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: Full Name, Degree, Department

This thesis is accepted by the

Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

December 2021

© Tolulope Oluwaseun Olugbenga, 2021

ABSTRACT

Load forecasting is critical for power system operators to maintain a safe and efficient network. By ensuring that consumers receive an adequate amount of energy, load forecasting helps maintain the supply-demand balance. Load forecasting can benefit companies such as load aggregators, power marketers, and independent system operators. Excess production and expense are a result of over-forecasting. An unexpectedly high load results in an electricity deficit. Both scenarios result in inefficient generation scheduling and technical difficulties for the operator. Developing a forecasting model for a particular power network is not straightforward. In load forecasting, statistical and machine-learning techniques have been used. Deep learning techniques have recently gained popularity due to their improved ability to interpret complex data relationships. The purpose of this study is to compare deep learning forecasting techniques to some conventional forecasting techniques currently used by utilities in order to determine whether deep learning can better meet 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 undertaken. It has demonstrated how much I can grow and achieve when I believe in myself and put in the effort. It was an eye-opening experience, and I am grateful that I did not give up and instead persevered 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. Then, after hearing what other students had to say about how difficult a thesis master's degree is, I began to doubt my abilities and became concerned that I would not be able to finish it.

I want to express my gratitude to my supervisors, Dr. Dawn MacIsaac and Dr. Julian Cardenas; without them, I would not have completed this program. I appreciate your patience and encouraging words, which reminded me that anything is possible and that all I need to do is keep going. I would also like to thank my family for always being there for me and constantly motivating me to finish this program.

If I am completely honest, the person who first entered the lab in December 2018 would not be able to complete this degree. In order to finish, I needed to improve both personally and intellectually. 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 had to do it all over again, I would 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](#_Toc84959270)

[DEDICATION iii](#_Toc84959271)

[ACKNOWLEDGEMENTS iv](#_Toc84959272)

[Table of Contents v](#_Toc84959273)

[List of Tables viii](#_Toc84959274)

[List of Figures ix](#_Toc84959275)

[List of Abbreviations x](#_Toc84959276)

[1 Introduction 1](#_Toc84959277)

[1.1 Significant Achievements and Contributions 3](#_Toc84959278)

[1.2 Outline of the Thesis 4](#_Toc84959279)

[2 Overview of Load Forecasting 5](#_Toc84959280)

[2.1 Overview of Load Forecasting Techniques 5](#_Toc84959281)

[2.1.1 Statistical and Machine Learning Techniques 5](#_Toc84959282)

[2.1.2 Deep Learning Techniques 6](#_Toc84959283)

[2.1.3 The Myth of Finding the One Size Fits All Technique 9](#_Toc84959284)

[2.2 Load Forecasting Horizons 9](#_Toc84959285)

[2.2.1 Very Short-Term Load Forecasting (VSTLF) 10](#_Toc84959286)

[2.2.2 Short Term Load Forecasting (STLF) 11](#_Toc84959287)

[2.2.3 Medium-Term Load Forecasting (MTLF) 12](#_Toc84959288)

[2.2.4 Long Term Load Forecasting (LTLF) 13](#_Toc84959289)

[2.3 Factors that affect the load demand 13](#_Toc84959290)

[2.3.1 Historical load 13](#_Toc84959291)

[2.3.2 Economic Factors 14](#_Toc84959292)

[2.3.3 Chronological Factors 15](#_Toc84959293)

[2.3.4 Meteorological Factors 15](#_Toc84959294)

[2.3.5 Random Factors 17](#_Toc84959295)

[2.4 Description of the Benchmark Techniques 17](#_Toc84959296)

[2.4.1 The Seasonal Naïve Forecaster (SNF) 18](#_Toc84959297)

[2.4.2 The Multiple Linear Regression Forecaster (MLR) 19](#_Toc84959298)

[2.4.3 The Auto-Regressive Integrated Moving Average Forecaster (ARIMA) 20](#_Toc84959299)

[2.4.4 Artificial Neural Networks (ANNs) 23](#_Toc84959300)

[3 Investigation 31](#_Toc84959301)

[3.1 Datasets 31](#_Toc84959302)

[3.2 Peak Load Demand 32](#_Toc84959303)

[3.3 The Deep Learning Techniques 33](#_Toc84959304)

[3.3.1 The Long Short Term Memory Forecaster (LSTM) 35](#_Toc84959305)

[3.3.2 The Convolutional Neural Network Forecaster (CNN) 38](#_Toc84959306)

[3.4 Implementation Specifications for Algorithms 42](#_Toc84959307)

[3.4.1 The Benchmark Forecasters 44](#_Toc84959308)

[3.4.2 The Deep Learning Forecasters 47](#_Toc84959309)

[3.5 Algorithms' Performance 48](#_Toc84959310)

[3.5.1 Overall Performance 48](#_Toc84959311)

[3.5.2 Daily Peak Accuracy 49](#_Toc84959312)

[4 Results and Discussion 51](#_Toc84959313)

[4.1 Performance Metrics 51](#_Toc84959314)

[4.2 Comprehensive Evaluation of Our Forecasters' Performance 53](#_Toc84959315)

[4.2.1 A Brief Note on Peak Detection Accuracy 53](#_Toc84959316)

[4.2.2 Analyses Based on Datasets 54](#_Toc84959317)

[5 Conclusion 61](#_Toc84959318)

[5.1 Our Analysis in Summary 61](#_Toc84959319)

[5.2 Contributions 61](#_Toc84959320)

[5.3 Future Work 61](#_Toc84959321)

[Bibliography 64](#_Toc84959322)

[Appendix 85](#_Toc84959323)

[Glossary 86](#_Toc84959324)

Curriculum Vitae

List of Tables

[Table 1 – Indicates the Toronto Dataset's Overall Accuracy. 48](#_Toc84959262)

[Table 2 - Indicates the Ottawa Dataset's Overall Accuracy 48](#_Toc84959263)

[Table 3 – Indicates the MAPE and MAE Values for the Toronto Dataset's Peak Values and Time Difference. 49](#_Toc84959264)

[Table 4 - Indicates the MAPE and MAE Values for the Ottawa Dataset's Peak Values and Time Difference. 50](#_Toc84959265)

[Table 5 – Indicates the Formulas for Several Common Performance Metrics 51](#_Toc84959266)

[Table 6 - Depicts the Algorithm's MAPE Values Over an Hourly Time Period - Toronto Dataset 56](#_Toc84959267)

[Table 7 – Shows the Weekly MAPE Values for Each Day for the Algorithms – Toronto Dataset 58](#_Toc84959268)

[Table 8 – Displays the Monthly Average MAPE Values for Each Algorithm – Toronto Dataset 60](#_Toc84959269)

List of Figures

[Figure 1 - A Simple Network Versus a Deep Learning Network [41] 7](#_Toc84959247)

[Figure 2 - An Artificial Neuron’s Workflow 24](#_Toc84959248)

[Figure 3 - Examples of the Most Frequently Used ANN Activation Functions [122] 25](#_Toc84959249)

[Figure 4 - The Structure of a Simple Feed-forward ANN [123] 26](#_Toc84959250)

[Figure 5 - The Block Diagram of the Third Generation ANNSTLF [85] 29](#_Toc84959251)

[Figure 6 - The Long Short-Term Memory Structure [149] 36](#_Toc84959252)

[Figure 7 - An Architecture of a One-dimensional CNN for Time Series Data [164] 40](#_Toc84959253)

[Figure 8 – The Rectified Linear Unit Activation Function [165] 41](#_Toc84959254)

[Figure 9 – Examples of Max and Average Pooling [166] 41](#_Toc84959255)

[Figure 10 – The Structure of the BLF and CLF Network 46](#_Toc84959256)

[Figure 11 – Shows the Load Demand on March 11, 2019, and the CNN Forecast. 54](#_Toc84959257)

[Figure 12 - Shows the Test Dataset for the City of Toronto 55](#_Toc84959258)

[Figure 13 – Displays the Hourly Average Values for Each Hour - Toronto Dataset 56](#_Toc84959259)

[Figure 14 – Displays the Weekly Average Values for Each Day - Toronto Dataset 57](#_Toc84959260)

[Figure 15 – Displays the Monthly Average Values for Each Month – Toronto Dataset 59](#_Toc84959261)

List of Abbreviations

ANN – Artificial Neural Networks

ANNSTLF – Artificial Neural Networks Short Term Load Forecaster

ARIMA – Auto-Regressive Integrated Moving Average

BLF – Base Load Forecaster

CLF – Change in Load Forecaster

CNN – Convolutional Neural Networks

GRU – Gated Neural Networks

ICDAR - International Conference on Document Analysis and Recognition

ILSVRC - ImageNet Large Scale Visual Recognition Challenge

ISBI - International Symposium on Biomedical Imaging

LSTM – Long Short-Term Memory networks

LTLF – Long Term Load Forecasting

MAE – Mean Absolute Error

MAPE – Mean Absolute Percentage Error

MATLAB – Matrix Laboratory

MBE – Mean Biased Error

MICCAI – Medical Image Computing and Computer Assisted Intervention Society

ML - Machine Learning

MLP – Multilayer Perceptron

MLR – Multiple Linear Regression

MSE – Mean Squared Error

MTLF – Medium-Term Load Forecasting

MW – Mega-Watt

ReLU – Rectified Linear Unit

RLS – Recursive Least Squares

RMSE – Root Mean Square Error

RNN – Recurrent Neural Networks

SARIMAX - Seasonal Auto Regressive Integrated Moving Average With Exogenous Variables

SD – Standard Deviation

SNF – Seasonal Naïve Forecaster

STLF – Short Term Load Forecasting

SVM – Support Vector Machine

UNB – University of New Brunswick

VMD - Variational Mode Decomposition

VSTLF – Very Short-Term Load Forecasting

# 1 Introduction

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 [1]–[8]. Load forecasting is a critical building component for power system operators to ensure the network is continuously operating and managed safely and efficiently. An important objective for load forecasting is to ensure that consumers have an adequate energy supply to maintain the balance of supply and demand. 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], [9]. These organizations use load forecasting in power systems planning/operations, revenue projection, rate design, energy trading, and other activities [2], [3], [10].

