually operated home represented by the secondary data. This implies that aside social behavioral factor that propels the users' choice of time to use electricity, environmental and real-time control factors are also contributory factors that determines the rate power is consumed at homes and offices.

RMSE of 361kwh was recorded compared with 465kwh on persistence model and an improved RMSE of 358kwh was achieved when validated in holdout validation data from the automated office. However, overall performance on error, forecast time and computational speed was later compared with research efforts in literature and the result obtained showed a significant improvement.

Suitability for On-device Systems: This model is most appropriate for low-powered devices with low processing power and storage capabilities like smartphones, tablets and iPads considering the size reduction achieved with Squeeze Layer technique. With this tinyML concept, machine learning models for embedded devices with resource constrained challenges can be optimized. Ordinarily, traditional machine learning models' deployment to embedded device is difficult because embedded devices run on batteries and has limited processing power and memory availability.

Energy Management Potentials: Analysis result showed that the proposed HEMS reduced the daily average power consumption of SGtech estimated at 8.670kW down to 6.589kW. By extrapolation, applying this energy saving scheme across 25.23million households in Thailand for an example would reduce Thailand's annual power consumption significantly from the current 45, 205GWh [51] to 34, 355.8GWh. The total rated daily power of table 2 differs from estimated daily power consumption of SGtech because of appliance selection. From the comparative analysis of table 2, only 9 appliances were used in our smart office in their order of importance, and 24% energy reduction recorded on application of our proposed HEMS.

The difficult situation we found ourselves in this pandemic period; locking up indoors most times as a way of stopping the spread of deadly coronavirus has made energy conservation and planning more relevant than ever. Nowadays, we consume much energy with higher cost burden and stand the risk of spreading the disease in our various homes if we do not maintain social distancing, and continue sharing home gadgets such as remote control etc. Luckily, this thesis succeeded in encapsulating all the efforts made towards achievement of the stated research objectives.

7.2 Future Work

Popular opinion has it that PhD is an unending academic journey, hence, there are always room for future work so that existing research efforts can be consolidated. In this chapter, potential ideas for future works are stipulated. Interestingly, field of research is evolving with new concepts and solutions emerging, so, perfect efficiency is unattainable (Grossman and Stiglitz, 1980).

However, the potential new avenues for the application of intelligent techniques are in the areas of algorithmic data learning, Web Apps with offline functionality. Machine learning could potentially be used to improve algorithmic learning in such a way that can guarantee a systemic model size reduction without necessarily increasing the number of model parameters.

On the hand, future work can also be centered on expanding the Human Machine Interface (HMI) web application to function offline. It is one thing to implement an edge computing system and it is another to make its operation seamless through offline functionality.

Finally, further efforts can be made in finding ways to bring AI capabilities to microcontrollers that power most of the emerging consumer electronic devices. Combining two popular techniques: AutoML and TinyML will go a long way in bringing AI to microcontrollers, so efforts should be made in this regards to ensure models are run on the smallest unit microcontroller based electronic devices seamlessly.

