Deep Learning Techniques for Forecasting Electrical Loads

by

Tolulope Oluwaseun Olugbenga

BSc in Computer Science Engineering, University of Debrecen, 2018

A Thesis Submitted in Partial Fulfillment   
of the Requirements for the Degree of

Master of Science in Engineering

in the Graduate Academic Unit of Electrical and Computer Engineering

Supervisors: Dr. Dawn MacIsaac, Ph.D., Electrical and Computer Engineering

Dr. Julian Cardenas, Ph.D., Electrical and Computer Engineering

Examining Board: (name, degree, department/field), Chair

(name, degree, department/field)

(continue as required)

This thesis is accepted by the

Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

December 2021

© Tolulope Oluwaseun Olugbenga, 2021

ABSTRACT

For over a century, load forecasting has been used to plan and operate electric grids. Load aggregators, power marketers, independent system operators, regulatory commissions, industrial/commercial firms all benefit from load forecasting for revenue projection, energy trading, rate design, and other purposes. Load forecasting has been extensively studied in the literature [1]–[9]. Forecasting load has been done using statistical and machine learning (ML) techniques, and the line between the two is becoming increasingly blurred [1]. However, machine learning algorithms are more intelligent and can adapt to complex relationships between load and other influencing factors (weather) [10]. Deep learning approaches like recurrent neural networks (RNN) [11], long-short-term memory networks (LSTM) [12], and 1-D convolution neural networks (CNN) [3], [7] have recently caught the attention of researchers in this field. This study compares deep learning forecasters to some conventional forecasters used by utilities to see if deep learning better suits their needs.

DEDICATION

This thesis is dedicated to my future self; I want him to look back and understand that all his struggle, anguish, and late nights were not in vain. I adore you, and I am excited to meet the man you are going to become.

ACKNOWLEDGEMENTS

Without a doubt, this is one of the most challenging journeys I have ever completed. It has shown me how much I can grow and do when I believe in myself and put in the work. It was indeed a learning experience, and I am grateful that I did not give up but instead persisted in my efforts to cross the finish line. When I first arrived at UNB and saw what my colleagues in the lab were working on, I honestly wondered what I had gotten myself into. In addition to what I had heard from other students about taking a thesis-based master’s degree. I began to doubt my ability and became concerned that I might be unable to complete it.

I want to express my gratitude to my supervisors, Dr. Dawn MacIsaac and Dr. Julian Cardenas; I would not have completed this program without them. I appreciate your patience and words of encouragement, which reminded me that everything is possible and that all I need to do is keep going. I would also like to express my gratitude to my family for standing by my side and constantly motivating me to complete this program.

If I am entirely truthful, the individual who entered the lab for the first time in December 2018 will be unable to complete this degree. To complete, I needed to improve on both a personal and intellectual level. I want to applaud myself for not throwing in the towel and giving up; I want to commend myself for persevering through difficult times and even when the going became tougher. In short, this has been an educational experience and a game of physical and cognitive development. If I have to go through it all over again, I will because I would not be the man I am today without it. As a result, I would like to convey my appreciation to my supervisors and the University of New Brunswick for providing me with this opportunity to learn and develop as a person.

Table of Contents

[ABSTRACT ii](#_Toc82186650)

[DEDICATION iii](#_Toc82186651)

[ACKNOWLEDGEMENTS iv](#_Toc82186652)

[Table of Contents v](#_Toc82186653)

[List of Tables vii](#_Toc82186654)

[List of Figures viii](#_Toc82186655)

[List of Symbols, Nomenclature or Abbreviations ix](#_Toc82186656)

[1 Introduction 1](#_Toc82186657)

[1.1 Load Forecasting Overview 1](#_Toc82186658)

[1.2 Contributions 6](#_Toc82186659)

[1.3 Outline of the Thesis 6](#_Toc82186660)

[2 Investigation 7](#_Toc82186661)

[3 The Benchmark Algorithms 9](#_Toc82186662)

[3.1 The Seasonal Naïve Forecaster (SNF) 9](#_Toc82186663)

[3.2 The Multiple Linear Regression Forecaster (MLR) 10](#_Toc82186664)

[3.2.1 Assumptions of the MLR forecaster 11](#_Toc82186665)

[3.2.2 The Inputs and Target variables 11](#_Toc82186666)

[3.3 The Auto-Regressive Integrated Moving Average Forecaster (ARIMA) 12](#_Toc82186667)

[3.4 Artificial Neural Networks (ANNs) 14](#_Toc82186668)

[3.4.1 Artificial Neural Network Short Term Load Forecaster – Generation Three (ANNSTLF-G3) 19](#_Toc82186669)

[4 Deep Learning Techniques 22](#_Toc82186670)

[4.1 The Long Short Term Memory Forecaster (LSTM) 23](#_Toc82186671)

[4.2 The Convolutional Neural Network Forecaster (CNN) 26](#_Toc82186672)

[5 Results and Discussion 29](#_Toc82186673)

[5.1 Performance Metrics 29](#_Toc82186674)

[6 Conclusion 32](#_Toc82186675)

[Bibliography 33](#_Toc82186676)

[Appendix Title 46](#_Toc82186677)

[Glossary 47](#_Toc82186678)

Curriculum Vitae

List of Tables

[Table 1 29](#_Toc82186648)

List of Figures

[Figure 1 - An artificial neuron’s workflow 15](#_Toc82186641)

[Figure 2 - Examples of the most frequently used ANN activation functions [69] 16](#_Toc82186642)

[Figure 3 - The Structure of a simple feed-forward ANN [71] 16](#_Toc82186643)

[Figure 4 - The Block Diagram of the third generation ANNSTLF [80] 19](#_Toc82186644)

[Figure 5 – The structure of the BLF and CLF network 21](#_Toc82186645)

[Figure 6 - The Long Short-Term Memory Structure [90] 24](#_Toc82186646)

[Figure 7 - An Architecture of a one-dimensional CNN for time series data [103] 27](#_Toc82186647)

List of Symbols, Nomenclature or Abbreviations

Start writing here; remove page if there is no content

# 1 Introduction

## 1.1 Load Forecasting Overview

[ref]. Load forecasting is a critical component of electric utility design, planning, and operation; it has played a vital role in the power industry for over a century. For example, to have a stable electricity supply, reserve power must be prepared beforehand to serve consumers in the future (e.g., in case of high demand or failure in the current grid supply). However, load forecasting can also be helpful to organizations other than electric utilities, such as load aggregators, power marketers, independent system operators, regulatory commissions, and even industrial/commercial companies, banks, trading firms, and insurance companies [1], [2]. These organizations use load forecasting in power systems planning/operations, revenue projection, rate design, energy trading, and other activities [3]–[5].

Over the recent decade, there has been an increase in the grid’s adoption of renewables and distributed generation sources and progress and implementation of smart grids and buildings to meet expanding energy demands effectively. In order to integrate these distributed energy resources without creating system disruptions, reliable load forecasting across many time horizons is required [13]. Electric load forecasting is well studied [1], [7]–[9], and most current research focuses on developing more accurate forecasts. Load forecasting is particularly relevant in today’s context, with the advent of new smart grid technologies. The demand patterns used to drive these technologies are complex due to the deregulation of energy markets and the number of random variables, often governed by human behaviour, which needs to be considered to predict future electricity demand. Developing a forecasting model that is appropriate for a particular power network is not a simple task [4]–[6].

Different factors can affect load forecasts, such as the area’s location, the type of customers in the region, weather factors (e.g., temperature), the time of the day, day of the week, and other unpredictable factors (i.e., coronavirus outbreak). Electricity demand can be assessed periodically - hourly, daily, weekly, monthly, or yearly. Forecasting processes can be applied to various horizons: very short-term load forecasting (VSTLF, <1-day), short-term load forecasting (STLF, <2-weeks), medium-term load forecasting (MTLF <3-years), and long-term load forecasting (LTLF >3years) [10]. Short-term forecasting has been the focus in most current research, concentrating on horizons of less than two weeks [1], [10], [14].

According to Mandal et al. [15], STLF is a critical instrument in a utility system’s day-to-day operations and planning activities, such as energy transactions, unit commitment, economic dispatch, security analysis, fuel scheduling, and unit maintenance. STLF is a highly complex process influenced by various factors, including economic conditions, time of day, season, weather, and random influences. The electricity demand is determined by meteorological variables, human social activities, and industrial activities. Hippert et al. [16] explain that forecasting short-term load becomes complicated when the load at a given hour is dependent not only on the load at the previous hour but also on the load at the same hour on previous days and the load at the same hour on the same denomination day in the previous week [17]. Additionally, the forecaster must model the relationship between the load and other variables such as weather, leisure activities, and so on. However, due to a lack of knowledge about future events, longer forecasting horizons are more susceptible to unanticipated changes in future demand.