Over the last decade, there has been a surge in the adoption of renewable energy and distributed generation sources and the advancement and implementation of smart grids and buildings to meet growing energy demands effectively. To integrate these developments without causing system disruptions, it is necessary to have reliable load forecasting across multiple time horizons [11]. Electric load forecasting is well studied [1], [7], [12], [13], and most current research focuses on developing more accurate forecasts. The demand patterns used to drive modern technologies are complex due to the deregulation of energy markets and several random factors, often governed by human behaviour, which must be considered to predict future electricity demand. Therefore, developing a forecasting model appropriate for a particular power network is not a simple task [3], [10], [14]. From a financial standpoint, over-forecasting, or forecasting more power than required, results in the start-up of an excessive number of generating units, resulting in over-production and unnecessary expense. Conversely, underestimating the required demand because of higher than expected loads results in electricity deficit. When this occurs, the system operator is forced to purchase potentially pricey peaking power to cover the difference at significantly higher than the market price. Both situations result in suboptimal generation scheduling and present technical challenges to the operator.

In February 2008, the Electric Reliability Council of Texas documented a power system incident that prompted them to respond to a big ramp down of wind generation by ramping up evening demand more quickly than predicted to maintain load/generation balance [15]. They drew on reserve power and relieved the load of power consumers who agreed to act as temporary curtailment loads during the event. After-incident analysis indicates that more precise forecasting of generation and demand may easily have averted the need for an emergency reaction. In September 2011, a heatwave in South Korea increased the electricity demand significantly. Due to a lack of available energy to satisfy the uptick caused by the heatwave, South Korea's power supply was disrupted for nearly 1.5 million people [16]. Although these scenarios are uncommon, they provide extreme examples of the potential repercussions of imbalance and, hence, the significant importance of accurate load forecasting.

In load forecasting, statistical and machine learning (ML) techniques have been applied. Deep learning techniques have gained popularity in recent years due to their ability to better interpret complex relationships in the data [2], [17]. The purpose of this work is to compare deep learning forecasting techniques against some conventional forecasters in use by specific utilities to determine if deep learning can better suit their needs.

## 1.1 Significant Achievements and Contributions

To begin, we selected three distinct datasets that aggregate data from three distinct cities: Toronto, Ontario, Ottawa, Ontario, and Saint John, New Brunswick. Environment Canada provided the weather data. We chose well-known benchmark algorithms that have been extensively used in the load forecasting literature. We chose two distinct deep learning algorithms, the CNN and the LSTM, that have demonstrated remarkable results when applied to a variety of problems. Additionally, these algorithms demonstrated promising results in the literature on load forecasting.

We implemented all benchmarks and deep algorithms and compared their overall performance as well as their accuracy in detecting daily peaks in the data. Our comparison demonstrates the additional value that deep learning algorithms provide over popular benchmark algorithms. We created algorithms that can identify complex data relationship without explicit user input. We developed algorithms that are adaptable to external factors such as annual increases in power demand or temperature shifts.

Finally, because we used publicly available data and algorithms with extensive documentation on how they were developed, this work will be reproducible and serve as a benchmark for future research both within and beyond our smart-grid team. This also contributes to the maturation of the ongoing debate over the use of deep learning methods in load forecasting that have not been thoroughly tested.

## 1.2 Outline of the Thesis

The following is the outline of this dissertation. Chapter 2 discusses load forecasting techniques, forecasting horizons, factors affecting electricity demand, and our benchmark algorithms. Chapter 3 summarizes the details of our implementation; it also includes a description of peak load demand and the deep learning techniques we used. Chapter 4 analyzes our findings and comparisons in detail, as well as the performance metrics we used in this study. Chapter 5 summarizes our findings, contributions, and future directions for research.

# 2 Overview of Load Forecasting

## 2.1 Overview of Load Forecasting Techniques

### 2.1.1 Statistical and Machine Learning Techniques

Statistical techniques and machine learning (ML) have both been used to forecast load, and with the widespread adoption of data science, the line between these two approaches is becoming increasingly ambiguous [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 [24]. Examples are Artificial Neural Networks (ANNs) [25][26], Fuzzy Regression Models [27], [28], Support Vector Machines [29], Gradient Boosting Machines [30]; they have all been applied to electrical load forecasting.

The authors of [12] discussed many regression-based approaches for forecasting within 2-week horizons STLF [31]. Another study [32] examined various Multiple Linear Regression (MLR) algorithms for load forecasting. The ARIMA model is the most frequently used among all regression models since it consistently produces good prediction results; for example, the author in [33] combined ARIMA and Box-Jenkins methods to do hourly forecasting. []

### 2.1.2 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” [34]. Although this kind of model has existed for a long period, it has only recently garnered widespread popularity. A pivotal moment in developing deep learning occurred in 2006 when Geoffrey Hinton et al. [35] proved that a greedy layer-wise pre-training strategy enabled the construction of deep belief networks (DBNs). Geoffrey Hinton’s quantum leap in inventing an effective neural network training algorithm paved the way for deep learning implementations [36]. Previously, the utility of deep architectures had been limited by their proclivity to become stuck in suboptimal solutions. Since then, research has established that the same idea holds for different types of networks [37], ushering in the deep learning era.

Goodfellow et al. [38] examine the concept of expanding network depth to improve generalization error rather than merely increasing the size of a single hidden layer. Numerous neurons may be required to accurately represent input data in a one-layered network, which is solved more effectively by shifting to a deeper design. Deep learning's success has been linked to the numerous levels of representation introduced by multiple layers [39]. By providing more powerful outputs than conventional benchmark neural networks, deep learning has revolutionized research fields like image processing and sequence learning. Their popularity is primarily due to their success in resolving a wide variety of previously believed unsolvable issues using shallow networks. 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 [40]. 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.

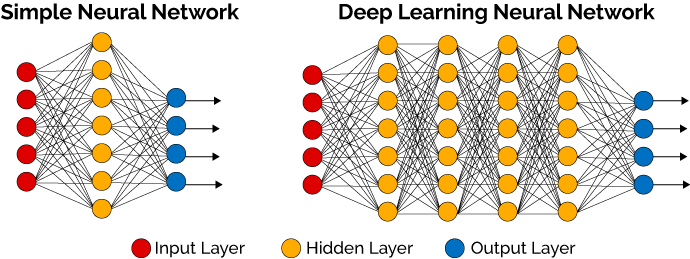


Figure - A Simple Network Versus a Deep Learning Network [41]

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 [42]. Reinforcement learning has also benefited from the revolution in deep learning. Mnih et al. [43] experimented with demonstrating that deep networks may be trained to perform at a professional human level when playing computer games. 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 [44]. 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.

Deep learning is a class of networks that encompasses a variety of architectures. Deep neural networks, recurrent neural networks, long short-term memory networks, deep belief networks, and convolutional neural networks are the most prevalent. 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. As illustrated in Figure 1, a neural network with more than three layers—including the inputs and outputs—is referred to as a deep learning technique. A neural network with only two or three layers is referred to as a simple neural network.

The annual ImageNet Large Scale Visual Recognition Challenge (ILSVRC) is a competition for computer vision. For the first time, a Convolutional Neural Network (CNN) won this competition in 2012, substantially lowering the error rate from 26.1 percent to 15.3 percent [45]. Since then, deep learning models have reduced the error rate to 3.57 percent by employing residual nets with a depth of up to 152 [46]. The authors ascribed the network's strong performance to its depth. Other competitions have been won by CNNs, including the ICDAR Chinese handwriting competition [47], the ISBI image segmentation competition [48], and the MICCAI Grand Challenge on cancer detection from medical pictures [49].

They have also had a significant impact on speech recognition. Dahl et al. [50] and Seide et al. [51] transcribed voice data using DBNs. Dahl et al. [50] discovered that increasing the depth of their model from one to eight hidden layers consistently increased performance. Additionally, their approach outperformed earlier models generated for the same dataset by around 2%. Abdel-Hamid et al. [52] utilized CNNs for speech recognition. In comparison to deep belief Hidden Markov Models, a CNN lowered the error rate on a benchmark phone call dataset by 6% – 10%. Deng et al. [53] predicted the same dataset using an ensemble deep learning approach, which improved the accuracy of single CNNs by around 1%.

### 2.1.3 The Myth of Finding the One Size Fits All Technique

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.

## 2.2 Load Forecasting Horizons

Electricity demand can be assessed periodically - hourly, daily, weekly, monthly, or yearly and forecasting 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) [24]. Short-term forecasting has been the focus in most current research, concentrating on horizons of less than two weeks; it is critical in the areas of planning, contingency analysis, load flow assessment, and power system planning and maintenance [1], [24], [54]. Disparities in time horizons have implications for the models and methodologies used in forecasting, and for what is available and selected for input data. The analyst must determine the most appropriate model type and the critical external factors that must be taken into account in order to obtain the most precise forecast [55].

### 2.2.1 Very Short-Term Load Forecasting (VSTLF)

VSTLF generates forecasts for loads up to one day in the future. Throughout the power industry, utilities and grid operators typically use such forecasts for real-time scheduling of electricity generation, load frequency control, and demand response. Very short-term load forecasts are also critical to retailers, power marketers, and trading firms' operations. VSTLF contributes to the immediate balancing of supply and demand. Trading in power markets is another application that relies on this type of forecasting. VSTLF is frequently viewed as a subproblem of short-term load forecasting (STLF) because both can use weather forecasts as forecasting period inputs. However, to achieve high accuracy over a very short time horizon, it is necessary to recognize the practical distinction between VSTLF and STLF.

From a modelling perspective, VSTLF models can incorporate lagged load as an independent variable in addition to those commonly used in STLF, such as weather and calendar variables (ex…). VSTLF, from an implementation standpoint, requires the model to be estimated quickly to produce the forecast on time. Additionally, the short lead time complicates the data collection process. While smart grid technologies have enabled the transmission of recent load data to the operation room, many power companies still lack access to high-quality load data for the most recent hour(s) when predicting the next hour's load [56].

The VSTLF literature has primarily focused on the modelling aspect. Researchers have experimented with a variety of techniques for forecasting the next few minutes to hours' load. Liu et al. compared five VSTLF techniques in [57]. The study concluded that it is possible to develop a simple, satisfactory dynamic forecaster capable of online prediction of very short-term load trends, and that Fuzzy Logic (FL) and Neural Networks (NN) are suitable candidates. Charytoniuk and Chen proposed an approach based on using a set of ANNs to model load dynamics rather than actual loads; they noted that it results in increased precision and reliability [58]. Taylor evaluated various methods for VSTLF using minute-by-minute observations of British electricity demand. The optimal results were obtained by combining the Holt-Winters' adaptation and a novel intraday cycle exponential smoothing method [59].

### 2.2.2 Short Term Load Forecasting (STLF)

According to Mandal et al. [60], STLF is a critical instrument in a utility system’s day-to-day operations and planning activities, such as energy transactions, hydrothermal coordination, unit commitment, calculating load flows, economic dispatch, security analysis, fuel scheduling, unit maintenance, and making decisions to avoid overloading. STLF estimates load up to two weeks in advance. STLF is a complex process influenced by various factors, including economic conditions, time of day, season, weather. The electricity demand is determined by meteorological variables, human social activities, and industrial activities.