REFERENCES

1. A. Carvallo, J.C., *The Advanced Smart Grid: Edge Power Driving Sustainability.* 2011.
2. Pawar, P. and P. Vittal K, *Design and development of advanced smart energy management system integrated with IoT framework in smart grid environment.* Journal of Energy Storage, 2019. **25**: p. 100846.
3. Makridakis, S., E. Spiliotis, and V. Assimakopoulos, *Statistical and Machine Learning forecasting methods: Concerns and ways forward.* PLOS ONE, 2018. **13**(3): p. e0194889.
4. Chandramitasari, W., B. Kurniawan, and S. Fujimura. *Building Deep Neural Network Model for Short Term Electricity Consumption Forecasting*. in *2018 International Symposium on Advanced Intelligent Informatics (SAIN)*. 2018.
5. Berriel, R.F., et al. *Monthly energy consumption forecast: A deep learning approach*. in *2017 International Joint Conference on Neural Networks (IJCNN)*. 2017.
6. Liang, F., et al., *Deep Learning-Based Power Usage Forecast Modeling and Evaluation.* Procedia Computer Science, 2019. **154**: p. 102-108.
7. Shi, X., et al., *Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting.* eprint arXiv:1506.04214, 2015: p. arXiv:1506.04214.
8. Ketkar, N., *Deep Learning with Python: A Hands -on Introduction.* 2017.
9. Sahani et al, *IoT Based Smart Energy Meter.* International Research Journal of Engineering and Technology (IRJET), 2017. **04**(04): p. 96-102.
10. Laudon, K.C. and J.P. Laudon, *Information Systems and the Internet*. 1998: Harcourt College Publishers.
11. Zhao, H.-z., F.-x. Liu, and L.-y. Li, *Improving deep convolutional neural networks with mixed maxout units.* PLOS ONE, 2017. **12**(7): p. e0180049.
12. Putchala, M.K., O.E. Theses, and D. Center, *Deep Learning Approach for Intrusion Detection System (IDS) in the Internet of Things (IoT) Network Using Gated Recurrent Neural Networks (GRU)*. 2017: Wright State University.
13. Mellit, A., et al., *A simplified model for generating sequence of daily global solar radiation data: Using artificial neural network and a library of Markov transition matrices.* Solar Energy - SOLAR ENERG, 2005. **79**: p. 469-482.
14. Marwala, L. and B. Twala. *Forecasting electricity consumption in South Africa: ARMA, neural networks and neuro-fuzzy systems*. in *2014 International Joint Conference on Neural Networks (IJCNN)*. 2014.
15. Nichiforov, C., et al. *Energy consumption forecasting using ARIMA and neural network models*. in *2017 5th International Symposium on Electrical and Electronics Engineering (ISEEE)*. 2017.
16. Haydari, Z., et al. *Time-series load modelling and load forecasting using neuro-fuzzy techniques*. in *2007 9th International Conference on Electrical Power Quality and Utilisation*. 2007.
17. Fahmi, F. and H. Sofyan. *Forecasting household electricity consumption in the province of Aceh using combination time series model*. in *2017 International Conference on Electrical Engineering and Informatics (ICELTICs)*. 2017.
18. Ahmed, K.M.U., et al. *Application of time-series and Artificial Neural Network models in short term load forecasting for scheduling of storage devices*. in *2014*

49th International Universities Power Engineering Conference (UPEC). 2014.
19. Din, G.M.U. and A.K. Marnerides. *Short term power load forecasting using Deep Neural Networks*. in *2017 International Conference on Computing, Networking and Communications (ICNC)*. 2017.
20. Dua, D.a.K.T., E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science, *Household Electricity Consumption Dataset.* 2017.
21. Du, S., et al., *Multivariate time series forecasting via attention-based encoder–decoder framework.* Neurocomputing, 2020. **388**: p. 269-279.
22. Asadi, R. and A.C. Regan, *A spatio-temporal decomposition based deep neural network for time series forecasting.* Applied Soft Computing, 2020. **87**: p. 105963.
23. Zhu, L. and N. Laptev, *Deep and Confident Prediction for Time Series at Uber*. 2017. 103-110.
24. Galicia, A., et al., *Multi-step forecasting for big data time series based on ensemble learning.* Knowledge-Based Systems, 2019. **163**: p. 830-841.
25. Mellit, A. and A.M. Pavan, *A 24-h forecast of solar irradiance using artificial neural network: Application for performance prediction of a grid-connected PV plant at Trieste, Italy.* Solar Energy, 2010. **84**(5): p. 807-821.
26. Huang, G., et al., *Snapshot Ensembles: Train 1, get M for free.* eprint arXiv:1704.00109, 2017: p. arXiv:1704.00109.
27. Qiu, X., et al. *Ensemble deep learning for regression and time series forecasting*. in *2014 IEEE Symposium on Computational Intelligence in Ensemble Learning (CIEL)*. 2014.
28. Chen, J., et al., *Wind speed forecasting using nonlinear-learning ensemble of deep learning time series prediction and extremal optimization.* Energy Conversion and Management, 2018. **165**: p. 681-695.
29. Caruana, R., et al., *Ensemble selection from libraries of models*, in *Proceedings of the twenty-first international conference on Machine learning*. 2004, ACM: Banff, Alberta, Canada. p. 18.
30. Ribeiro, M.H.D.M., et al. *Multi-Objective Ensemble Model for Short-Term Price Forecasting in Corn Price Time Series*. in *2019 International Joint Conference on Neural Networks (IJCNN)*. 2019.
31. Souza, L.V.d., et al. *An Empirical Study of Time Series Forecasting Using Boosting Technique with Correlation Coefficient*. in *Seventh International Conference on Intelligent Systems Design and Applications (ISDA 2007)*. 2007.
32. Mayrink, V. and H.S. Hippert. *A hybrid method using Exponential Smoothing and Gradient Boosting for electrical short-term load forecasting*. in *2016 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*. 2016.
33. Jiang, Y., et al., *Short-term wind power forecasting using hybrid method based on enhanced boosting algorithm.* Journal of Modern Power Systems and Clean Energy, 2017. **5**(1): p. 126-133.
34. Singh, S., A. Yassine, and R. Benlamri. *Internet of Energy: Ensemble Learning through Multilevel Stacking for Load Forecasting*. in *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl*

Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech). 2020.
35. Wolpert, D.H., *Stacked generalization.* Neural Networks, 1992. **5**(2): p. 241-259.
36. Izmailov, P., et al., *Averaging weights leads to wider optima and better generalization.* arXiv preprint arXiv:1803.05407, 2018.
37. Hu, J., L. Shen, and G. Sun. *Squeeze-and-Excitation Networks*. in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018.
38. Srivastava, S. and S. Lessmann, *A comparative study of LSTM neural networks in forecasting day-ahead global horizontal irradiance with satellite data.* Solar Energy, 2018. **162**: p. 232-247.
39. Nair, V. and G.E. Hinton, *Rectified linear units improve restricted boltzmann machines*, in *Proceedings of the 27th International Conference on International Conference on Machine Learning*. 2010, Omnipress: Haifa, Israel. p. 807-814.
40. Yousefi, M., et al., *Predictive Home Energy Management System With Photovoltaic Array, Heat Pump, and Plug-In Electric Vehicle.* IEEE Transactions on Industrial Informatics, 2021. **17**(1): p. 430-440.
41. Poolsawat, K., et al., *Electricity consumption characteristics in Thailand residential sector and its saving potential.* Energy Reports, 2020. **6**: p. 337-343.
42. Alexander N. Ndife, Y.M., Wattanapong Rakwichian and Paisarn Muneesawang, *A Smart On-device Based Power Consumption Forecast Model with an Optimized Weighted Average Ensemble Method.* unpublised journal submitted to IEEE Access, 2021.
43. Xia, C., et al. *Edge-based Energy Management for Smart Homes*. in *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*. 2018.
44. Energy Policy and Planning Office, T., *Energy Statistics of Thailand, 2020.* 2021.
45. Reddi, V.J., et al., *Widening Access to Applied Machine Learning with TinyML.* arXiv preprint arXiv:2106.04008, 2021.
46. Warden, P. and D. Situnayake, *Tinyml: Machine learning with tensorflow lite on arduino and ultra-low-power microcontrollers*. 2019: O'Reilly Media.
47. Nanfack, G., A. Elhassouny, and R.O.H. Thami, *Squeeze-SegNet: a new fast deep convolutional neural network for semantic segmentation*. Tenth International Conference on Machine Vision. Vol. 10696. 2018: SPIE.
48. Boshell, B. *Average App File Size: Data for Android and iOS Mobile Apps*. 2017.
49. Alfaverh, F., M. Denaï, and Y. Sun, *Demand Response Strategy Based on Reinforcement Learning and Fuzzy Reasoning for Home Energy Management.* IEEE Access, 2020. **8**: p. 39310-39321.
50. Coelho, I.M., et al., *A GPU deep learning metaheuristic based model for time series forecasting.* Applied Energy, 2017. **201**: p. 412-418.
51. Policy, E. and P. Officer, *Energy Statistics of Thailand 2018.* 2016.

APPENDIX

APPENDIX A MACHINE LEARNING/TIME SERIES GLOSSARY

ARMA (Autoregressive Moving Average) A statistical tool that models for a time series with no trend. It is a combination of both an autoregressive (AR) model and a moving average (MA) model. The model is usually then referred to as an ARMA (p, q) model where p is the order of the autoregressive part and q is the order of the moving average part.

ARIMA (autoregressive integrated moving average) A time series model like ARMA except that it is presumed the time series has a steady underlying trend. Therefore, it works with the differences between the successive observed values, instead of the values themselves. A nonseasonal ARIMA model is generally denoted ARIMA (p, d, q) where parameters p, d, and q are non-negative integers, p is the order of the autoregressive model, d is the degree of differencing, and q is the order of the moving-average model.

Autocorrelation A measure of the linear relationship between two separate instances of the same random variable

Deep learning A branch of machine learning that utilizes multiple processing layers to learn representations of data at the instance of multiple levels of abstraction.

Neural Networks A set of algorithms that learns underlying relationships in a set of data in a human brain-like manner. It is referred to as systems of neurons either organic or artificial in nature.