Both statistical techniques and machine learning (ML) have been applied to provide load forecasts, and with the advent of the widespread application of data science, the boundary between these two approaches is becoming more equivocal [1]. Examples of statistical techniques applied to electrical load forecasting include multiple linear regression analysis [18], [19] exponential smoothing [20], [21], and auto-regressive integrated moving average (ARIMA) modelling [22], [23]. On the other hand, ML algorithms are more intelligent and can be better, as they provide the capacity to learn and adapt to the non-linear and complex relationships between load and other influencing factors (e.g., weather, time of day) automatically [10]. Examples are Artificial Neural Networks (ANNs) [24][25], Fuzzy Regression Models [26], [27], Support Vector Machines [28], Gradient Boosting Machines [29]; they have all been applied to electrical load forecasting. The authors of [8] discussed many regression-based approaches for STLF [30]. Another study [31] examined various MLR algorithms for load forecasting. The disadvantage of MLR techniques is that they require external factors such as temperature and time of day. The ARIMA model is the most frequently used among all regression models since it consistently produces good prediction results; for example, the author in [32] combined ARIMA and Box-Jenkins methods to do hourly forecasting.

Deep learning approaches have had remarkable success in the last few years at handling complex sequential data [33], [34]. As a result, deep learning approaches have been effectively used to load forecasting applications, where they have been shown to outperform a variety of benchmark models, including simple ANNs and standard statistical time series methods such as ARIMA [35]. With improved computational power, more datasets, and the granularity of available data, deep learning models are expected to dominate the load forecasting field. Deep learning approaches like the recurrent neural network (RNN) [11], long-short-term-memory network (LSTM) [12], and the 1-D convolution neural network (CNN) [3], [7] have become enticing to researchers in this field, primarily because of their ability to learn about temporal dependencies in data inputs, and their ability to quickly adapt to abrupt changes in load patterns, as they occur.

The authors of [36] examined seven distinct models using three real-world data sets and demonstrated that deep learning methods could be employed in load forecasting applications in place of more traditional mathematical techniques such as ARIMA. The authors of [37] offered a novel parallel model that is a combination of convolutional neural networks (CNN) and recurrent neural networks (RNN). Because RNNs employ control theory in their structure, they can determine the relationship between old and new data, making them an appealing network for load forecasting applications in recent years. In terms of how RNNs work, the authors in [38] conducted an appropriate study on these networks. Similar to [37], the authors of [39] presented a mix of long short-term memory (LSTM) and convolutional neural networks (CNN). The proposed model’s performance in load forecasting was more stable than that of other machine learning techniques. Similarly, the authors of [25] suggested a new Deep-Energy model that combines a 1-D CNN for feature extraction with a fully connected network for forecasting future load data. They forecasted the data for the next three days using an hourly electricity consumption data set from the United States. To train, data from the preceding seven days was used. They compared the proposed model’s performance to that of five different machine learning techniques using the RMSE and MAPE metrics. The findings indicated that the Deep-Energy model is more capable of doing accurate short-term load predictions than other models.

In another paper [40], the authors presented a new model that incorporates three algorithms: Variational Mode Decomposition (VMD), Convolutional Neural Networks (CNN), and Gated Neural Networks (GRU), and named it SEPNet. This model was created to forecast hourly power prices, and to assess it, hourly data from the city of New York, USA was used. The data set included hourly electricity prices from 2015 to 2018. Compared to other models such as LSTM, CNN, and VMD-CNN, the SEPNet model fared better, improving the RMSE and MAPE by 25% and 19%, respectively. Additionally, several writers, for example [41], employed ANNs to forecast other load data types, such as photovoltaic system output data. They proposed a robust CNN-based model named PVPNet and assessed it using daily data from 2015.

However, Tao Hong spoke about the myth of finding the best technique [1]. He concluded that it is essential that researchers and users know that a universally best technique does not exist. The approach applied to load forecast should be based on forecasting needs and the dataset being analyzed. It is not likely that one approach will be helpful in all load forecasting scenarios. Different algorithms perform better or worse with different datasets. Furthermore, forecast errors differ significantly for different utilities, utility zones, different horizons, etc.

The purpose of this work is to compare deep learning forecasting against some conventional forecasters in use by specific utilities to determine if deep learning can better suit their needs. Overall accuracy and accuracy in peak detection were compared. Peak demand forecasts are critical for securing adequate generation, transmission, and distribution capacity. Accurate peak forecasts improve capital expenditure, decision making and system reliability.

## 1.2 Contributions

This research will assess the value added by deep learning algorithms (like CNN and LTSM) by comparing their performance to traditional forecasters. We aim to explore deep learning approaches to see if they are more adaptive to changes in extraneous factors like annual increases in power demand or temperature shifts. The goal is to develop forecasters that can adapt to complex data relationships without explicit user intervention. This is an important goal for our Smart-Grid Project industry collaborators, and we will provide an analysis tuned explicitly to their data set. Furthermore, because we have also included analysis on data that is publicly available, this work will be reproducible, making it a valuable comparison point for future research within and beyond our smart-grid team.

## 1.3 Outline of the Thesis

// To be filled

# 2 Investigation

This work aims to determine whether deep learning approaches can improve forecasting accuracy for data sets by comparing the accuracy of deep learning forecasters to some of the current forecasters used by utilities. This work will focus on STLF horizons. Three data sets will be investigated. Two sets come from an Independent Electrical System Operator in Ontario and have been included because the data is publicly available, which helps with the reproducibility of this work. One set is from Ottawa [42], and the other is from Toronto [42], and they both consist of city-wide load aggregation measurements taken hourly, spanning ten years from 2010-2019.

The third set comes from Saint john Energy, a municipally-owned utility reseller. This data is included because the work proposed here supports efforts in a larger Smart Grid Technologies project underway at UNB, which partners with that utility reseller. The Saint John Energy data set is smaller than the others, spanning about 3.5 years, from 2018 to the present, but otherwise matches the hourly measurements of city-wide Saint John load aggregates. In some parts of this work, weather data (temperature) obtained from Environment Canada [43] will augment the time-series data. The training dataset for both the Toronto and Ottawa datasets is from the years “2010-2018,” while the testing dataset is from 2019. The training dataset for the Saint John dataset is from the years “2018 - 2020”, while the testing dataset is from January 2021 until present.

Four benchmark forecasters will be used for comparison: a Seasonal Naïve forecaster, a Multiple Linear Regression (MLR) forecaster, an Auto-Regressive Integrated Moving Average (ARIMA) forecaster, and a forecaster based on a shallow Artificial Neural Network (ANN). These benchmark algorithms have been available for many years and have been implemented and used by researchers and utilities [1], [4]–[6], [44]–[46]. We began by implementing each of the benchmark algorithms. Then, the CNN and the LSTM forecasters were implemented. Finally, the performance of the deep learning forecasters was assessed by comparing them against the performance of the benchmark algorithms, using the data sets available.

# 3 The Benchmark Algorithms

Many publications lack detailed information about their experimental set-ups, making conducting direct comparisons with reported results challenging. The benchmark algorithms proposed for this work have been selected because they are relevant and because they are sufficiently well documented to reproduce [1], [4]–[6], [44]–[46].

## 3.1 The Seasonal Naïve Forecaster (SNF)

The naïve forecaster is a simple forecaster based on a random walk model [47]; it has often been implemented as a ground-level benchmark for developing more sophisticated forecasters [44], [48]–[50]. It is used to demonstrate how much value is added by forecasters under comparison – when a naïve forecaster outperforms a more complex forecasting model, we know that the complex model offers little value. Bracale [49] et al. point out that; “The simplest method to anticipate the next value in a time series is to assume it will have the same values as the current value.” which forms the basis of the naive forecaster. The Seasonal Naïve Forecaster (SNF) improves the naïve forecaster by considering seasonal trends [51]. The SNF can be expressed by the simple mathematical relationship shown in (1):

where  is the time series, and is the seasonal period (for hourly data, m=24 if we take the hourly sample from the day before). The naive formula takes the last observed value as the future value, while the seasonal naive formula takes the value from the previous season. In this work, we used the previous week’s lag for the SNF forecaster. The SNF forecaster is excellent for making short-term forecasts of variables that are generally stable or vary consistently. However, it is highly ineffective at forecasting time series data that fluctuate significantly or are susceptible to irregular elements such as temperature [48].

## 3.2 The Multiple Linear Regression Forecaster (MLR)

Multiple linear regression (MLR) is one of the most commonly used statistical techniques for load forecasting [14], [19], [26], [31], [44], [52]–[56]. MLR forecasters model the relationships between a continuous dependent variable and one or more independent variables. An MLR with two independent variables can be expressed mathematically as:

In the case of load forecasting,  is the load, and  are independent variables such as temperature and time-of-day, s are coefficients estimated, and is an error term. The error term  is typically assumed to have a mean of zero and a constant variance [19]. MLR models are fitted such that the sum-of-squares of differences of actual and forecasted values are minimized.

MLRs’ accuracy is determined mainly by the relationships between the data and the independent variables included. Amral et al. state in [57] that multi-linear regression models for short-term load forecasting are relatively simple to develop and maintain. Moreover, MLRs primary shortcoming is its reliance on the accuracy of previously recorded load and temperature data, which considerably impacts the predicted output. Increasing the number of relevant independent variables generally improves predictive accuracy, but it eventually reaches a point where the improvement is negligible. However, while MLRs can simulate non-linear relationships, they cannot do so without explicit user specifications [58]. Additionally, MLRs are incapable of intelligently learning and adapting to data changes caused by newer factors.