This area has become increasingly important in recent years due to two main factors: deregulation of the power systems, which introduces new challenges to the forecasting problem, and the fact that no two utilities are identical, necessitating a detailed case study analysis of the various geographical, meteorological, load type, and social factors affecting load demand [61]. Hippert et al. [62] 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 [63]. Additionally, the forecaster must model the relationship between the load and other variables such as weather, leisure activities, and so on.

### 2.2.3 Medium-Term Load Forecasting (MTLF)

MTLF is another type of load forecasting which operates on a longer timescale, ranging from two weeks to three years. The MTLF guides decisions about network operations, schedule maintenance, fuel procurement for power plants, capacity planning and infrastructure development, and financial budgeting [64]. Additionally, MTLF enables a company to forecast load demand over a longer time, which can aid in negotiations with other companies. Demographic and economic factors influence MTLF. MTLF typically produces the daily peak and average load [65], [66]. MTLF and STLF have a close association; long-term decision-making must be integrated into short-term decision-making. This coordination between different decision-making levels is critical to ensure that specific operational objectives that develop in the medium term are explicitly considered in the short term [67]. Additionally, coordination between decision-making levels has become critical for generation businesses seeking to boost their profitability.

### 2.2.4 Long Term Load Forecasting (LTLF)

LTLF is the final type of load forecasting. LTLF covers a period of more than three years. LTLF is required for planning purposes, such as constructing new power plants, expanding the transmission system, and electric utility expansion planning. There are indicators affecting LTLF in terms of demographic and economic development. The population growth, industrial expansion, local area development, gross domestic product, and annual energy consumption in the past are all factors to consider. Annual peak load demand and annual energy demand for the years ahead are the outputs of the LTLF [68]. Although these longer-term forecasts receive less attention than their more visible short-term counterparts, their inaccuracy has significant financial consequences. It may result in either wasted investment in new generation facilities or a shortage of supply capability when there is an under-forecast.

## 2.3 Factors that affect the load demand

Different factors can affect load demand, such as the region in question, 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). These factors can be classified into five distinct categories: historical load, economic, chronological, meteorological, and random.

### 2.3.1 Historical load

Hippert et al. stated in [62] that forecasting load is difficult because the load series is complex and exhibits multiple levels of seasonality: 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 the previous day and on the load at the same hour on the previous week's day with the same denomination. As a result, historical load data is used as input to models for forecasting short-term load. The literature makes extensive use of multiple input variables.

For example, Houimli et al. [69] forecasted the subsequent 48 half-hourly loads using the previous 48 half-hourly loads, Park et al. [70] used the previous two hours of load data to forecast the next hour, Bakirtzis et al. [71] used the previous two days of hourly load data to forecast the next day's hourly load, and Velasco et al. [72] used the previous 24 hours of load data to forecast the next. As demonstrated, the literature uses a wide variety of historical load input data, but there is agreement that some sort of historical load input is necessary.

### 2.3.2 Economic Factors

Economic factors include capital expenditure in the facility's infrastructure by constructing new buildings, laboratories, and experiments that increase the load to the electric grid. The location's funding profile dictates what equipment, processes, and experiments can be conducted and when. Demand pricing and demand management programs influence how customers use electricity during system peaking [73]. Economic factors will have little effect on short-term load forecasting because they usually affect consumption patterns over a more extended period [1]. However, economic factors might serve as an impetus for examining a system's load pattern and adopting load reduction strategies.

### 2.3.3 Chronological Factors

Seasonal, weekly, and daily cycles and holidays can influence load [73], [74]. Autumn and spring often have a lower load. Weekdays differ from weekends, with weekends having a lighter load [62], [74]. In general, public holidays have a lower load than weekdays and are more comparable to weekends. The load on days immediately preceding and following the holiday is also impacted [73]. Because time influences how electricity is used, it is incorporated into load forecast models using calendar data [75]. Some, or all, of the patterns, might be considered. The pattern of weekday-weekend/holiday can be explained by establishing distinct models for each category [76]. Alternatively, many indicator variables could be used [9], [72]. Weekends and holidays are particularly difficult for studies that do not differentiate between these days [77]. Other patterns are accounted for using variables such as the hour of the day, the day of the week, the month, and the week number [9], [69], [72]

### 2.3.4 Meteorological Factors

The most frequently used and most significant weather variable is temperature [55], [62], [70], [73], [78]. The majority of load forecasting models incorporate one or more temperature-related variables [79]. The relationship between temperature and load is non-linear. According to Hong and Shahidehpour [80], temperature factors alone can account for more than 70% of the variability in load. This nonlinear relationship contributes to the widespread use of nonlinear approaches for load forecasting [1], [62]. Since the early 1930s, the relation between temperature and load has been recognized [62].

Humidity, solar irradiance, wind speed, barometric pressure, and precipitation are other weather variables that might alter the electric hourly load profile. Days with high humidity require cooling equipment to operate at a higher duty cycle to remove surplus moisture from the conditioned air [81]. Long periods of high sun irradiation will radiantly heat the interiors of buildings, requiring cooling systems to run longer and with less diversity. Precipitation has a propensity to chill the air, hence decreasing the cooling load [73]. Wind speed and barometric pressure can also influence the hourly load profile and frequently do so in conjunction with other variables such as precipitation. Wind speeds may amplify the effect of low temperatures, resulting in a greater wind chill index as well as increased demand. Wind speeds greater than 15 mph generate renewable energy, reducing the reliance on central sources of generation.

Janicki provides an in-depth description of the many types of meteorological variables that are utilized in load forecasting, as well as instances of their application in the literature [82]. The efficacy of these variables in forecasting load varies according to geographic location, industry, and regional climate. Friedrich and Afshari [83] discovered that incorporating four meteorological variables (temperature, specific humidity, wind speed, and sun irradiation) produced more accurate findings than relying solely on temperature. This was not a direct comparison because the models were of two distinct types (ANN vs. transfer function). Taylor and Buizza [84] employ a variety of meteorological factors. They modify the model's temperature, wind speed, and cloud cover to employ effective temperature, wind cooling power, and lighting. They do not make comparisons to a model based solely on temperature. Khotanzad et al. [85] use an effective temperature to adjust for humidity and wind speed. Specific studies concentrating exclusively on temperature imply that additional weather variables could be incorporated to improve forecasts [9], [70].

For load forecasting, the location of the weather data input must be determined. Forecasting loads can be pretty location-specific, such as forecasting for individual buildings or local regions. This form of forecasting is possible due to the availability of smart grid data, although system load forecasting is still required. Utilizing weather data for a significant load center is one approach of selecting weather data (e.g., Toronto used for Ontario). Weather stations located throughout a region can also be averaged to provide a single input variable [86]. Additionally, weather station selection can be used to discover which stations are the most accurate predictors of load [87]–[89]. Distributed or multi-region forecasting is a technique for anticipating load by utilizing meteorological data from different locations [90], which is particularly useful in vast geographic areas.

### 2.3.5 Random Factors

Random factors affecting the electrical load profile are other random disruptions in the load pattern that cannot be described by the preceding factors [73]. These disturbances might include considerable loads that operate on an ad hoc basis, making prediction impossible [91]. Other disruptions, such as extensive employee absenteeism (due to illness, severe weather, etc.) and planned or unforeseen power system outages can substantially impact the facility's load profile.

## 2.4 Description of the Benchmark Techniques

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], [3], [10], [14], [92]–[94].

### 2.4.1 The Seasonal Naïve Forecaster (SNF)

The naïve forecaster is a simple forecaster based on a random walk model [95]; it has often been implemented as a ground-level benchmark for developing more sophisticated forecasters [92], [96]–[98]. 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 [97] 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 [99]. 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 [96].

### 2.4.2 The Multiple Linear Regression Forecaster (MLR)

Multiple linear regression (MLR) is one of the most commonly used statistical techniques for load forecasting [19], [27], [32], [54], [92], [100]–[104]. 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 [105] 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 [106]. Additionally, MLRs are incapable of intelligently learning and adapting to data changes caused by newer factors.

#### 2.4.2.1 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.

### 2.4.3 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 [107]. The ARIMA forecaster is arguably one of the most popular and commonly utilized statistical forecasting techniques for load forecasting [108]. 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 [108]. 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 [104], [109]. 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) [110]–[112]:



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 [113], 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.

### 2.4.4 Artificial Neural Networks (ANNs)

Human brains are uniquely capable of comprehending the context of real-world situations in ways that machines cannot. Neural networks were designed to overcome this issue. Artificial neural networks were founded on the work of McCulloch and Pitts in 1943 [114], who developed a binary unit whose value is determined by the linear sum of the network's weighted inputs. Another seminal development occurred in 1949 when Hebb [115] proposed a learning rule stating that neuronal connections are adaptable and can be reinforced through the frequent activation of a neuron by another. The feed-forward network was invented by Rosenblatt's [116] perceptron network with signals connected in a single direction. Backpropagation [117] was a significant development that enabled multilayer perceptron network training.

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 [14], [118]–[120]. 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. A neural network is made up of neurons. The primary neuronal workflow can be separated into the following components, as illustrated in Figure 2. 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.

Diagram

Description automatically generated

Figure - An Artificial Neuron’s Workflow

The activation functions of an ANN are critical because they enable the solution of non-linear problems. Figure 3 shows some frequently used activation functions. The activation function has to be non-decreasing but differentiable [2], [110], as the backpropagation algorithm computes the error function's gradient. Linear transfer functions are typically used on neurons in the output layer, whereas tanh transfer functions are typically used on neurons in the hidden layer. If the neuron’s workflow does not include activation functions, an ANN will perform similarly to a linear regression model [121]. Neurons in an ANN can be classified into three layers, as demonstrated in Figure 4: input, hidden, and output.