Tiny machine learning (tinyML) An evolving technology that acts as an intersection between machine learning and embedded internet of things (IoT) devices.

APPENDIX B MODEL SOURCE CODE

```
#!/usr/bin/python -tt
# load and clean-up data
from numpy import nan
from numpy import isnan
from pandas import read_csv
from pandas import to_numeric
from matplotlib import pyplot
import matplotlib.pyplot as plt
import seaborn as sns
import tensorflow as tf

# fill missing values with a value at the same time one day ago
def fill_missing(values):
        one_day = 60 * 24
        for row in range(values.shape[0]):
            for col in range(values.shape[1]):
                if isnan(values[row, col]):
                    values[row, col] = values[row - one_day, col]

# load all data
dataset = read_csv('household_power_consumption.txt', sep=';', header=0,
low_memory=False, infer_datetime_format=True, parse_dates={'datetime':[0,1]},
index_col=['datetime'])
# mark all missing values
dataset.replace('?', nan, inplace=True)
# make dataset numeric
dataset = dataset.astype('float32')
# fill missing
fill_missing(dataset.values)
# add a column for for the remainder of sub metering
```

```
values = dataset.values
dataset['sub_metering_4'] = (values[:,0] * 1000 / 60) - (values[:,4] + values[:,5] +
values[:,6])
# save updated dataset
dataset.to_csv('household_power_consumption.csv')

# resampling data over day, and show the sum and mean of Global_active_power
which have similar structure.
dataset.Global_active_power.resample('W').sum().plot(title='Global_active_power
resampled over week for sum')
#dataset.Global_active_power.resample('W').mean().plot(title='Global_active_power
resampled over week', color='red')
plt.tight_layout()
plt.show()

dataset.Global_active_power.resample('W').mean().plot(title='Global_active_power
resampled over week for mean', color='red')
plt.tight_layout()
plt.show()
# mean and std of 'Global_intensity' resampled over week
r = dataset.Global_intensity.resample('w').agg(['mean', 'std'])
r.plot(subplots = True, title='Global_intensity resampled over week')
plt.show()
## resampling over week and computing mean
dataset.Global_reactive_power.resample('W').mean().plot(color='y', legend=True)
dataset.Global_active_power.resample('W').mean().plot(color='r', legend=True)
dataset.Global_intensity.resample('W').mean().plot(color='g', legend=True)
dataset.Sub_metering_1.resample('W').mean().plot(color='b', legend=True)
dataset.Sub_metering_2.resample('W').mean().plot(color='black', legend=True)
dataset.Sub_metering_3.resample('W').mean().plot(color='brown', legend=True)
plt.show()
## The correlations between 'Global_intensity', 'Global_active_power'
```

```
data_returns = dataset.pct_change()
sns.jointplot(x='Global_intensity', y='Global_active_power', data=data_returns)
plt.show()
## The correlations between 'Voltage' and  'Global_active_power'
sns.jointplot(x='Voltage', y='Global_active_power', data=data_returns)
plt.show()
sns.pairplot(dataset[["Global_active_power", "Global_intensity", "Voltage",
"Global_reactive_power"]], diag_kind="kde")



# univariate multi-step encoder-decoder convlstm
from math import sqrt
from numpy import split
from numpy import array
from pandas import read_csv
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Flatten
from keras.layers import LSTM
from keras.layers import RepeatVector
from keras.layers import TimeDistributed
from keras.layers import ConvLSTM2D
from keras.layers.convolutional import MaxPooling2D

import h5py
import altair as alt
from pandas import to_numeric
from matplotlib import pyplot
from time import time
import numpy as np
import pandas as pd
```

```
import keras.callbacks as callbacks
from keras.callbacks import Callback
np.random.seed(1337)  # for reproducibility

# split a univariate dataset into train/test sets
def split_dataset(data):
        # split into standard weeks
        train, test = data[1:-328], data[-328:-6]
        # restructure into windows of weekly data
        train = array(split(train, len(train)/7))
        test = array(split(test, len(test)/7))
        return train, test
# resample minute data to total for each day
from pandas import read_csv
# load the new file
dataset = read_csv('household_power_consumption.csv', header=0,
infer_datetime_format=True, parse_dates=['datetime'], index_col=['datetime'])
# resample data to daily
daily_groups = dataset.resample('D')
daily_data = daily_groups.sum()
# summarize
print(daily_data.shape)
print(daily_data.head())
# save
daily_data.to_csv('household_power_consumption_days.csv')
# resampling data over day, and show the sum and mean of Global_active_power
which have similar structure.
dataset.Global_active_power.resample('W').sum().plot(title='Global_active_power
resampled over week for sum')
#dataset.Global_active_power.resample('W').mean().plot(title='Global_active_power
resampled over week', color='red')
plt.tight_layout()
```

```
plt.show()

dataset.Global_active_power.resample('W').mean().plot(title='Global_active_power
resampled over week for mean', color='red')
plt.tight_layout()
plt.show()
Var_Corr = daily_data.corr()
# plot the heatmap and annotation on it
sns.heatmap(Var_Corr, xticklabels=Var_Corr.columns,
yticklabels=Var_Corr.columns, annot=True)

# save
daily_data.to_csv('household_power_consumption_days.csv')
from time import time
import h5py

# evaluate one or more weekly forecasts against expected values
def evaluate_forecasts(actual, predicted):
        scores = list()