### 3.2.1 Assumptions of the MLR forecaster

1. The dependent variable and each of the independent variables should have a linear relationship.
2. Correlations between any of the independent variables are low. Multicollinearity exists when various variables are correlated with each other and with the dependent variable. When independent variables exhibit multicollinearity, obtaining the variable that contributes to the variance in the dependent variable can be difficult.
3. The residuals have a constant variance. The magnitude of our forecast error does not change much while the independent variable’s value changes.
4. Observations are autonomous. The MLR model presupposes that all observations are independent of one another; in other words, the residuals values are also independent of one another.
5. The data is normally distributed.

### 3.2.2 The Inputs and Target variables

The MLR forecaster was developed using ten independent variables, also referred to as inputs, and one target variable, which is the actual demand at a given hour. The independent variables are Temperature, Hour of the day, Month of the year, day of the week – Sunday is the first day of the week, Weekend Indicator – one or zero, Maximum hourly demand from the previous day, Minimum hourly demand from the previous day, Average hourly demand from the previous day, Hourly lag from the previous day, Hourly lag from the previous week.

## 3.3 The Auto-Regressive Integrated Moving Average Forecaster (ARIMA)

In 1970, Box and Jenkins proposed the autoregressive integrated moving average (ARIMA) forecaster. Therefore it is also known as the Box-Jenkins model [59]. The ARIMA forecaster is arguably one of the most popular and commonly utilized statistical forecasting techniques for load forecasting [60]. The ARIMA model seeks to explain data by utilizing time-series data on previous values and making linear regression predictions. It allows regression techniques to be applied to non-stationary data. If the data contains a trend, it is said to be non-stationary.

As the name implies, this family of techniques consists of three main components: a) an “autoregression” portion that models the series’ relationship with its past values (lagged values); b) a “moving average” portion that model the forecast as a function of past forecast errors (lagged forecast errors); and c) an “integrated” portion that makes the series stationary. A lag feature is a fancy phrase for a variable that holds data from earlier time steps. Lags are essential in time series research because of a phenomenon known as autocorrelation. For example, if we want to anticipate the demand for today t, we can use the demand from yesterday t-1 as a feature.

The term “AR” in ARIMA stands for autoregression, suggesting that the model is dependent on the relationship between the present values of the data and their previous values. In other words, it indicates that the data has been regressed against its previous values. The letter “I” stands for integrated, indicating that the data is stationary. Stationary data is time-series data that has been stabilized by subtracting the observations from the prior values. The term “MA” refers to a moving average model, which indicates that the model’s forecast or outcome is linearly related to its historical values [60]. This implies that forecasting errors are linear functions of previous errors.

Each  AR, I, and MA part is included in the model as  parameters p, d, q, respectively. Specific integer values are assigned to the parameters to denote the ARIMA model type. The ARIMA model is denoted by ARIMA (p, d, q). The parameter p denotes the number of autoregressive terms or “lag observations”; it is also called the “lag order” because it influences the model’s output by giving lagged data points. The parameter d is the degree of differentiation; it specifies how many times the lagging indicators have been subtracted from the data to make it stationary. Differencing is required since linear regression models work better when applied to stationary signals [56], [61]. The parameter q denotes the model’s forecast error and is often referred to as the size of the moving average window. The result is an estimate based on a linear combination of weighted differentiated lagged values and lagged errors as delineated in (3) [62]–[64]:

Here  is estimated to account for the average change between consecutive observations, the lag operator is the nth differentiated lag value of the time series, is the nth lag error of the time series. Generally, the error terms are assumed to be independent, uniformly distributed variables taken from a normal distribution with a mean of zero. and respectively are the parameters of the autoregressive and moving average parts; they represent the nth coefficients of that lag term estimated by the model to minimize the error.

In [65], Fernandez et al. forecasted energy load for non-residential buildings using an ARIMA model, a polynomial model, a neural network model, and a support vector machine model. The study analyzed energy consumption data from Spain’s University of Deusto in Donostia-San Sebastian. The goal was to forecast six days in advance at hourly intervals. Compared to the other models, the ARIMA model had the highest accuracy among all the models. Additionally, the authors noted that the ARIMA model runs 200 times quicker than the Support Vector Machine model because of the lower number of parameters.

The ARIMA model is mainly dependent on the quality of historical data and data differencing. It is critical to verify that data collection was reliable and extensive for the model to produce accurate results and forecasts. While ARIMA models can be accurate and dependable under the right conditions and with sufficient data, one of the model’s primary disadvantages is that the parameters (p, d, q) must be manually set. These numbers may vary slightly among datasets and forecast horizons. Therefore, finding the best accurate fit can be a lengthy trial-and-error process. The hyperparameters used for the Toronto, Ottawa and Saint John datasets are: (24, 2, 25), (23, 2, 24), and (24, 2, 24), respectively.

## 3.4 Artificial Neural Networks (ANNs)

Human brains are uniquely capable of comprehending the context of real-world situations in ways that machines cannot. In the 1950s, neural networks were designed to overcome this issue. An artificial neural network is an effort to imitate the network of neurons that comprise the human brain to enable the computer to learn and make decisions in a similar way to humans [6], [66]–[68]. ANNs are built by programming conventional computers to act like interconnected brain cells. ANNs are modelled like the human brain in that it learns the relationship between inputs and outputs via experience.

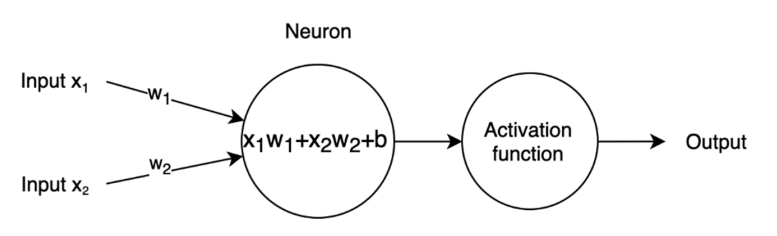


Figure 1 - An artificial neuron’s workflow

A neural network is made up of neurons. The primary neuronal workflow can be separated into the following components, as illustrated in Figure 1. A neuron gets two inputs x1 and x2, each of which has a unique weight, w1 and w2, reflecting its relative importance. Each neuron calculates the weighted total of those inputs and adds a bias b that is unique to it. Following that, the result is subjected to the activation function. Finally, the output of this activation function is the neuron’s final output.

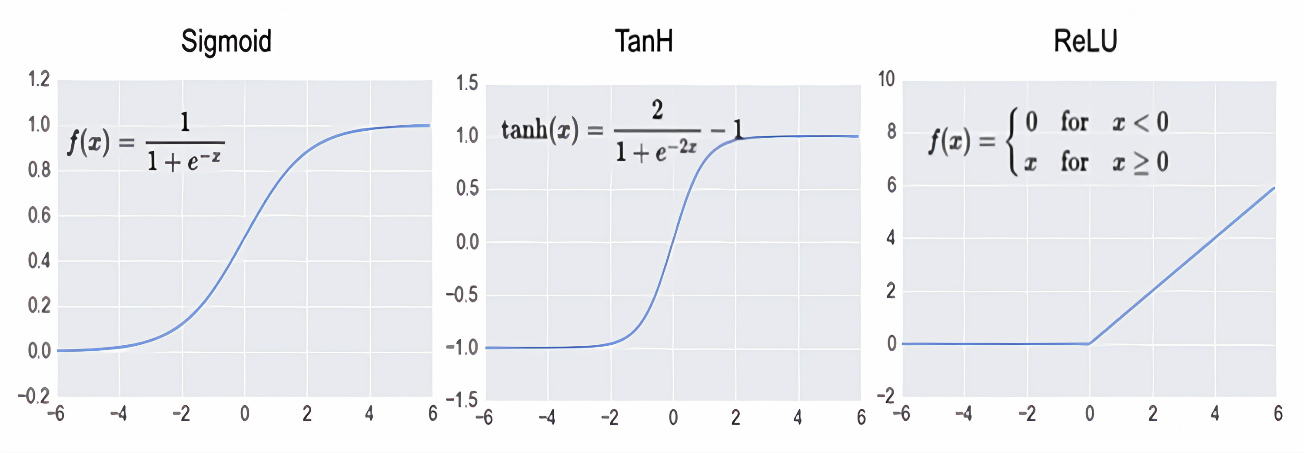


Figure 2 - Examples of the most frequently used ANN activation functions [69]

The activation functions of an ANN are critical because they enable the solution of non-linear problems. Figure 2 shows some frequently used activation functions. If the neuron’s workflow does not include activation functions, an ANN will perform similarly to a linear regression model [70]. Neurons in an ANN can be classified into three layers, as seen in Figure 3: input, hidden, and output.