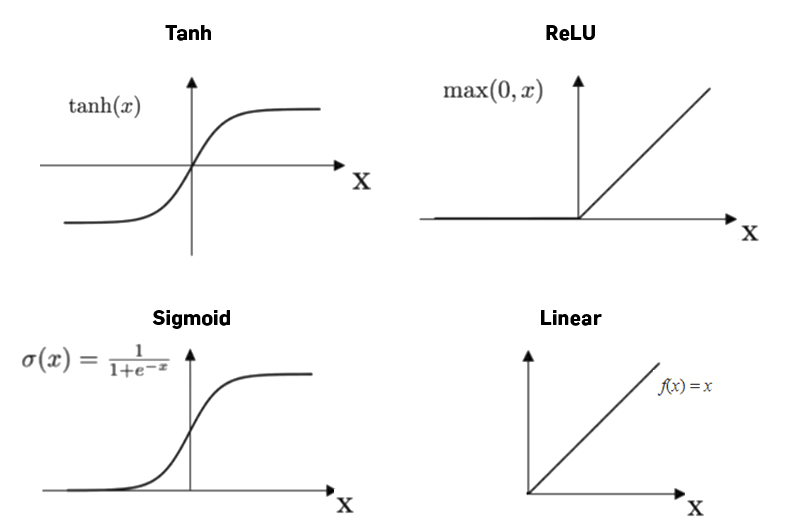


Figure - Examples of the Most Frequently Used ANN Activation Functions [122]

The multilayer perceptron (MLP) is one of the most well-known and widely used artificial neural networks due to its universal approximation capability and ability to scale well with input dimensions. It is a feed-forward network architecture with one or more hidden layers of connected neurons, as seen in figure 4. Each layer is connected in a single direction, and there are no connections between layers; thus, the term feed-forward. The network is presented with model inputs in the first layer, and the neuron inputs in subsequent layers are the outputs of all preceding layer neurons. The MLP is trained using a supervised learning algorithm in which each sample vector represents a set of inputs with desired outputs. The function that maps the inputs to the outputs should be written in such a way that it is generalizable to previously unseen inputs [114]–[117].

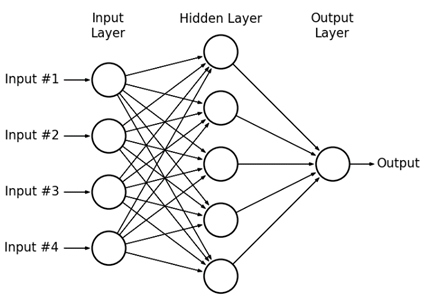


Figure - The Structure of a Simple Feed-forward ANN [123]

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 [9], [124]. 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. A Mean Square Error (MSE) is one of the available metrics in 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 [13]. 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. Back-propagation is a technique for training a network that involves computing the error signal at the output then propagating it back through the network layers. Using the chain rule, it calculates gradients for every hidden and output layer neuron to determine their error sensitivity.

Adjusting the weights and biases of a neural network continuously improves its ability to correctly predict the training dataset. This eventually results in overtraining and a decrease in the network's ability to generalize to previously unknown data. Cross validation enables us to avoid overtraining by comparing the algorithm's performance to its own test data; an illustration of this is provided in the following sentence. The training set is divided into two sections. The first set is used to train the artificial neural network, while the second set is used to validate it. This error typically decreases as the number of runs over the training set increases, until the ANN becomes over-trained, as indicated by a rise in this error. As a result, training is halted when the error on the validation set begins to grow.

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 [14]. Neural networks have produced excellent results in load forecasting [1], [13], [14], [30], [102], [118]. 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 [125] 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. [126] 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 [127] and [128], 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.

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

One of the most popular ML-based load forecasters is the ANNSTLF [1], [93], [104]. The ANNSTLF-G3 has improved prediction accuracy and generated economic benefits for over a dozen utilities [106], [129]. Some publications have named ANNSTLF-G3 the best forecaster for short-term load forecasting [1], [104]. The configuration of this load forecaster has undergone a few revisions since it was first proposed [130], [131], and we will discuss the third-generation design (G3) [85], which uses two shallow multi-layer feed-forward ANNs together with a recursive least squares (RLS) combiner to predict short-term load. Additional information about the RLS algorithm is available in [132]. The figure below shows the block diagram of the system:

Diagram, schematic

Description automatically generated

Figure - The Block Diagram of the Third Generation ANNSTLF [85]

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 [104], [133], [134]. Both blocks are presented with the same 79 inputs (see Figure 5) and output a 24x1 vector representing hourly forecasts. The CLF generates its final output by adding predicted changes to actual last-day values. 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.

# 3 Investigation

The purpose of this work was to determine whether deep learning approaches could improve forecasting accuracy for specific data sets by comparing the accuracy of deep learning forecasters to some of the current forecasters used by utilities. The work concentrated on STLF horizons because they are a critical tool in the day-to-day operations and planning of a utility system. To accomplish this, a CNN forecaster and an LSTM forecaster were compared agains 4 benchmark forecasters: a Seasonal Naive forecaster, a Multiple Linear Regression (MLR) forecaster, an Auto-Regressive Integrated Moving Average (ARIMA) forecaster, and a shallow Artificial Neural Network forecaster (ANN). These benchmark algorithms have been available for many years and have been implemented and used by researchers and utilities [1], [3], [10], [14], [92]–[94]. Performance of the deep learning forecasters was assessed by comparing them against the performance of the benchmark algorithms. Overall accuracy and accuracy in peak detection were compared. The peak demand forecasts are critical for securing adequate generation, transmission, and distribution capacity. Accurate peak forecasts improve capital expenditure, decision making and system reliability. .

## 3.1 Datasets

A significant strength of this work Three datasets were analaysed. 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 [135], and the other is from Toronto [135], 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 with city-wide Saint John load aggregates.

Weather data (temperature) was also used to augment the datasets for this work. These data were obtained from Environment Canada [136]. []

## 3.2 Peak Load Demand

Peak load refers to the maximum amount of energy that a consumer draws from the grid during a specified time period. The distinction between a peak and a spike is that a value is only deemed a peak if it persists for at least 15 minutes; anything less than 15 minutes is simply a random spike. Understanding peak load is critical for any business energy management strategy, as it is utilized to calculate a portion of the energy cost [137]. Peaks have three distinct characteristics: magnitude, temporal location, and width or duration. The temporal location of the peak is the most critical of all the characteristics, even more so than the peak's value. Knowing the precise time a peak will occur enables utilities to plan reserve power and demand response strategies in advance to help reduce the peak, which could result in significant savings for both the utility and its customers. Numerous electric utilities charge customers for peak load in addition to their consumption.

On the other hand, base load is the very minimum amount of electrical demand required over a 24-hour period. Also referred to as constant load, base load requirements are relatively constant. Consider the electricity requirements of a house. The base load is the constant electricity required by the electrical grid. Peak load occurs when additional power is required, such as when the entire family is at home watching television and consuming a large amount of electricity. It is a brief moment of high demand, as the family will soon be sleeping, shutting off the television and lights and consuming less electricity. The base load is steadier, but lower, because electricity is still required for heating, cooling, and power outlets, among other things. Peak load electricity is less predictable than base load [138]. It can surge when air conditioners are switched on or when a snowstorm hits and the heat must be turned up. Electricity is generally more expensive during peak periods.

We used daily peaks in this study, which included the peak's value and time of occurrence. The time difference, the value difference, and the mean absolute percent error were all calculated.

## 3.3 The Deep Learning Techniques

Deep learning approaches have had remarkable success in the last few years at handling complex sequential data [139], [140]. 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 [141]. 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) [8], long-short-term-memory network (LSTM) [5], and the 1-D convolution neural network (CNN) [2], [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 [142] 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 [4] offered a novel parallel model that is a combination of convolutional neural networks (CNN) and recurrent neural networks (RNN). Additionally, 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. Regarding how RNNs work, the authors in [143] conducted an appropriate study on these networks.

Similar to [4], the authors of [144] 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 [26] 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 [145]. During the training phase, data from the preceding seven days was used. They compared the proposed model’s performance to five different machine learning techniques using the Root Mean Square Error (RMSE) and Mean Absolute Percent Error (MAPE) metrics. The findings indicated that the Deep-Energy model could make accurate short-term load predictions than the other models.

In another paper [146], 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, United States 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 [147], 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.

### 3.3.1 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 [148]. 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 [6], [7], [119]. 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 [119]. LSTM is perhaps the most well-known deep learning architecture for time series forecasting, which is built specifically to remember past data to retrieve it at a suitable time in the future to produce the output prediction.

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[119] et al. argue that LSTM is better than other deep neural networks because of its memory cell configuration.

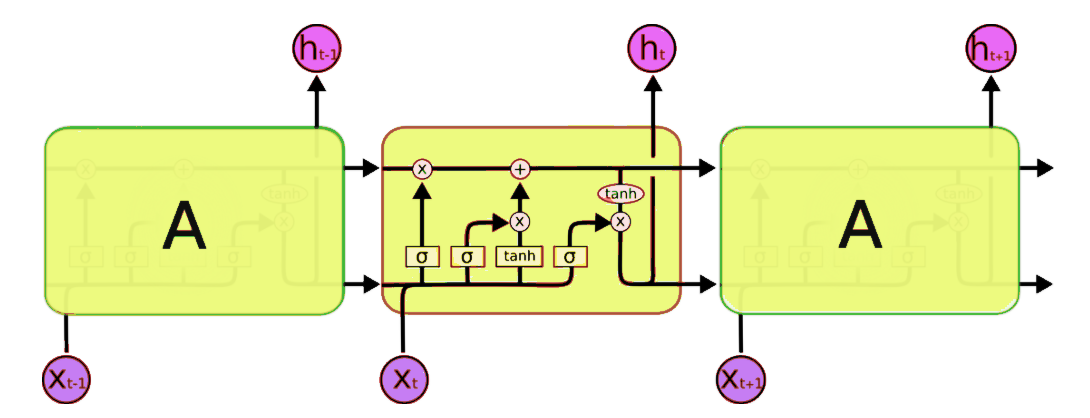


Figure - The Long Short-Term Memory Structure [149]

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 6 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 can remove or add 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 [150], [151]. 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. Bouktif et al. [152] are one of the authors who applied the LSTM to load forecasting. They examined half-hourly French electricity demand from 2008 to 2016. 70% of the data has been used to train the model, while 30% was used as the test set. A genetic algorithm was used to determine the optimal time lags to include in the input vector and the appropriate amount of stacked LSTM layers. The final structure used six LSTM layers with 100, 60, and 50 cells, and 100-time lags were transmitted into the input layer. The test data revealed a mean absolute error of 250 MW and a root mean square error of 341 MW.

### 3.3.2 The Convolutional Neural Network Forecaster (CNN)

In recent years, Convolutional Neural Networks (CNNs) have gained the attention of researchers studying load forecasting [2], [4], [141], [153], [154]. CNNs are a type of deep learning network used for data processing with a grid-like topology [2], [148], [155]. This can comprise time series and image data, which can be viewed as a one-dimensional and two-dimensional data grid, respectively [2], [155]–[157]. CNN is like the ANN in that it is a feed-forward neural network designed to mimic human neurons [2], [34]. 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], [158]–[163]. 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], [142]. In at least one of its layers, CNN employs a particular linear mathematical technique called convolution [148].