        # calculate an RMSE score for each day
        for i in range(actual.shape[1]):
            # calculate mse
            mse = mean_squared_error(actual[:, i], predicted[:, i])
            # calculate rmse
            rmse = sqrt(mse)
            # store
            scores.append(rmse)
        # calculate overall RMSE
        s = 0
        for row in range(actual.shape[0]):
            for col in range(actual.shape[1]):
                s += (actual[row, col] - predicted[row, col])**2
```

```
        score = sqrt(s / (actual.shape[0] * actual.shape[1]))
        return score, scores

# summarize scores
def summarize_scores(name, score, scores):
        s_scores = ', '.join(['%.1f' % s for s in scores])
        print('%s: [%.3f] %s' % (name, score, s_scores))

# convert history into inputs and outputs
def to_supervised(train, n_input, n_out=7):
        # flatten data
        data = train.reshape((train.shape[0]*train.shape[1], train.shape[2]))
        X, y = list(), list()
        in_start = 0
        # step over the entire history one time step at a time
        for _ in range(len(data)):
            # define the end of the input sequence
            in_end = in_start + n_input
            out_end = in_end + n_out
            # ensure we have enough data for this instance
            if out_end < len(data):
                x_input = data[in_start:in_end, 0]
                x_input = x_input.reshape((len(x_input), 1))
                X.append(x_input)
                y.append(data[in_end:out_end, 0])
            # move along one time step
            in_start += 1
        return array(X), array(y)

# train the model
def build_model(train, n_steps, n_length, n_input):
        # prepare data
```

```
	train_x, train_y = to_supervised(train, n_input)
	# define parameters
	verbose, epochs, batch_size = 1, 20, 8
	n_timesteps, n_features, n_outputs = train_x.shape[1], train_x.shape[2], train_y.shape[1]
	# reshape into subsequences [samples, time steps, rows, cols, channels]
	train_x = train_x.reshape((train_x.shape[0], n_steps, 1, n_length, n_features))
	# reshape output into [samples, timesteps, features]
	train_y = train_y.reshape((train_y.shape[0], train_y.shape[1], 1))
	# define model
	model = Sequential()
	model.add(ConvLSTM2D(filters=64, kernel_size=(1,3), activation='relu', input_shape=(n_steps, 1, n_length, n_features)))
	model.add(Flatten())
	model.add(RepeatVector(n_outputs))
	model.add(LSTM(100, activation='relu', return_sequences=True))
	model.add(TimeDistributed(Dense(100, activation='relu')))
	model.add(LSTM(100, activation='relu', return_sequences=True))
	model.add(TimeDistributed(Dense(1, activation = 'linear')))
	model.compile(loss='mse', metrics =['mse', 'mape', 'mae'], optimizer='adam')
	# fit network
	model.fit(train_x, train_y, epochs=epochs, batch_size=batch_size, verbose=verbose)
	model.save("model.hdf5")
	return model

# make a forecast
def forecast(model, history, n_steps, n_length, n_input):
	# flatten data
	data = array(history)
	data = data.reshape((data.shape[0]*data.shape[1], data.shape[2]))
	# retrieve last observations for input data
```

```
        input_x = data[-n_input:, 0]
        # reshape into [samples, time steps, rows, cols, channels]
        input_x = input_x.reshape((1, n_steps, 1, n_length, 1))
        # forecast the next week
        yhat = model.predict(input_x, verbose=0)
        # we only want the vector forecast
        yhat = yhat[0]
        return yhat

# evaluate a single model
def repeat_evaluate(train, test, n_steps, n_length, n_input, n_repeat=30):
        t0 = time()
        # fit model
        model = build_model(train, n_steps, n_length, n_input)
        tt = time() - t0
        print("Training time in {} seconds".format(round(tt,3)))
        # history is a list of weekly data
        history = [x for x in train]
        # walk-forward validation over each week
        predictions = list()
        t0 = time()
        for i in range(len(test)):
            # predict the week
            yhat_sequence = forecast(model, history, n_steps, n_length, n_input)
            # store the predictions
            predictions.append(yhat_sequence)
            # get real observation and add to history for predicting the next week
            history.append(test[i, :])
        # evaluate predictions days for each week
        tt = time() - t0
        print("Forecast time in {} seconds".format(round(tt,3)))
        predictions = array(predictions)
```

```
        t0 = time()
        score, scores = evaluate_forecasts(test[:, :, 0], predictions)
        tt = time() - t0
        print("Evaluate_forecasts time in {} seconds".format(round(tt,3)))
        return score, scores

# load the new file
dataset = read_csv('household_power_consumption_days.csv', header=0,
infer_datetime_format=True, parse_dates=['datetime'], index_col=['datetime'])
# split into train and test
train, test = split_dataset(dataset.values)
# define the number of subsequences and the length of subsequences
n_steps, n_length = 2, 14
# define the total days to use as input
n_input = n_length * n_steps
score, scores = repeat_evaluate(train, test, n_steps, n_length, n_input)

#Code for check size of the model 'name of the model' of model.hdf5
import os
print("Model size is " +str(os.path.getsize('model.hdf5')/1000000)+" MB")



# summarize scores
summarize_scores('lstm', score, scores)
# plot scores
days = ['sun', 'mon', 'tue', 'wed', 'thr', 'fri', 'sat']
pyplot.plot(days, scores, marker='o', label='lstm')
pyplot.show()
```