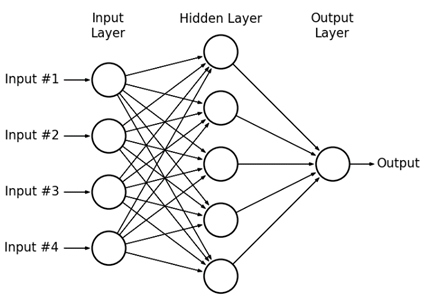


Figure 3 - The Structure of a simple feed-forward ANN [71]

The number of layers in an ANN is calculated as the sum of hidden and output layers. The input layer is nothing more than a vector representation of the input. Similarly, the number of neurons may be estimated by deleting the input layer and counting only the neurons in the hidden and output layers [2], [72]. An ANN’s operation can be characterized in terms of cycles, each of which consists of two phases: forward propagation and back-propagation. The forward propagation phase proceeds from left to right, intending to produce an output for specific input. When the final output of an ANN is produced, it is critical to calculating the error or how far off the target value is from the final output; this is the cost or loss function’s assignment. An MSE is one of the available metrics in the case of a regression problem. In this scenario, an optimization procedure such as gradient descent assists in minimizing this loss function by guiding it in the appropriate direction toward the function minimum.

Training an ANN entails fine-tuning its parameters: weights and biases. An ANN begins by randomly generating those parameters. After calculating the cost function, the network parameters are modified according to an optimization technique during the back-propagation phase [9]. Typically, training instances are handled in batches, which speeds up the training operation due to the time required to calculate the gradient descent and update the network’s parameters. The back-propagation phase proceeds from right to left.

The cross-validation technique is used to avoid overtraining. The training set is separated into two distinct halves. For example, if three years of data are provided, they are separated into two years and one-year sets. The first set is used to train the ANN, while the second set is used to validate the learned model after a few hundred runs over the training data. The validation set’s error is evaluated. Typically, as the number of runs over the training set increases, this error reduces until the ANN is over-trained, as shown by a rise in this error. As a result, when the error on the validation set begins to grow, the training is stopped. This approach generates the necessary number of epochs throughout the training set. After that, the whole three years of data is used to re-train the MLP with this amount of epochs.

The capacity for generalization is one of the critical properties of ANNs [18]; this means that the ANN should not remember the data on which it was trained but rather grasp, extract, and learn the patterns, trends, and dependencies to perform well with new, previously unseen data [6]. Neural networks have produced excellent results in load forecasting [1], [6], [9], [29], [54], [66]. They have gained popularity due to their capacity to discover complex and non-linear correlations from historical data, which is extremely difficult to do using statistical techniques. Adya and Collopy [73] draw two major findings from their evaluation: they demonstrated that neural networks have the capacity for prediction and that neural network research must be validated through comparisons to alternative methodologies. Zhang et al. [74] evaluated the use of neural networks in load forecasting and demonstrated that while neural networks could deal with huge amounts of historical load data with non-linear features, they neglected the linear relationship between the data. In [75] and [76], Papalexopoulos et al. developed a neural network-based approach in addition to a regression-based approach. Both models were validated using training data from 1986 to 1990 on peak and hourly loads for 1991. It was shown that the ANN model enhanced forecasting accuracy for both peak load and hourly forecasts.

### 3.4.1 Artificial Neural Network Short Term Load Forecaster – Generation Three (ANNSTLF-G3)

One of the most popular ML-based load forecasters is the ANNSTLF [1], [45], [56]. The ANNSTLF-G3 has improved prediction accuracy and generated economic benefits for over a dozen utilities [58], [77]. Some publications have named ANNSTLF-G3 the best forecaster for short-term load forecasting [1], [56]. The configuration of this load forecaster has undergone a few revisions since it was first proposed [78], [79], and we will discuss the third-generation design (G3) [80], which uses two shallow multi-layer feed-forward ANNs together with a recursive least squares (RLS) combiner to predict short-term load. The figure below shows the block diagram of the system:

Diagram, schematic

Description automatically generated

Figure 4 - The Block Diagram of the third generation ANNSTLF [80]

Both ANN blocks are multi-layer perceptron trained with the error back-propagation algorithm. The base-load forecaster (BLF) is trained to forecast regular next-day load, while the change-load forecaster (CLF) is trained to forecast changes in the load demand from one day to the next. The two ANN forecasters complement one another since the BLF places a stronger emphasis on normal load patterns, while the CLF places a stronger emphasis on yesterday’s load. Combining these two independent forecasts improves accuracy. This is especially true in instances of abrupt load changes brought about by weather changes. The BLF has a proclivity for a delayed response to sudden changes in load.

On the other hand, because the CLF uses yesterday’s load as a baseline and forecasts future changes in that load, it responds more quickly to changing conditions. It is argued that the CLF forecaster allows the model to rapidly adapt to abrupt changes in temperature [56], [81], [82]. Both blocks are presented with the same 79 inputs (see Figure 4) and output a 24x1 vector representing hourly forecasts. The CLF sums predicted changes with actual last-day values to produce its output. The final forecast is based on a weighted average of each block’s outputs, with the weights adaptively determined using an RLS algorithm. ANNSTLF-G3 forecasts are typically produced one day at a time (24 hrs.). To extend the forecast horizon beyond one day, the previous day’s forecast load is substituted for the actual load to produce the next day’s load forecast. According to the paper’s authors, the ANNSTLF-G3 algorithm performs best when the hidden layer contains between 30 and 60 neurons and is trained using two to three years of data.

A picture containing text, clock

Description automatically generated

Figure 5 – The structure of the BLF and CLF network

The resilient back-propagation algorithm is used to train the BLF and CLF networks. According to the MATLAB handbook, this training method is effective and is frequently used for pattern recognition problems [83]. Additionally, we observed that this training method outperformed the Levenberg-Marquardt back-propagation method. The hidden layer is comprised of 60 neurons.   In the hidden and output layers, the activation function is a hyperbolic tangent sigmoid transfer function. We observed improved results when we changed the activation function of the output layer from linear to tangent sigmoid. The training data was divided into two groups, 80% used for training and 20% for validation. The RLS combiner has an initial weight for each hour for both the BLF and CLF outputs; after each iteration, it automatically updates the weights for each hour based on the algorithm’s calculation.

# 4 Deep Learning Techniques

According to Yann Lecun and colleagues, “deep learning enables computer models built of many processing layers to learn representations of data at different levels of abstraction” [84]. Although the concept of ‘deep learning’ has been bandied about for decades, it was frequently dismissed as a fanciful notion rather than a feasible technology. This was primarily due to three constraints: (i) insufficient training data, (ii) insufficient processing power, and (iii) insufficient efficient training algorithms [85]. With improvements in the semiconductor industry resulting in powerful graphics processing units (GPUs) and the rising digitization of the world, these limits have been overcome. Additionally, Geoffrey Hinton’s quantum leap in inventing an effective neural network training algorithm paved the way for deep learning implementations. Deep learning models have grown in popularity during the last several years in fields such as computer vision, speech recognition, machine translation, and board game programs, where they have delivered results comparable to expert human performance, if not beyond it [86].

The significant benefit of deep learning models over traditional models is that they acquire high-level features incrementally from data, eliminating the Need for topic knowledge and time-consuming feature extraction [87]. The primary reason for utilizing deep learning models in this study is that they are superior to the conventional models in terms of their ability to (i) learn extremely non-linear relationships and (ii) learn shared uncertainties. In discussion, the terms deep learning and neural networks are frequently used interchangeably, which can be confusing. As a result, it is worth emphasizing that the term “deep” refers to the number of layers in a neural network. A neural network with more than three layers—including the inputs and outputs—is termed a deep learning technique. A neural network with only two or three layers is referred to as a simple neural network.

## 4.1 The Long Short Term Memory Forecaster (LSTM)

Recurrent Neural Network (RNNs) introduced memory into neural networks, which helps to model sequential data. RNNs have been successfully applied in machine translation, speech synthesis, and time series prediction [88]. Typically, back-propagation or real-time recurrent learning algorithms are used to train RNNs. These training methods expose traditional RNNs to vanishing gradient issues, reducing their effectiveness when dealing with large data sets [7], [67], [89]. The LSTM is an RNN created to fix vanishing gradient problems and store information for long periods. Its memory cell configuration helps retain information more than any deep neural network currently available [67].

LSTMs are a classification of recurrent neural networks that can learn the order of dependencies between elements in a sequence. LSTMs overcome the issue of vanishing gradients using gates that regulate the input flow, making them excellent for dealing with time series data with lengthy temporal dependencies. Unlike a conventional recurrent unit, which overwrites its memory at each time step, the LSTM unit can select whether to retain existing memory via the introduced gates. The LSTM provides a model capable of storing information for an extended period and better control of gradients. Munem[67] et al. argue that LSTM is better than other deep neural networks because of its memory cell configuration.