Convolution is performed in CNNs by repeatedly applying filters or kernels to the input data to build a feature map. CNNs are used to extract a large number of features. As a result, a CNN may perform the convolution process multiple times in each network's convolution layers. The number of times the convolution process is performed is determined by the number of filters in the layer, which the operator can specify. Each kernel will focus on a distinct feature of the input data. The convolutional layer performs three distinct actions. The feature map is created because of the first procedure mentioned above.

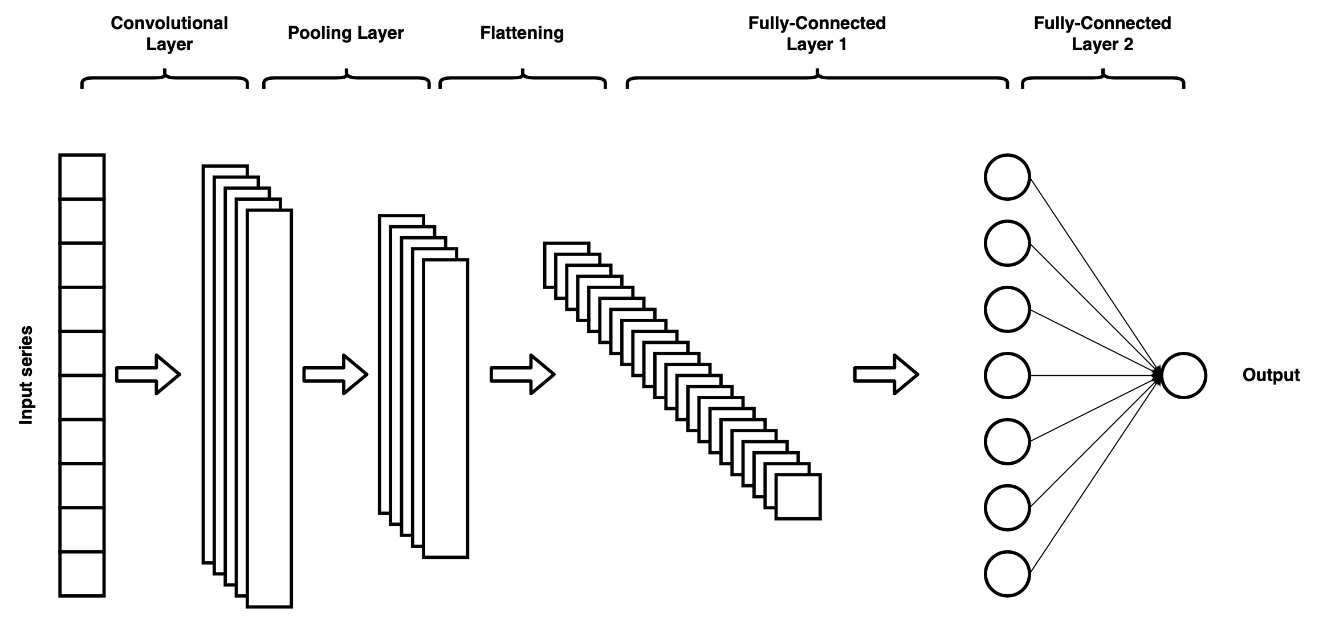


Figure - An Architecture of a One-dimensional CNN for Time Series Data [164]

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. Convolution is a linear process in and of itself. The rectified linear unit (ReLU) activation functions utilized in the convolutional layers introduce non-linearity. ReLU is a linear piecewise function. Because they behave similarly to linear functions, they are simple to create and train. When non-linear activation functions are utilized, propagating errors through multiple layers of a network frequently results in the so-called "vanishing gradient" problem, which inhibits deep networks from learning effectively. This is overcome by employing an activation function with similar qualities to that of a linear function. Similar to the sigmoid activation functions, the ReLU activation function squashes the inputs z, clamping negative values to zero, as shown in figure 8.



Figure – The Rectified Linear Unit Activation Function [165]

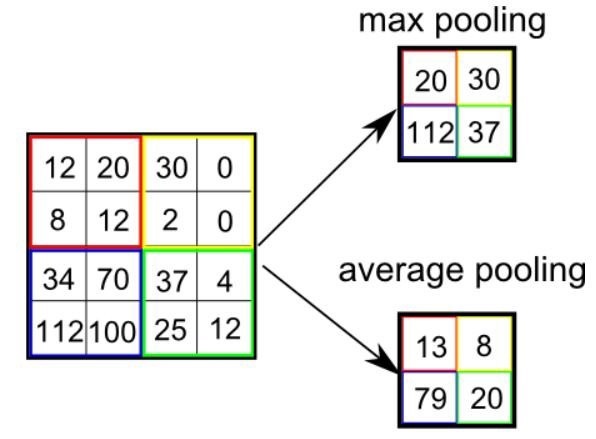


Figure – Examples of Max and Average Pooling [166]

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 neighbourhood [148]. Pooling layers are used to lower the size of the previous layer's output. A single value represents the pool from the output of a specified pool of neighbouring neurons from the preceding layer. In other words, the pooling layer aggregates the responses from individual areas into a single value. For instance, as seen in Figure 9, a max-pooling operation keeps the highest value inside a region as the item to pass through to the next layer. As a result, the following layer processes fewer inputs, increasing computing efficiency.

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. When Amaradinghe[2] 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.

## 3.4 Implementation Specifications for Algorithms

All forecasters were implemented using MATLAB version R2021b. The training dataset for both the Toronto and Ottawa datasets are 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..

A Hampel filter was used to find and replace outliers in the datasets. The filter is a sliding window with a variable width that glides across the time series. The filter determines the median and standard deviation for each window. If a point in the window is more than three standard deviations from the window's median, the Hampel filter flags it as an outlier and substitutes it with the window's median [167]. We used MATLAB’s default median window, which contains the sample and six adjacent samples, three on each side.

Prior to passing our datasets to the algorithms, we normalized the variables using the datasets maximum and minimum values. This step was carried out for both the load demand and temperature variables. The equation below illustrates the normalization formula we used.



where  denotes the normalized value,  denotes the actual value,  denotes the variable's minimum value in the timeseries, and denotes the variable's maximum value in the timeseries. Panigrahi et al. stated in [168] that data normalization has a significant effect on the performance of any model because its sole purpose is to ensure the quality of data before it is fed to the model. Additionally, they stated that the literature indicates that normalization techniques have a significant impact on a model's performance and that the appropriate normalization technique should be chosen based on the problem and model.

The normalization method described above is called Min-Max normalization; it scales the values between zero and one; the normalized values are dependent on their proximity to or separation from the set's minimum and maximum values. The minimum and maximum values were saved and used to de-normalize the final forecasts prior to the calculation of any performance metrics. Numerous researchers have used this method of normalization in the area of load forecasting [72], [169]–[172].

### 3.4.1 The Benchmark Forecasters

#### 3.4.1.1 The Seasonal Naïve Forecaster (SNF)

The seasonal naive method was straightforward to implement; we used the previous week's hourly lag as the current hour's value. Additionally, we repeated the preceding step for each hour of our test set.

#### 3.4.1.2 The Multiple Linear Regression Forecaster (MLR)

The MLR forecaster was developed using ten independent variables (inputs), and one target variable, which was the actual demand at a particular hour. The independent variables were 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.

#### 3.4.1.3 The ARIMA forecaster

In the ARIMA model, these hyperparameters were used for the Toronto, Ottawa, and Saint John datasets, respectively: (24, 2, 25), (23, 2, 24), and (24, 2, 24). After much trial and error and utilizing information from the partial autocorrelation and autocorrelation functions, we were able to determine p and q respectively. The parameters listed above were optimal for the datasets we used. Finding the optimal number of differencing ‘d’ needed was relatively simple; we simply used the auto-correlation function of the data to determine when it was under or over-differenced. The proper order of differencing is the smallest amount of differencing required to obtain a near-stationary series that oscillates around a defined mean and the auto-correlation plot rapidly approaches zero.

Partial autocorrelation can be thought of as the correlation between the time series and its lag after the intermediate lags are eliminated. Thus, partial autocorrelation conveys the unambiguous correlation between a lag and the series. This way, we can know whether or not the lag is required in the AR term. The auto-correlation plot can be utilized to determine the number of MA terms that is appropriate. A MA term is theoretically the lagged forecast's error. At times, our time series may be under- or over-differenced. The first problem can be solved by adding one or more AR terms, while the second problem can be solved by adding one or more MA terms.

#### 3.4.1.4 The ANNSTLF-G3 Forecaster

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 [173]. 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.

A picture containing text, clock

Description automatically generated

Figure – The Structure of the BLF and CLF Network

##### 3.4.1.4.1 The ANNs' Inputs That Were Used

As demonstrated in the section on artificial neural networks, the input set for ANNs consists of the following: the previous day's actual load (24 inputs), the previous day's actual temperature (24 inputs), the current day's actual temperature (24 inputs), and the current day's type (7 inputs containing ones and zeros). Both the BLF and the CLF used the same set of inputs, but their target variables were distinct; the BLF focused on actual load demand, whereas the CLF focused on load changes from yesterday to today. The final CLF forecasts were calculated by adding the predicted changes to the actual load from yesterday. The BLF and CLF used a total of 79 inputs, as illustrated in the ANNSTLF architecture.

### 3.4.2 The Deep Learning Forecasters

#### 3.4.2.1 The LSTM Forecaster

Other researchers on UNB's smart-grid team used the LSTM algorithm to forecast load, but only on the Saint John dataset. We modified their implementation to accommodate all of our datasets and input feature sets. Additionally, because the ANNSTLF structure was recognized as the best forecaster for short-term load forecasting [1], [104], we emulated the ANNSTLF structure by developing a Base Load Forecaster, a Change in the Load Forecaster, and an RLS combiner using the LSTM algorithm instead of the ANN. The architecture was identical to the ANNSTLF in terms of inputs and structure, but the BLF and CLF algorithms were trained using LSTMs. We were curious to see if this adjustment would result in an improvement in forecasting performance.

#### 3.4.2.2 The CNN Forecaster

We implemented the CNN algorithm similarly to the LSTM using the ANNSTLF structure. The CNNs used in this study have a six-layer architecture: 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 CNNs were trained using the adam optimization training algorithm. Additional details about this algorithm is available in [174].

## 3.5 Algorithms' Performance

### 3.5.1 Overall Performance

This section will examine the algorithms' overall performance on the three distinct datasets. The MAPE and RMSE values will be used to determine the performance in this case. In general, the lower the MAPE and RMSE values, the better the algorithm's performance. A quick note about the algorithms that make use of the RLS combiner, which include the CNN, LSTM, and ANN algorithms; for the sake of simplicity, only the RLS combiner results will be used in this analysis.