APPENDIX C SKETCH OF MACHINE-TO-MACHINE CONNECTION (MQTT) VIA ESP32 MICROCONTROLLER

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
// Update these with values suitable for your network.
const char* ssid = ".....";
const char* password = ".....";
const char* mqtt_server = "broker.mqtt-dashboard.com";
WiFiClient espClient;
PubSubClient client(espClient);
unsigned long lastMsg = 0;
#define MSG_BUFFER_SIZE  (50)
char msg[MSG_BUFFER_SIZE];
int value = 0;
void setup_wifi() {
  delay(10);
  // We start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  randomSeed(micros());
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
```

```
void callback(char* topic, byte* payload, unsigned int length) {
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
  }
  Serial.println();
  // Switch on the LED if an 1 was received as first character
  if ((char)payload[0] == '1') {
    digitalWrite(BUILTIN_LED, LOW);   // Turn the LED on (Note that LOW is the
voltage level
    // but actually the LED is on; this is because
    // it is active low on the ESP-01)
  } else {
    digitalWrite(BUILTIN_LED, HIGH);  // Turn the LED off by making the voltage
HIGH
  }
}
void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Create a random client ID
    String clientId = "ESP32Client-";
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if (client.connect(clientId.c_str())) {
      Serial.println("connected");
      // Once connected, publish an announcement...
      client.publish("outTopic", "hello world");
      // ... and resubscribe
```