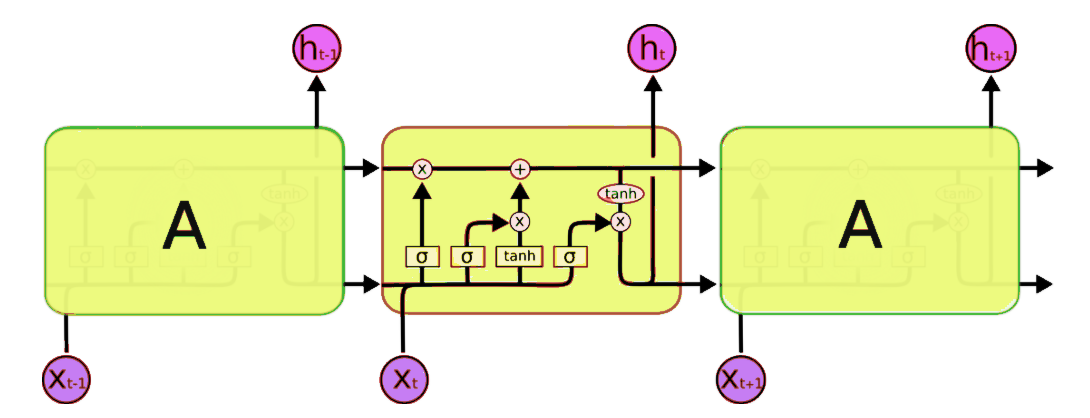


Figure 6 - The Long Short-Term Memory Structure [90]

This repeating module in standard RNNs will have a relatively simple structure, such as a single tanh layer. LSTMs have the structure as well, although the repeating module is structured differently. Rather than a single neural network layer, there are four that interact in a unique way. Each memory block is composed of the following: a memory cell, an input gate, a forget gate, and an output gate. Each line in Figure 3 represents a whole vector, from one node’s output to the inputs of others. The pink circles denote operations performed at the point level, such as vector addition. The tiny yellow boxes represent layers of learned neural networks. Concatenation occurs when two lines merge, whereas forking occurs when a line’s content is replicated, and the copies are sent to various locations. The key to LSTMs is the cell state, represented by the horizontal line running across the diagram’s top. It maintains the integrity of data travelling through it. By adequately regulating gates, the LSTMs are capable of removing or adding information to the cell state. Gates typically allow information to pass through on an optional basis. They are constructed using a sigmoid neural network layer and pointwise multiplication. The sigmoid layer generates values between 0 and 1, indicating how much of each element should be allowed to pass through. A number of zero indicates that “everything is forgotten,” whereas a value of one indicates that “everything is retained.” Three gates protect and govern the cell state in an LSTM [7].

The initial stage in LSTM is for a sigmoid layer dubbed the “forget gate layer” to decide what information should be discarded from the cell state. It examines the preceding hidden layer and input and returns a number between 0 and 1 for each number in the cell state. The following step is to decide what new information will be stored in the cell state by merging two pieces to make a state update. The first is that a sigmoid layer known as the “input gate layer” determines which values need to be updated. The second is that a tanh layer generates a vector of new candidate values that could be inserted into the state. Following that, multiplying the old state by forgetting the items and adding the new candidate’s values to update the old cell state into the new cell state. Finally, the net executes the output, a filtered version of our cell state [91]. First, a sigmoid layer uses the cell state to execute outputs. Then we run the cell state through tanh and multiply it by the output of the sigmoid gate to output only the sections we want. There will be no improvement in the state cell memory if the input gate value is minimal and close to zero. In a network model, stacked LSTM can be implemented by using multiple LSTM layers [7]. The technique of forgetting and retaining information within a cell makes LSTM perfect for dealing with sequential data.

Other researchers on the smart-grid team at UNB have used the LSTM algorithm for load forecasting, but only with the Saint John dataset. We will take the present implementation and alter it to meet our datasets and input feature sets. Additionally, because the ANNSTLF structure was recognized as the best forecaster for short-term load forecasting [1], [56]; our approach mimics the ANNSTLF structure by creating a Base Load Forecaster, Change in the Load Forecaster, and RLS combiner; while using the LSTM algorithm in place of the ANN. The architecture will have the same inputs and structure as the ANNSTLF, but the BLF and CLF algorithms will be trained using LSTMs. It will be interesting to see if this adjustment can improve forecasting performance.

## 4.2 The Convolutional Neural Network Forecaster (CNN)

In recent years, Convolutional Neural Networks (CNNs) have gained the attention of researchers studying load forecasting [3], [35], [37], [92], [93]. CNNs are a type of deep learning network used for data processing with a grid-like topology [3], [88], [94]. This can comprise time series and image data, which can be viewed as a one-dimensional and two-dimensional data grid, respectively [3], [94]–[96]. CNN is like the ANN in that it is a feed-forward neural network designed to mimic human neurons [3], [84]. They have been successfully applied in computer vision, audio processing, activity recognition, natural language processing, drug discovery, video recognition, and time series forecasting, among other applications [7], [97]–[102]. In load forecasting, CNNs are known to boost the power of the ANN because they have deeper layers and have model parameters such as a receptive field length and dilation, which can help interpret load data better [7], [36]. In at least one of its layers, CNN employs a particular linear mathematical technique called convolution [88]. Convolution is performed in CNNs by repeatedly applying filters or kernels to the input data to build a feature map.

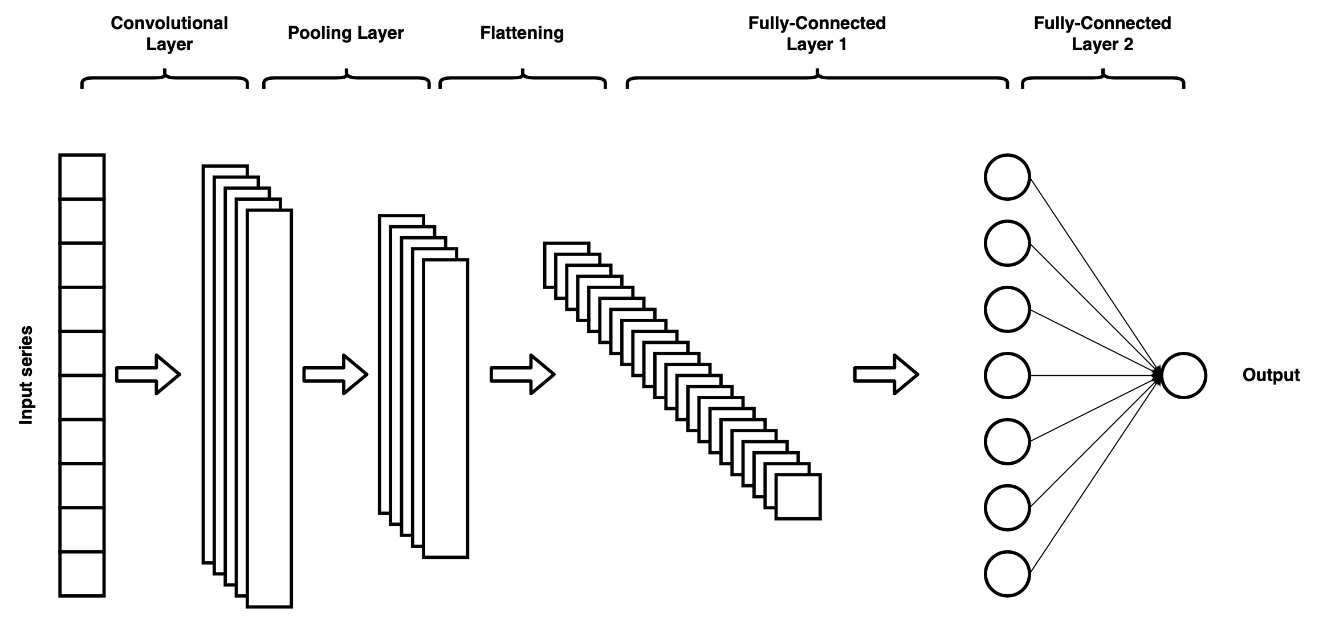


Figure 7 - An Architecture of a one-dimensional CNN for time series data [103]

The convolutional layer performs three distinct actions. The feature map is created because of the first procedure mentioned above. The second stage involves activating the elements in the feature map using a non-linear activation function, most commonly a RELU or rectified linear activation function [88]. The third stage employs a pooling procedure to smooth and minimize the dimensions of the resulting feature map. The max-pooling method is commonly used; it returns an array of the maximum output values within the previous layer’s rectangle neighborhood [88]. One or more convolutional layers may be present in the CNN network. After the convolutional layers generate their outputs, the hidden or fully connected layers receive them. The output layer is positioned immediately after the hidden layer and serves the same purpose as an output layer in a typical neural network. Amaradinghe[3] et al. compared the CNN with the LSTM, SVM, ANN, and other algorithms for individual building level load forecasting. They concluded that CNN is a viable technique that produces accurate load forecasts.

We implemented the CNN algorithm similarly to the LSTM using the ANNSTLF structure. The architecture of the CNNs used in this study consists of six layers: an input layer, a convolutional layer, a rectified linear unit activation layer (relu), a max-pooling layer, a fully connected layer, and a regression output layer. The adam optimization training algorithm was used to train the CNNs.