// Would need to add the results from the LSTM later

// Saint John Dataset to be added

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metrics** | **CNN** | **LSTM** | **ANN** | **MLR** | **ARIMA** | **SNF** |
| **MAPE (%)** | 2.16 |  | 2.30 | 3.75 | 4.86 | 6.09 |
| **RMSE (MW)** | 189.76 |  | 201.32 | 293.94 | 418.11 | 488.07 |

Table 1 – Indicates the Toronto Dataset's Overall Accuracy.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metrics** | **CNN** | **LSTM** | **ANN** | **MLR** | **ARIMA** | **SNF** |
| **MAPE (%)** | 2.72 |  | 3.09 | 4.78 | 5.06 | 7.33 |
| **RMSE (MW)** | 37.13 |  | 41.93 | 65.77 | 70.65 | 102.83 |

Table 2 - Indicates the Ottawa Dataset's Overall Accuracy

According to the MAPE and RMSE values in Tables 1 and 2, the best performing algorithms in the Toronto and Ottawa datasets are, in order, the CNN, ANN, MLR, ARIMA, and SNF. Given the similarity of the MAPE and RMSE values, it is reasonable to conclude that the CNN and SNF perform the best and worst overall on these datasets, respectively. It is encouraging that the SNF algorithm performs the worst of all algorithms; this indicates that all of the other algorithms contribute value to the forecasts in some way. Chapter 4 will analyze the algorithms' performance in detail.

### 3.5.2 Daily Peak Accuracy

This section will discuss the algorithms' accuracy in identifying daily peaks. We will calculate the mean absolute time difference (MAE) between the actual and forecasted time of occurrence, as well as the mean absolute percentage difference (MAPE) between the actual and forecast peak values. The reason we chose the MAE over the mean biased difference (MBE) is that positive and negative time differences can cancel out, resulting in biased results; however, in this case, we are using the absolute value of each time difference. It is worth noting that the MAEs of the time difference are denoted in the tables below by "hh:mm:ss", where hh denotes hours, mm denotes minutes, and ss denotes seconds.

// Would need to add the results from the LSTM later

// Saint John Dataset to be added

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metrics** | **CNN** | **LSTM** | **ANN** | **MLR** | **ARIMA** | **SNF** |
| **MAPE of the Peaks** | 2.36 |  | 2.45 | 3.65 | 5.28 | 6.18 |
| **MAE of the Time** | 01:19:36 |  | 01:33:07 | 01:36:09 | 01:43:33 | 01:32:23 |

Table 3 – Indicates the MAPE and MAE Values for the Toronto Dataset's Peak Values and Time Difference.

Table 3 shows the overall analysis of the algorithms utilizing the value and time differences in the Toronto dataset. The highest performing algorithms, from left to right, are the CNN, ANN, MLR, ARIMA, and SNF, as assessed by the MAPE of the value differences. The highest performing algorithms, from left to right, are the CNN, SNF, ANN, MLR, and ARIMA, as assessed by the MAE of time differences. In both value and time difference comparisons, the CNN algorithm outscored all other algorithms. The SNF fared poorly in comparisons of value differences, whereas the ARIMA performed poorly in comparisons of time differences.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metrics** | **CNN** | **LSTM** | **ANN** | **MLR** | **ARIMA** | **SNF** |
| **MAPE of the Peaks** | 2.38 |  | 2.78 | 4.21 | 5.00 | 6.98 |
| **MAE of the Time** | 00:51:45 |  | 01:03:47 | 01:10:21 | 01:11:10 | 01:18:34 |

Table 4 - Indicates the MAPE and MAE Values for the Ottawa Dataset's Peak Values and Time Difference.

The overall analysis of the algorithms based on the value and time differences in the Ottawa dataset is shown in Table 4. The best performing algorithms, from left to right, are the CNN, ANN, MLR, ARIMA, and the SNF. All other algorithms were outperformed by the CNN algorithm. The SNF fared the worst in comparisons of both value and time differences. Chapter 4 will contain additional details and information analyzing the performance of all the algorithms.

# 4 Results and Discussion

## 4.1 Performance Metrics

This study will compare the performance of all forecasters across all datasets, examining both overall performance and performance on a hourly, daily, and monthly timeframe. It will aid us in identifying instances when forecasters outperform or underperform expectations. The performance will be evaluated based on forecast accuracy and peak load localization accuracy. Table 1 delineates the main error measures used to quantify accuracy:

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

Table 5 – Indicates the Formulas for Several Common Performance Metrics

Mean Absolute Error (MAE) is the simplest way to measure forecast error [17], 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], [175]. 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) [26]. 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 [176], [177]. 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.

Our primary focus in this study will be on the MAPE and RMSE, as these are the most frequently used metrics for load forecasting. The MAPE's limitations do not apply to our datasets because there are no values very close to zero, and the RMSE enables us to detect large forecast errors. This document's appendix will include information on the overall performance of all metrics, including the ones mentioned above.

## 4.2 Comprehensive Evaluation of Our Forecasters' Performance

In Section 3.5, we examined the overall performance of all algorithms using both regular load forecasts and daily peak forecasts from the various datasets. All load demand variables are expressed in megawatts (MW).

### 4.2.1 A Brief Note on Peak Detection Accuracy

Concerning the accuracy of peak detection, it's worth noting that daily peaks are affected by a variety of random variables, making their prediction difficult. This is because random peaks can result in a significantly higher peak than the regular peak, and because we are calculating the daily maximum, we use the random peak rather than the regular peak.

The Toronto dataset typically peaks between 15:00 and 20:00 in the evenings, but as shown in the figure below, a random peak occurred at 10:00 with a value of 6594 MW, which was higher than the second highest peak at 18:00 with a value of 6590 MW. It was only 4 MW higher, but because we used the daily maximum, the peak at 10:00 was recorded as the daily maximum. The CNN, however, predicted a peak at 18:00 with a value of 6603 MW. Because we will compare the predicted time to the one at 10:00, the random peak has an effect on our time difference MAE accuracy metric. This is just something to keep in mind in terms of the algorithms' accuracy in detecting peaks. In future work, a significantly more accurate metric for comparing time differences could be used.



Figure – Shows the Load Demand on March 11, 2019, and the CNN Forecast.

### 4.2.2 Analyses of Individual Datasets

This section would analyze each dataset on an hourly, daily, monthly, seasonal, and daily peak basis. To keep the information contained within the scope of this thesis, only the most pertinent information is included here. Additional plots, such as box plots of the error distribution, plots of MAPE values over various horizons, and the source code for our implementation, will be included in the appendix.

Notably, the SNF average forecasts in the average figures' plots exhibit a high degree of similarity to the actual values' average because they are calculated using the previous week's values; thus, they are identical to the actual data but with a week lag. This means that calculating the SNF forecasts' averages over a long period will be nearly identical to calculating the test dataset's averages. This is critical to remember when examining the figure plots that are constructed using the averages of various horizons.

In the later chapters of each dataset, we will also analyze the algorithms' performance over the course of the year's four seasons, namely Winter, Summer, Autumn, and Spring. Winter is comprised of the months of December, January, and February. March, April, and May are considered spring. Summer is defined as June, July, and August. Autumn/Fall is comprised of the months of September, October, and November.

#### 4.2.2.1 The Toronto Dataset

The Toronto dataset is comprised of hourly load aggregation measurements taken throughout the city from 2010 to 2019. The years 2010-2018 were used to train the algorithms, and 2019 was used to test them. The figure below depicts the test dataset from January to December 2019.



Figure - Shows the Test Dataset for the City of Toronto

##### 

##### 4.2.2.1.1 The Hourly Horizon

The figure below illustrates the average demand for each hour in the testing dataset, while the table below summarizes the mean absolute percent error values for each algorithm when aggregated as average values for each hour using the Toronto dataset.



Figure 13 – Displays the Hourly Average Values for Each Hour - Toronto Dataset



Table 6 - Depicts the Algorithm's MAPE Values Over an Hourly Time Period - Toronto Dataset

The CNN makes the best overall prediction; it is similar to the ANN in some ways, but statistically produces superior results. Around 15:00, CNN and ANN make their worst predictions. The CNN and ANN error patterns are similar to those of the regular load in that the algorithms predicted quieter times more accurately than they predicted peak times of increased average demand. MLR and ARIMA performed poorly in comparison to the preceding. The MLR produces the most errors between 01:00 and 08:00, while the ARIMA produces the most errors between 07:00 and 10:00. The SNF made its worst prediction between 13:00 and 17:00; this is unsurprising given that it is based on previous week's values, and these are commonly used times for electricity. As a result of the information presented here and our statistical analysis, we can conclude that CNN was the clear winner when it came to hourly predictions for the Toronto dataset.

##### 4.2.2.1.2 The Daily Horizon

The figure below depicts weekly average values for each day of the week, including actual and forecast values, and the table below depicts MAPE values by weekday. Monday was the worst performing day for all algorithms; this could be due to something unique to the Toronto dataset, or it could be that this day is the most difficult to predict due to the fact that it is the first working day of the week.

On Sundays, CNN, ANN, MLR, and SNF all had their second-worst performances. Wednesdays and Fridays were the most predictable days for CNN. From Wednesday to Saturday, the ANN made similar predictions. ARIMA had a difficult time forecasting Mondays, with errors exceeding 10%. Thursdays were the easiest day for the ARIMA, SNF, and MLR. CNN's MAPE values were the lowest for all seven days of the week, making it the winner of the weekly predictions for each day of the week.



Figure 14 – Displays the Weekly Average Values for Each Day - Toronto Dataset

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Days** | **Monday** | **Tuesday** | **Wednesday** | **Thursday** | **Friday** | **Saturday** | **Sunday** |
| **CNN** | 2.64 | 2.12 | 1.91 | 2.04 | 1.88 | 2.06 | 2.45 |
| **LSTM** |  |  |  |  |  |  |  |
| **ANN** | 2.98 | 2.33 | 2.16 | 2.17 | 2.17 | 2.13 | 2.57 |
| **MLR** | 5.21 | 3.67 | 3.41 | 3.05 | 3.35 | 3.71 | 3.89 |
| **ARIMA** | 10.19 | 3.97 | 3.14 | 2.61 | 2.81 | 7.18 | 4.10 |
| **SNF** | 8.10 | 5.51 | 5.76 | 5.10 | 6.02 | 5.92 | 6.24 |

Table 7 – Shows the Weekly MAPE Values for Each Day for the Algorithms – Toronto Dataset

##### 4.2.2.1.3 Monthly Horizon

The figure below illustrates the average monthly actual load and forecast demand for each month of 2019, while the table that follows details the algorithms' average MAPE values. All algorithms struggled to forecast the month of July; this could be because July is the hottest month of the year and has the highest electricity demand. Additionally, because the summer air in Toronto is typically hot all day, including evenings, there are a lot of people who use air conditioning, which could result in a really high demand.