```
      client.subscribe("inTopic");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}
void setup() {
  pinMode(BUILTIN_LED, OUTPUT);     // Initialize the BUILTIN_LED pin as an output
  Serial.begin(115200);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}
void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();
  unsigned long now = millis();
  if (now - lastMsg > 2000) {
    lastMsg = now;
    ++value;
    snprintf (msg, MSG_BUFFER_SIZE, "hello world #%ld", value);
    Serial.print("Publish message: ");
    Serial.println(msg);
    client.publish("outTopic", msg);
  }
}
```

APPENDIX D HOME ENERGY MANAGEMENT SYSTEMS' DEVELOPMENT PROCESSES

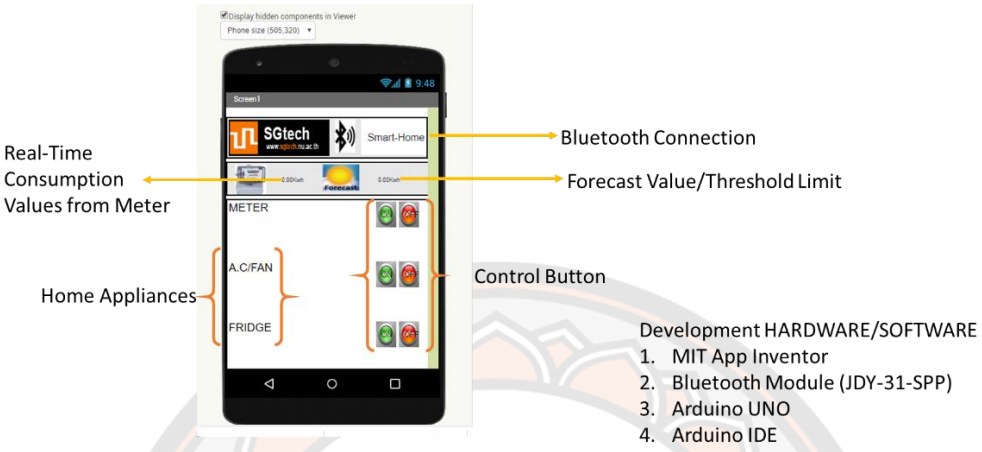


Figure 56 HEMS showing Operational Functions

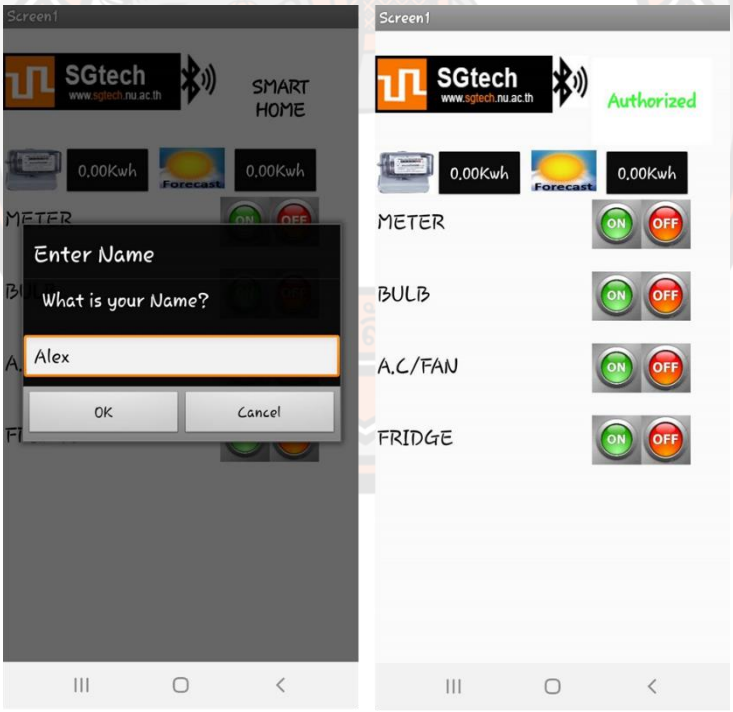


Figure 57 Authentication Process

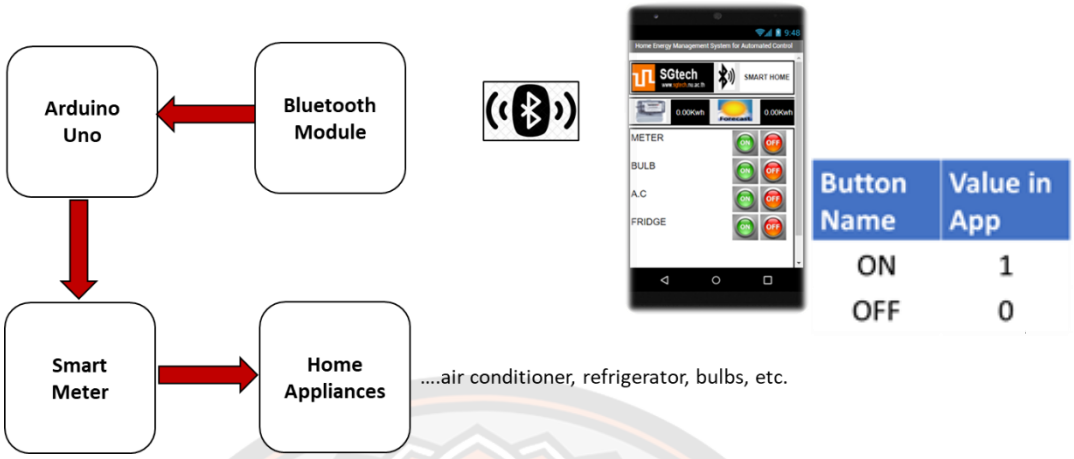


Figure 58 Energy Management Control Process

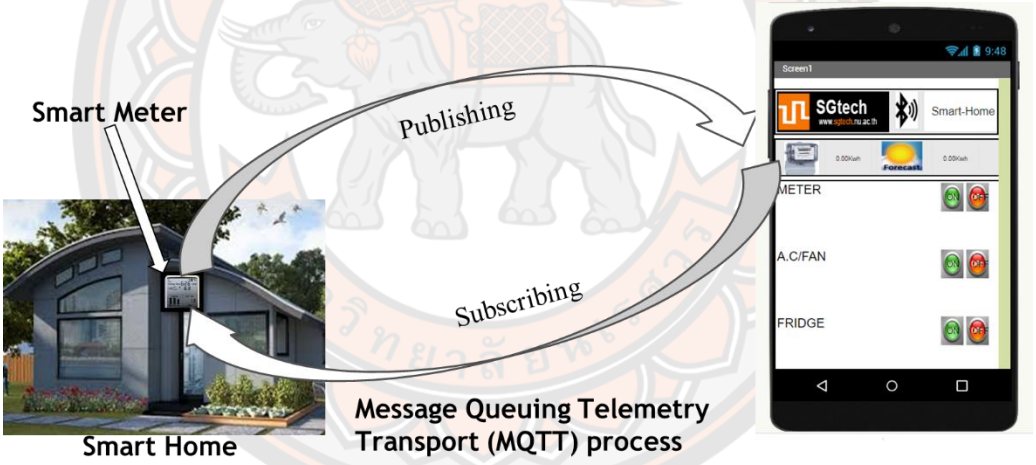


Figure 59 Machine-to-Machine Communiction Protocol

BIOGRAPHY

BIOGRAPHY

Name-Surname	Alexander Nnamdi Ndife
Date of Birth	
Address	
Current Workplace	School of Renewable Energy & Smart Grid Technology Naresuan University, Phitsanulok, Thailand.
Current Position	Research Assistant
Work Experience	- United Nations World Food Programme (WFP), IT Solutions Associate (Digital), Bangkok Thailand [2017 – 2018]. - Anambra Broadcasting Service (ABS), ICT Officer, Awka, Nigeria [2012 – 2015]. - Igbajo Polytechnic, Assistant Lecturer, Nigeria [2009 – 2010]
Education Background	- Ph.D. in Smart Grid Technology, Naresuan University, Thailand. [2018-2021] - M. Eng. in Electronics & Computer Engineering, Nnamdi Azikiwe University, Nigeria (GPA: 3.75/5.0) [2011-2014] - B.Eng. in Electrical/Electronics Engineering, Anambra State University, Nigeria (GPA: 3.48/5.0) [2003-2008]