# 5 Results and Discussion

## 5.1 Performance Metrics

This study will compare all forecasters’ performance across all forecasters and subsets of the forecasts such as weekdays, weekends, mornings, or evenings. It will assist us in identifying instances where forecasters perform better or worse than expected. The performance will be evaluated according to accuracy in forecast values and accuracy in peak load localization. Table 1 delineates the main error measures used to quantify accuracy:

|  |  |
| --- | --- |
|  |  |
|  |  |

Table 1

Mean Absolute Error (MAE) is the simplest way to measure forecast error [104], but because it is an absolute measure, it does not provide a way to compare measurements across forecast scenarios of different scales. For this reason, Mean Absolute Percent Error (MAPE) is commonly used [1] since the interpretation of comparisons is straightforward. The MAPE value indicates the magnitude of the forecasted values’ error in percentage terms; it is also the most frequently used load forecasting metric [1]. However, as is the case with demand forecasting, MAPE returns undefined values when the actuals are zero. It produces extreme values when the actuals are close to zero and penalizes negative errors (when forecasts exceed actuals) more severely than positive errors. This is because the percentage error for too low forecasts cannot surpass 100%, while there is no maximum limit to overly high forecasts [1], [105]. Both the MAE and MAPE risk underestimating the effect of significant but infrequent errors. By focusing exclusively on the mean, we run the danger of being blindsided by a colossal error. To accommodate for severe, unusual errors, we incorporated the Root Mean Square Error (RMSE) [25]. With the RMSE, when we square the errors before computing the mean and then take the square root, we get an error size measure favouring significant but rare errors above the mean. However, RMSE is not scaled to the original error, so it is more difficult to interpret.

To fully capture bias and precision, Mean Biased Error (MBE) and standard deviation (SD) can also be used [106], [107]. MBE denotes the mean error of all forecasts across the entire forecast horizon; it quantifies the model’s overall bias and determines if the model produces over-or under-estimation (MBE > 0 or MBE < 0). Since a positive error on one pair can compensate for a negative error on another, a forecast model can attain a very low bias while remaining imprecise. However, evaluating our forecast precision solely based on the MBE value will be insufficient; but a significantly biased forecast already indicates that something is amiss with the model.

Finally, standard deviation indicates the spread of errors by quantifying how far apart individual errors are from the mean error.  The standard deviation is calculated using the total number of values in the time series; it is one of the most accurate measures of dispersion. The standard deviation is less impacted than other measurements by irregularities in the time series.  In comparison to other measures of dispersion, the standard deviation is more difficult to compute and interpret. The standard deviation is also strongly influenced by extreme values in the time series. However, while these metrics have their limits, they are simple instruments for assessing forecast accuracy.

# 6 Conclusion

Bibliography

[1] T. Hong and S. Fan, “Probabilistic electric load forecasting: A tutorial review,” *Int. J. Forecast.*, vol. 32, no. 3, pp. 914–938, 2016, doi: 10.1016/j.ijforecast.2015.11.011.

[2] S. Saurabh, H. Shoeb, A. B. Mohammad, S. Singh, S. Hussain, and M. A. Bazaz, “Short term load forecasting using artificial neural network,” in *2017 4th International Conference on Image Information Processing, ICIIP 2017*, 2018, pp. 159–163, doi: 10.1109/ICIIP.2017.8313703.

[3] K. Amarasinghe, D. L. Marino, and M. Manic, “Deep neural networks for energy load forecasting,” 2017, doi: 10.1109/ISIE.2017.8001465.

[4] J. Zhang, Y. M. Wei, D. Li, Z. Tan, and J. Zhou, “Short term electricity load forecasting using a hybrid model,” *Energy*, 2018, doi: 10.1016/j.energy.2018.06.012.

[5] C. Kuster, Y. Rezgui, and M. Mourshed, “Electrical load forecasting models: A critical systematic review,” *Sustainable Cities and Society*. 2017, doi: 10.1016/j.scs.2017.08.009.

[6] I. K. Nti, M. Teimeh, O. Nyarko-Boateng, and A. F. Adekoya, “Electricity load forecasting: a systematic review,” *J. Electr. Syst. Inf. Technol.*, 2020, doi: 10.1186/s43067-020-00021-8.

[7] A. Almalaq and G. Edwards, “A review of deep learning methods applied on load forecasting,” *Proc. - 16th IEEE Int. Conf. Mach. Learn. Appl. ICMLA 2017*, vol. 2017-Decem, pp. 511–516, 2017, doi: 10.1109/ICMLA.2017.0-110.

[8] B. Yildiz, J. I. Bilbao, and A. B. Sproul, “A review and analysis of regression and machine learning models on commercial building electricity load forecasting,” *Renewable and Sustainable Energy Reviews*. 2017, doi: 10.1016/j.rser.2017.02.023.

[9] A. Baliyan, K. Gaurav, and S. Kumar Mishra, “A review of short term load forecasting using artificial neural network models,” 2015, doi: 10.1016/j.procs.2015.04.160.

[10] Z. Deng, B. Wang, Y. Xu, T. Xu, C. Liu, and Z. Zhu, “Multi-scale convolutional neural network with time-cognition for multi-step short-Term load forecasting,” *IEEE Access*, vol. 7, pp. 88058–88071, 2019, doi: 10.1109/ACCESS.2019.2926137.

[11] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, “Short-Term Residential Load Forecasting Based on LSTM Recurrent Neural Network,” *IEEE Trans. Smart Grid*, vol. 10, no. 1, pp. 841–851, 2019, doi: 10.1109/TSG.2017.2753802.

[12] J. Zheng, C. Xu, Z. Zhang, and X. Li, “Electric load forecasting in smart grids using Long-Short-Term-Memory based Recurrent Neural Network,” 2017, doi: 10.1109/CISS.2017.7926112.

[13] A. Rahman, V. Srikumar, and A. D. Smith, “Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks,” *Appl. Energy*, 2018, doi: 10.1016/j.apenergy.2017.12.051.

[14] T. Hong, J. Wilson, and J. Xie, “Long term probabilistic load forecasting and normalization with hourly information,” *IEEE Trans. Smart Grid*, vol. 5, no. 1, pp. 456–462, 2014, doi: 10.1109/TSG.2013.2274373.

[15] P. Mandal, T. Senjyu, N. Urasaki, and T. Funabashi, “A neural network based several-hour-ahead electric load forecasting using similar days approach,” *Int. J. Electr. Power Energy Syst.*, 2006, doi: 10.1016/j.ijepes.2005.12.007.

[16] H. S. Hippert, C. E. Pedreira, and R. C. Souza, “Neural networks for short-term load forecasting: A review and evaluation,” *IEEE Trans. Power Syst.*, 2001, doi: 10.1109/59.910780.

[17] Ö. Ö. Bozkurt, G. Biricik, and Z. C. Taysi, “Artificial neural network and SARIMA based models for power load forecasting in Turkish electricity market Ö,” *PLoS One*, 2017, doi: 10.1371/journal.pone.0175915.

[18] S. Kumar, S. Mishra, and S. Gupta, “Short term load forecasting using ANN and multiple linear regression,” 2016, doi: 10.1109/CICT.2016.44.

[19] A. Y. Saber and A. K. M. R. Alam, “Short term load forecasting using multiple linear regression for big data,” *2017 IEEE Symp. Ser. Comput. Intell. SSCI 2017 - Proc.*, vol. 2018-Janua, pp. 1–6, 2018, doi: 10.1109/SSCI.2017.8285261.

[20] P. Ji, D. Xiong, P. Wang, and J. Chen, “A study on exponential smoothing model for load forecasting,” 2012, doi: 10.1109/APPEEC.2012.6307555.

[21] J. F. Rendon-Sanchez and L. M. de Menezes, “Structural combination of seasonal exponential smoothing forecasts applied to load forecasting,” *Eur. J. Oper. Res.*, 2019, doi: 10.1016/j.ejor.2018.12.013.

[22] L. Tang, Y. Yi, and Y. Peng, “An ensemble deep learning model for short-term load forecasting based on ARIMA and LSTM,” 2019, doi: 10.1109/SmartGridComm.2019.8909756.

[23] B. Nepal, M. Yamaha, A. Yokoe, and T. Yamaji, “Electricity load forecasting using clustering and ARIMA model for energy management in buildings,” *Japan Archit. Rev.*, 2020, doi: 10.1002/2475-8876.12135.

[24] A. Badri, Z. Ameli, and A. Motie Birjandi, “Application of artificial neural networks and fuzzy logic methods for short term load forecasting,” 2012, doi: 10.1016/j.egypro.2011.12.965.

[25] P. H. Kuo and C. J. Huang, “A high precision artificial neural networks model for short-Term energy load forecasting,” *Energies*, 2018, doi: 10.3390/en11010213.

[26] T. Hong and P. Wang, “Fuzzy interaction regression for short term load forecasting,” *Fuzzy Optim. Decis. Mak.*, 2014, doi: 10.1007/s10700-013-9166-9.