August was the second month that all algorithms struggled to forecast. The SNF and CNN had difficulty forecasting December; however, the ARIMA found December to be one of the easiest months to forecast. April was a difficult month to forecast for the ANN. The MLR and ARIMA had difficulty forecasting September. March and November were relatively easy for CNN, ANN, MLR, and ARIMA to forecast. March, October, and May were found to be the most predictable months by the SNF. The CNN and MLR found February's prediction relatively simple. And the ANN found June to be an easy month to predict.



Figure 15 – Displays the Monthly Average Values for Each Month – Toronto Dataset

It's critical to note, however, that just because an algorithm found some months easier to predict does not mean they offer the highest accuracy. This simply means that in comparison to other months, these were the easiest. According to the aggregate statistics, the CNN provided the most accurate forecasts on a monthly basis and was only slightly outperformed by the ANN in March and June. This results in CNN being the winner on a monthly average basis, as it had the best MAPE values in ten of the twelve months included in the test dataset.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Months | Jan | Feb | Mar | Apr | May | Jun | Jul | Aug | Sep | Oct | Nov | Dec |
| CNN | 1.83 | 1.66 | 1.78 | 2.18 | 2.02 | 1.99 | 3.48 | 2.95 | 2.21 | 1.90 | 1.57 | 2.23 |
| LSTM |  |  |  |  |  |  |  |  |  |  |  |  |
| ANN | 2.38 | 2.04 | 1.75 | 2.60 | 2.23 | 1.92 | 3.76 | 3.01 | 2.56 | 2.03 | 1.72 | 2.27 |
| MLR | 3.06 | 2.97 | 2.78 | 3.82 | 3.45 | 3.48 | 5.61 | 4.66 | 4.62 | 3.15 | 2.91 | 4.46 |
| ARIMA | 4.35 | 4.24 | 3.95 | 5.52 | 4.50 | 5.26 | 6.02 | 5.74 | 5.78 | 4.63 | 4.12 | 4.15 |
| SNF | 6.44 | 5.32 | 3.78 | 5.36 | 4.15 | 5.47 | 9.90 | 8.16 | 7.37 | 4.04 | 5.31 | 7.70 |

Table 8 – Displays the Monthly Average MAPE Values for Each Algorithm – Toronto Dataset

##### 4.2.2.1.4 Seasonal Performances

The table below summarizes the MAPE and RMSE values obtained for the Toronto test dataset's average of various seasons. Summer was the horizon that all algorithms had the most difficulty predicting. From most difficult to least difficult, the CNN's next most difficult seasons are Spring, Winter, and Autumn. Winter, Spring, and Autumn are the ANN's next three most difficult seasons. Autumn, Winter, and Spring are the MLR's next three most difficult seasons. Autumn, Spring, and Winter are the ARIMA's next three most difficult seasons. Winter, Autumn, and Spring are the next most difficult seasons in the SNF. The CNN had the lowest MAPE and RMSE values across all four seasons, making it the category winner.



Table 9 – Displays the MAPE and RMSE Values for the Various Seasons in the Toronto Dataset

##### 4.2.2.1.5 Peak Performance

// To be filled

##### 4.2.2.1.6 Discussion of the Comprehensive Analysis

// To be filled

# 5 Conclusion

## 5.1 Our Analysis in Summary

// To be completed

## 5.2 Contributions

Deep learning techniques are considered due to their remarkable performance when applied to a variety of problems. The CNN and LSTM were evaluated for their added value by comparing their performance to that of conventional forecasters. We compared accuracy in terms of both overall and peak detection. This contributes to the maturation of the ongoing debate over the use of deep learning methods in load forecasting that have not been thoroughly tested. We developed algorithms that are more adaptable to changes in external factors such as annual increases in electricity demand or temperature shifts. We developed forecasters that are capable of adapting to complex data relationships without explicit user input. We conducted analysis that was specifically tailored to three distinct datasets. Additionally, because we conducted analysis on publicly available data, this work will be reproducible, serving as a valuable benchmark for future research both within and outside our smart-grid team.

## 5.3 Future Work

Several directions for future work include improving forecaster accuracy, incorporating more exogenous variables, developing more complex algorithms that are either hybrids or improved models of the ones used here, forecasting the width of daily peaks, implementing more deep learning algorithms, and developing separate models to forecast different days and months (e.g. summer, winter). We will discuss a few of the aforementioned points briefly.

Regarding forecasters' accuracy, I would recommend the following. The accuracy of the ARIMA forecaster can be increased by using the SARIMAX model, which incorporates exogenous variables and seasonality. The SARIMAX model can utilize all of the exogenous variables used by the MLR. The performance of the ANN, CNN, and LSTM forecasters can be improved by including additional exogenous variables, such as those used by the MLR forecaster. These variables include the hour of the day, the month of the year, the weekend or holiday indicator, the previous day's maximum, minimum, and average demand, and the previous week's hourly lag. Along with temperature, weather variables such as humidity, dewpoint, and wind direction/speed can be used, depending on the analyst's objectives.

In terms of daily peaks, utilities benefit from knowing when the peaks will occur and how long they will last. As a result, another approach is to determine the width of demand peaks. Additionally, hybrid models incorporating CNNs and LSTMs may also be an option, as some researchers have observed improved performance when the two are combined, as discussed in the deep learning sections. Tao Hong et al. [1] also stated in their review paper that the majority of the best load forecasting algorithms have been discovered to be hybrid models. Additionally, certain researchers have observed improved performance when distinct models are created to forecast specific days, such as weekdays and weekends, or specific months, such as winter and summer [60], [178].

As a result of the preceding paragraphs, we can see that there are numerous possibilities, and that additional research is necessary. These are quite intriguing paths that could be taken, and they can assist utilities in the future in planning for and ensuring a stable supply of electricity to everyone.

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

[3] 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.

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

[5] 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.

[6] 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.

[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] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] E. Ela and B. Kirby, “ERCOT Event on February 26, 2008: Lessons Learned,” 2008, Accessed: Sep. 17, 2021. [Online]. Available: http://www.osti.gov/bridge.

[16] “Freak Blackouts Plunge Korea into Darkness - The Chosun Ilbo (English Edition): Daily News from Korea - national/politics > national,” 2011. http://english.chosun.com/site/data/html\_dir/2011/09/16/2011091600558.html (accessed Sep. 17, 2021).

[17] 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.

[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] 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.

[25] 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.

[26] 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.

[27] 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.

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

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

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

[35] G. E. Hinton, S. Osindero, and Y. W. Teh, “A fast learning algorithm for deep belief nets,” *Neural Comput.*, 2006, doi: 10.1162/neco.2006.18.7.1527.

[36] S. Suresh, “An Analysis of Short-term Load Forecasting on Residential Buildings Using Deep Learning Models,” Virginia Polytechnic Institute and State University, Blacksburg, 2020.

[37] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, “Greedy layer-wise training of deep networks,” 2007, doi: 10.7551/mitpress/7503.003.0024.

[38] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” 2015.

[39] A. Graves, A. R. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” 2013, doi: 10.1109/ICASSP.2013.6638947.

[40] 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.

[41] “What is the difference between Machine Learning and Deep Learning | by Neeraj Kumar | Medium,” 2017. https://medium.com/@Say2neeraj/what-is-the-difference-between-machine-learning-and-deep-learning-5795e4415be9 (accessed Sep. 18, 2021).

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

[43] V. Mnih *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, 2015, doi: 10.1038/nature14236.

[44] 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.

[45] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” 2012.

[46] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” 2016, doi: 10.1109/CVPR.2016.90.

[47] C. L. Liu, F. Yin, Q. F. Wang, and D. H. Wang, “ICDAR 2011 Chinese handwriting recognition competition,” 2011, doi: 10.1109/ICDAR.2011.291.

[48] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, “Deep neural networks segment neuronal membranes in electron microscopy images,” 2012.

[49] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, “Mitosis detection in breast cancer histology images with deep neural networks,” 2013, doi: 10.1007/978-3-642-40763-5\_51.

[50] G. E. Dahl, M. Ranzato, A. R. Mohamed, and G. Hinton, “Phone recognition with the mean-covariance restricted Boltzmann machine,” 2010.

[51] F. Seide, G. Li, and D. Yu, “Conversational speech transcription using Context-Dependent Deep Neural Networks,” 2011, doi: 10.21437/interspeech.2011-169.

[52] O. Abdel-Hamid, A. R. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, “Convolutional neural networks for speech recognition,” *IEEE Trans. Audio, Speech Lang. Process.*, 2014, doi: 10.1109/TASLP.2014.2339736.

[53] L. Deng and J. C. Platt, “Ensemble deep learning for speech recognition,” 2014, doi: 10.21437/interspeech.2014-433.

[54] 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.

[55] H. Hahn, S. Meyer-Nieberg, and S. Pickl, “Electric load forecasting methods: Tools for decision making,” *Eur. J. Oper. Res.*, 2009, doi: 10.1016/j.ejor.2009.01.062.

[56] J. Luo, T. Hong, and M. Yue, “Real-time anomaly detection for very short-term load forecasting,” *J. Mod. Power Syst. Clean Energy*, 2018, doi: 10.1007/s40565-017-0351-7.

[57] K. Liu, “Comparison of very short-term load forecasting techniques,” *IEEE Trans. Power Syst.*, 1996, doi: 10.1109/59.496169.

[58] W. Charyloniuk and M. S. Chen, “Very short-term load forecasting using artificial neural networks,” *IEEE Trans. Power Syst.*, 2000, doi: 10.1109/59.852131.

[59] J. W. Taylor, “An evaluation of methods for very short-term load forecasting using minute-by-minute British data,” *Int. J. Forecast.*, 2008, doi: 10.1016/j.ijforecast.2008.07.007.

[60] 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.

[61] E. Kyriakides and M. Polycarpou, “Short term electric load forecasting: A tutorial,” *Stud. Comput. Intell.*, 2006, doi: 10.1007/978-3-540-36122-0\_16.

[62] 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.

[63] 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.

[64] S. Dwijayanti, “Short Term Load Forecasting Using a Neural Network Based Time Series Approach,” Oklahoma State University, 2013.

[65] G. J. Tsekouras, N. D. Hatziargyriou, and E. N. Dialynas, “An optimized adaptive neural network for annual midterm energy forecasting,” *IEEE Trans. Power Syst.*, 2006, doi: 10.1109/TPWRS.2005.860926.

[66] E. Doveh, P. Feigin, D. Greig, and L. Hyams, “Experience with FNN models for medium term power demand predictions,” *IEEE Trans. Power Syst.*, 1999, doi: 10.1109/59.761878.