Publication	1. “Cyber-Security Audit for Smart Grid Networks: An Optimized Detection Technique Based on Bayesian Deep Learning” Sustainable Energy, Grids and Networks (SEGAN), under peer-review, submitted in July 2021. 2. “IoT Based Smart Home Energy Management System – a centrepiece of optimized real-time control” under peer-review in International Journal of Smart Sensing and Intelligent Systems (IJSSIS), 2021. 3. “A Smart On-device Based Power Consumption Forecast Model with an Optimized Weighted Average Ensemble Method” under peer-review in IAES International Journal of Artificial Intelligence, 2021. 4. "A Systematic Technique for Attenuation and Dispersion in Fibre Optics Communications using an Erbium Doped Amplifier" A proceeding of world Congress on engineering and computer science 2015, vol. II, WCESC 2015, October 21-23, 2015, San Francisco USA.

5. "Evaluation and Optimization of Quality of Service (QoS) of Mobile Cellular Networks" International Journal of Information and Communication Technology Research (IJICTR), VOL.3 N0.9, Oct.2013
6. "An Enhanced Technique in ATM Risk Reduction using Automated Biometrics Fingerprint in Nigeria" International Journal of Scientific Engineering Technology (IJSET), VOL.2 N0.11, Nov. 2013.
7. Design and Construction of Wireless based Burglar Alarm Detection System (unpublished thesis).

Awards

1. Atlas Corps Blended Fellowship Young Leadership Award, 2020.
2. Naresuan University Competitive Grant for Doctoral studies in specialized college, 2018.
3. Local Council Chairmanship Award for Best Student in Junior High School Certificate Examination, 1999.
4. Second Best Postgraduate Student in Electronics & Computer Engineering, 2014 Masters' graduates.