[27] M. Hanmandlu and B. K. Chauhan, “Load forecasting using hybrid models,” *IEEE Trans. Power Syst.*, 2011, doi: 10.1109/TPWRS.2010.2048585.

[28] A. Yang, W. Li, and X. Yang, “Short-term electricity load forecasting based on feature selection and Least Squares Support Vector Machines,” *Knowledge-Based Syst.*, 2019, doi: 10.1016/j.knosys.2018.08.027.

[29] V. Mayrink and H. S. Hippert, “A hybrid method using Exponential Smoothing and Gradient Boosting for electrical short-term load forecasting,” 2017, doi: 10.1109/LA-CCI.2016.7885697.

[30] S. Humeau, T. K. Wijaya, M. Vasirani, and K. Aberer, “Electricity load forecasting for residential customers: Exploiting aggregation and correlation between households,” 2013, doi: 10.1109/SustainIT.2013.6685208.

[31] G. Dudek, “Pattern-based local linear regression models for short-term load forecasting,” *Electr. Power Syst. Res.*, 2016, doi: 10.1016/j.epsr.2015.09.001.

[32] N. Amjady, “Short-term hourly load forecasting using time-series modeling with peak load estimation capability,” *IEEE Trans. Power Syst.*, vol. 16, no. 4, pp. 798–805, 2001, doi: 10.1109/59.962429.

[33] M. Baccouche, F. Mamalet, and C. Wolf, “（RGB)Sequential deep learning for human action recognition,” *Int. Work. Hum. Behav. Underst.*, 2011.

[34] D. Yu, L. Deng, I. Jang, P. Kudumakis, M. Sandler, and K. Kang, “Deep learning and its applications to signal and information processing,” *IEEE Signal Process. Mag.*, 2011, doi: 10.1109/MSP.2010.939038.

[35] M. Vos, C. Bender-Saebelkampf, and S. Albayrak, “Residential Short-Term Load Forecasting Using Convolutional Neural Networks,” 2018, doi: 10.1109/SmartGridComm.2018.8587494.

[36] A. Gasparin, S. Lukovic, and C. Alippi, “Deep Learning for Time Series Forecasting: The Electric Load Case,” 2019, [Online]. Available: http://arxiv.org/abs/1907.09207.

[37] W. He, “Load Forecasting via Deep Neural Networks,” 2017, doi: 10.1016/j.procs.2017.11.374.

[38] C. Gallicchio, A. Micheli, and L. Pedrelli, “Design of deep echo state networks,” *Neural Networks*, 2018, doi: 10.1016/j.neunet.2018.08.002.

[39] C. Tian, J. Ma, C. Zhang, and P. Zhan, “A deep neural network model for short-term load forecast based on long short-term memory network and convolutional neural network,” *Energies*, 2018, doi: 10.3390/en11123493.

[40] C. J. Huang, Y. Shen, Y. H. Chen, and H. C. Chen, “A novel hybrid deep neural network model for short-term electricity price forecasting,” *Int. J. Energy Res.*, 2021, doi: 10.1002/er.5945.

[41] C. J. Huang and P. H. Kuo, “Multiple-Input Deep Convolutional Neural Network Model for Short-Term Photovoltaic Power Forecasting,” *IEEE Access*, 2019, doi: 10.1109/ACCESS.2019.2921238.

[42] “IESO - Hourly Zonal Demand Report.” http://reports.ieso.ca/public/DemandZonal/ (accessed Jun. 05, 2021).

[43] “Historical Climate Data - Climate - Environment and Climate Change Canada.” https://climate.weather.gc.ca/ (accessed Jan. 05, 2021).

[44] T. Hong, P. Wang, and H. L. Willis, “A naïve multiple linear regression benchmark for short term load forecasting,” 2011, doi: 10.1109/PES.2011.6038881.

[45] K. Methaprayoon, W. J. Lee, S. Rasmiddatta, J. R. Liao, and R. J. Ross, “Multistage artificial neural network short-term load forecasting engine with front-end weather forecast,” *IEEE Trans. Ind. Appl.*, 2007, doi: 10.1109/TIA.2007.908190.

[46] A. K. Singh, Ibraheem, S. Khatoon, M. Muazzam, and D. K. Chaturvedi, “Load forecasting techniques and methodologies: A review,” 2012, doi: 10.1109/ICPCES.2012.6508132.

[47] G. Papacharalampous, H. Tyralis, and D. Koutsoyiannis, “Predictability of monthly temperature and precipitation using automatic time series forecasting methods,” *Acta Geophys.*, 2018, doi: 10.1007/s11600-018-0120-7.

[48] P. Wang, B. Liu, and T. Hong, “Electric load forecasting with recency effect: A big data approach,” *Int. J. Forecast.*, 2016, doi: 10.1016/j.ijforecast.2015.09.006.

[49] A. Bracale, G. Carpinelli, P. De Falco, and T. Hong, “Short-term industrial load forecasting: A case study in an Italian factory,” 2017, doi: 10.1109/ISGTEurope.2017.8260176.

[50] M. Rana and I. Koprinska, “Forecasting electricity load with advanced wavelet neural networks,” *Neurocomputing*, 2016, doi: 10.1016/j.neucom.2015.12.004.

[51] Da Liu, K. Sun, H. Huang, and P. Tang, “Monthly load forecasting based on economic data by decomposition integration theory,” *Sustain.*, 2018, doi: 10.3390/su10093282.

[52] T. Hong, M. Gui, M. E. Baran, and H. L. Willis, “Modeling and forecasting hourly electric load by multiple linear regression with interactions,” *IEEE PES Gen. Meet. PES 2010*, pp. 1–8, 2010, doi: 10.1109/PES.2010.5589959.

[53] M. Abuella and B. Chowdhury, “Solar power probabilistic forecasting by using multiple linear regression analysis,” 2015, doi: 10.1109/SECON.2015.7132869.

[54] K. Panklib, C. Prakasvudhisarn, and D. Khummongkol, “Electricity Consumption Forecasting in Thailand Using an Artificial Neural Network and Multiple Linear Regression,” *Energy Sources, Part B Econ. Plan. Policy*, 2015, doi: 10.1080/15567249.2011.559520.

[55] X. Sun, Z. Ouyang, and D. Yue, “Short-term load forecasting based on multivariate linear regression,” 2017, doi: 10.1109/EI2.2017.8245401.

[56] R. Weron, *Modeling and forecasting electricity loads and prices: A statistical approach*. wiley, 2006.

[57] N. Amral, C. S. Özveren, and D. King, “Short term load forecasting using multiple linear regression,” 2007, doi: 10.1109/UPEC.2007.4469121.

[58] T. Hong, “Short Term Electric Load Forecasting,” North Carolina State University, 2010.

[59] E. Stellwagen and L. Tashman, “ARIMA : The Models of Box and Jenkins,” *Foresight Int. J. Appl. Forecast.*, 2013.

[60] K. Goswami, A. Ganguly, and A. K. Sil, “Day ahead forecasting and peak load management using multivariate auto regression technique,” *Proc. 2018 IEEE Appl. Signal Process. Conf. ASPCON 2018*, no. 1, pp. 279–282, 2018, doi: 10.1109/ASPCON.2018.8748661.

[61] R. Bonetto and M. Rossi, “Parallel multi-step ahead power demand forecasting through NAR neural networks,” *2016 IEEE Int. Conf. Smart Grid Commun. SmartGridComm 2016*, pp. 314–319, Dec. 2016, doi: 10.1109/SmartGridComm.2016.7778780.

[62] G. N. Shilpa and G. S. Sheshadri, “ARIMAX Model for Short-Term Electrical Load Forecasting,” *Int. J. Recent Technol. Eng.*, 2019, doi: 10.35940/ijrte.d7950.118419.

[63] H. Cui and X. Peng, “Short-Term City Electric Load Forecasting with Considering Temperature Effects: An Improved ARIMAX Model,” *Math. Probl. Eng.*, 2015, doi: 10.1155/2015/589374.

[64] A. Shadkam, “USING SARIMAX TO FORECAST ELECTRICITY DEMAND AND CONSUMPTION,” 2020.

[65] I. Fernández, C. E. Borges, and Y. K. Penya, “Efficient building load forecasting,” 2011, doi: 10.1109/ETFA.2011.6059103.

[66] X. H. Le, H. V. Ho, G. Lee, and S. Jung, “Application of Long Short-Term Memory (LSTM) neural network for flood forecasting,” *Water (Switzerland)*, 2019, doi: 10.3390/w11071387.

[67] M. Munem, T. M. Rubaith Bashar, M. H. Roni, M. Shahriar, T. B. Shawkat, and H. Rahaman, “Electric power load forecasting based on multivariate LSTM neural network using bayesian optimization,” *2020 IEEE Electr. Power Energy Conf. EPEC 2020*, vol. 3, 2020, doi: 10.1109/EPEC48502.2020.9320123.

[68] V. Dehalwar, A. Kalam, M. L. Kolhe, and A. Zayegh, “Electricity load forecasting for urban area using weather forecast information,” *2016 IEEE Int. Conf. Power Renew. Energy, ICPRE 2016*, pp. 355–359, 2017, doi: 10.1109/ICPRE.2016.7871231.