[67] J. Reneses, E. Centeno, and J. Barquín, “Coordination between medium-term generation planning and short-term operation in electricity markets,” *IEEE Trans. Power Syst.*, 2006, doi: 10.1109/TPWRS.2005.857851.

[68] M. S. Kandil, S. M. El-Debeiky, and N. E. Hasanien, “Long-term load forecasting for fast developing utility using a knowledge-based expert system,” *IEEE Trans. Power Syst.*, 2002, doi: 10.1109/TPWRS.2002.1007923.

[69] R. Houimli, M. Zmami, and O. Ben-Salha, “Short-term electric load forecasting in Tunisia using artificial neural networks,” *Energy Syst.*, 2020, doi: 10.1007/s12667-019-00324-4.

[70] D. C. Park, R. J. Marks, L. E. Atlas, and M. J. Damborg, “Electric load forecasting using an artificial neural network - Power Systems, IEEE Transactions on,” *IEEE Transadions Power Syst.*, 1991.

[71] A. G. Bakirtzis, V. Petridis, S. J. Klartzis, M. C. Alexiadis, and A. H. Maissis, “A neural network short term load forecasting model for the greek power system,” *IEEE Trans. Power Syst.*, 1996, doi: 10.1109/59.496166.

[72] L. C. P. Velasco, C. R. Villezas, P. N. C. Palahang, and J. A. A. Dagaang, “Next day electric load forecasting using Artificial Neural Networks,” 2016, doi: 10.1109/HNICEM.2015.7393166.

[73] G. Gross and F. D. Galiana, “SHORT-TERM LOAD FORECASTING.,” *Proc. IEEE*, 1987, doi: 10.1109/PROC.1987.13927.

[74] A. Muñoz, E. F. Sánchez-Úbeda, A. Cruz, and J. Marín, “Short-term Forecasting in Power Systems: A Guided Tour,” 2010.

[75] E. J. Wicksteed, “Short term electric load forecasting for British Columbia, Canada: an exploration of the use of numerical weather prediction data as a predictor in an artificial neural network,” University of British Columbia, 2021.

[76] D. Srinivasan and M. A. Lee, “Survey of hybrid fuzzy neural approaches to electric load forecasting,” 1995, doi: 10.1109/icsmc.1995.538416.

[77] C. N. Lu, H. T. Wu, and S. Vemuri, “Neural Network Based Short Term Load Forecasting,” *IEEE Trans. Power Syst.*, 1993, doi: 10.1109/59.221223.

[78] T. Hong, “Short Term Electric Load Forecasting dissertation,” *3442639*, 2010.

[79] J. Foster, “Electric load forecasting with increased embedded renewable generation,” Queen’s University, 2020.

[80] T. Hong and M. Shahidehpour, “Load Forecasting Case Study,” *U.S. Dep. Energy*, 2015.

[81] E. Taylor, “Short-term Electrical Load Forecasting for an Institutional/Industrial Power System Using an Artificial Neural Network,” The University of Tennessee, Knoxville, 2013.

[82] M. JANICKI, “Methods of weather variables introduction into short-term electric load forecasting models - a review,” *PRZEGLĄD ELEKTROTECHNICZNY*, 2017, doi: 10.15199/48.2017.04.18.

[83] L. Friedrich and A. Afshari, “Short-term Forecasting of the Abu Dhabi Electricity Load Using Multiple Weather Variables,” 2015, doi: 10.1016/j.egypro.2015.07.616.

[84] J. W. Taylor and R. Buizza, “Neural network load forecasting with weather ensemble predictions,” *IEEE Trans. Power Syst.*, 2002, doi: 10.1109/TPWRS.2002.800906.

[85] 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.

[86] M. Sobhani, A. Campbell, S. Sangamwar, C. Li, and T. Hong, “Combining weather stations for electric load forecasting,” *Energies*, 2019, doi: 10.3390/en12081510.

[87] T. Hong, P. Wang, and L. White, “Weather station selection for electric load forecasting,” *Int. J. Forecast.*, 2015, doi: 10.1016/j.ijforecast.2014.07.001.

[88] S. N. Fallah, M. Ganjkhani, S. Shamshirband, and K. wing Chau, “Computational intelligence on short-term load forecasting: A methodological overview,” *Energies*. 2019, doi: 10.3390/en12030393.

[89] S. Moreno-Carbonell, E. F. Sánchez-Úbeda, and A. Muñoz, “Rethinking weather station selection for electric load forecasting using genetic algorithms,” *Int. J. Forecast.*, 2020, doi: 10.1016/j.ijforecast.2019.08.008.

[90] S. Fan, K. Methaprayoon, and W. J. Lee, “Multi-area load forecasting for system with large geographical area,” 2008, doi: 10.1109/ICPS.2008.4606287.

[91] E. L. Taylor, “Short-term Electrical Load Forecasting for an Institutional/ Industrial Power System Using an Artificial Neural Network,” University of Tennessee, 2013.

[92] 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.

[93] 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.

[94] 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.

[95] 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.

[96] 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.

[97] 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.

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

[99] 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.

[100] 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.

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

[102] 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.

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

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

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

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

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

[108] 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.

[109] 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.

[110] 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.

[111] 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.

[112] A. Shadkam, “Using SARIMAX to forecast electricity demand and consumption in university buildings,” The University of British Columbia, 2020.

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

[114] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *Bull. Math. Biophys.*, 1943, doi: 10.1007/BF02478259.

[115] D. O. Hebb, “The first stage of perception: growth of the assembly,” *Organ. Behav.*, 1949, doi: 10.1016/0301-0082(84)90021-2.

[116] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain,” *Psychol. Rev.*, 1958, doi: 10.1037/h0042519.

[117] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, 1986, doi: 10.1038/323533a0.

[118] 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.

[119] 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.

[120] 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.

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

[122] “Activation Function - AI Wiki,” 2019. https://docs.paperspace.com/machine-learning/wiki/activation-function (accessed Sep. 18, 2021).

[123] “Artificial Neural Network (ANN) with Practical Implementation | by Amir Ali | Wavy AI Research Foundation | Medium,” 2019. https://medium.com/machine-learning-researcher/artificial-neural-network-ann-4481fa33d85a (accessed Sep. 10, 2021).

[124] 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.

[125] 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.

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

[127] 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.

[128] 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.

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

[130] 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.

[131] 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.

[132] “Recursive least squares filter - Wikipedia.” https://en.wikipedia.org/wiki/Recursive\_least\_squares\_filter (accessed Oct. 08, 2021).

[133] 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.

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

[135] “Independent Electricity System Operator - Hourly Zonal Demand Report.” http://reports.ieso.ca/public/DemandZonal/ (accessed Jun. 05, 2021).

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

[137] “What is Peak Load? | Aquicore.” https://aquicore.com/blog/what-is-peak-load/ (accessed Oct. 07, 2021).

[138] “Peak Load & Base Electricity - Understand Differences - EnergyWatch.” https://energywatch-inc.com/peak-load-base-load-electricity/ (accessed Oct. 07, 2021).

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

[140] 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.

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

[142] 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.

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

[144] 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.

[145] B. Farsi, “On Short-Term Load Forecasting Using Machine Learning Techniques,” Concordia University, 2020.

[146] 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.

[147] 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.

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

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

[150] P. P. Phyo, “Deep Learning for Short-term Electricity Load Forecasting,” Sirindhorn International Institute of Technology, 2018.

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

[152] S. Bouktif, A. Fiaz, A. Ouni, and M. A. Serhani, “Optimal deep learning LSTM model for electric load forecasting using feature selection and genetic algorithm: Comparison with machine learning approaches,” *Energies*, 2018, doi: 10.3390/en11071636.

[153] 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.

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

[155] 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.

[156] 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.

[157] 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.

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

[159] 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.

[160] 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.

[161] 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.

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

[163] 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.

[164] “Convolutional neural networks for time series forecasting | Python for Finance Cookbook,” 2020. https://subscription.packtpub.com/book/data/9781789618518/10/ch10lvl1sec63/convolutional-neural-networks-for-time-series-forecasting (accessed Aug. 30, 2021).

[165] “ReLU : Not a Differentiable Function: Why used in Gradient Based Optimization? and Other Generalizations of ReLU. | by Kanchan Sarkar | Medium,” 2018. https://medium.com/@kanchansarkar/relu-not-a-differentiable-function-why-used-in-gradient-based-optimization-7fef3a4cecec (accessed Sep. 17, 2021).

[166] “What is max pooling in convolutional neural networks? - Quora,” 2017. https://www.quora.com/What-is-max-pooling-in-convolutional-neural-networks (accessed Sep. 17, 2021).

[167] D. C. Wu, B. Bahrami Asl, A. Razban, and J. Chen, “Air compressor load forecasting using artificial neural network,” *Expert Syst. Appl.*, 2021, doi: 10.1016/j.eswa.2020.114209.

[168] S. Panigrahi, Y. Karali, and H. S. Behera, “Normalize Time Series and Forecast using Evolutionary Neural Network,” *Int. J. Comput. Appl.*, 2013.

[169] L. Kuan *et al.*, “Short-term electricity load forecasting method based on multilayered self-normalizing GRU network,” 2017, doi: 10.1109/EI2.2017.8245330.

[170] L. Li, K. Ota, and M. Dong, “Everything is image: CNN-based short-term electrical load forecasting for smart grid,” 2017, doi: 10.1109/ISPAN-FCST-ISCC.2017.78.

[171] M. Dong and L. Grumbach, “A Hybrid Distribution Feeder Long-Term Load Forecasting Method Based on Sequence Prediction,” *IEEE Trans. Smart Grid*, 2020, doi: 10.1109/TSG.2019.2924183.

[172] L. Yin and J. Xie, “Multi-temporal-spatial-scale temporal convolution network for short-term load forecasting of power systems,” *Appl. Energy*, 2021, doi: 10.1016/j.apenergy.2020.116328.

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

[174] I. K. M. Jais, A. R. Ismail, and S. Q. Nisa, “Adam Optimization Algorithm for Wide and Deep Neural Network,” *Knowl. Eng. Data Sci.*, 2019, doi: 10.17977/um018v2i12019p41-46.

[175] 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.

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

[177] 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.

[178] M. Barman and N. B. Dev Choudhury, “Season specific approach for short-term load forecasting based on hybrid FA-SVM and similarity concept,” *Energy*, 2019, doi: 10.1016/j.energy.2019.03.010.

Appendix

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

// Overall Error Metrics for both base load and peak detection.

// Box plots of the error distribution

// Plots of the algorithms performance on the hourly, daily, monthly, on different seasons. // Plots of all the daily peaks in the datasets and the algorithms prediction of the peaks

// You can add your source code here as well.

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