[69] “Attention mechanism + relu activation function: adaptive parameterized relu activation function | Develop Paper.” https://developpaper.com/attention-mechanism-relu-activation-function-adaptive-parameterized-relu-activation-function/ (accessed Sep. 09, 2021).

[70] A. Si. Walia, “Activation functions and it’s types-Which is better?,” *Towards Data Science*, 2017. .

[71] “Artificial Neural Network (ANN) with Practical Implementation | by Amir Ali | Wavy AI Research Foundation | Medium.” https://medium.com/machine-learning-researcher/artificial-neural-network-ann-4481fa33d85a (accessed Sep. 10, 2021).

[72] C. L. COCIANU and H. GRIGORYAN, “An Artificial Neural Network for Data Forecasting Purposes,” *Inform. Econ.*, 2015, doi: 10.12948/issn14531305/19.2.2015.04.

[73] M. Adya and F. Collopy, “How effective are neural networks at forecasting and prediction? A review and evaluation,” *J. Forecast.*, 1998, doi: 10.1002/(sici)1099-131x(1998090)17:5/6<481::aid-for709>3.0.co;2-q.

[74] Zhang, G., E. Patuwo, and M. Y. Hu, “Forecasting with Artificial neural networds,” *Int. J. Forecast.*, 1998.

[75] A. D. Papalexopoulos, S. Hao, and T. M. Peng, “An implementation of a neural network based load forecasting model for the EMS,” *IEEE Trans. Power Syst.*, 1994, doi: 10.1109/59.331456.

[76] A. D. Papalexopoulos and T. C. Hesterberg, “A regression-based approach to short-term system load forecasting,” *IEEE Trans. Power Syst.*, 1990, doi: 10.1109/59.99410.

[77] B. F. Hobbs, “Analysis of the value for unit commitment of improved load forecasts,” *IEEE Trans. Power Syst.*, 1999, doi: 10.1109/59.801894.

[78] A. Khotanzad, R. C. Hwang, A. Abaye, and D. Maratukulam, “An Adaptive Modular Artificial Neural Network Hourly Load Forecaster and its Implementation at Electric Utilities,” *IEEE Trans. Power Syst.*, 1995, doi: 10.1109/59.466468.

[79] A. Khotanzad, R. Afkhami-Rohani, T. L. Lu, A. Abaye, M. Davis, and D. J. Maratukulam, “ANNSTLF - A neural-network-based electric load forecasting system,” *IEEE Trans. Neural Networks*, 1997, doi: 10.1109/72.595881.

[80] A. Khotanzad, R. Afkhami-Rohani, and R. Af, “ANNSTLF - Artificial neural network short-term load forecaster - generation three,” *IEEE Trans. Power Syst.*, vol. 13, no. 4, pp. 1413–1422, 1998, doi: 10.1109/59.736285.

[81] A. Khotanzad, E. Zhou, and H. Elragal, “A neuro-fuzzy approach to short-term load forecasting in a price-sensitive environment,” *IEEE Trans. Power Syst.*, vol. 17, no. 4, pp. 1273–1282, Nov. 2002, doi: 10.1109/TPWRS.2002.804999.

[82] P. R. J. Campbell and K. Adamson, “Methodologies for load forecasting,” 2006, doi: 10.1109/IS.2006.348523.

[83] M. H. Beale, M. T. Hagan, and H. B. Demuth, *Neural Network Toolbox TM 7 User ’ s Guide*. 2010.

[84] G. H. Yann LeCun, Yoshua Bengio, “Deep learning (2015), Y. LeCun, Y. Bengio and G. Hinton,” *Nature*, 2015.

[85] H. Shi, M. Xu, and R. Li, “Deep Learning for Household Load Forecasting-A Novel Pooling Deep RNN,” *IEEE Trans. Smart Grid*, 2018, doi: 10.1109/TSG.2017.2686012.

[86] D. Silver, J. Schrittwieser, K. Simonyan, I. A.- Nature, and U. 2017, “Mastering the game of Go without human knowledge,” *Nature*. 2016.

[87] Y. Cao, M. Raoof, S. Montgomery, J. Ottosson, and I. Näslund, “Predicting Long-Term Health-Related Quality of Life after Bariatric Surgery Using a Conventional Neural Network: A Study Based on the Scandinavian Obesity Surgery Registry,” *J. Clin. Med.*, 2019, doi: 10.3390/jcm8122149.

[88] B. Y. Goodfellow I., “Courville A-Deep learning-MIT (2016),” *Nature*, 2016.

[89] D. L. Marino, K. Amarasinghe, and M. Manic, “Building energy load forecasting using Deep Neural Networks,” *IECON Proc. (Industrial Electron. Conf.*, pp. 7046–7051, 2016, doi: 10.1109/IECON.2016.7793413.

[90] “Long Short Term Memory | Architecture Of LSTM.” https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/ (accessed Aug. 30, 2021).

[91] C. Olah, “Understanding LSTM Networks [Blog],” *Web Page*, 2015.

[92] H. J. Sadaei, P. C. de Lima e Silva, F. G. Guimarães, and M. H. Lee, “Short-term load forecasting by using a combined method of convolutional neural networks and fuzzy time series,” *Energy*, 2019, doi: 10.1016/j.energy.2019.03.081.

[93] I. Koprinska, D. Wu, and Z. Wang, “Convolutional Neural Networks for Energy Time Series Forecasting,” 2018, doi: 10.1109/IJCNN.2018.8489399.

[94] N. Singh, C. Vyjayanthi, and C. Modi, “Multi-step Short-term Electric Load Forecasting using 2D Convolutional Neural Networks,” 2020, doi: 10.1109/HYDCON48903.2020.9242917.

[95] R. Fukuoka, H. Suzuki, T. Kitajima, A. Kuwahara, and T. Yasuno, “Wind Speed Prediction Model Using LSTM and 1D-CNN,” *J. Signal Process.*, 2018, doi: 10.2299/jsp.22.207.

[96] A. Brunel *et al.*, “A CNN adapted to time series for the classification of Supernovae,” 2019, doi: 10.2352/ISSN.2470-1173.2019.14.COLOR-090.

[97] M. Imani and H. Ghassemian, “Sequence to Image Transform Based Convolutional Neural Network for Load Forecasting,” 2019, doi: 10.1109/IranianCEE.2019.8786456.

[98] R. Garg, B. G. Vijay Kumar, G. Carneiro, and I. Reid, “Unsupervised CNN for single view depth estimation: Geometry to the rescue,” 2016, doi: 10.1007/978-3-319-46484-8\_45.

[99] T. T. Um, V. Babakeshizadeh, and D. Kulic, “Exercise motion classification from large-scale wearable sensor data using convolutional neural networks,” 2017, doi: 10.1109/IROS.2017.8206051.

[100] Y. Zhang, S. Roller, and B. C. Wallace, “MGNC-CNN: A simple approach to exploiting multiple word embeddings for sentence classification,” 2016, doi: 10.18653/v1/n16-1178.

[101] E. Gawehn, J. A. Hiss, and G. Schneider, “Deep Learning in Drug Discovery,” *Molecular Informatics*. 2016, doi: 10.1002/minf.201501008.

[102] M. Cai, M. Pipattanasomporn, and S. Rahman, “Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques,” *Appl. Energy*, 2019, doi: 10.1016/j.apenergy.2018.12.042.

[103] “Convolutional neural networks for time series forecasting | Python for Finance Cookbook.” https://subscription.packtpub.com/book/data/9781789618518/10/ch10lvl1sec63/convolutional-neural-networks-for-time-series-forecasting (accessed Aug. 30, 2021).

[104] S. Khan, N. Javaid, A. Chand, A. B. M. Khan, F. Rashid, and I. U. Afridi, “Electricity Load Forecasting for Each Day of Week Using Deep CNN,” 2019, doi: 10.1007/978-3-030-15035-8\_107.

[105] A. Dedinec, S. Filiposka, A. Dedinec, and L. Kocarev, “Deep belief network based electricity load forecasting: An analysis of Macedonian case,” *Energy*, 2016, doi: 10.1016/j.energy.2016.07.090.

[106] S. Papadopoulos and I. Karakatsanis, “Short-term electricity load forecasting using time series and ensemble learning methods,” 2015, doi: 10.1109/PECI.2015.7064913.

[107] W. Kim, Y. Han, K. J. Kim, and K. W. Song, “Electricity load forecasting using advanced feature selection and optimal deep learning model for the variable refrigerant flow systems,” *Energy Reports*, 2020, doi: 10.1016/j.egyr.2020.09.019.

Appendix Title

Text begins here. To add additional chapters, simply start a new page and format title as the ‘Appendix’ style. Remove this text.

Glossary

Start writing here; remove page if there is no content

**Curriculum Vitae**

Candidate’s full name: Tolulope Oluwaseun Olugbenga

Universities attended:

BSc in Computer Science Engineering, University of Debrecen, 2018

Publications: None

Conference Presentations: None